CS579: Foundations of Cryptography

Spring 2023

Pseudorandomness

Instructor: **Nikos Triandopoulos**

STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

Randomness Vs. pseudorandomness

Randomness is important!

- the key stream in the one-time pad
- the secret key used in ciphers
- the initialization vectors (IVs) used in ciphers

Pseudo-randomness

- we need an efficiently computable process that expands a short random string x into a long string f(x) that “appears” random
- not truly random in that
  - derived by a deterministic algorithm
  - output is dependent on initial values
- “anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin” – John von Neumann
- objectives
  - fast
  - secure

Types of Pseudorandom Generators (PRGs)

- Classical PRGs
  - linear congruential generator
- Cryptographically secure PRGs
  - e.g., Blum-Micali generator

Linear congruential generator – algorithm

Based on the linear recurrence: $x_i = a\,x_{i-1} + b \mod m$ $i \geq 1$ where

- x_0 is the **seed** or start value
- a is the multiplier
- b is the increment
- m is the modulus

Output

- $(x_1, x_2, \ldots, x_k)$
- $y_i = x_i \mod 2$
- $Y = (y_1 y_2 \ldots y_k)$ ← pseudo-random sequence of K bits

Linear congruential generator - example

Let $x_n = 3\,x_{n-1} + 5 \bmod 31$, $n \geq 1$, and $x_0 = 2$

- 3 and 31 are relatively prime, one-to-one
- 31 is prime, order is 30

We have the 30 residues in a cycle

2, 11, 7, 26, 21, 6, 23, 12, 10, 4, 17, 25, 18, 28, 27, 24, 15, 19, 0, 5, 20, 3, 14, 16, 22, 9, 1, 8, 29, 30

Pseudo-random sequences of 10 bits

- when $x_0 = 2$

  01101010001

- when $x_0 = 3$

  10001101001

Linear congruential generator – security

- Fast, but insecure
  - sensitive to the choice of parameters a, b, and m
  - correlation between successive values
  - short period, often $m=2^{32}$ or $m=2^{64}$

Linear congruential generator – application

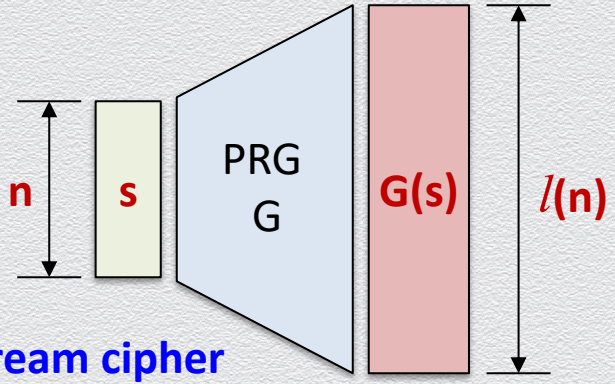
- Used commonly in compilers
  - rand()
- Not suitable for high-quality randomness applications
- Not suitable for cryptographic applications
  - need to use cryptographically secure PRGs!

Pseudorandom generators

Pseudorandom generators – PRGs

Deterministic PPT algorithm G that

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



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

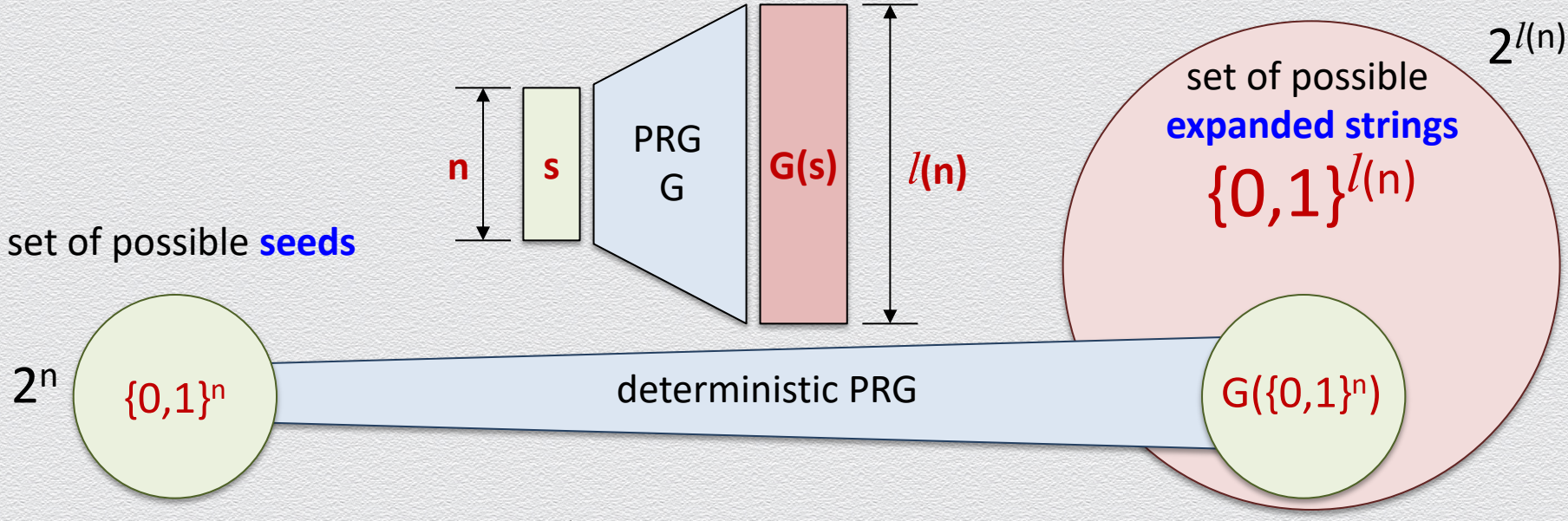
G is a PRG if:

- **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 r

PRG – visual interpretation

Suppose $s \in \{0,1\}^n$, is chosen uniformly at random – Is $G(s) \in \{0,1\}^{l(n)}$ uniformly random? No!

expansion: **low-entropy** random source → "**high-entropy**" random output

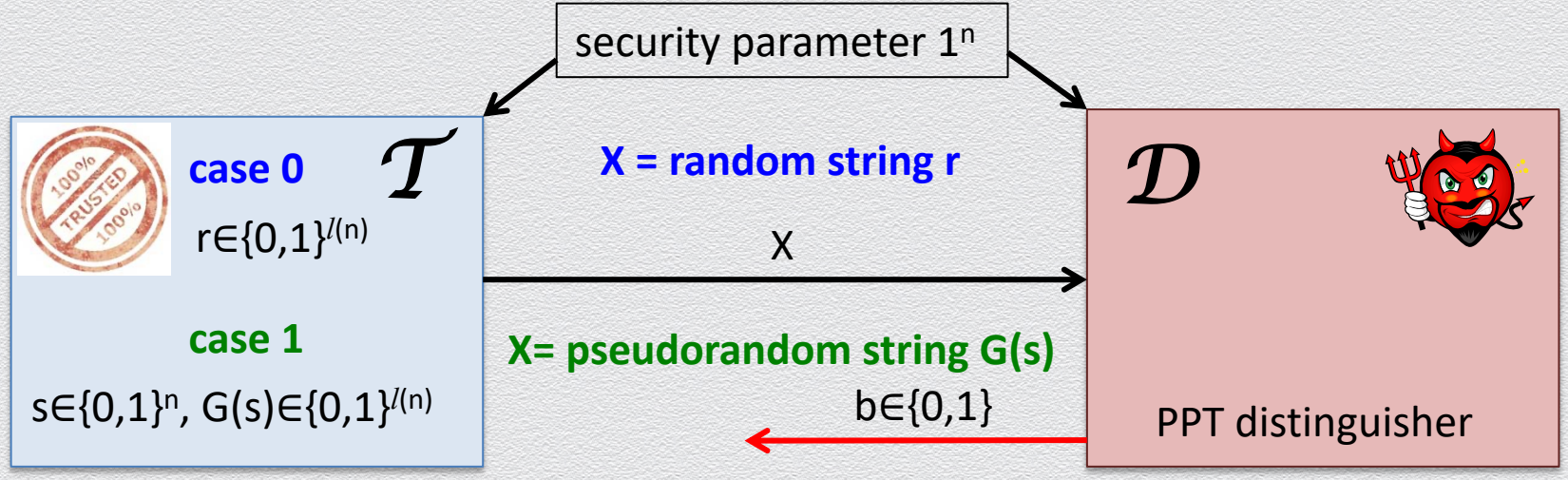


pseudorandomness: all possible $2^{l(n)}$ random strings r are "**securely represented**" by 2^n seeds

PRG – security

b = 0 when $\mathcal{D}$ thinks that its input X is random

b = 1 when $\mathcal{D}$ thinks that its input X is pseudorandom

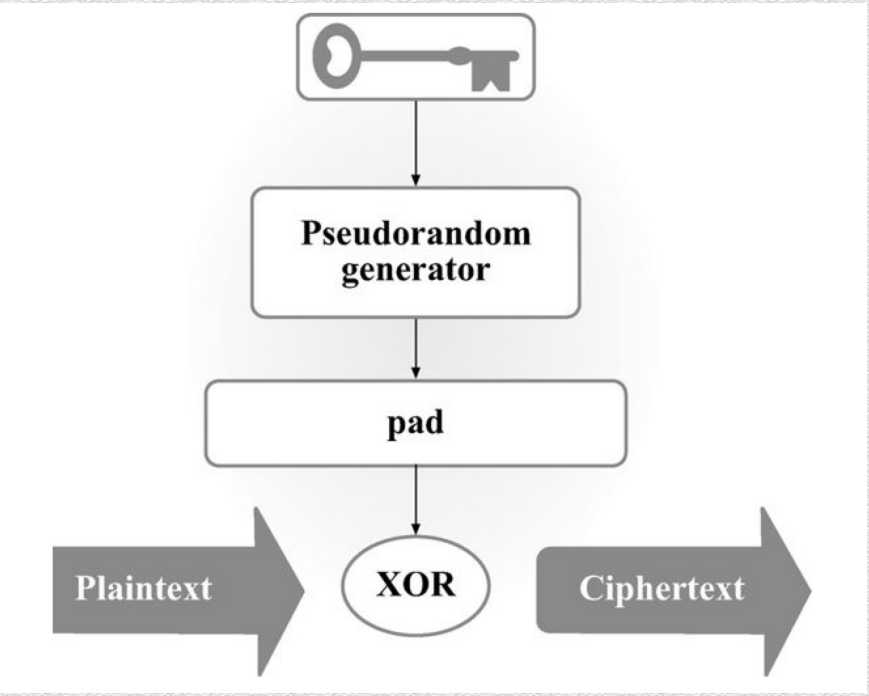


$| \Pr[\mathcal{D}(G(s)) = 1] - \Pr[\mathcal{D}(r) = 1] | \leq negl(n)$

$\mathcal{D}$ behaves the same no matter what its input is!

PRG-based symmetric-key encryption scheme

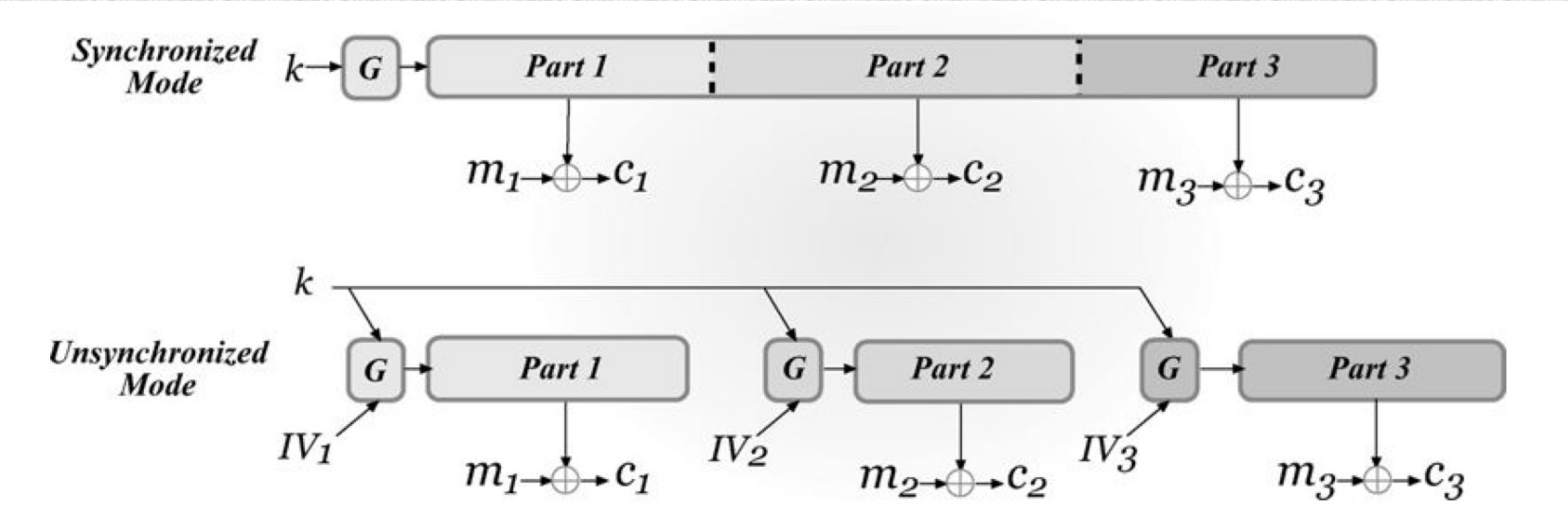
- Either fixed-length or arbitrary-length encryption scheme



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

Modes of operation for stream ciphers

on-the-fly computation of new pseudorandom bits, no IV needed, EAV-security



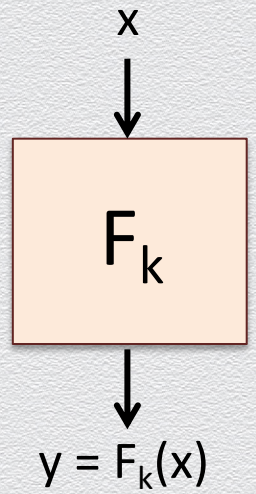
random IV used for every new message is sent along with ciphertext, CPA-security

Pseudorandom functions

Pseudorandom functions – PRFs

Generalize the concept of a PRG

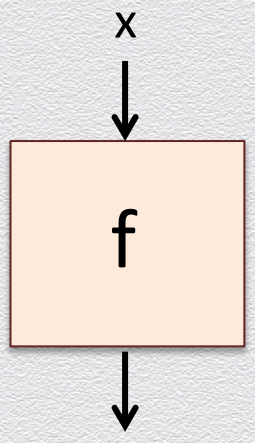
- produce pseudorandom bits that also **depend** on specific input
- **keyed functions** of the form $F_k : \{0,1\}^n \rightarrow \{0,1\}^n$



a.k.a. **block cipher**

Operate essentially as a **random** function

- F_k is PRF if it is **indistinguishable** from a truly random function f
  - e.g., f is an random permutation
- $f: \{0,1\}^n \rightarrow \{0,1\}^n$ is randomly selected for the set of all length-preserving functions mapping n-bit inputs to n-bit outputs

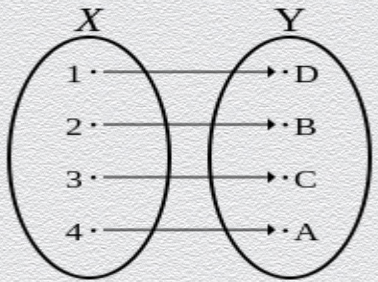


consistent, uniform

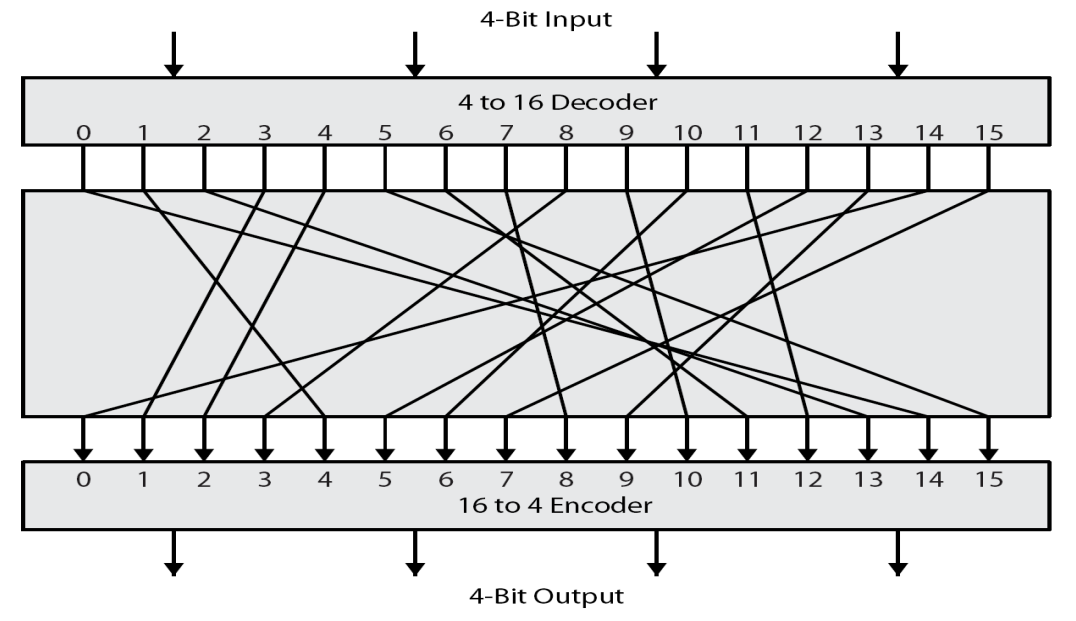
A perfect encryption of a block

- Goal: encrypt a block of n bits using the same key all the time
  - but not have the problem of one-time pad (i.e., be semantically secure)
- Approach: encryption via a bijective random mapping T from $\{0,1\}^n$ to $\{0,1\}^n$
  - mapped pairs are computed uniformly at random
  - to encrypt x, just output T[x]
  - to decrypt y, just output $T^{-1}[y]$
  - the secret key is T
- Problem with this approach: T has size ~ $n\ 2^n$
- Improvement: making it randomized (and semantically-secure)
  - pick random r & encrypt x as: (y = T[r] XOR x, r)
  - decrypt (y,r) as: y XOR T[r]

a mapping T

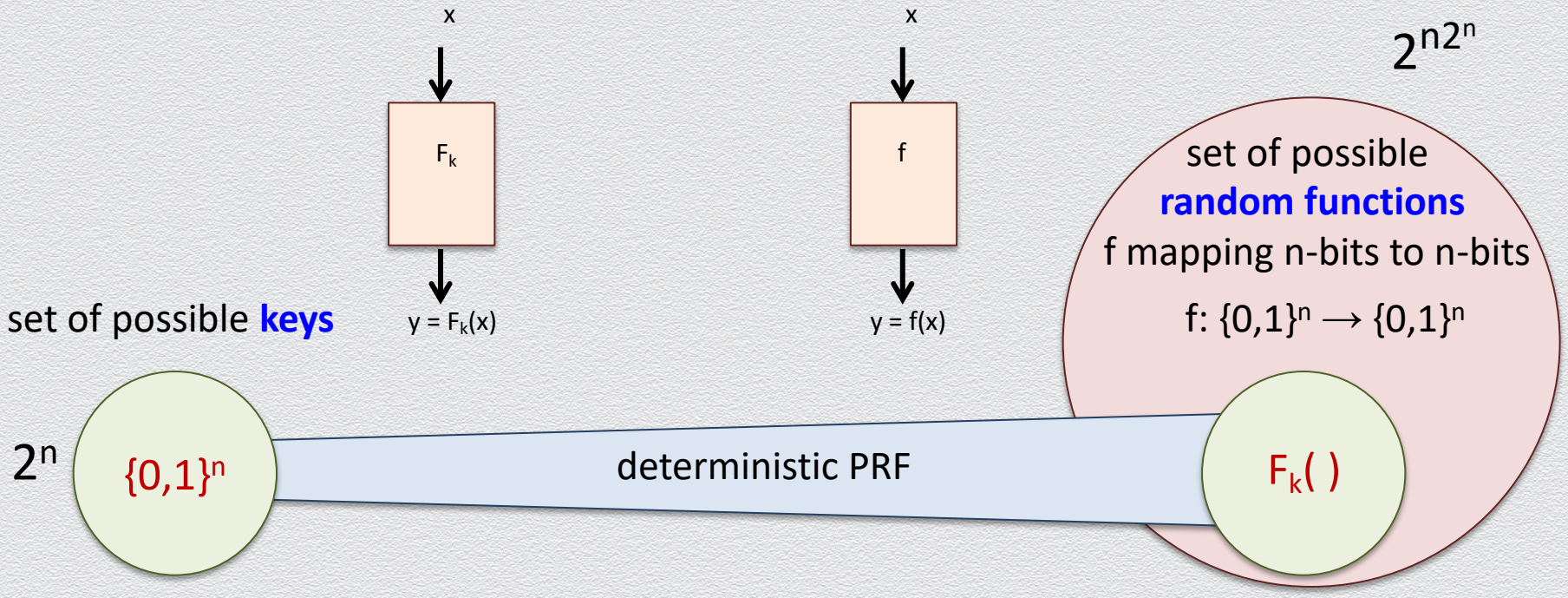


Ideal block cipher



PRF – visual interpretation

Suppose $k \in \{0,1\}^n$ is chosen uniformly at random

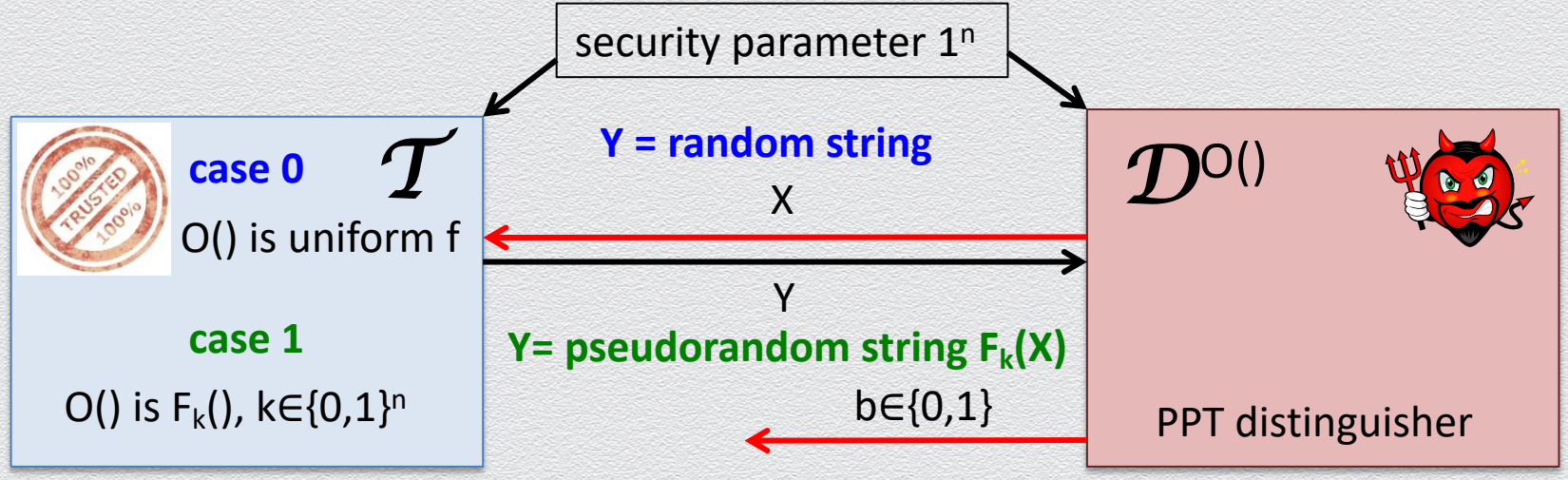


pseudorandomness: all possible random functions f are "**securely represented**" by 2^n keys

PRF – security

b = 0 when $\mathcal{D}$ thinks that its oracle is f()

b = 1 when $\mathcal{D}$ thinks that its oracle is $F_k()$

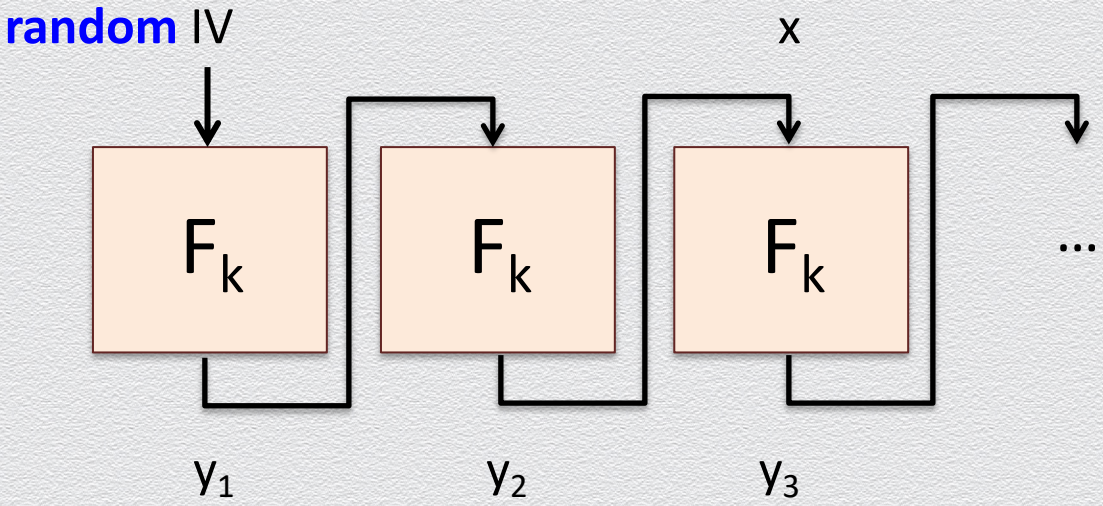


$\mathcal{D}$ behaves the same no matter what its oracle is!

$$| \Pr[\mathcal{D}^{F(k,\cdot)}(1^n) = 1] - \Pr[\mathcal{D}^{f()}(1^n) = 1] | \leq \text{negl}(n)$$

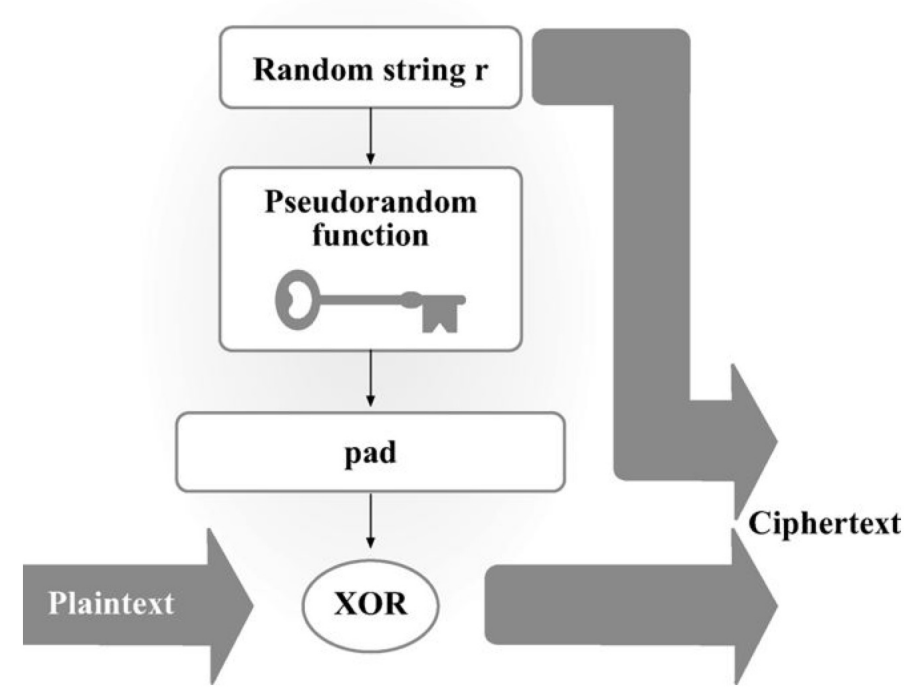
PRFs imply PRGs

- via chaining



PRF-based symmetric-key encryption scheme

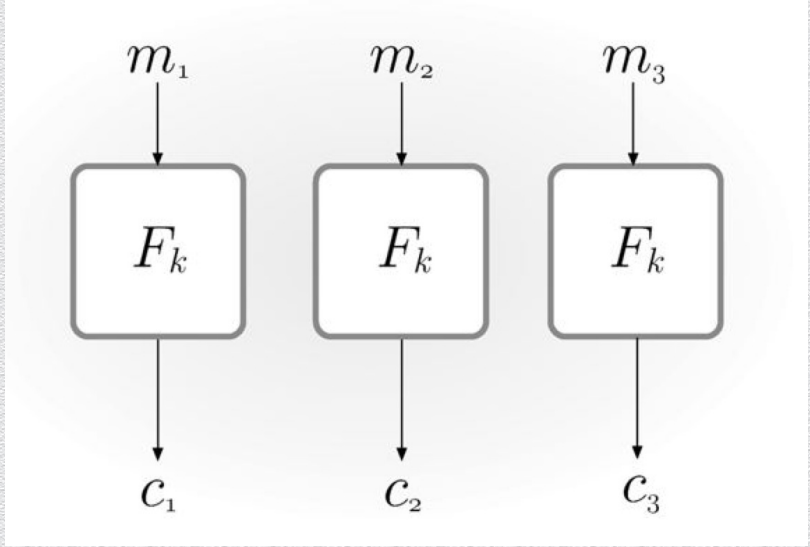
- Fixed-length encryption scheme



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

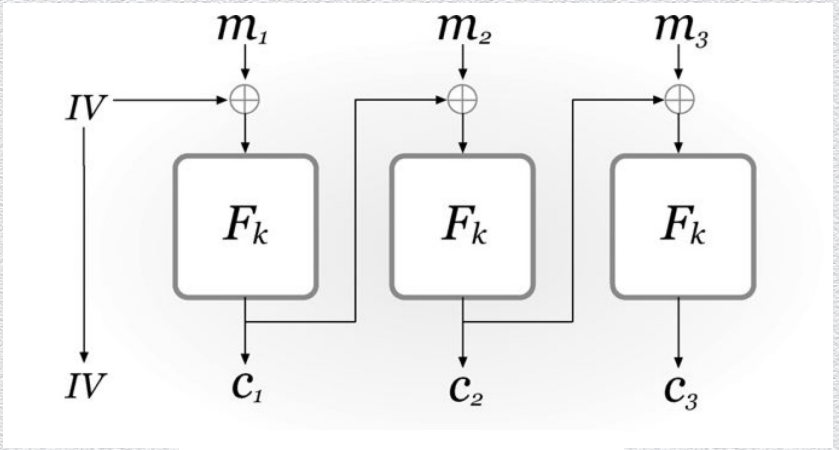
Modes of operation for block ciphers (I)

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

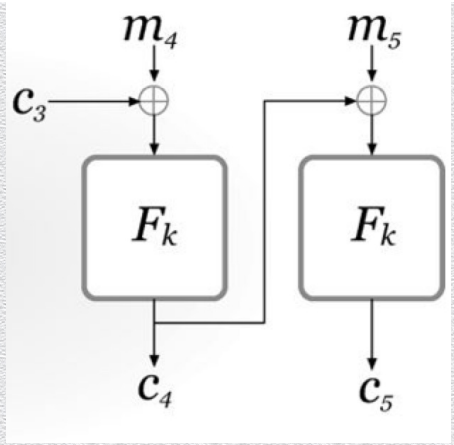


Modes of operation for block ciphers (II)

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

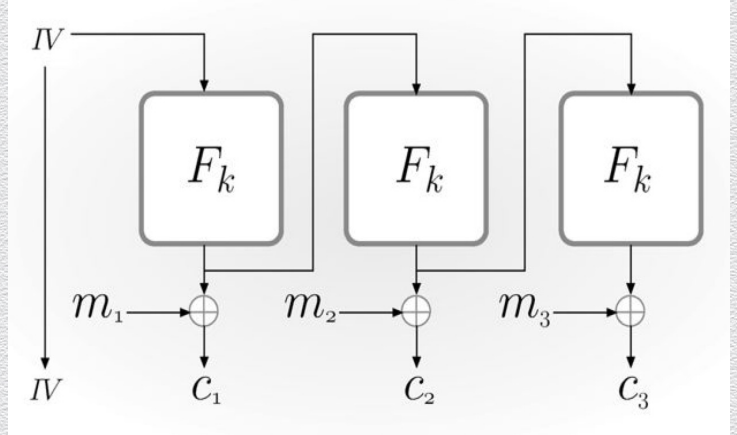


- Chained CBC
  - use last block ciphertext of current message as IV of next message
  - saves bandwidth but not CPA-secure



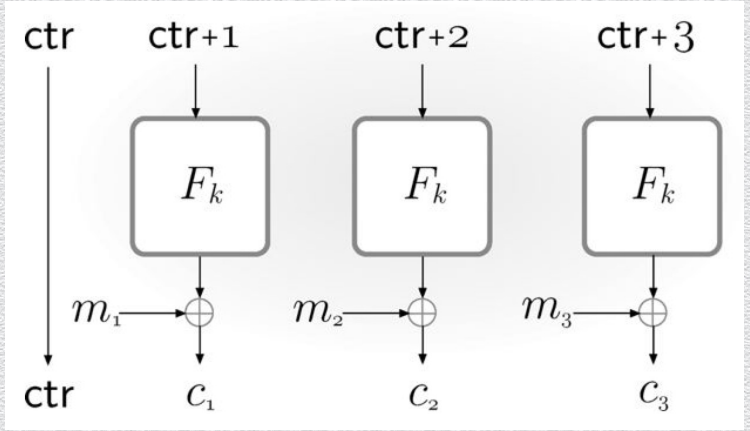
Modes of operation for block ciphers (III)

- OFB – output feedback
  - IV uniform
  - no need message length to be multiple of n
  - resembles unsynchronized stream-cipher mode
  - stateful variant (chaining) is secure
  - CPA-secure if F_k is PRF



Modes of operation for block ciphers (IV)

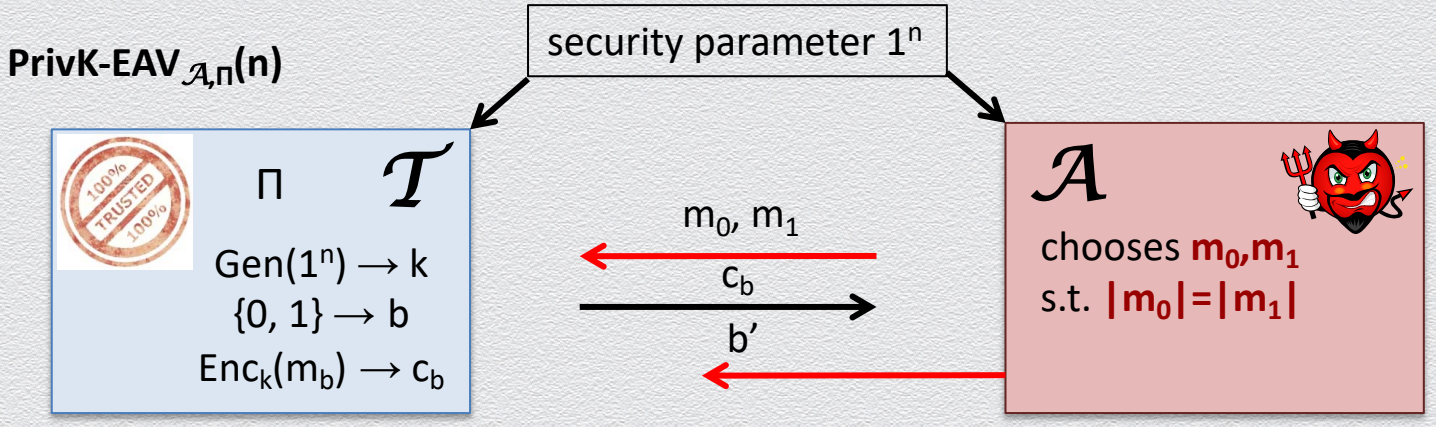
- CTR – counter mode
  - ctr uniform
  - no need message length to be multiple of n
  - resembles synchronized stream-cipher mode
  - CPA-secure if F_k is PRF
  - no need for Fk to be invertible
  - parallelizable



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 uniformly random
  - in this case, CBC is a better option than OFB/CTR

Recall: EAV-security for secrecy



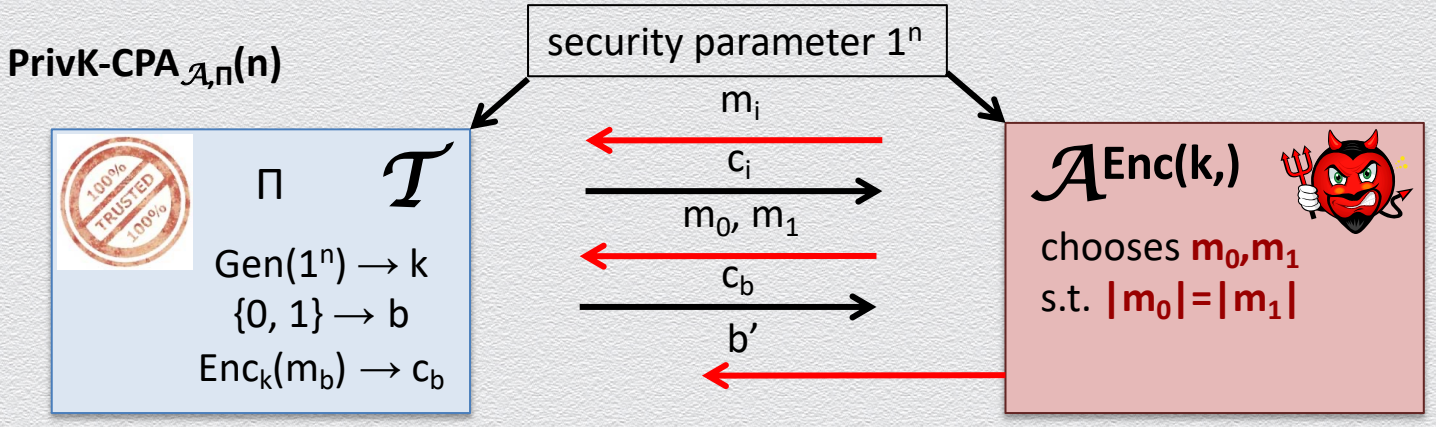
Security definition

We say that Π = (Enc,Dec) is **EAV-secure**, if ∀PPT adversaries $\mathcal{A}$, ∃a negligible function ε(n) such that it holds that

$$\Pr[\text{PrivK-EAV}_{\mathcal{A},\Pi}(n)=1] \leq \tfrac{1}{2} + \varepsilon(n)$$

I.e., no PPT $\mathcal{A}$ computes b correctly non-negligibly better than randomly guessing

Recall: CPA-security for secrecy



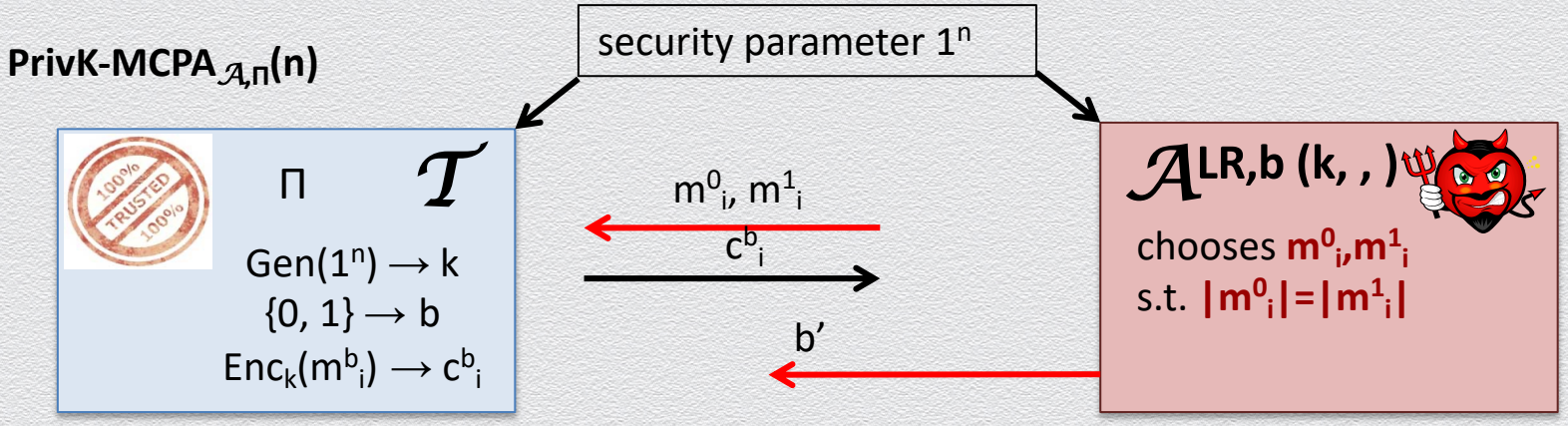
Security definition

We say that Π = (Enc,Dec) is **CPA-secure**, if ∀PPT adversaries $\mathcal{A}$, ∃a negligible function ε(n) such that it holds that

$$\Pr[\text{PrivK-CPA}_{\mathcal{A},\Pi}(n)=1] \leq \tfrac{1}{2} + \varepsilon(n)$$

I.e., no PPT $\mathcal{A}$ computes b correctly non-negligibly better than randomly guessing, **even when it learns the encryptions of messages of its choice**

CPA-security for secrecy for multiple messages



Security definition

We say that Π = (Enc,Dec) is **CPA-secure for multiple encryptions**, if ∀PPT adversaries $\mathcal{A}$, ∃a negligible function ε(n) such that it holds that

$$\Pr[\text{PrivK-MCPA}_{\mathcal{A},\Pi}(n)=1] \leq \tfrac{1}{2} + \varepsilon(n)$$

I.e., no PPT $\mathcal{A}$ computes **b for many challenge ciphertexts** correctly non-negligibly better than randomly guessing, even when it learns the encryptions of messages of its choice

On CPA and multi-message encryption security

Facts

- CPA security implies probabilistic encryption
- EAV-security for multiple messages implies probabilistic encryption
- Any encryption scheme that is CPA-secure is also CPA-secure for multiple encryptions

Block ciphers in practice

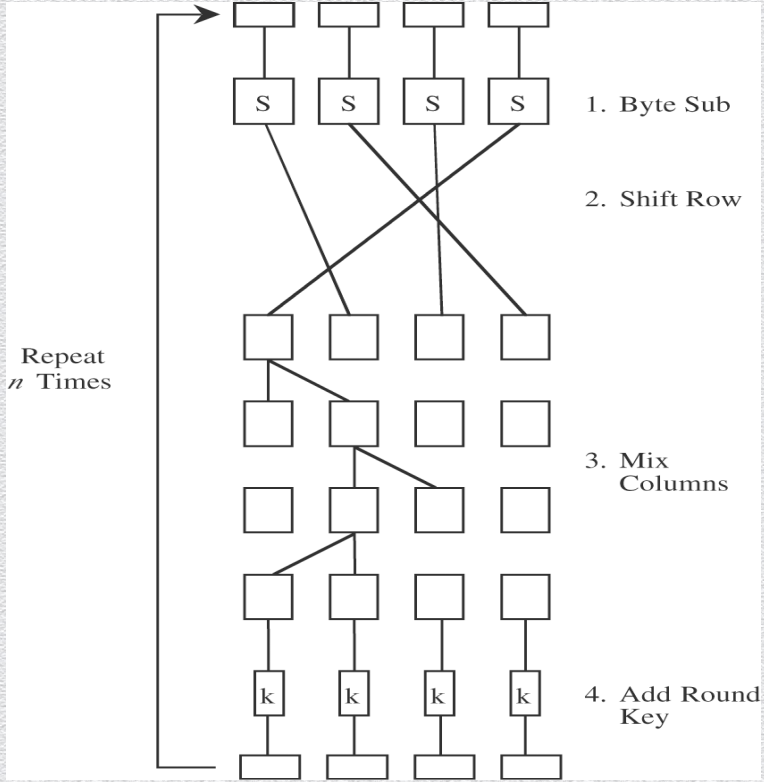
DES: The Data Encryption Standard

- Symmetric block cipher
- Developed in 1976 by IBM for the US National Institute of Standards and Technology (NIST)

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

AES: Advanced Encryption System

- Symmetric block cipher
- Developed in 1999 by independent Dutch cryptographers
- Still in common use



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

DES: The Data Encryption Standard

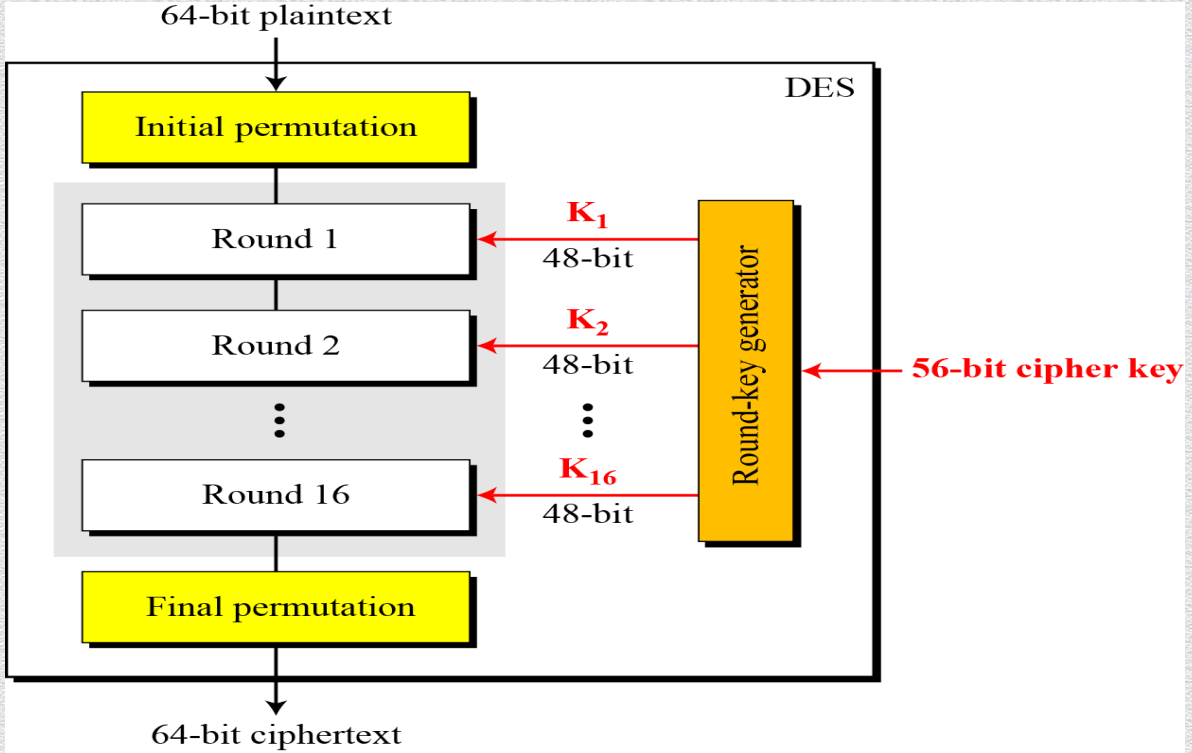
- Symmetric block cipher
- Developed in 1976 by IBM for the US National Institute of Standards and Technology (NIST)
- Employs substitution & transposition, on top of each other, for 16 rounds
  - block size = 64 bits, key size = 56 bits
- Strengthening (since 56-bit security is not considered adequately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

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

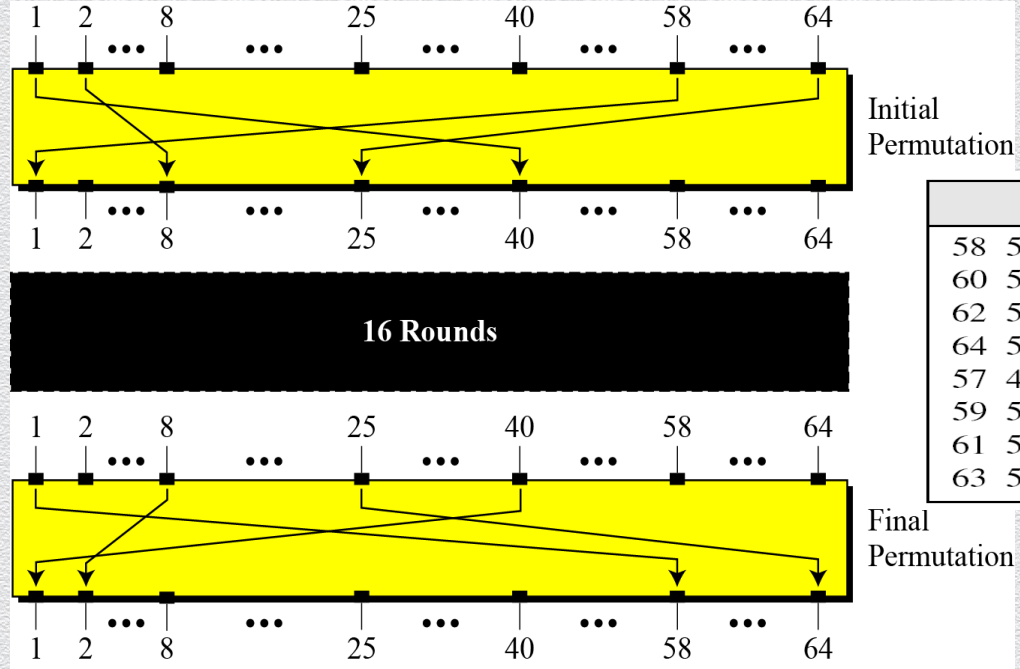
High-level DES view

DES structure



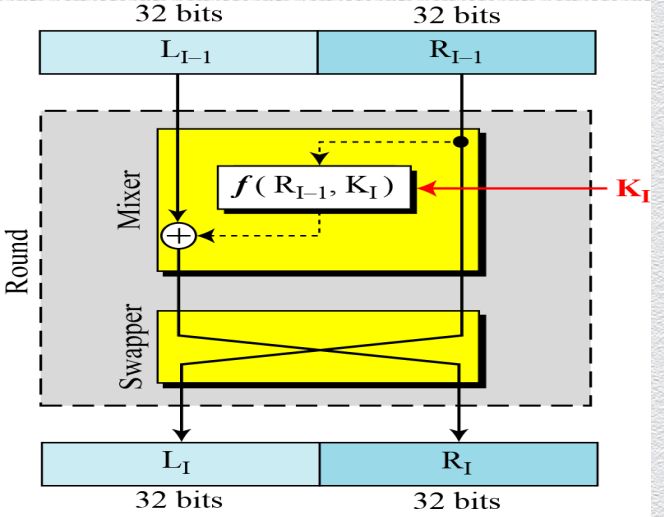
Initial and final permutations

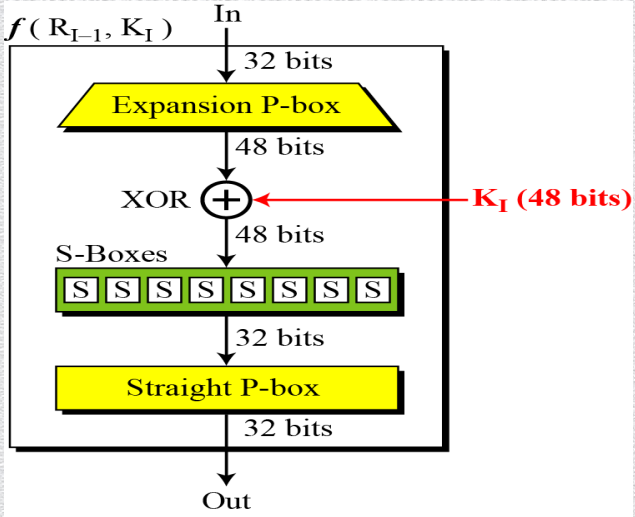
- Straight P-boxes that are inverses of each other w/out crypto significance



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: Feistel network

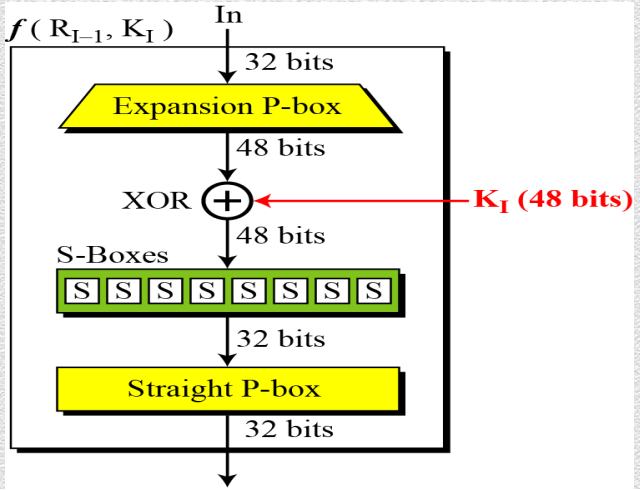


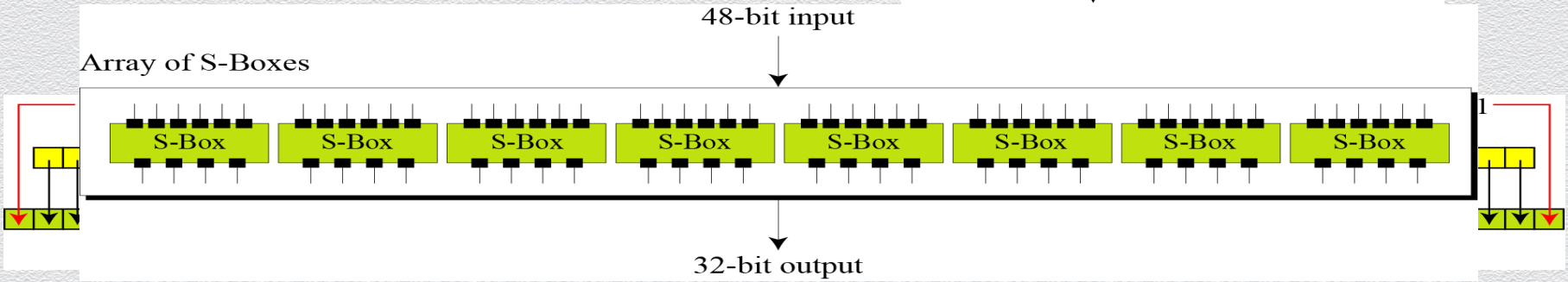


- DES uses 16 rounds, each applying a Feistel cipher
  - L(i) = R(i-1)
  - R(i) = L(i-1) XOR *f* (K(i),R(i-1)),
    where *f* applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output

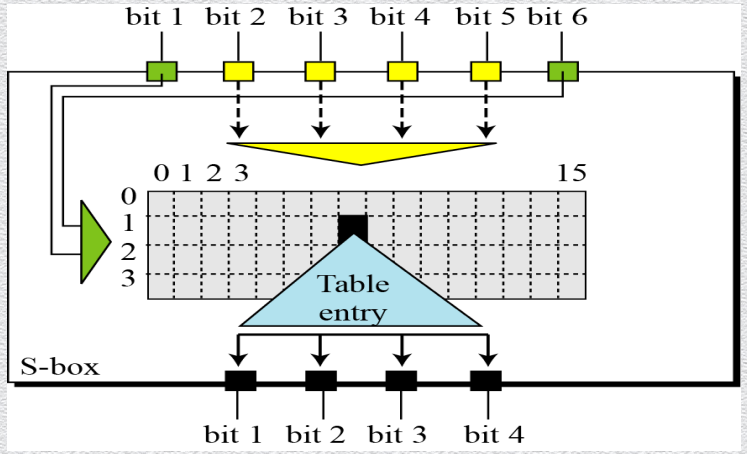
Low-level DES view

- 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





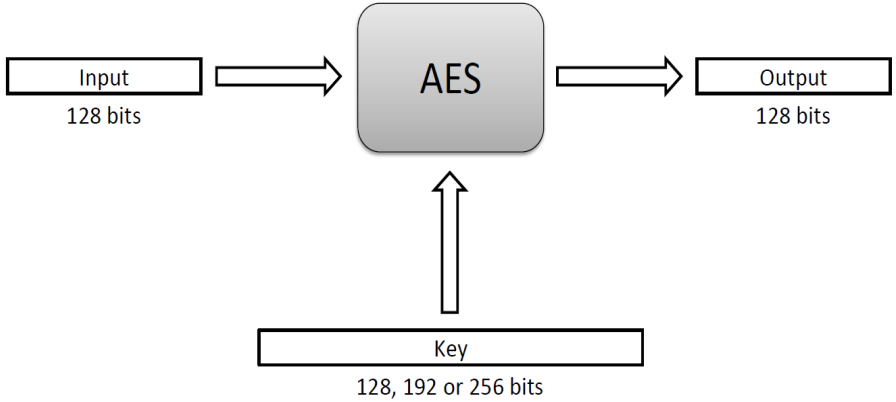
S-box in detail



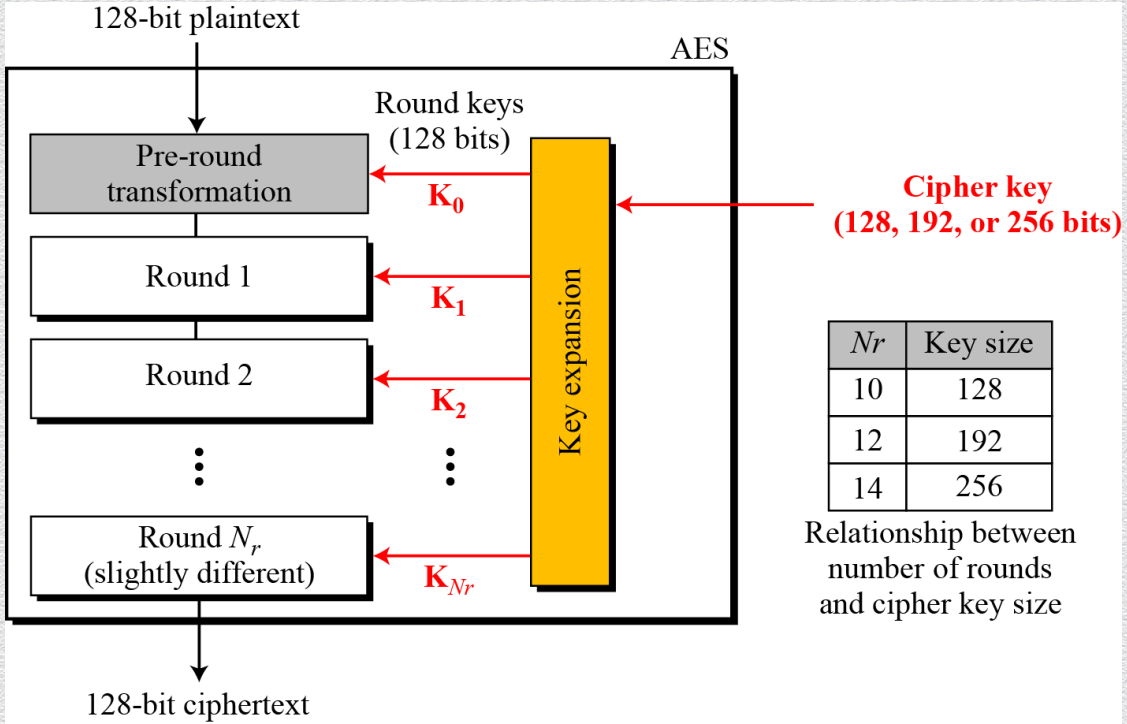
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

AES: Advanced Encryption System

- Symmetric block cipher (called Rijndael)
  - developed in 1999 by independent Dutch cryptographers in response to the 1997 NIST's public call for a replacement to DES
- Employs substitution, confusion & diffusion
  - on blocks of 128 bits, in 10, 12 or 14 rounds for keys of 128, 192, 256 bits
  - depending on key size, yields ciphers known as AES-128, AES-192, and AES-256
- Still in common use
  - on the longevity of AES
    - larger key sizes possible to use
    - not known serious practical attacks



AES structure



Cipher key
(128, 192, or 256 bits)

Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size