CS306: Introduction to IT Security

Fall 2020

Lecture 5: Ciphers in Practice (II)

Instructor: **Nikos Triandopoulos**

September 29, 2020

STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

5.0 Announcements

CS306: Other announcements

- HW1 is due this Friday
  - please
    - start early
    - ask for help if needed
    - respect the non-collaboration policy
- HW2 to come out next week

CS306: Tentative Syllabus

Week	Date	Topics	Reading	Assignment
1	**Sep 1**	**Introduction**	**Lecture 1**	**-**
2	**Sep 8**	**Symmetric-key encryption**	**Lecture 2**	**Lab 1**
3	**Sep 15**	**Perfect secrecy**	**Lecture 3**	**Lab 2, HW 1**
4	**Sep 22**	**Ciphers in practice I**	**Lecture 4**	**Lab 3, HW 1**
5	Sep 29	Public-key crypto II		
6	Oct 6	Access control & authentication		
-	Oct 13	**No class (Monday schedule)**		
7	Oct 20	**Midterm**	All materials covered	

CS306: Tentative Syllabus (continued)

Week	Date	Topics	Reading	Assignment
8	Oct 27	Software & Web security		
9	Nov 3	Network security		
10	Nov 10	Database security		
11	Nov 17	Cloud security		
12	Nov 24	Privacy		
13	Dec 1	Economics		
14	Dec 8	Legal & ethical issues		
15	Dec 10 (or later)	**Final** (closed "books")	All materials covered*	

* w/ focus on what covered after midterm

Last week

- Ciphers in practice
  - The big picture
  - Computational security
  - Pseudo-randomness
    - stream ciphers, pseudorandom generators
- Demo
  - The Caesar and Vigenère ciphers and their cryptanalysis (Evening)
  - Pseudo-randomness in practice (Afternoon)

Today

- Ciphers in practice
  - Revision
    - the big picture, computational security, pseudo-randomness, stream ciphers, PRGs
  - Block ciphers, pseudorandom functions
  - Modes of operations
  - DES, AES
- Demo
  - The Caesar and Vigenère ciphers and their cryptanalysis (Afternoon)
  - Pseudo-randomness in practice (Evening)

5.1 Revision

Recall: Formal treatment in modern cryptography

Problem is formulated as an abstract crypto primitive

- captures the essence of the problem at hand, provides clarity and focus

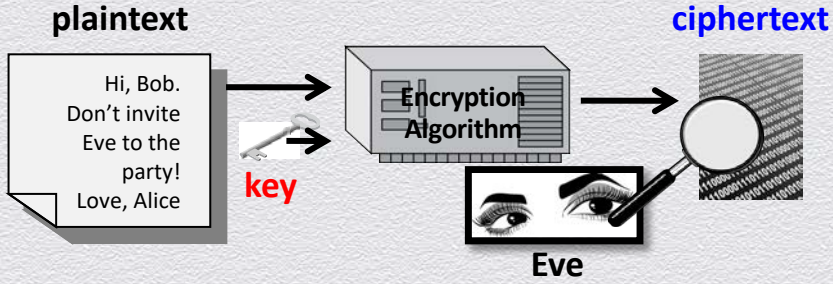
Design & evaluation of crypto primitives follows a systematic process

- (A) **formal definitions** (what it means for a crypto primitive to be secure?)
- (B) **precise assumptions** (which forms of attacks are allowed – and which aren't?)
- (C) **provable security** (why a candidate solution is secure – or not)?

Recall: Main security properties against eavesdropping

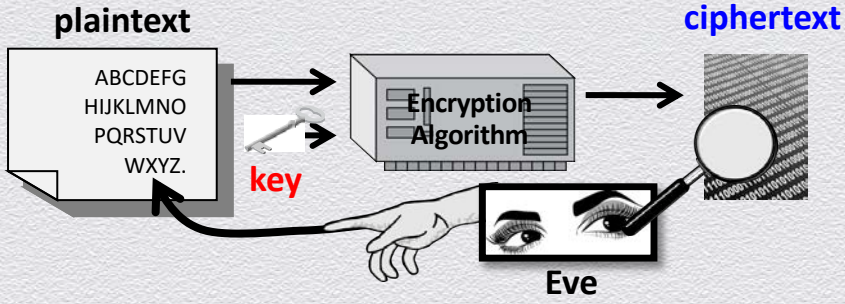
“plain” security

- protects against ciphertext-only attacks



“advanced” security

- protects against chosen plaintext attacks



Recall: Computational security to relax “perfectness

Refined model

- a relaxed notion of security, called **computational security**, requires that
  - the ciphertext leaks **a tiny amount of extra information** about the plaintext
  - to adversaries with **bounded computational power**

Asymptotic approach

- “A scheme is secure if any efficient attacker A succeeds in breaking the scheme with at most negligible probability”

Recall: Security relaxation for encryption

Perfect security: |K| = 128 bits, M, $Enc_K(M)$ are independent, **unconditionally**

- no extra information is leaked to any attacker

Computational security: M, $Enc_K(M)$ are independent, **for all practical purposes**

- no extra information is leaked but a tiny amount
  - e.g., with prob. 2^{-128} (or much less than the likelihood of being hit by lighting)
- to computationally bounded attackers
  - e.g., who cannot count to 2^{128} (or invest work of more than one century)
- attacker's best strategy remains **ineffective**
  - random guess a secret key or exhaustive search over key space (brute-force attack)

Recall: Computational plain & advanced secrecy

Relax the definition of perfect secrecy that is based on indistinguishability

- target messages m_0, m_1 are chosen by a **PPT** attacker
- no such attacker can tell $Enc_k(m_0)$, $Enc_k(m_1)$ apart **non-negligibly better** than guessing

Strengthen the definition of computational plain-security for advanced secrecy

- allow attacker to have access to an **encryption "box"**

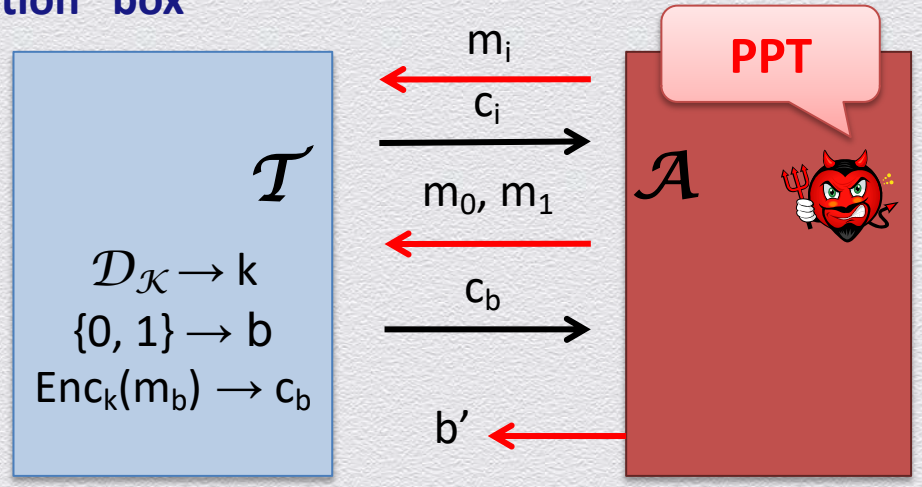
3) indistinguishability

For every PPT $\mathcal{A}$, it holds that

Pr[b' = b] = 1/2 + negligible

PPT

something that can be safely ignored



Recall: Symmetric encryption as "OPT w/ pseudorandomness"

Stream cipher

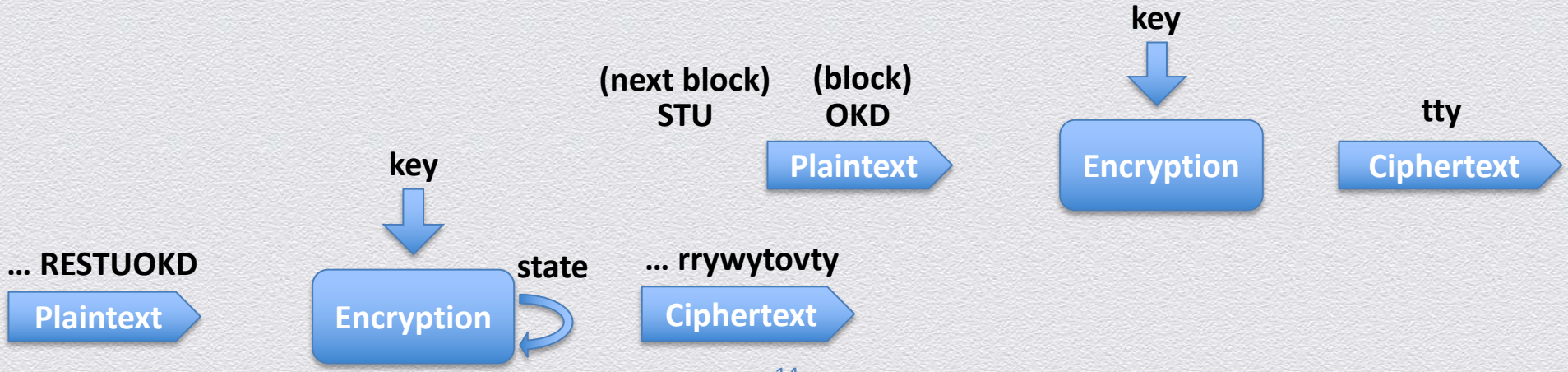
Uses a **short** key to encrypt **long** symbol **streams** into a **pseudorandom** ciphertext

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

Block cipher

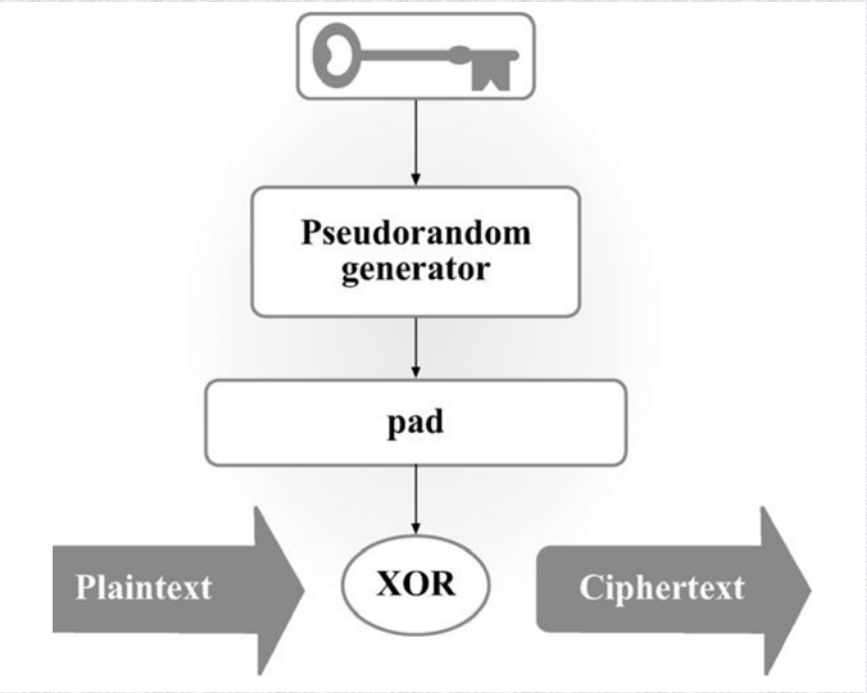
Uses a **short** key to encrypt **blocks** of symbols into **pseudorandom** ciphertext blocks

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



Recall: Generic PRG-based symmetric encryption

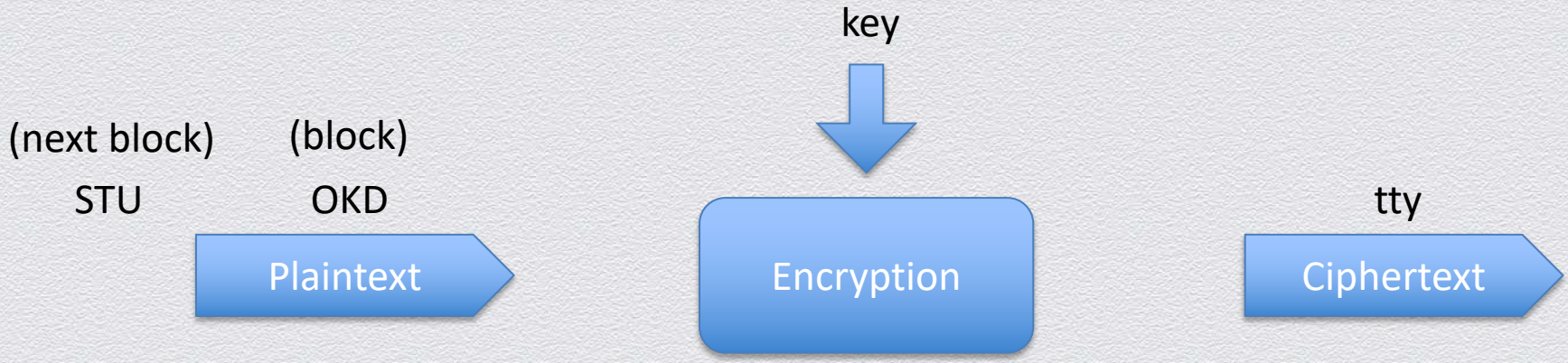
- **Fixed-length** message encryption



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

5.2 Pseudorandom functions

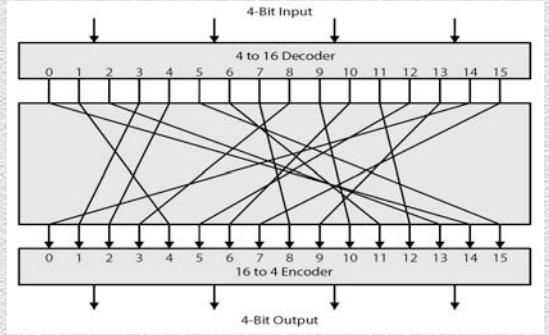
Block ciphers



Realizing ideal block ciphers in practice

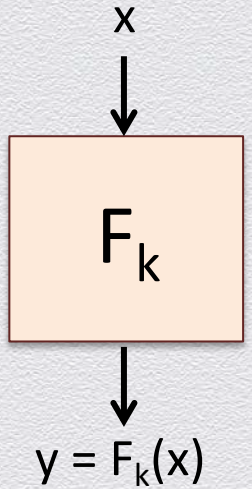
We want a **random** mapping of n-bit inputs to n-bit outputs

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



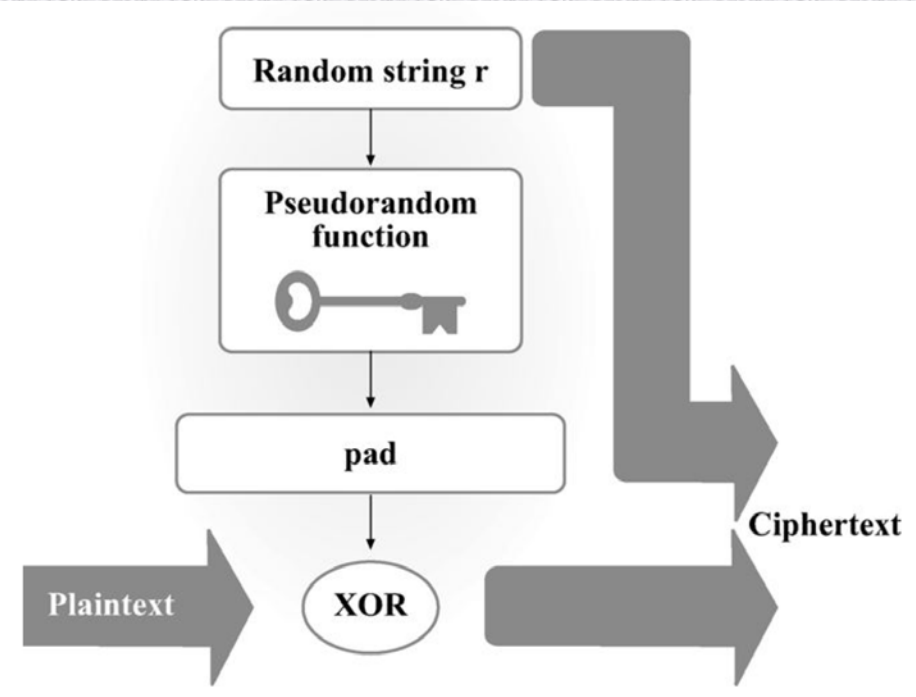
Instead, we use a keyed function $F_k : \{0,1\}^n \rightarrow \{0,1\}^n$

- 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



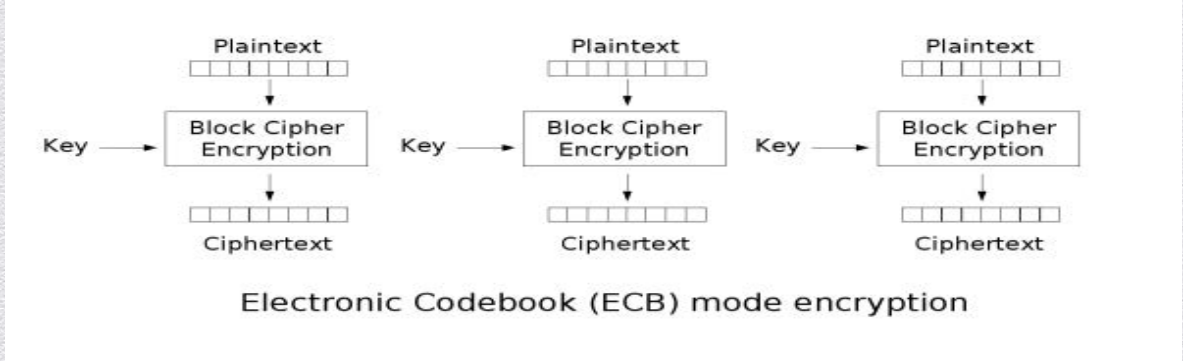
encryption scheme is advanced-secure as long as the underlying PRF is secure

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

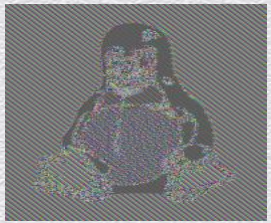
Strengths

- 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.,

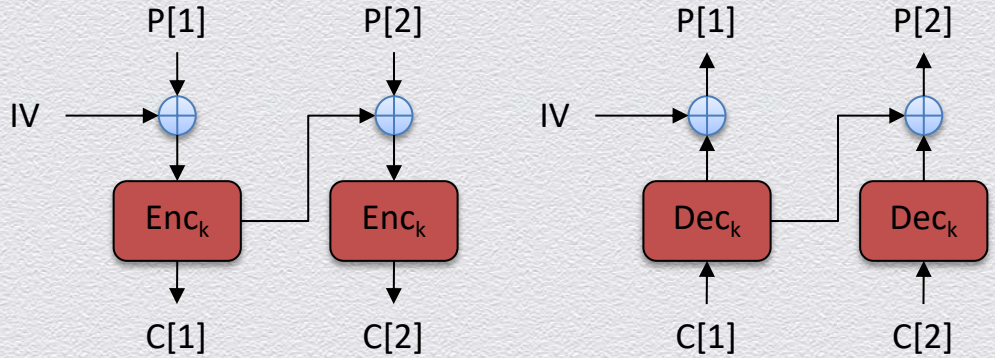
ECB



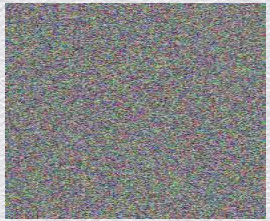
Cipher Block Chaining (CBC) [or chaining]

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



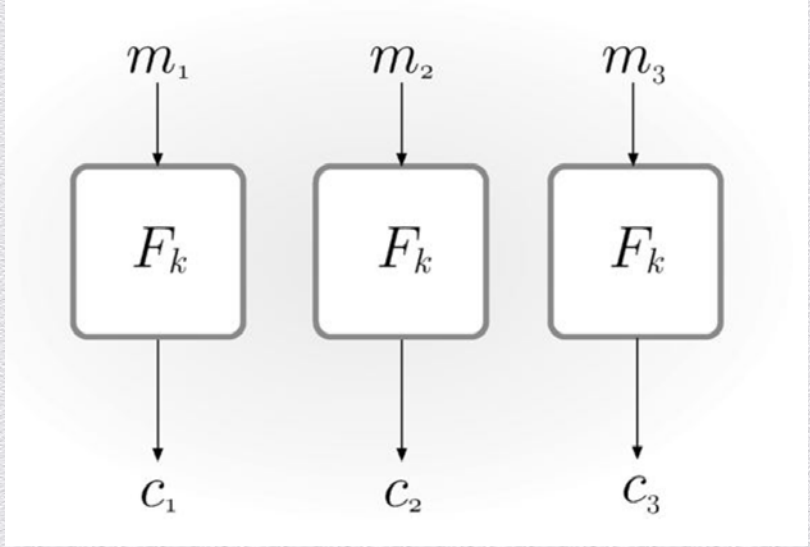
CBC



5.3 Modes of operations

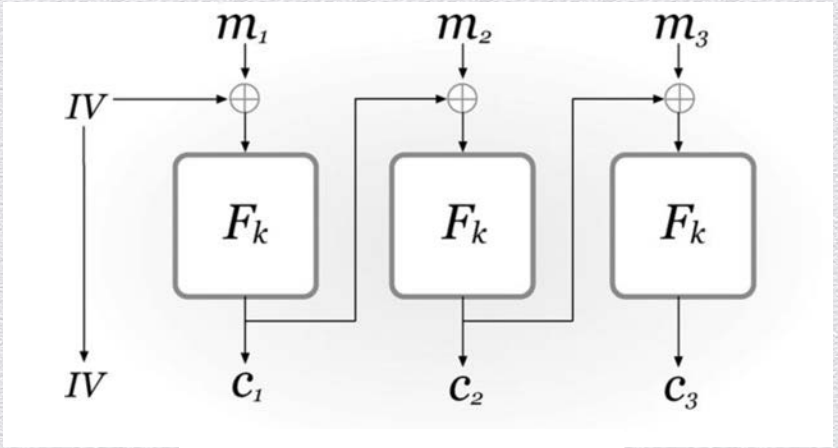
Block ciphers: Modes of operations (I)

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

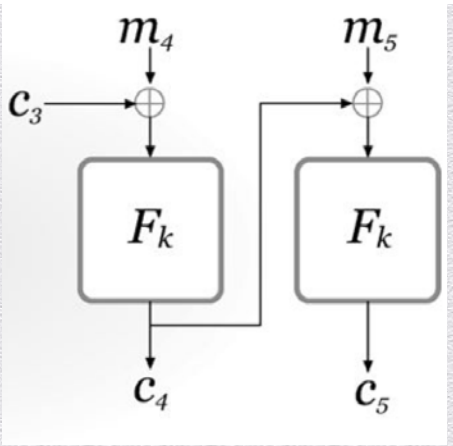


Block ciphers: Modes of operations (II)

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

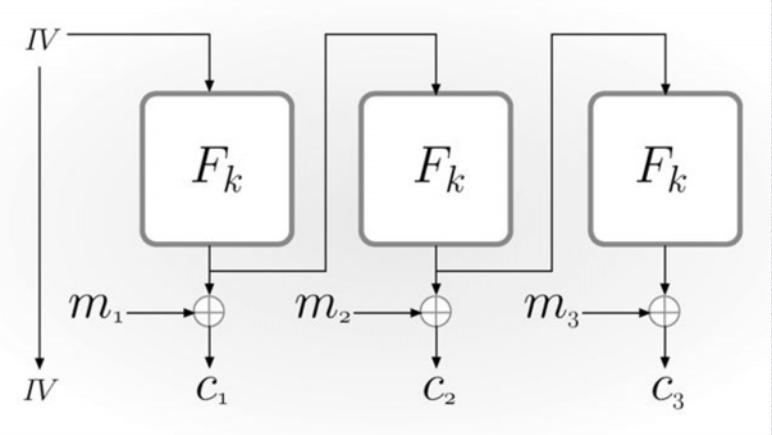


- Chained CBC
  - use last block ciphertext of current message as IV of next message
  - saves bandwidth but not CPA-secure



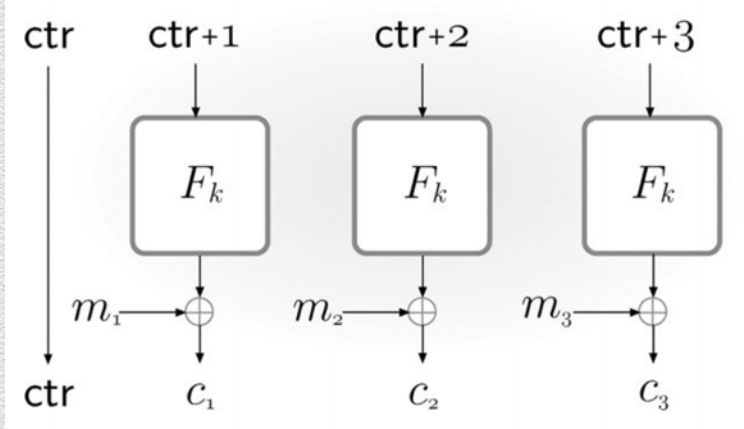
Block ciphers: Modes of operations (III)

- 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

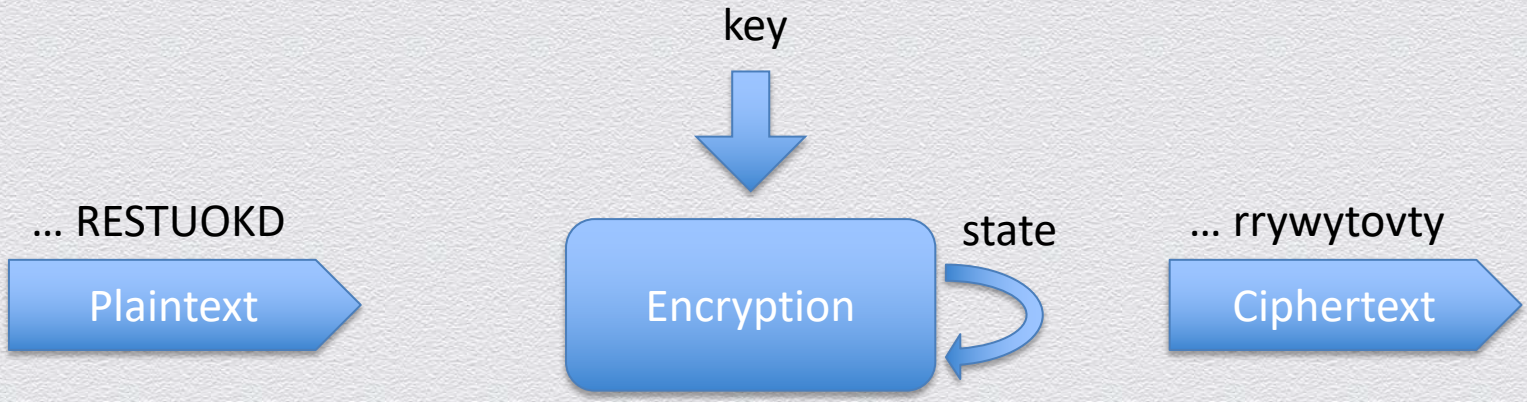
Brute-force attacks against stream/block ciphers

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

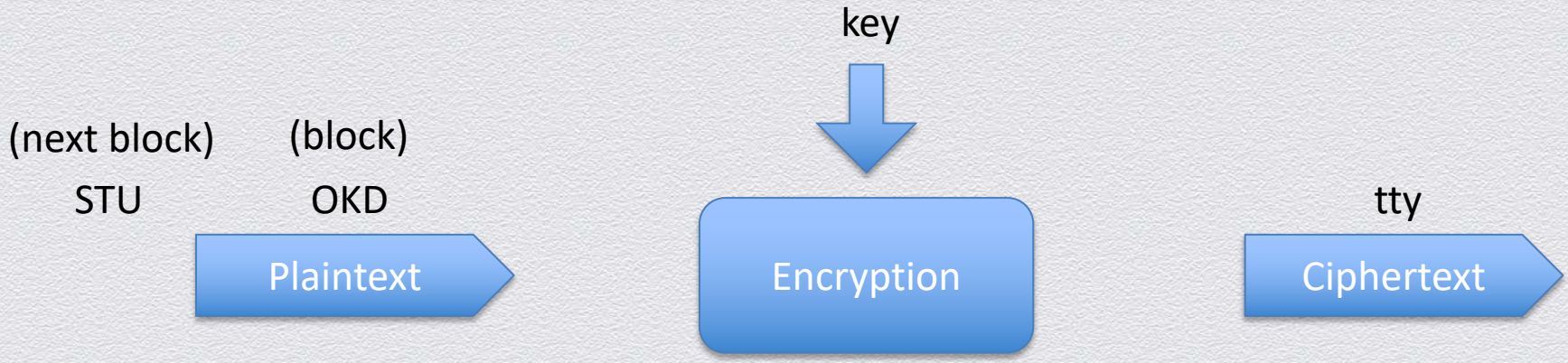
- for block ciphers, by construction (due to confusion & diffusion, as we will see), the key cannot be extracted even if a valid plaintext/ciphertext pair is captured
- thus, as expected, **the longer the key size the stronger the security**

5.4 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)

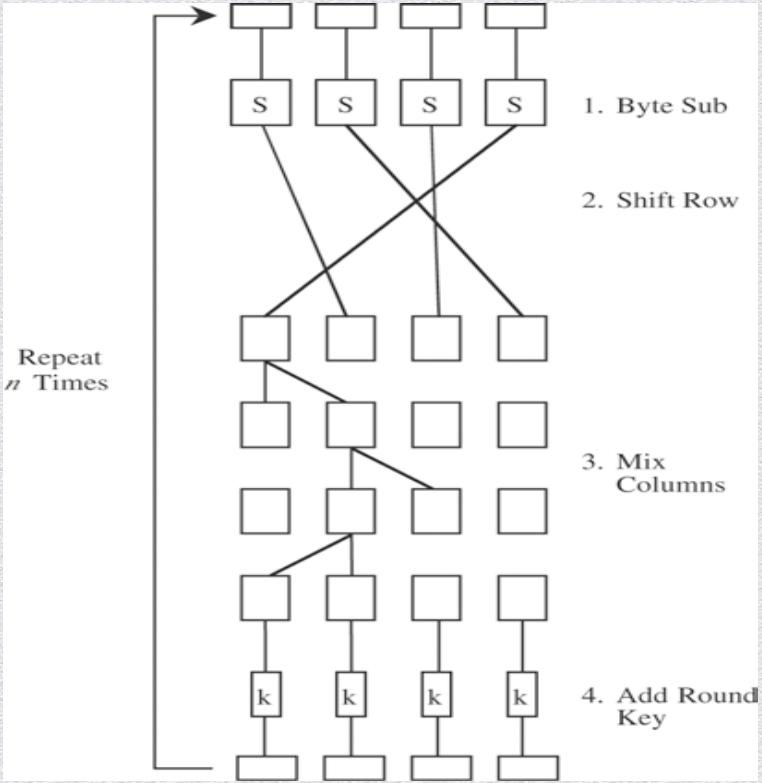
Figure 8.3: A 4-bit S-box (a) An S-box in binary. (b) The same S-box in decimal.

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

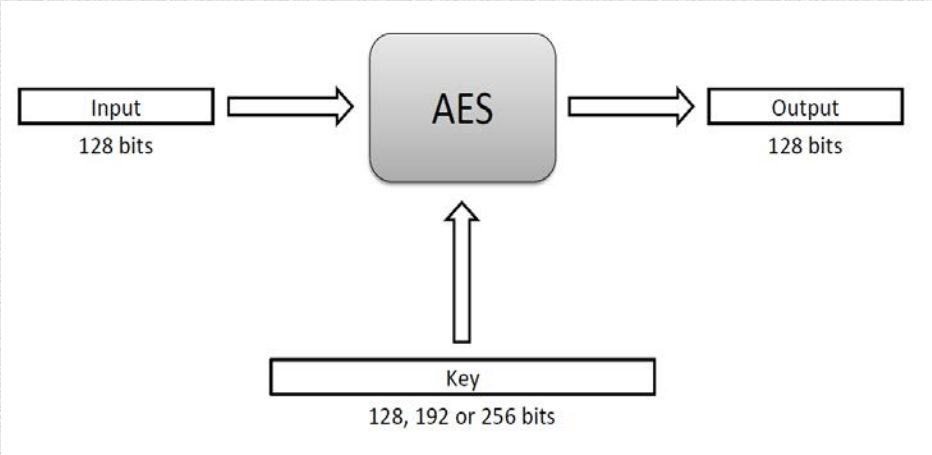
AES: Advanced Encryption System

- 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

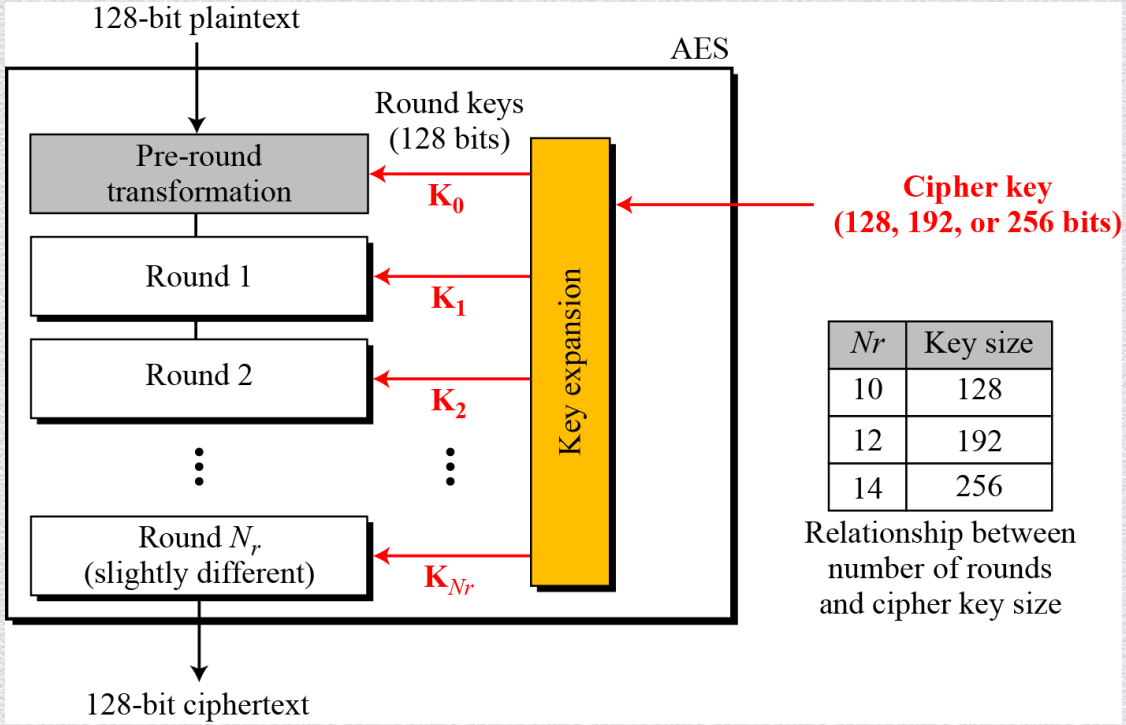


AES: Key design features

- 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



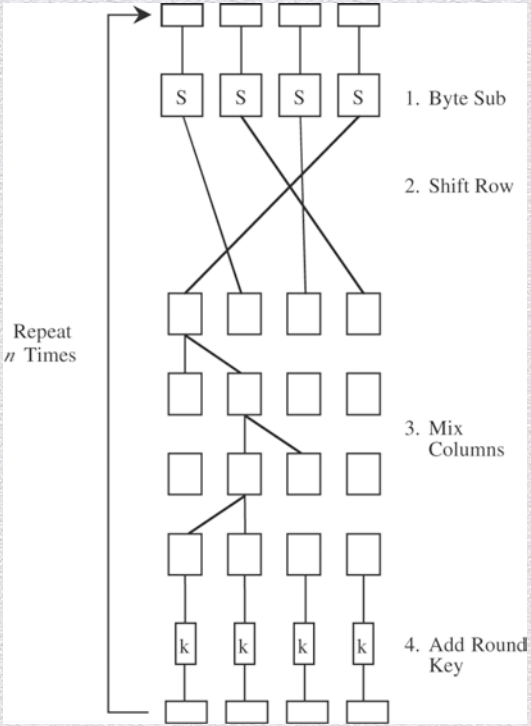
AES: Basic structure



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

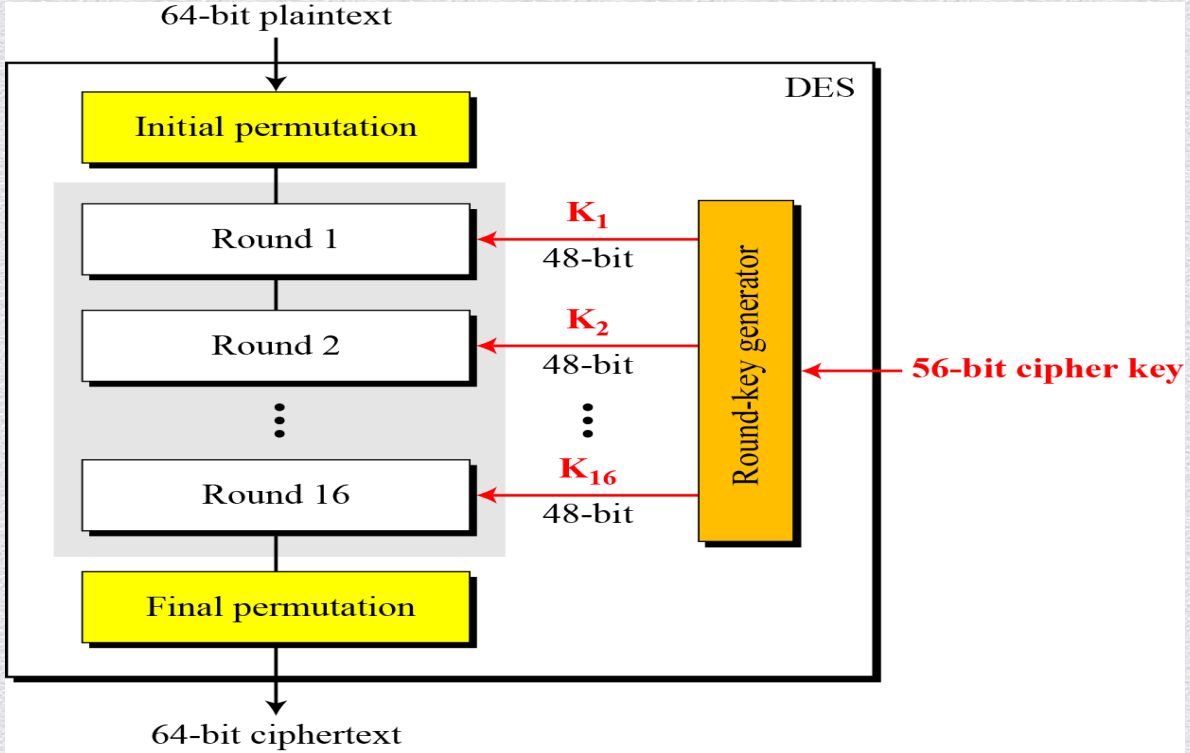
- Symmetric block cipher
- Developed in 1976 by IBM for the US National Institute of Standards and Technology (NIST)
- Employs substitution & transposition, on top of each other, for 16 rounds
  - block size = 64 bits, key size = 56 bits
- Strengthening (since 56-bit security is not considered adequately 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

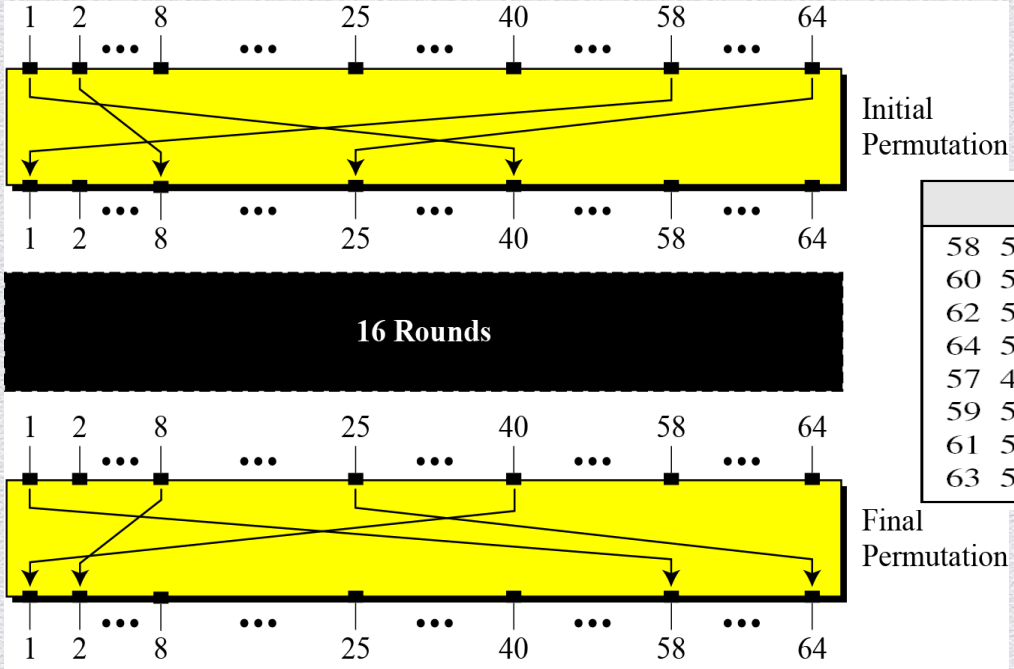
DES: High-level view

DES: Basic structure



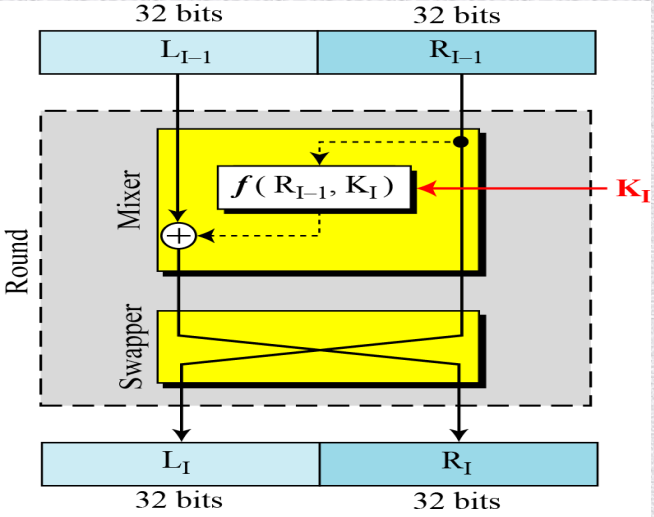
DES: Initial and final permutations

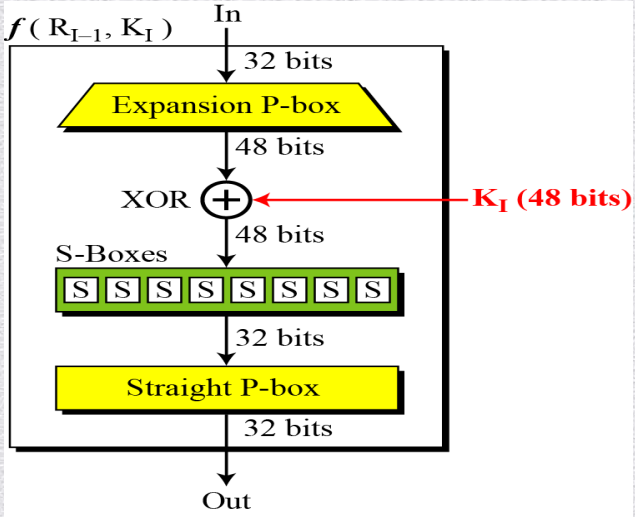
- Straight P-boxes that are inverses of each other w/out crypto significance



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

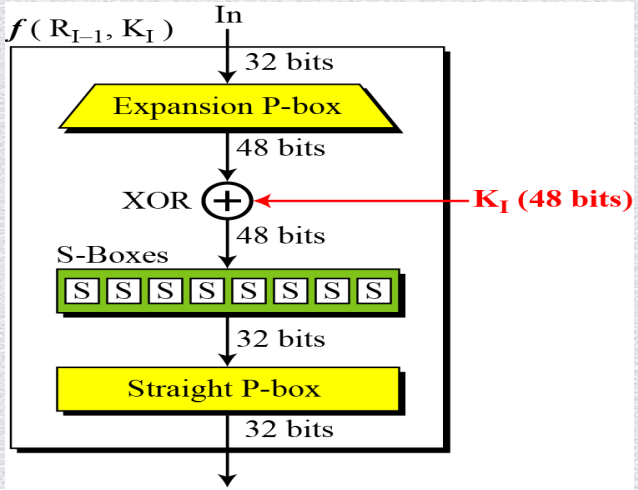


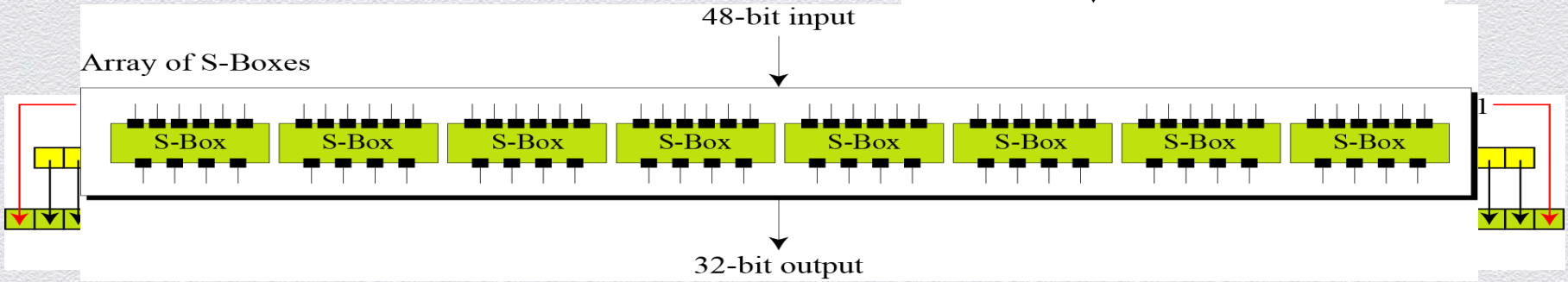


- DES uses 16 rounds, each applying a Feistel cipher
  - L(i) = R(i-1)
  - R(i) = L(i-1) XOR *f* (K(i),R(i-1)),
    where *f* applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output

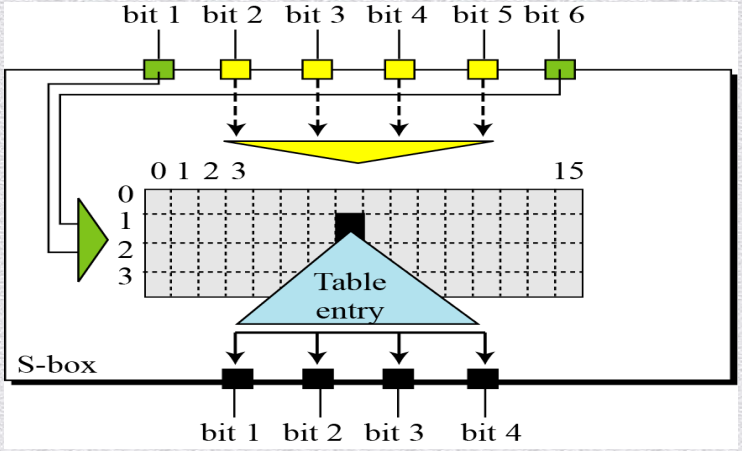
DES: Low-level view

- 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