CS396: Security, Privacy & Society

Fall 2022

Lecture 8: Cryptographic System I I

Instructor: Abrar Alrumayh

September 28, 2022

Outline

- Pseudo-randomness
  - Pseudorandom generators
  - Pseudorandom functions
- stream ciphers/PRGs, block ciphers/PRFs, DES Vs. AES
- Modes of operations

Symmetric encryption, revisited: OTP with pseudorandomness

Perfect secrecy & randomness

Role of randomness in encryption is **integral**

- in a perfectly **secret cipher**, the ciphertext **doesn't depend** on the message
  - the ciphertext appears to be **truly random**
  - the uniform key-selection distribution **is imposed also onto** produced ciphertexts
    - e.g., c = k XOR m (for uniform k and any distribution over m)

When security is **computational**, randomness is **relaxed** to "pseudorandomness"

- the ciphertext appears to be "**pseudorandom**"
  - it **cannot be efficiently distinguished** from truly random

pseudorandomness is a computational relaxation of true randomness

Symmetric encryption as “OPT with pseudorandomness”

Stream cipher

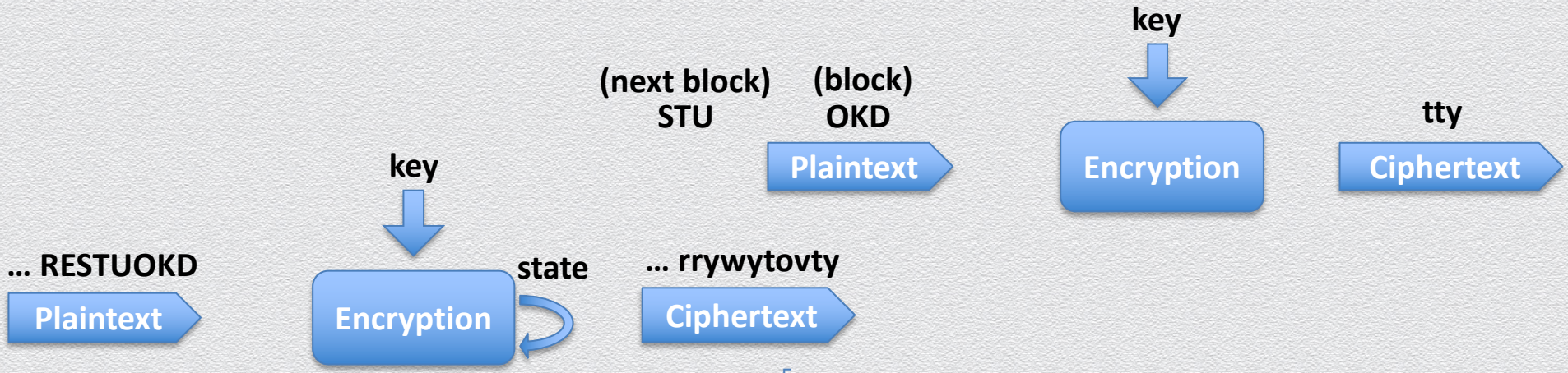
Uses a **short** key to encrypt **long** symbol **streams** into a **pseudorandom** ciphertext

- based on abstract crypto primitive of **pseudorandom generator (PRG)**

Block cipher

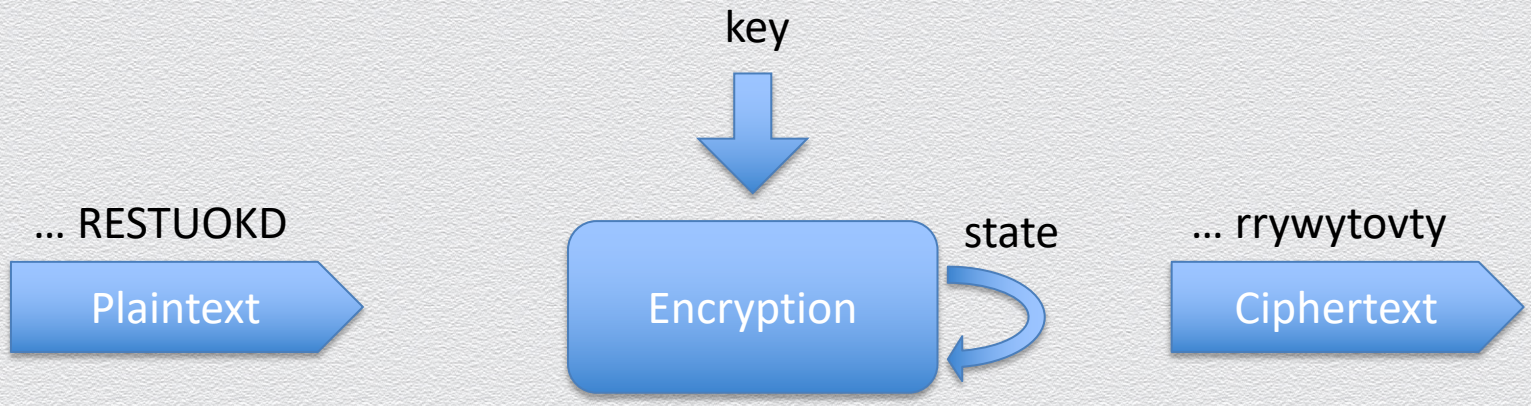
Uses a **short** key to encrypt **blocks** of symbols into **pseudorandom** ciphertext blocks

- based on abstract crypto primitive of **pseudorandom function (PRF)**



Pseudorandom generators

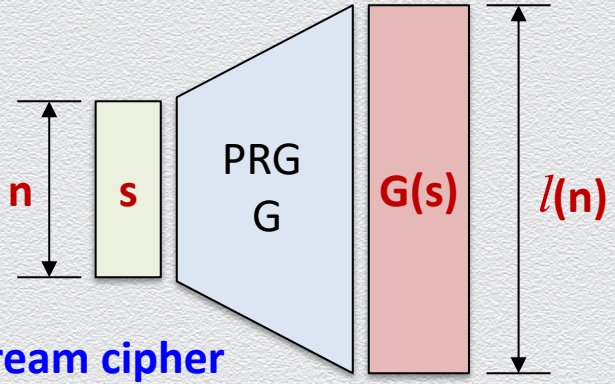
Stream ciphers



Pseudorandom generators (PRGs)

Deterministic algorithm G that

on input a **seed** $s \in \{0,1\}^t$, outputs $G(s) \in \{0,1\}^{l(t)}$



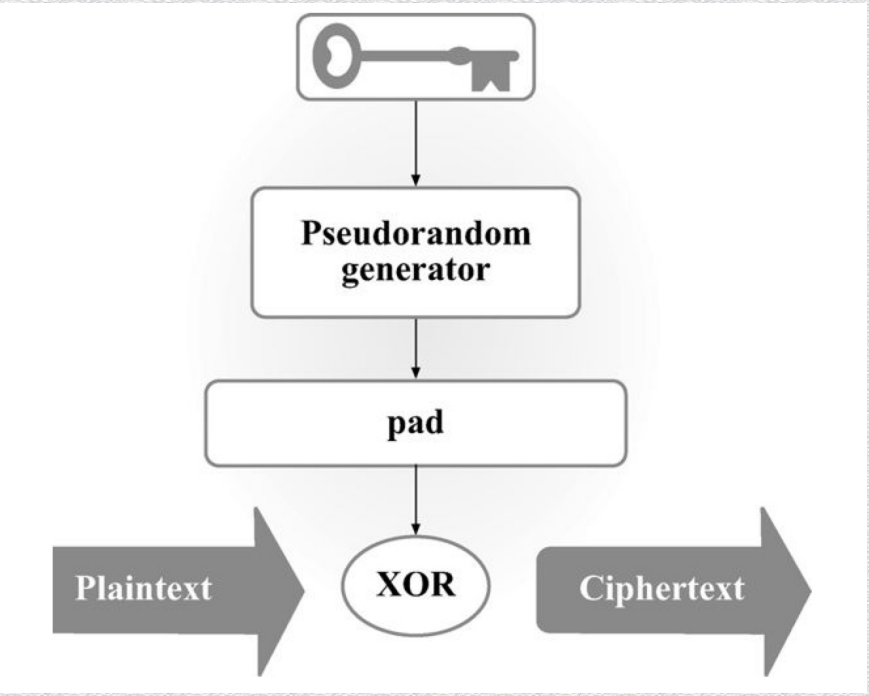
a.k.a. **stream cipher**

G is a PRG if:

- **expansion**
  - for polynomial *l*, it holds that for any n, *l*(n) > n
  - models the process of **extracting** randomness from a short random string
- **pseudorandomness**
  - **no efficient statistical test can tell apart** G(s) from a **truly random** string

Generic PRG-based symmetric encryption

- **Fixed-length** message encryption



encryption scheme is plain-secure
as long as the underlying PRG is secure

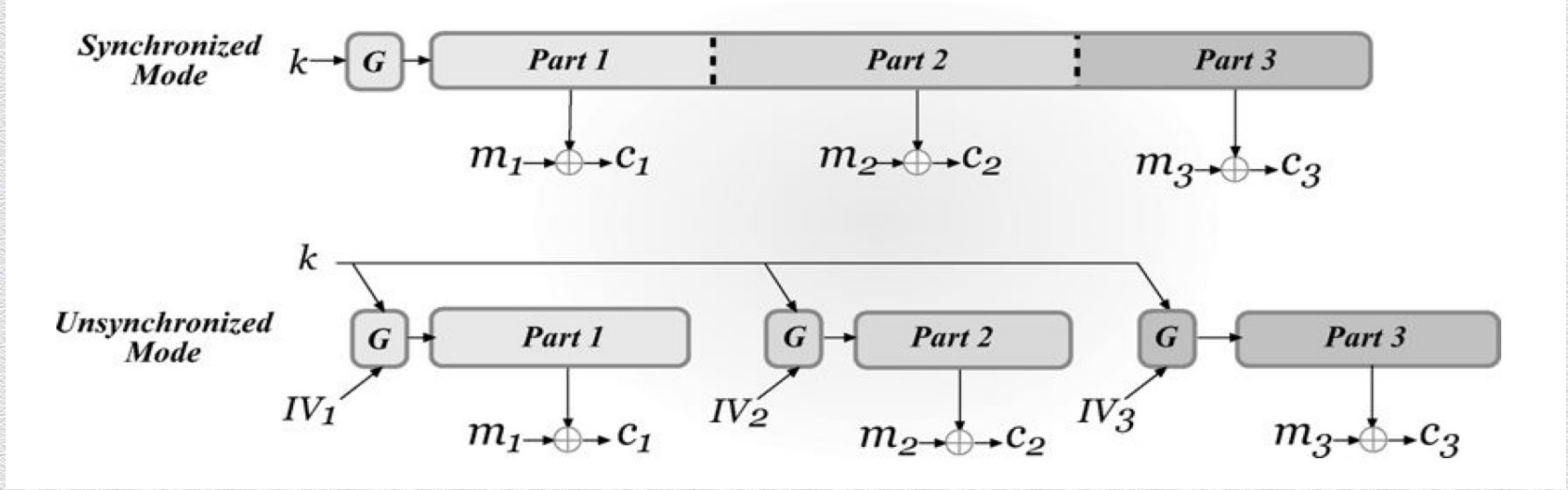
Generic PRG-based symmetric encryption (cont.)

- **Bounded- or arbitrary-length** message encryption
  - specified by a mode of operation for using an underlying stateful stream cipher, repeatedly, to encrypt/decrypt a stream of symbols

Stream ciphers: Modes of operations

- **Bounded or arbitrary-length** message encryption

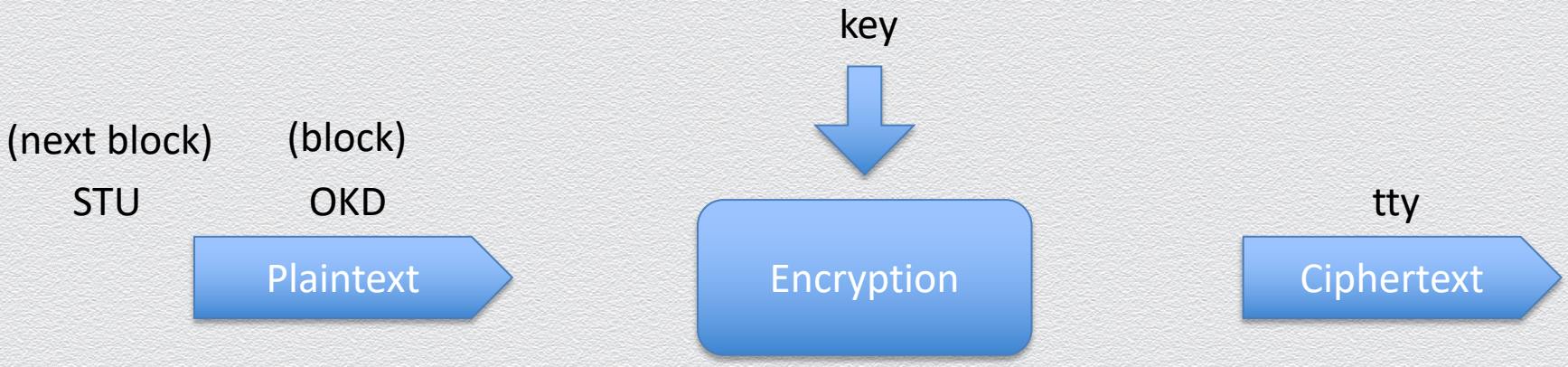
on-the-fly computation of new pseudorandom bits, no IV needed, plain-secure



random IV used for every new message is sent along with ciphertext, advanced-secure

Pseudorandom functions

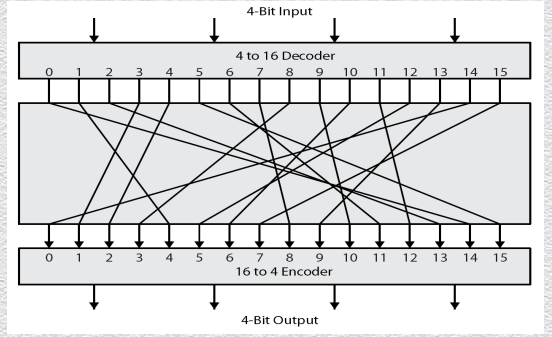
Block ciphers



Realizing ideal block ciphers in practice

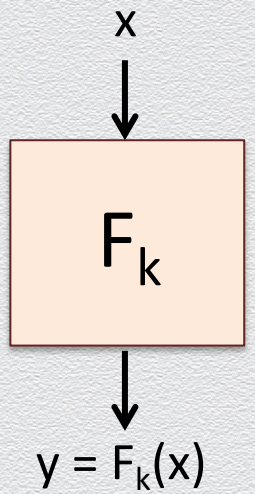
We want a **random** mapping of n-bit inputs to n-bit outputs

- there are ~2^(n2^n) possible such mappings
- none of the above can be implemented in practice



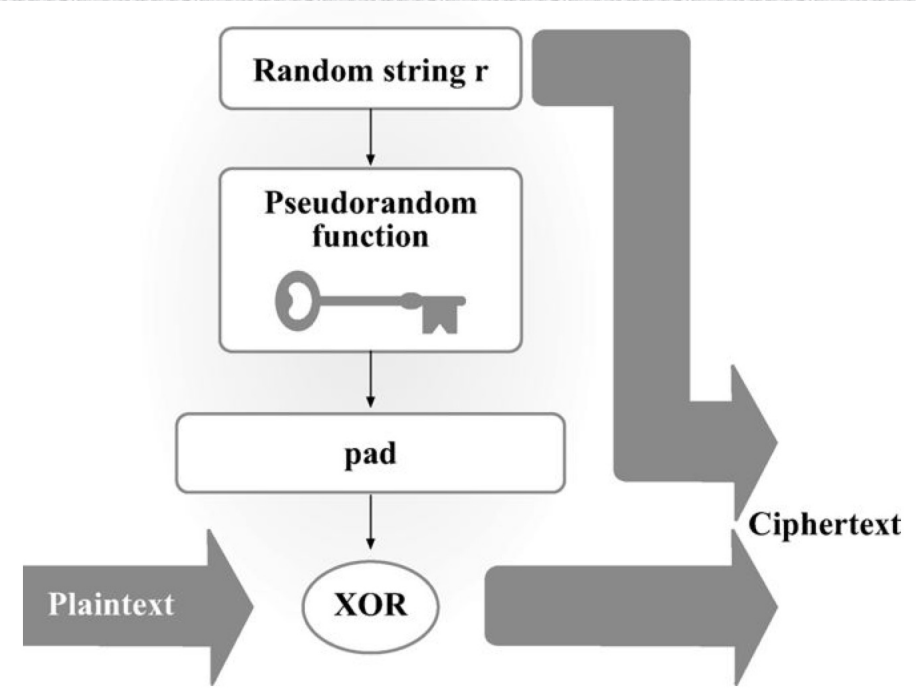
Instead, we use a keyed function $F_k : \{0,1\}^n \rightarrow \{0,1\}^n$

- indexed by a t-bit key k
- there are only 2^t such keyed functions
- a random key selects a "random-enough" mapping or a **pseudorandom function**



Generic PRF-based symmetric encryption

- **Fixed-length** message encryption



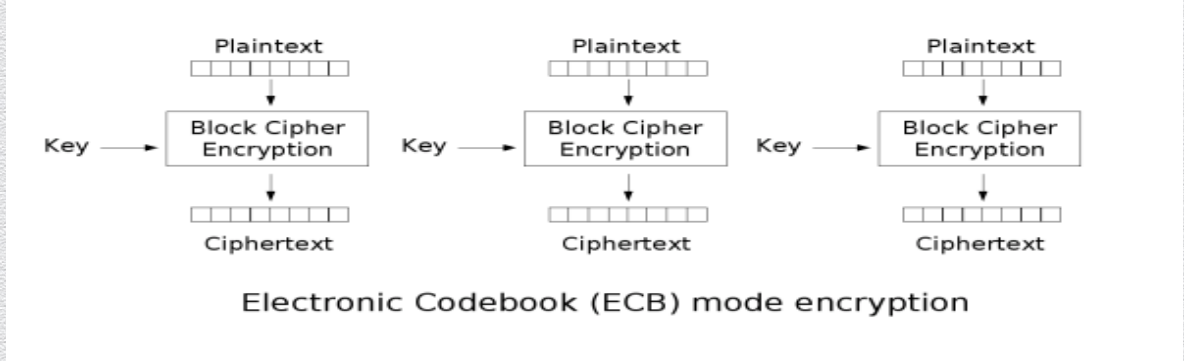
encryption scheme is advanced-secure as long as the underlying PRF is secure

Generic PRF-based symmetric encryption (cont.)

- **Arbitrary-length** message encryption
  - specified by a mode of operation for using an underlying stateless block cipher, repeatedly, to encrypt/decrypt a sequence of message blocks

Electronic Code Book (ECB)

- The simplest mode of operation
  - block P[i] encrypted into ciphertext block $C[i] = Enc_k(P[i])$
  - block C[i] decrypted into plaintext block $M[i] = Dec_k(C[i])$



Electronic Codebook (ECB) mode encryption

Strengths & weaknesses of ECB

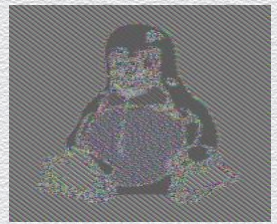
Strengths

- very simple
- allows for parallel encryptions of the blocks of a plaintext
- can tolerate the loss or damage of a block

Weaknesses

- poor security
- produces the same ciphertext on the same plaintext (under the same key)
- documents and images are not suitable for ECB encryption, since patterns in the plaintext are repeated in the ciphertext
- e.g.,

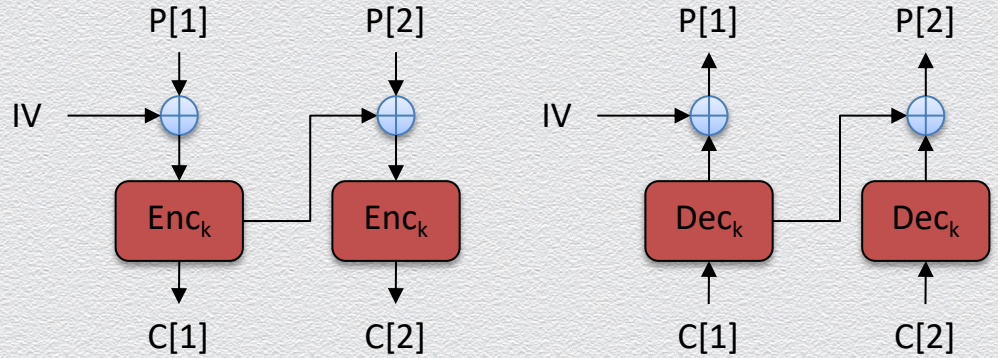
ECB



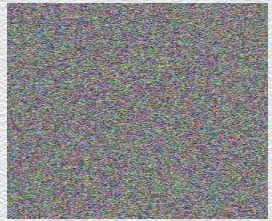
Cipher Block Chaining (CBC) [or chaining]

Alternatively, the previous-block ciphertext is "mixed" with the current-block plaintext

- e.g., using XOR
  - each block is encrypted as $C[i] = Enc_k(C[i-1] \oplus P[i])$,
  - each ciphertext is decrypted as $P[i] = C[i-1] \oplus Dec_k(C[i])$
  - here, C[0] = IV is a uniformly random initialization vector that is transmitted separately



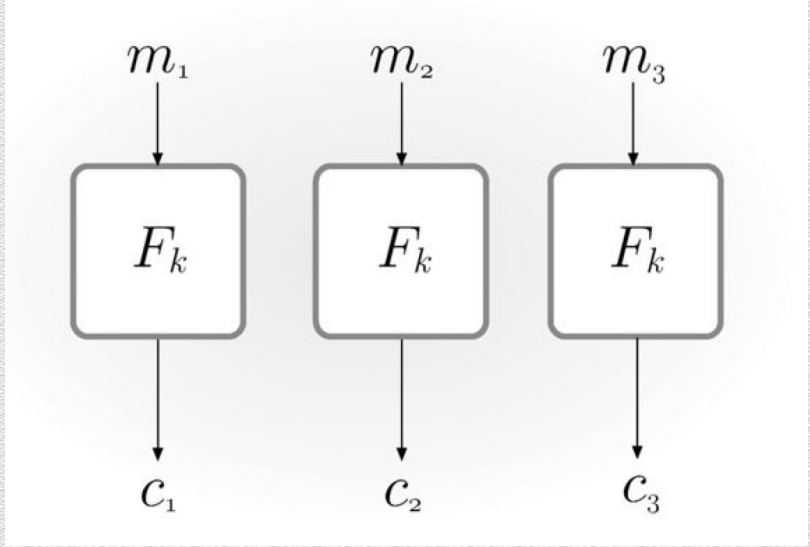
CBC



Modes of operations

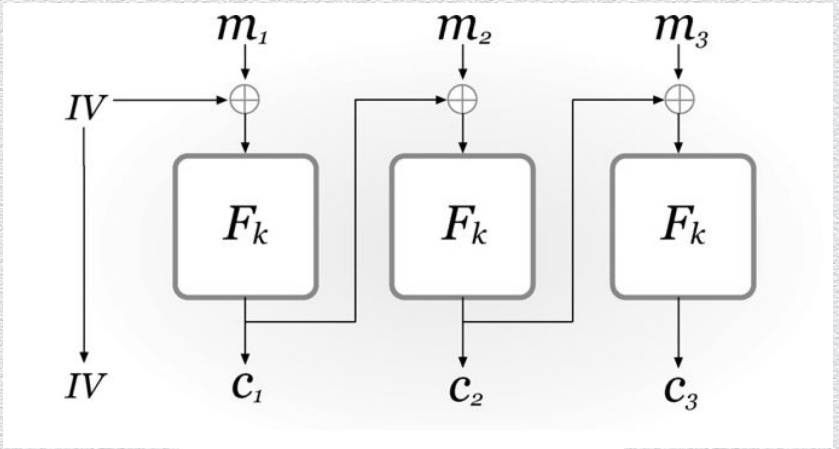
Block ciphers: Modes of operations (I)

- ECB - electronic code book
  - insecure, of only historic value
  - deterministic, thus not CPA-secure
  - actually, not even EAV-secure

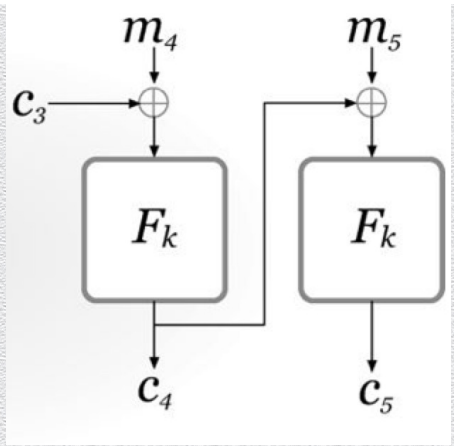


Block ciphers: Modes of operations (II)

- CBC – cipher block chaining
  - CPA-secure if F_k a permutation
  - uniform IV
    - otherwise security breaks

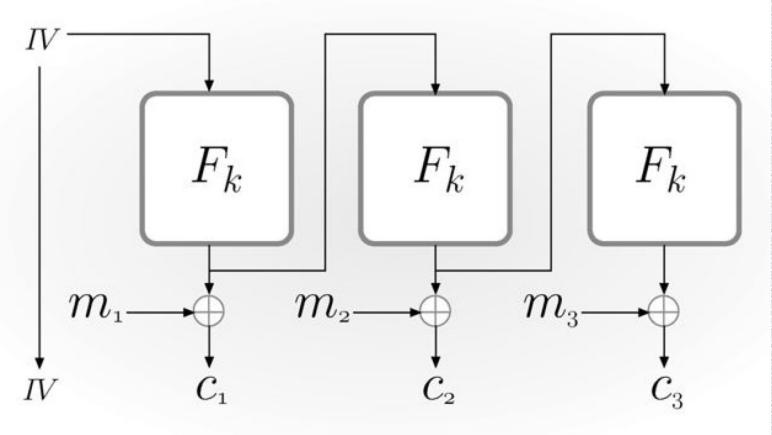


- Chained CBC
  - use last block ciphertext of current message as IV of next message
  - saves bandwidth but not CPA-secure



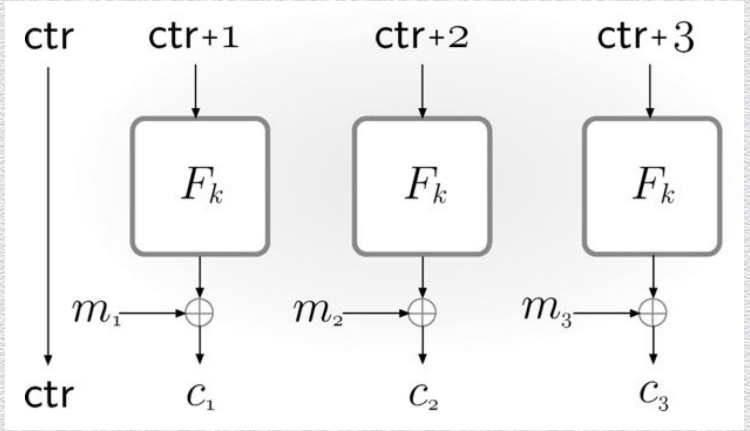
Block ciphers: Modes of operations (III)

- OFB – output feedback
  - uniform IV
  - no need message length to be multiple of n
  - resembles synchronized stream-cipher mode
  - CPA-secure if F_k is PRF



Block ciphers: Modes of operations (IV)

- CTR – counter mode
  - uniform ctr
  - no need message length to be multiple of n
  - resembles synchronized stream-cipher mode
  - CPA-secure if F_k is PRF
  - no need for F_k to be invertible
  - parallelizable



Notes on modes of operation

- block length matters
  - if small, IV or ctr can be “recycled”
- IV are often misused
  - e.g., reused or not selected uniformly at random
  - in this case, CBC is a better option than OFB/CTR

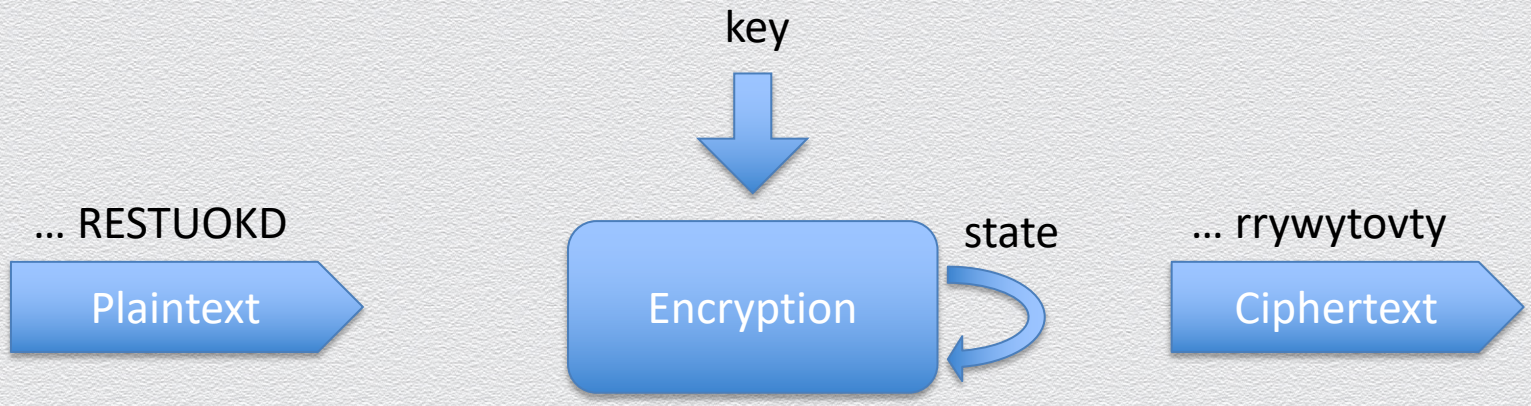
Brute-force attacks against stream/block ciphers

Brute-force attack amounts to checking all possible 2^t seeds/keys

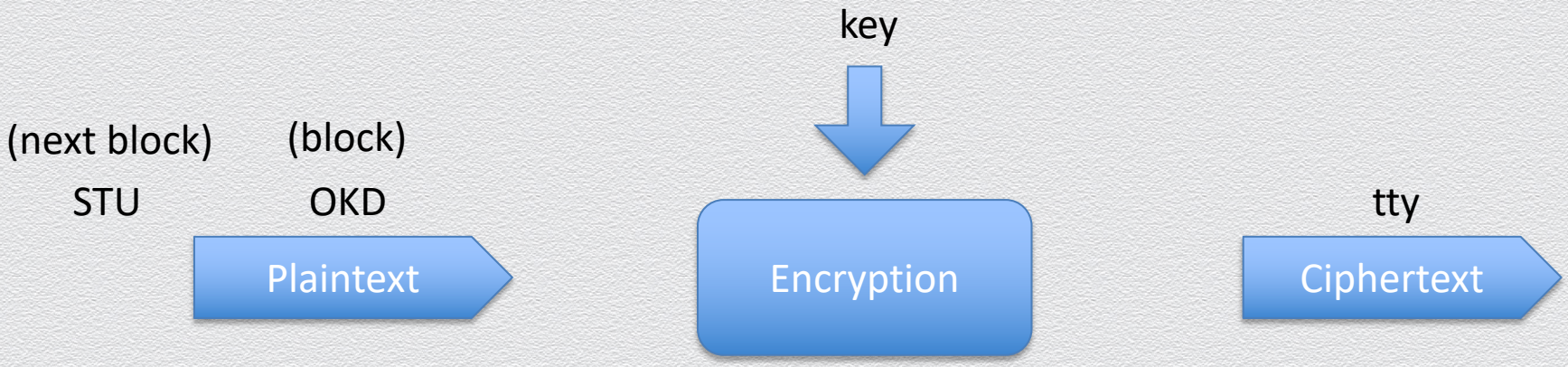
- for block ciphers, by construction (due to confusion & diffusion, as we will see), the key cannot be extracted even if a valid plaintext/ciphertext pair is captured
- thus, as expected, **the longer the key size the stronger the security**

Block ciphers in practice: DES & AES

Recall: Stream ciphers



Recall: Block ciphers



Stream Vs. Block ciphers

	Stream	Block
Advantages	• Speed of transformation • Low error propagation	• High diffusion • Immunity to insertion of symbol
Disadvantages	• Low diffusion • Susceptibility to malicious insertions and modifications	• Slowness of encryption • Padding • Error propagation

Techniques used in practice for symmetric encryption

- Substitution
  - exchanging one set of bits for another set
- Transposition
  - rearranging the order of the ciphertext bits
    - to break any regularities in the underlying plaintext
- Confusion
  - enforcing complex functional relationship between the plaintext/key pair & the ciphertext
    - e.g., flipping a bit in plaintext or key causes unpredictable changes to new ciphertext
- Diffusion
  - distributes information from single plaintext characters over entire ciphertext output
    - e.g., even small changes to plaintext result in broad changes to ciphertext

Substitution boxes

- substitution can also be done on binary numbers
- such substitutions are usually described by substitution boxes, or S-boxes

	00	01	10	11
00	0011	0100	1111	0001
01	1010	0110	0101	1011
10	1110	1101	0100	0010
11	0111	0000	1001	1100

(a)

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

(b)

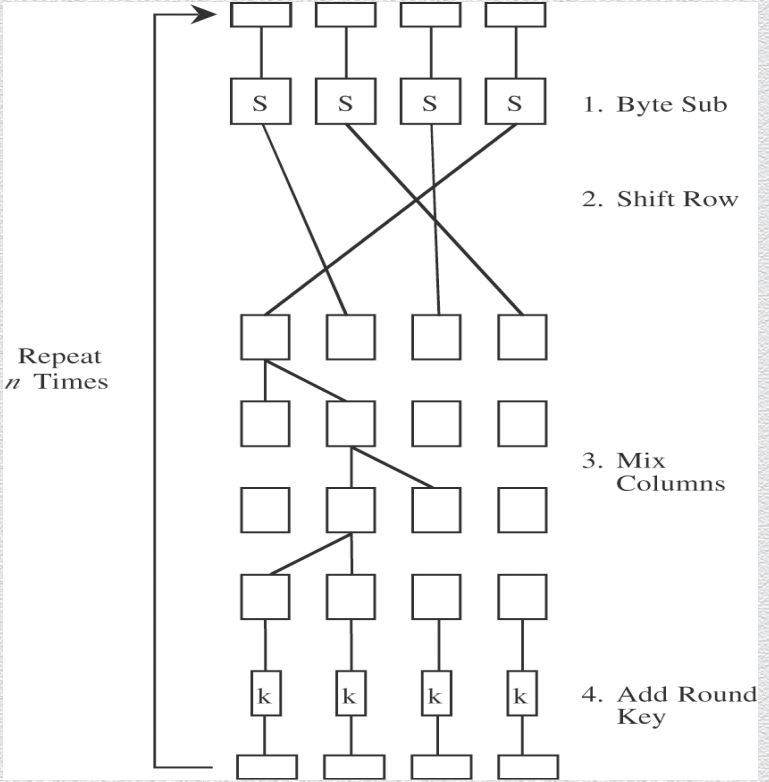
Figure 8.3: A 4-bit S-box (a) An S-box in binary. (b) The same S-box in decimal.

DES vs. AES

	DES	AES
Date designed	1976	1999
Block size	64 bits	128 bits
Key length	56 bits (effective length); up to 112 bits with multiple keys	128, 192, 256 (and possibly more) bits
Operations	16 rounds	10, 12, 14 (depending on key length); can be increased
Encryption primitives	Substitution, permutation	Substitution, shift, bit mixing
Cryptographic primitives	Confusion, diffusion	Confusion, diffusion
Design	Open	Open
Design rationale	Closed	Open
Selection process	Secret	Secret, but open public comments and criticisms invited
Source	IBM, enhanced by NSA	Independent Dutch cryptographers

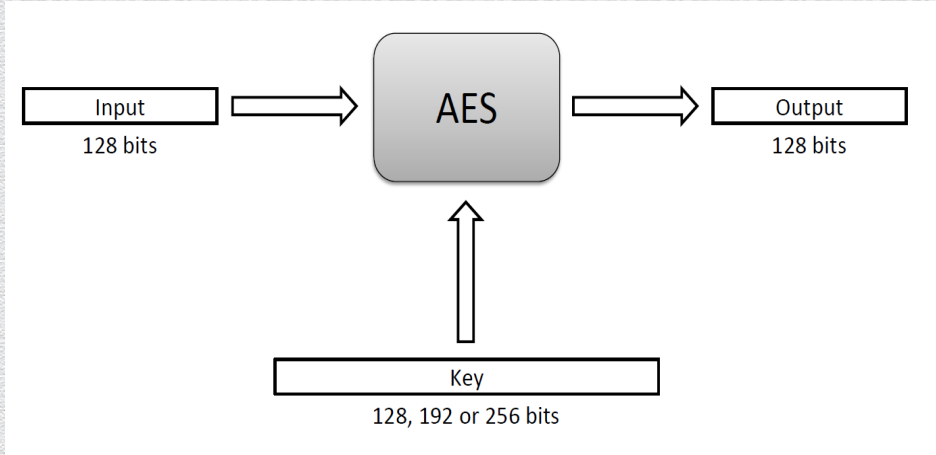
AES: Advanced Encryption System

- symmetric block cipher, a.k.a. Rijndael
- developed in 1999 by independent Dutch cryptographers in response to the 1997 NIST's public call for a replacement to DES
- still in common use
  - on the longevity of AES
    - larger key sizes possible to use
    - not known serious practical attacks

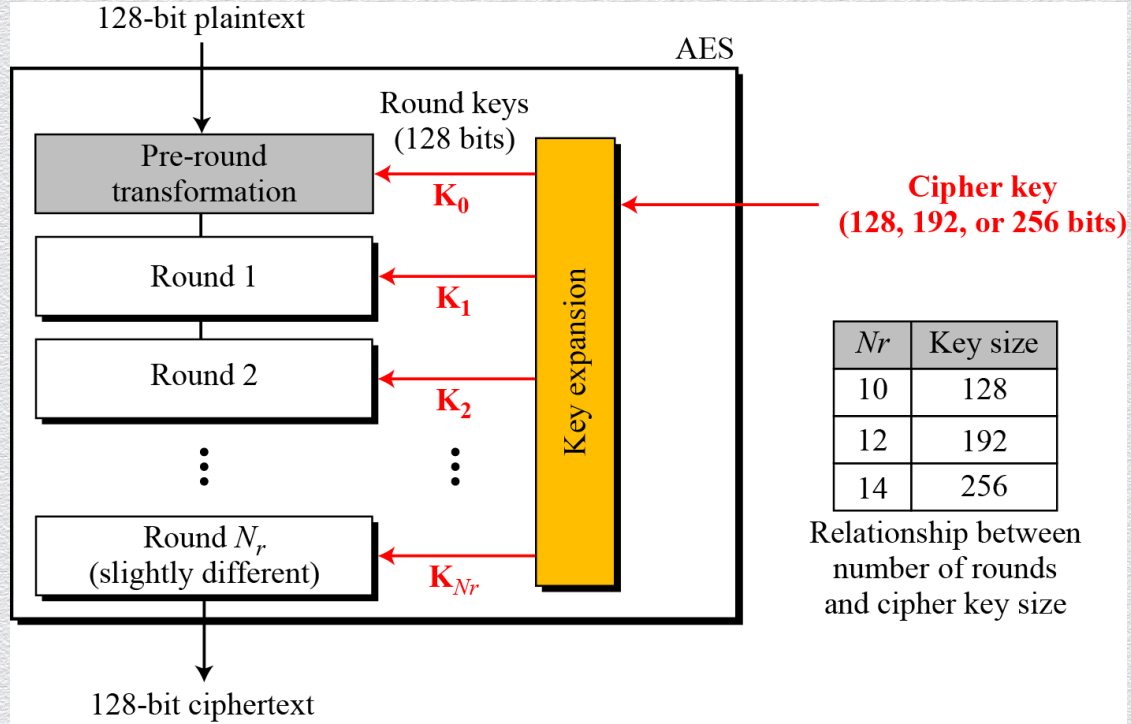


AES: Key design features

- use of substitution, confusion & diffusion
- block size is 128 bits
- variable-length keys: key size is 128, 192 or 256 bits
  - variable number of rounds: 10, 12 or 14 rounds for keys of resp. 128, 192 or 256 bits
  - depending on key size, yields ciphers known as AES-128, AES-192, and AES-256



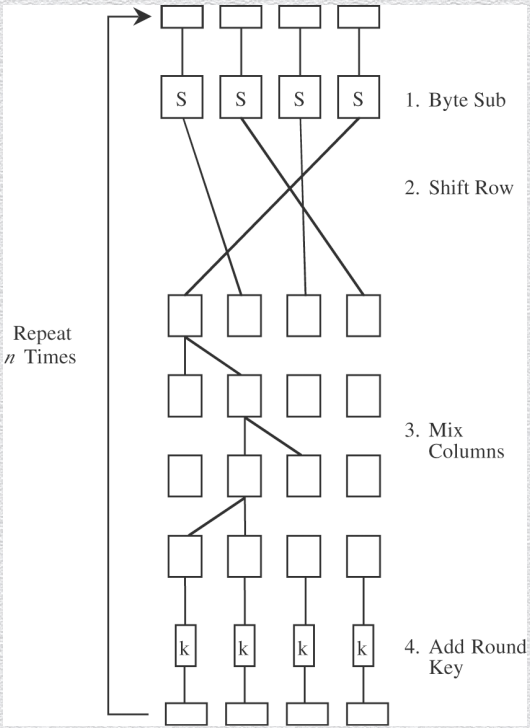
AES: Basic structure



Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

AES: Basic structure (cont.)



DES: The Data Encryption Standard

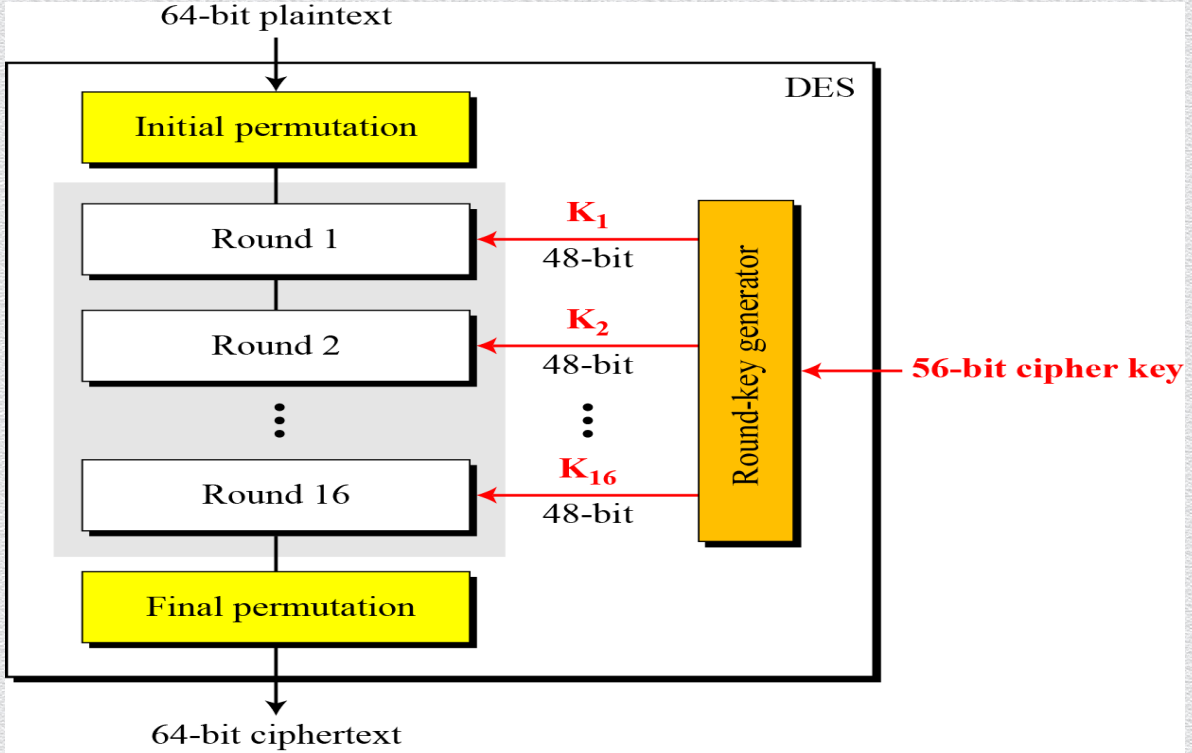
- Symmetric block cipher
- Developed in 1976 by IBM for the US National Institute of Standards and Technology (NIST)
- Employs substitution & transposition, on top of each other, for 16 rounds
  - block size = 64 bits, key size = 56 bits
- Strengthening (since 56-bit security is not considered adequately strong)
  - double DES: $E(k_2, E(k_1, m))$, not effective!
  - triple DES: $E(k_3, \mathbf{E}(k_2, E(k_1, m)))$, more effective
    - two keys, i.e., $k_1=k_3$, with E-D-E pattern, 80-bit security
    - three keys with E-E-E pattern, 112-bit security

DES: Security strength

Form	Operation	Properties	Strength
DES	Encrypt with one key	56-bit key	Inadequate for high-security applications by today's computing capabilities
Double DES	Encrypt with first key; then encrypt result with second key	Two 56-bit keys	Only doubles strength of 56-bit key version
Two-key triple DES	Encrypt with first key, then encrypt (or decrypt) result with second key, then encrypt result with first key (E-D-E)	Two 56-bit keys	Gives strength equivalent to about 80-bit key (about 16 million times as strong as 56-bit version)
Three-key triple DES	Encrypt with first key, then encrypt or decrypt result with second key, then encrypt result with third key (E-E-E)	Three 56-bit keys	Gives strength equivalent to about 112-bit key about 72 quintillion ($72*10^{15}$) times as strong as 56-bit version

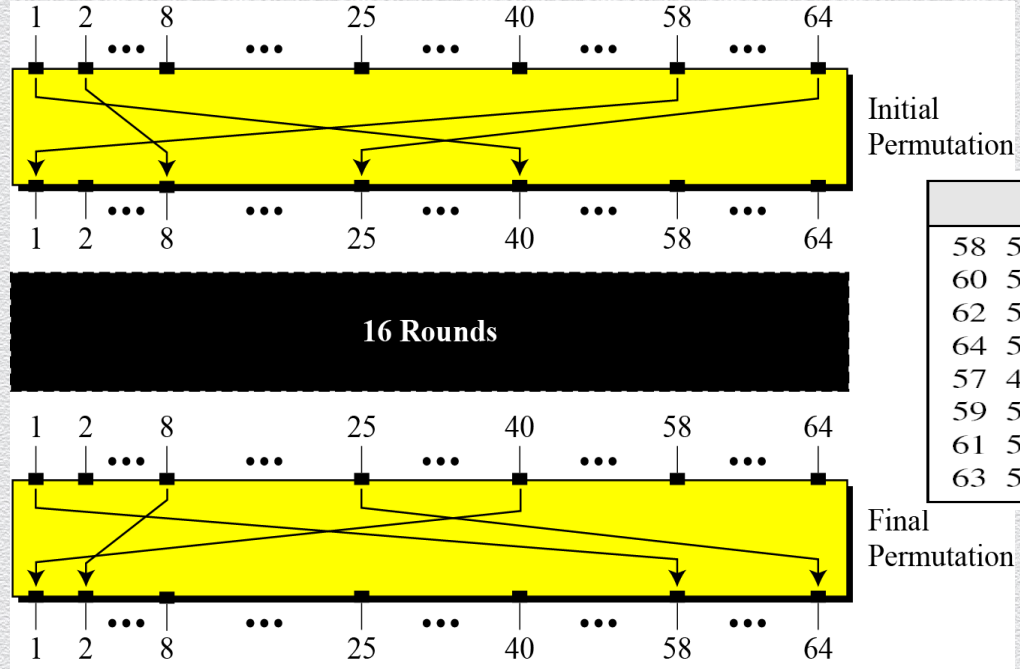
DES: High-level view

DES: Basic structure



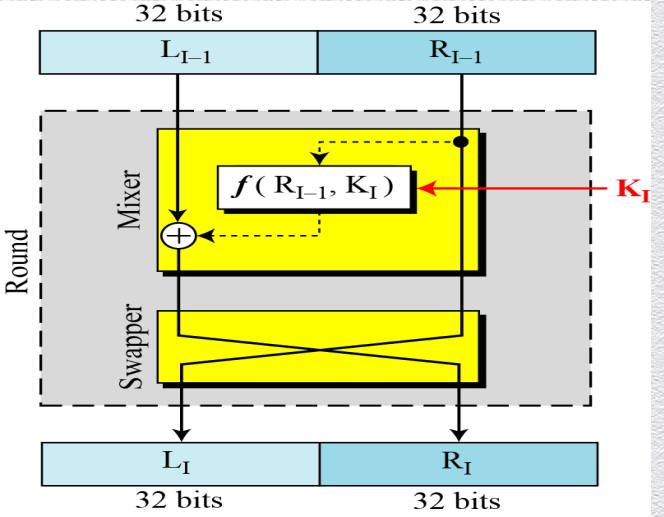
DES: Initial and final permutations

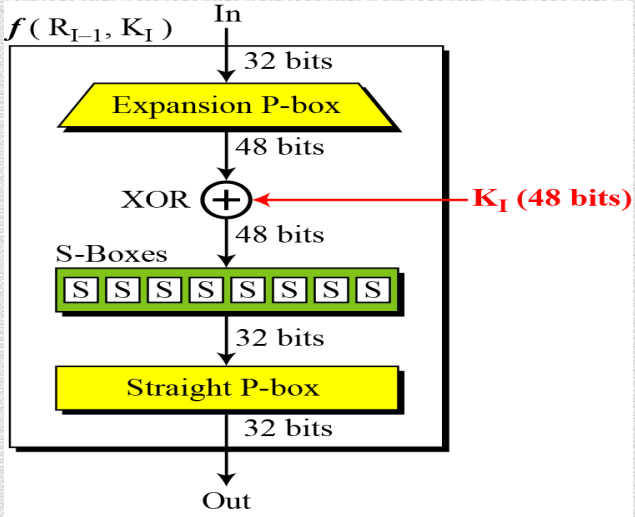
- Straight P-boxes that are inverses of each other w/out crypto significance



Initial Permutation								*Final Permutation*							
58	50	42	34	26	18	10	02	40	08	48	16	56	24	64	32
60	52	44	36	28	20	12	04	39	07	47	15	55	23	63	31
62	54	46	38	30	22	14	06	38	06	46	14	54	22	62	30
64	56	48	40	32	24	16	08	37	05	45	13	53	21	61	29
57	49	41	33	25	17	09	01	36	04	44	12	52	20	60	28
59	51	43	35	27	19	11	03	35	03	43	11	51	19	59	27
61	53	45	37	29	21	13	05	34	02	42	10	50	18	58	26
63	55	47	39	31	23	15	07	33	01	41	09	49	17	57	25

DES: Round via Feistel network

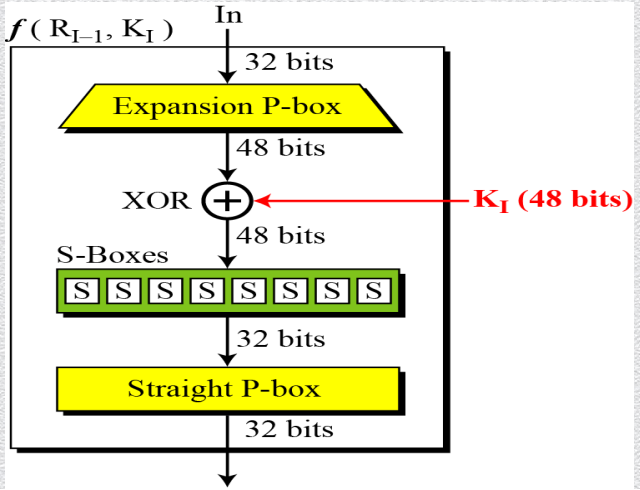


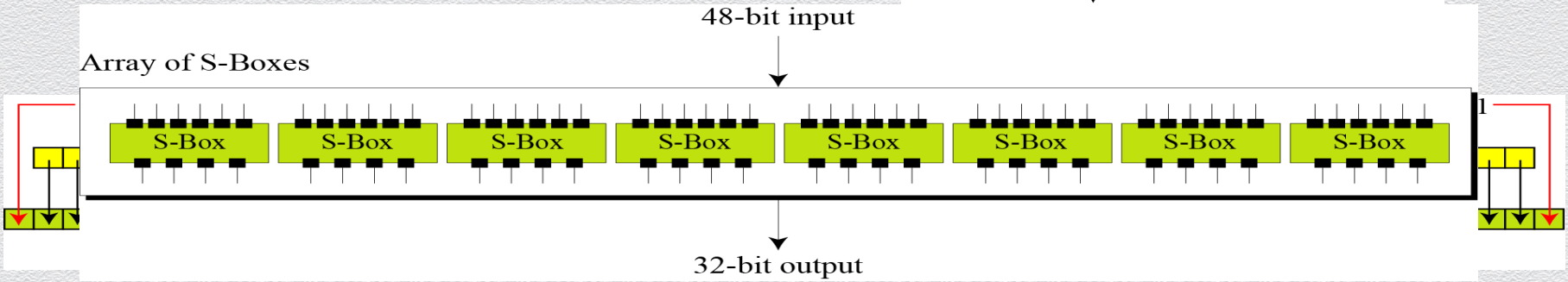


- DES uses 16 rounds, each applying a Feistel cipher
  - L(i) = R(i-1)
  - R(i) = L(i-1) XOR *f* (K(i),R(i-1)),
    where *f* applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output

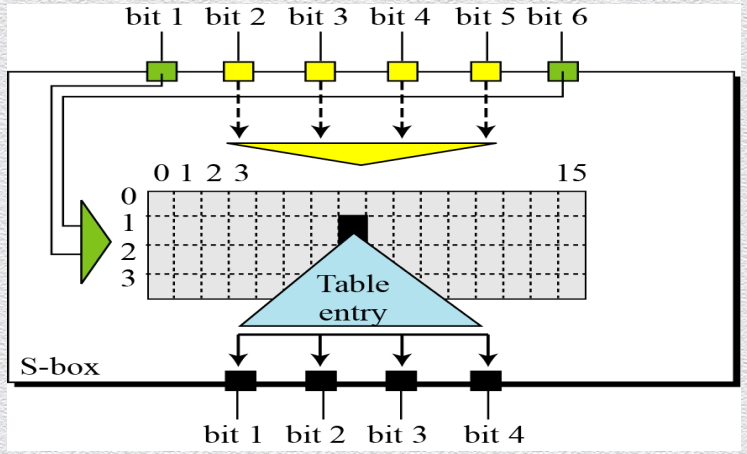
DES: Low-level view

- Expansion box
  - since R_{I-1} is a 32-bit input & K_I is a 48-bit key, we first need to expand R_{I-1} to 48 bits
- S-box
  - where real mixing (confusion) occurs
  - DES uses 8 6-to-4 bits S-boxes





DES: S-box in detail



	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	10	03	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