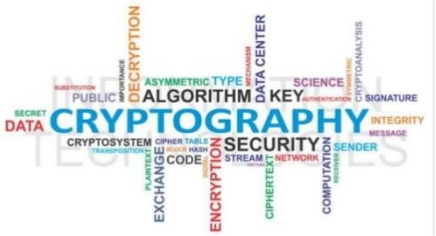
Cryptography & Security

4. Block ciphers

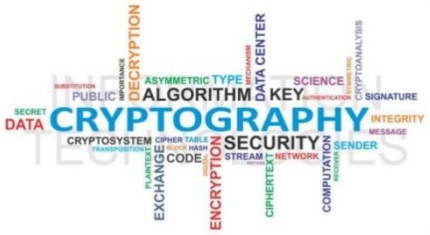
Aureliu Zgureanu,
PhD, Associate Professor

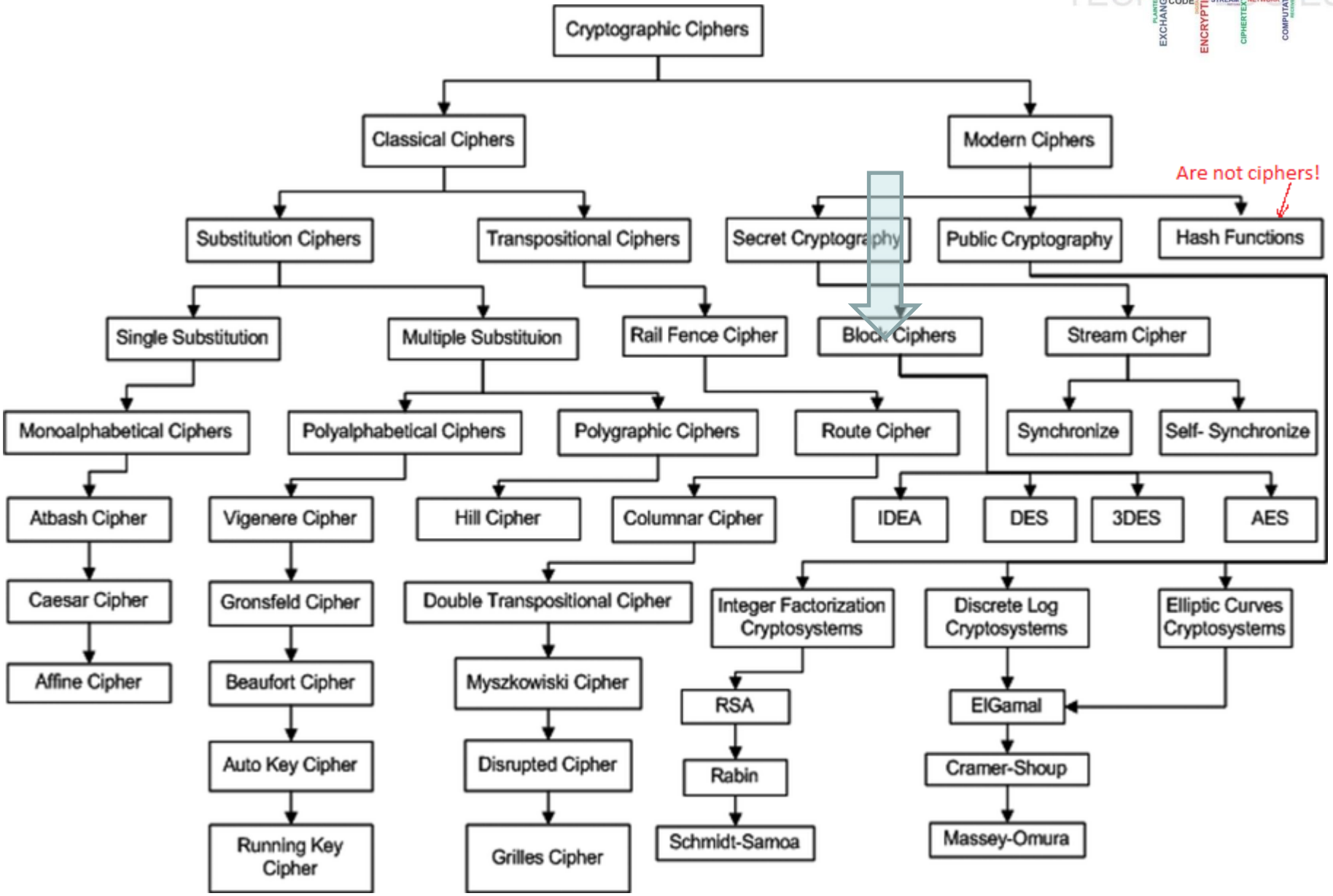
Content of this part



- Introduction to DES
- Overview of the DES Algorithm
- Internal Structure of DES
- Decryption
- Security of DES
- Advanced Encryption Standard
- Block Cipher Modes of Operation

Classification Of Ciphers





Private Key Encryption Syntax

- Message Space: $\mathcal{M}$
- Key Space: $\mathcal{K}$
- Three Algorithms
  - $\mathrm{Gen}(R)$ (Key-generation algorithm)
    - Input: Random Bits R
    - Output: Secret key $\mathrm{k} \in \mathcal{K}$
  - $\mathrm{Enc}_{\mathrm{k}}(m)$ (Encryption algorithm)
    - Input: Secret key $\mathrm{k} \in \mathcal{K}$ and message $\mathrm{m} \in \mathcal{M}$
    - Output: ciphertext *c*
  - $\mathrm{Dec}_{\mathrm{k}}(c)$ (Decryption algorithm)
    - Input: Secret key $\mathrm{k} \in \mathcal{K}$ and a ciphertex c
    - Output: a plaintext message $\mathrm{m} \in \mathcal{M}$

Typically picks $\mathrm{k} \in \mathcal{K}$ uniformly at random

Trusted Parties (e.g., Alice and Bob) must run Gen in advance to obtain secret k.

Assumption: Adversary does not get to see output of Gen

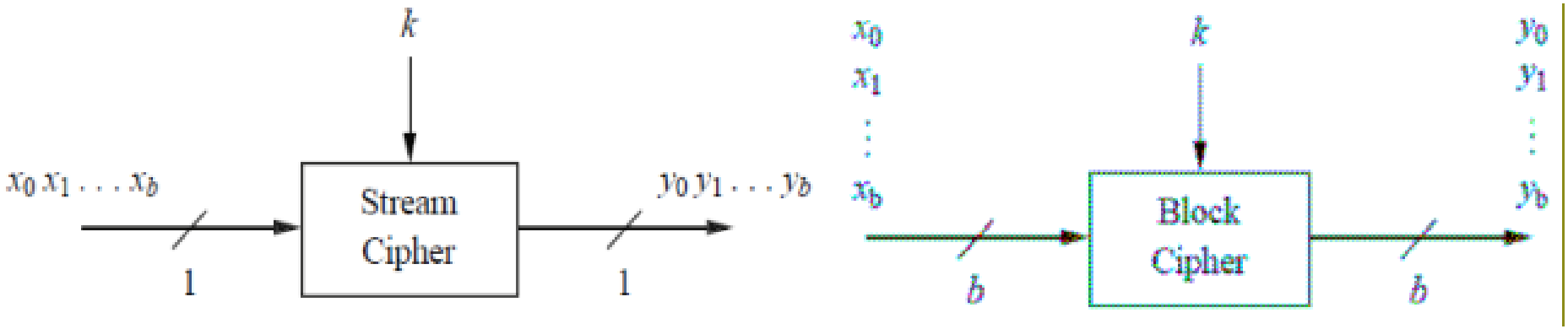
- Invariant: $\mathrm{Dec}_{\mathrm{k}}(\mathrm{Enc}_{\mathrm{k}}(\mathrm{m}))=\mathrm{m}$

Shannon's Theorem

Theorem: Let (Gen,Enc,Dec) be an encryption scheme with $|\mathcal{K}| = |\mathcal{M}| = |\mathcal{C}|$. Then the scheme is perfectly secret if and only if:

1. Every key k ∈ $\mathcal{K}$ is chosen with (equal) probability $1/|\mathcal{K}|$ by the algorithm Gen, and
2. For every m ∈ $\mathcal{M}$ and every c ∈ $\mathcal{C}$ there exists a unique key k ∈ $\mathcal{K}$ such that $Enc_k(m)=c$.

Stream Cipher vs. Block Cipher



- **Stream Ciphers**
  - Encrypt bits individually
  - Usually small and fast → common in embedded devices (e.g., A5/1 for GSM phones)
- **Block Ciphers:**
  - Always encrypt a full block (several bits)
  - Are common for Internet applications

DES Facts

- Data Encryption Standard (DES) encrypts blocks of size 64 bit.
- Developed by IBM based on the cipher *Lucifer* under influence of the *National Security Agency* (NSA), the design criteria for DES have not been published
- Standardized 1977 by the National Bureau of Standards today called *National Institute of Standards and Technology* (NIST)
- Most popular block cipher for most of the last 30 years.
- By far best studied symmetric algorithm.
- Nowadays considered insecure due to the small key length of 56 bit.
- But: 3DES yields very secure cipher, still widely used today.
- Replaced by the *Advanced Encryption Standard* (AES) in 2000

Block Cipher Primitives: Confusion and Diffusion

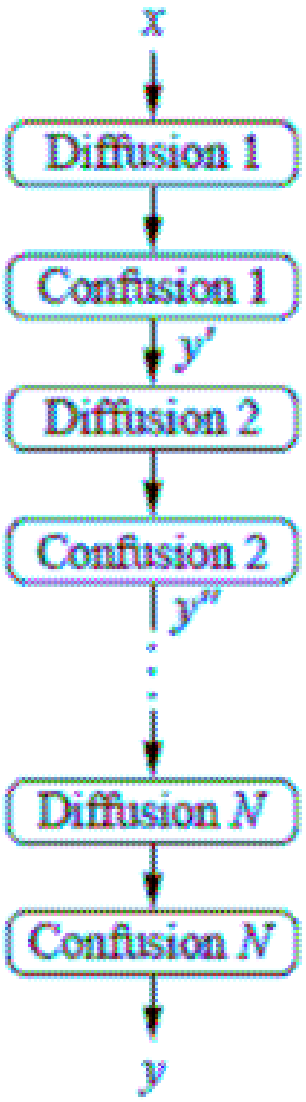
- **Shannon: There are two primitive operations with which strong encryption algorithms can be built:**

1. **Confusion:** **An encryption operation where the** relationship between key and ciphertext is obscured.
   Today, a common element for achieving confusion is substitution, **which is found in both AES and DES.**

2. **Diffusion:** **An encryption operation where the** influence of one plaintext symbol is spread over many ciphertext symbols **with the goal of hiding statistical properties of the plaintext.**
   A simple diffusion element is the bit permutation, **which is frequently used within DES.**

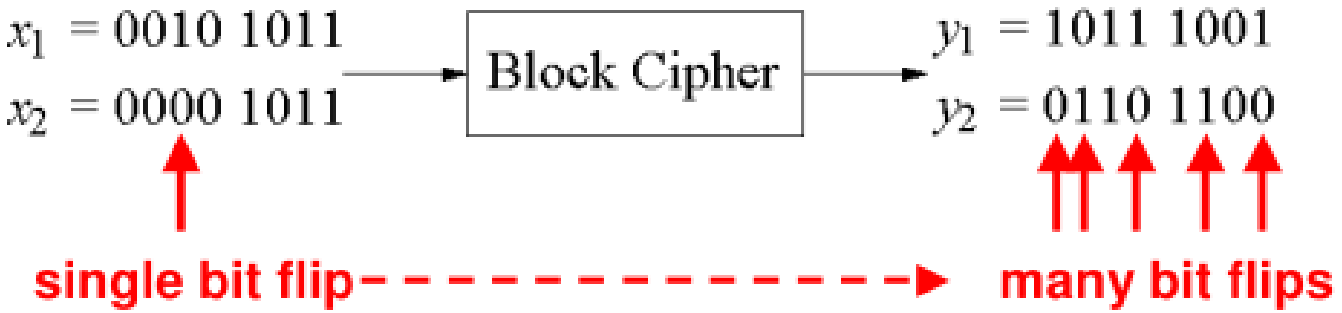
- **Both operations by themselves cannot provide security. The idea is to concatenate confusion and diffusion elements to build the so called *product ciphers*.**

Product Ciphers

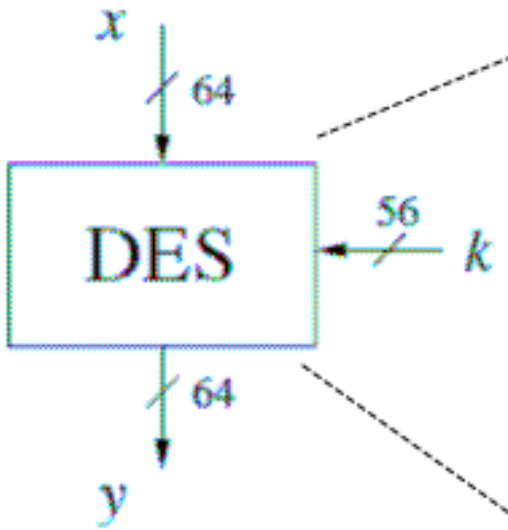


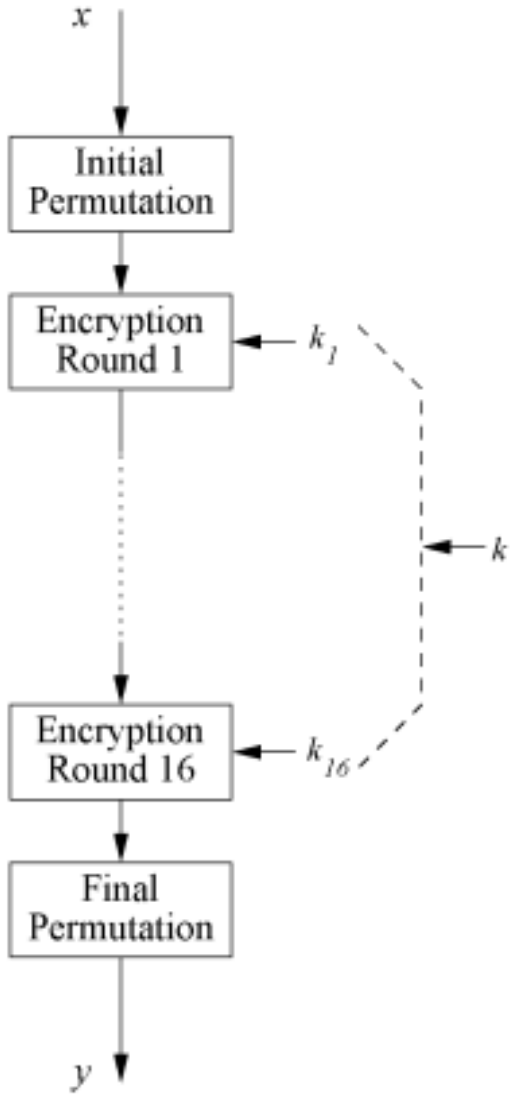
- Most of today's block ciphers are *product ciphers* as they consist of rounds which are applied repeatedly to the data.
- Can reach excellent diffusion: changing of one bit of plaintext results *on average* in the change of half the output bits.

Example:



Overview of DES

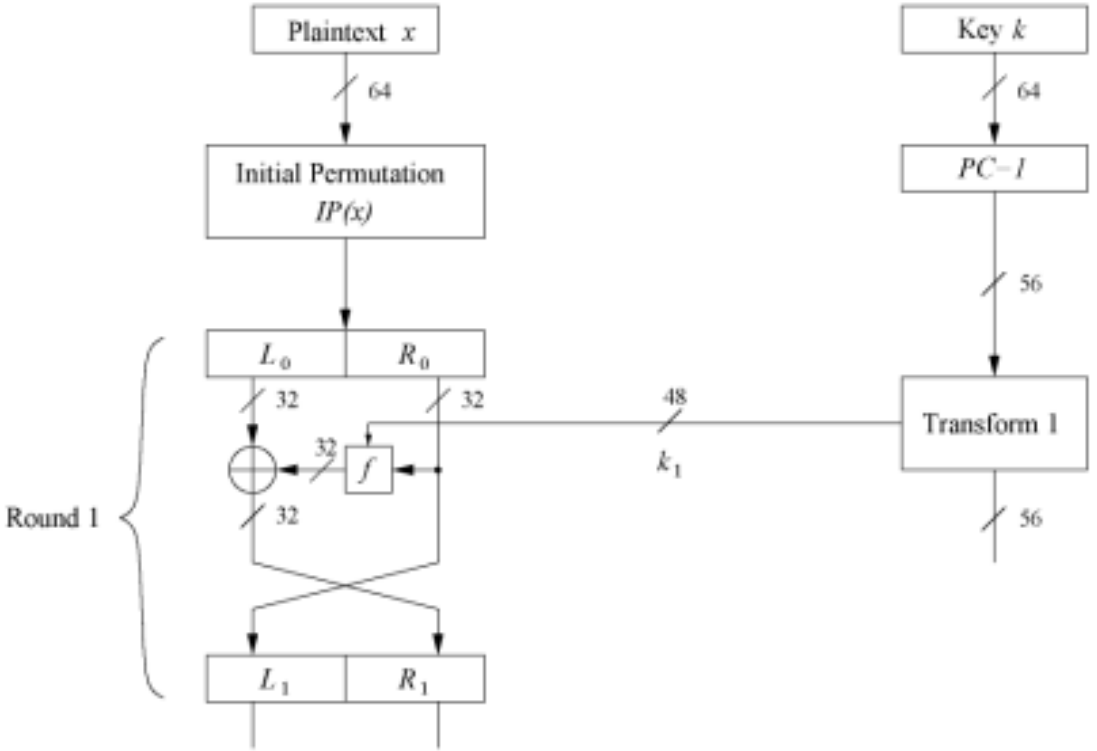




- Encrypts blocks of size 64 bits.
- Uses a key of size 56 bits.
- **Symmetric cipher: uses same key for encryption and decryption**
- **Uses 16 rounds performing identical operation**
- **Different subkey in each round derived from main key**

The DES Feistel Network (1)

- DES structure is a *Feistel network*
- Advantage: encryption and decryption differ only in keyschedule

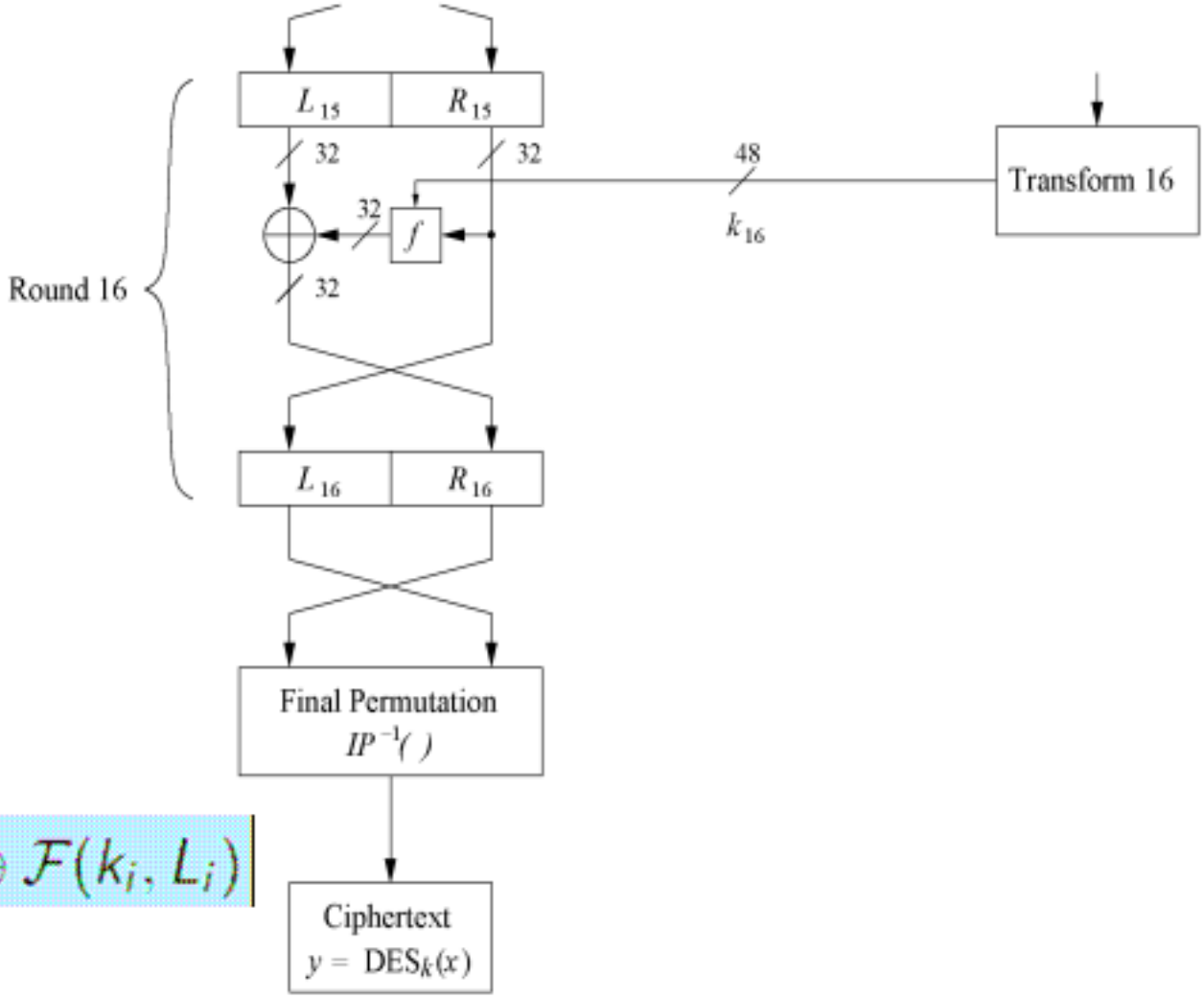


- Bitwise initial permutation, then 16 rounds
  1. Plaintext is split into 32-bit halves L_i and R_i
  2. R_i is fed into f, the output of which is then XORed with L_i
  3. Left and right half are swapped
- Rounds can be expressed as:

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

The DES Feistel Network (2)

- L and R swapped again at the end of the cipher, i.e., after round 16 followed by a final permutation
  - Permutations do not add security, only simplify layout



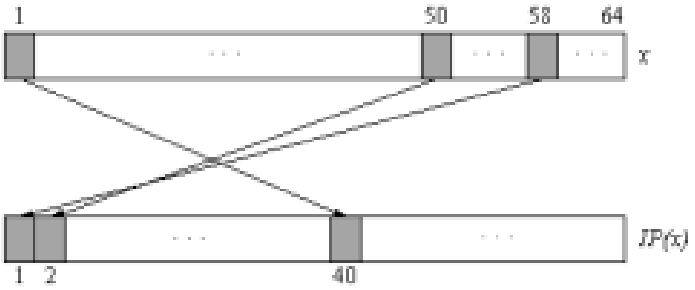
$$R_{i-1} = L_i, L_{i-1} = R_i \oplus \mathcal{F}(k_i, L_i)$$

Initial and Final Permutation

- Bitwise Permutations.
- Inverse operations.
- Described by tables IP and IP^{-1}.

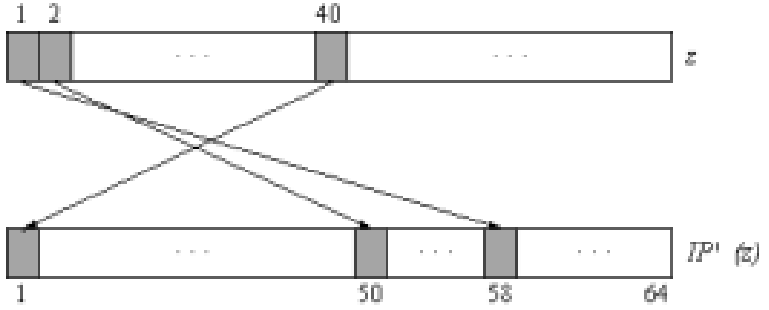
Initial Permutation

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7



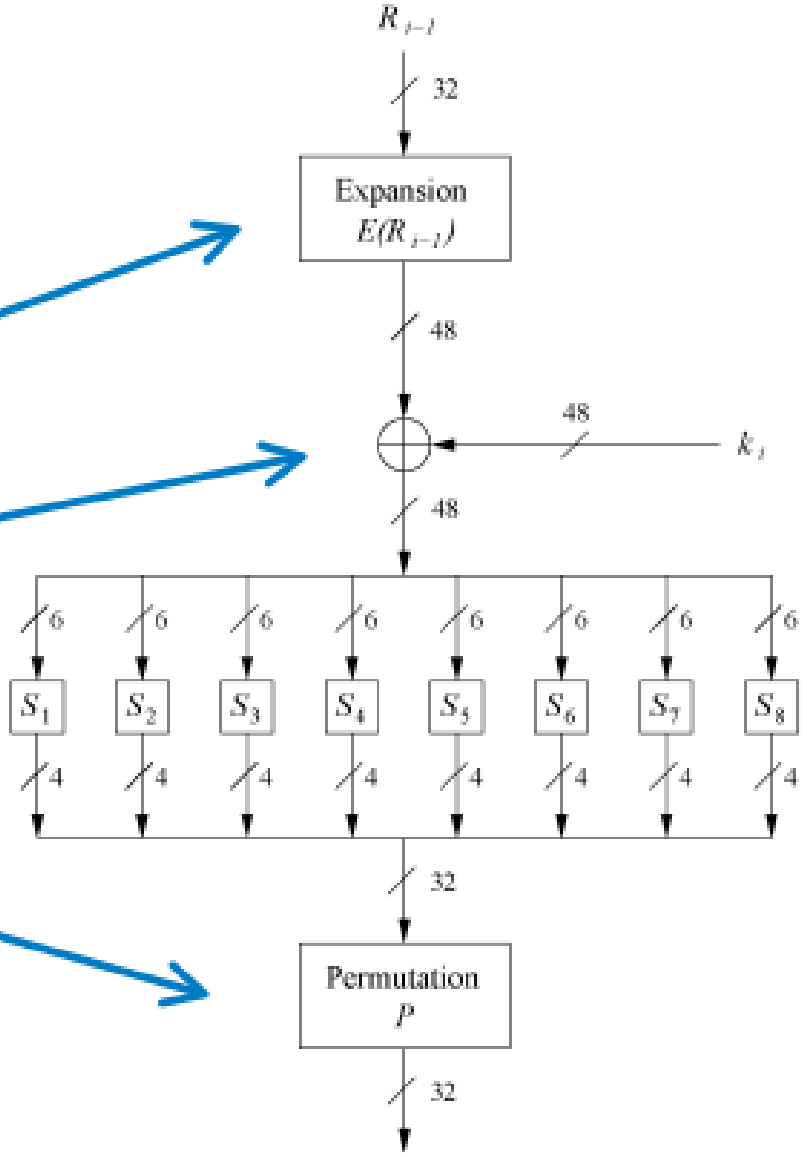
Final Permutation

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25



The f Function

- main operation of DES
- *f*-Function inputs: R_{i-1} and round key k_i
- 4 Steps:
  1. Expansion *E*
  2. XOR with round key
  3. S-box substitution
  4. Permutation

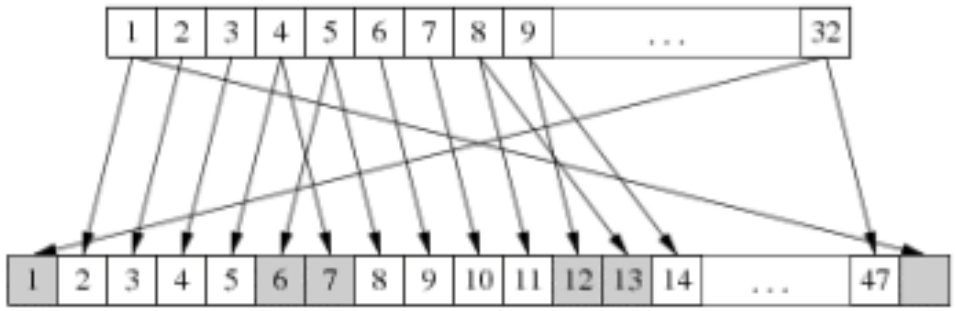


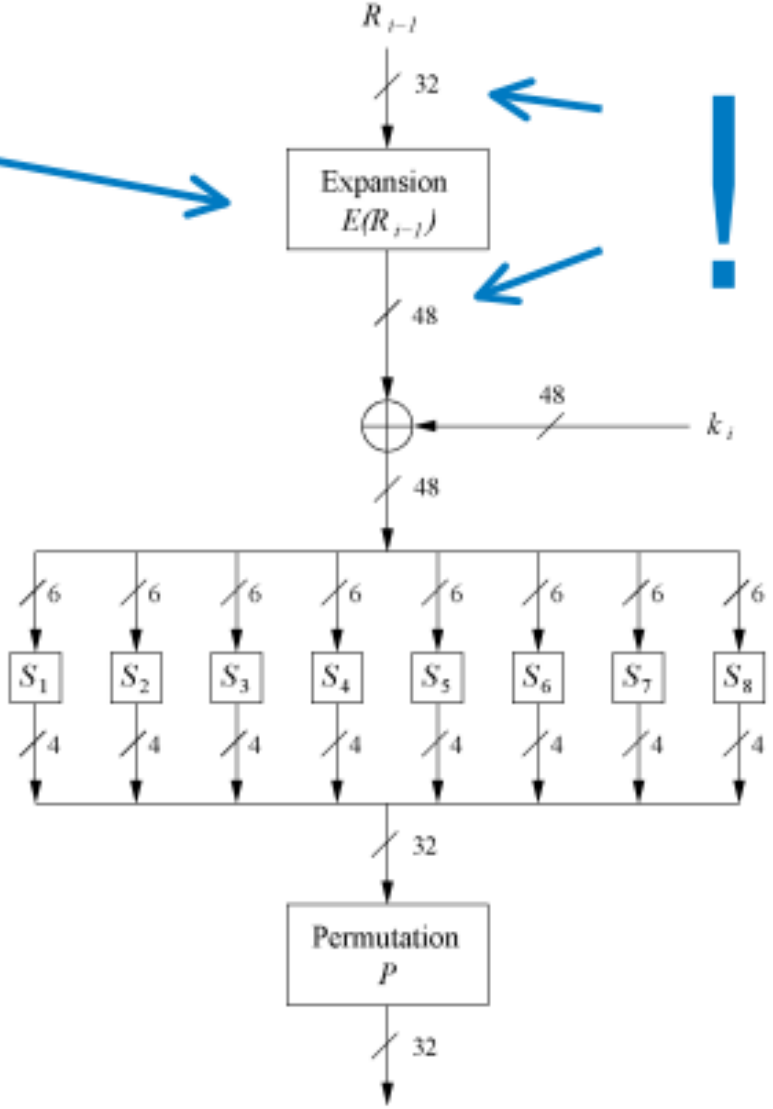
The Expansion Function E

1. Expansion *E*

- main purpose: increases diffusion

E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

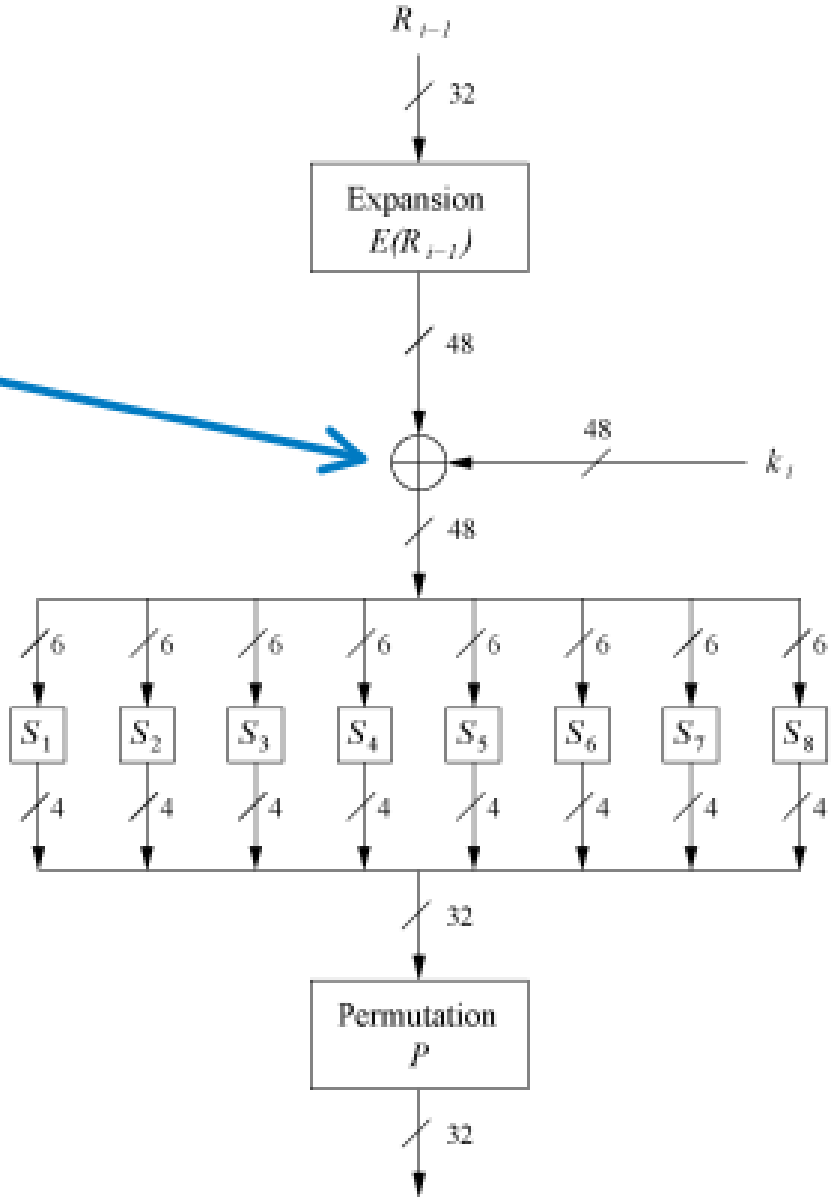




Add Round Key

2. XOR Round Key

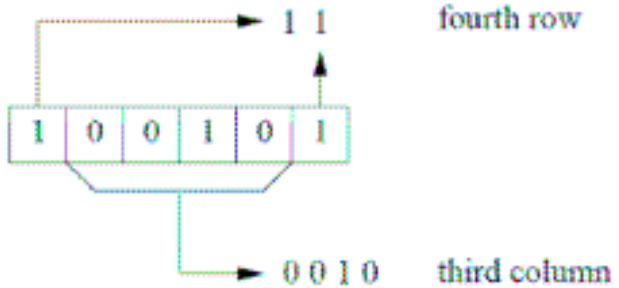
- Bitwise XOR of the round key and the output of the expansion function *E*
- Round keys are derived from the main key in the DES key schedule (in a few slides)

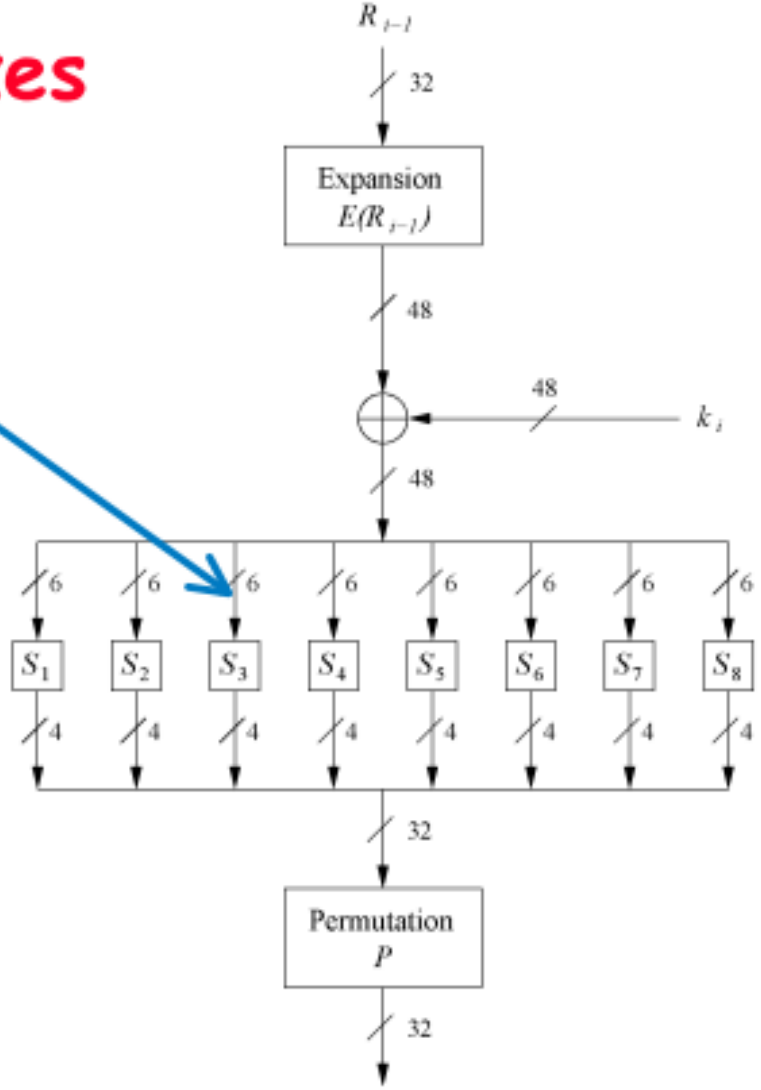


The DES S-Boxes

3. S-Box substitution

- Eight substitution tables.
- 6 bits of input, 4 bits of output.
- **Non-linear** and resistant to differential cryptanalysis.
- Crucial element for DES security!





S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

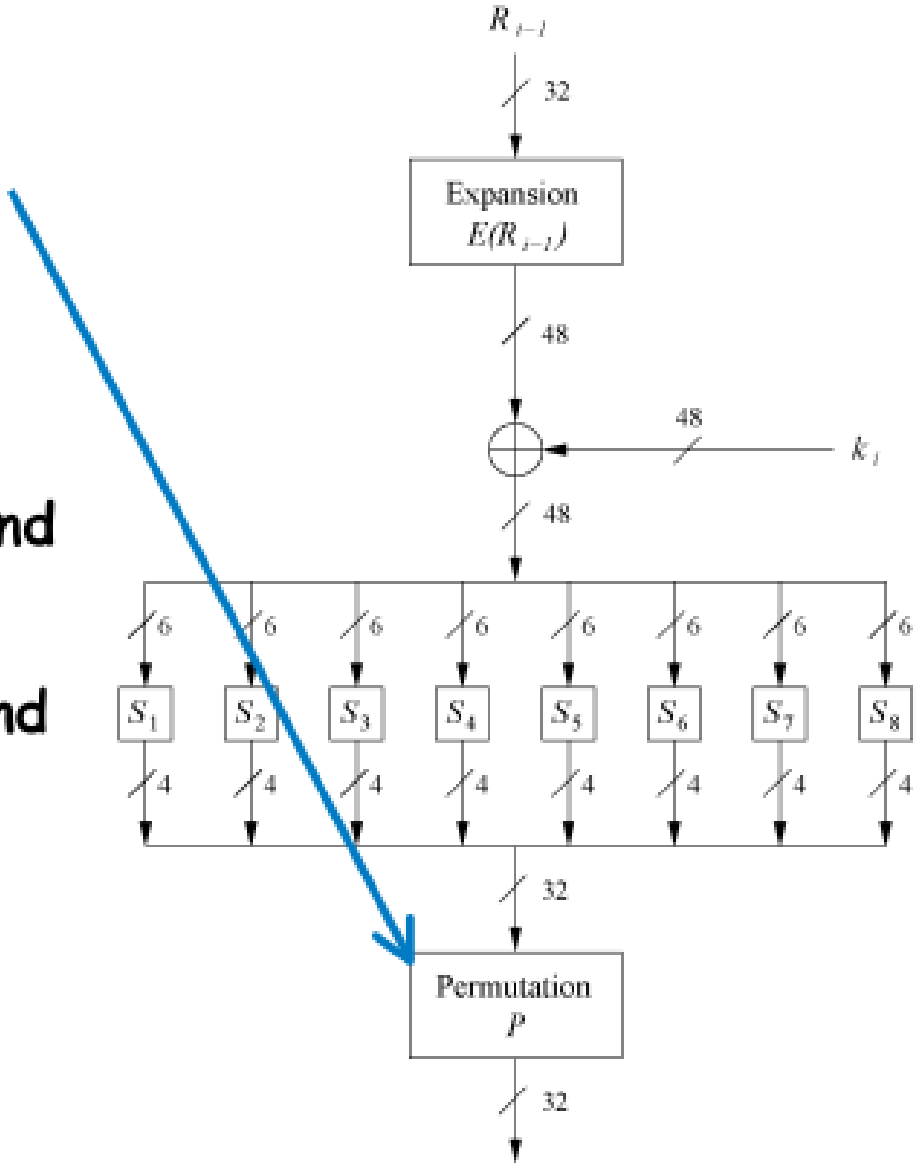
S_8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	02	08	04	06	15	11	01	10	09	03	14	05	00	12	07
1	01	15	13	08	10	03	07	04	12	05	06	11	00	14	09	02
2	07	11	04	01	09	12	14	02	00	06	10	13	15	03	05	08
3	02	01	14	07	04	10	08	13	15	12	09	00	03	05	06	11

The Permutation P

4. Permutation P

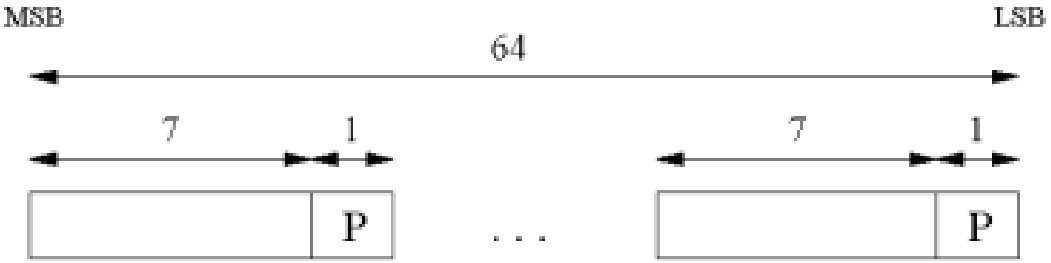
- Bitwise permutation.
- Introduces diffusion.
- Output bits of one S-Box effect several S-Boxes in next round
- Diffusion by E, S-Boxes and P guarantees that after Round 5 every bit is a function of each key bit and each plaintext bit.

P							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25



Key Schedule (1)

- Derives 16 round keys (or *subkeys*) k_i of 48 bits each from the original 56 bit key.
- The input key size of the DES is 64 bit: 56 bit key and 8 bit parity:



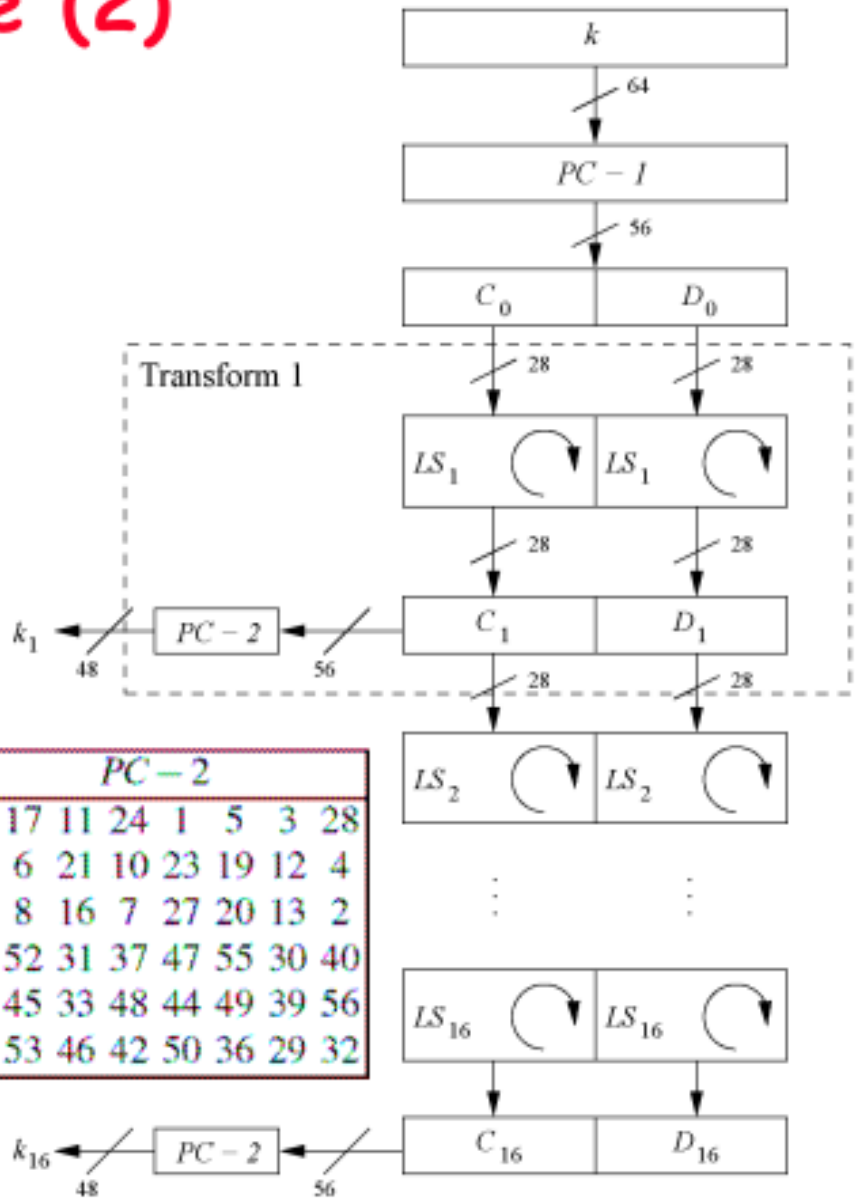
P = parity bit

- Parity bits are removed in a first permuted choice *PC-1*:
  (note that the bits 8, 16, 24, 32, 48, 56 and 64 are not used at all)

PC – 1							
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	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

Key Schedule (2)

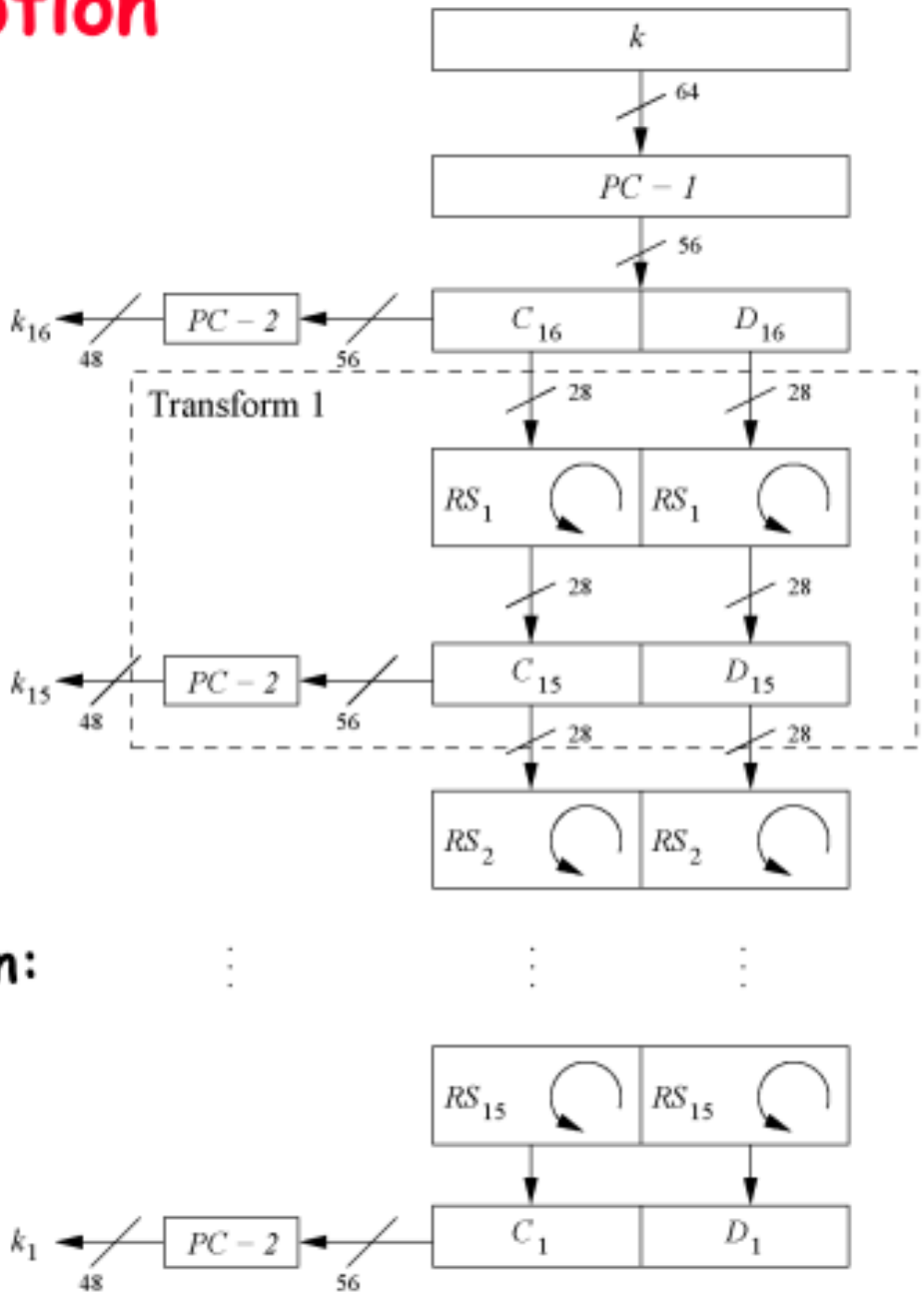
- **Split key into 28-bit halves C_0 and D_0.**
- **In** rounds $i = 1, 2, 9, 16$, **the two halves are each rotated left by** one bit.
- **In** all other rounds **the two halves are each rotated left by** two bits.
- ***In each round i permuted choice*** *PC-2* **selects a permuted subset of 48 bits of C_i and D_i as round key k_i, i.e.** each k_i is a permutation of k
- Note: **The total number of rotations:**
  $4 \times 1 + 12 \times 2 = 28 \Rightarrow D_0 = D_{16}$ and $C_0 = C_{16}$



PC – 2							
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

Decryption

- In Feistel ciphers only the keyschedule has to be modified for decryption.
- Generate the same 16 round keys in reverse order. (for a detailed discussion see Ch. 3 *Understanding Crptography*)
- Reversed key schedule:
  As $D_0 = D_{16}$ and $C_0 = C_{16}$ the first round key can be generated by applying *PC-2* right after *PC-1* (no rotation here).
  All other rotations of *C* and *D* can be reversed to reproduce the other round keys resulting in:
  - No rotation in round 1.
  - One bit rotation to the right in rounds 2, 9 and 16.
  - Two bit rotations to the right in all other rounds.



Properties of DES

- Complementation: $\mathrm{DES}(k, m) = \overline{\mathrm{DES}(\overline{k}, \overline{m})}$
- Key must be randomly chosen
- „Weak" keys: encryption=decryption

$$\mathrm{DES}(k, \mathrm{DES}(k, m)) = m, \ \forall\ m \in \mathcal{M}$$

- „Semi-weak" keys:

$$\mathrm{DES}(k, \mathrm{DES}(k', m)) = m, \ \forall\ m \in \mathcal{M}$$

Group_1
0x 0101010101010101
0x FEFEFEFEFEFEFEFE
0x E0E0E0E0F1F1F1F1
0x 1F1F1F1F0E0E0E0E

⟨ 0x 011F011F010E010E, 0x 1F011F010E010E01 ⟩
⟨ 0x 01E001E001F101F1, 0x E001E001F101F101 ⟩
⟨ 0x 01FE01FE01FE01FE, 0x FE01FE01FE01FE01 ⟩
⟨ 0x 1FE01FE00EF10EF1, 0x E01FE01FF10EF10E ⟩
⟨ 0x 1FFE1FFE0EFE0EFE, 0x FE1FFE1FFE0EFE0E ⟩
⟨ 0x E0FEE0FEF1FEF1FE, 0x FEE0FEE0FEF1FEF1 ⟩

Security of DES

- After proposal of DES two major criticisms arose:
  1. Key space is too small (2^{56} keys)
  2. S-box design criteria have been kept secret: Are there any hidden analytical attacks (*backdoors*), only known to the NSA?
- Analytical Attacks: DES is highly resistent to both *differential* and *linear cryptanalysis*, which have been published years later than the DES. This may mean that IBM and NSA had been aware of these attacks for 15 years.
  So far there is no known analytical attack which breaks DES in realistic scenarios.
- Exhaustive key search: For a given pair of plaintext-ciphertext (x, y) test all 2^{56} keys until the condition $DES_k^{-1}(x)=y$ is fulfilled.
  $\Rightarrow$ Relatively easy given today's computer technology

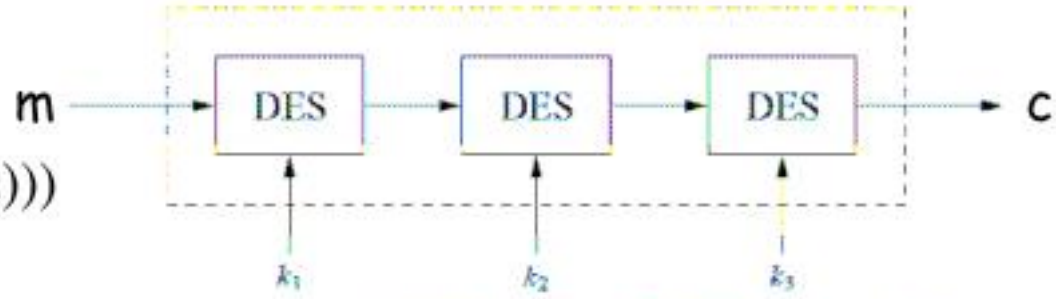
History of Attacks on DES

Year	Proposed/ implemented DES Attack
1993	Wiener proposes design of a very efficient key search machine: Average search requires 36h. Cost: $1M
1993	Matsui proposes linear cryptanalysis (2^{43} ciphertexts)
1997	DES Challenge I broken, 4.5 months of distributed search
1998	DES Challenge II-(1) broken, 39 days (distributed search)
Jul. 1998	DES Challenge II-(2) broken, key search machine *Deep Crack*: 1800 ASICs with 24 search engines each, Costs: $250,000 15 days average search time
Jan. 1999	DES Challenge III broken in 22h 15min (distributed search assisted by *Deep Crack*)
2006-2008	Reconfigurable key search machine *COPACOBANA* (Germany), uses 120 FPGAs to break DES in 6.4 days (avg.) at a cost of $10,000.

Triple DES – 3DES3

- Triple encryption using DES is often used in practice to extend the effective key length of DES to either 168 or 112 (if $k_3=k_1$).
  - Became standard in 1998

$$c = DES(k_3, DES(k_2, DES(k_1, m)))$$



- Why not use 2DES with 112-bit key?

$$c = \mathrm{DES}(k_1, \mathrm{DES}(k_2, m))$$

$$\mathrm{DES}^{-1}(k_1, c) = \mathrm{DES}(k_2, m)$$

- A „meet-in-the-middle" attack is possible:
  - For all 2^{56} possible values x of k_2 calculate A_i=DES(x,m)
  - For all 2^{56} possible values y of k_1 calculate $B_j = DES^{-1}(y, c)$
  - Compare A_i to B_j and find k_1 and k_2
  - Large storage space but almost same complexity as DES

Alternate Triple DES

- Alternative version of *3DES:* $c = \mathrm{DES}(k_1, \mathrm{DES}^{-1}(k_2, \mathrm{DES}(k_3, m)))$
  *A*dvantage: choosing $k_1 = k_2 = k_3$ performs single DES encryption.
- No practical attack known today for both versions of 3DES3 and even 3DES2.
- Used in many legacy applications, i.e., in banking systems.
- 3DES3 vs 2DES2: 3DES3 with $k_1 \neq k_2 \neq k_3$ can be subject to „meet-in-the-middle" attack $\mathrm{DES}^{-1}(k_1, c) = \mathrm{DES}^{-1}(k_2, \mathrm{DES}(k_3, m))$
- Calculate and store 2^{112} values of $\mathrm{DES}^{-1}(k_2, \mathrm{DES}(k_3, m))$
- 3DES2 is preferred $\mathrm{DES}(k_1, \mathrm{DES}^{-1}(k_2, \mathrm{DES}(k_1, m)))$
- Another variation DES-X: $k_2 \oplus \mathrm{DES}(k, m \oplus k_1)$
- 56-bit k and 64-bit k_1 and k_2

Alternatives to DES

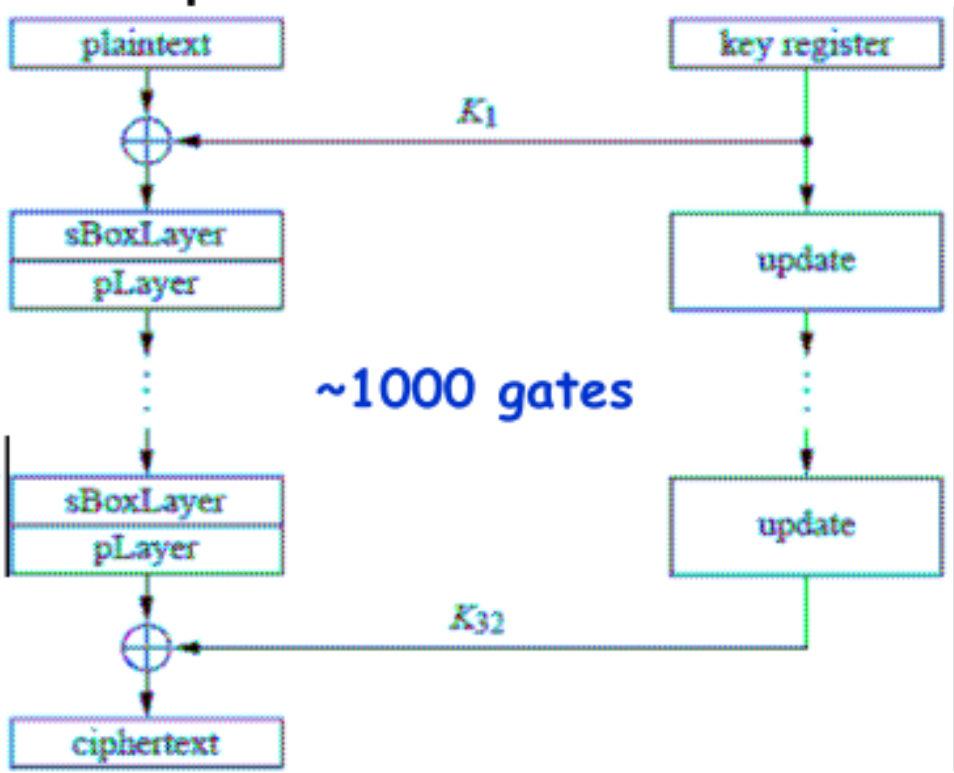
Algorithm	Data Bits	key lengths	remarks
AES / Rijndael	128	128/192/256	DES ''replacement'', worldwide used standard
Triple DES	64	112 (effective)	conservative choice
Mars	128	128/192/256	AES finalist
RC6	128	128/192/256	AES finalist
Serpent	128	128/192/256	AES finalist
Twofish	128	128/192/256	AES finalist
IDEA	64	128	patented

Lightweight alternative - *PRESENT*

- An example of a new block cipher for power constrained devices (e.g., RFID tags)
- Data block=64, key=80,128
- 31 rounds using substitution-permutation (SP) network
- A single 4 to 4 S-Box
- Permutation:

$$P(i) = \begin{cases} i \cdot 16 \mod 63, & i \in \{0,\ldots,62\} \\ 63, & i = 63. \end{cases}$$

- Key schedule (64 MSBs):
- 1. Rotate left 61 positions
- 2. S-Box to leftmost 4 bits
- 3. XOR 5-bit round-counter



x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

1. $[k_{79}k_{78}\ldots k_1k_0] = [k_{18}k_{17}\ldots k_{20}k_{19}]$
2. $[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$
3. $[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus \texttt{round\_counter}$

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

Data Encryption Standard (video)

The Advanced Encryption Standard (AES)

Encryption with Block Ciphers: **Modes of Operation**

- ~~*Electronic Code Book mode (ECB)*~~
- *Cipher Block Chaining mode (CBC)*
- *Output Feedback mode (OFB)*
- *Cipher Feedback mode (CFB)*
- *Counter mode (CTR)*
- *Galois Counter Mode (GCM)*

Block Ciphers

- A block cipher is much more than just an encryption algorithm, it can be used ...
  - to build different types of block-based encryption schemes
  - to realize stream ciphers
  - to construct hash functions
  - to make message authentication codes
  - to build key establishment protocols
  - to make a pseudo-random number generator
  - ...
- The security of block ciphers also can be increased by
  - key whitening
  - multiple encryption

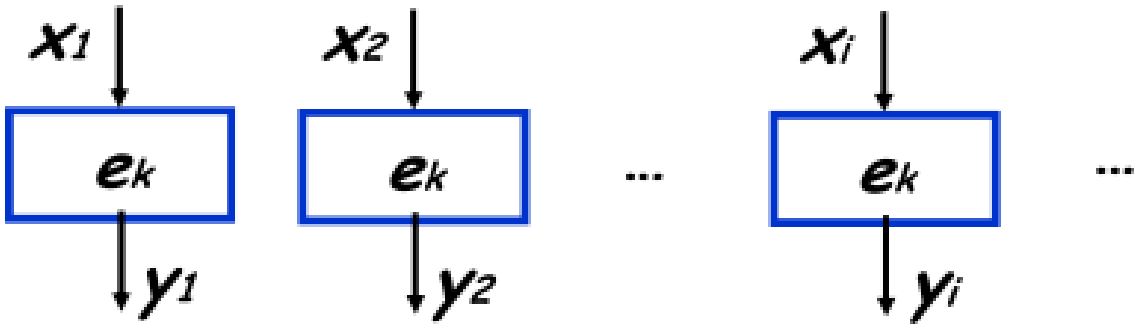
Encryption with Block Ciphers

- There are several ways of encrypting long plaintexts, e.g., an e-mail or a computer file, with a block cipher ("modes of operation")

- These modes of operation have three goals:
  - In addition to confidentiality, some of them provide authenticity and integrity:
    - » Is the message really coming from the original sender? (authenticity)
    - » Was the ciphertext altered during transmission? (integrity)

Electronic Code Book mode (ECB)

- $e_k(x_i)$ = encryption of a b-bit plaintext block x_i with key k
- $e_k^{-1}(y_i)$ = decryption of b-bit ciphertext block y_i with key k
- Messages which exceed b bits are partitioned into b-bit blocks
- Each Block is encrypted separately



- Padding of last block:
  - Include non-data values (e.g., Null)
  - Include number of bytes in padding
  - And/or number of plaintext bytes

ECB: advantages/disadvantages

- Advantages
  - no block synchronization between sender and receiver is required
  - bit errors caused by noisy channels only affect the corresponding block but not succeeding blocks
  - Block cipher operations can be parallelized
    - » advantage for high-speed implementations
- Disadvantages
  - ECB encryption highly deterministic
    - » identical plaintexts result in identical ciphertexts
    - » an attacker recognizes if the same message has been sent twice
    - » plaintext blocks are encrypted independently of previous blocks
      - an attacker may reorder ciphertext blocks which results in valid but incorrect plaintext

Substitution Attack on ECB

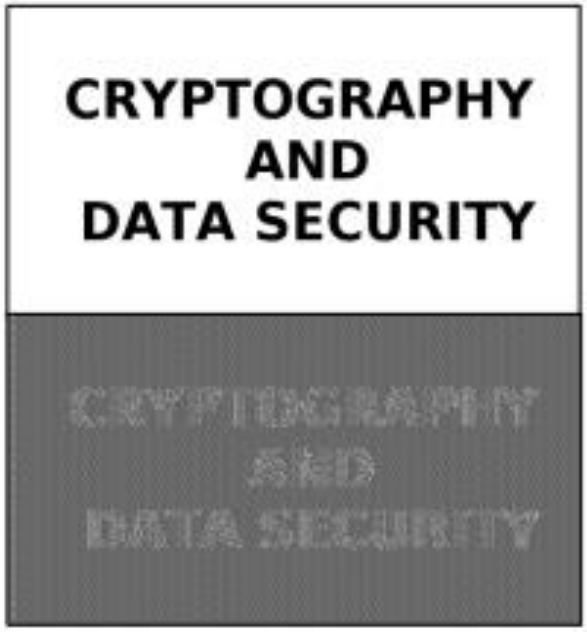
- Once a particular plaintext to ciphertext block mapping $x_i \rightarrow y_i$ is known, a sequence of ciphertext blocks can easily be manipulated
- Suppose an *electronic bank transfer*

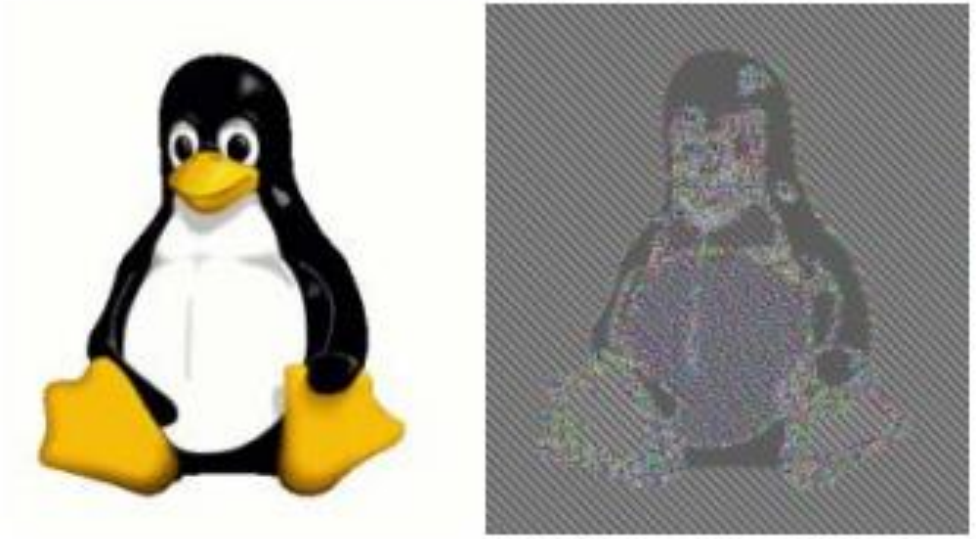
Block #	1	2	3	4	5
	Sending Bank A	Sending Account #	Receiving Bank B	Receiving Account #	Amount $

  - encryption key between banks does not change frequently
  - The attacker sends $1 transfers from his account at bank A to his account at bank B several times
    - » He can check for ciphertext blocks that repeat, and he stores blocks 1,3 and 4 of these transfers
  - He can now replace block 4 of other transfers (having the same blocks 1&3) with the block 4 that he stored before
    - » *all transfers* from some account of bank A to some account of bank B are redirected to go into the attacker's B account
    - » Integrity violation – does not break the cipher

Example of encrypting bitmaps in ECB mode

- Identical plaintexts are mapped to identical ciphertexts





- Statistical properties in the plaintext are preserved in the ciphertext

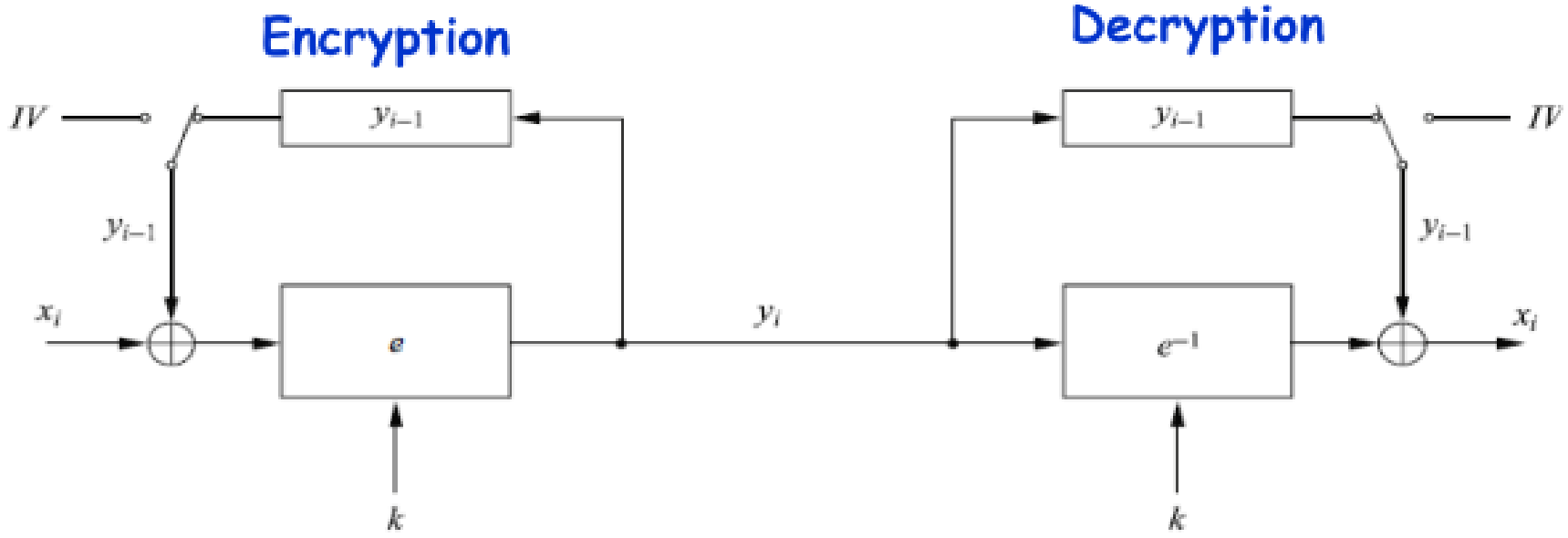
Cipher Block Chaining mode (CBC)

- There are two main ideas behind the CBC mode:
  - The encryption of all blocks are "chained together"
    - » ciphertext y_i depends not only on block x_i but on all previous plaintext blocks as well
  - The encryption is randomized by using an initialization vector (IV)

Encryption (first block):	$y_1 = e_k(x_1 \oplus \mathrm{IV})$
Encryption (general block):	$y_i = e_k(x_i \oplus y_{i-1}), \quad i \geq 2$
Decryption (first block):	$x_1 = e_k^{-1}(y_1) \oplus \mathrm{IV}$
Decryption (general block):	$x_i = e_k^{-1}(y_i) \oplus y_{i-1}, \quad i \geq 2$

Cipher Block Chaining mode (CBC)

- For the 1st plaintext block x_1 there is no previous ciphertext
  - an IV is added to the first plaintext to make each CBC encryption nondeterministic
  - the first ciphertext y_1 depends on plaintext x_1 and the IV
- The 2nd ciphertext y_2 depends on the IV, x_1 *and* x_2
- The 3rd ciphertext y_3 depends on the IV and x_1, x_2 *and* x_3, and so on



Substitution Attack on CBC

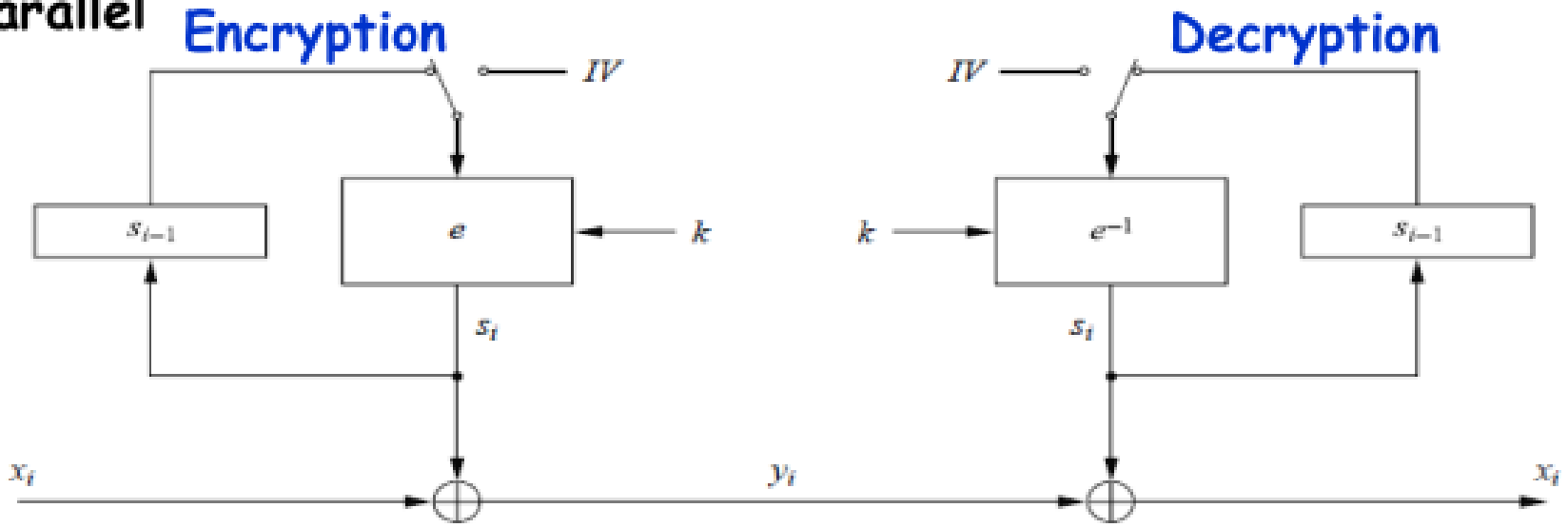
- Consider the last example:

Block #	1	2	3	4	5
	Sending Bank A	Sending Account #	Receiving Bank B	Receiving Account #	Amount $

- If the IV is properly chosen for every wire transfer, the attack will not work at all
- If the IV is kept the same for several transfers, the attacker would recognize the transfers from his account at bank A to bank B
- If we choose a new IV every time we encrypt, the CBC mode becomes a probabilistic encryption scheme, i.e., two encryptions of the same plaintext look entirely different
- It is not needed to keep the IV *secret* - the IV should be a non-secret nonce (value used only once)
- Integrity can still be violated (e.g., replacing blocks 4&5)
- Can not be encrypted in parallel, decryption?
- If a bit in a block is flipped in transmission all remaining blocks are garbled

Output Feedback mode (OFB)

- Used to build a *synchronous* stream cipher from a block cipher
- Key stream is not generated bitwise but in a block-wise fashion
- The cipher output generates key stream bits S_i
- Key stream independent of ciphertext – can be generated in parallel



Encryption (first block): $s_1 = e_k(\mathrm{IV})$ *and* $y_1 = s_1 \oplus x_1$

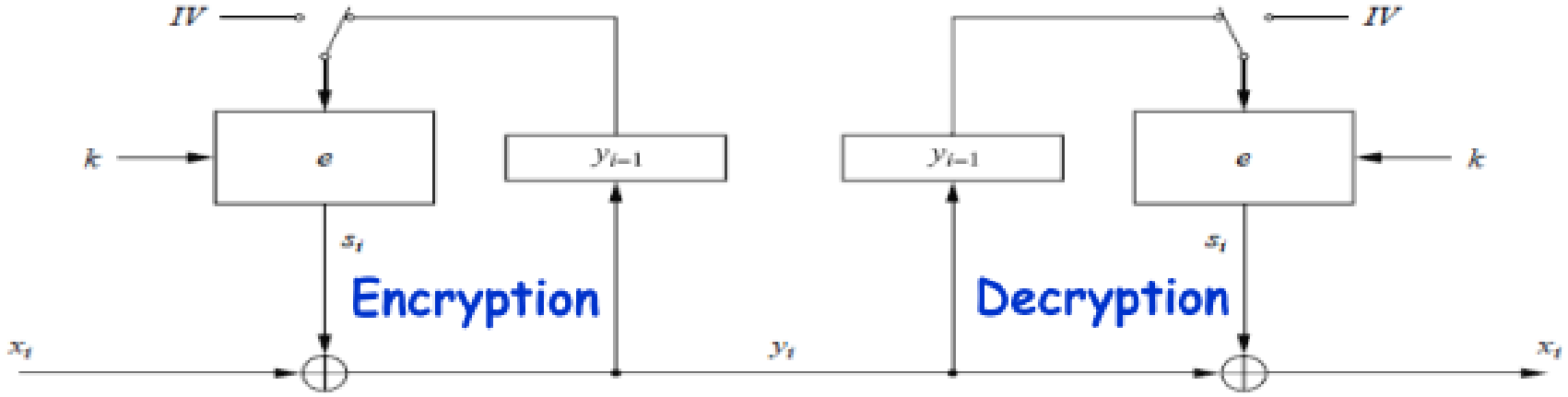
Encryption (general block): $s_i = e_k(s_{i-1})$ *and* $y_i = s_i \oplus x_i, \quad i \geq 2$

Decryption (first block): $s_1 = e_k(\mathrm{IV})$ *and* $x_1 = s_1 \oplus y_1$

Decryption (general block) : $s_i = e_k(s_{i-1})$ *and* $x_i = s_i \oplus y_i, \quad i \geq 2$

Cipher Feedback mode (CFB)

- Uses a block cipher as a building block for asynchronous stream cipher (similar to OFB mode), better name: "Ciphertext Feedback Mode"
- Key stream S_i generated in a block-wise fashion and is also a function of the ciphertext
- By using IV, CFB encryption is also nondeterministic



Encryption (*first block*): $y_1 = e_k(\mathrm{IV}) \oplus x_1$

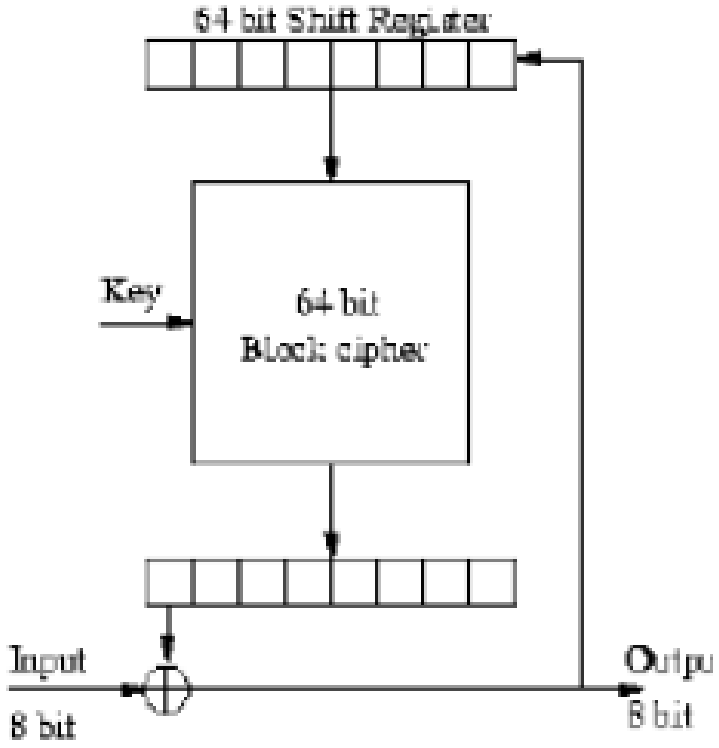
Encryption (*general block*): $y_i = e_k(y_{i-1}) \oplus x_i, \quad i \geq 2$

Decryption (*first block*): $x_1 = e_k(\mathrm{IV}) \oplus y_1$

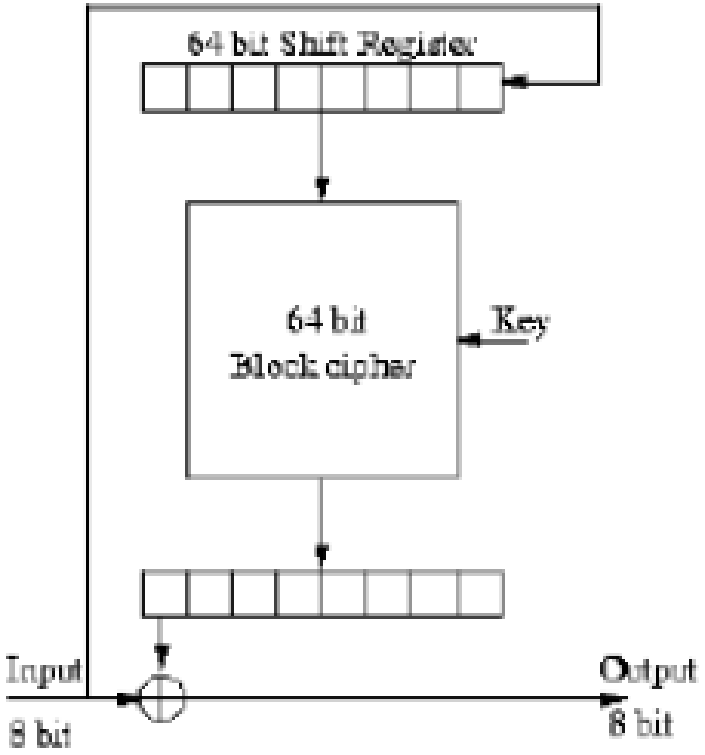
Decryption (*general block*): $x_i = e_k(y_{i-1}) \oplus y_i, \quad i \geq 2$

- Can be used when short plaintext blocks (e.g., byte) are to be encrypted – shift input to the left and use the 8-bit ciphertext

CFB – byte at a a time



Encryption

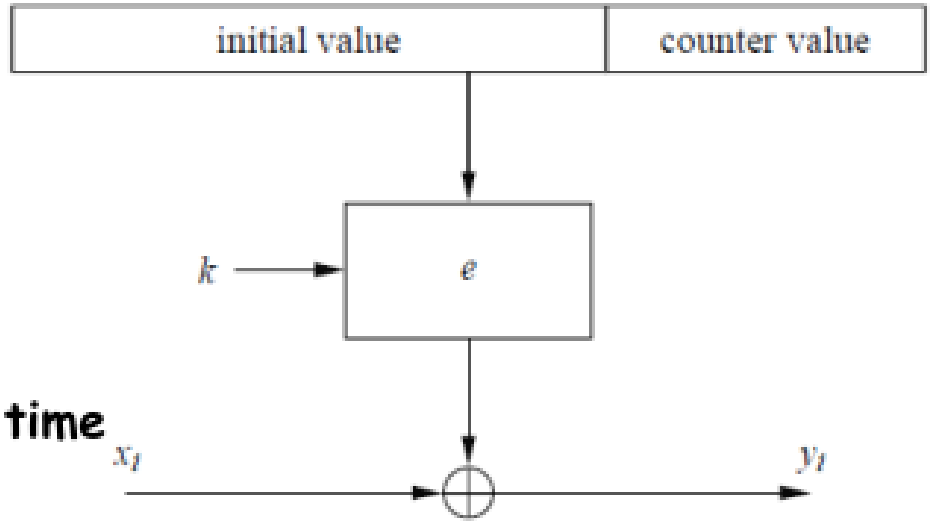


Decryption

- If one ciphertext byte is "lost" all remaining bytes will be decrypted incorrectly.

Counter mode (CTR)

- A block cipher generates a stream cipher (like OFB and CFB)
- The key stream is computed in a block-wise fashion
- The input to the block cipher is a counter which assumes a different value every time the block cipher computes a new key stream block
  - Shorter IV
  - Need not to be replaced every time



- Unlike OFB and CFB modes, the CTR mode can be parallelized since the 2nd encryption can begin before the 1st one has finished
  - Desirable for high-speed implementations, e.g., in network routers

Encryption: $y_i = e_k(\mathrm{IV} \,||\, \mathrm{CTR}_i) \oplus x_i, \quad i \geq 1$

Decryption : $x_i = e_k(\mathrm{IV} \,||\, \mathrm{CTR}_i) \oplus y_i, \quad i \geq 1$

- AES-CTR is the basis for Wi-fi Protected Access WPA2

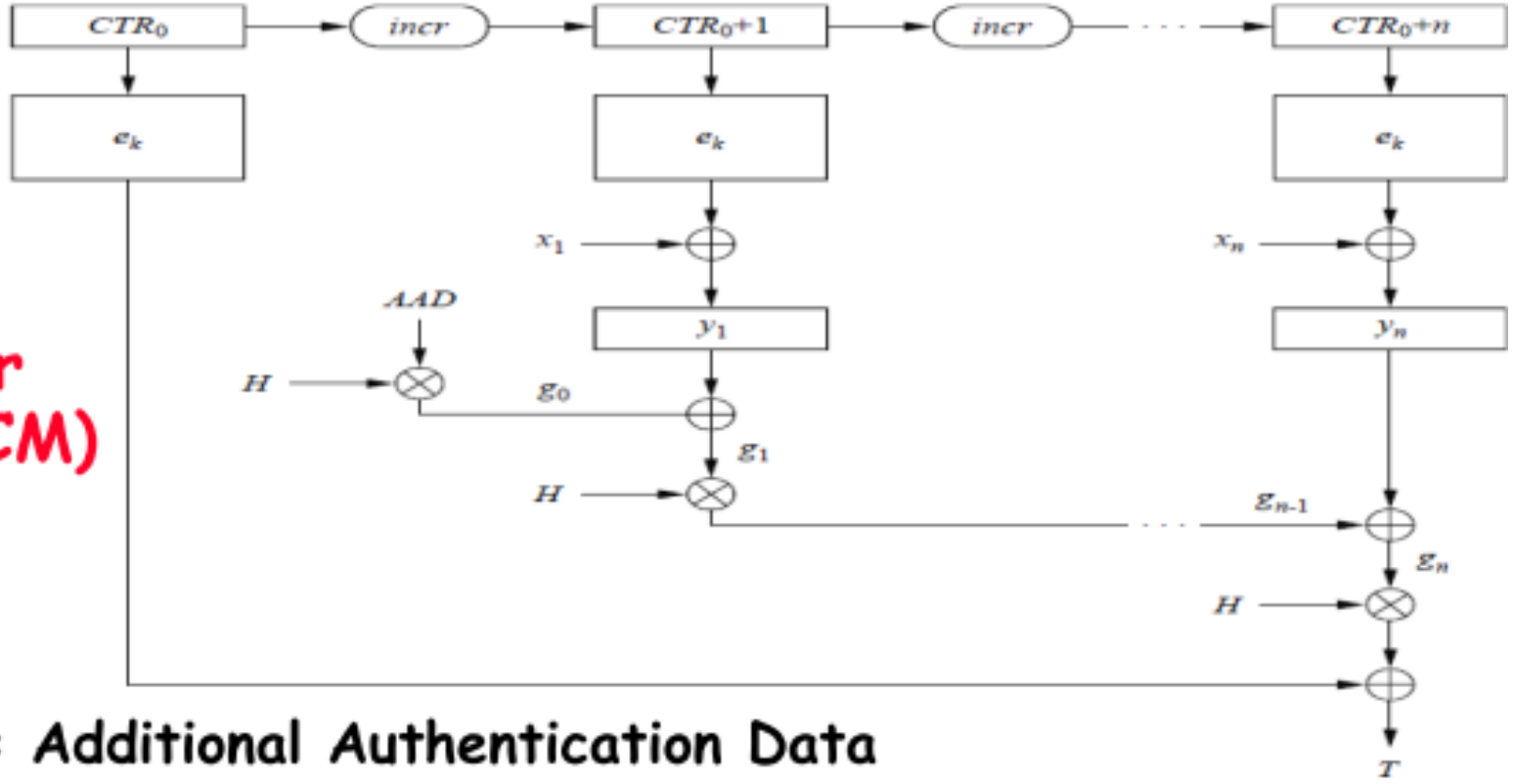
Galois Counter Mode (GCM)

- It also computes a *message authentication code* (MAC), i.e., a cryptographic checksum is computed for a message (see Chapter 12 in *Understanding Cryptography*)
- By making use of GCM, two additional services are provided:
  - Message Authentication
    - » the receiver can make sure that the message was really created by the original sender
  - Message Integrity
    - » the receiver can make sure that nobody tampered with the ciphertext during transmission
- Inputs: plaintext (x_i), IV and AAD = Additional Authentication Data
  - IV and AAD transmitted in the clear
  - AAD can include network address, port, sequence number etc
- Outputs: ciphertext (y_i) and authentication tag T

Galois Counter Mode (GCM)

- For encryption
  - An initial counter is derived from an IV
  - The initial counter value is incremented then encrypted and XORed with the first plaintext block
  - For subsequent plaintexts, the counter is incremented and then encrypted
- For authentication
  - For every plaintext an intermediate authentication parameter g_i is derived
    - » g_i is computed as the XOR of the current ciphertext and the last g_{i-1}, and multiplied by a constant H
      - H is generated by encryption of the zero input with the block cipher
      - Initial g: $g_0 = H \times AAD$ (Galois field multiplication)
  - All multiplications are in the 128-bit Galois field $\mathrm{GF}(2^{128})$ mod the irreducible polynomial $P(x) = x^{128} + x^7 + x^2 + x + 1$

Galois Counter Mode (GCM)



- AAD = Additional Authentication Data

Encryption:

a. Derive a counter value CTR_0 from the IV and compute $\mathrm{CTR}_1 = \mathrm{CTR}_0 + 1$

b. Compute ciphertext: $y_i = e_k(\mathrm{CTR}_i) \oplus x_i, \quad i \geq 1$

Authentication:

a. Generate authentication subkey $H = e_k(0)$

b. Compute $g_0 = AAD \times H$ (Galois field multiplication)

c. Compute $g_i = (g_{i-1} \oplus y_i) \times H, \ 1 \leq i \leq n$ (Galois field multiplication)

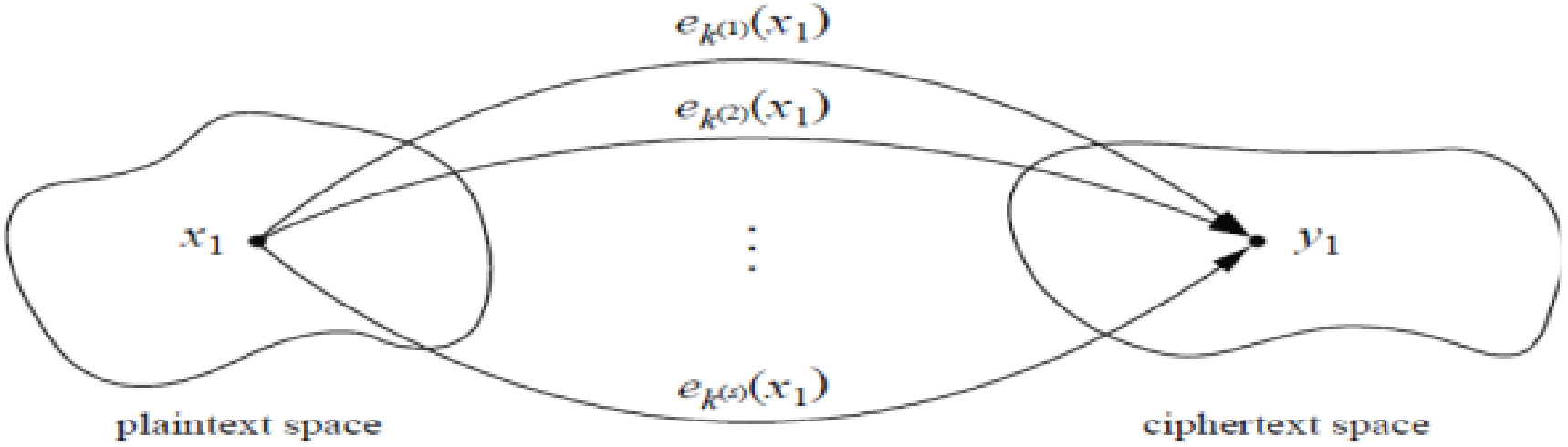
d. Final authentication tag: $T = (g_n \times H) \oplus e_k(\mathrm{CTR}_0)$

Exhaustive Key Search Revisited

- A simple exhaustive search for a DES key knowing one pair (x_1, y_1):

$$DES_k^{(i)}(x_1) \stackrel{?}{=} y_1, \quad i = 0,1,\ldots,2^{56}-1$$

- However, for most other block ciphers a key search is somewhat more complicated
- A brute-force attack can produce *false positive* results



- The likelihood of this is related to the relative size of the key space and the plaintext space
- A brute-force attack is still *possible*, but several pairs of plaintext–ciphertext are needed

An Exhaustive Key Search Example

- Assume cipher block width 64 bit and key size 80 bit
- If we encrypt x_1 under all possible 2^{80} keys, we obtain 2^{80} ciphertexts
  - However, there exist only 2^{64} different ones
- If we run through all keys for a given plaintext–ciphertext pair, we find up to $2^{80}/2^{64} = 2^{16}$ keys that perform the mapping $e_k(x_1) = y_1$

> Given a block cipher with a key length of k bits and block size of n bits, as well as t plaintext–ciphertext pairs $(x_1, y_1), \ldots, (x_t, y_t)$, the expected number of ***false*** keys which encrypt all plaintexts to the corresponding ciphertexts is: 2^{k-tn}

- In this example assuming two plaintext-ciphertext pairs, the likelihood is $2^{80-2\cdot 64} = 2^{-48}$
  - for almost all practical purposes two plaintext-ciphertext pairs are sufficient

Increasing the Security of Block Ciphers

- If we wish to increase the security of block ciphers, e.g., DES available in hardware for legacy reasons
- Two approaches are possible
  - Multiple encryption: theoretically more secure, but in practice may increase the security very little
  - Key whitening

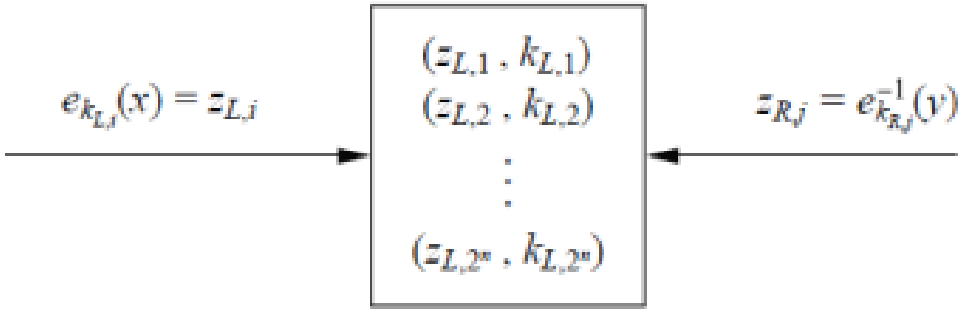
Double Encryption:

- A plaintext x is first encrypted with key k_L, and the resulting ciphertext is encrypted again using a 2nd key k_R

- Assuming a key length of k bits, an exhaustive key search would require $2^k \cdot 2^k = 2^{2k}$ encryptions or decryptions

Meet-in-the-Middle Attack

- A Meet-in-the-Middle attack requires $2^k+2^k = 2^{k+1}$ operations



  - Phase I: for the given (x_1, y_1) the left encryption is brute-forced for all $k_{L,i}$, i=1,2, ..., 2^k and a lookup table with 2^k entry (each $n+k$ bits wide) is computed
    - » the lookup table should be ordered by the result of the encryption ($z_{L,i}$)
  - Phase II: the right encryption is brute-forced (using decryption) and for each $z_{R,i}$ it is checked whether $z_{R,i}$ is equal to any $z_{L,i}$ value in the table of the first phase

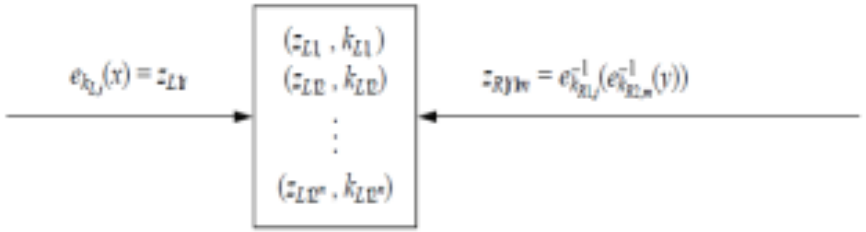
- Computational Complexity

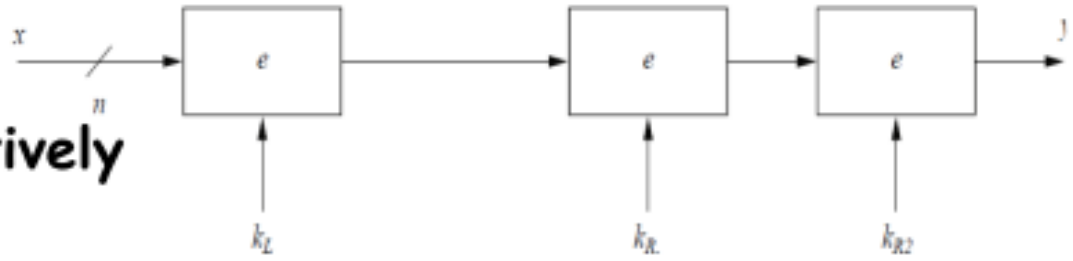
number of encryptions and decryptions = $2^k + 2^k = 2^{k+1}$
number of storage locations = 2^k

- Double encryption is not much more secure then single encryption

Triple Encryption

- Encryption of a block three times $y = e_{k3}(e_{k2}(e_{k1}(x)))$
- In practice a variant scheme is often used EDE (encrypt-decrypt-encrypt) $y = e_{k3}(e^{-1}{}_{k2}(e_{k1}(x)))$
  - Advantage: choosing $k1=k2=k3$ performs single DES encryption
- Still we can perform a meet-in-the middle attack, and it reduces the *effective key length* of triple encryption from $3k$ to $2k$
  - The attacker must run 2^{112} tests in the case of 3DES





- Triple encryption effectively doubles the key length

Cryptanalysis

- **Linear cryptanalysis** – search for an approximated linear relation among bits of plaintext-ciphertext pairs and bits of the key due to imperfect cipher structure
  - DES: 2^{43} known plaintext-ciphertext pairs to find the key (1993)
- **Differential cryptanalysis** – track how differences between two plaintexts affect the corresponding difference between the ciphertexts. Search for non-random behavior
  - DES: 2^{47} plaintext-ciphertext pairs to find the key (1990)

Block Cipher Modes of Operation

Lessons Learned

- DES was the dominant symmetric encryption algorithm from the mid-1970s to the mid-1990s. Since 56-bit keys are no longer secure, the Advanced Encryption Standard (AES) was created.
- Standard DES with 56-bit key length can be broken relatively easily nowadays through an exhaustive key search.
- DES is quite robust against known analytical attacks: In practice it is very difficult to break the cipher with differential or linear cryptanalysis.
- By encrypting with DES three times in a row, triple DES (3DES) is created, against which no practical attack is currently known.
- The "default" symmetric cipher is nowadays often AES. In addition, the other four AES finalist ciphers all seem very secure and efficient.

Lessons Learned

- There are many different ways to encrypt with a block cipher. Each mode of operation has some advantages and disadvantages
- Several modes turn a block cipher into a stream cipher
- There are modes that perform encryption together with authentication, i.e., a cryptographic checksum protects against message manipulation
- The straightforward ECB mode has security weaknesses, independent of the underlying block cipher
- The counter mode allows parallelization of encryption and is thus suited for high speed implementations
- Double encryption with a given block cipher only marginally improves the resistance against brute-force attacks
- Triple encryption with a given block cipher roughly *doubles* the key length, e.g., Triple DES (3DES) has an effective key length of 112 bits
- Key whitening enlarges the DES key length without much computational overhead.

Thank You!


DECRYPTION
ASYMMETRIC TYPE
PUBLIC
ALGORITHM
KEY
SCIENCE
DATA CENTER
SIGNATURE
SECRET
DATA
CRYPTOGRAPHY
INTEGRITY
MESSAGE
CRYPTOSYSTEM
CIPHER
SECURITY
SENDER
EXCHANGE
CODE
STREAM
NETWORK
ENCRYPTION
CIPHERTEXT
COMPUTATION
CRYPTOANALYSIS

Questions???