


Data Security: Symmetric Cryptography

Dr. Dan Massey

Read Computer Security: Principle and Practices Chapter 2

Additional Details From Chapters 20

Chap 20: Symmetric Encryption and Message Confidentiality

Responsible only for the content in these slides

Motivating Problems:

Confidentiality: how could we encrypt a message? (email, web/http, sms, etc)

Integrity: how could we authenticate a message?

(availability – not primary motivation now, provided the approach is feasible)

Cryptography

Classified along three independent dimensions:

The type of operations used for transforming plaintext to ciphertext

- Substitution – each element in the plaintext is mapped into another element
- Transposition – elements in plaintext are rearranged

The number of keys used

- Sender and receiver use same key – symmetric
- Sender and receiver each use a different key - asymmetric

The way in which the plaintext is processed

- Block cipher – processes input one block of elements at a time
- Stream cipher – processes the input elements continuously

Symmetric Encryption

- The universal technique for providing confidentiality for transmitted or stored data
- Also referred to as conventional encryption or single-key encryption
- Two requirements for secure use:
  - Need a strong encryption algorithm
  - Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure

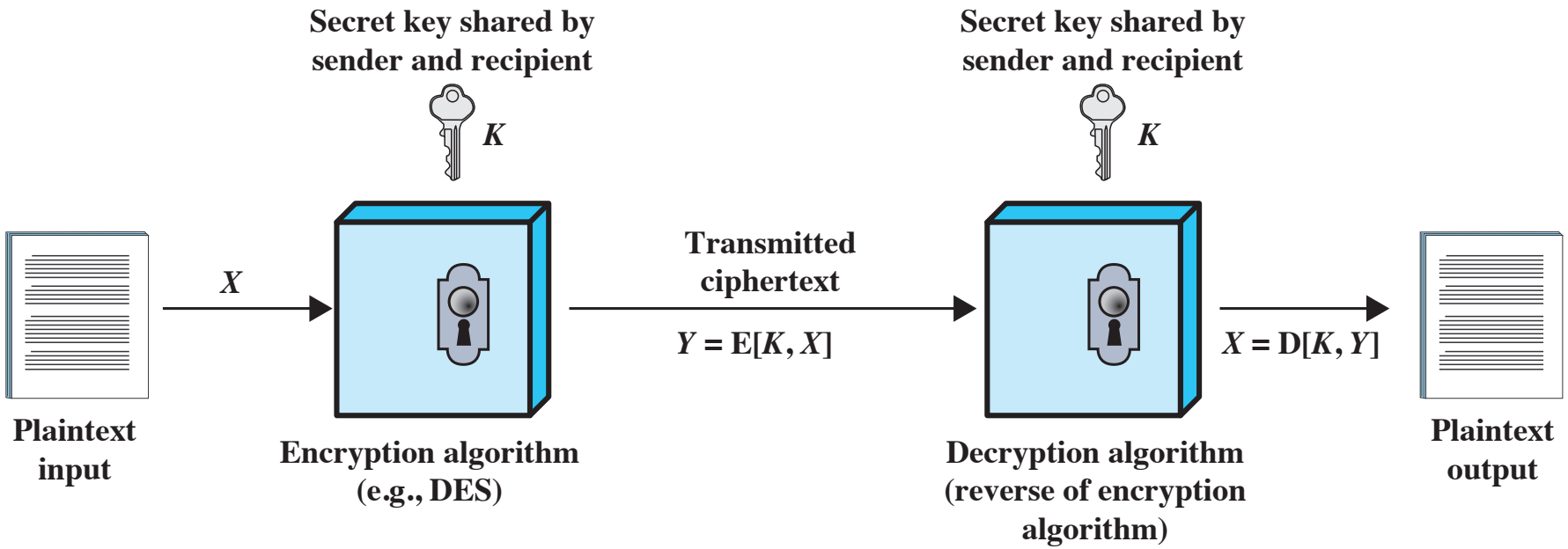


Figure 2.1 Simplified Model of Symmetric Encryption

Caeser and Vigènere Ciphers (not covered in textbook)

http://practicalcryptography.com/ciphers/caesar-cipher/

Classical Symmetric Cryptography

- Classic Substitution Cipher: Caesar cipher
- Change characters in plaintext to produce ciphertext
  - Pick a key: a letter from a-z
  - Equivalently pick a number from from 0-25
  - Encrypt by add adding key to each letter in the plaintext
- Example:
  - Key is “D” (or equivalently 3)
  - Shift each letter forward by 3, wrapping around at the end so X goes to A
  - Plaintext is `HELLO WORLD`
  - Resulting Ciphertext is `KHOOR ZRUOG`

Formal Definition of Caesar Cipher

- Quintuple ($\mathcal{E}$, $\mathcal{D}$, $\mathcal{M}$, $\mathcal{K}$, C)
  - $\mathcal{M}$ set of plaintexts
  - $\mathcal{K}$ set of keys
  - C set of ciphertexts
  - $\mathcal{E}$ set of encryption functions e: $\mathcal{M} \times \mathcal{K} \rightarrow C$
  - $\mathcal{D}$ set of decryption functions d: $C \times \mathcal{K} \rightarrow \mathcal{M}$
- Example: Cæsar cipher
  - $\mathcal{M}$ = { sequences of letters }
  - $\mathcal{K}$ = { i | i is an integer and $0 \leq i \leq 25$ }
  - $\mathcal{E}$ = { E_k | $k \in \mathcal{K}$ and for all letters m,

    $E_k(m) = (m + k) \bmod 26$ }
  - $\mathcal{D}$ = { D_k | $k \in \mathcal{K}$ and for all letters c,

    $D_k(c) = (26 + c - k) \bmod 26$ }
  - $C = \mathcal{M}$

Attacking the Caesar Cipher (1/2)

- Brute Force/Exhaustive search
  - Simply try all keys until you find the correct one
  - Decrypt using each key and consider whether it produces a valid plaintext msg
  - Caesar cipher has only 26 possible keys
- Known Plaintext attack
  - Adversary has ciphertext and corresponding plaintext
  - Recover the key (presumably to decrypt future msgs)
  - Example: know ciphertext is KHOOR ZRUOG and know plaintext was HELLO WORLD

Attacking the Caesar Cipher (2/2)

- Ciphertext only attack
  - Adversary only has ciphertext, does not possess plaintext or key
  - Recover the plaintext and the key
  - Example: only know ciphertext is `KHOOR ZRUOG`
- Cryptanalytic attacks: apply statistical techniques to learn key
  - Clearly unnecessary for Caeser cipher, but illustrative of concept
  - Compute frequency of each letter in ciphertext: (KHOOR ZRUOG)

G	0.1	H	0.1	K	0.1	O	0.3
R	0.2	U	0.1	Z	0.1		

10 characters total
3 are “O” so 3/10

  - Frequency of characters in English is on next slide

Character Frequencies

a	0.080	h	0.060	n	0.070	t	0.090
b	0.015	i	0.065	o	0.080	u	0.030
c	0.030	j	0.005	p	0.020	v	0.010
d	0.040	k	0.005	q	0.002	w	0.015
e	0.130	l	0.035	r	0.065	x	0.005
f	0.020	m	0.030	s	0.060	y	0.020
g	0.015					z	0.002

Statistical Analysis

- $f(c)$ frequency of character c in ciphertext
- $\varphi(i)$ correlation of frequency of letters in ciphertext with corresponding letters in English, assuming key is i
  - $\varphi(i) = \Sigma_{0 \le c \le 25} f(c)p(c - i)$ so here,

    $\varphi(i) = 0.1p(6 - i) + 0.1p(7 - i) + 0.1p(10 - i) + 0.3p(14 - i) + 0.2p(17 - i) + 0.1p(20 - i) + 0.1p(25 - i)$
    - $p(x)$ is frequency of character x in English
- Recall Ciphertext is `KHOOR ZRUOG`
- Frequency of each letter in ciphertext:

G(6) 0.1	H(7)0.1	K(10) 0.1	O(14) 0.3
R(17) 0.2	U(20) 0.1	Z(25) 0.1	

Ex: O is character 14. O's frequency in cipher text is 0.3. if the key were "i", then it would be character 14-i in the plain text. What is frequency of character "14-i" in English text

Correlation: $\varphi(i)$ for $0 \le i \le 25$

i	$\varphi(i)$	i	$\varphi(i)$	i	$\varphi(i)$	i	$\varphi(i)$
0	0.0482	7	0.0442	13	0.0520	19	0.0315
1	0.0364	8	0.0202	14	0.0535	20	0.0302
2	0.0410	9	0.0267	15	0.0226	21	0.0517
3	0.0575	10	0.0635	16	0.0322	22	0.0380
4	0.0252	11	0.0262	17	0.0392	23	0.0370
5	0.0190	12	0.0325	18	0.0299	24	0.0316
6	0.0660					25	0.0430

Red indicates top 4 highest correlations

The Result

- Most probable keys, based on φ:
  - $i = 6$, $\varphi(i) = 0.0660$
    - Results in plaintext of `EBIIL TLOLA`
  - $i = 10$, $\varphi(i) = 0.0635$
    - Results in plaintext `AXEEH PHKEW`
  - $i = 3$, $\varphi(i) = 0.0575$
    - Results in plaintext `HELLO WORLD`
  - $i = 14$, $\varphi(i) = 0.0535$
    - Results in plaintext `WTAAD LDGAS`
- Only English phrase is for $i = 3$
  - That's the key (3 or 'D')

Caesar's Problem

- Key is too short
  - Can be found by exhaustive search
  - Statistical frequencies not concealed well
    - They look too much like regular English letters
- So make it longer
  - Multiple letters in key
  - Idea is to smooth the statistical frequencies to make cryptanalysis harder

Vigènere Cipher

- Like Caesar cipher, but use a phrase
- Example
  - Message `THE BOY HAS THE BALL`
  - Key `VIG`
  - Encipher using Caesar cipher for each letter:

```
key     VIGVIGVIGVIGVIGV
plain   THEBOYHASTHEBALL
cipher  OPKWWECIYOPKWIRG
```

Data Encryption Standard (DES)

Until recently was the most widely used encryption scheme

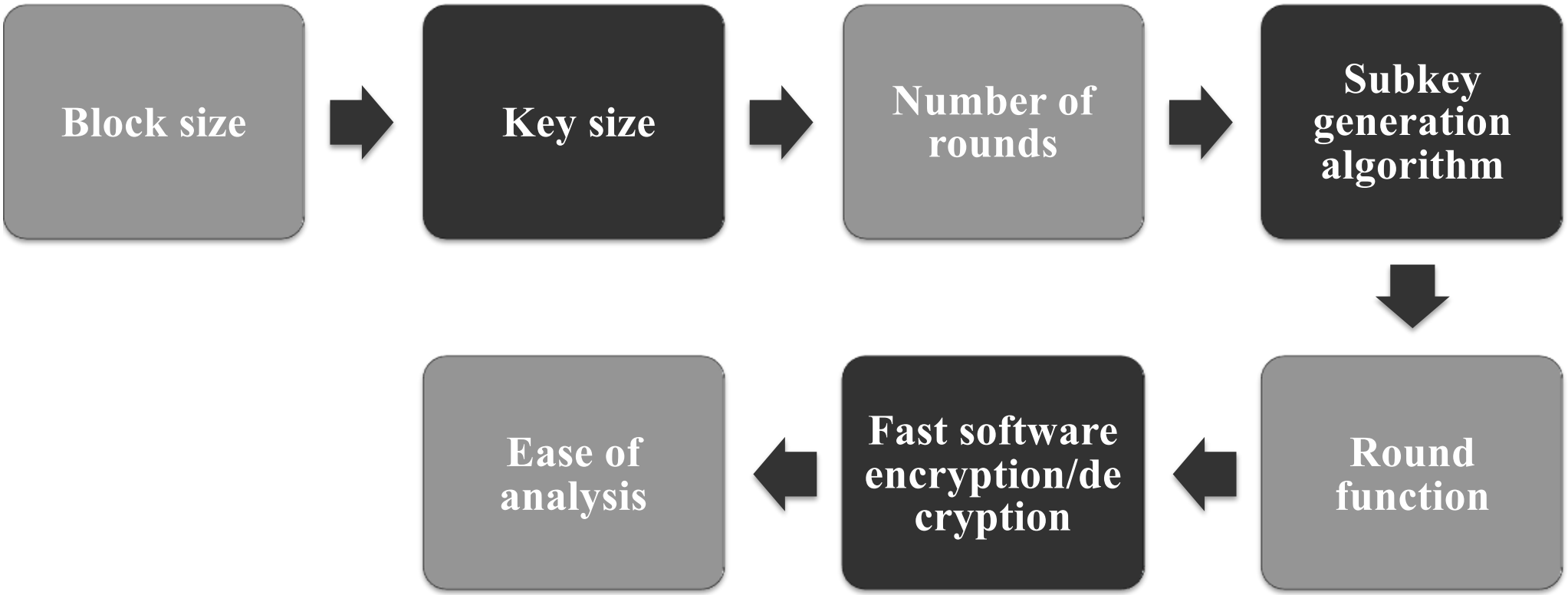
- FIPS PUB 46
- Referred to as the Data Encryption Algorithm (DEA)
- Uses 64 bit plaintext block and 56 bit key to produce a 64 bit ciphertext block

Strength concerns:

- Concerns about the algorithm itself
  - DES is the most studied encryption algorithm in existence
- Concerns about the use of a 56-bit key
  - The speed of commercial off-the-shelf processors makes this key length woefully inadequate

Block Cipher Structure

- Symmetric block cipher consists of:
  - A sequence of rounds
  - With substitutions and permutations controlled by key
- Parameters and design features:



Key Concept Behind DES

16 rounds for DES

Decryption uses the same structure with keys in reverse order

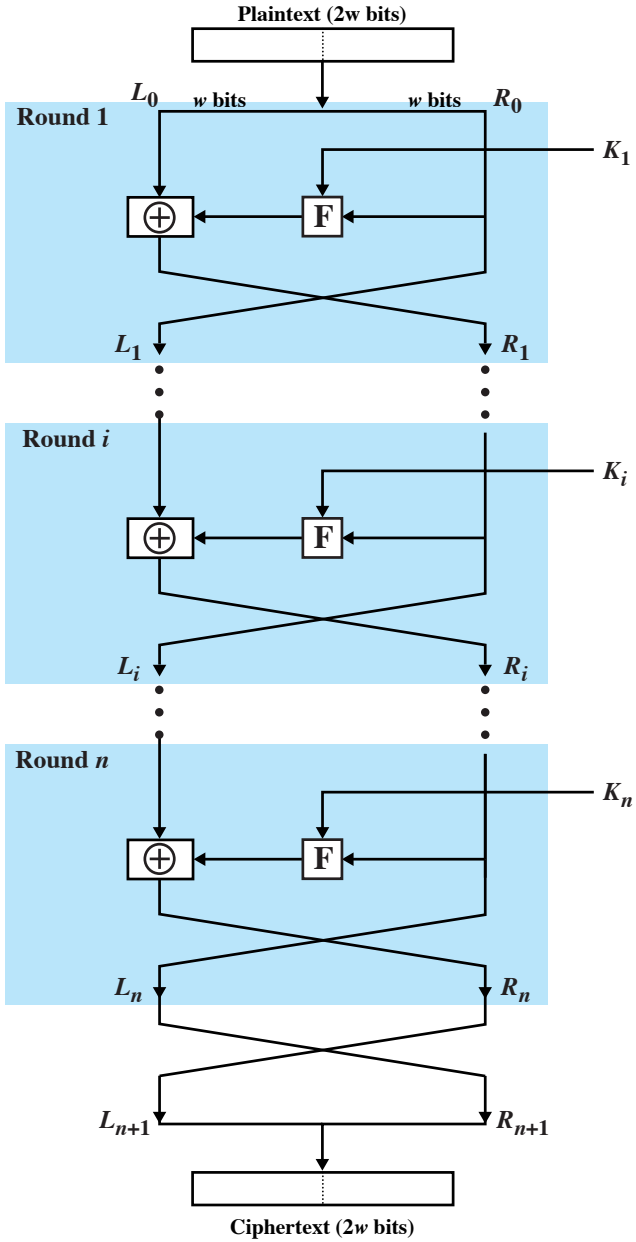


Figure 20.1 Classical Feistel Network

Triple DES (3DES)

- Repeats basic DES algorithm three times using either two or three unique keys
- First standardized for use in financial applications in ANSI standard X9.17 in 1985
- Attractions:
  - 168-bit key length overcomes the vulnerability to brute-force attack of DES
  - Underlying encryption algorithm is the same as in DES
- Drawbacks:
  - Algorithm is sluggish in software
  - Uses a 64-bit block size

Step 2 Decrypt provides backwards compatibility with DES

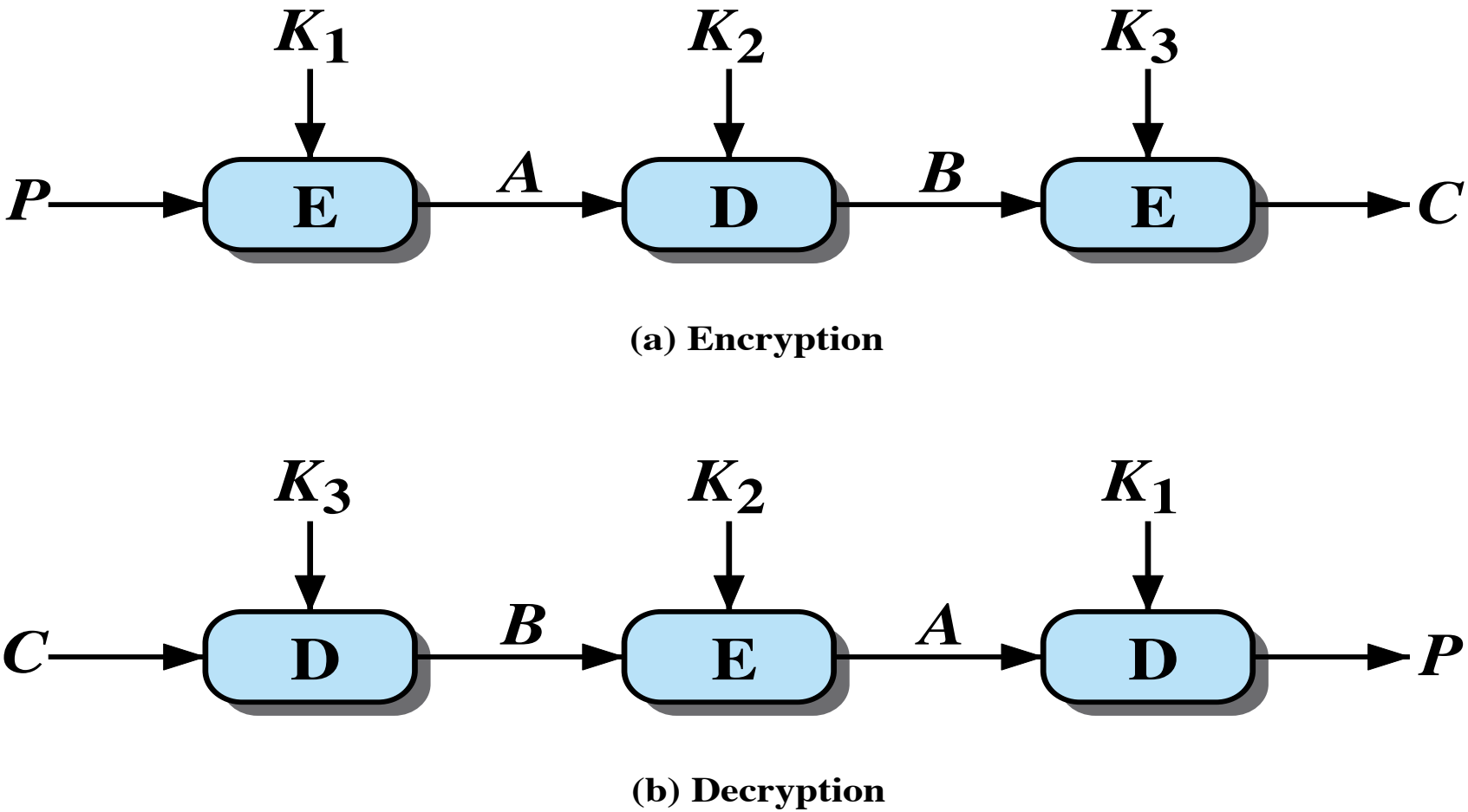


Figure 20.2 Triple DES

K1=K2=K3 is simply DES; K1 = K3 implies 112 bit key (2*56)

Advanced Encryption Standard (AES)

Needed a replacement for 3DES

3DES was not reasonable for long term use

NIST called for proposals for a new AES in 1997

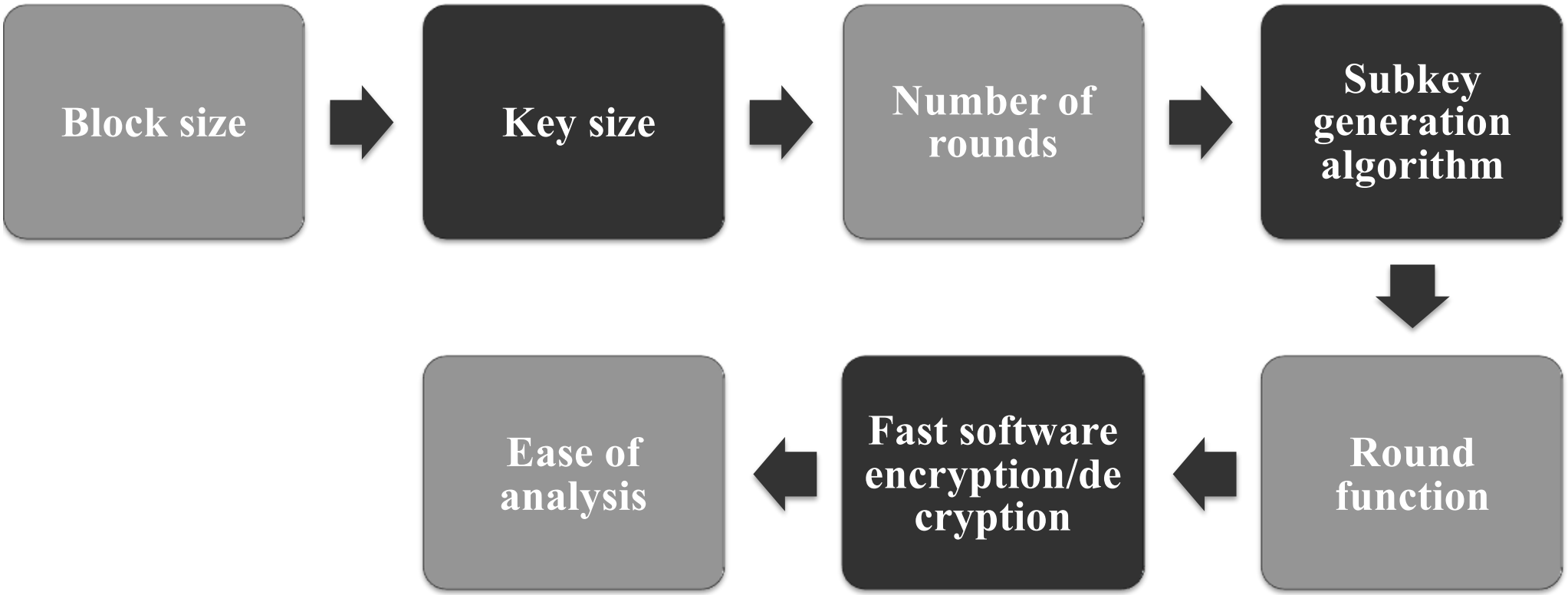
- Should have a security strength equal to or better than 3DES
- Significantly improved efficiency
- Symmetric block cipher
- 128 bit data and 128/192/256 bit keys

Selected Rijndael in November 2001

Published as FIPS 197

Recall: Block Cipher Structure

- Symmetric block cipher consists of:
  - A sequence of rounds
  - With substitutions and permutations controlled by key
- Parameters and design features:



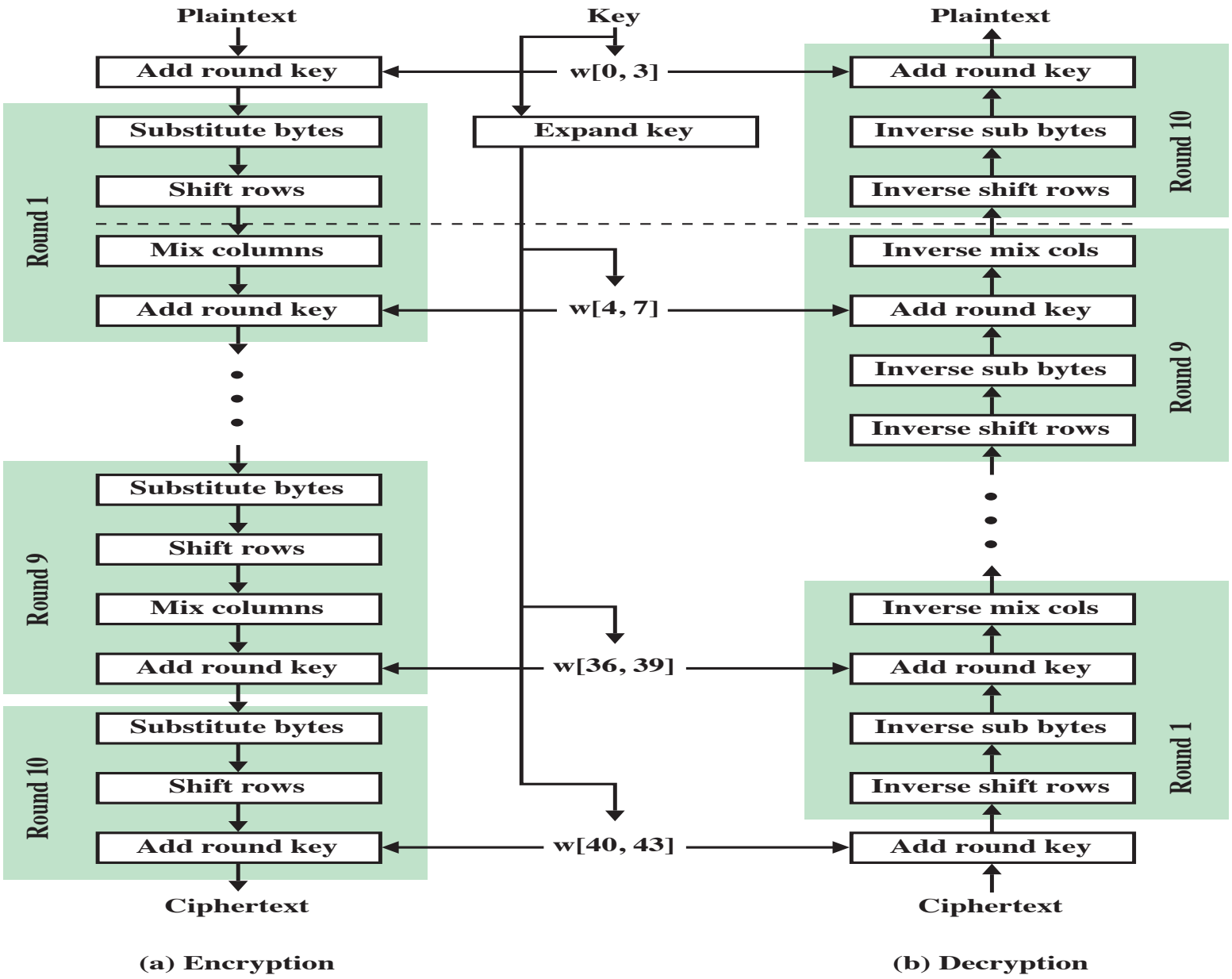


Figure 20.3 AES Encryption and Decryption

Comparison of Three Popular Symmetric Encryption Algorithms

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard
AES = Advanced Encryption Standard

Attacking Symmetric Encryption

Brute-Force Attacks

- Try all possible keys on some ciphertext until an intelligible translation into plaintext is obtained
- On average half of all possible keys must be tried to achieve success

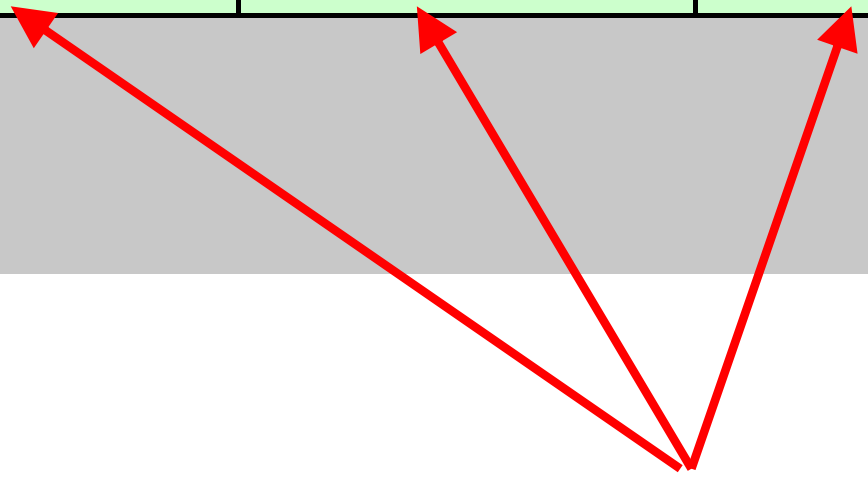
Cryptanalytic Attacks

- Exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or the key being used
  - If successful all future and past messages encrypted with that key are compromised
- Rely on:
  - Nature of the algorithm
  - Some knowledge of the general characteristics of the plaintext
  - Some sample plaintext-ciphertext pairs encrypted with that key are compromised

Comparison of Three Popular Symmetric Encryption Algorithms

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard
AES = Advanced Encryption Standard



Brute force focus on the key size

Brute Force Attacks

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

Computationally Secure Encryption Schemes

- Can estimate time/cost of a brute-force attack
- Encryption is computationally secure if:
  - Cost of breaking cipher exceeds value of information
  - Time required to break cipher exceeds the useful lifetime of the information
- Usually very difficult to estimate the amount of effort required to break with cryptanalytics

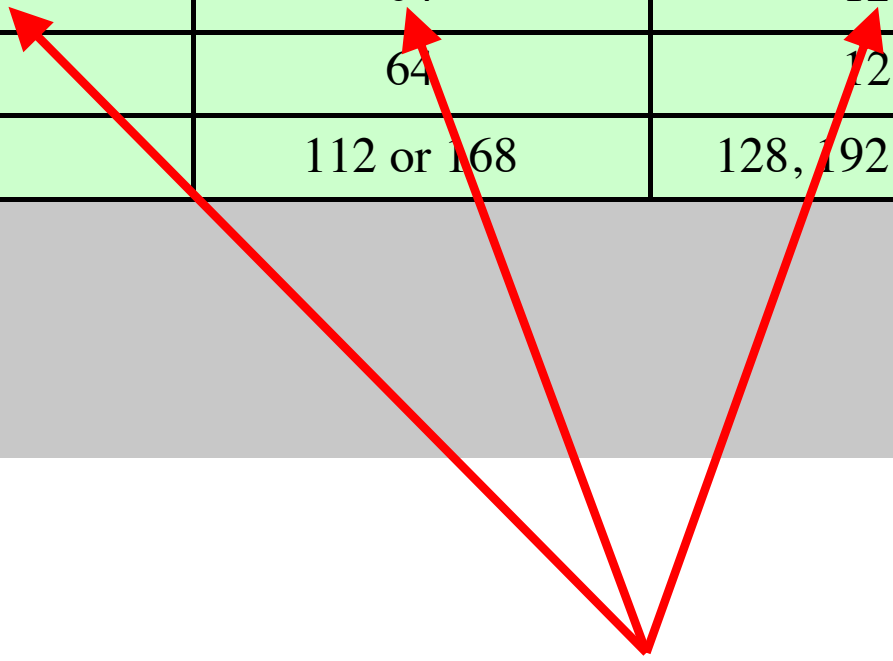
Table 20.1 Types of Attacks on Encrypted Messages

Type of Attack	Known to Cryptanalyst
Ciphertext only	•Encryption algorithm •Ciphertext to be decoded
Known plaintext	•Encryption algorithm •Ciphertext to be decoded •One or more plaintext-ciphertext pairs formed with the secret key
Chosen plaintext	•Encryption algorithm •Ciphertext to be decoded •Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen ciphertext	•Encryption algorithm •Ciphertext to be decoded •Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen text	•Encryption algorithm •Ciphertext to be decoded •Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key •Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

Comparison of Three Popular Symmetric Encryption Algorithms

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard
AES = Advanced Encryption Standard



Another concern is the block size
What if the message is larger than 64 or 128 bits?

Practical Security Issues

- Typically symmetric encryption is applied to a unit of data larger than a single 64-bit or 128-bit block
- Basic idea is to break the plaintext into multiple blocks
- Electronic codebook (ECB) mode is the simplest approach to multiple-block encryption
  - Each block of plaintext is encrypted using the same key
  - Cryptanalysts may be able to exploit regularities in the plaintext
- Several Alternate Modes of Operation
  - Alternative techniques developed to increase the security of symmetric block encryption for large sequences
  - Overcomes the weaknesses of ECB

Block Cipher Modes of Operation

Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	•General-purpose block-oriented transmission •Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission •Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission •Useful for high-speed requirements

Electronic Codebook (ECB)

- Simplest mode
- Plaintext is handled *b* bits at a time and each block is encrypted using the same key
- "Codebook" is used because there is an unique ciphertext for every *b*-bit block of plaintext
  - Not secure for long messages since repeated plaintext is seen in repeated ciphertext
- To overcome security deficiencies you need a technique where the same plaintext block, if repeated, produces different ciphertext blocks

Block Cipher Modes of Operation

Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	•General-purpose block-oriented transmission •Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission •Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission •Useful for high-speed requirements

The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.

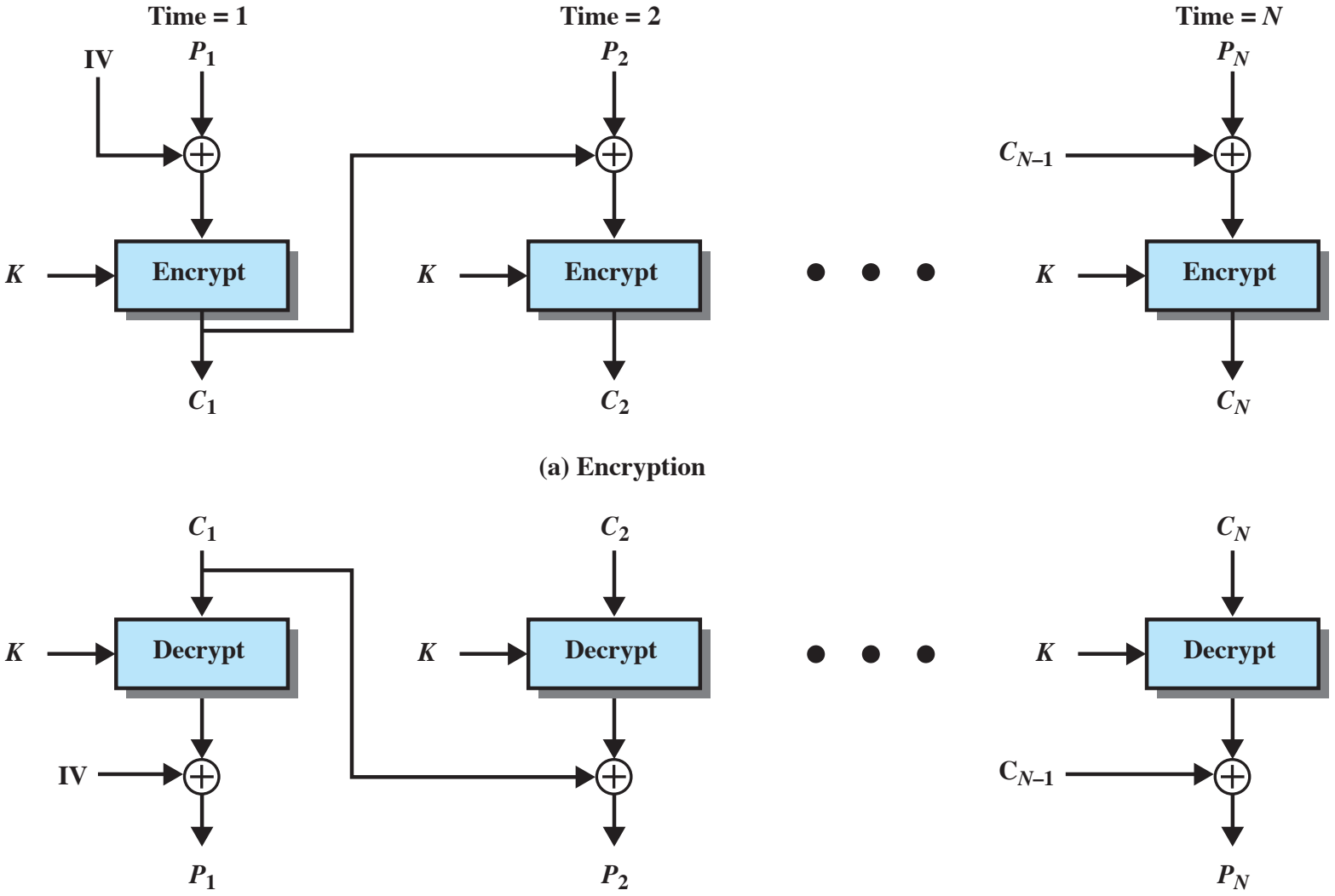


Figure 20.7 Cipher Block Chaining (CBC) Mode

Block Cipher Modes of Operation

Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	•General-purpose block-oriented transmission •Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission •Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission •Useful for high-speed requirements

Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.

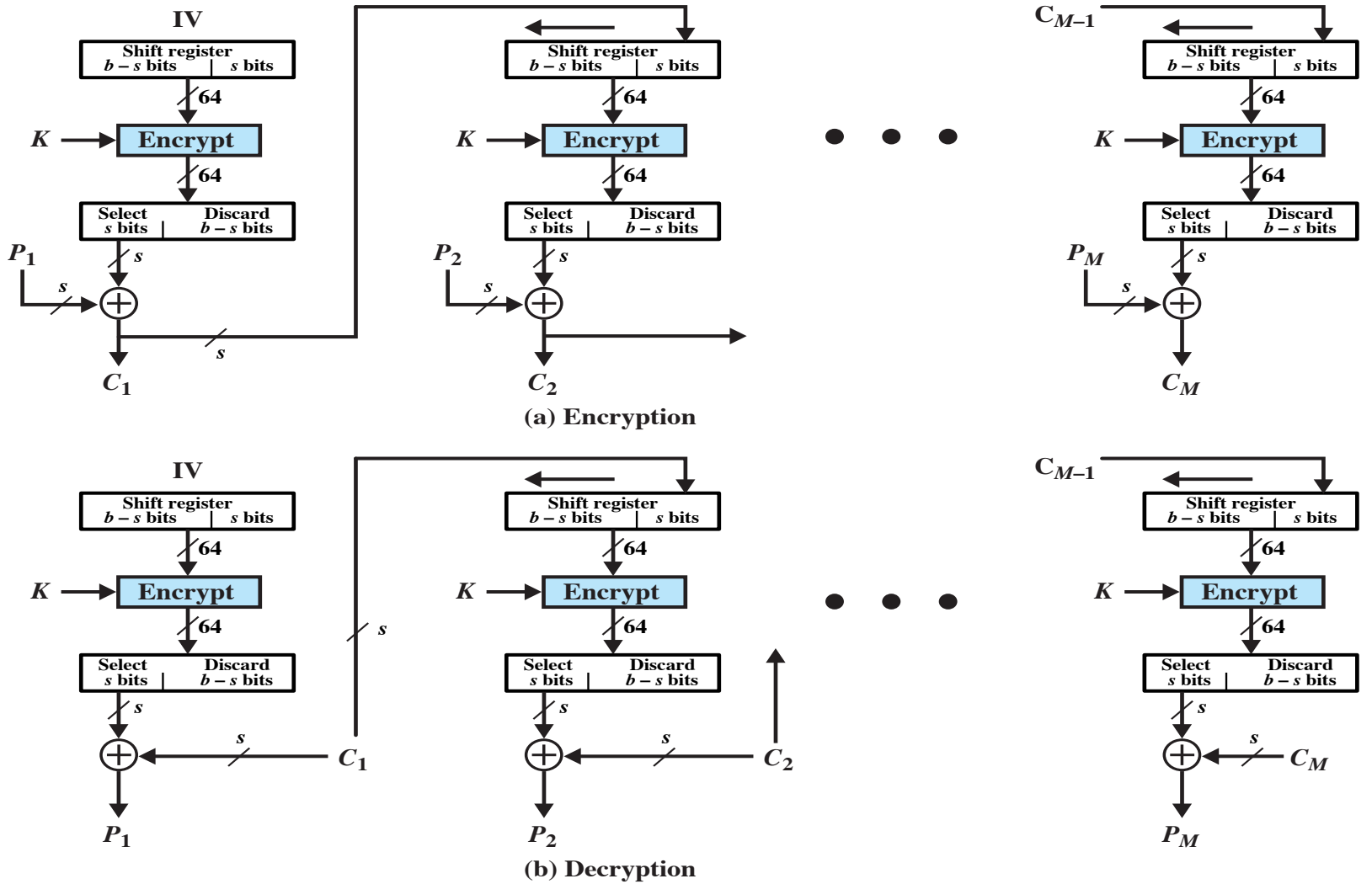
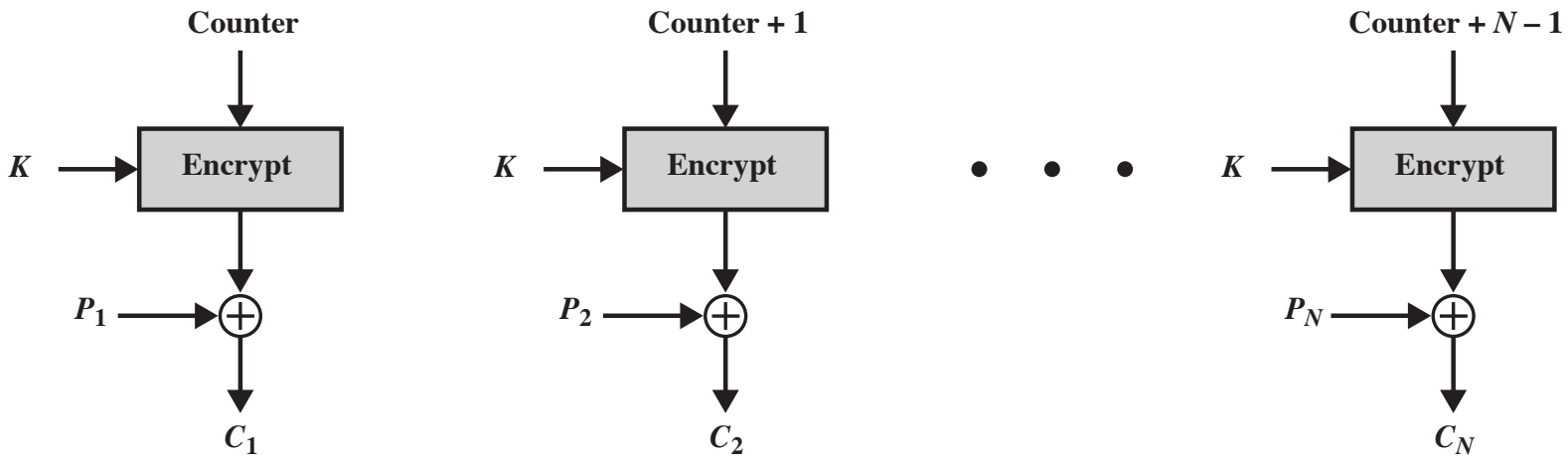


Figure 20.8 *s*-bit Cipher Feedback (CFB) Mode

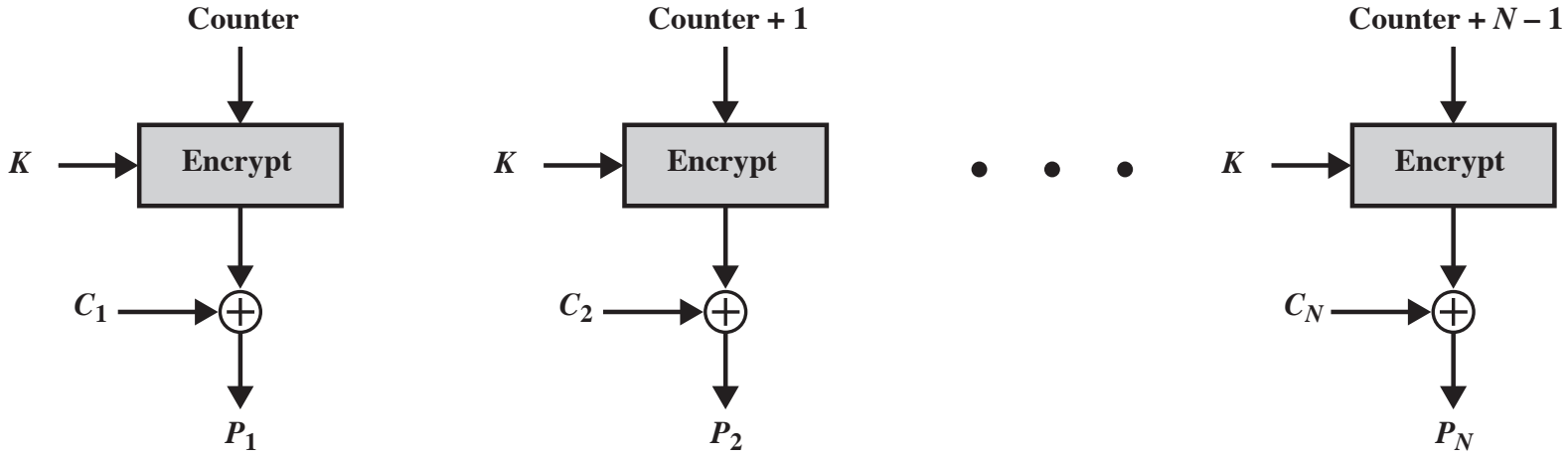
Block Cipher Modes of Operation

Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	•General-purpose block-oriented transmission •Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission •Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission •Useful for high-speed requirements

Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.



(a) Encryption



(b) Decryption

Figure 20.9 Counter (CTR) Mode

Block & Stream Ciphers

Block Cipher

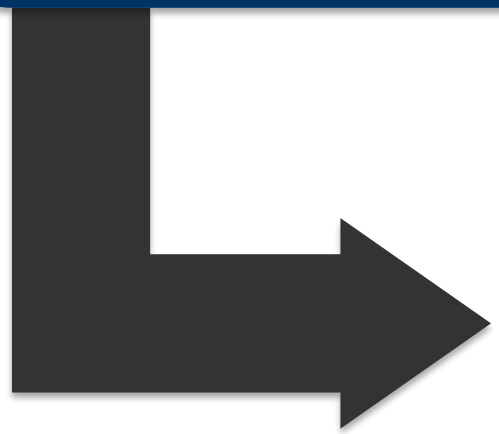
- Processes the input one block of elements at a time
- Produces an output block for each input block
- Can reuse keys
- More common

Stream Cipher

- Processes the input elements continuously
- Produces output one element at a time
- Primary advantage is that they are almost always faster and use far less code
- Encrypts plaintext one byte at a time
- Pseudorandom stream is one that is unpredictable without knowledge of the input key

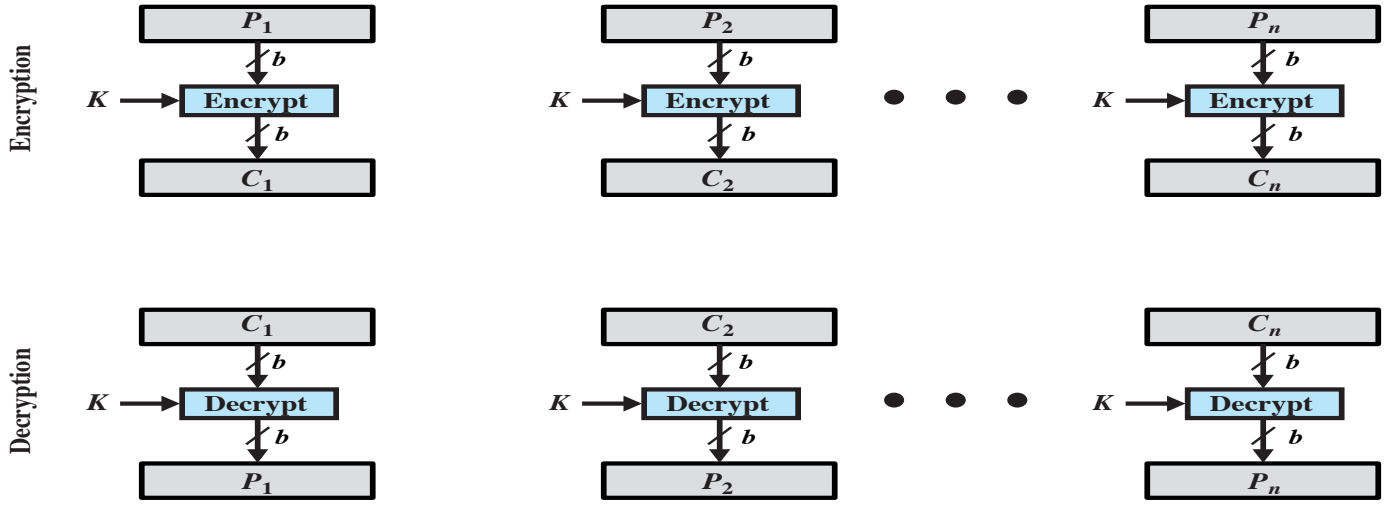
Stream Ciphers

Processes input elements continuously

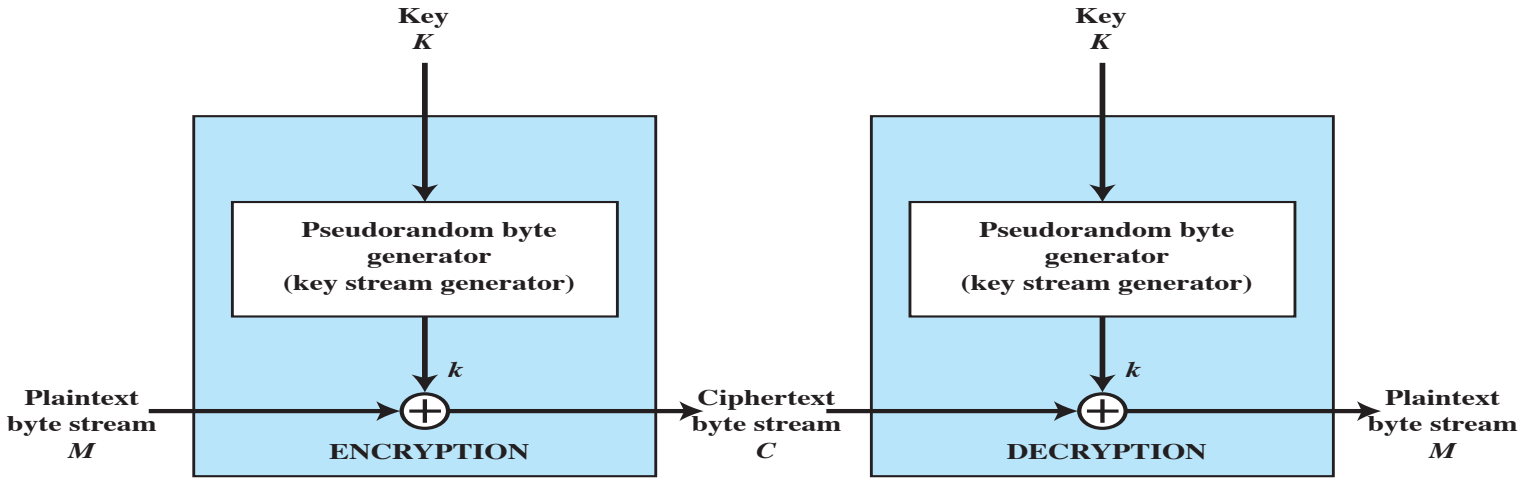


Key input to a pseudorandom bit generator

- Produces stream of random like numbers
- Unpredictable without knowing input key
- XOR keystream



(a) Block cipher encryption (electronic codebook mode)



(b) Stream encryption

Figure 2.2 Types of Symmetric Encryption

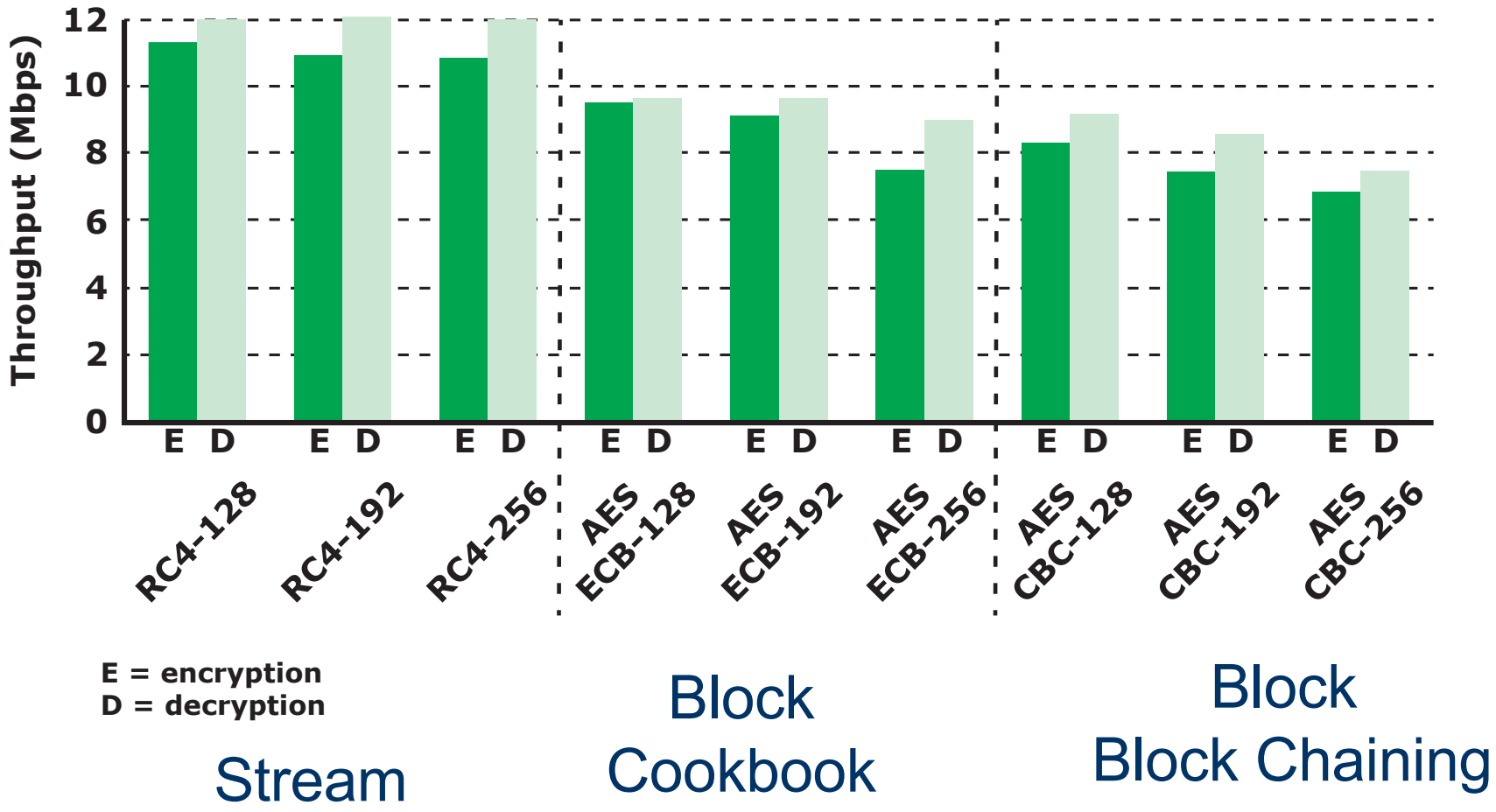


Figure 20.5 Performance Comparison of Symmetric Ciphers on a 3-GHz Processor

Returning to Our Motivating Problems:

Confidentiality: how could we encrypt a message? (email, web/http, sms, etc)

Apply AES – select key size and mode

Integrity: how could we authenticate a message?

Is it authentic because it's encrypted?
No, what if AES-ECB blocks re-ordered

(availability – not primary motivation now, provided the approach is feasible)

Message Authentication Without Confidentiality

- Message encryption by itself does not provide a secure form of authentication
- It is possible to combine authentication and confidentiality in a single algorithm by encrypting a message plus its authentication tag
- Typically message authentication is provided as a separate function from message encryption
- Situations in which message authentication without confidentiality may be preferable include:
  - There are a number of applications in which the same message is broadcast to a number of destinations
  - An exchange in which one side has a heavy load and cannot afford the time to decrypt all incoming messages
  - Authentication of a computer program in plaintext is an attractive service
- Thus, there is a place for both authentication and encryption in meeting security requirements

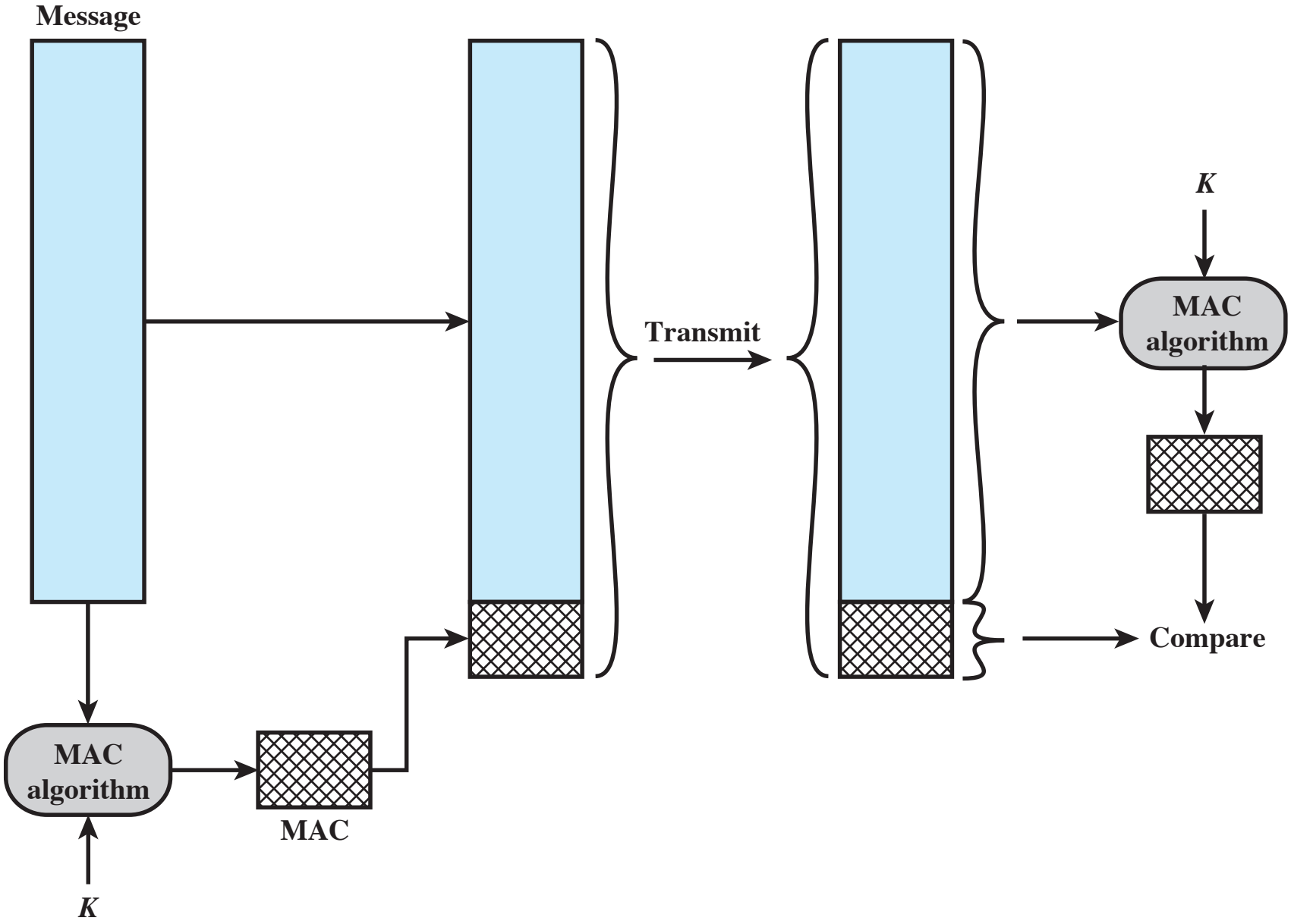


Figure 2.3 Message Authentication Using a Message Authentication Code (MAC).

Message Authentication Code (1/2)

- Sender computes the MAC and sends Message and MAC
  - Let M1 = original message and K = secret key
  - Compute MAC1 = function(M1, K) and Send {M1} || {MAC1}
  - Example: M1 = “Move Left”, MAC1 = $\{Move\ Left\}_K$ where $\{msg\}_K$ denotes the result of encrypting msg with AES key K
  - Send {Move Left} || {MAC1}
- The adversary can read and modify the message!
  - Adversary can change M1. Let M2 = modified message.
  - Example M2 = “Move Right” so receiver gets {Move Right} || {MAC1}
- But the modified message no longer matches the MAC
  - Receiver computes ExpectedMac = function(M2, K)
  - ExpectedMac != ReceivedMac
  - In our example, ExpectedMac = $\{Move\ Right\}_K$, != MAC1

Message Authentication Code (2/2)

- The adversary can also read and modify the MAC!
  - Adversary knows the MAC won't match and knows the MAC function
  - Adversary needs to change the MAC to match the modified message
- But the adversary can't compute the correct MAC
  - MAC = function(M2,K) and adversary doesn't know the key K
- In our example, adversary knows the MAC must be
  - $\{Move\ Right\}_K$ – the message "Move Right" encrypted by AES Key K
  - Without knowing AES key K, it is computationally unfeasible for the adversary to find $\{Move\ Right\}_K$

Returning to Our Motivating Problems:

Confidentiality: how could we encrypt a message? (email, web/http, sms, etc)

Apply AES – select key size and mode

Integrity: how could we authenticate a message?

Apply Message Authentication Code

(availability – not primary motivation now, provided the approach is feasible)

Heilmeier Questions

- What are you trying to do? Articulate your objectives using absolutely no jargon.
- How is it done today, and what are the limits of current practice?
- What is new in your approach and why do you think it will be successful?
- Who cares? If you succeed, what difference will it make?
- What are the risks?
- How much will it cost?
- How long will it take?
- What are the mid-term and final “exams” to check for success?