DES and AES

Chun-Jen Chung

- Q: Why does the ciphers introduced so far not secure?
- A: because of **language characteristics**

- Q: Any ideas to improve them (you already know the answer)?

- A: Use both substitution and transposition

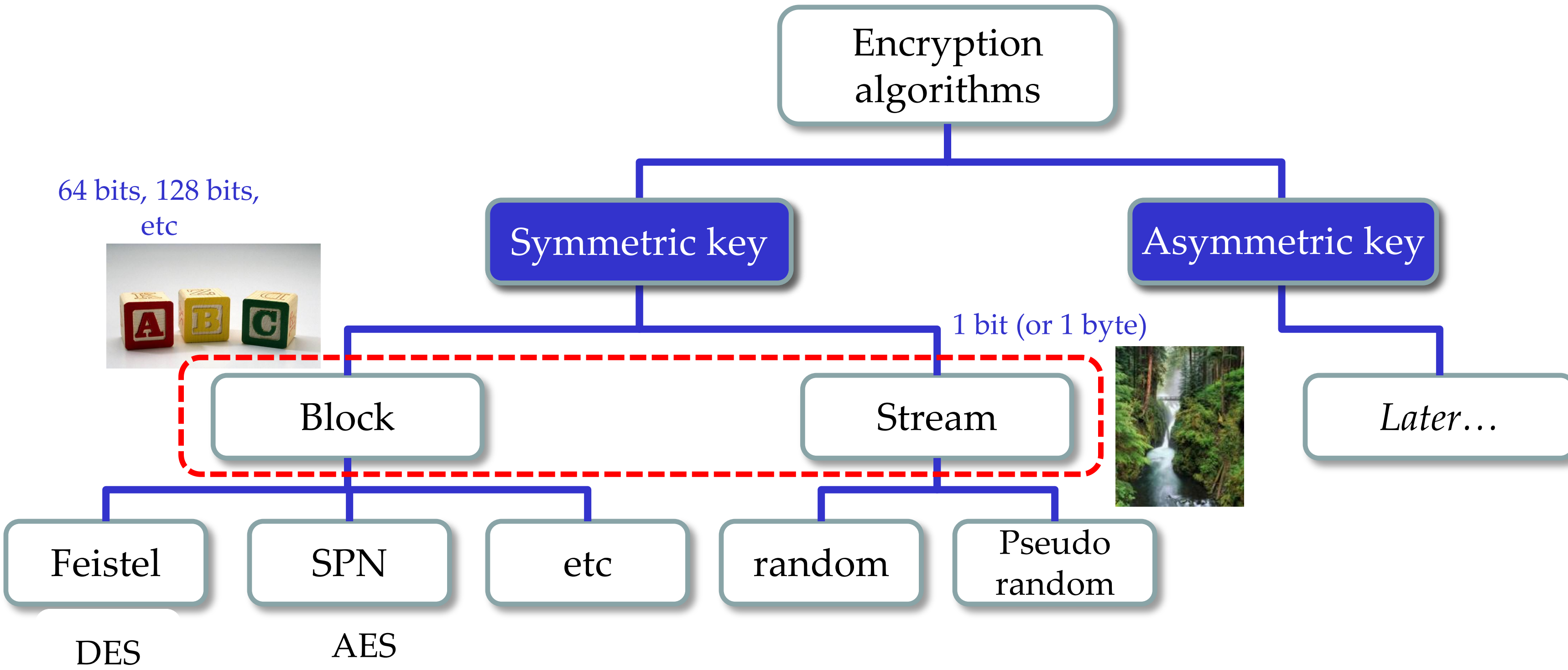
From classical to modern ciphers

- Consider using several ciphers in succession to make harder, but:
  - Two substitutions make a more complex substitution
  - Two transpositions make more complex transposition
  - But a substitution followed by a transposition makes a new much harder cipher
- Q: What is this type of ciphers called?
- A: **product** ciphers

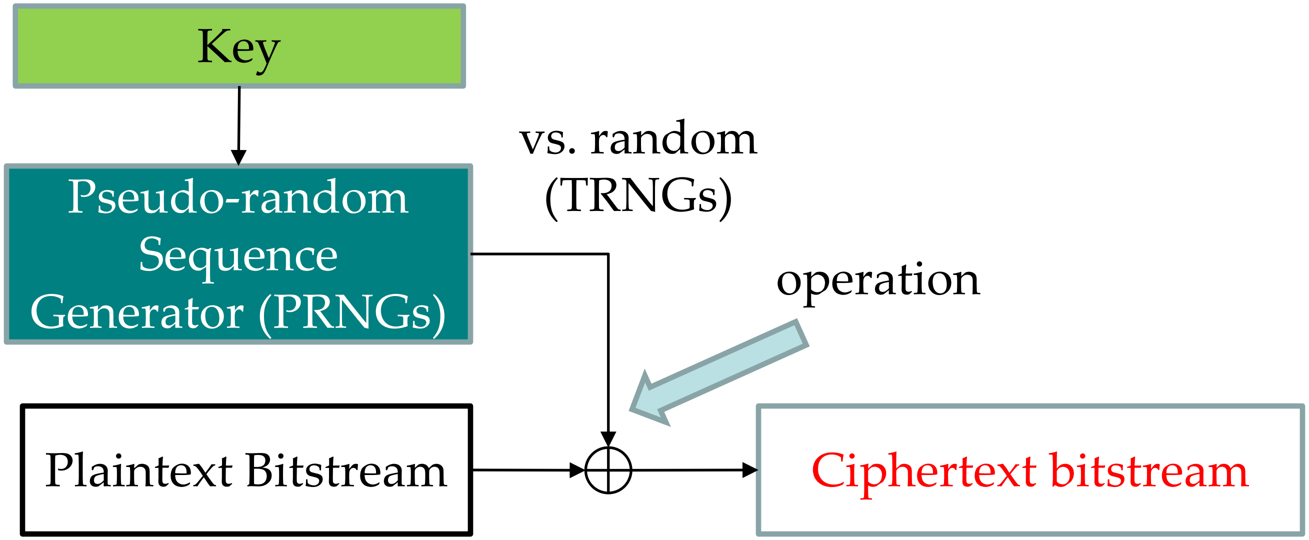
- This is ***bridge*** from classical to modern ciphers

- Q: What is most well-known and widely used morden cipher(s)?
- A: DES, AES,…

Classification of encryption algorithms



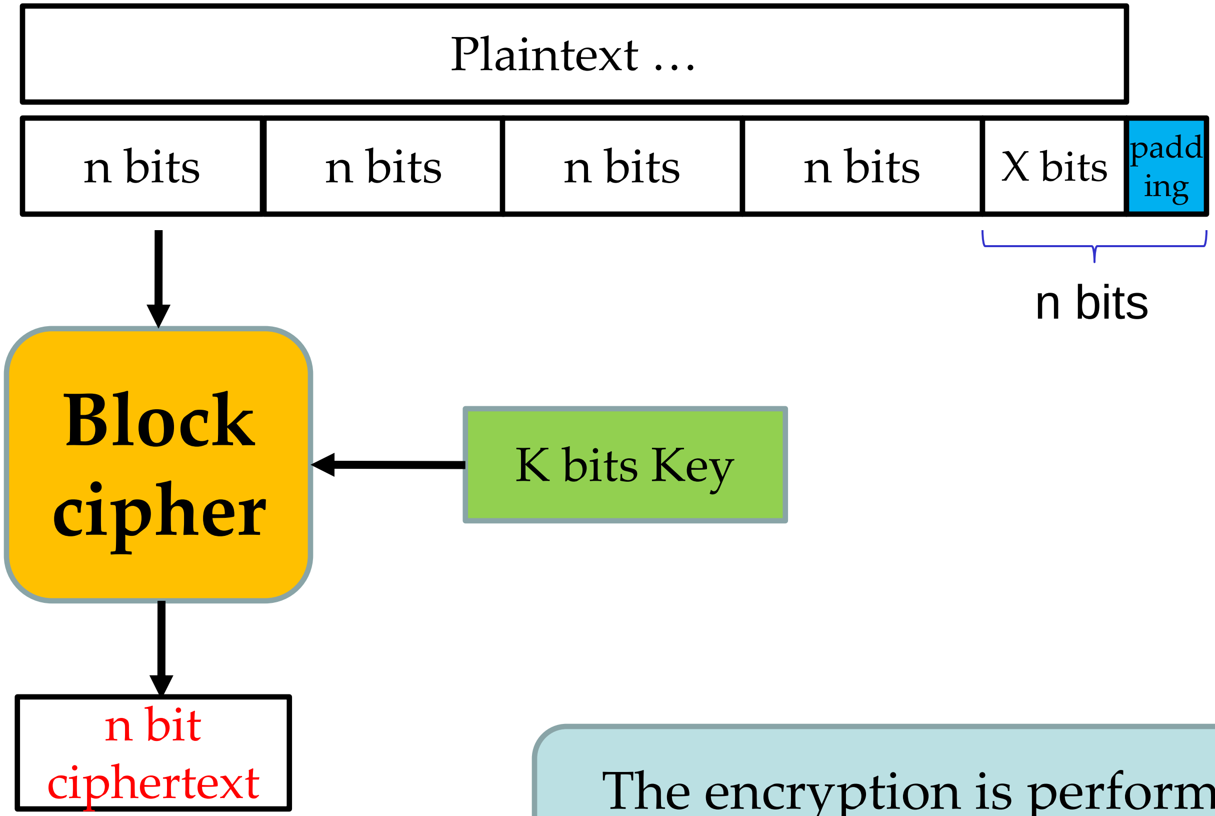
Stream cipher

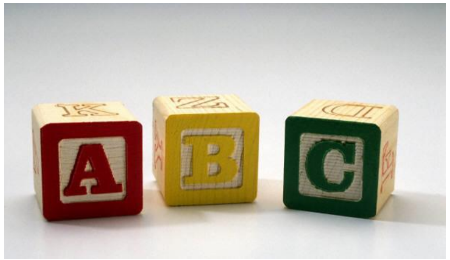


Plaintext bitstream	1 1 1 1 1 1 1 0 0 0 0 0 0 0 …
Pesudo-random stream	1 0 0 1 1 0 1 0 1 1 0 1 0 0 …
Ciphertext stream	0 1 1 0 0 1 0 1 1 1 0 1 0 0 …

Q: Caesar is a stream cipher?

Block cipher





The encryption is performed using one of the operation modes, we will visit it later.

Common block sizes:
n = 64, 128, 256 bits

Common key sizes:
k = 40, 56, 64, 80, 128,168, 192, 256 bits

Stream cipher vs. Block cipher

	Stream cipher	Block cipher
Pros.	• **Speed of transformation:** Because each symbol is encrypted without regard for any other plaintext symbols, each symbol can be encrypted as soon as it is read. • **Low error propagation:** Because each symbol is separately encoded	• **High diffusion:** Information from the plaintext is diffused into several ciphertext symbols. • **Immunity to insertion of symbols:** Because blocks of symbols are enciphered, it is impossible to insert a single symbol into one block. The length of the block would then be incorrect
Cons.	• Low diffusion • Susceptibility to malicious insertions and modifications	• Slowness of encryption (c.f. faster than public key) • Error propagation

DES (Data Encryption Standard)

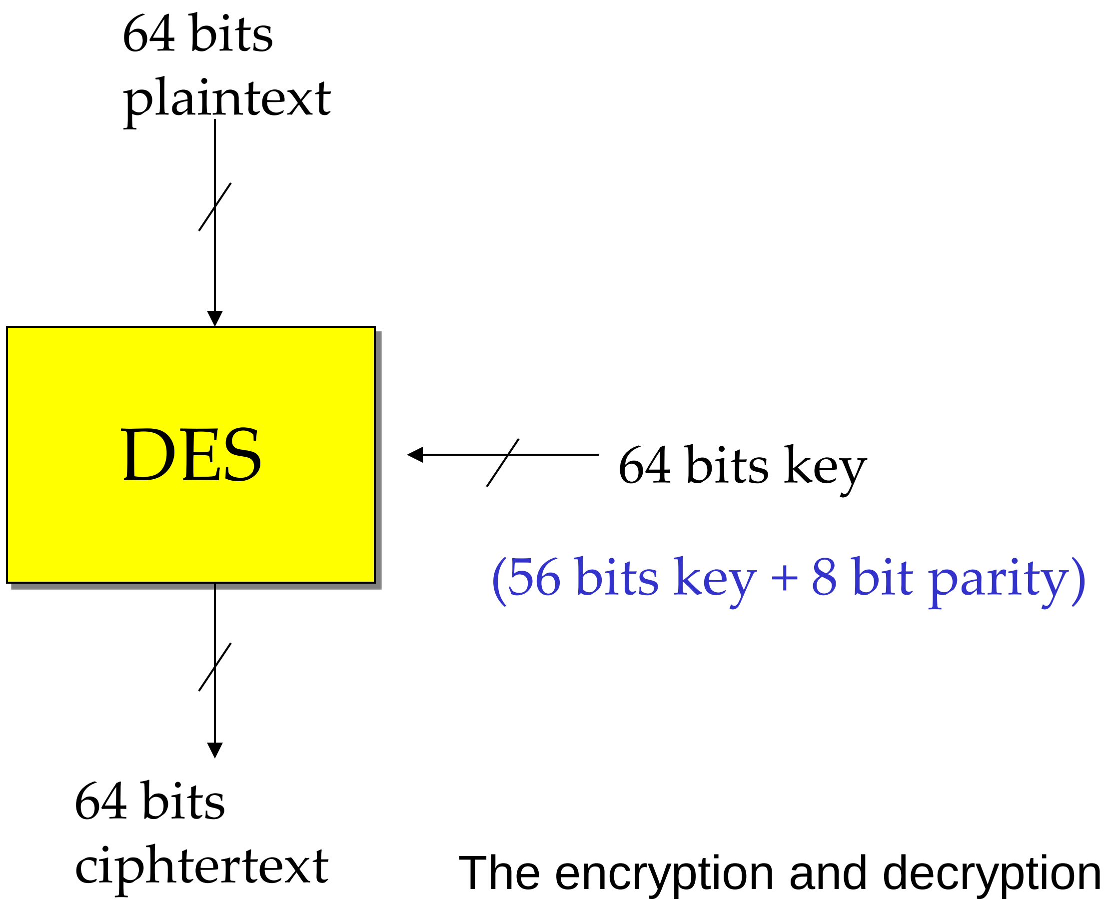
Block cipher: DES, AES

Symmetric key
- Block
  - Feistel — DES
  - SPN — AES
  - etc
- Stream
  - random
  - Pseudo random

DES: Data Encryption Standard (1970s)
or
DEA: Data Encryption Algorithm

AES: Advanced Encryption Standard (2001)

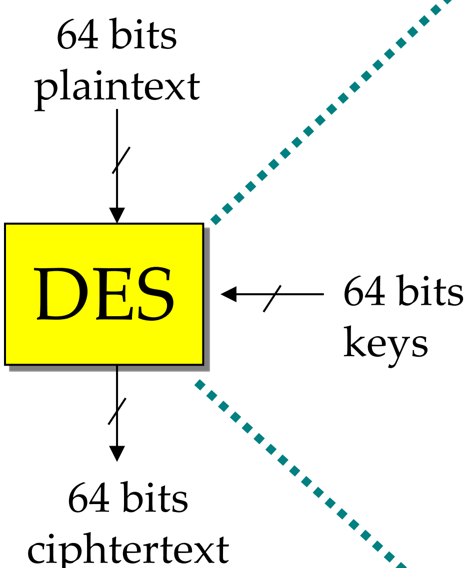
DES Structure

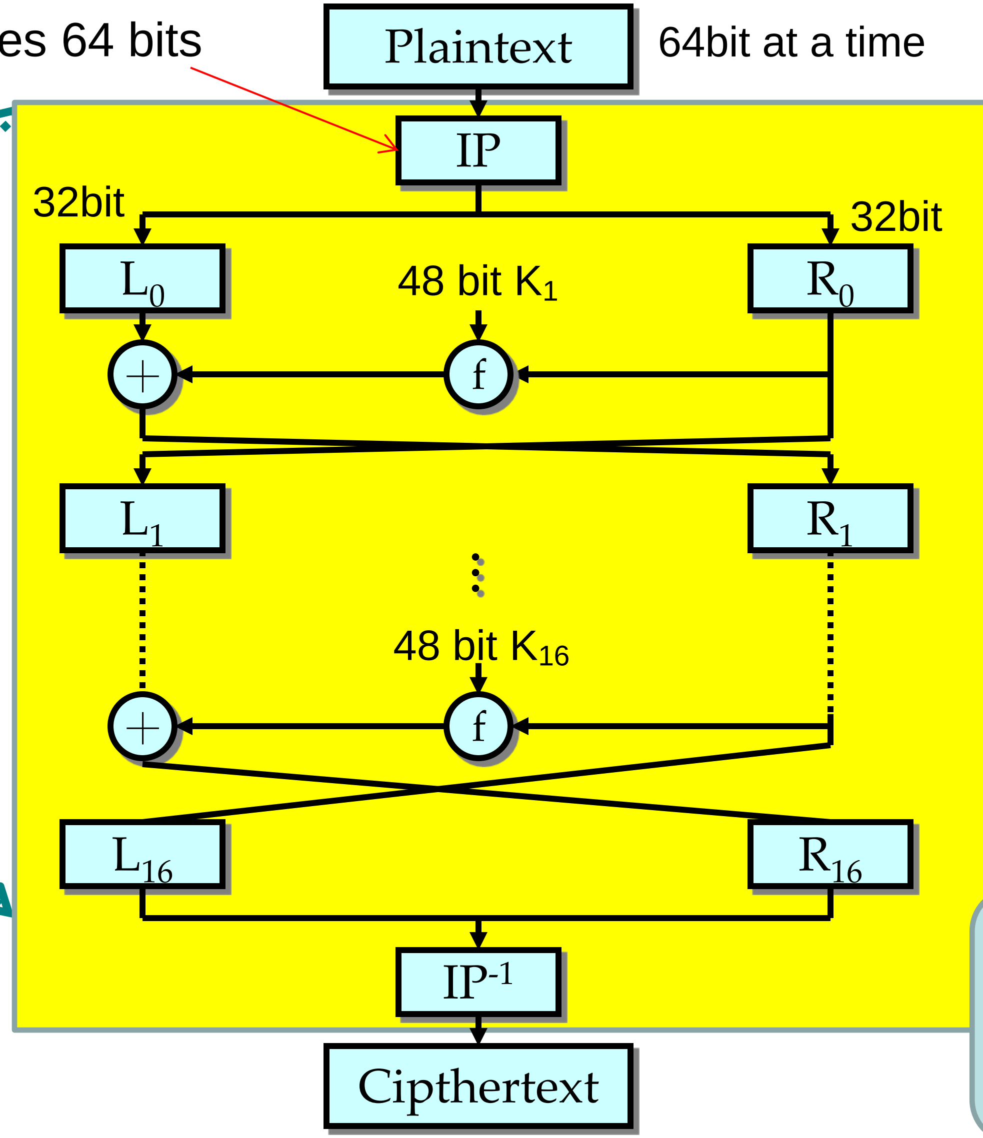


The encryption and decryption operations are very similar, even identical in some cases, requiring only a reversal of the key schedule.

DES Structure

Initial permutation rearranges 64 bits (no cryptographic effect)





$IP(M) = L_0R_0$

$L_i = R_{i-1}$

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

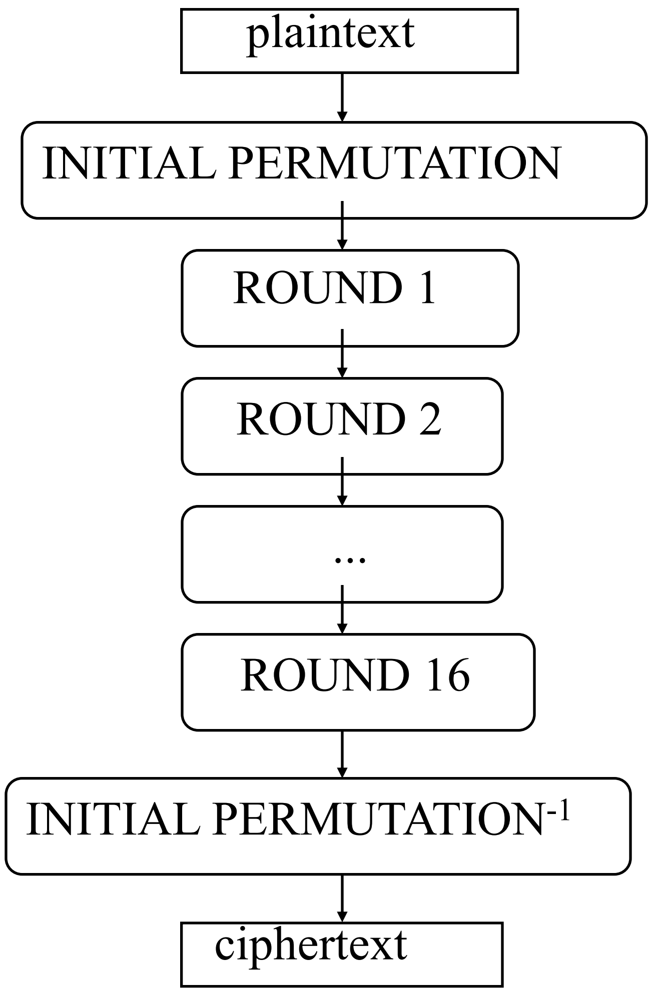
$C = IP^{-1}(R_{16}L_{16})$

$1 <= i <=15$

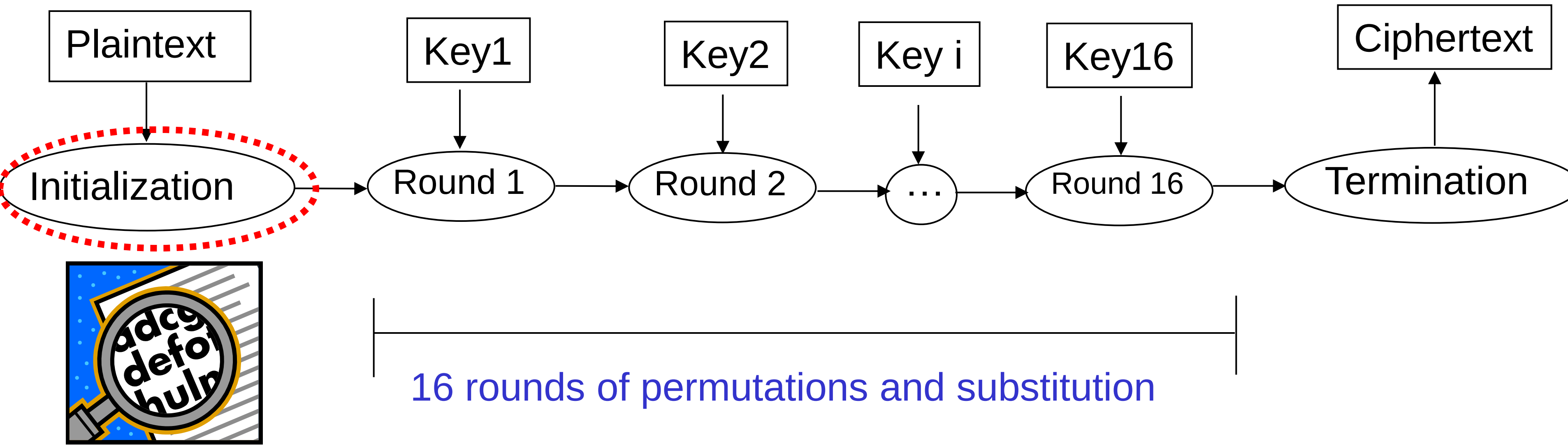
16
rounds of
permutations
and
substitution

Overview of DES

- Block cipher: 64 bits at a time
- Initial permutation rearranges 64 bits (no cryptographic effect)
- Encoding is in 16 rounds



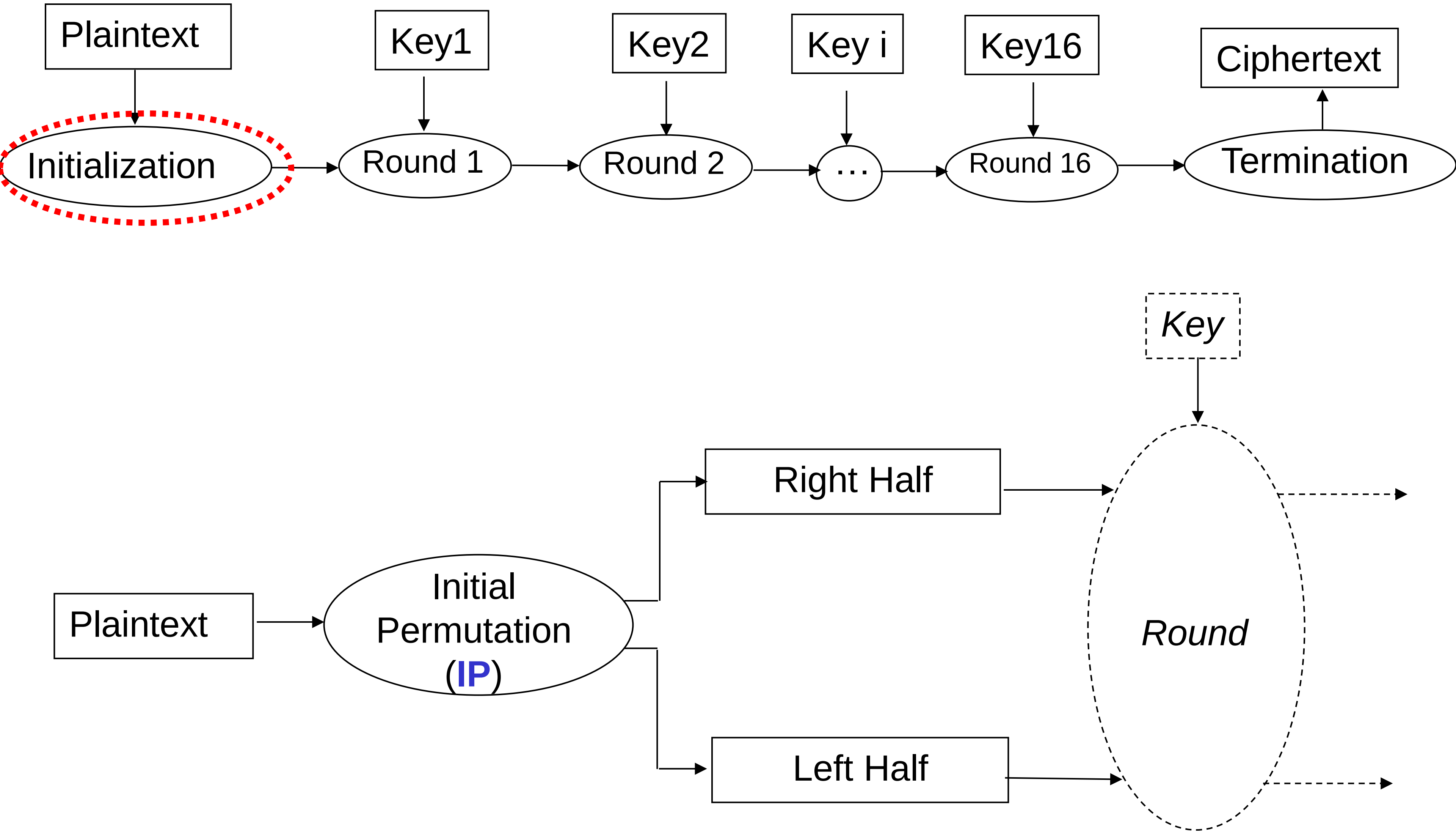
Overview of DES



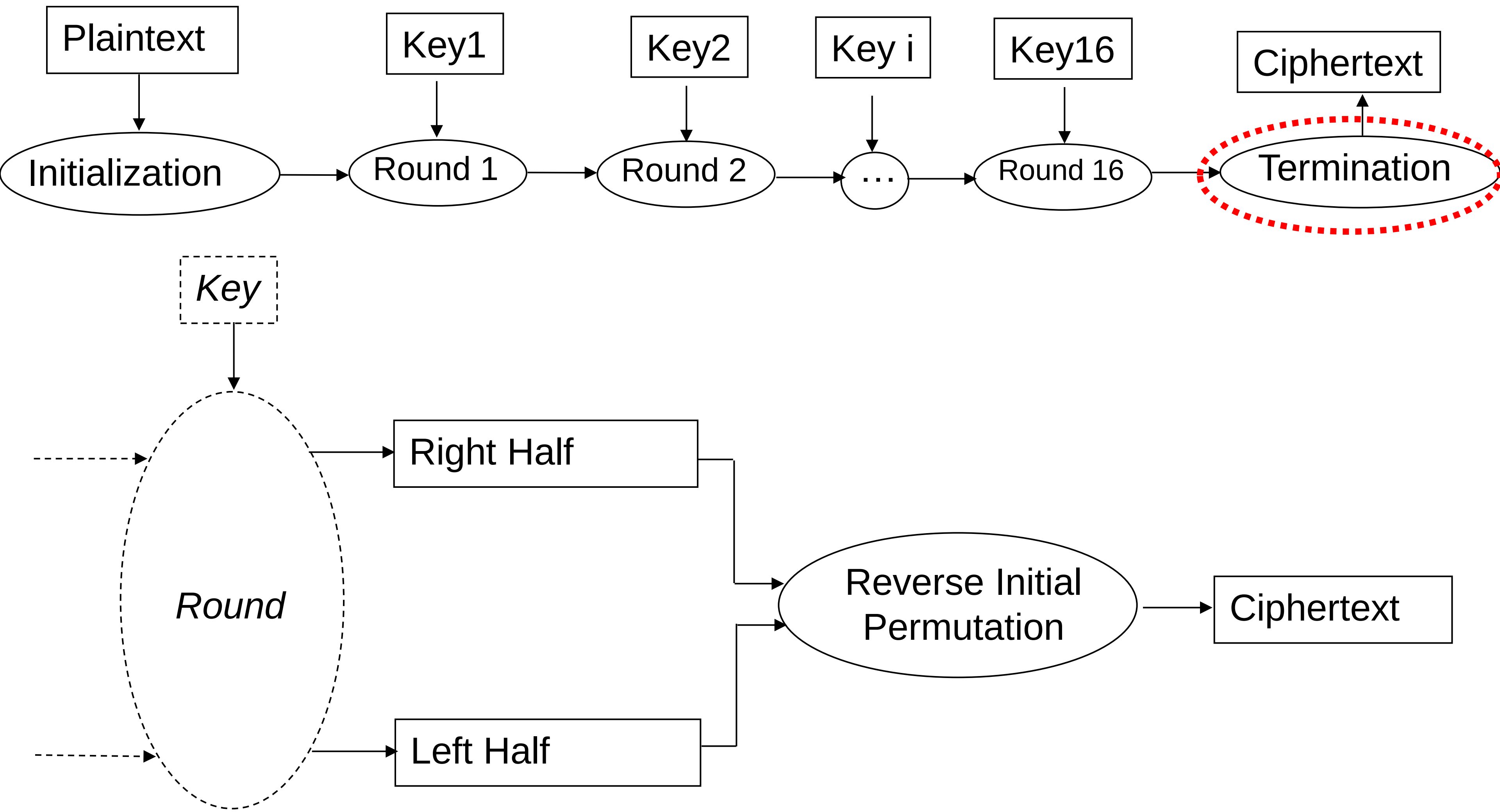
DES is a 64-bit block cipher. Both the plaintext and ciphertext are 64 bits wide.

The key is 64-bits wide, but every eighth bit is a parity bit yielding a 54-bit key.

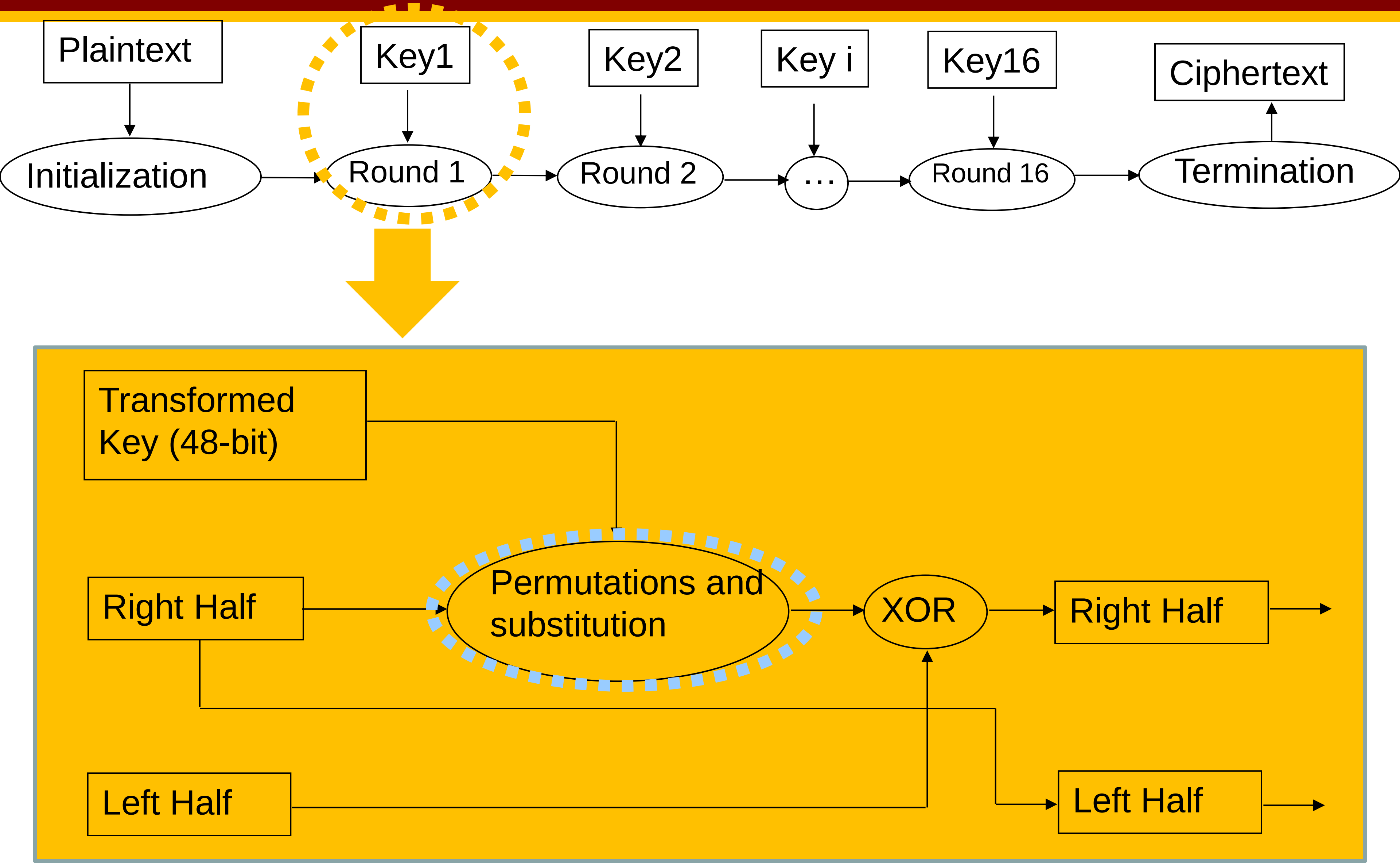
Initialization

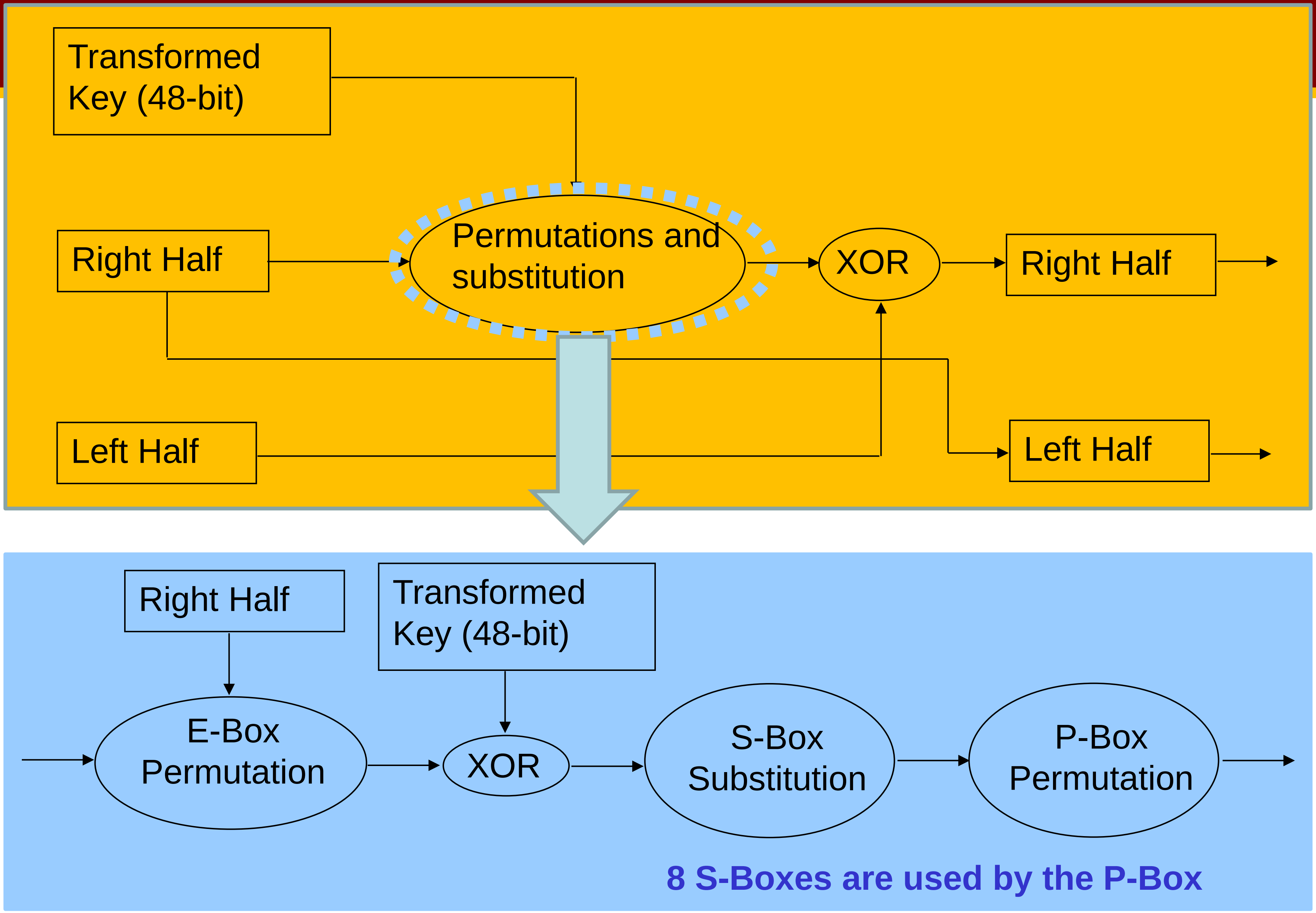


Termination



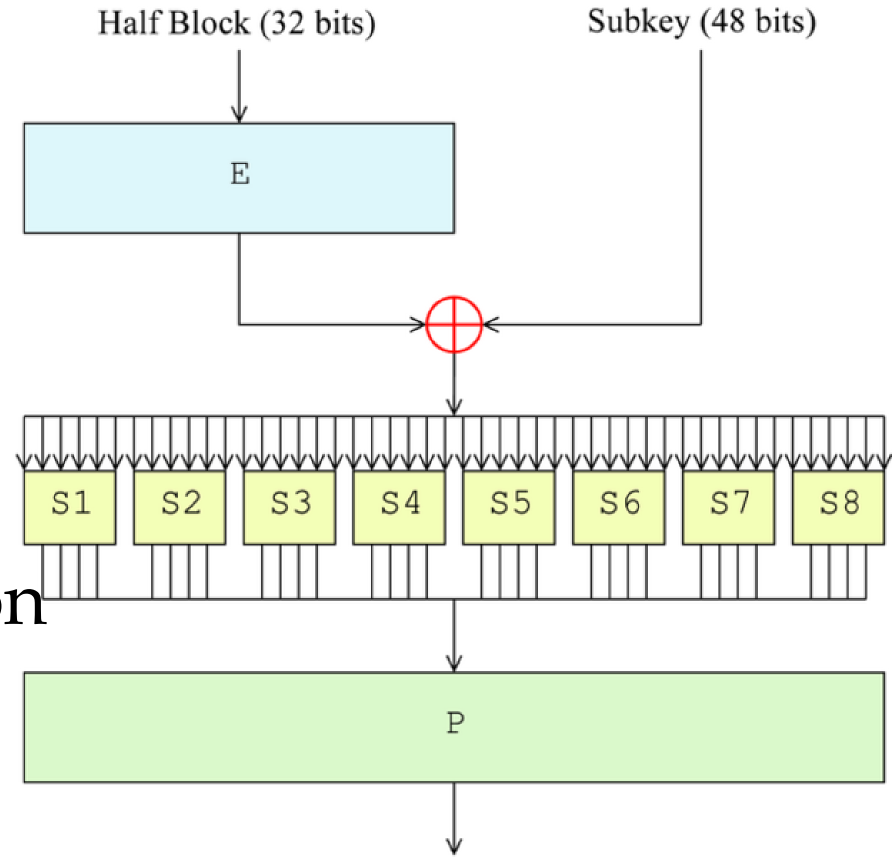
A round


Plaintext
Key1
Key2
Key i
Key16
Ciphertext
Initialization
Round 1
Round 2
…
Round 16
Termination
Transformed Key (48-bit)
Right Half
Permutations and substitution
XOR
Right Half
Left Half
Left Half


Transformed Key (48-bit)
Right Half
Permutations and substitution
XOR
Right Half
Left Half
Left Half
Right Half
Transformed Key (48-bit)
E-Box Permutation
XOR
S-Box Substitution
P-Box Permutation
8 S-Boxes are used by the P-Box

Feistel Function (f function)

- E-box
  - Expansion permutation 32-bits → 48-bits
- Key mixing
  - XOR with 48-bits subkey
- S-boxes (substitution)
  - Non-linear transformation
- P-box (permutation)
  - Rearrange output with fixed permutation function

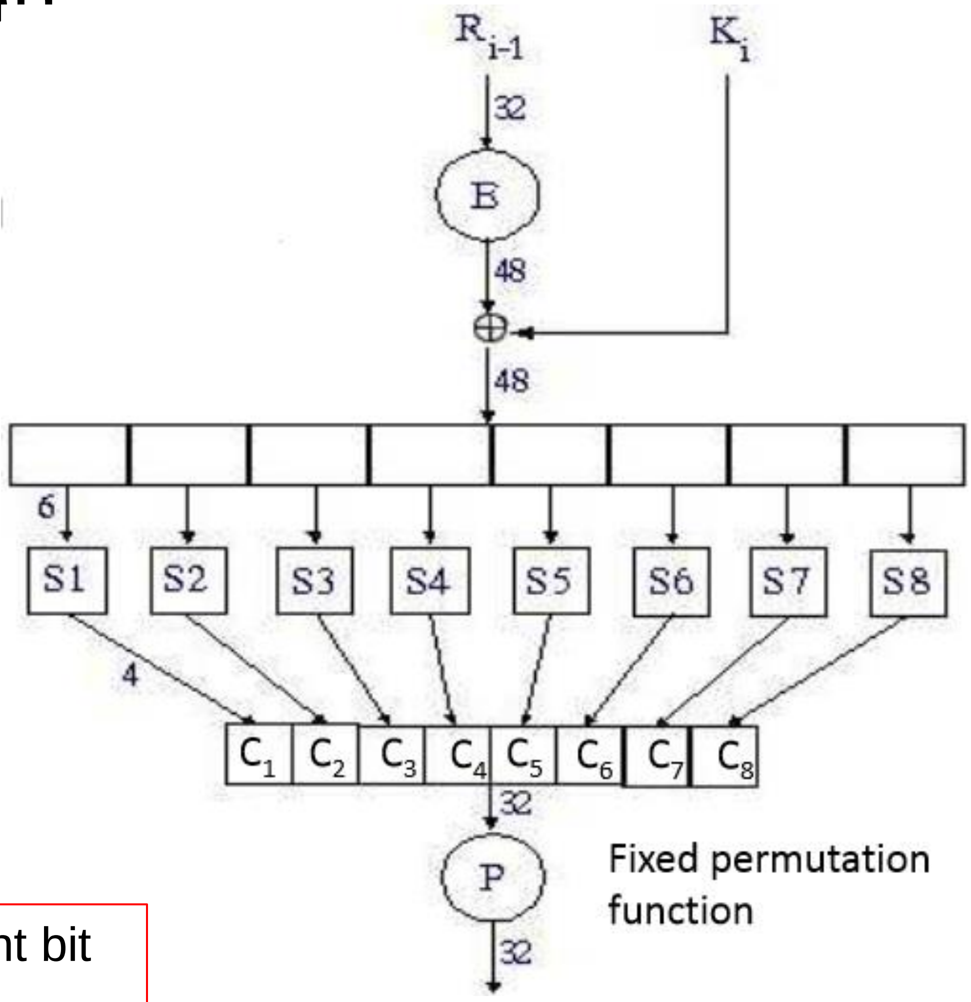


E-box

- Expansion function
  - 32 bits → 48 bits

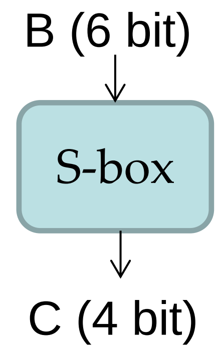
S_1	32	1	2	3	4	5
S_2	4	5	6	7	8	9
S_3	8	9	10	11	12	13
S_4	12	13	14	15	16	17
S_5	16	17	18	19	20	21
S_6	20	21	22	23	24	25
S_7	24	25	26	27	28	29
S_8	28	29	30	31	32	1

Add a copy of the immediately adjacent bit
16 bits appear twice, in the expansion



S-box

- Only *non-linear transformation* in DES, the core of security of DES.
- $B = b_1b_2b_3b_4b_5b_6$
  - b_1b_6 → row (2^2: 0~3)
  - $b_2b_3b_4b_5$ → column (2^4: 0~15)



- C = S(row, column)
- E.g.

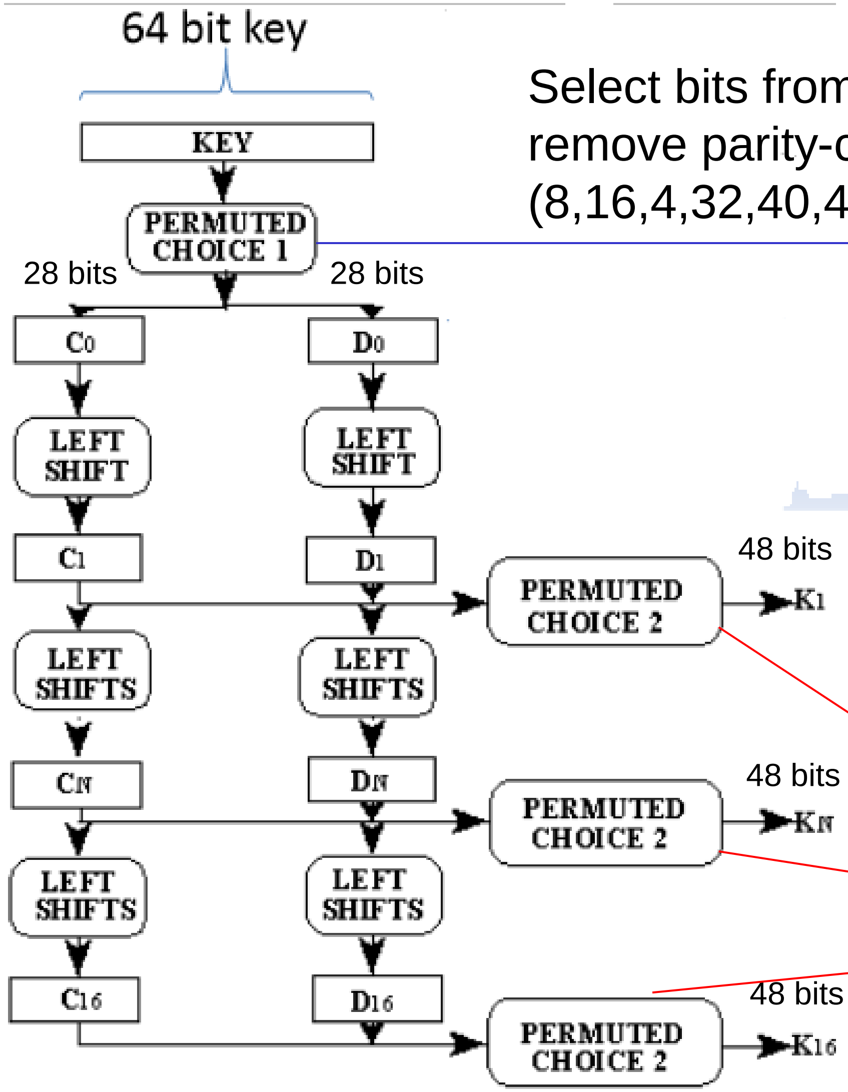
  B = 101111

  C = S(3,7) = 7

  = 0111
- B = 011011, C=?

S_1		1	2	3	...			7								15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

DES Key Generation



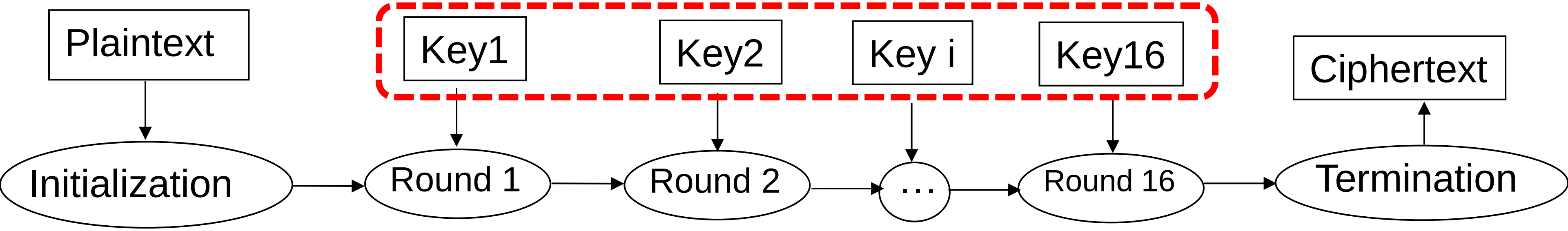
Select bits from PC1, remove parity-check bits (8,16,4,32,40,48,56,64)

Left						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
Right						
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

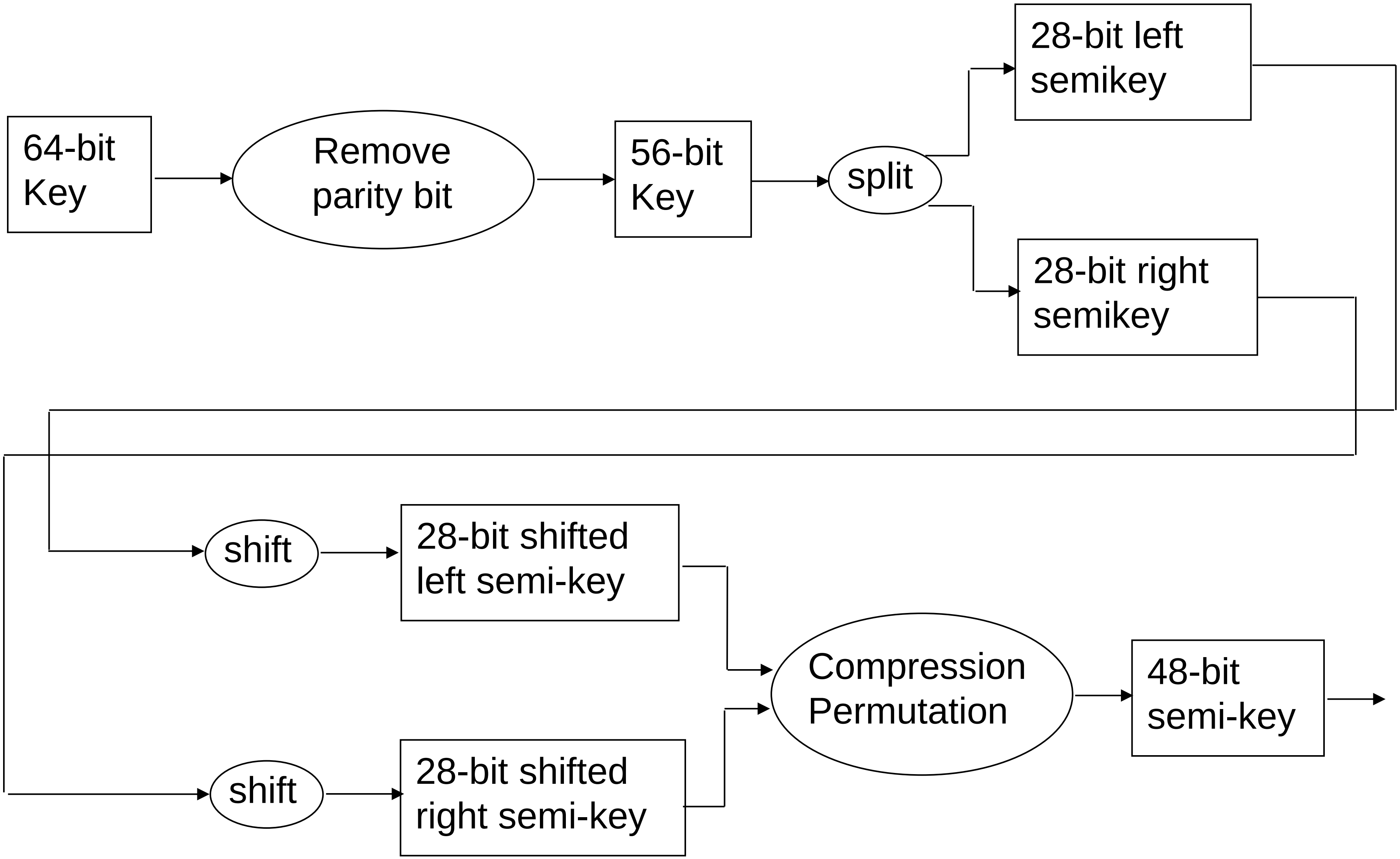
PC-2 selects the 48-bit subkey for each round from the 56-bit key-schedule state

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

Key transform



Key transform



Study simple DES

- 8 bits block with a 10 bits key
- The encryption process is :
  - Initial Permutation
  - Function f_{k1}
  - Switch of the key halves
  - Function f_{k2}
  - Final Permutation (inverse of initial permutation)

DES: security concern

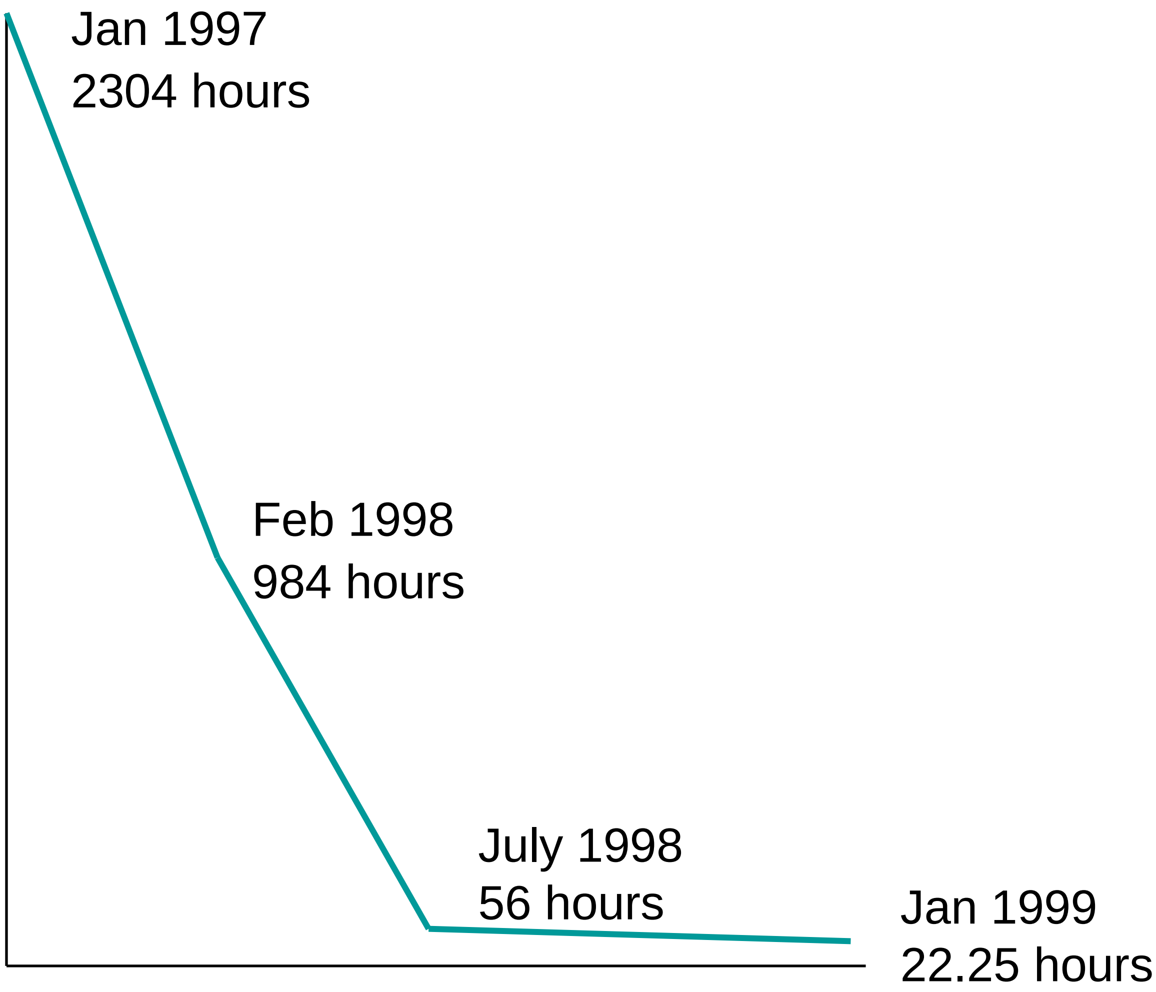
- 56 bit key is too short
  - Can be broken on average in 2^55 ≈3.6*10^16 trials
  - Moore's law: speed of processor doubles per 1.5 yr
  - 1997: 3500 machines broke DES in about 4 months
  - 1998: 1M dollar machine broke DES in about 4 days
  - …

DES: security concern

- Weak Keys
  - 56 bit key is too short
    - Can be broken on average in $2^{56} \approx 7.21*10^{16}$ trials
    - Moore's law: speed of processor doubles per 1.5 yr
  - Keys make the same sub-key in more then 1 round.
  - DES has 4 week keys
    - 01010101 01010101
    - FEFEFEFE FEFEFEFE
    - E0E0E0E0 F1F1F1F1
    - 1F1F1F1F 0E0E0E0E
    - Using weak keys, the outcome of the PC1 to sub-keys being either all 0, all 1, or alternating 0-1 patterns.
    - Another problem: $E_{weak\text{-}key}(E_{weak\text{-}key}(x)) = x$.

DES: security concern

- Cracking the 56-bit DES Encryption Algorithm



Multiple Encryption & DES

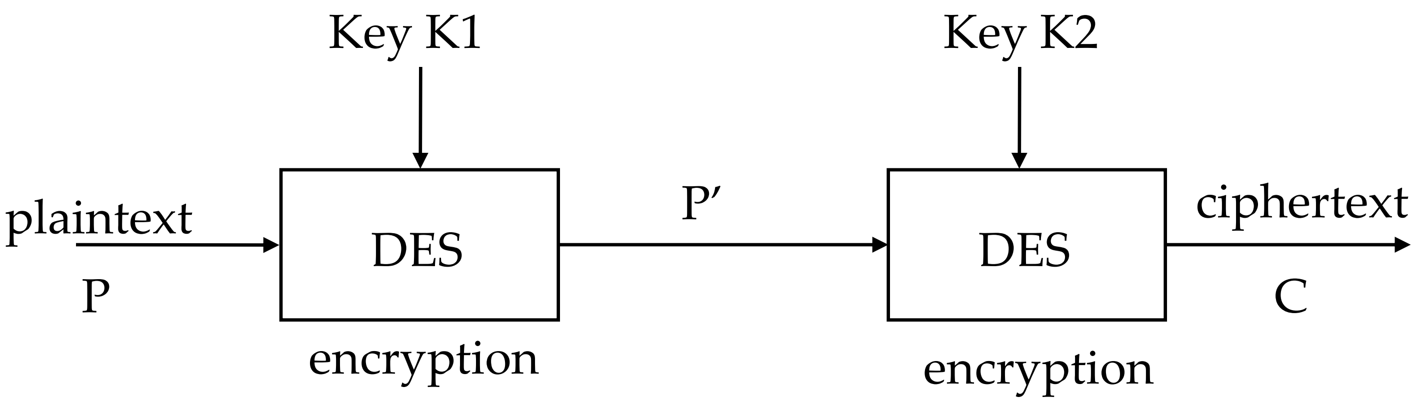
- DES is not secure enough.
- The once large key space, 2^{56}, is now too small.
- In 2001, NIST published the Advanced Encryption Standard (AES) as an alternative.
- But users in commerce and finance are not ready to give up on DES.
- Solution: to use multiple DES with multiple keys

  Q: how many times can we use?

  A: 2, 3, …

Double-DES

- 2-DES



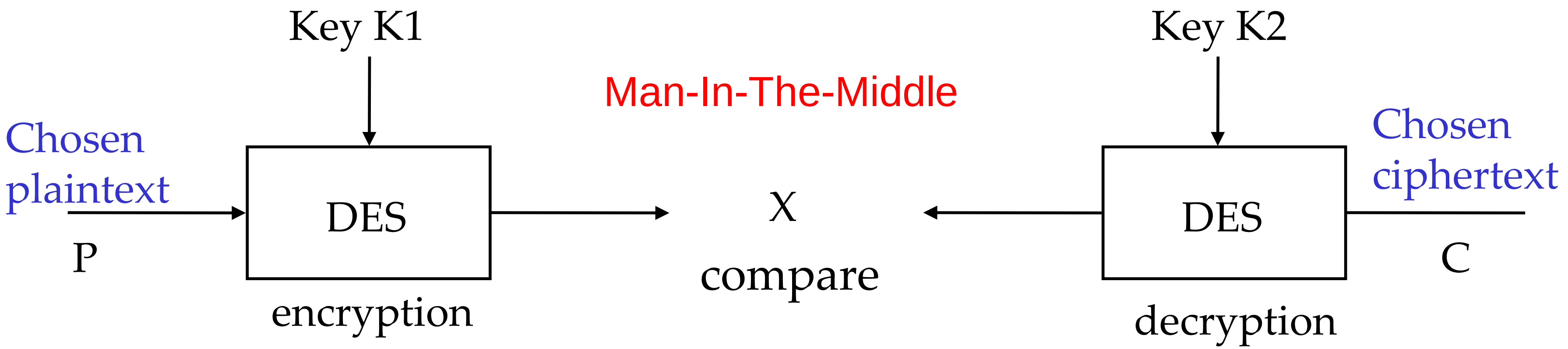
$P' = E_{K1}(P)$ $C = E_{K2}(P')$

$P = D_{K1}(C')$ $C' = D_{K2}(C)$

Any problem for this scheme?

Attack Double-DES

- 2-DES: $C = E_{K2}(E_{K1}(P))$, $P = D_{K1}(D_{K2}(C))$
- So, $X = E_{K1}(P) = D_{K2}(C)$



(1) try all 2^{56} possible keys for K1

(2) try all 2^{56} possible keys for K2

(3) If $E_{K1'}(P) = D_{K2'}(C)$, try the keys on another (P', C')

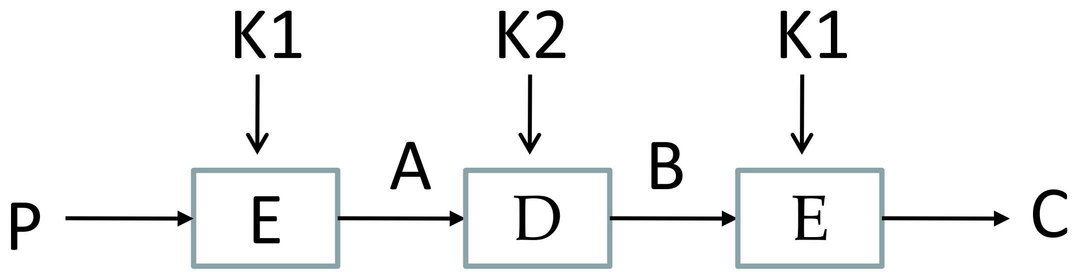
(4) If $E_{K1'}(P') = D_{K2'}(C')$, (K1', K2') = (K1, K2) with high probability

Takes $2 \times 2^{56} = 2^{57}$ steps; not much more than attacking 1-DES.

Triple-DES with Two-Keys

- hence must use 3 encryptions
  - would seem to need 3 distinct keys
- In practice: $C = E_{K1}(D_{K2}(E_{K1}(P)))$
  - Also referred to as **EDE** encryption
- Reason:
  - if K1=K2, then 3DES = 1DES. Thus, a 3DES software can be used as a single-DES.
- Standardized in ANSI X9.17 & ISO8732
- No current known practical attacks
  - Q: What about the meet-in-the-middle attack?

Meet-in-the-Middle Attack on 3DES



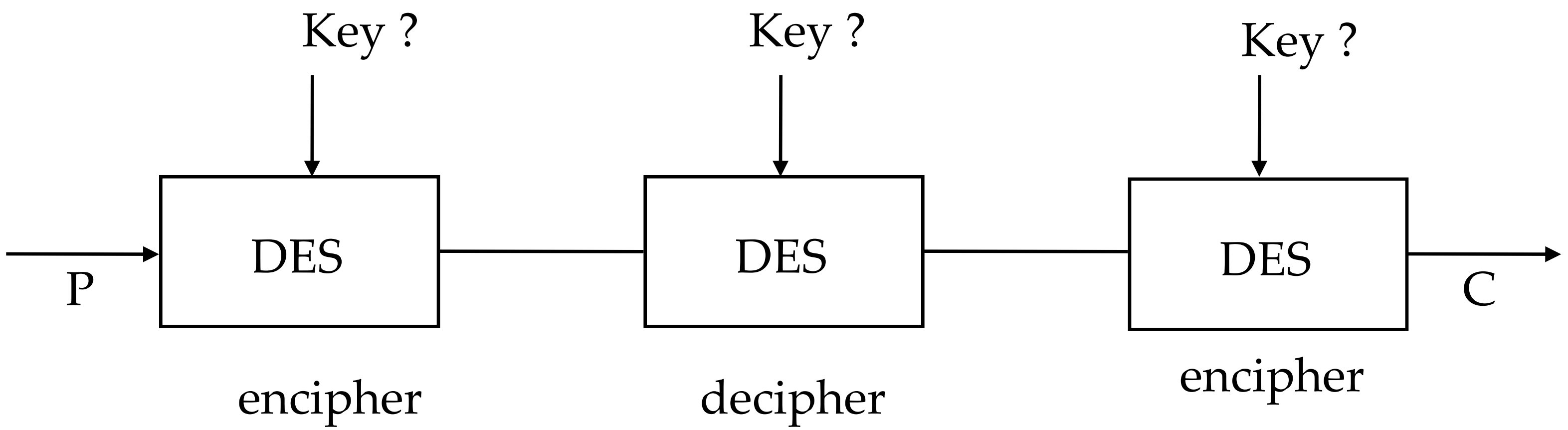
1. For each possible key for K1, encrypt P to produce a possible value for A.
2. Using this A, and C, attack the 2DES to obtain a pair of keys (K2, K1′).
3. If K1′ = K1, try the key pair (K1, K2) on another (C′,P′).
4. If it works, (K1, K2) is the key pair with high probability.
5. It takes $O(2^{55} \times 2^{56}) = O(2^{111})$ steps on average.

Triple-DES with Three-Keys

- Encryption: $C = E_{K3}(D_{K2}(E_{K1}(P)))$.
- If K1 = K3, we have 3DES with 2 keys.
- If K1 = K2 = K3, we have the regular DES.
- So, 3DES w/ 3keys is backward compatible with 3DES w/ 2 keys and with the regular DES
- Some internet applications have adopted 3DES with three keys.
  - E.g., PGP (pretty good privacy) and S/MIME (Secure/Multipurpose Internet Mail Extensions).

Triple-DES

- Triple DES



With two keys: $E_{K1}(D_{K2}(E(_{K1}(M))) = C$

With three keys: $E_{K1}(D_{K2}(E_{K3}(M))) = C$

AES (Advanced Encryption Standard)

AES

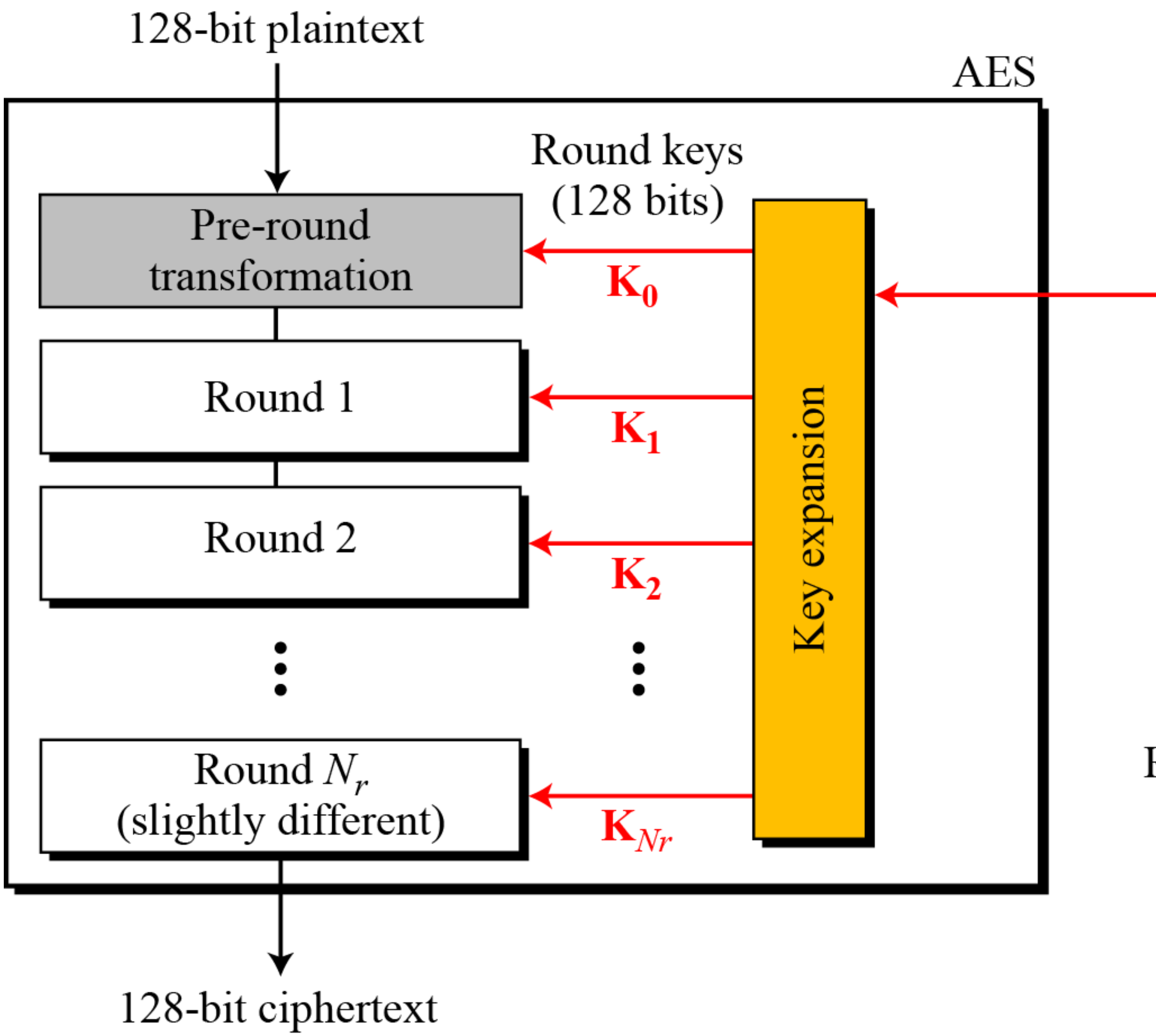
- DES cracked, Triple-DES slow: what next?
- 1997 NIST called for algorithms
- Final five
  - Rijndael (Two Belgians: Joan Daemen, Vincent Rijmen),
  - Serpent(Ross Anderson),
  - Twofish(Bruce Schneier),
  - RC6(Don Rivest, Lisa Yin),
  - MARS (Don Coppersmith, IBM)
- 2000 Rijndael won
- 2002 Rijndael became AES

Overview of AES

- Based on a design principle known as *substitution-permutation network (SPN)*
- Block length is limited to 128 bit
- The key size can be independently specified to 128, 192 or 256 bits

Key size (words/bytes/bits)	4/16/**128**	6/24/**192**	8/32/**256**
Number of rounds	**10**	**12**	**14**
Expanded key size (words/byte)	44/176	52/208	60/240

General design of AES encryption cipher



Nr	Key size
10	128
12	192
14	256

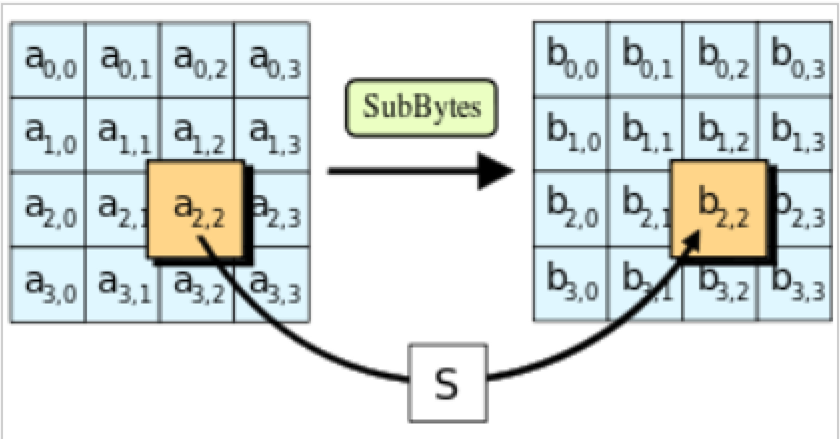
Relationship between number of rounds and cipher key size

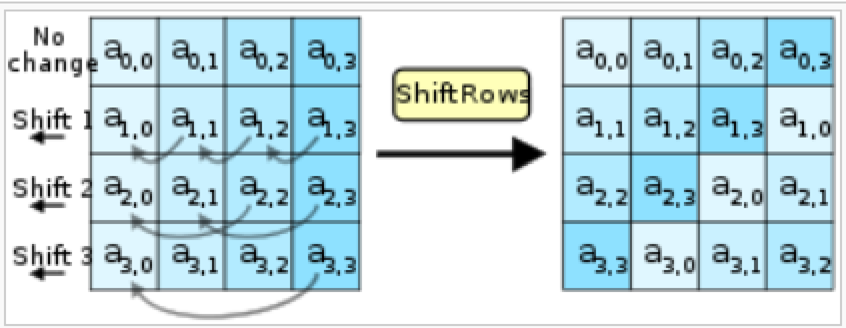
AES

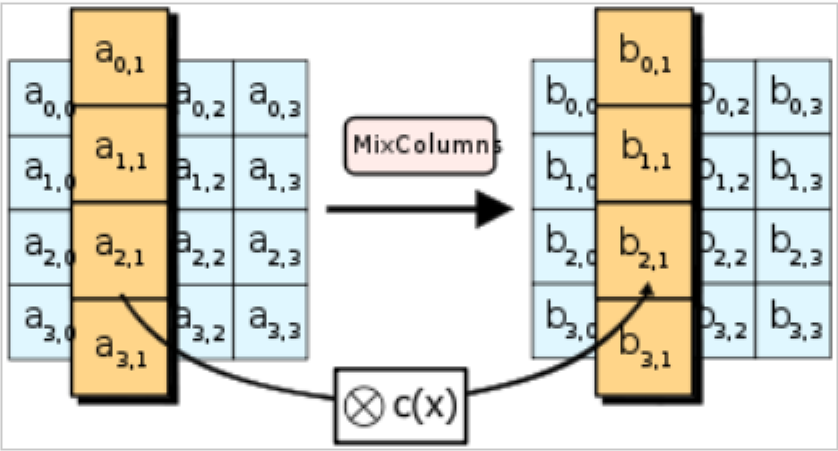
- Each round uses 4 functions
  - ByteSub (nonlinear layer) :
    - referred to as an S-box; byte-by-byte substitution
  - ShiftRow (linear mixing layer)
    - A simple permutation row by row
  - MixColumn (nonlinear layer)
    - A substitution that alters each bye in a column as function of all of the bytes in column
  - AddRoundKey (key addition layer)
    - A simple bitwise XOR of the current block with a portion of the expanded key

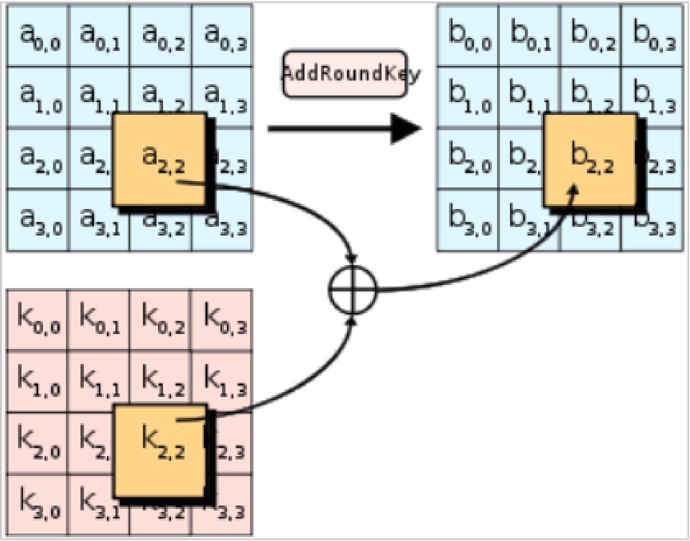
http://www.cs.bc.edu/~straubin/cs381-05/blockciphers/rijndael_ingles2004.swf

AES 4 Steps


SubBytes
S


No change
Shift 1
Shift 2
Shift 3
ShiftRows


MixColumns
c(x)


AddRoundKey

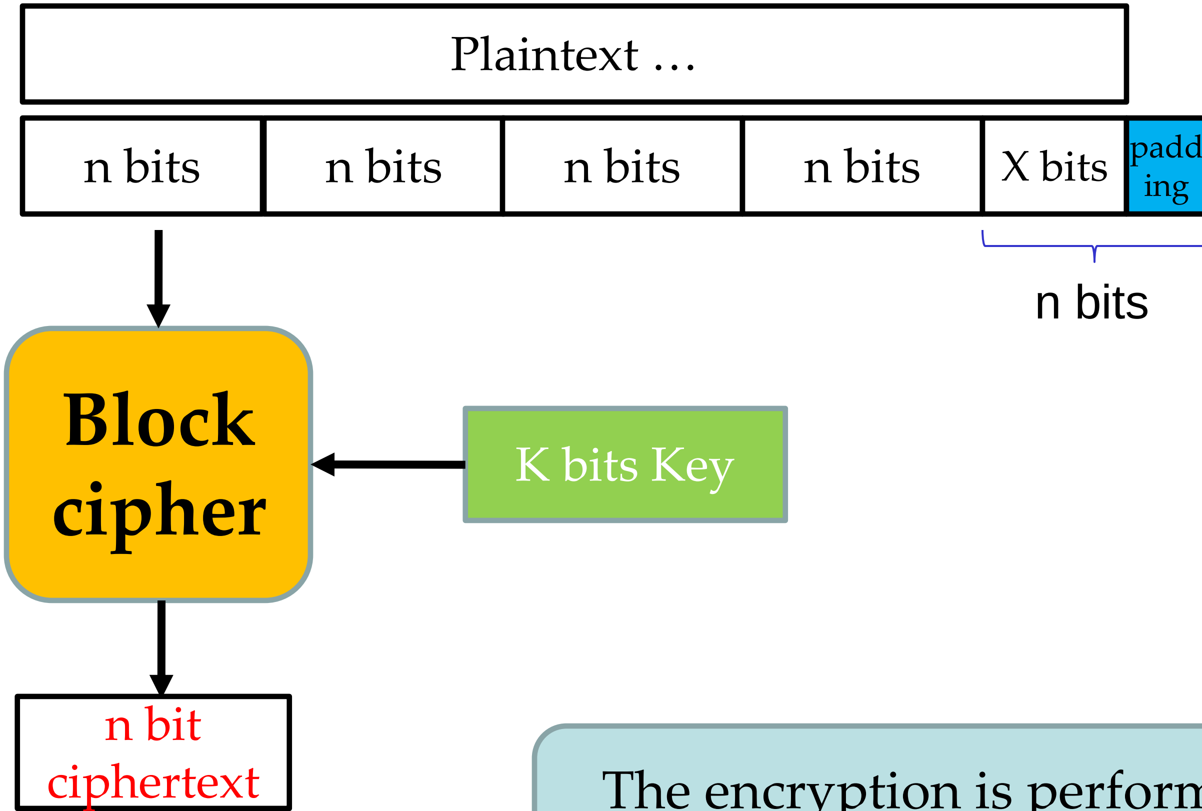
DES vs. AES

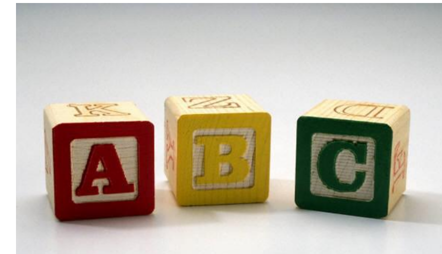
	DES	**AES**
Date	1976	1999
Block size	64	**128**
Key length	56	128, 192, 256
Number of rounds	16	10,12,14
Encryption primitives	Substitution, permutation	Substitution, shift, bit mixing
Cryptographic primitives	Confusion, diffusion	Confusion, diffusion
Structure	**Feistel**	**SPN(** substitution-permutation network**)**
Design	Open	Open
Design rationale	Closed	Open
Selection process	Secret	Secret, but accept open public comment
Source	IBM, enhanced by NSA	Independent cryptographers

Modes of operation

Q: If block size is bigger than 64 bits in case of using DES?

Block cipher





The encryption is performed using one of the operation modes

Common block sizes:
n = 64, 128, 256 bits

Common key sizes:
k = 40, 56, 64, 80, 128,168, 192, 256 bits

Modes of Operation

- block ciphers encrypt fixed size blocks
  - e.g., DES encrypts 64-bit blocks with 56-bit key
- need some way to en/decrypt arbitrary amounts of data in practice
- **ANSI X3.106-1983 Modes of Use** (now FIPS 81) defines 4 possible modes
- subsequently 5 defined for AES & DES
- have **block** and **stream** modes

Modes of Operation

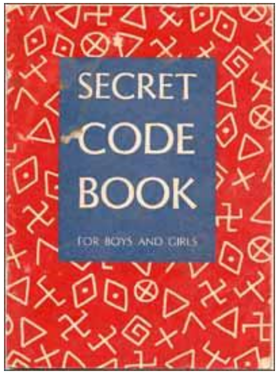
- **ECB** – Electronic Code Book
- **CBC** – Cipher Block Chaining Most popular
- **OFB** – Output Feed Back
- **CFB** – Cipher Feed Back
- **CTR** - Counter

Electronic Codebook Book (ECB)

- Message (plaintext) is broken into independent blocks
- Each block is encrypted independently of the other blocks

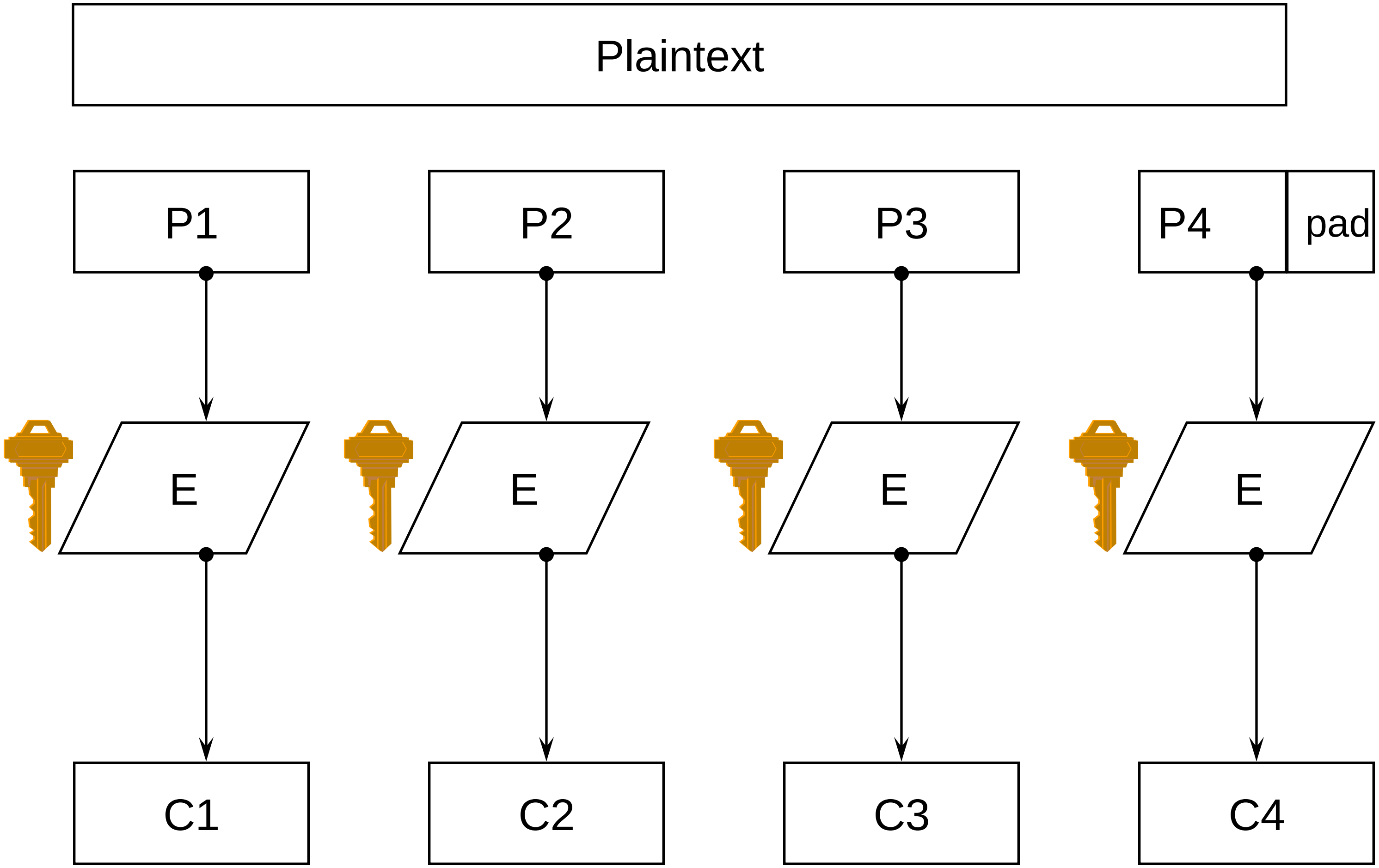
$C_i = DES_{K1}(P_i)$

- Each block is a value which is substituted, and then encrypted like using a codebook.
  - If the same message (e.g., your IRD #) is encrypted (with the same key) and sent twice, their ciphertexts are the same.



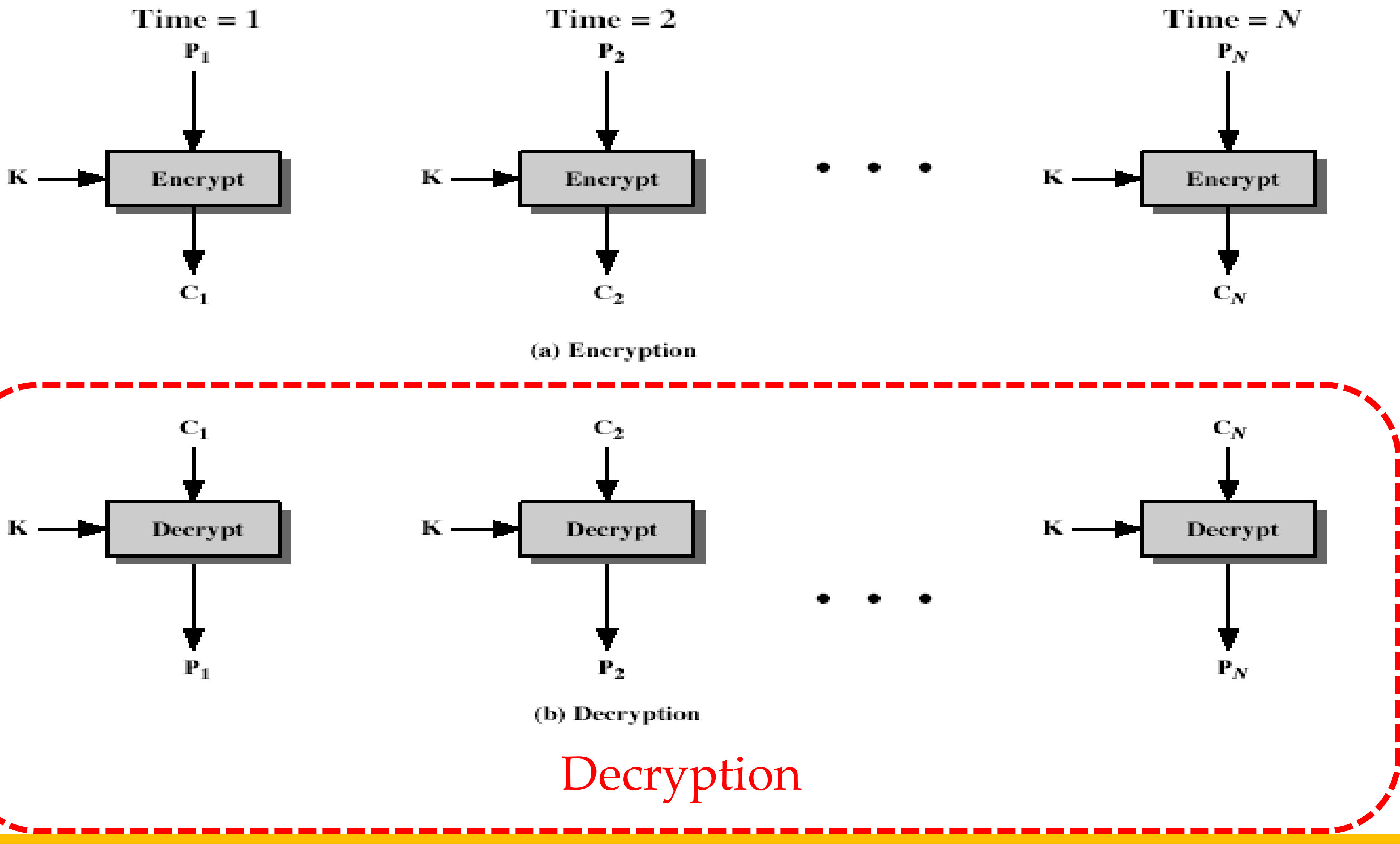
- uses: secure transmission of single values

Electronic Codebook Book mode



- Pad last block, if necessary

ECB (both encryption/decryption)



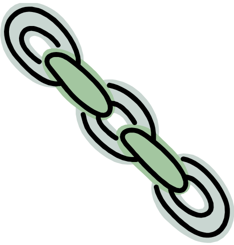
Advantages and Limitations of ECB

- Message repetitions may show in ciphertext
  - if aligned with message block
  - particularly with data such graphics
  - or with messages that change very little, which become a code-book analysis problem
- Weakness is due to the encrypted message blocks being independent
- Main use is sending a few blocks of data

Any ideas to overcome the ECB mode?

Cipher Block Chaining (CBC)

- message is broken into blocks
- linked together in encryption operation
- each previous cipher blocks is chained with current plaintext block



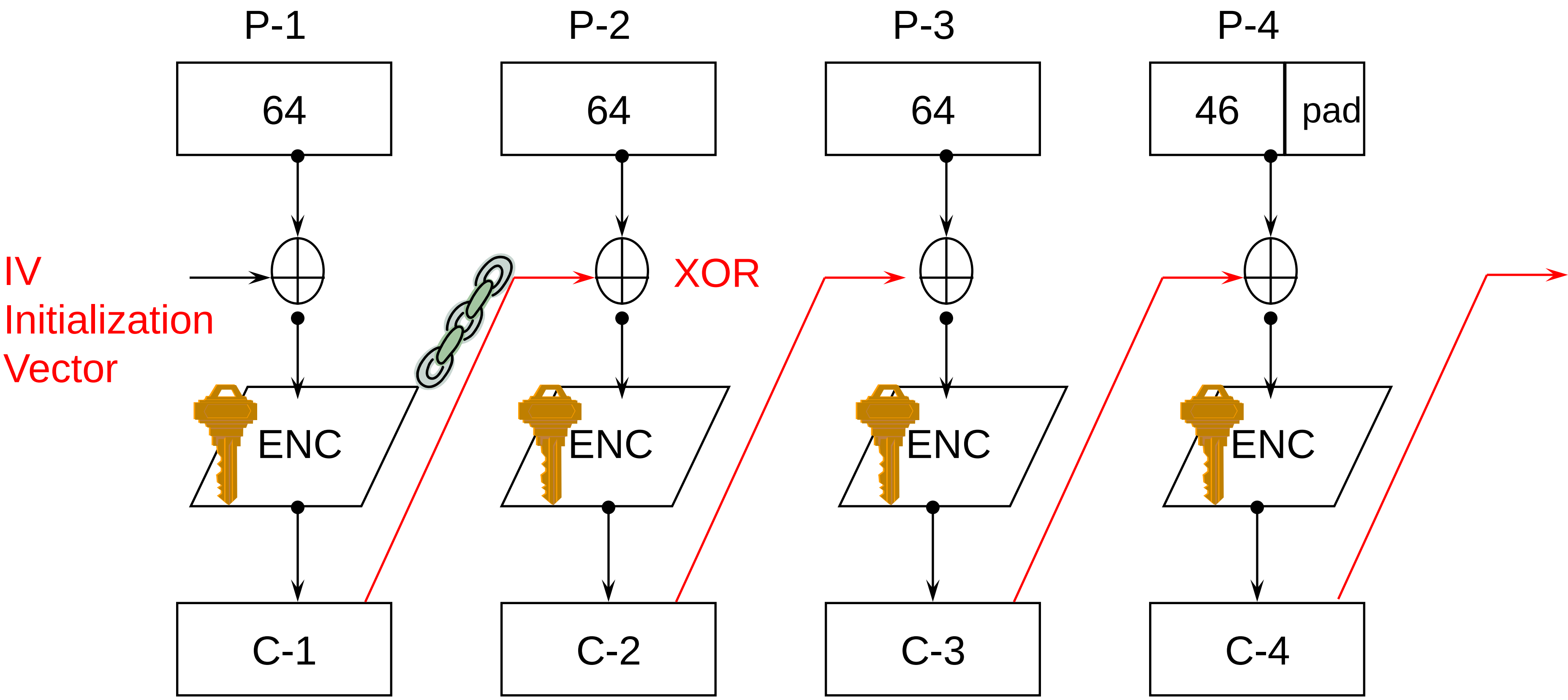
- use Initial Vector (IV) to start process

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

  $C_{-1} = IV$

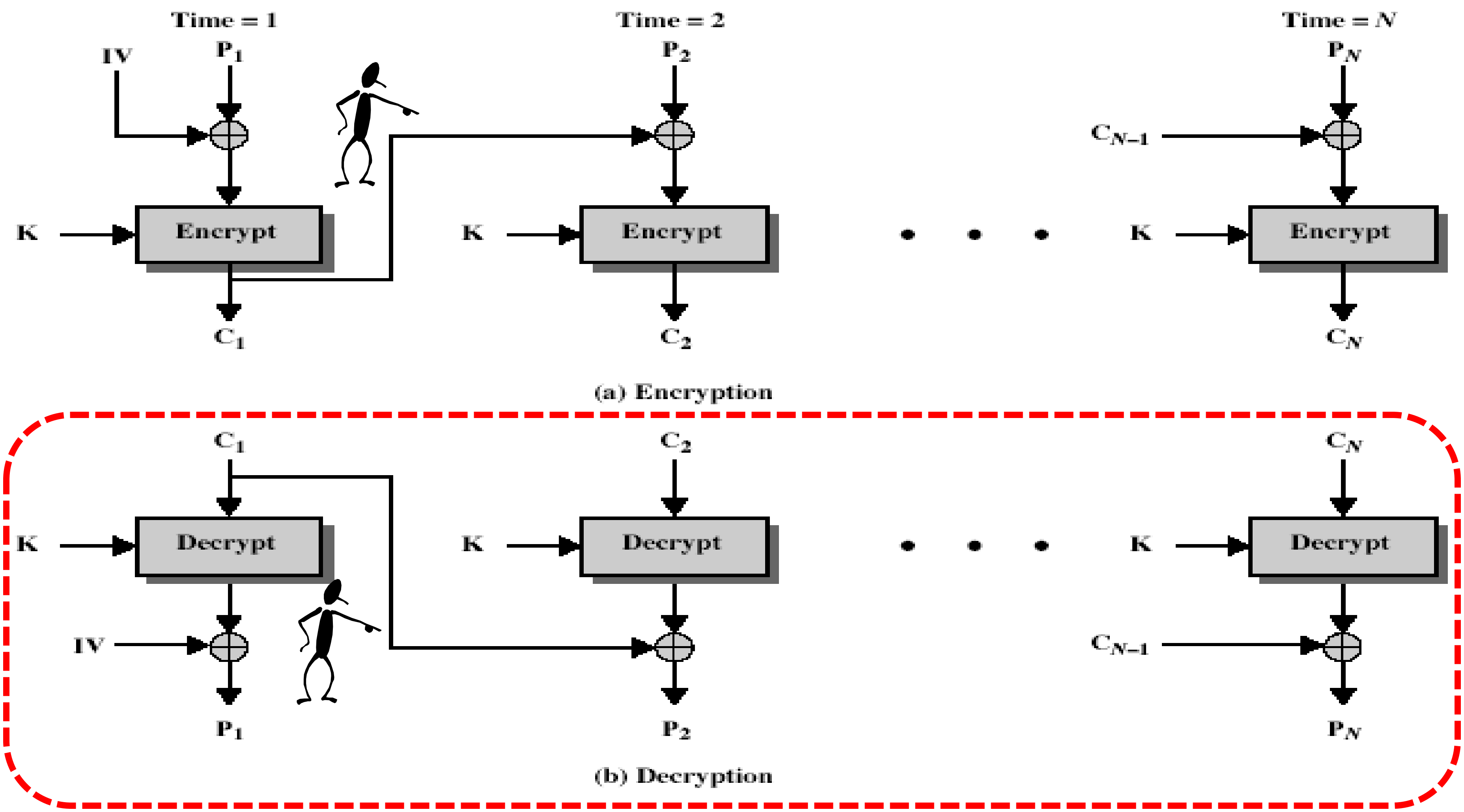
- uses: general block oriented transmission
  - e.g., IPsec uses 3DES-CBC, AES-CBC

Cipher Block Chaining (CBC)



- Pad last block, if necessary
- Random Block called IV is required to be random/pseudo random.

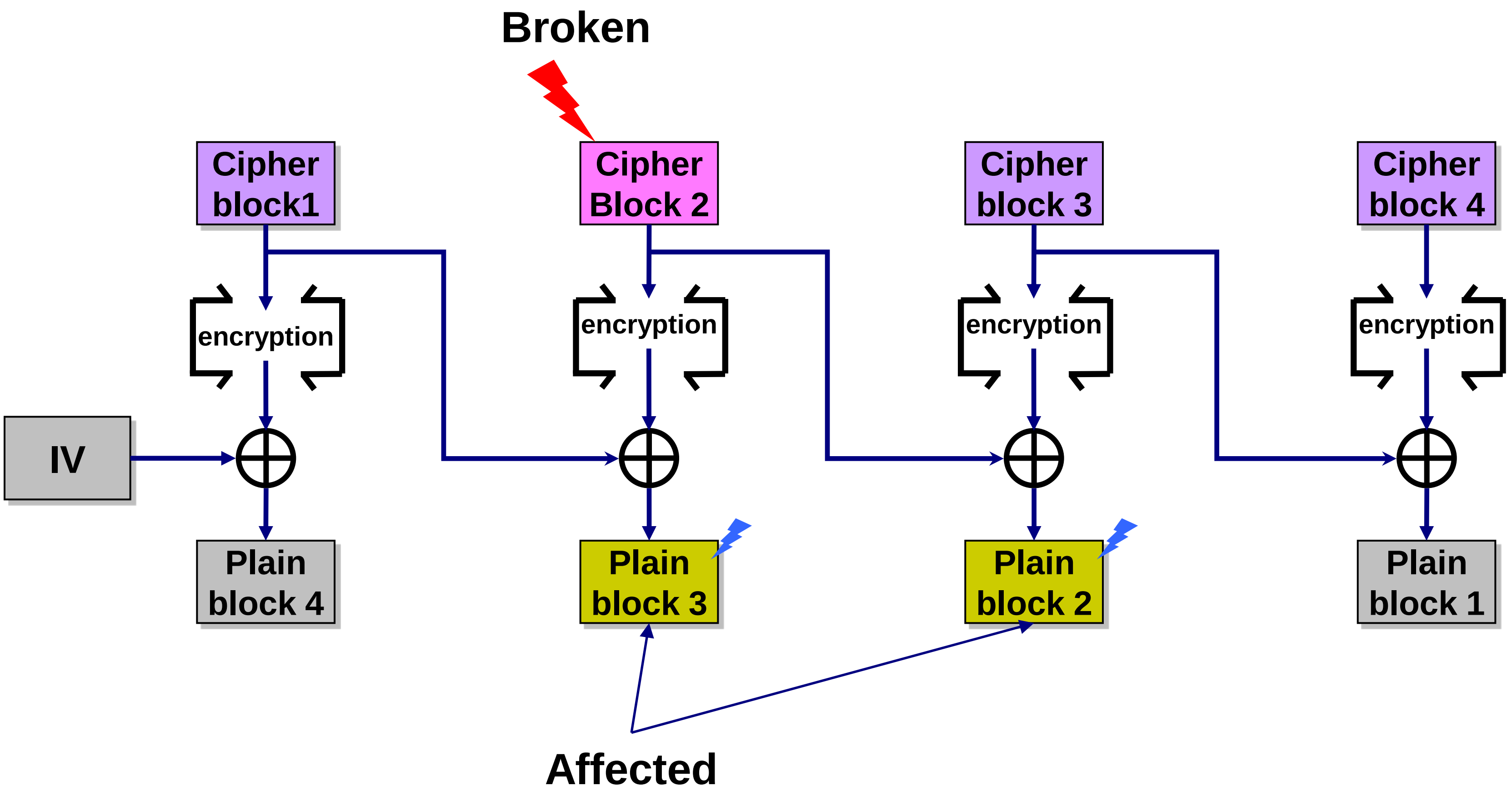
Cipher Block Chaining (CBC) : E/D



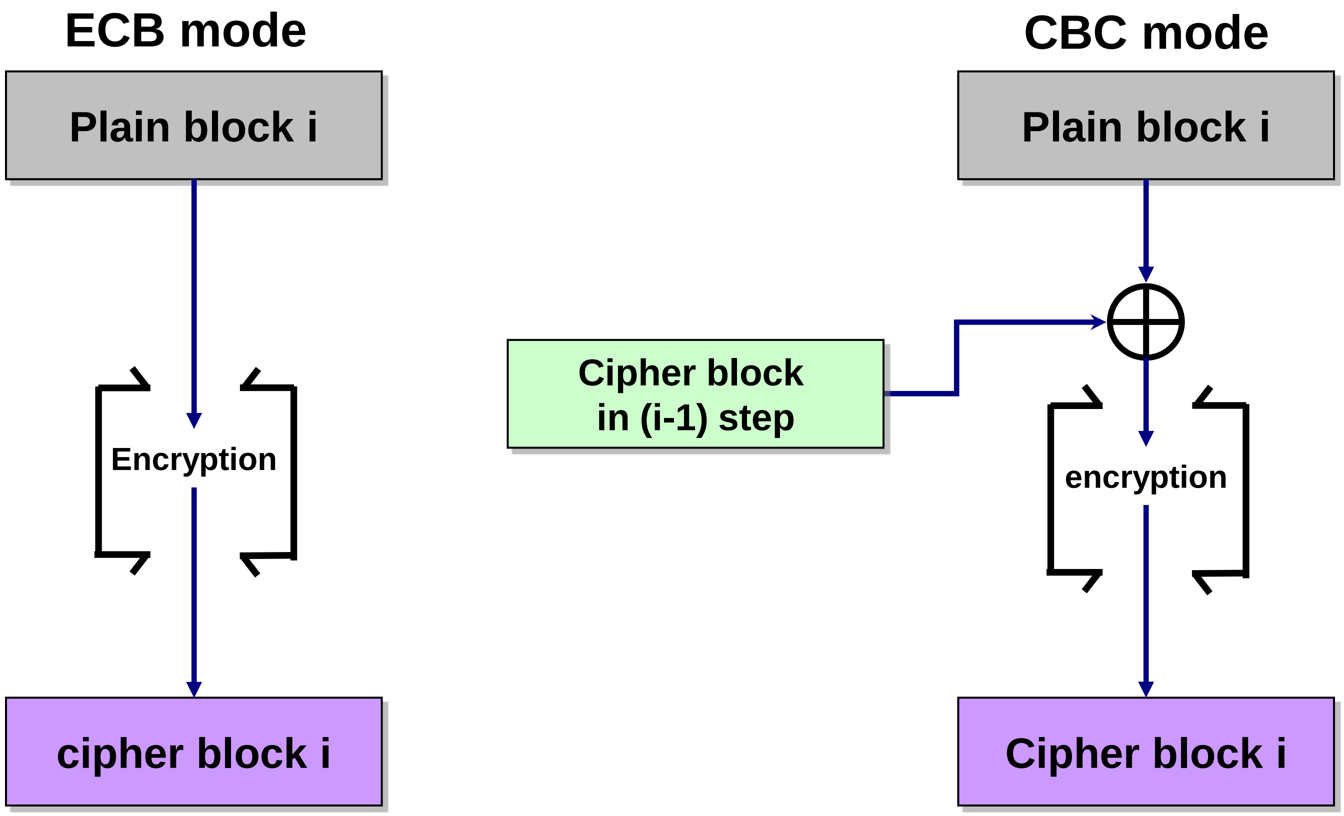
Advantages and Limitations of CBC

- A ciphertext block depends on **all** blocks before it
- So, repeated plaintext blocks are encrypted differently.
- need **Initialization Vector** (IV)
  - must be known to sender & receiver
  - if sent in clear, attacker can change bits of first block, and change IV to compensate, hence IV must either be a fixed value (Integrity of IV should be guaranteed)
  - or must be sent encrypted in **ECB** mode before rest of message

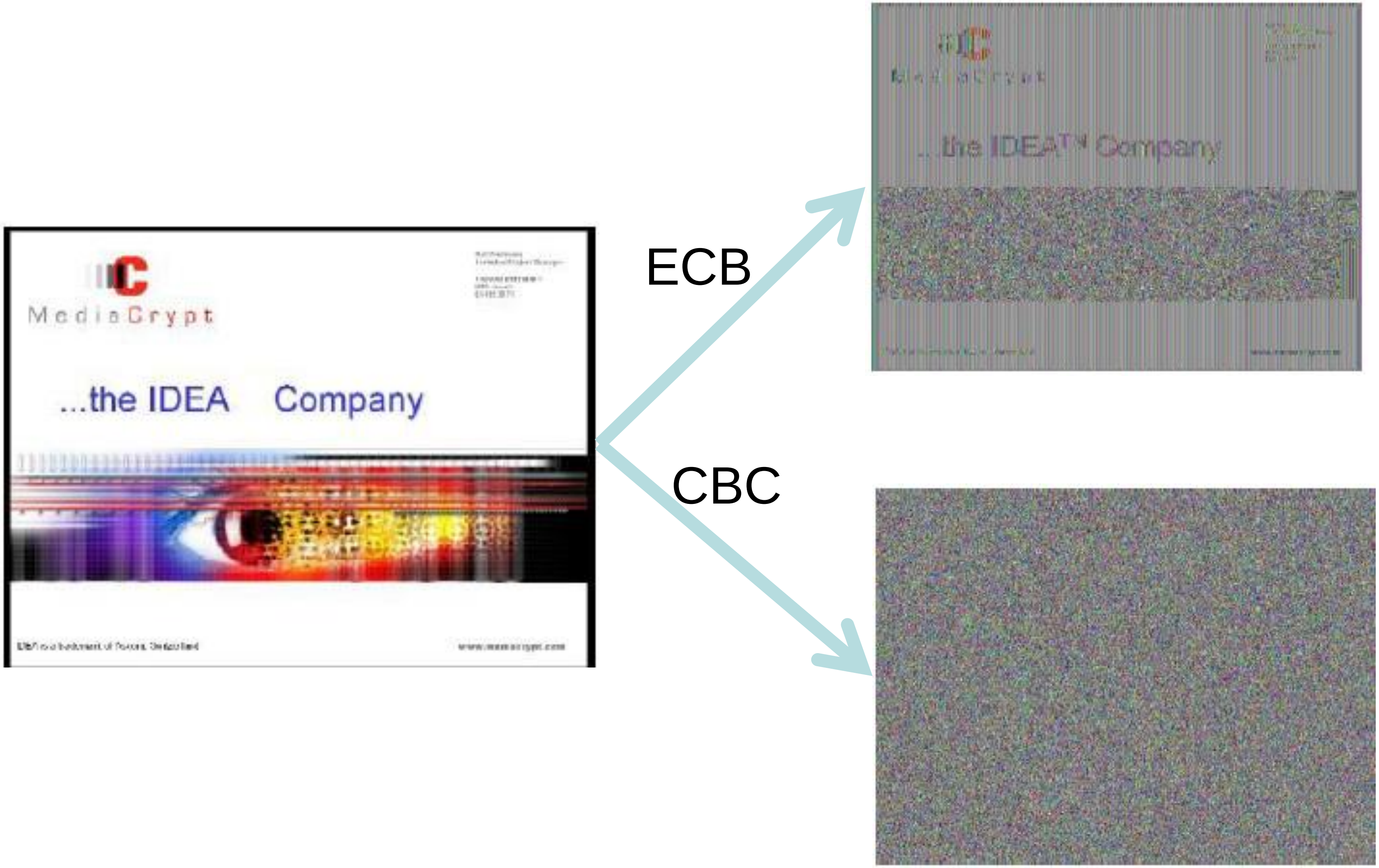
Error propagation in CBC



ECB vs. CBC mode



ECB vs. CBC mode



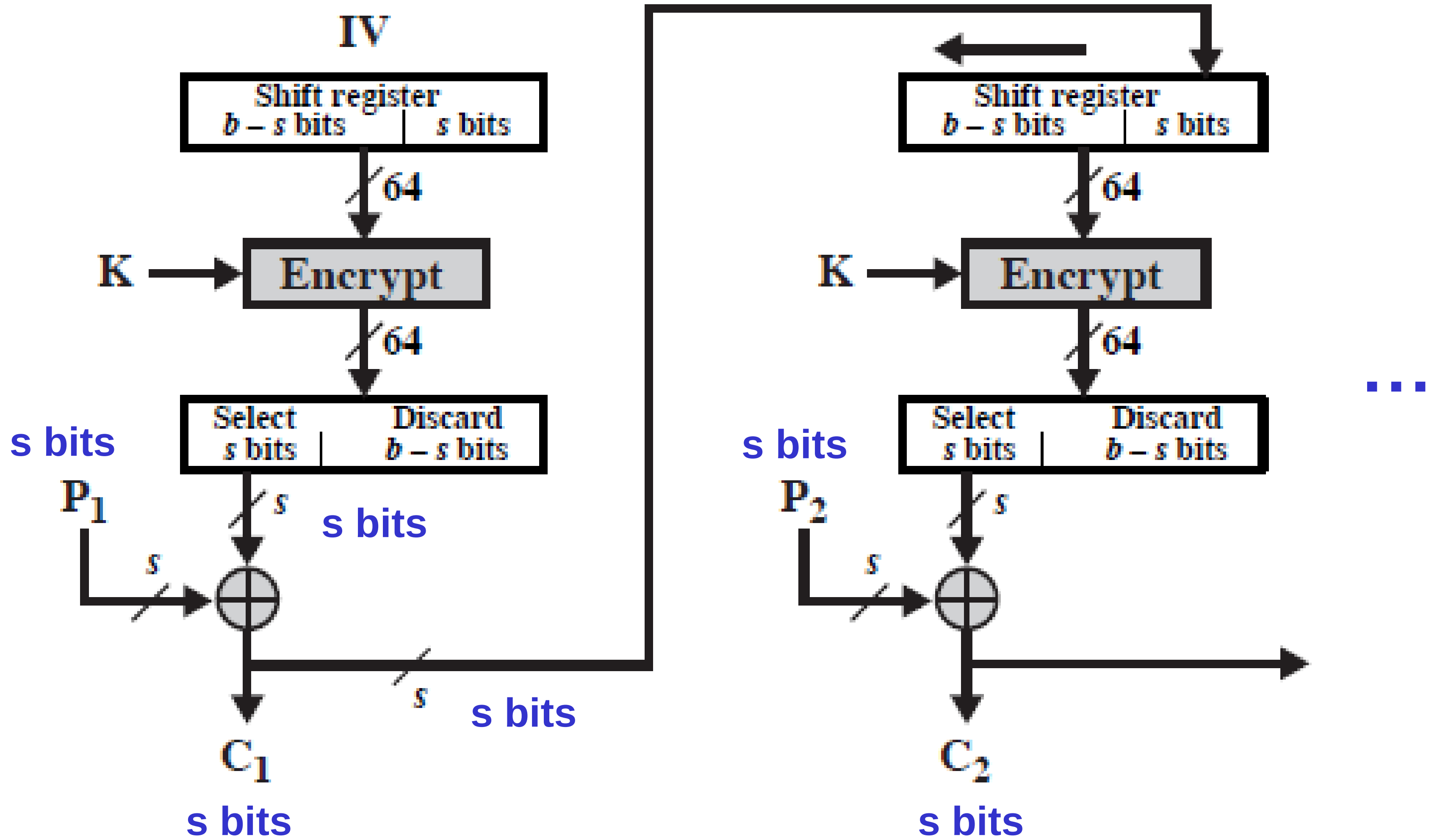
Cipher Feed back (CFB) Mode

- The plaintext is divided into segments of s bits (where $s \leq$ block-size): $P_1,\ P_2,\ P_3,\ P_4,\ \ldots$
- Encryption is used to generate a sequence of keys, each of s bits: $K_1,\ K_2,\ K_3,\ K_4,\ \ldots$
- The ciphertext is $C_1,\ C_2,\ C_3,\ C_4,\ \ldots,$ where $C_i = P_i \oplus K_i$

Cipher Feed back (CFB) Mode

- Uses cipher block used in the previous step as input of cipher in the next step
- What does it mean “feedback”?
  - Cipher is used as input of the cipher

Cipher Feed Back (CFB): Encryption

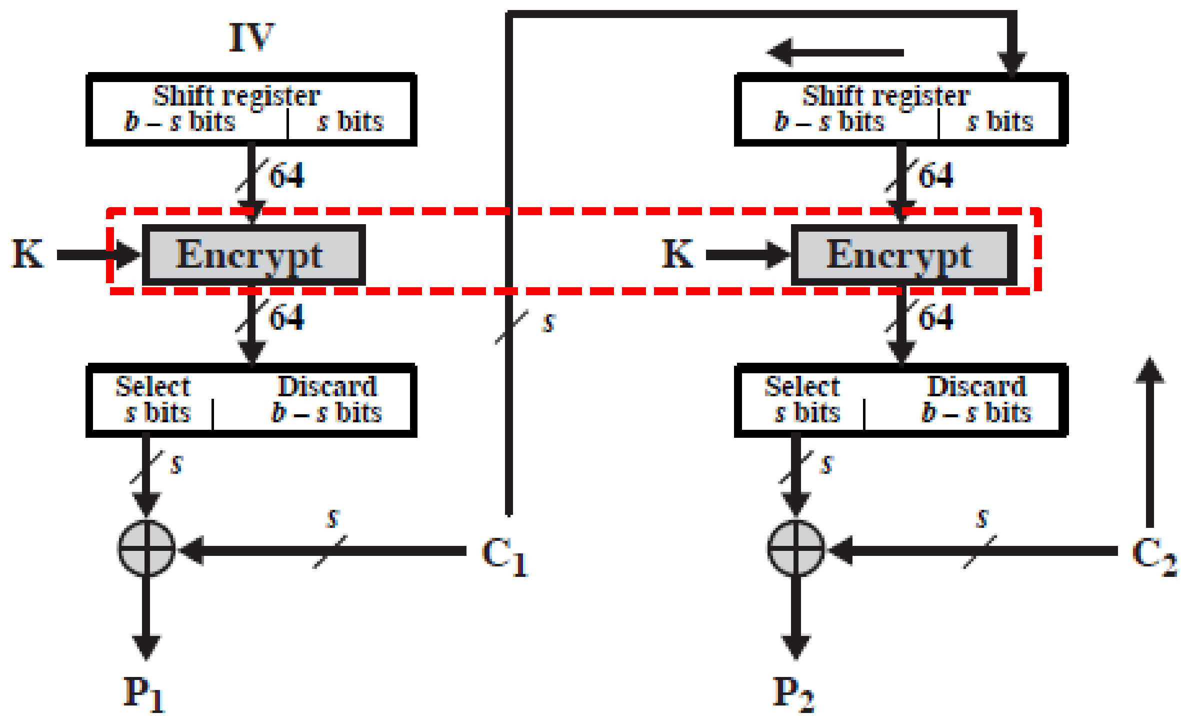


Cipher Feed Back (CFB): Decryption

- Generate key stream $K_1, K_2, K_3, K_4, \ldots$ the same way as for encryption.
- Then decrypt each ciphertext segment as:

$$P_i = C_i \oplus K_i$$

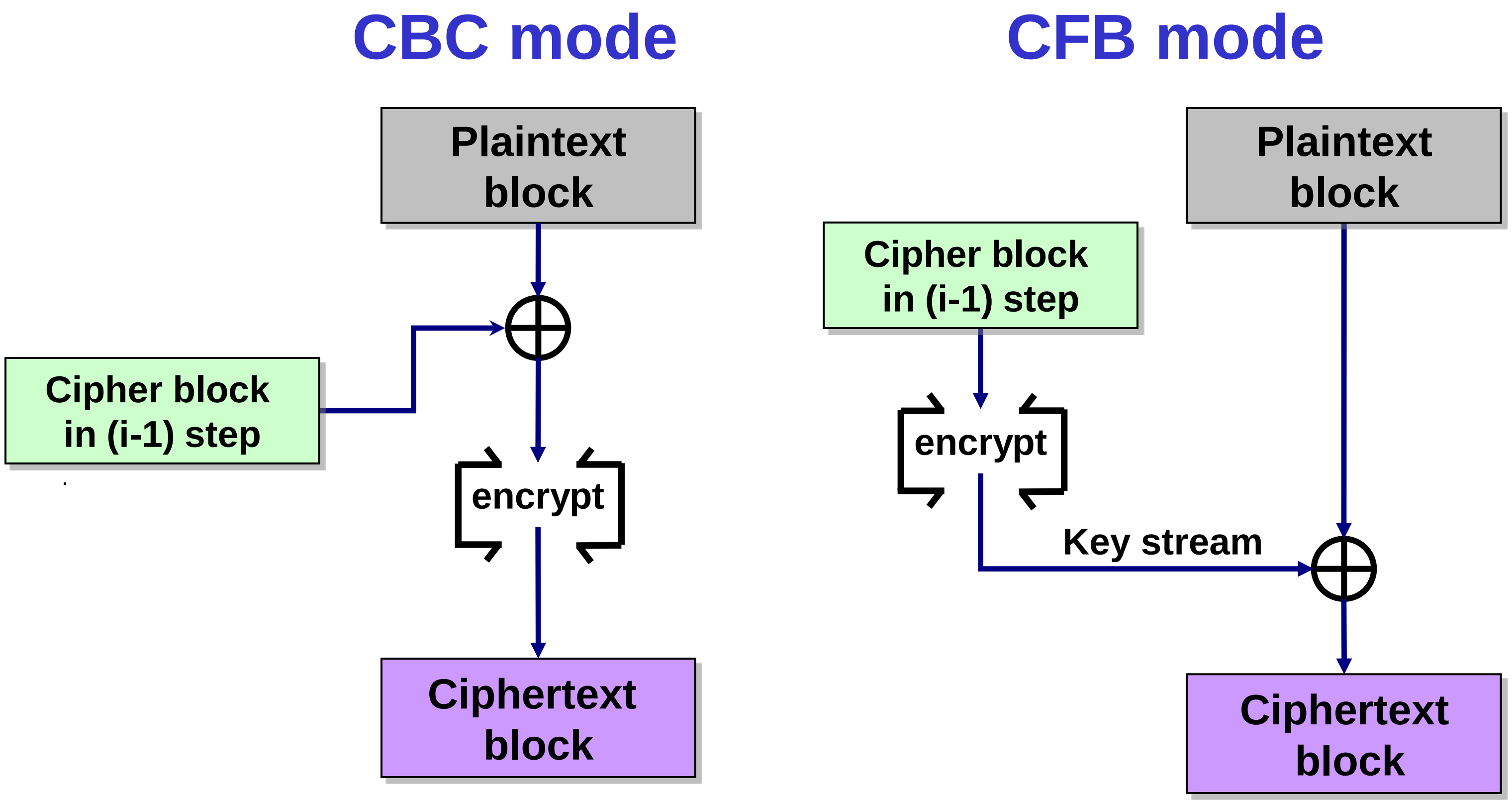
It does not decrypt but encrypt



Cipher Feed Back (CFB)

- The block cipher is used as a **stream cipher**.
- Appropriate when data arrives in bits/bytes.
  - s can be any value; a common value is s = 8.
  - standard allows any number of bit (1, 8, 64 or 128 etc) to be feed back denoted CFB-1, CFB-8, CFB-64, CFB-128 etc
- A ciphertext segment depends on the current and all preceding plaintext segments.
- A corrupted ciphertext segment during transmission will affect the current and next several plaintext segments.

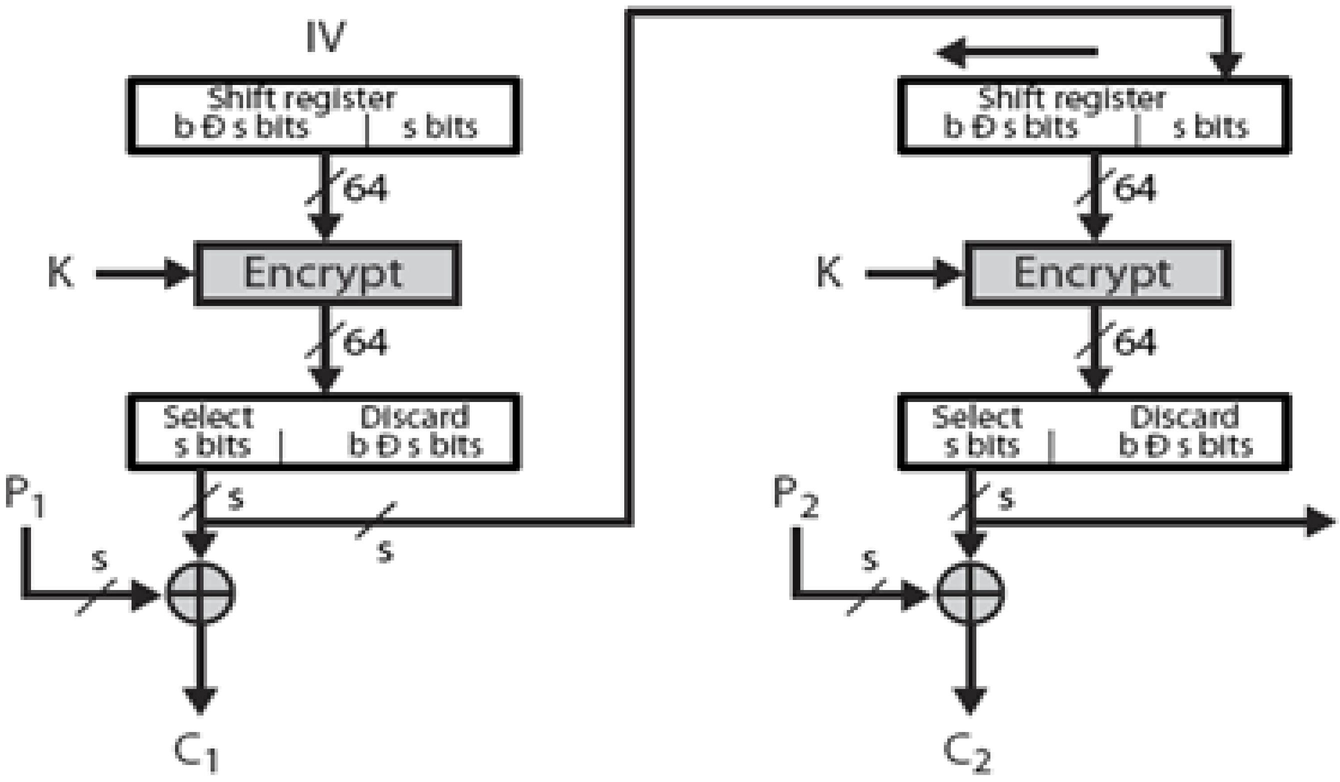
CBC vs. CFB



CBC mode
Plaintext block
Cipher block in (i-1) step
encrypt
Ciphertext block
CFB mode
Plaintext block
Cipher block in (i-1) step
encrypt
Key stream
Ciphertext block

Output Feed Back (OFB) mode

OFB



Output Feed Back (OFB) mode

- message is treated as a stream of bits (s bits)
- **output of cipher** is added to message
- output is then feed back
- feedback is independent of message
- can be computed in advance

  $C_i = P_i \text{ XOR } O_i$

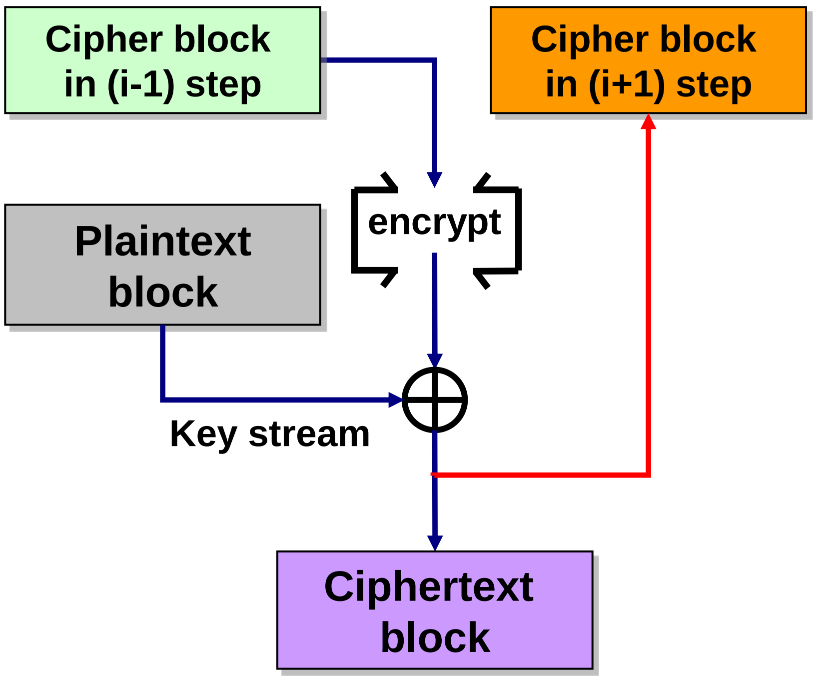
  $O_i = DES_{K1}(O_{i-1})$

  $O_{-1} = IV$

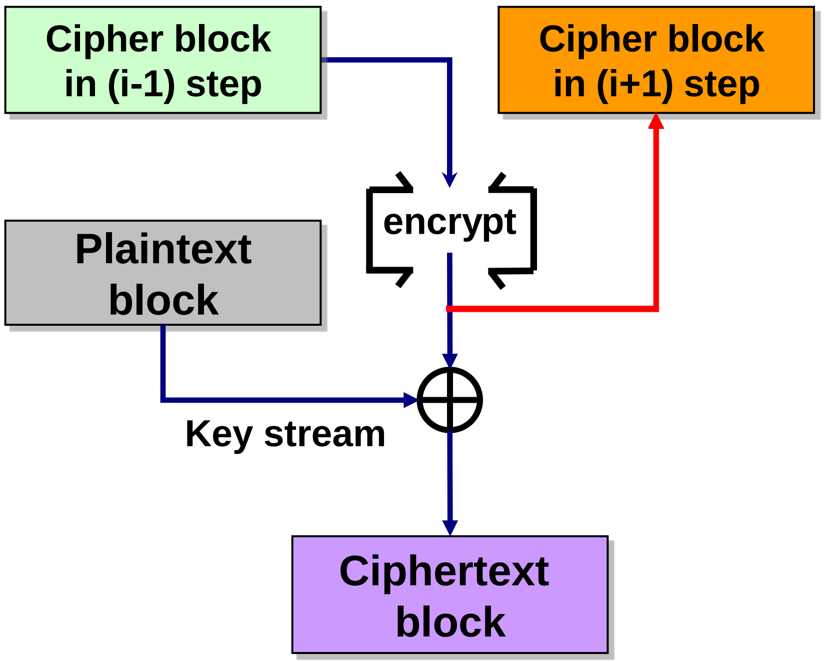
- uses: stream encryption on noisy channels (e.g., satellite TV transmissions etc)

CFB vs. OFB

CFB mode

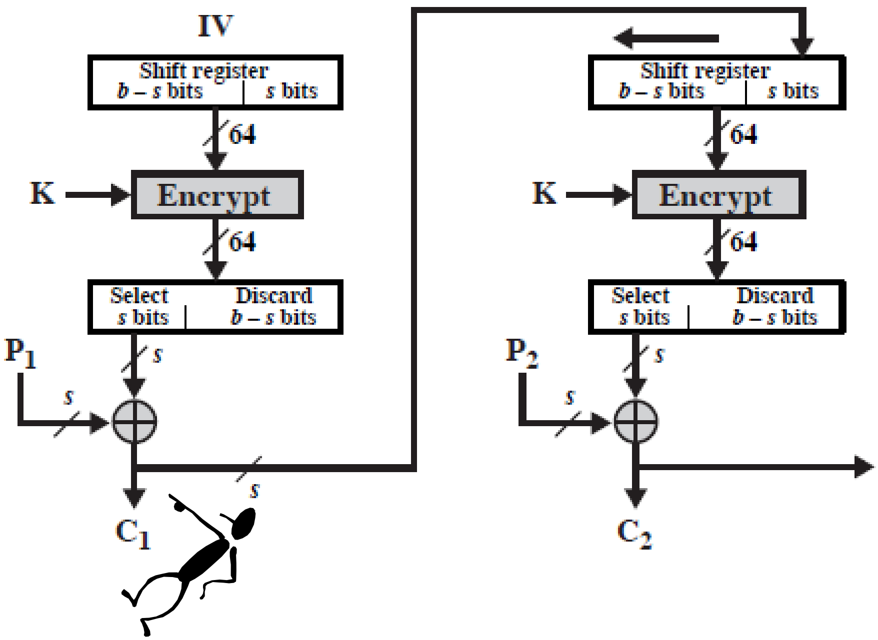


OFB mode

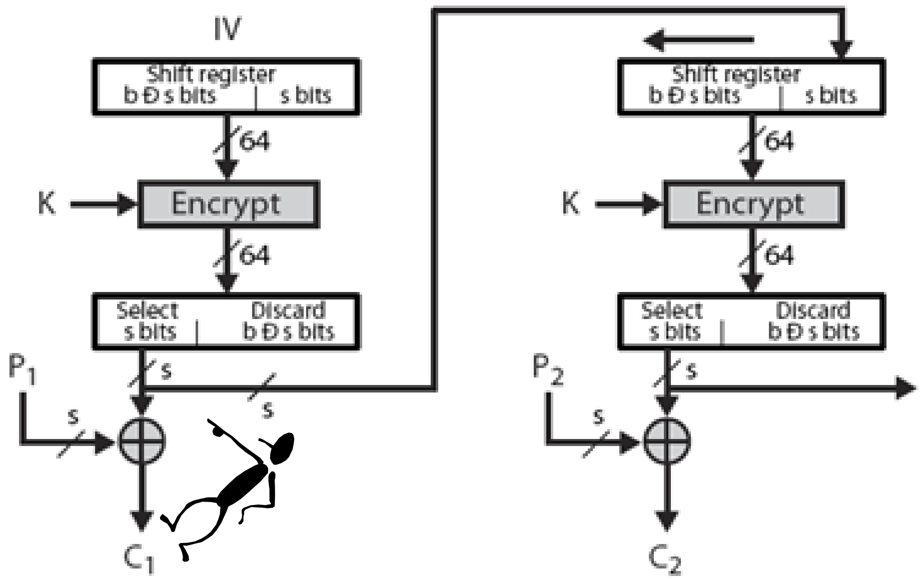


CFB vs. OFB (contd)

CFB



OFB



Advantages and Limitations of OFB

- bit errors do not propagate
- more vulnerable to message stream modification
- a variation of a Vernam cipher
  - hence must **never** reuse the same sequence (key+IV) ;
  - otherwise 2 ciphertexts can be combined, cancelling these bits
- sender & receiver must remain in sync

Vernam cipher: the plaintext is XORed with a random or pseudorandom stream of data (the "keystream") of the same length to generate the ciphertext

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)

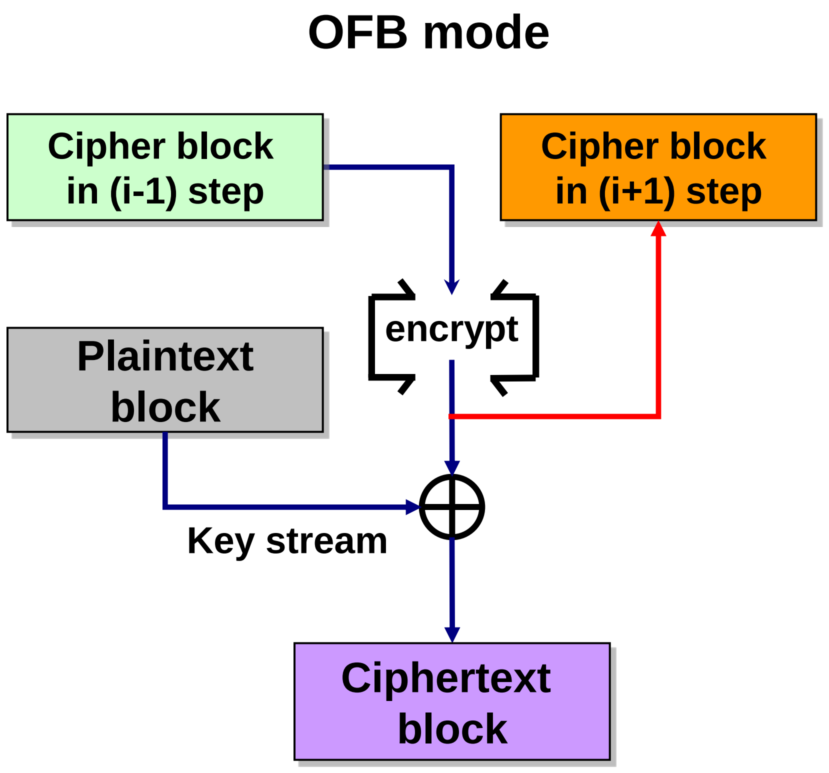
$$C_i = P_i \text{ XOR } O_i$$

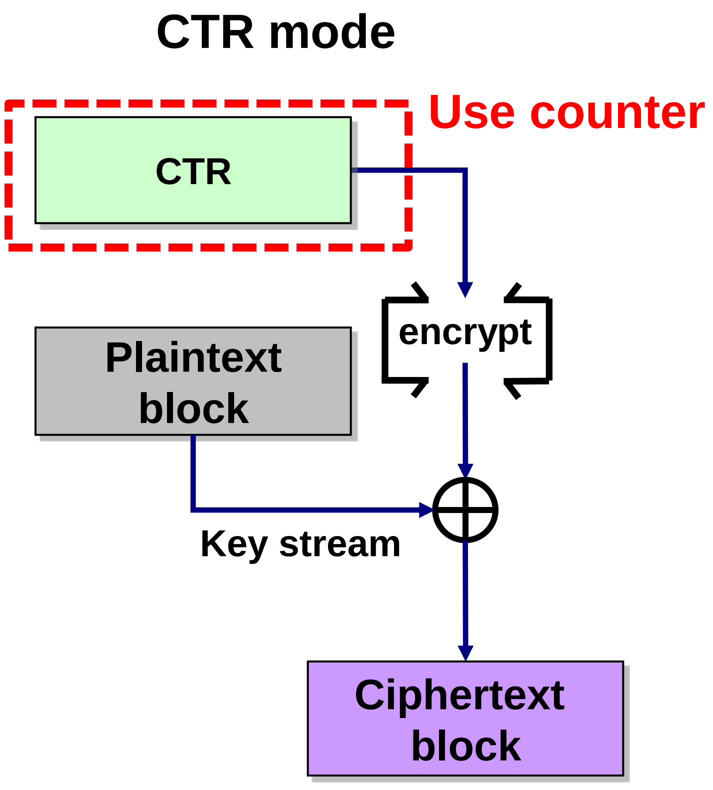
$$O_i = DES_{K1}(i)$$

- uses: high-speed network encryptions
  - e.g., AES-CTR (i.e., AES in CTR mode)

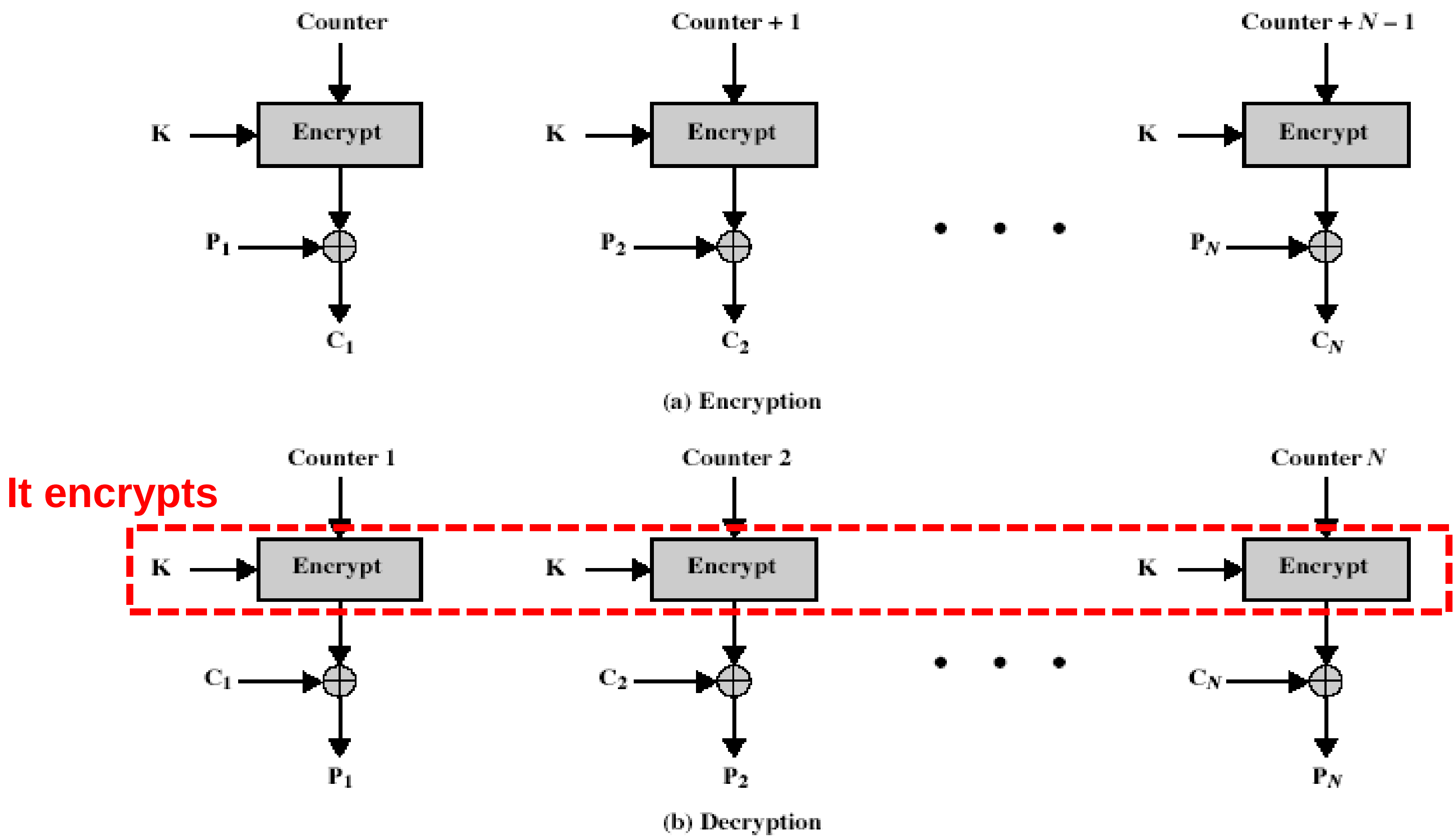
OCB (Offset Codebook Mode) (Counter Mode)
[new] Ref: P Rogaway, OCB Mode, http://csrc.nist.gov/encryption/aes

OFB vs. CTR mode





Counter (CTR)



Q: how to generate counter?

CTR

- A counter T is initialized to some IV (nonce) and then incremented by 1 for each subsequent plaintext block.
- Counter example (128 bits/16 bytes).

66 1F 98 CD 37 A3 8B 4B 00 00 00 00 00 00 00 01

Nonce (an arbitrary number **)** | **Block number**

- **66 1F 98 CD 37 A3 8B 4B 00 00 00 00 00 00 00 01** (initial)
- **66 1F 98 CD 37 A3 8B 4B 00 00 00 00 00 00 00 02** (counter 2)
- **66 1F 98 CD 37 A3 8B 4B 00 00 00 00 00 00 00 03** (counter 3)
- **66 1F 98 CD 37 A3 8B 4B 00 00 00 00 00 00 00 04** (counter 4)

⋮ ⋮

Advantages and Limitations of CTR

- Needs only the encryption algorithm (so do CFB and OFB)
- Fast encryption/decryption;
  - blocks can be processed (encrypted or decrypted) in parallel in SW/HW; good for high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but as in OFB, must ensure never reuse key/counter values, otherwise could break

Modes of Operation: summary

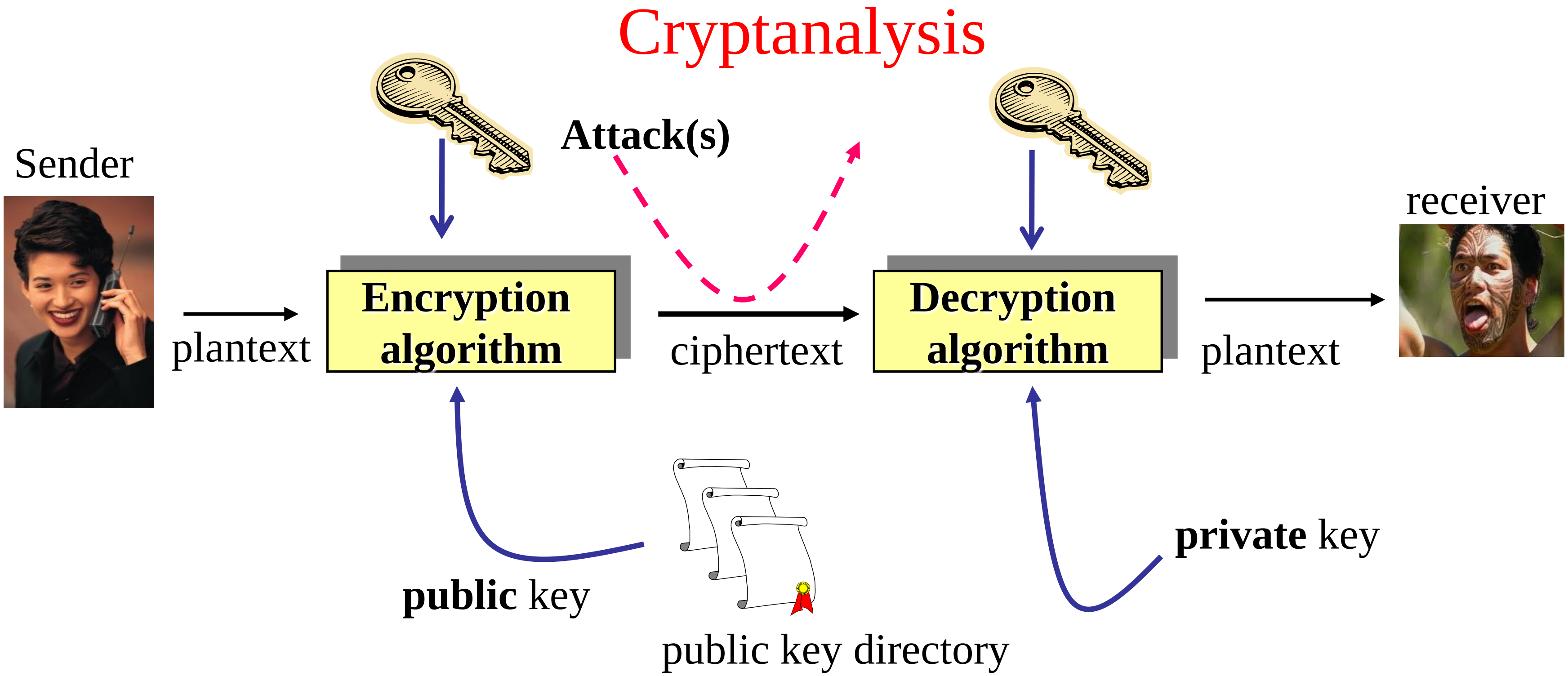
- **ECB** – Electronic Code Book   Don't use
- **CBC** – Cipher Block Chaining   Most popular, e.g., DES-CBC
- **OFB** – Output Feed Back
- **CFB** – Cipher Feed Back
  (OFB, CFB:) Use CTR
- **CTR** - Counter   e.g., AES-CTR

Q: What security objective does this provide?

A: Confidentiality

Q: How do we know the encryption (block cipher) is secure?

Cryptanalysis



Cryptanalysis (cont'd)

- objective to recover key not just message
- general approaches:
  - cryptanalytic attack
  - brute-force attack

Breaking Ciphers

- **Ciphertext only** (COA, Known-ciphertext)
  - Attacker can only access to a set of ciphertext
- **Known plaintext** (KPA)
  - know/suspect plaintext & ciphertext
- **Chosen plaintext** (CPA)
  - select plaintext to be encrypted and obtain ciphertext
- **Chosen ciphertext**
  - select ciphertext and obtain plaintext under an unknown key
- **Chosen text**
  - select plaintext or ciphertext to en/decrypt

Ciphertext-only attack

Known to attacker	**$C_1, C_2, \ldots, C_n$**
Objective	1) $P_1, P_2, \ldots, P_n$
	2) Key K
	3) Algorithm: $C_{n+1} \rightarrow P_{n+1}$

Ciphertexts generated using the same key

Find an algorithm that can decrypt any message encrypted using the key *K*.

Known-plaintext attack

Known to attacker	**$(P_{1,}C_1), (P_{2,}C_2),, \ldots(P_{n,}C_n),$**
Objective	1) Key K
	2) Algorithm: C_{n+1} -> P_{n+1}

Attacker **cannot** select these pairs

Chosen-plaintext attack

Attackers **can** **select** P_1, P_2, …, P_n before the attack begins and **cannot** obtain additional pair after the attack has begun.

Known to attacker	**(P_1, C_1), (P_2, C_2),, …(P_n, C_n),**
Objective	1) Key K
	2) Algorithm: C_{n+1} -> P_{n+1}

Chosen-ciphertext attack

Attackers **can** select C_1, C_2, …, C_n before the attack begins.

Known to attacker	**(P_1, C_1), (P_2, C_2),, …(P_n, C_n),**
Objective	1) Key K
	2) Algorithm: C_{n+1} -> P_{n+1}

This attack is used against **public key algorithm.** Attacker can by itself generate the ciphertexts using the public key of the target.

Result of Attacks

Objective	**1) Key K**
	2) Algorithm: C_{n+1} -> P_{n+1}

- Total break:
  - found the key
- Global deduction:
  - Was not successful in finding the key, but successful in finding an algorithm that can decrypt any ciphertexts of the target.
- Instance deduction:
  - Obtained some plaintexts from some ciphertexts.
- Information deduction:
  - Obtained a partial bits of plaintext of partial bits of the target key

Secureness of an cipher

- **Computational secure**
  - Cost of breaking the cipher exceeds the value of the encrypted information (e.g., 1 million NZD cost vs. 1000 NZD secret)
  - The time required to break the cipher exceeds the useful lifetime of the information (e.g., 1 month to break the all black's tactics)
- Provably secure:
  - the security of the system can be proven to be equivalent to a hard problem
- Unconditional security
  - Even if the attacker has infinite amount of computing resource, the attacker cannot succeed in cryptanalyzing the algorithm
  - Only one-time pad is proven to be unconditionally secure

Brute Force Search

- always possible to simply try every key
  - e.g., PIN number (0000)
- most basic attack, proportional to key size
- assume either know / recognise plaintext

Key Size (bits)	Number of Alternative Keys	Time required at 1 decryption/µs	Time required at 10^6 decryptions/µs
32	$2^{32} = 4.3 \times 10^9$	2^{31} µs = 35.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	2^{55} µs = 1142 years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} µs= 5.4×10^{24} years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	2^{167} µs= 5.9×10^{36} years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	2×10^{26} µs= 6.4×10^{12} years	6.4×10^{6} years

Q: Is DES computationally secure?

Q: Why do we need public key encryptions?