NETWORK SECURITY ESSENTIALS CHAPTER 2

by William Stallings
(Inspired by Lecture slides by Lawrie Brown)
Lecturer: Taghinezhad
Site:ataghinezhad.github.io
Email:a0taghinezhad@gmail.com

OUTLINE

Symmetric encryption

Block encryption algorithms

Stream ciphers

Block cipher modes of operations

SYMMETRIC ENCRYPTION(SINGLE KEY ENCRYPTION)

- or conventional / private-key / single-key
- **sender and recipient share a common key**
- all classical encryption algorithms are private-key
- was only type prior to invention of public-key in 1970's
- and by far most widely used

SOME BASIC TERMINOLOGY

- **plaintext** - original message
- **ciphertext** - coded message
- **cipher** - algorithm for transforming plaintext to ciphertext
- **key** - info used in cipher known only to sender/receiver
- **encipher (encrypt)** - converting plaintext to ciphertext
- **decipher (decrypt)** - recovering ciphertext from plaintext
- **cryptography** - study of encryption principles/methods
- **cryptanalysis (codebreaking)** - study of principles/ methods of deciphering ciphertext *without* knowing key
- **cryptology** - field of both cryptography and cryptanalysis

SYMMETRIC ENCRYPTION

A symmetric encryption scheme has **five ingredients**

1. **Plaintext**: This is the original message or data that is fed into the algorithm as input.

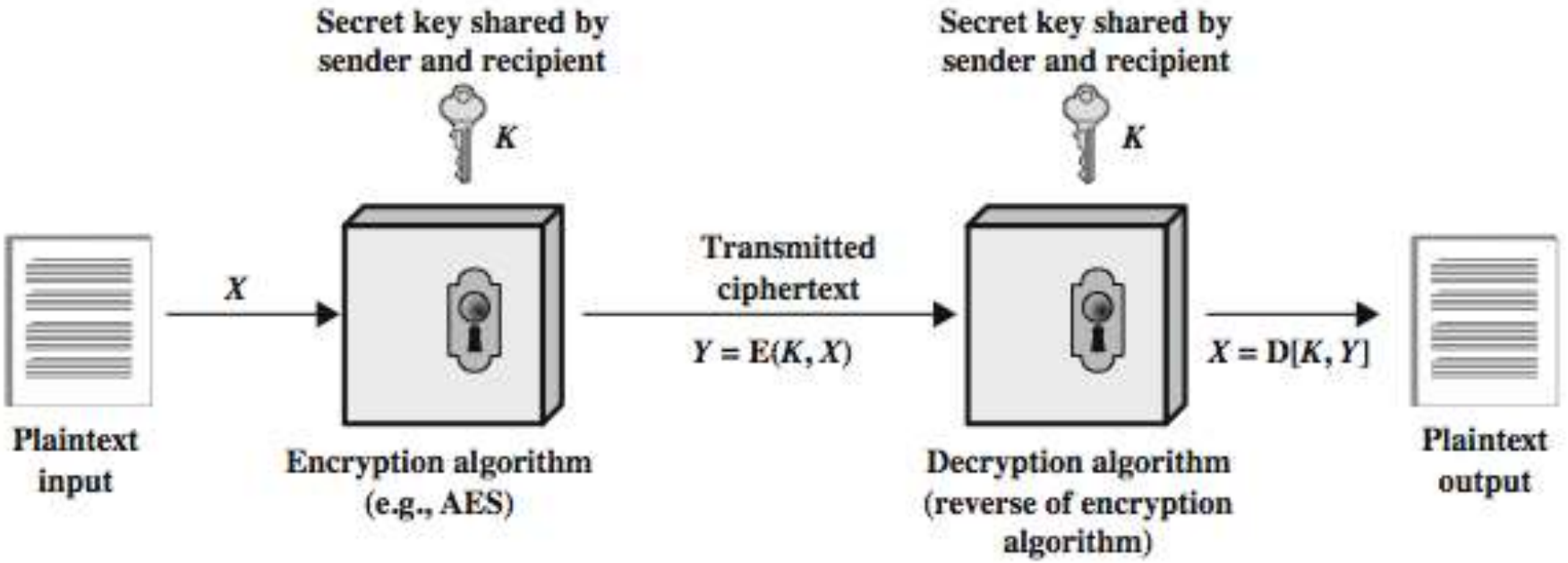
2. **Encryption algorithm**: The encryption algorithm performs various substitutions and transformations on the plaintext.

3. **Secret key**: The secret key is also input to the algorithm. The exact substitutions and transformations performed by the algorithm depend on the key

4. **Ciphertext**: This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts.

5. **Decryption algorithm**: This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the same secret key and produces the original plaintext.

SYMMETRIC CIPHER MODEL



REQUIREMENTS

- two requirements for secure use of symmetric encryption:
  - a **strong encryption algorithm**
  - **a secret key known only to sender / receiver**
- mathematically have:

  $Y = E(K, X)$

  $X = D(K, Y)$
- assume encryption algorithm is known
- implies a secure channel to distribute key

CRYPTOGRAPHY

- can characterize cryptographic system by 3 dimensions:
  - **type of encryption** operations used
    - Substitution جایگزینی
    - Transposition جابجایی
    - Product سیستم های ترکیبی
  - **number of keys** used
    - single-key or private
    - two-key or public
  - **way** in which **plaintext** is **processed**
    - Block رمز قالبی یا بلوکی
    - Stream رمز دنباله ای

کشف رمز CRYPTANALYSIS

- objective to recover key not just message
- general approaches:
  - cryptanalytic attack حمله تحلیلگر رمز
  - brute-force attack حمله جستجوی جامع
- if either succeed all key use compromised

CRYPTANALYTIC ATTACKS

- **ciphertext only**
  - only know algorithm & ciphertext, is statistical, know or can identify plaintext
- **known plaintext**
  - **know/suspect plaintext & ciphertext**
- **chosen plaintext**
  - select plaintext and obtain ciphertext
- **chosen ciphertext**
  - select ciphertext and obtain plaintext
- **chosen text**
  - select plaintext or ciphertext to en/decrypt

نوع حمله	آنچه برای کشف کنندهٔ رمز معلوم است
فقط- متن رمزشده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود
متن سادهٔ معلوم	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • یک یا چندجفت متن ساده- متن رمزشده بتوسط کلید سرّی
متن سادهٔ انتخاب شده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • پیام سادهٔ انتخاب شده بتوسط کشف‌رمزکننده، بهمراه متن رمزشدهٔ نظیر آن بتوسط کلید سرّی
متن رمزشدهٔ انتخاب شده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • متن رمزشدهٔ انتخاب‌شده بتوسط شکنندهٔ رمز، بهمراه متن رمزگشائی شده نظیر آن با استفاده از کلید سرّی
متن انتخاب شده	• الگوریتم رمزنگاری • متن رمزشده‌ای که باید باز شود • پیام سادهٔ انتخاب‌شده بتوسط کشف‌رمزکننده، بهمراه متن رمزشدهٔ نظیرآن با استفاده از کلید سرّی • متن رمزشدهٔ انتخاب‌شده بتوسط کشف‌رمزکننده، بهمراه متن رمزگشائی شده نظیر آن با استفاده از کلید سرّی

ENCRYPTION SCHEME

- An encryption scheme: computationally secure if
  - The **cost** of breaking the cipher **exceeds** the **value** of information
  - The **time** required to break the cipher **exceeds** the **lifetime** of information
- **very difficult to estimate** the amount of effort required to cryptanalyze ciphertext successfully

BRUTE FORCE SEARCH

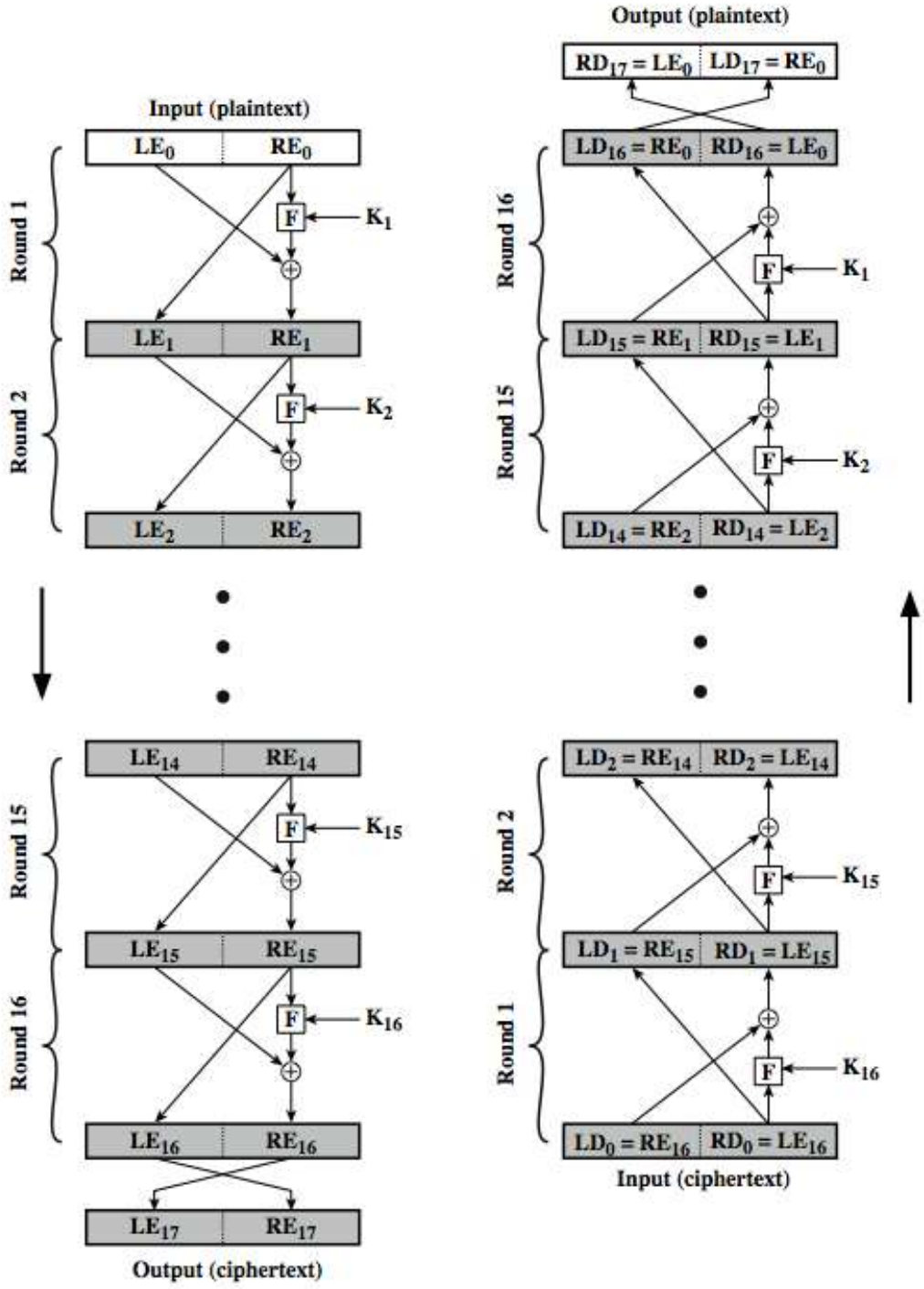
- always possible to simply **try every key**
- most **basic** attack, **proportional to key size**
- **assume** either **know / recognise plaintext**

BRUTE FORCE SEARCH

اندازهٔ کلید (bits)	تعداد کلیدهای ممکن	زمان لازم کشف رمز با یک رمزگشائی در هر میکروثانیه	زمان لازم کشف رمز با یک میلیون رمزگشائی در هر میکروثانیه
۳۲	$۲^{۳۲} = ۴/۳ \times ۱۰^{۹}$	$۲^{۳۱}$ میکروثانیه = ۳۵/۸ دقیقه	۲/۱۵ میلی ثانیه
۵۶	$۲^{۵۶} = ۷/۲ \times ۱۰^{۱۶}$	$۲^{۵۵}$ میکروثانیه = ۱۱۴۲ سال	۱۰/۰۱ ساعت
۱۲۸	$۲^{۱۲۸} = ۳/۴ \times ۱۰^{۳۸}$	$۲^{۱۲۷}$ میکروثانیه = $۵/۴ \times ۱۰^{۲۴}$ سال	$۵/۴ \times ۱۰^{۱۸}$ سال
۱۶۸	$۲^{۱۶۸} = ۳/۷ \times ۱۰^{۵۰}$	$۲^{۱۶۷}$ میکروثانیه = $۵/۹ \times ۱۰^{۳۶}$ سال	$۵/۹ \times ۱۰^{۳۰}$ سال
۲۶ کاراکتر (جایگشت)	$۲۶ ! = ۴ \times ۱۰^{۲۶}$	۲×۱۰^{۲۶} میکروثانیه = $۶/۴ \times ۱۰^{۱۲}$ سال	$۶/۴ \times ۱۰^{۶}$ سال

FEISTEL CIPHER STRUCTURE ساختار رمز فیستل

- Horst Feistel ('hȯrst faɪstɪl') devised the **feistel cipher**
  - based on concept of invertible product cipher
- partitions input block into two halves (halfs)
  - process through **multiple rounds** which
  - perform a **substitution** on left data half
  - based on **round function of right half** & **subkey**
  - then have **permutation swapping** halves
- implements Shannon's S-P net concept



FEISTEL CIPHER STRUCTURE

FEISTEL CIPHER DESIGN ELEMENTS

- block size: 128 bits increasing size improves security, but slows cipher
- key size: 128 bits
- number of rounds: 16
- subkey generation algorithm
- round function
- fast software en/decryption
- ease of analysis

FEISTEL CIPHER DESIGN ELEMENTS

- The exact realization of a Feistel network depends on the choice of the following parameters and design features:
- block size - increasing size improves security, but slows cipher
- key size - increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- number of rounds - increasing number improves security, but slows cipher
- subkey generation algorithm - greater complexity can make analysis harder, but slows cipher
- round function - greater complexity can make analysis harder, but slows cipher
- fast software en/decryption - more recent concern for practical use
- ease of analysis - for easier validation & testing of strength

SYMMETRIC BLOCK CIPHER ALGORITHMS

- DES (Data Encryption Standard)
- 3DES (Triple DES)
- AES (Advanced Encryption Standard)

DATA ENCRYPTION STANDARD (**DES**)

- most **widely used block cipher** in world
- adopted in 1977 by NBS (now NIST)
  - as FIPS PUB 46
- encrypts 64-bit data using 56-bit key
- has widespread use
- has considerable controversy over its security

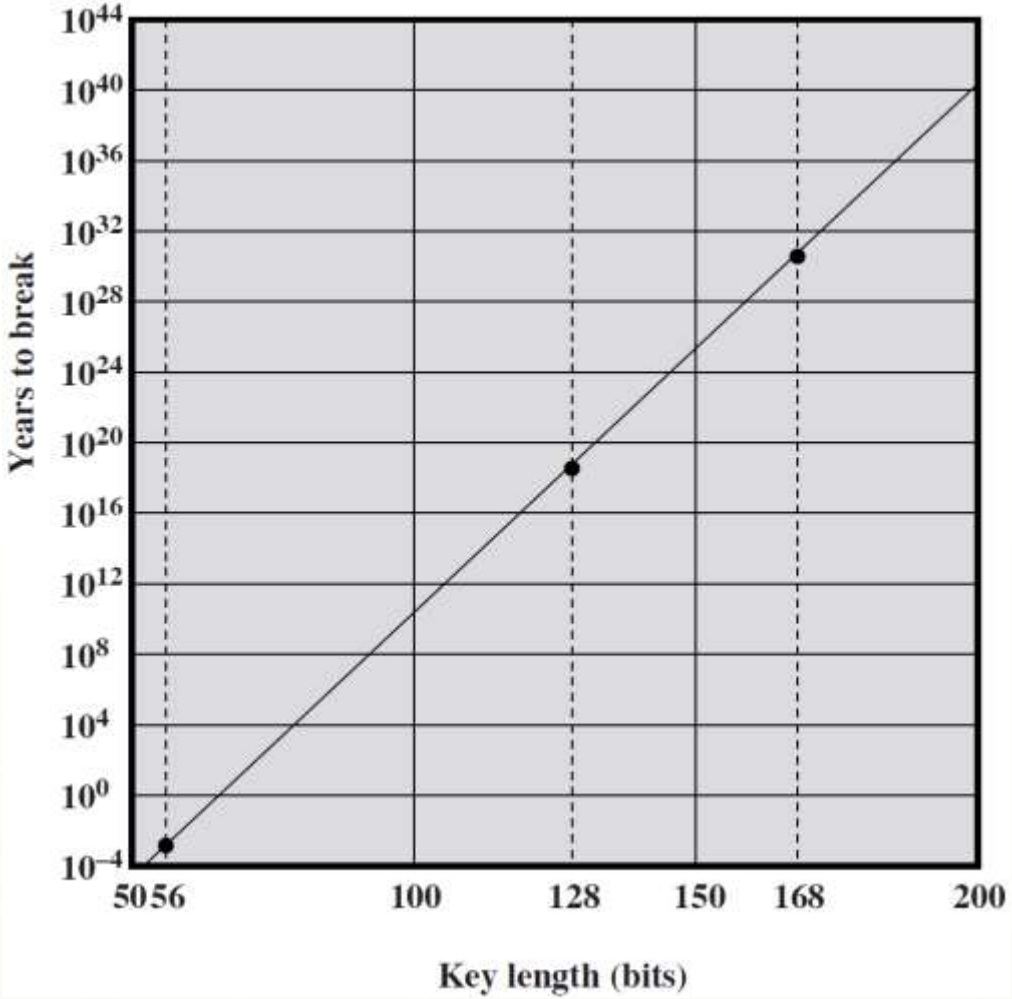
DES HISTORY

- **IBM** developed **Lucifer cipher**
  - by team led by Feistel in late 60's
  - used 64-bit data blocks with 128-bit key
- then **redeveloped** as a **commercial cipher** with input from NSA and others
- in 1973 NBS issued **request** for **proposals** for a national cipher standard
- **IBM submitted** their **revised Lucifer** which was eventually accepted as the **DES**

DES DESIGN CONTROVERSY
نگرانی طراحی

- DES standard is public had **considerable** controversy over **design**
  - in choice of **56-bit key (vs Lucifer 128-bit)**
  - and because **design criteria were classified**
- subsequent events and **public analysis show in fact design was appropriate**
- **DES** is widely **used**
  - especially in **financial** applications
  - still standardised for **legacy application**

TIME TO BREAK A DES CODE (ASSUMING 10^6 DECRYPTIONS/μS)



- With new supercomputers DES with 56bits key can be easily cracked.

Table 2.2 Average Time Required for Exhaustive Key Search

Key Size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 Decryptions/s	Time Required at 10^{13} Decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	2^{167} ns = 5.8×10^{33} years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	2^{191} ns = 9.8×10^{40} years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	2^{255} ns = 1.8×10^{60} years	1.8×10^{56} years

MULTIPLE ENCRYPTION & DES

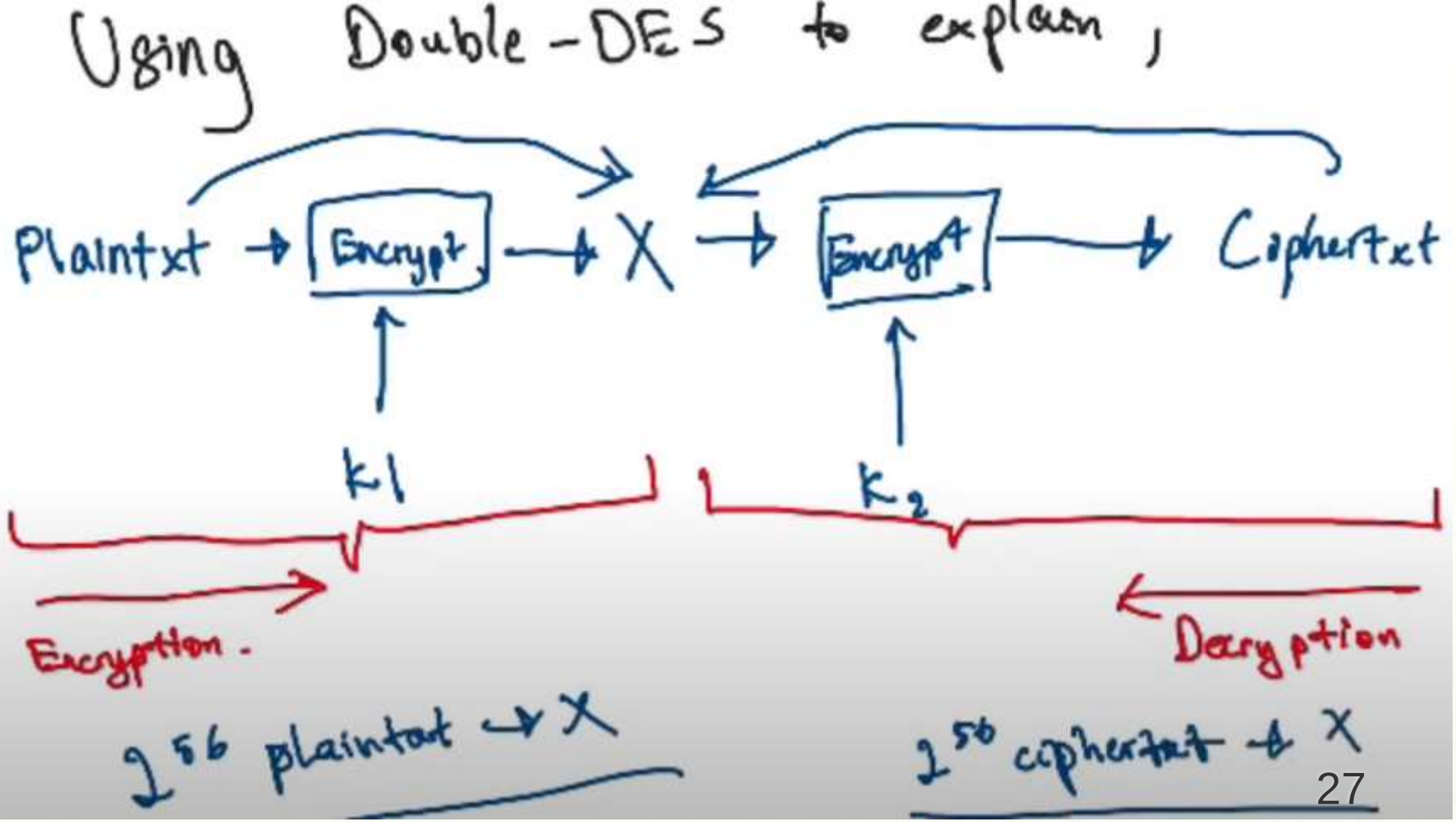
- a replacement for **DES** was **needed**
  - **theoretical attacks that can break it**
  - demonstrated **exhaustive key search** attacks
- **AES** is a **new cipher alternative**
  - prior to this alternative was to use **multiple encryption with DES implementations**
  - Triple-DES is the chosen form

DOUBLE-DES?

- could use 2 DES encrypts on each block
  - $C = E_{K2}(E_{K1}(P))$
- issue of reduction to single stage
- and have "meet-in-the-middle" attack
  - works whenever use a cipher twice
  - E(P,K_1)-> X->E(X, K_2)->C
  - since $X = E_{K1}(P) = D_{K2}(C)$
  - attack by encrypting P with all keys and store
  - then decrypt C with keys and match X value
  - Takes 2.2^{56}
  - takes $O(2^{56})$ steps

MEET IN THE MIDDLE

- Known plaintext attack



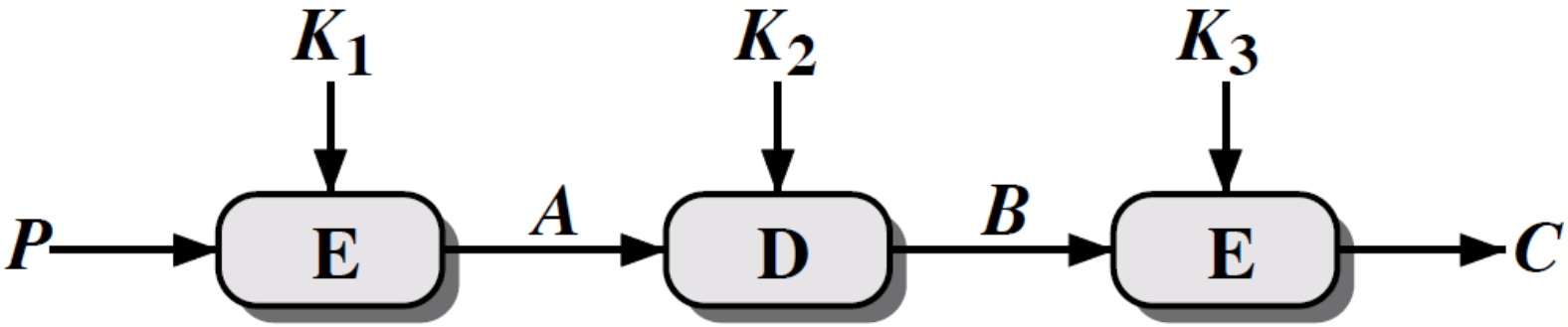
TRIPLE-DES WITH TWO-KEYS

- hence must use 3 encryptions
  - would seem to need 3 distinct keys (3 times slower)
- but can use 2 keys with E-D-E sequence
  - $C = E_{K1}(D_{K2}(E_{K1}(P)))$
  - nb encrypt & decrypt equivalent in security
  - if K1=K2 then can work with single DES
- standardized in ANSI X9.17 & ISO8732
- no current known practical attacks
  - several proposed impractical attacks might become basis of future attacks

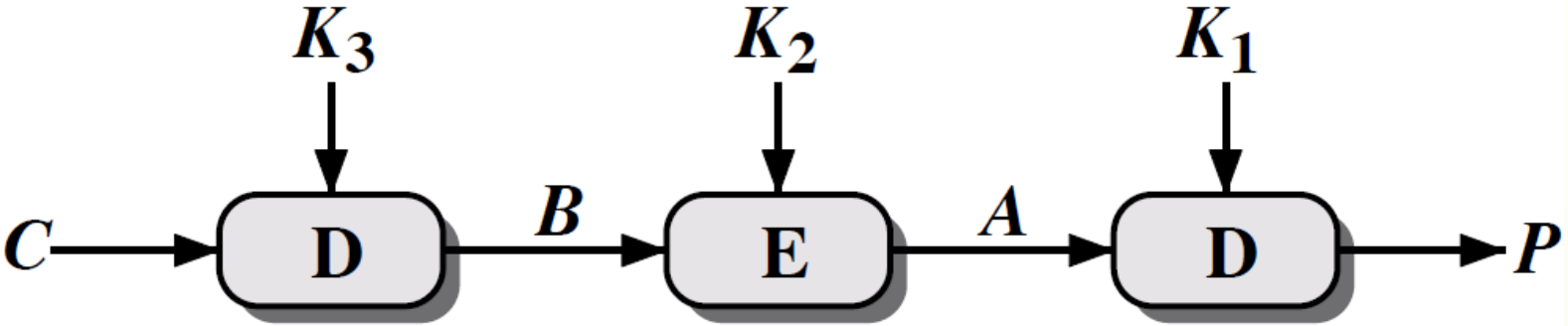
TRIPLE-DES WITH THREE-KEYS

- although no practical attacks on two-key Triple-DES have some concerns
  - Two-key: key length = 56*2 = 112 bits
  - Three-key: key length = 56*3 = 168 bits
- can use Triple-DES with Three-Keys to avoid even these
  - $C = E_{K3}(D_{K2}(E_{K1}(P)))$
- has been adopted by some Internet applications, eg PGP, S/MIME

TRIPLE DES



(a) Encryption



(b) Decryption

ORIGINS

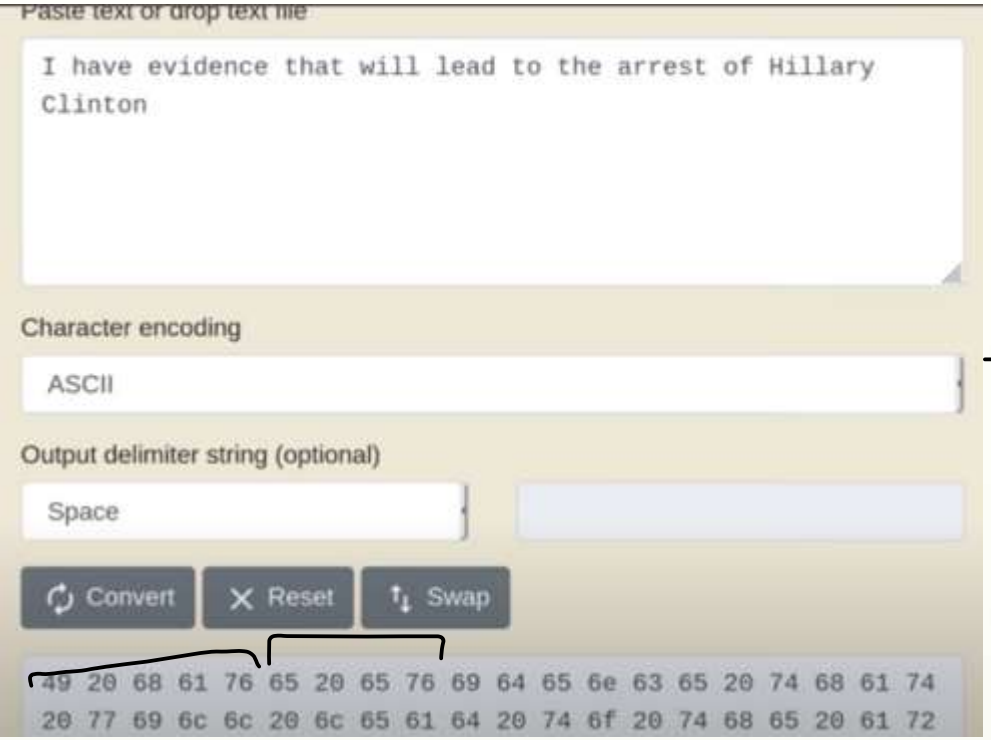
- clearly a replacement for DES was needed
  - have theoretical attacks that can break it
  - have demonstrated exhaustive key search attacks
- can use **Triple-DES – but slow**, has small blocks
- **US NIST issued call for ciphers in 1997**
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99
- **RijnDael** was selected as the AES in Oct-2000 **DEVELOPED BY Dr. Joan Daemen and Dr.Vincent Rijmen**
- issued as FIPS PUB 197 standard in Nov-2001

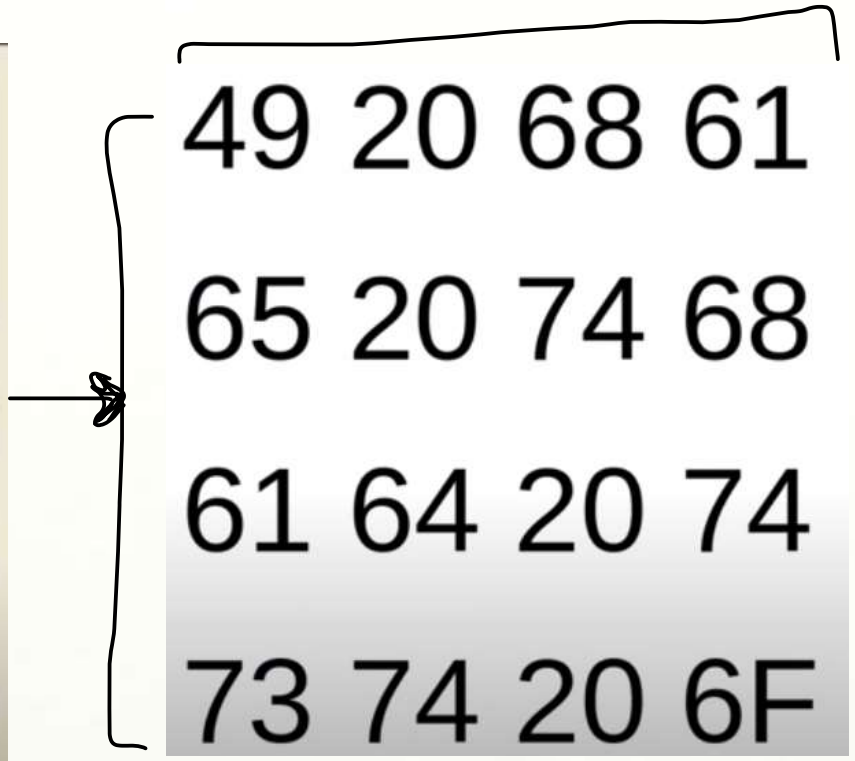
The AES Cipher - Rijndael

- designed by Rijmen-Daemen in Belgium
- has 128/192/256 bit keys, 128 bit data
- an **iterative** rather than **feistel** cipher
  - processes data as block of 4 columns of 4 bytes
  - operates on entire data block in every round
- designed to be:
  - resistant against known attacks
  - speed and code compactness on many CPUs
  - design simplicity

THE AES CIPHER - RIJNDAEL

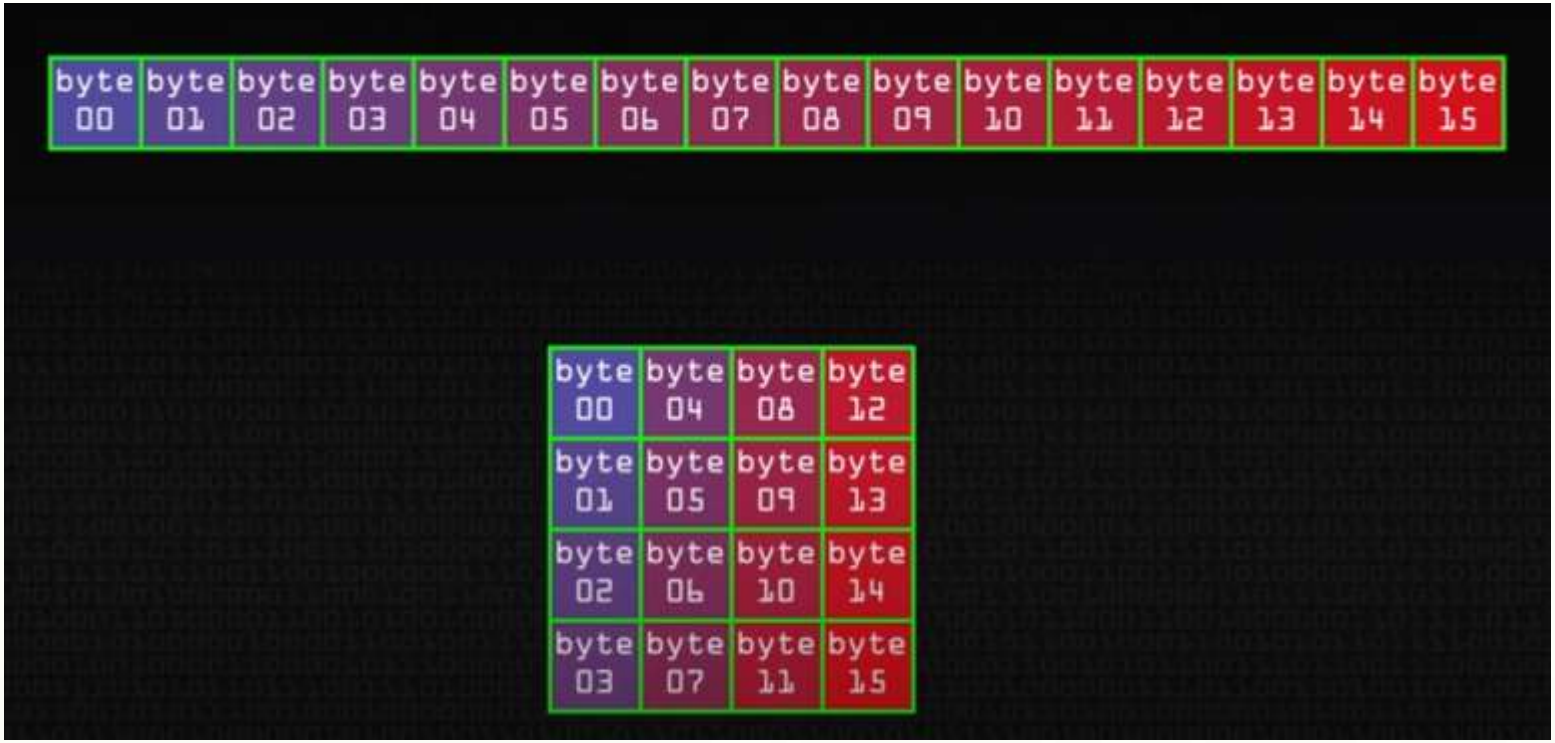
- Text is converted to Ascii HEX
- Then split to 128bit blocks, which is 4 columns and 4 rows



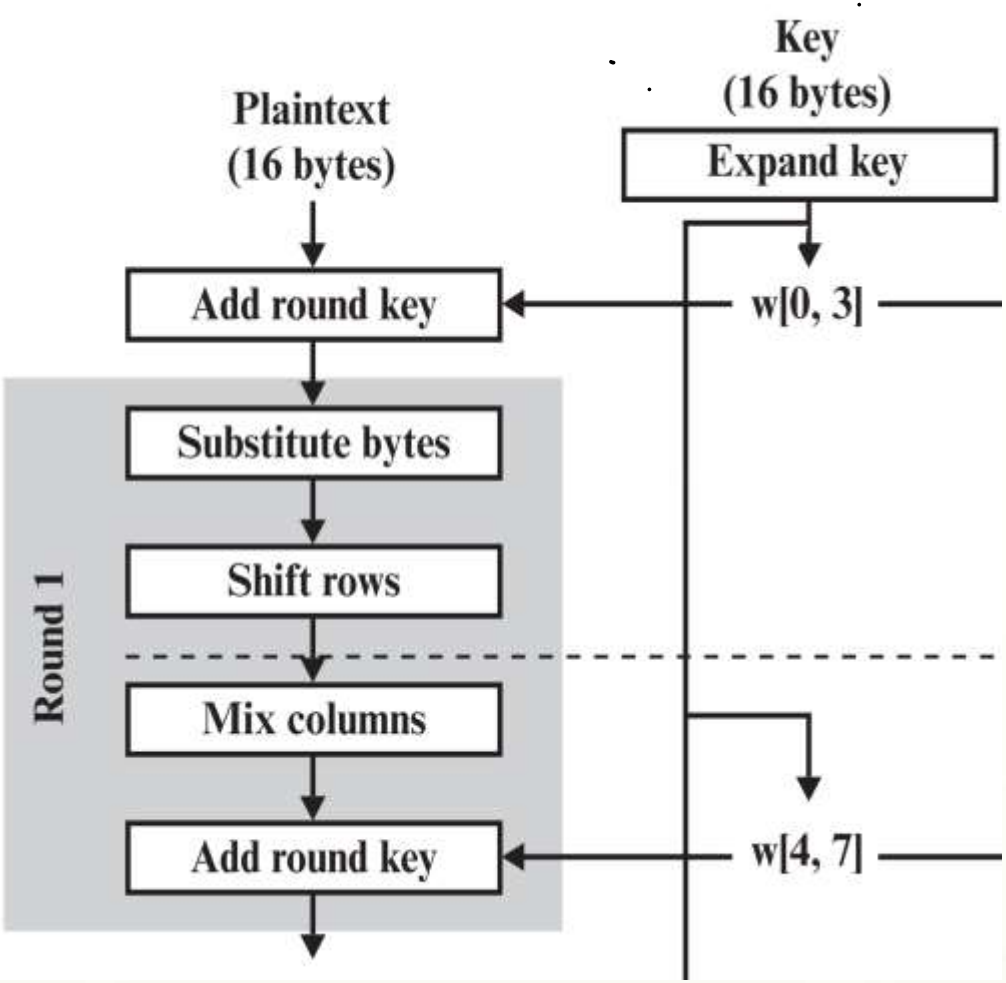


THE AES CIPHER - RIJNDAEL

- Text is converted to Ascii HEX
- Then split to 128bit blocks, which is 4 columns and 4 rows



THE AES CIPHER - RIJNDAEL



در هر دور:

گام جایگذاری بایتی: بایت‌ها جابجا شوند: از یک جدول که S-boxنامیده میشود استفاده کرده تا بایت به بایت بلوک را جابجا کند.

- **گام شیفت سطری**:سطرها شیفت داده شوند : یک جابجائی ساده که ردیف به ردیف انجام میشود.
- **گام مخلوط‌ساز ستونی(بجز دورآخر)**: یک جابجائی که هر بایت یک ستون را بصورت تابعی از تمام بایتهای همان ستون تغییر میدهد.
- **گام جمع با زیرکلید**: یک **XOR**ساده که بیتهای **بلوک فعلی** را با **بخشی از کلید** گسترش یافته XOR نماید.

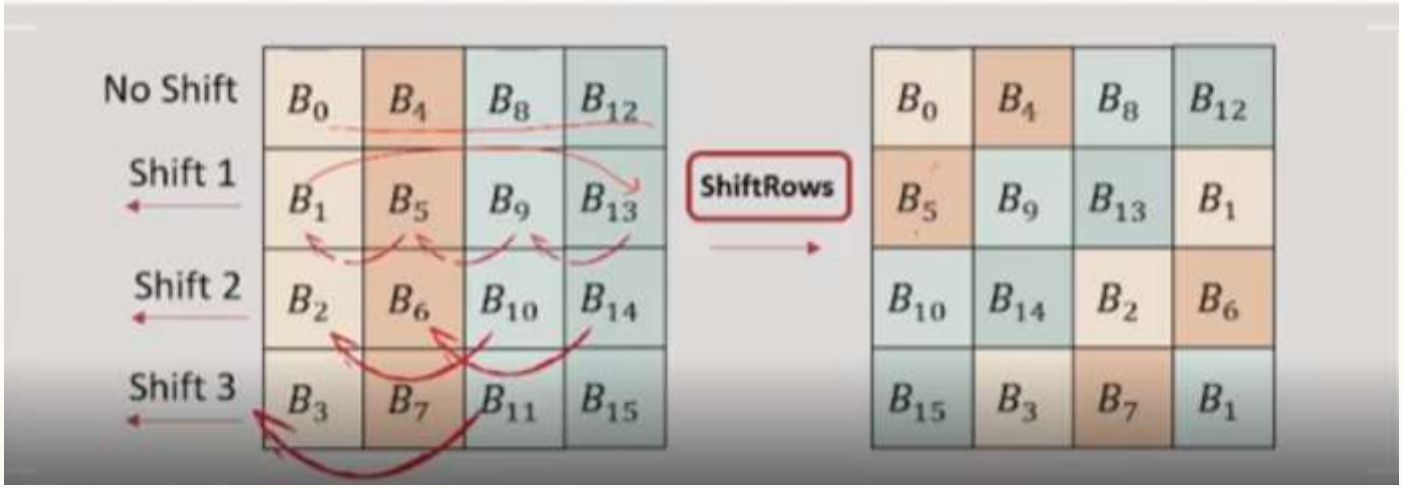
For example, for input 53(*h*ex), the output is determined by the intersection of the row with index ‘5’ and the column with index ‘3’. This results in the output havingvalue ed(*h*ex)

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure 7. S-box: substitution values for the byte xy (in hexadecimal format).

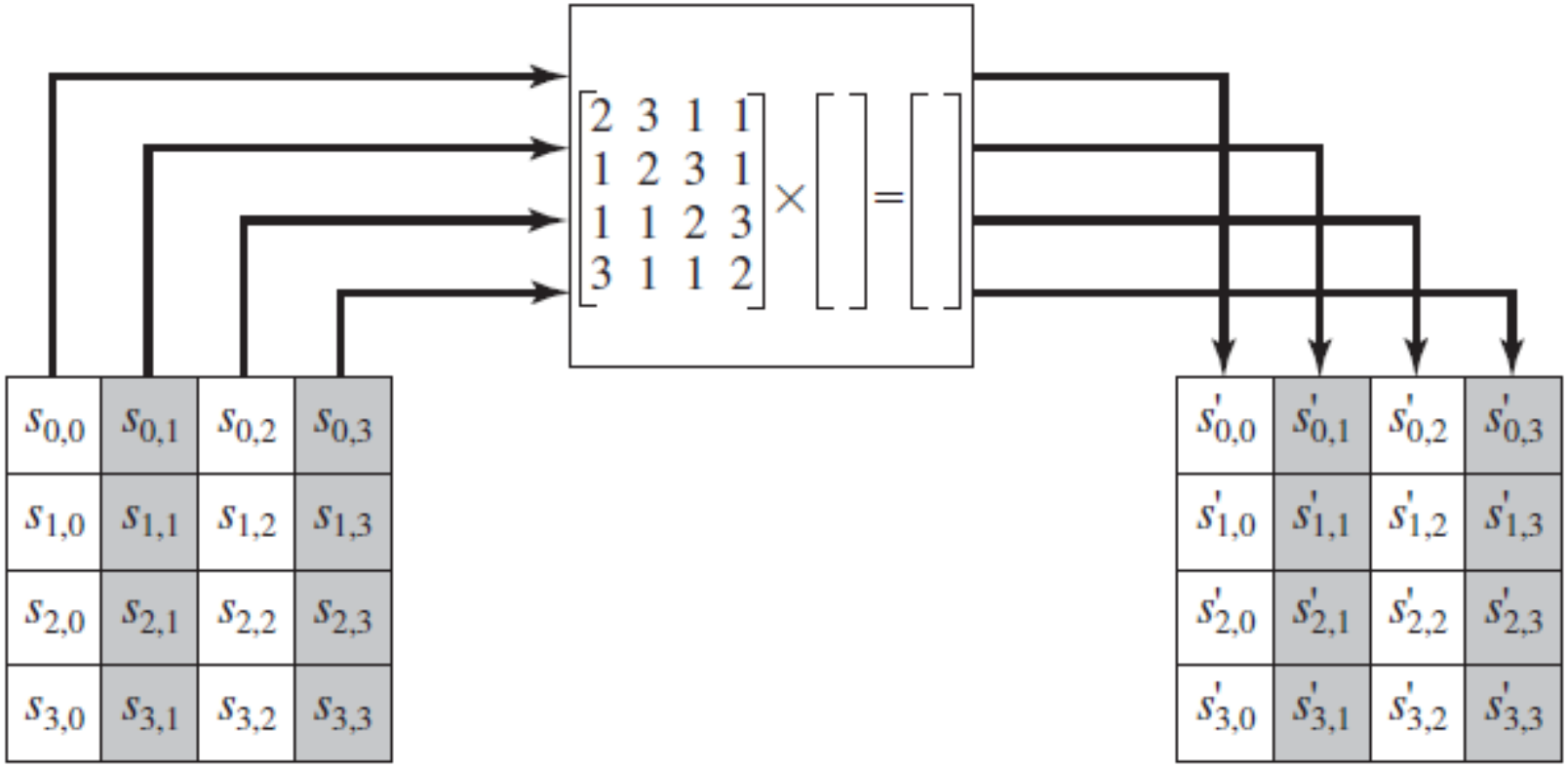
SHIFT ROWS IN AES

- سطر اول تغییر نمی کند.
- بایتهای سطر دوم به صورت چرخشی یک بایت به سمت چپ شیفت پیدا میکنند.
- بایت‌های سطر سوم به صورت چرخشی دوبایت به سمت چپ شیفیت پیدا میکنند.
- بایتهای سطر چهارم به صورت چرخشی سه بایت به سمت چپ شیفت پیدا می‌کنند.



MIX COLUMNS

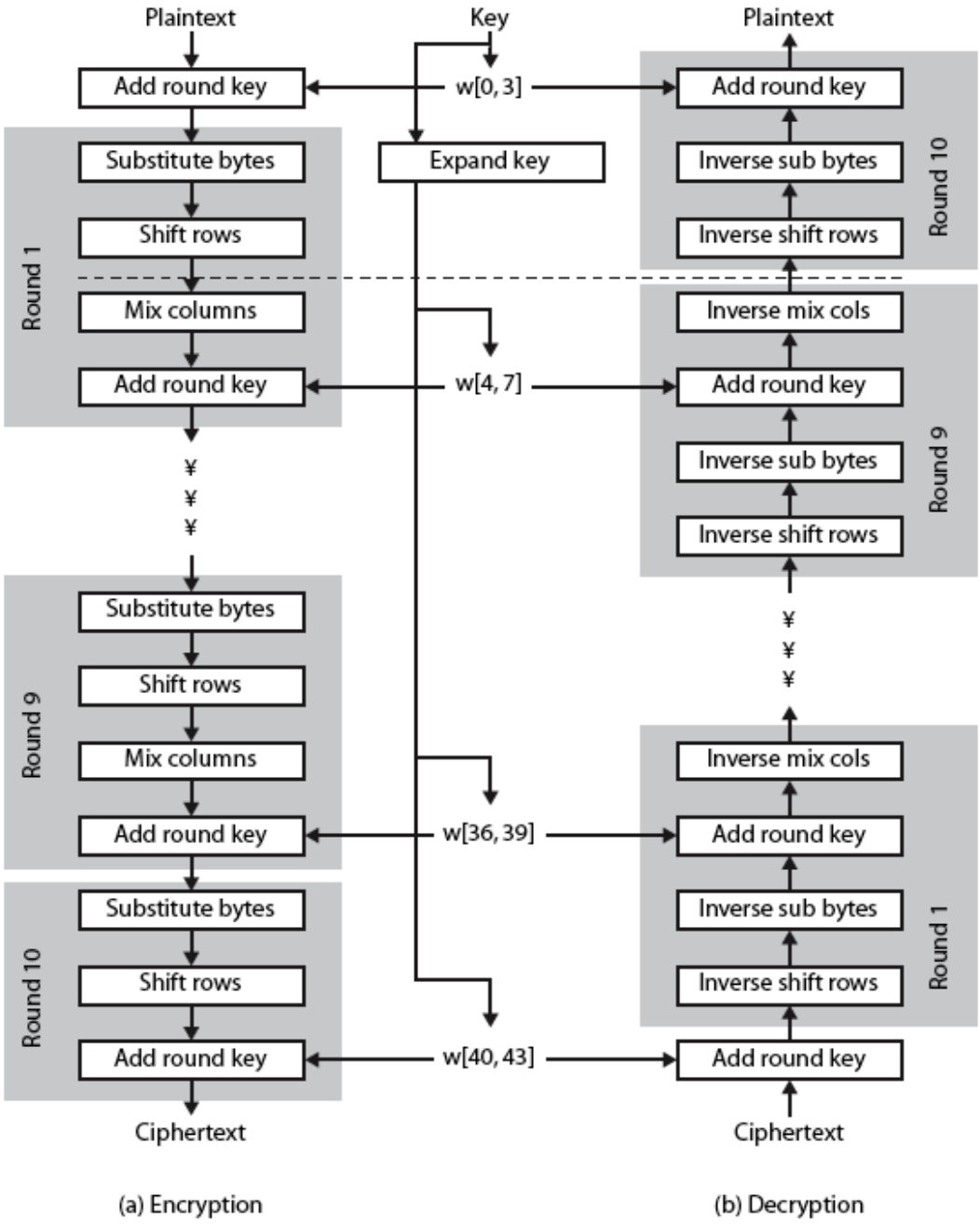
- Every Column is multiplied to a defined matrix

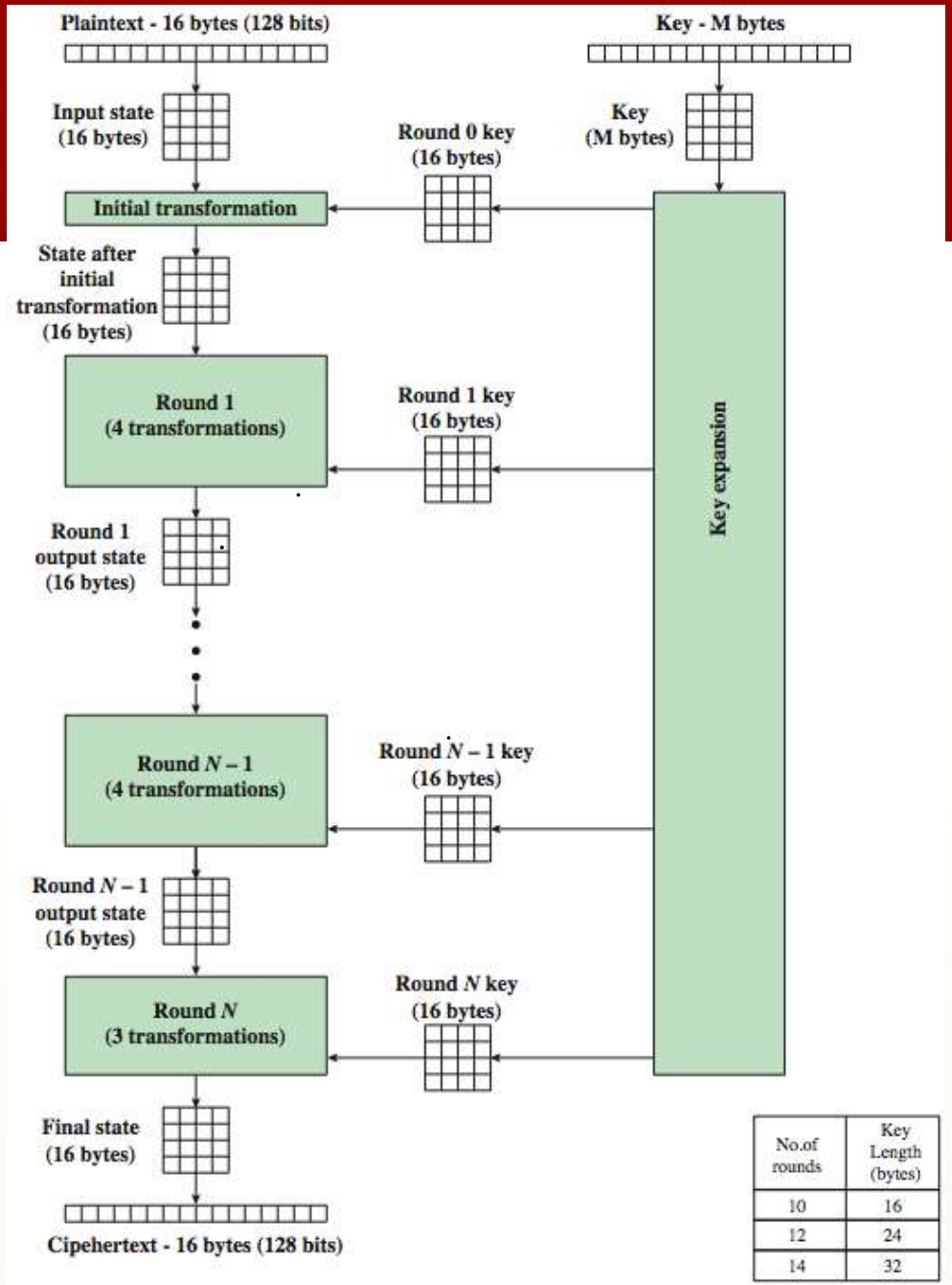


AES STRUCTURE

- data block of 4 columns of 4 bytes is state
- key is expanded to array of words
- has 9/11/13 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multiply of groups)
  - add round key (XOR state with key material)
  - view as alternating XOR key & scramble data bytes
- initial XOR key material & incomplete last round
- with fast XOR & table lookup implementation

AES STRUCTURE

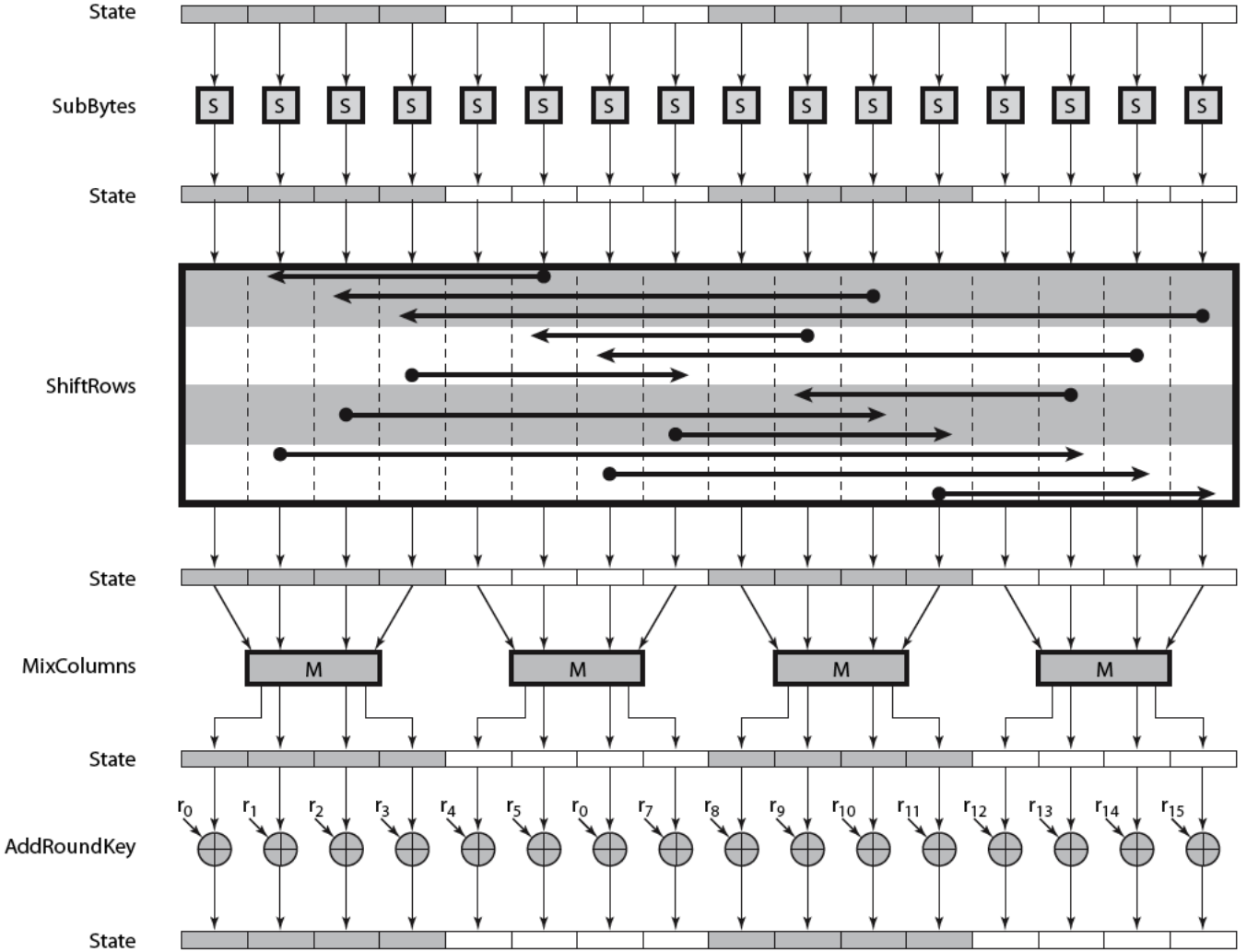




No.of rounds	Key Length (bytes)
10	16
12	24
14	32

AES ENCRYPTION PROCESS

AES ROUND


State
SubBytes
S
State
ShiftRows
State
MixColumns
M
State
AddRoundKey
r0 r1 r2 r3 r4 r5 r0 r7 r8 r9 r10 r11 r12 r13 r14 r15
State

RANDOM NUMBERS

- many uses of **random numbers** in cryptography
  - **nonces** in authentication protocols to prevent replay
  - session keys
  - **public key generation**
  - keystream for a one-time pad
- in all cases its **critical** that these values be
  - **statistically random**, **uniform distribution**, independent
  - unpredictability of future values from previous values
- true random numbers provide this
- care needed with generated random numbers

PSEUDORANDOM NUMBER GENERATORS (PRNGS)

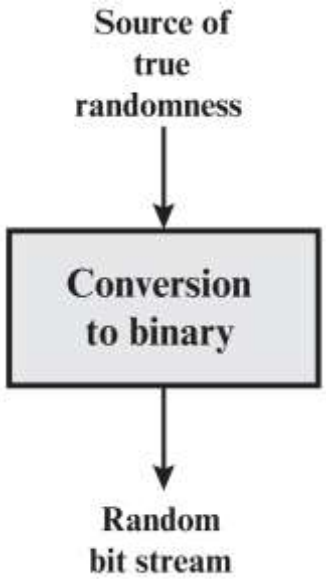
- The following criteria are used to validate that a sequence of numbers is random.
- Uniform **distribution**: The distribution of bits in the sequence should be uniform; that is, the frequency of occurrence of ones and zeros should be approximately the same.
  - Uniform distribution can be tested
- ■ **Independence**: No one subsequence in the sequence can be inferred from the others.
  - There are not test to prove independence

PSEUDORANDOM NUMBER GENERATORS (PRNGS)

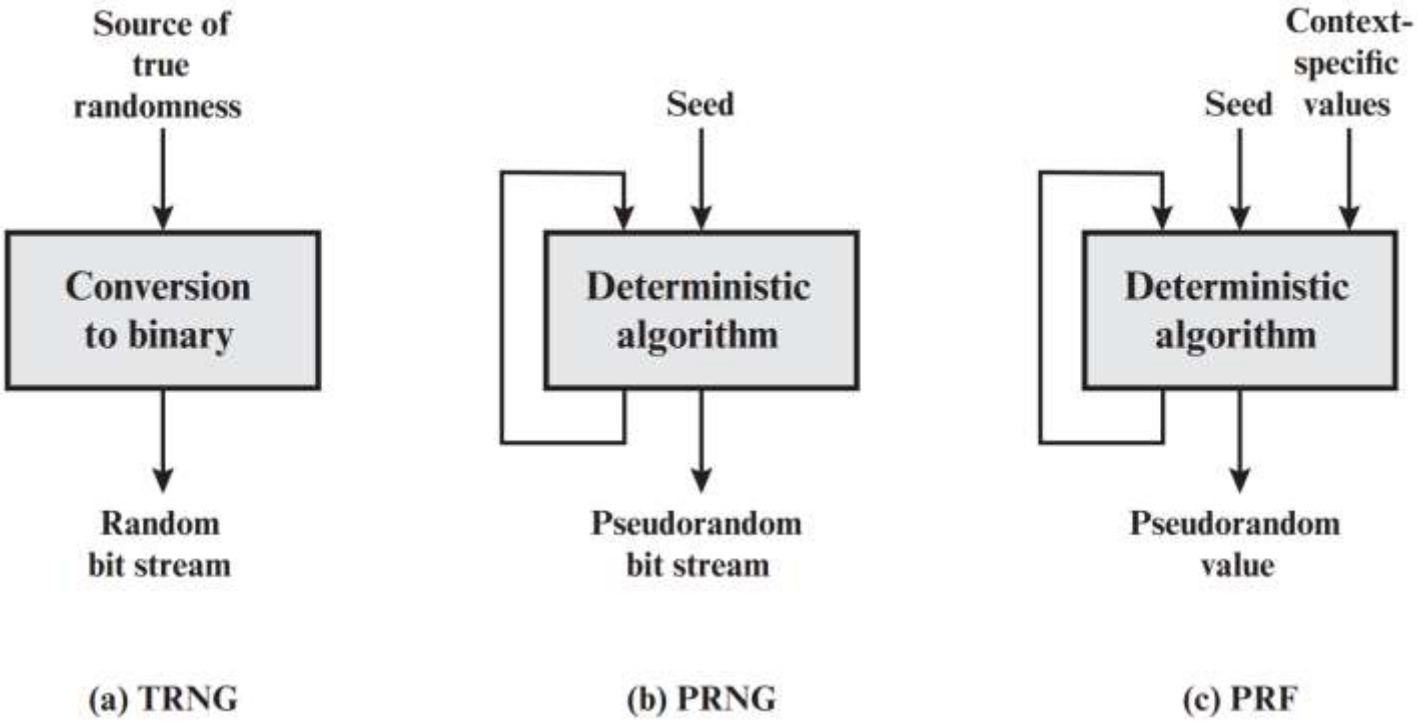
- often use deterministic algorithmic techniques to create "random numbers"
  - although are not truly random
  - can pass many tests of "randomness"
- known as "pseudorandom numbers"
- created by "Pseudorandom Number Generators (PRNGs)"

DETERMINISTIC ALGORITHMIC FOR RANDOM GENERATION

- TRNG
- Input: a source that is effectively random; (entropy source) from the physical environment of the computer
  - keystroke timing patterns, disk electrical activity, mouse movements, and instantaneous values of the system
- Output:
  - Simple or processed conversion of an Analog source to a binary output



RANDOM & PSEUDORANDOM NUMBER GENERATORS



(a) TRNG (b) PRNG (c) PRF

TRNG = true random number generator
PRNG = pseudorandom number generator
PRF = pseudorandom function

PRNG

- **PRNG** takes as input a **fixed value (seed)**,
- Output: a sequence of output bits using a deterministic algorithm.
- there is some feedback path by which some of the results of the algorithm are fed back as input as additional output bits are produced.
- the output bit stream is determined solely by the input value or values, an adversary who knows the algorithm and the seed can reproduce the entire bit stream.

PRNG Algorithm Design

- Purpose-built algorithms
  - E.g. RC4
- Algorithms based on existing cryptographic algorithms
  - Symmetric block ciphers
  - Asymmetric ciphers
  - Hash functions and message authentication codes

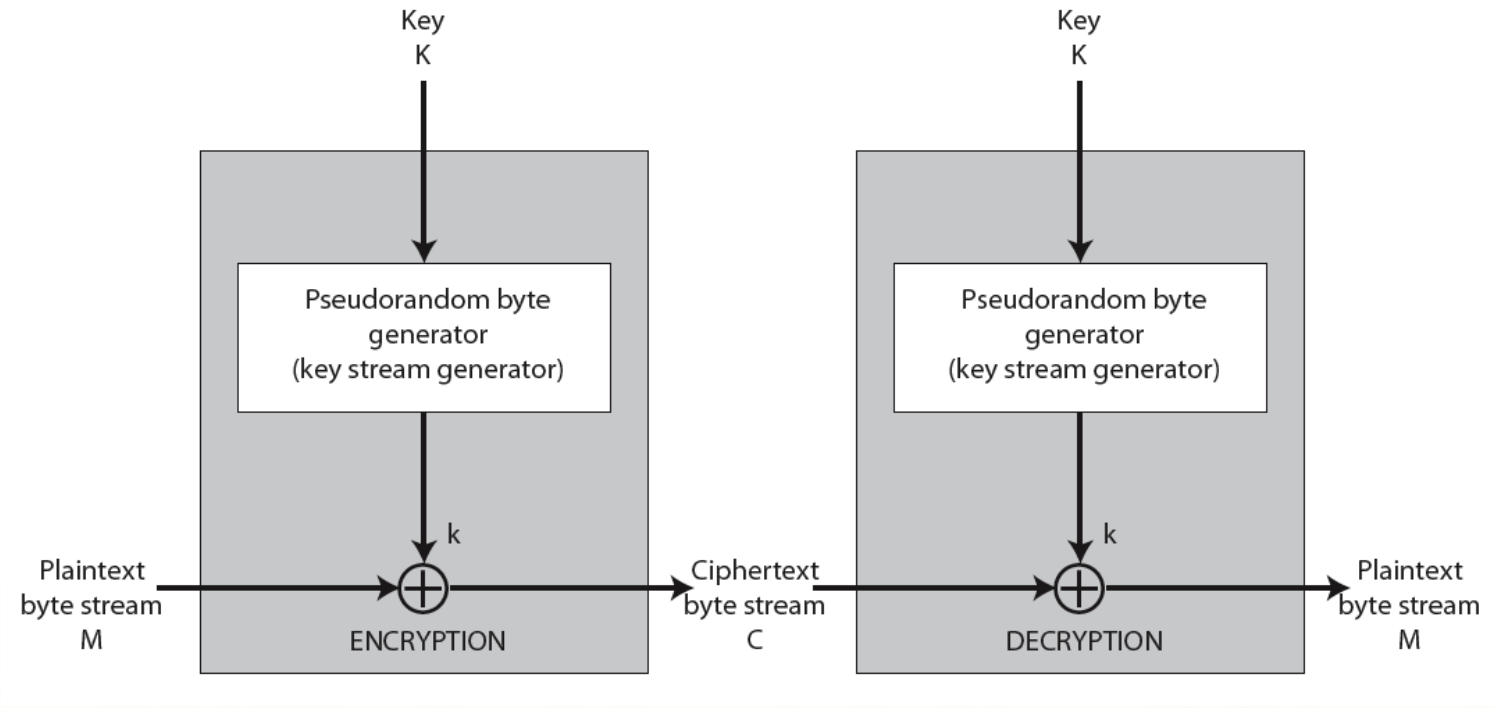
BLOCK CIPHER AND STREAM CIPHER

- A block cipher processes the input one block of elements at a time, producing an output block for each input block.
- A stream cipher processes the input elements continuously, producing output one element at a time as it goes along. Although block ciphers are far more common
- For applications that require encryption/decryption of a stream of data (such as over a data-communications channel or a browser/Web link), a stream cipher might be the better alternative.
- For applications that deal with blocks of data (such as file transfer, e-mail, and database), block ciphers may be more appropriate.

ADVANTAGE AND DISADVANTAGE

- A potential advantage of a stream cipher is that stream ciphers that do not use block ciphers as a building block are typically faster and use far less code than do block ciphers.
- One advantage of a block cipher is that you can reuse keys.
- if two plaintexts are encrypted with the same key using a stream cipher, then cryptanalysis is often quite simple [DAWS96].
  - If the two ciphertext streams are XORed together, the result is the XOR of the original plaintexts. If the plaintexts are text strings, credit card numbers, or other byte streams with known properties, then cryptanalysis may be successful

STREAM CIPHER STRUCTURE




```
  11001100 plaintext
⊕ 01101100 key stream
  10100000 ciphertext
```

STREAM CIPHER PROPERTIES

[KUMA97] Some design considerations for a stream cipher are:

- long period with no repetitions
- statistically random
- depends on large enough key, e.g. 128 bits
- large linear complexity

- Properly designed, can be as secure as a block cipher with same size key

- but usually simpler & faster

Table 2.3 Speed Comparisons of Symmetric Ciphers on a Pentium II

Cipher	Key Length	Speed (Mbps)
DES	56	9
3DES	168	3
RC2	Variable	0.9
RC4	Variable	45

RC4

- a proprietary cipher owned by by Ron Rivest for RSA Security
- variable key size, byte-oriented operation stream cipher
- widely used (web **SSL/TLS**, wireless **WEP/WPA**)
- Eight to sixteen machine operations are required per output byte
- uses that permutation to scramble input info processed a byte at a time

RC4

- RC4, a stream cipher created in 1987, was once used in SSL/TLS protocols. However, severe vulnerabilities were found in RC4 in 2003 and 2013. These vulnerabilities led to RC4 being cracked in less than a minute.
- IETF prohibited RC4 in TLS protocols in 2015.
- Several algorithms are used instead of RC4 :
  - AES-CTR: Advanced Encryption Standard (AES) in Counter mode is a widely used encryption algorithm. It's a block cipher that's used as a stream cipher.
  - Salsa20/ChaCha: These are modern stream ciphers that are considered secure and efficient. They are often used in protocols like TLS.
  - AES-GCM: AES in Galois/Counter Mode is another variant of AES that's used as a stream cipher. It's supported in TLS 1.2 and later versions.
  - AES-128-256: This refers to AES with key sizes of 128 or 256 bits. It's a common choice for many security protocols.

RC4 KEY SCHEDULE

- Starts with an array S of numbers: 0..255
- use key to well and truly shuffle
- S forms **internal state** of the cipher

/* Initialization */ S[0] = 0, S[1] = 1, c , S[255] = 255

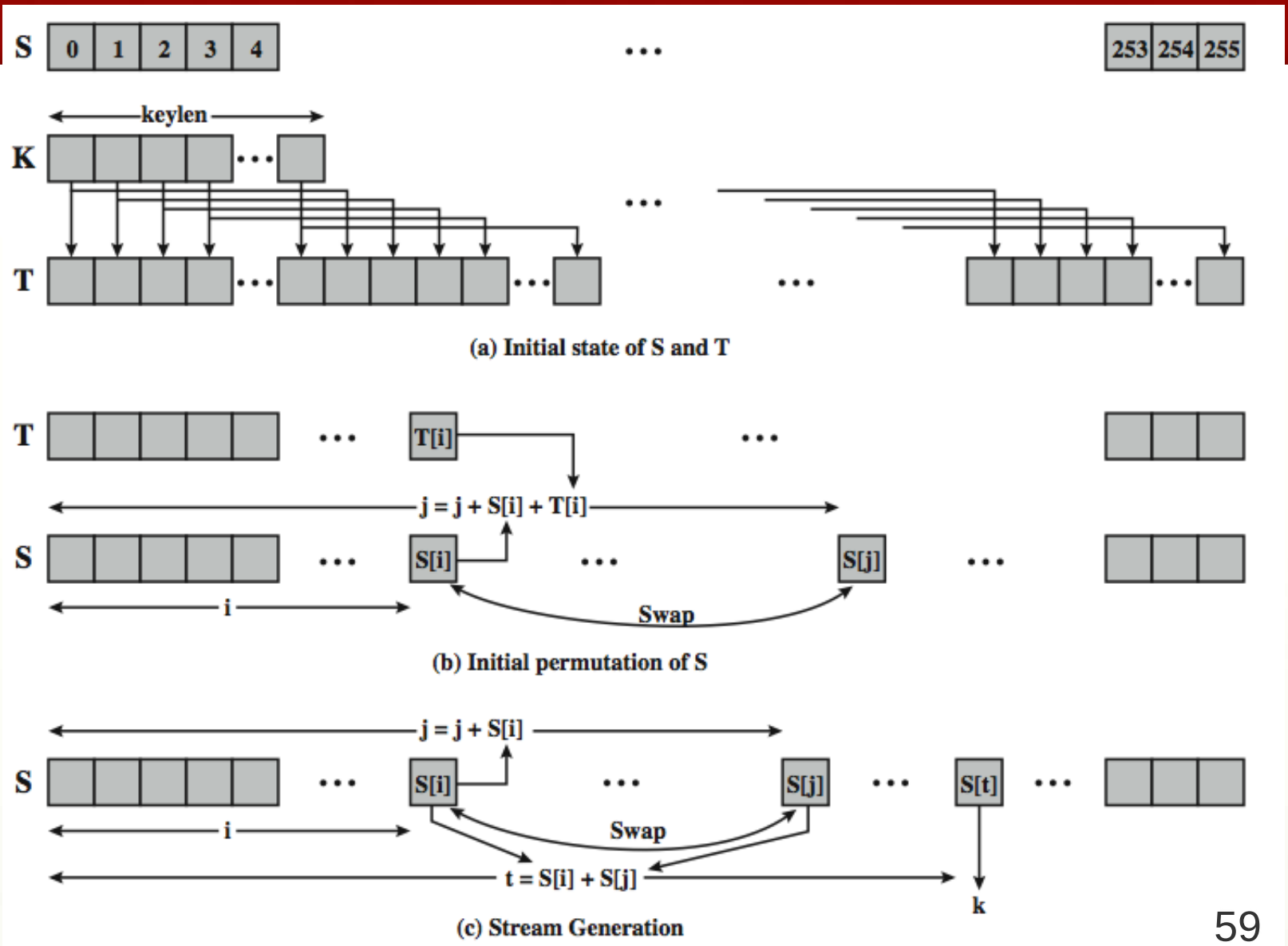
```
for i = 0 to 255 do
  S[i] = i;
  T[i] = K[i mod keylen];
  /* Initial Permutation of S */
j = 0
for i = 0 to 255 do
  j = (j + S[i] + T[i]) (mod 256);
  swap (S[i], S[j]);
```

RC4 ENCRYPTION

- encryption continues shuffling array values
- sum of shuffled pair selects "stream key" value from permutation
- XOR S[t] with next byte of message to en/decrypt

```
/* Stream Generation */
  i = j = 0;
  for each message byte Mi
    i = (i + 1) (mod 256);
    j = (j + S[i]) (mod 256);
    swap(S[i], S[j]);
    t = (S[i] + S[j]) (mod 256);
    Ci = Mi XOR S[t];
```

RC4 OVERVIEW



RC4 SECURITY

- claimed secure against known attacks
  - have some analyses, none practical
- since RC4 is a stream cipher, must **never reuse a key**
- have a concern with WEP, but due to key handling rather than RC4 itself

MODES OF OPERATION
مُودهاي عملياتي رمز هاي قالبي

- block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit blocks with 56-bit key
- NIST SP 800-38A defines **5 modes To apply a block cipher in a variety of application**
- The five modes are intended to cover virtually all of the possible applications of encryption for which **a block cipher** could be used
- These modes are intended for use with **any symmetric block cipher, including triple DES and AES.**

THE MOST IMPORTANT MODES

- Electronic Codebook Mode (ECB)
- Cipher Block Chaining Mode (CBC)
- Cipher Feedback Mode (CFB)
- Counter Mode (CTR)

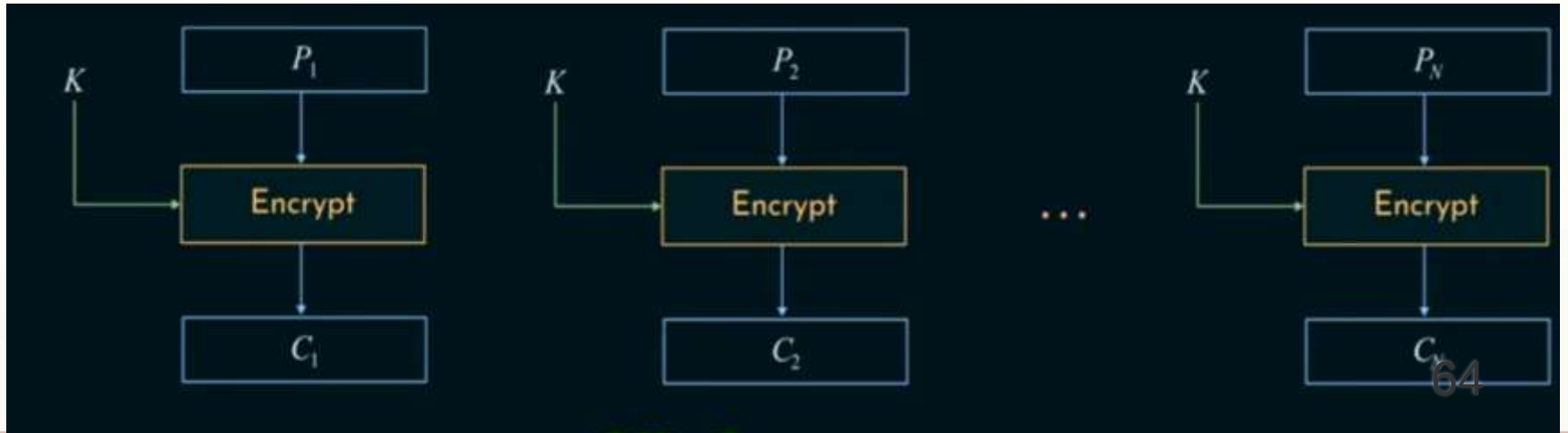
ELECTRONIC CODEBOOK BOOK (ECB)

- **message is broken into independent blocks which are encrypted**
- each block is a value which is substituted, like a codebook, hence name
- each block is encoded independently of the other blocks

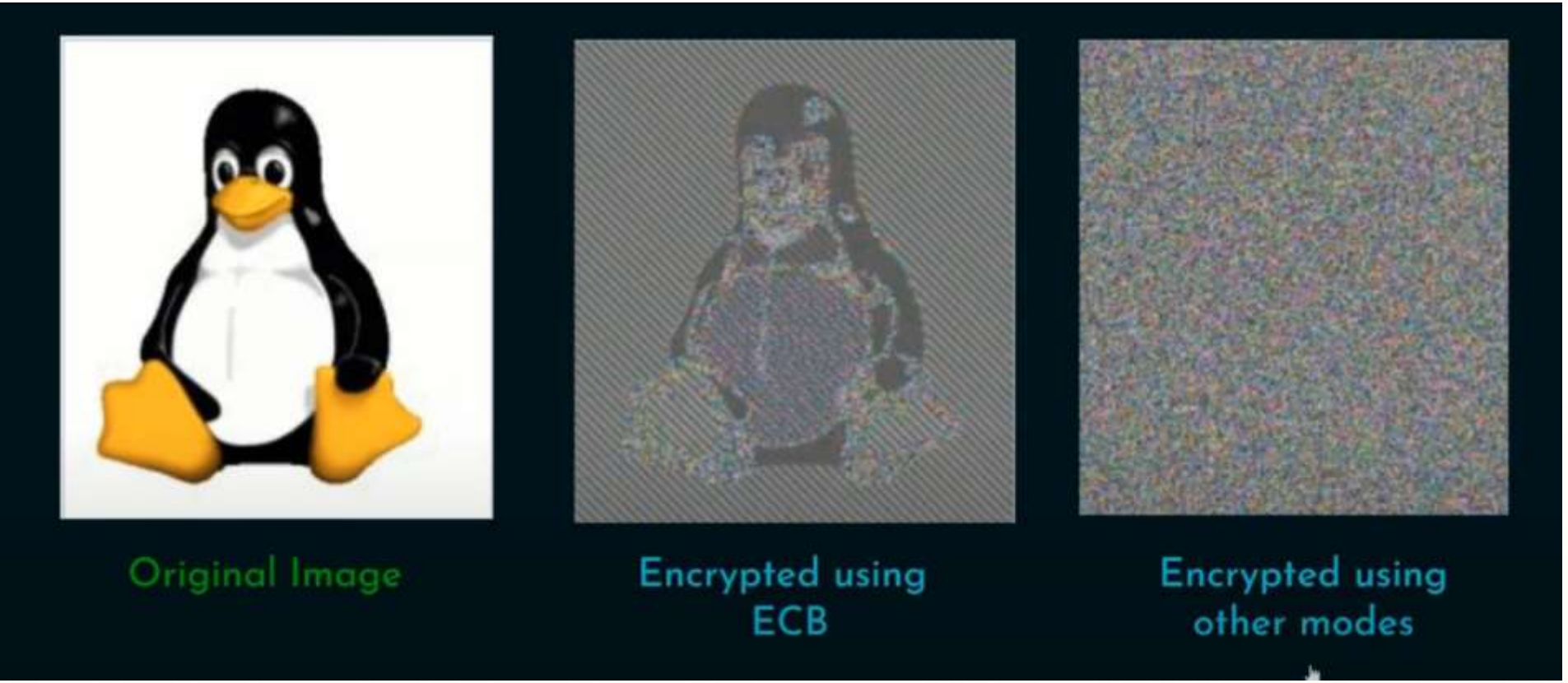
  $C_i = E_K(P_i)$
- uses: secure transmission of single values

1. ELECTRONIC CODEBOOK BOOK (ECB)

- **Pros:**
  - **Simplest mode**
  - **Ideal for short amount of data, e.x., secure transfer of AES**
  - **Independent**
  - **Fast,Parallelism due to independence of blocks**
- **Cons:**
  - **Not secure for length messages. E.X., in the picture blow, if P1 and P3 is the same C1 and C3, will be the same and attacker can understand from the pattern that you are sending a something with pattern like html. Also in Encrypting Images as the there are same pattern, the cipher could not be good enough.**



ECB VULNERABILITY



2.CIPHER BLOCK CHAINING (CBC)

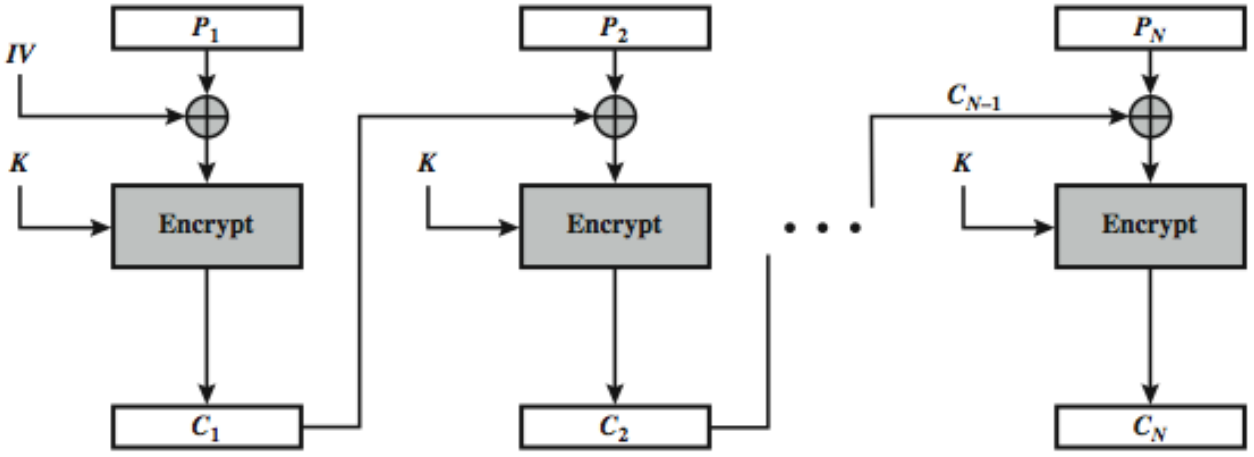
- **message is broken into blocks**
- **linked together in encryption operation**
- each previous cipher blocks is chained with current plaintext block, hence name
- use **Initial Vector (IV)** to start process

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

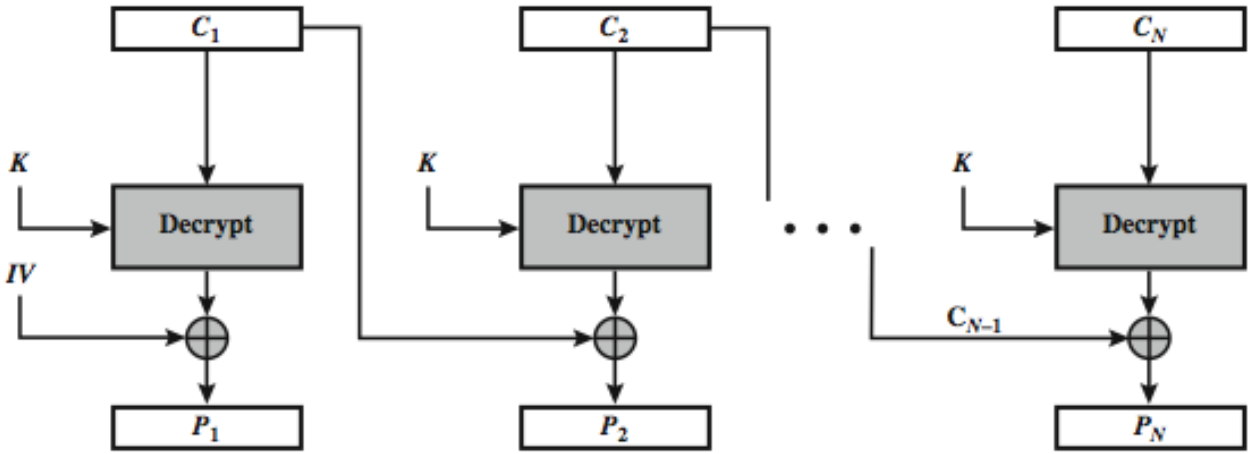
  $C_0 = IV$

- uses: bulk data encryption, authentication

CIPHER BLOCK CHAINING (CBC)



(a) Encryption



(b) Decryption

CIPHER FEEDBACK (CFB)

- **message is treated as a stream of bits (converted to s bits)**
- **result is feed back for next stage** (hence name)
- standard allows any number of bit (1,8, 64 or 128 etc) to be fed back
  - denoted CFB-1, CFB-8, CFB-64, CFB-128 etc
- most efficient to use all bits in block (64 or 128)

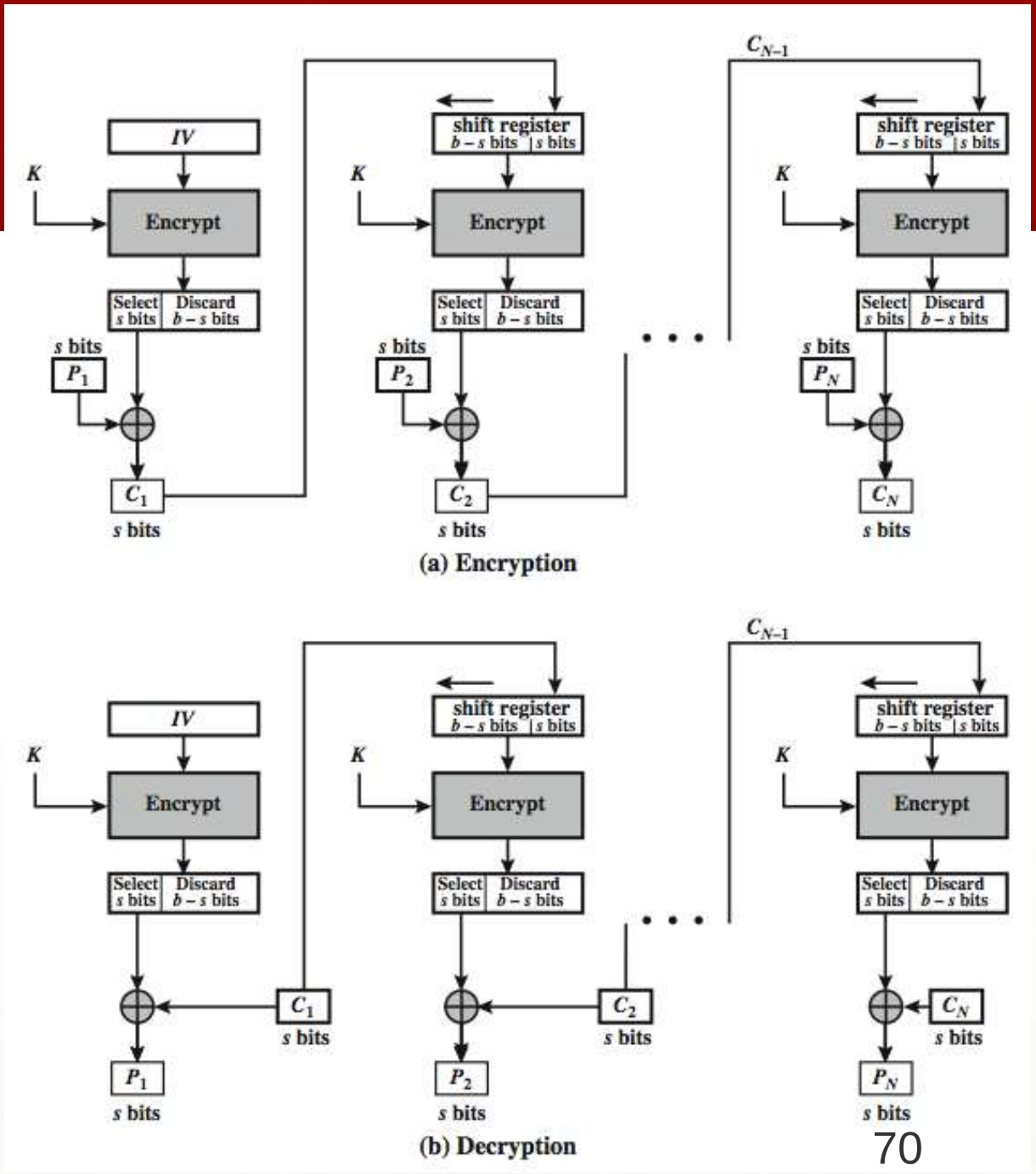
$$C_i = P_i \text{ XOR } E_K(C_{i-1})$$
$$C_0 = IV$$

- uses: stream data encryption, authentication

3.CIPHER FEEDBACK (CFB)

- In CFM: we convert a block cipher to a stream cipher because:
  - In stream cipher
    - No need of padding: i.e., In block cipher If your plain is smaller than block, you need to fill the rest of the block with zeros.
    - Real-Time application: no need to wait for the rest of the data to be received.
    - Length of the Plaintext= Length of the Cipher Text

S-BIT CIPHER FEEDBACK (CFB-S)



ADVANTAGES AND LIMITATIONS OF CFB

- Advantage:
  - Real-Time application: no need to wait for the rest of the data to be received.
  - Length of the Plaintext= Length of the Cipher Text
  - **No need of padding**
  - Encrypt Func is used both in Encipher and Decipher
  - Length of PlainText = Length of CipherText
- Cons:
  - **Chance of wastage of Transmission Capacity**: need to stall while doing block encryption after every n-bits – in noisy link, any corrupted bit will destroy values in the current and next segment.
  - Not a typical Steam cipher but a Stream mode of operation. So we are not going to generate a complete stream Cipher

CFB APPLICATION

- when we use CFB are:
- Encrypting data that is transmitted over a noisy or unreliable channel, such as a radio or a telephone line. CFB can detect and correct errors in the ciphertext by using error-correcting codes or checksums[3]
- Encrypting data that is variable in length, such as messages or files. CFB can encrypt and decrypt data of any size, without needing to pad or truncate the data to fit the block size[5]

4. COUNTER (CTR)

- a “new” mode, though proposed early on
- Applications:ATM (asynchronous transfer mode) network security and IPSec (IP security)
  - AES counter mode is used these days
- must have a different key & counter value for every plaintext block (never reused)

$$O_i = E_K(i)$$

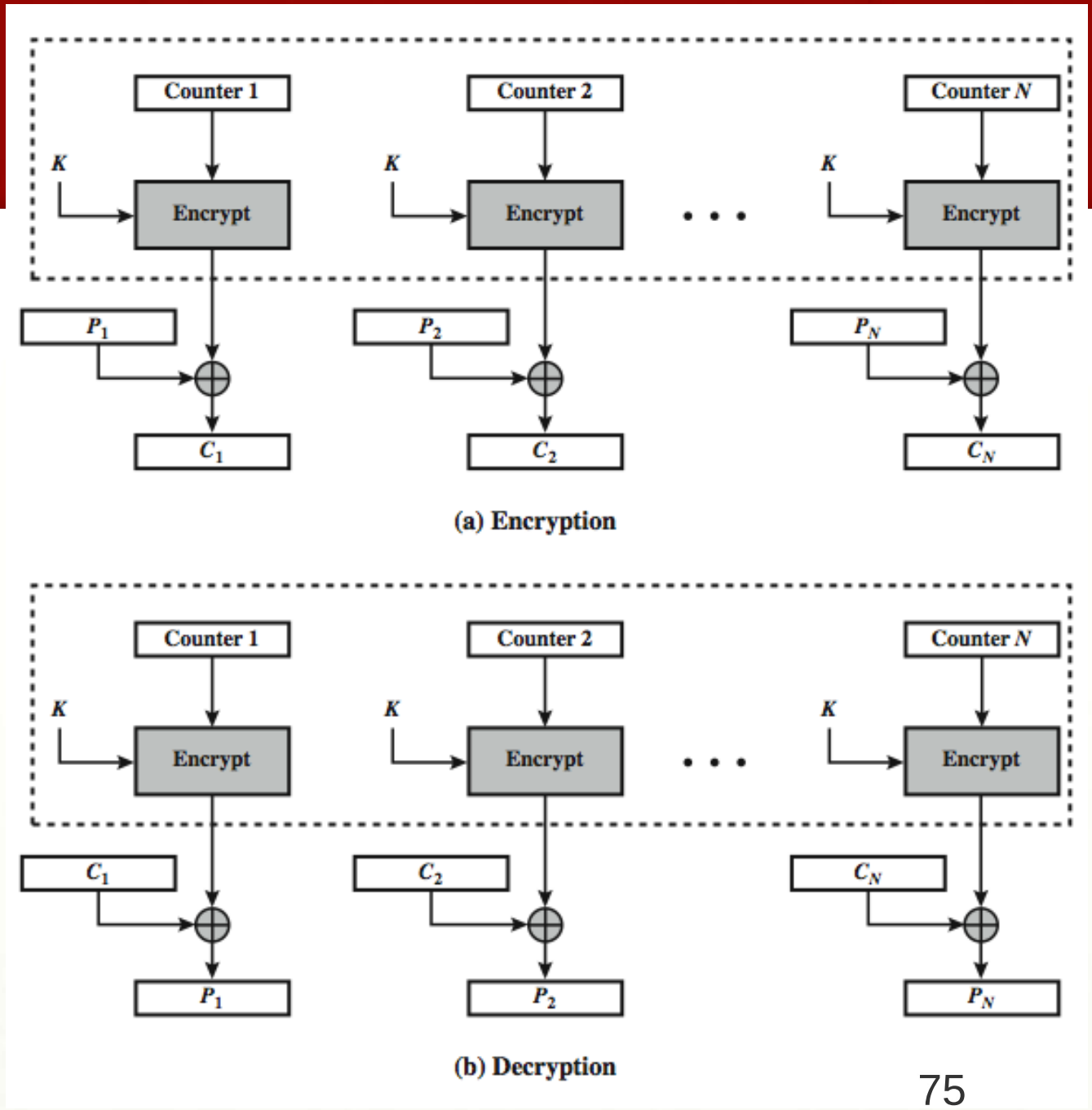
$$C_i = P_i \text{ XOR } O_i$$

- uses: high-speed network encryptions

4. COUNTER (CTR)

- Size of counter=Plaintext block size
- Different counter value for each plaintext
- Counter value is initialized
- Counter is ++ will be incremented

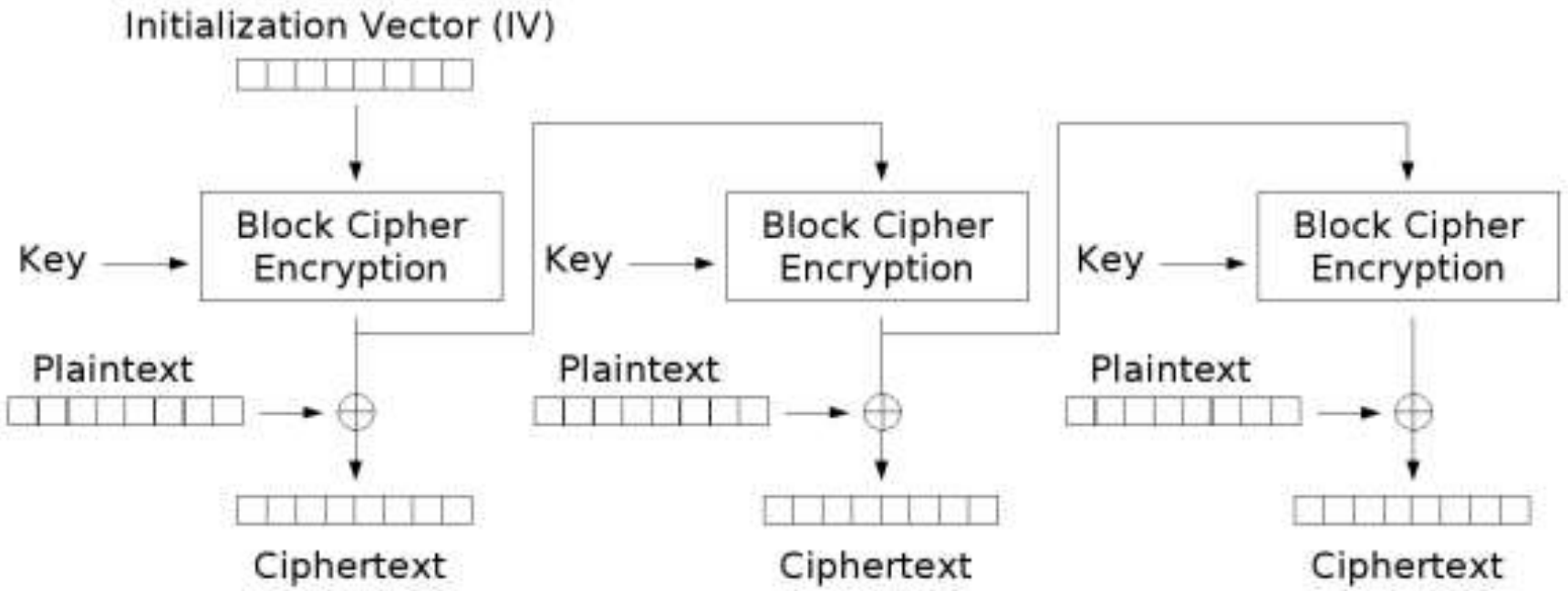
COUNTER (CTR)


Counter 1
Counter 2
Counter N
K
Encrypt
P_1
P_2
P_N
C_1
C_2
C_N
(a) Encryption
Counter 1
Counter 2
Counter N
K
Encrypt
C_1
C_2
C_N
P_1
P_2
P_N
(b) Decryption

ADVANTAGES AND LIMITATIONS OF CTR

- efficiency
  - can do parallel encryptions in h/w or s/w
  - can preprocess in advance of need
  - good for bursty high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but must ensure never reuse key/counter values, otherwise could break

OUTPUT FEEDBACK MODE (OFB)



Output Feedback (OFB) mode encryption