Part One:

Cryptography

Symmetric Encryption & Message Confidentiality

Before we begin let us take same terminology on cryptography:

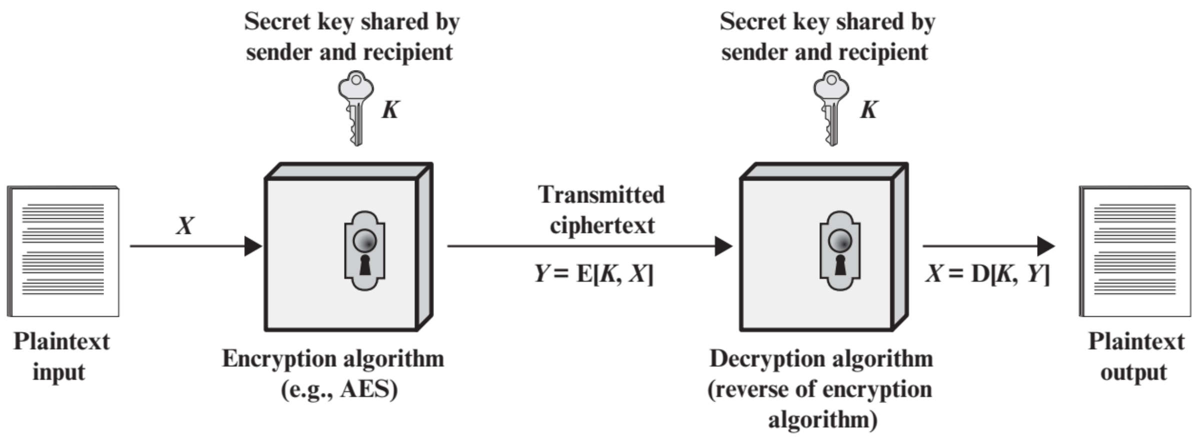
- Plaintext or cleartext: original message
- Ciphertext: coded message
- Cipher: algorithm for transforming plaintext to ciphertext
- Key: information used in cipher known only to sender and receiver (secret)
- Encrypt or Encipher: converting plaintext to ciphertext
- Decrypt or Decipher: recovering ciphertext from plaintext
- Cryptography: study of encryption principles or methods
- Cryptanalysis: study of principles of deciphering ciphertext without knowing key
- Cryptology: Cryptography + Cryptanalysis

Symmetric Encryption Principles

Symmetric Encryption and Decryption have 5 components as follows:

1. Plaintext
2. Encryption algorithm
3. Secret Key
4. Ciphertext

5. Decryption algorithm


Secret key shared by sender and recipient
K
Secret key shared by sender and recipient
K
X
Transmitted ciphertext
Y = E[K, X]
X = D[K, Y]
Plaintext input
Encryption algorithm (e.g., AES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Also, there are 2 requirements for secure use of symmetric encryption:

1. Strong encryption algorithm
2. Sender and receiver must have obtained copies of the secret key in a secure fashion or environment and must keep the key secure

Cryptography

The cryptographic systems are classified along 3 independent dimensions:

1. The type of operations used for transforming plaintext to ciphertext, and all encryption algorithms are based on:
    a. Substitution: each element in plaintext is mapped into another element
    b. Transposition: elements in the plaintext are rearranged
2. The number of keys used
    a. Symmetric: sender and receiver use the same key
    b. Asymmetric or public key: sender and receiver each use a different key
3. The way in which the plaintext is processed

Note

The Security of symmetric encryption depends on the secrecy of the key, not the secrecy of the algorithm. So, manufacturers can develop low-cost chip implementations of data encryption algorithm, and these chips are widely available. The only problem will be maintaining the secrecy of the key.

a. Block cipher: processes the input into one block of elements, producing an output block for each input block
b. Stream cipher: processes the input elements continuously, producing output one element at a time

Cryptanalysis

It is the process of attempting to discover the plaintext or key. The strategy that used by the cryptanalyst depends on:

- The nature of the encryption scheme
- The information available to the cryptanalysis

The cryptanalyst has many types of attacks based on the amount of information he knows. The most difficult one but it is the easiest to defend is ciphertext only attack where the cryptanalyst has only the ciphertext and encryption algorithm and he needs to get the key so he can decrypt the ciphertext. One approach that is used mostly with ciphertext only is to try all possible keys or also known as the brute-force approach. Also,

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

the cryptanalyst could know some patterns in the plaintext, so he could add to his knowledge and try to deduce the key, and this is called known plaintext. If the cryptanalyst has chosen the plaintext message, this is known as chosen plaintext. Finally, the least attacks been used by a cryptanalyst are chosen ciphertext and chosen text.

An encryption scheme is computationally secure if the ciphertext generated by the scheme meets one or both of the following criteria:

- The cost of breaking the cipher exceeds the value of the encrypted information
- The time required to break the cipher exceeds the useful lifetime of the information

Brute Force Attack

It involves trying every possible key until an intelligible translation of the ciphertext into plaintext is obtained. On average, half of all possible keys must be tried to achieve success. Unless known plaintext is provided, the analyst must be able to recognize plaintext as plaintext. To supplement the brute-force approach:

- Some degree of knowledge about the expected plaintext is needed
- Some means of automatically distinguishing plaintext from garble is also needed

Feistel Cipher Structure

It is a symmetric block encryption algorithm that many of ciphers based on it, such as DES. It is first described and named by Horst Feistel of IBM in 1973. The decryption of Feistel cipher is the same as encryption process by reverse ciphertext and plaintext and use subkeys in reverse order. Feistel cipher and all other symmetric block ciphers depends on the following parameters that are the design elements for it:

- Block size: The larger block sizes the greater security, but it reduced encryption/decryption speed.
- Key size: The larger key size the greater security, but it reduces encryption/decryption speed.
- Number of rounds: Multiple rounds offers great security.
- Subkey generation algorithm: greater complexity in it causes greater difficulty of cryptanalysis.
- Round function
- Fast software encryption/decryption
- Ease of analysis: make the algorithm as difficult as possible to cryptanalyze

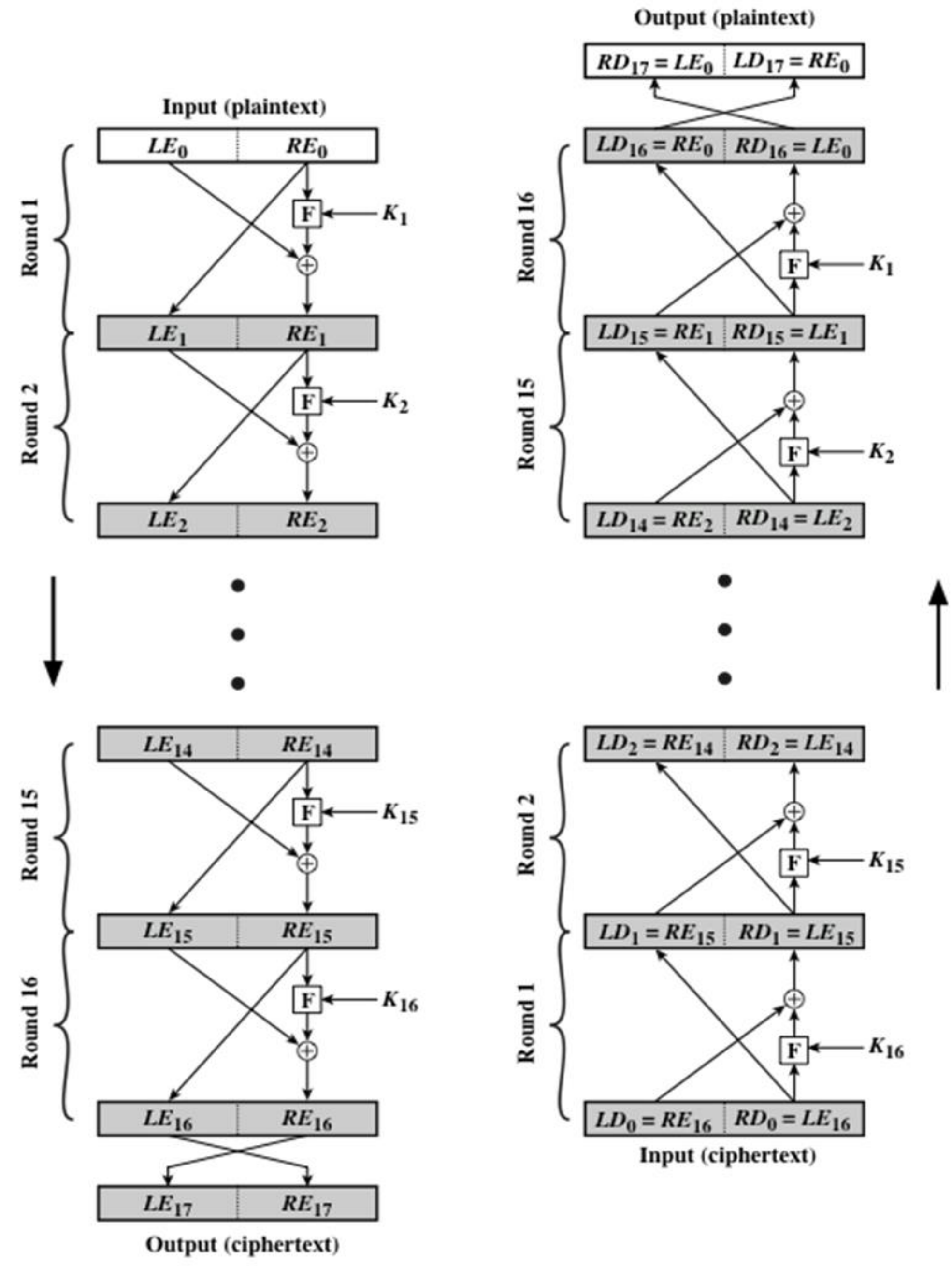


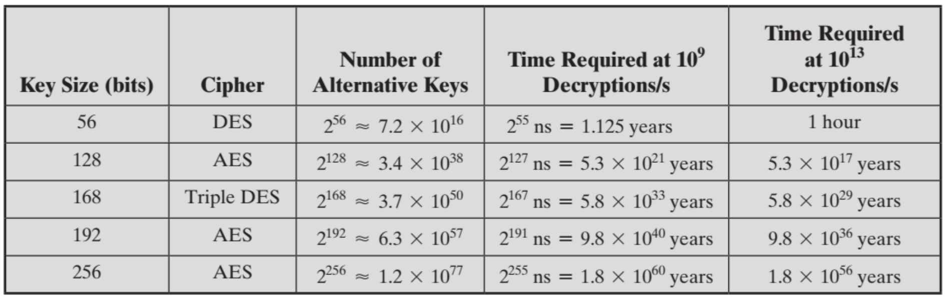
Figure 2.2 Feistel Encryption and Decryption (16 rounds)

Symmetric Block Encryption Algorithms

The block cipher is the most common used symmetric encryption algorithms. It processes the plaintext input in fixed-size blocks and produces a block of ciphertext of equal size for each plaintext block. The most important symmetric block ciphers are DES, 3DES, and AES.

Data Encryption Standard (DES)

DES or also known as Data Encryption Algorithm (DEA) is issued by NIST in 1977 as federal standard FIPS 46. The plaintext is 64 bits long and the key size is 56 bits, all processed in 16 rounds. The 56-bit key generate 16 subkeys. The structure of DES is same as Feistel cipher, and as Feistel cipher encryption and decryption are the same, DES also its encryption and decryption are the same. The strength of DES fall into 2 categories. First, the algorithm itself refers to the possibility that cryptanalysis is possible by exploiting the characteristics of the algorithm. Second, the use of a 56-bit key that is speed of commercial, off-the-shelf processors threatens the security. At that time, DES was unbreakable because with a key of 56 bits size there are 2^{56} possible keys which makes Brute-force attack impractical. In 1998, Electronic Frontier Foundation (EFF) announced that it had broken a DES encryption using a DES cracker machine that was built for less than $250,000. The table below shows the average time required for exhaustive key search.



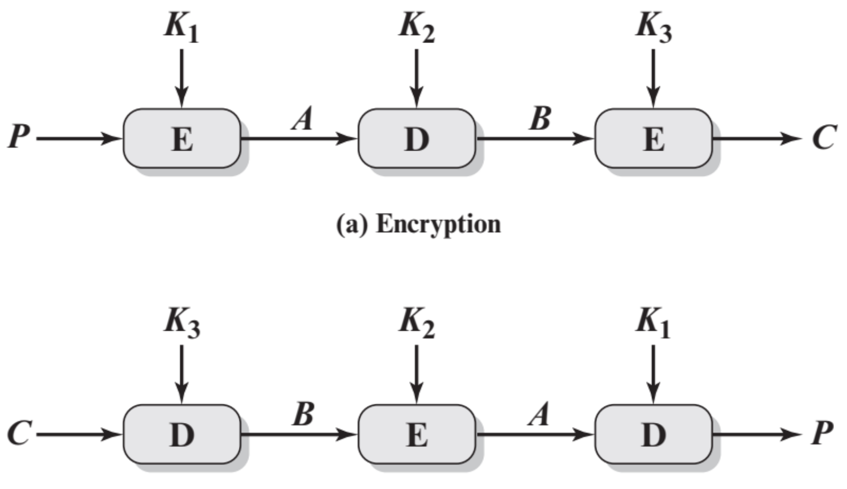
Key Size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 Decryptions/s	Time Required at 10^{13} Decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple 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

Triple DES (3DES)

It was first standardized for use in financial applications in ANSI standard X9.17 in 1985. It uses 3 keys and 3 executions of the DES algorithm. It follows encrypt-decrypt-encrypt (EDE) sequence in encryption and (DED) in decryption.

$$C = E(K_3, D(K_2, E(K_1, P)))$$

$$P = D(K_1, E(K_2, D(K_3, C)))$$



(b) Decryption

Decryption is the same operation but with the keys in reverse order. The key length for 3DES is 168 bits that is 3 keys of DES each 56 bits. The FIPS 46-3 guidelines for 3DES as follows:

1. 3DES is the FIPS-approved symmetric encryption algorithm of choice.
2. The original DES, which uses a single 56-bit key, is permitted under the standard for legacy systems only. New procurements should support 3DES.
3. Government organizations with legacy DES systems are encouraged to transition to 3DES.
4. It is anticipated that 3DES and the Advanced Encryption Standard (AES) will coexist as FIPS-approved algorithms, allowing for a gradual transition to AES.

Also, as the key length is 168 bits then Brute-force attack on 3DES is impossible. The 3DES has 2 attractions to be used widespread:

- Its 168-bit key length
- The encryption algorithm in 3DES is the same as in DEA, that this algorithm overcame over Brute-force attack, and it is very resistant to cryptanalysis.

The drawback of 3DES is that the algorithm is relatively sluggish or very slow in software, and it still use 56-bit key size and 64-bit block size as in DEA but in 3 times. 3DES was remain an approved algorithm for U.S. government until AES has appeared.

Advanced Encryption Standard (AES)

In 1997, NIST issued a proposal for a new Advanced Encryption Standard to be equal or better than 3DES. And after evaluating and narrowing between the algorithms, NIST selected Rijndael to be AES algorithm. Rijndael made by two Belgium researchers Dr. Joan Daemen, and Dr. Vincent Rijmen. AES uses block length of 128 bits, and a key length that can be 128, 192, or 256 bits. It does not use Feistel structure but processes the entire data in parallel. The key will split into 4 words each is 32-bit. AES has 4 different stages as follows:

1. Substitute bytes: use S-box to substitute byte-by-byte
2. Shift rows: permutation row-by-row
3. Mix columns: substitution that alters each byte in a column as a function of all of the bytes in the column
4. Add round key: XOR of the current block with expanded key

In decryption algorithm of AES, each stage is reversible, and the expanded keys will be in reverse order. The decryption algorithm is not identical to the encryption algorithm, but we need to reverse all steps to decrypt.

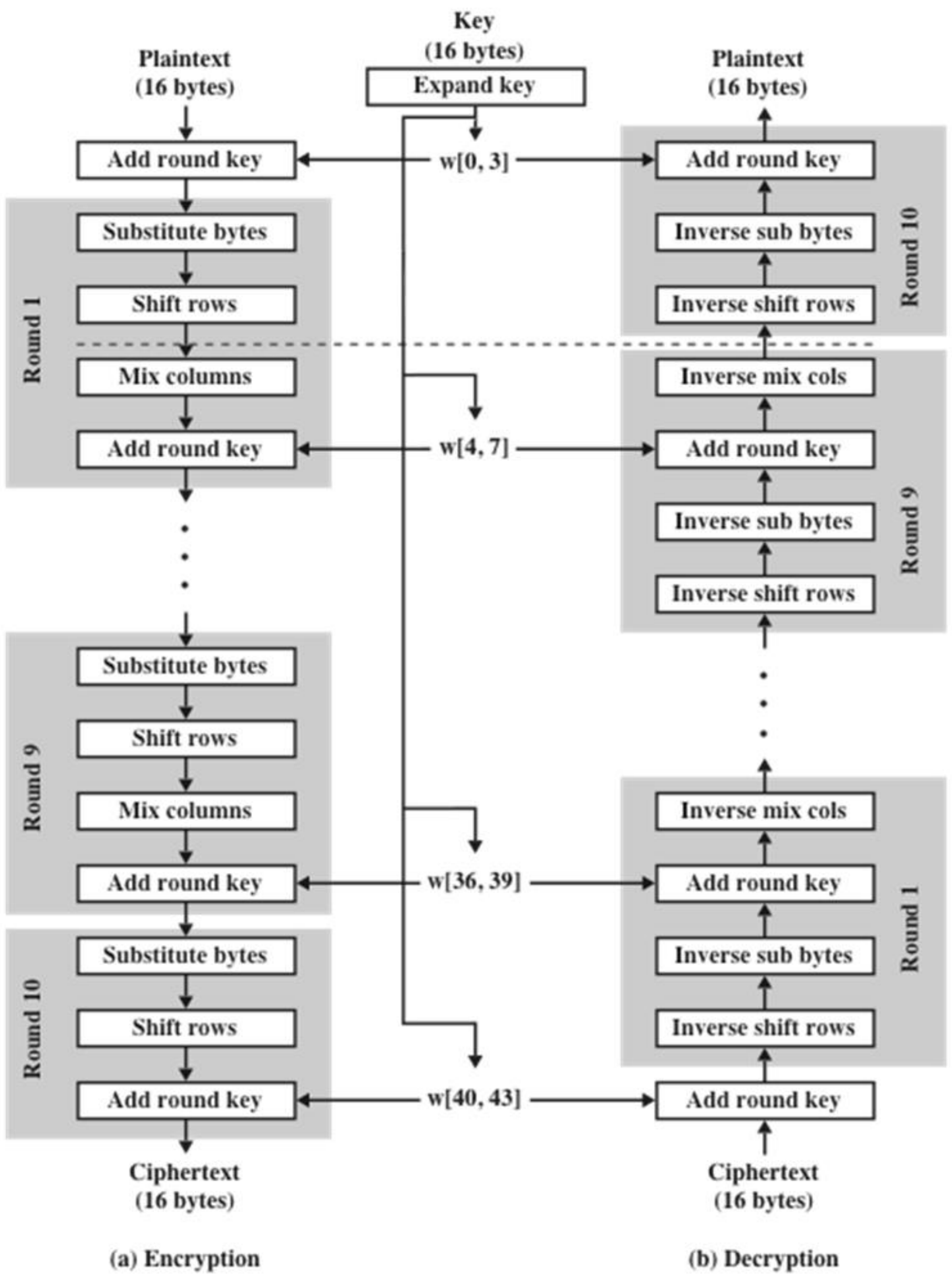


Figure 2.4 AES Encryption and Decryption

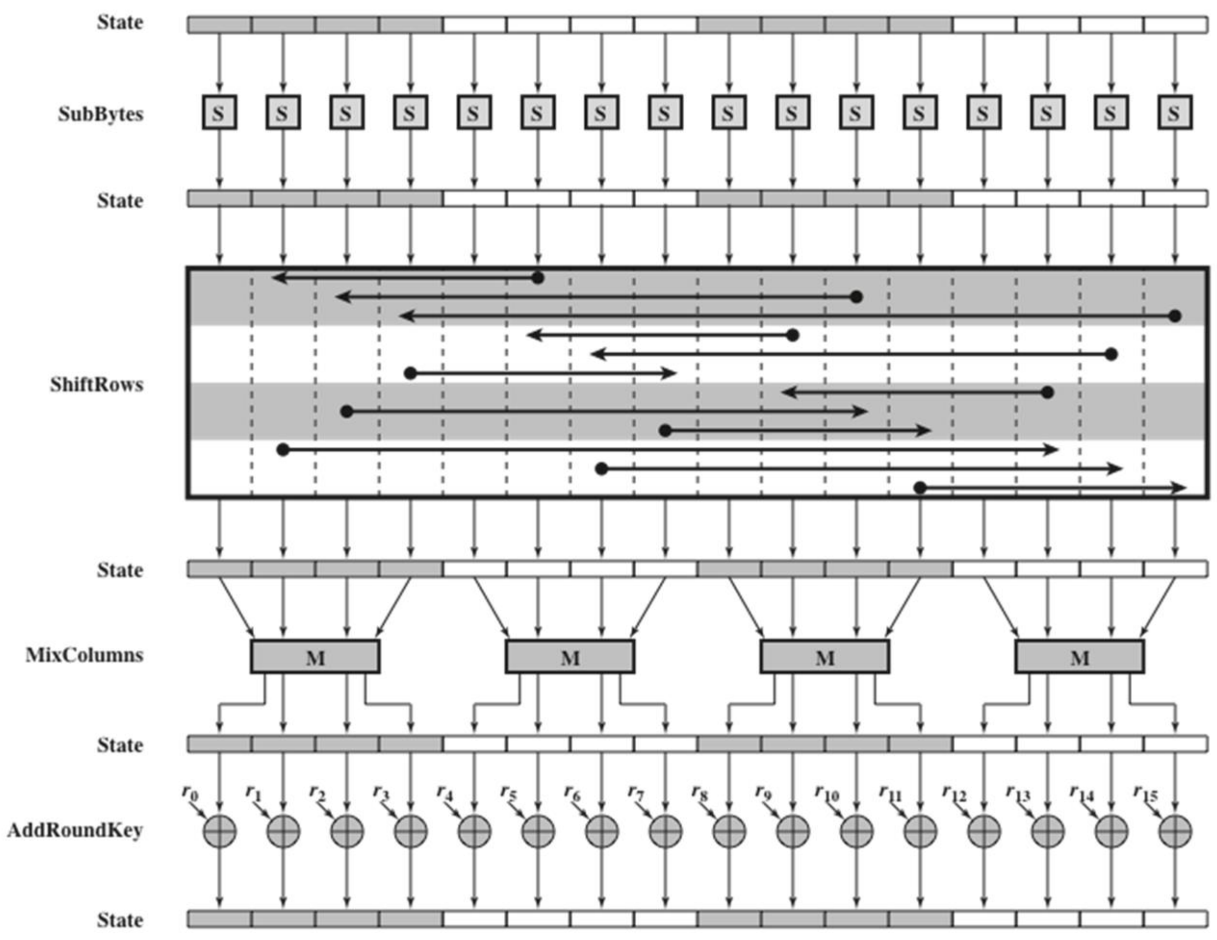

State
SubBytes
S
State
ShiftRows
State
MixColumns
M
State
AddRoundKey
r_0 r_1 r_2 r_3 r_4 r_5 r_6 r_7 r_8 r_9 r_{10} r_{11} r_{12} r_{13} r_{14} r_{15}
State

Figure 2.5 AES Encryption Round

<u>Random And Pseudorandom Numbers</u>

The use of Random Numbers

The random numbers have an important rule in the use of encryption. Many of network security algorithms use random numbers. For example, RSA public-key, stream key for stream ciphers, symmetric key for use as a session key in such as transport layer security, or Wi-Fi, and in key distribution scenarios such as Kerberos. A two distinct and not necessarily compatible requirements for a sequence of random numbers are randomness, and unpredictability.

Randomness

To validate that a sequence of numbers is random we use the following criteria:

- Uniform distribution: The frequency of occurrence of 1s and 0s should approximately the same. It is when all probabilities in the distribution are equal, meaning that all outcomes are equally likely to occur. For example, if 128-bit key is picked uniformly at random, then each of the 2^{128} possible keys should have a probability of $1/_{2^{128}}$.
- Independence: No one subsequence in the sequence can be inferred from the others. The is no such test to prove independence, but the general strategy is to apply several such tests until the confidence that independence exists is sufficiently strong.

Unpredictability

In applications such as reciprocal authentication and session key generation, the requirement is not so much that the sequence of numbers be statistically random but that the successive members of the sequence are unpredictable. With "true" random sequences, each number is statistically independent of other numbers in the sequence and therefore unpredictable. care must be taken that an opponent is not able to predict future elements of the sequence based on earlier elements. Finally, it must be hard to predict next value in sequence.

TRNGs, PRNGs, and PRFs

Cryptographic applications typically make use of deterministic algorithmic techniques for random number generation, producing sequences of numbers that are not statistically random, but if the algorithm is good, the resulting sequences will pass many reasonable tests of randomness. Such numbers are referred to as pseudorandom numbers, created by Pseudorandom Number Generators (PRNGs). We have 3 random generators as follows:

1. True Random Number Generator (TRNG): It come from non-deterministic source, from physical environment of the computer. It detects ionizing radiation events, leaky capacitors, thermal noise from resistors or audio inputs. Also, Mouse or keyboard activity, I/O operations, interrupts
2. Pseudorandom Number Generators (PRNG): It comes from deterministic algorithms to calculate numbers in relatively random sequence. The algorithm input is called the seed. It produces continuous stream of random bits.
3. Pseudorandom Function (PRF): It is same as PRNG but produces string of bits of some fixed length. So, it generates one value of fixed length, while PRNG generates continuous sequence.

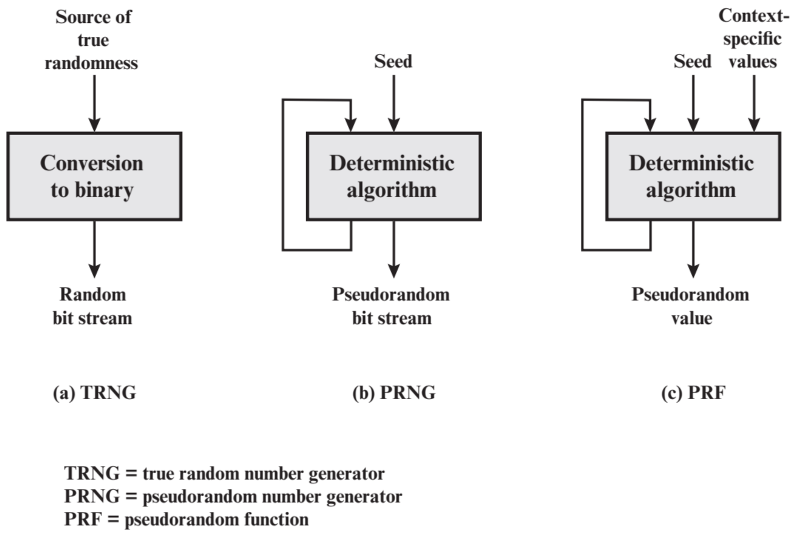


Figure 2.6 Random and Pseudorandom Number Generators

Algorithm Design

There are many algorithms been shown at that time to develop PRNGs. These algorithms fall into two categories as follows:

- Purpose-built algorithms: they algorithms designed only for the purpose of generating pseudorandom bit streams. For example, RC4.
- Algorithms based on existing cryptographic algorithms: they are designed based on 3 broad cryptographic algorithms:
    - Symmetric block ciphers
    - Asymmetric ciphers
    - Hash functions & Message authentication codes (MAC)

<u>Stream Ciphers and RC4</u>

As we say before, block ciphers such as DES, 3DES, and AES, processes the input one block of elements at a time, producing an output block for each input block. While stream ciphers such as RC4, processes the input elements continuously, producing output one element at a time.

Stream Cipher Structure

Stream cipher encrypts plaintext one byte at a time and produce ciphertext that is one byte at a time. In stream cipher, the key is input to a pseudorandom byte generator that produces a stream of 8-bit numbers that are random. The output of the pseudorandom generator known as keystream. The keystream XORed with plaintext to produce the ciphertext. And vice versa, the keystream XORed with ciphertext will produce the plaintext.

$$Ciphertext = Plaintext \oplus Keystream$$

$$Plaintext = Ciphertext \oplus Keystream$$

The design considerations for stream cipher as follows:

- The encryption sequence should have a large period. The pseudorandom number generator uses a function that produces a deterministic stream of bits that repeats, and the longer the period repeat, the more difficult to a cryptanalysis.
- The keystream should approximate the properties of a true random number stream. That should be an approximately equal number of 0s and 1s.
- The key needs to be long

The advantage of stream cipher is that its faster than block. And the advantage of block cipher is that you can reuse keys. Stream ciphers and block ciphers each used in specific applications. For example, stream ciphers used in application like data-communications channel, or a browser. While block ciphers used in applications like file transfer, email, and database.

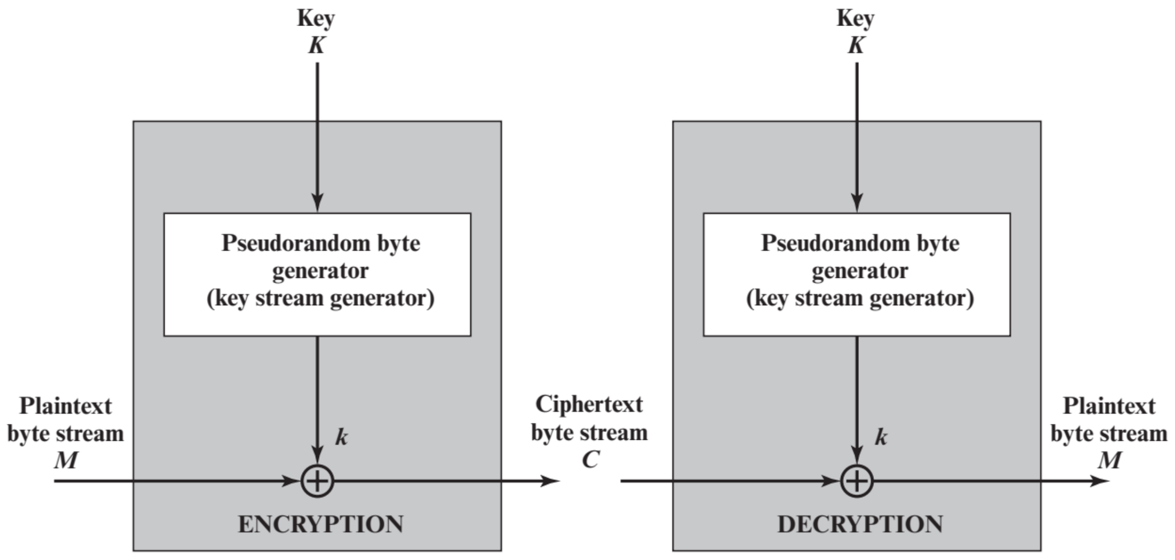

Key
K
Pseudorandom byte generator (key stream generator)
Plaintext byte stream
M
k
ENCRYPTION
Ciphertext byte stream
C
Key
K
Pseudorandom byte generator (key stream generator)
k
DECRYPTION
Plaintext byte stream
M

Figure 2.7 Stream Cipher Diagram

The RC4 Algorithm

It is a stream cipher designed by Ron Rivest in 1987. It is a variable key-size stream cipher with byte-oriented operations. It is based on the use of a random permutation. The period of RC4 is likely to be greater than 10100. 8 – 16 machine operations are required per output byte. It run very fast in software. It is used in SSL/TLS standards, WEP protocol, and WPA Wi-Fi protocol. RC4 uses a variable-length key of 256 bits (0 – 255), and it used to initialize a 256-byte state vector (S). S contains a permutation of all 8-bit numbers from 0 to 255. For encryption and decryption, a byte (k) is generated from S by selecting one of the 255 entries. As each value of k is generated, the entries in S are once again permuted. The algorithm of RC4 is shown below.

```
/* Initialization */
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]);

/* Stream Generation */
i, j = 0;
while (true)
  i = (i + 1) mod 256;
  j = (j + S[i]) mod 256;
  Swap (S[i], S[j]);
  t = (S[i] + S[j]) mod 256;
  k = S[t];
```

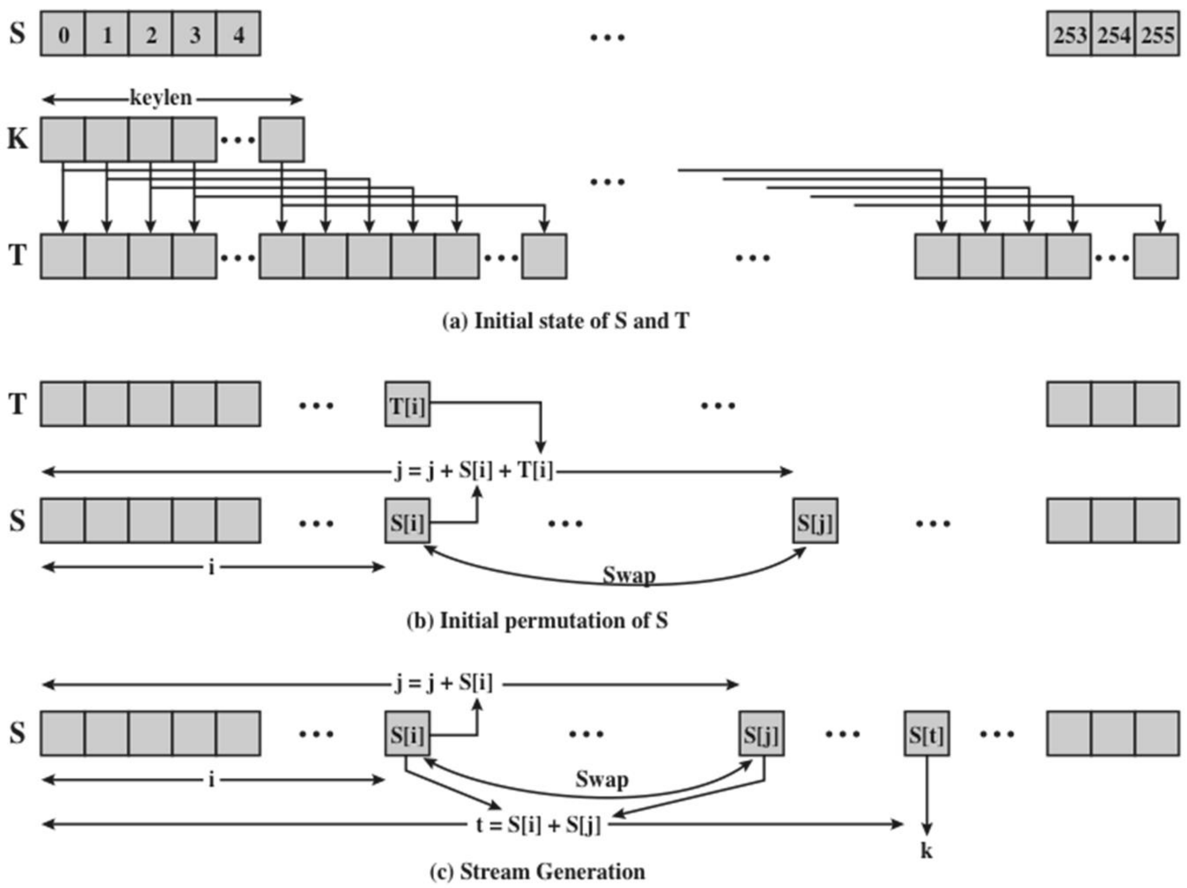


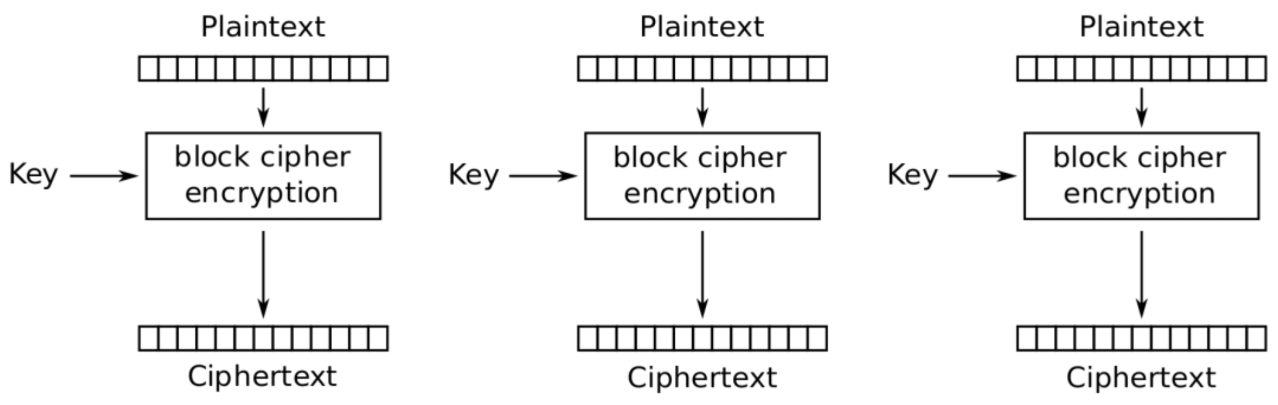
Figure 2.8 RC4

Cipher Block modes of Operation

NIST has published possible applications of encryption for which a symmetric block cipher could be used. They are 5 modes known as the modes of operation.

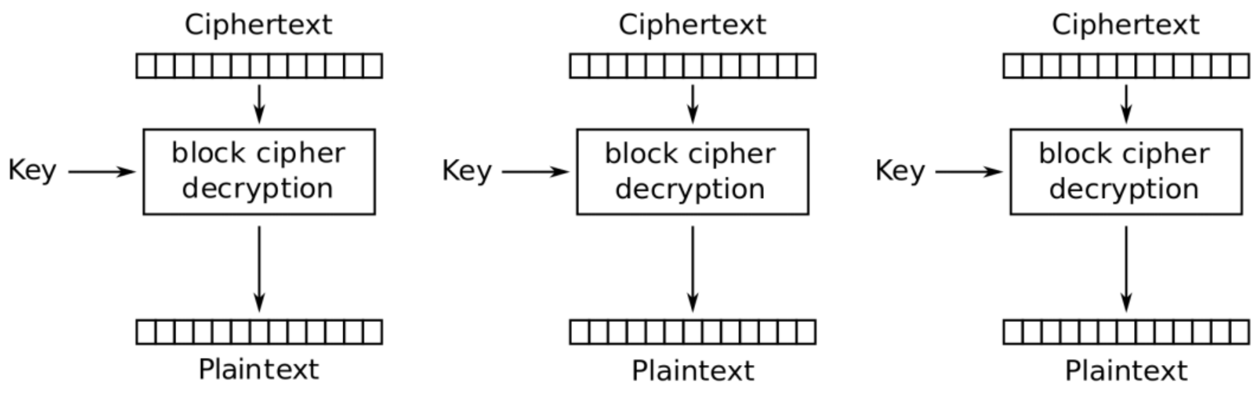
Electronic Codebook (ECB) Mode

It is the simplest mode of operation that it takes each plaintext block and encrypt it with the secret key to provide a ciphertext. The decryption process same as the encryption, just invert ciphertext and plaintext. The problem of ECB mode is that it could produce same ciphertext if the plaintext has been repeated.



Credit: Wikimedia https://commons.wikimedia.org/wiki/File:ECB_encryption.svg, public domain

Figure 11.1: ECB Encryption



Credit: Wikimedia https://commons.wikimedia.org/wiki/File:ECB_decryption.svg, public domain

Figure 11.2: ECB Decryption

Cipher Block Chaining (CBC) Mode

The input to encryption algorithm of CBC mode is XOR of next plaintext and preceding ciphertext. The first block only the plaintext will be XORed with initialization vector (IV) that is the initial state for the block, and it is pseudorandom number, and it must be known only to sender and receiver. On decryption, the IV will be XORed with the output of the decryption algorithm to recover the plaintext. It is recommended to send the IV using ECB encryption. The reason to protect the IV, because if the attacker trick the receiver into use a different IV value, then the attacker can invert selected bits in the first block of plaintext.

$$C_j = E(K, [C_{j-1} \oplus P_j])$$

We can decrypt the following encryption equation as follows

$$D(K, C_j) = D(K, E(K, [C_{j-1} \oplus P_j]))$$

$$D(K, C_j) = C_{j-1} \oplus P_j$$

$$C_{j-1} \oplus D(K, C_j) = C_{j-1} \oplus C_{j-1} \oplus P_j = P_j$$

The final decryption equation is $P_j = IV \oplus D(K, C_j)$.

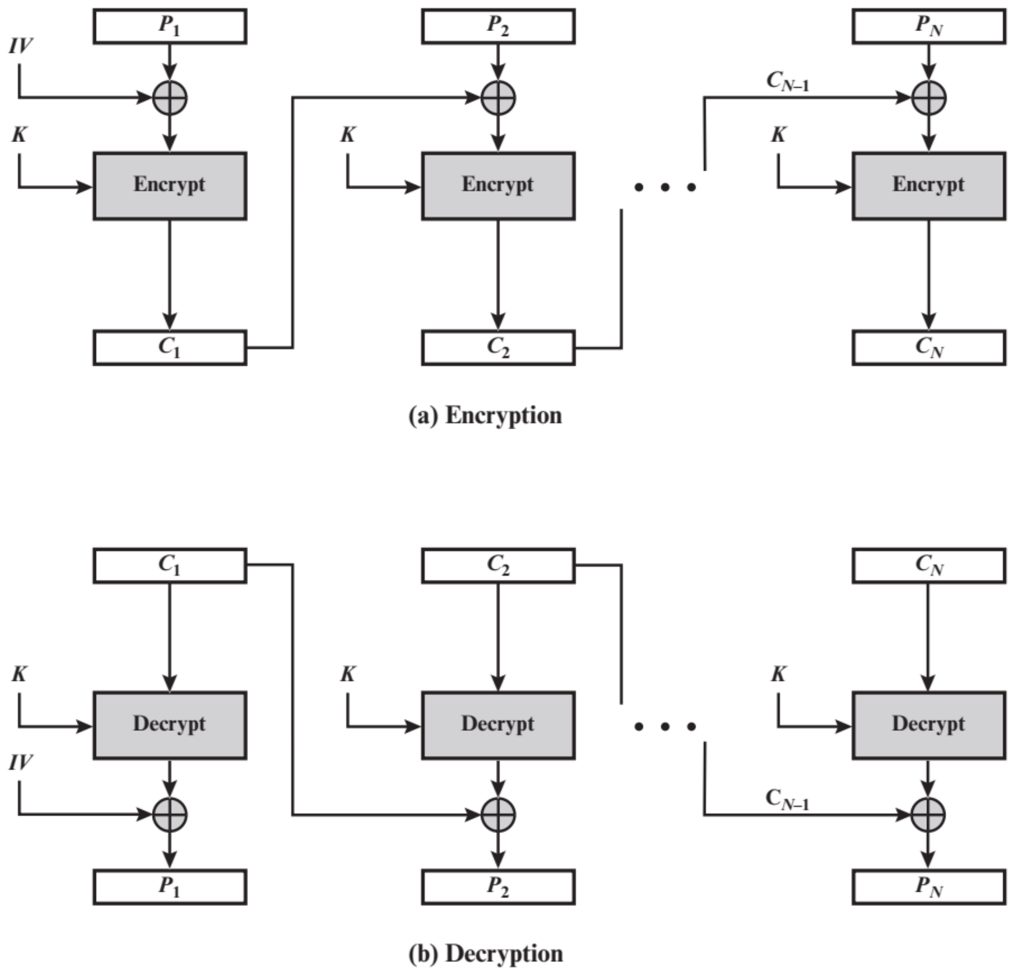


Figure 2.9 Cipher Block Chaining (CBC) Mode

Cipher Feedback (CFB) Mode

The CFB mode converts block cipher into stream cipher. It does not need to pad message to integral number of blocks. It operates in real-time that each character encrypted and transmitted immediately. The input processed s bits at a time. The preceding ciphertext used as input to produce pseudorandom output. XOR output with plaintext to produce ciphertext.

$$C = P \oplus S_s[E(K, IV)]$$

$$P = C \oplus S_s[E(K, IV)]$$

Where S_s is the most significant s bits.

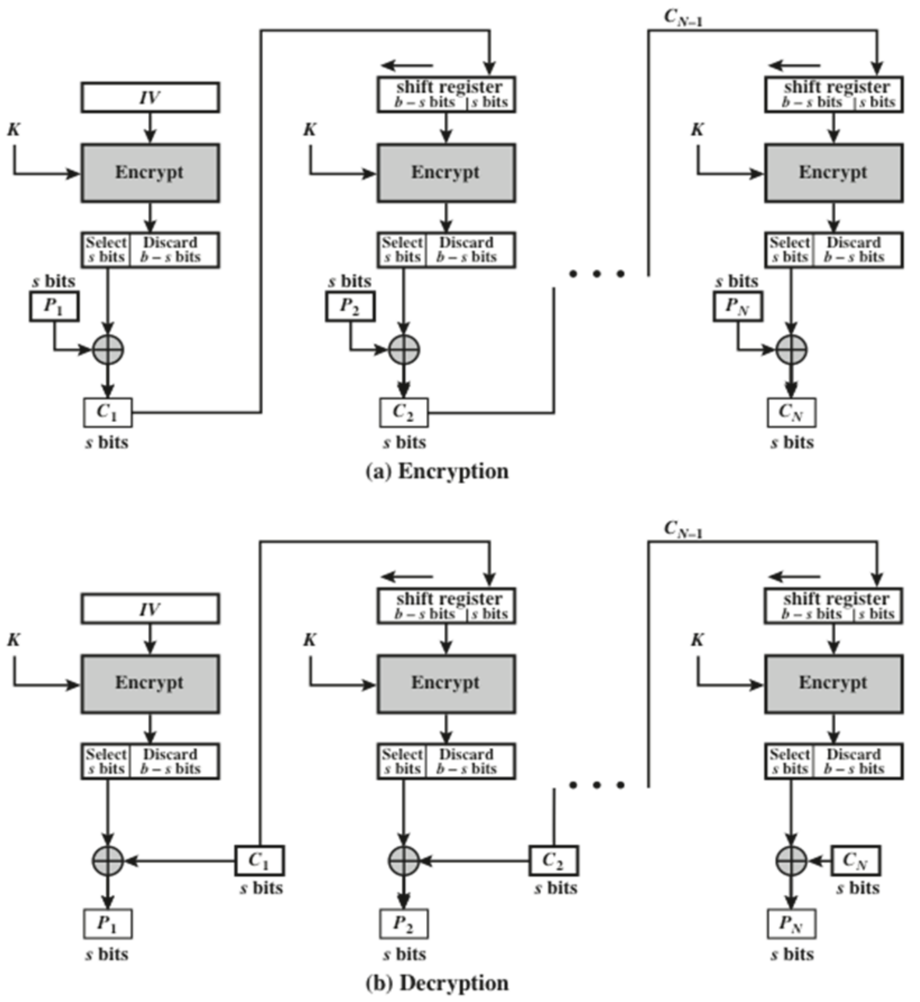


Figure 2.10 ***s*-bit Cipher Feedback (CFB) Mode**

Counter (CTR) Mode

CTR mode is also converts block cipher into stream cipher. Each block of plaintext in CTR mode XORed with encrypted counter. The counter must be equal to the plaintext block size. The counter value must be different for each plaintext block. Also, the counter is initialized to some value and then incremented by 1. In encryption, the counter is encrypted with the key then XORed with the plaintext block to produce the ciphertext block. There is no chaining in CTR mode.

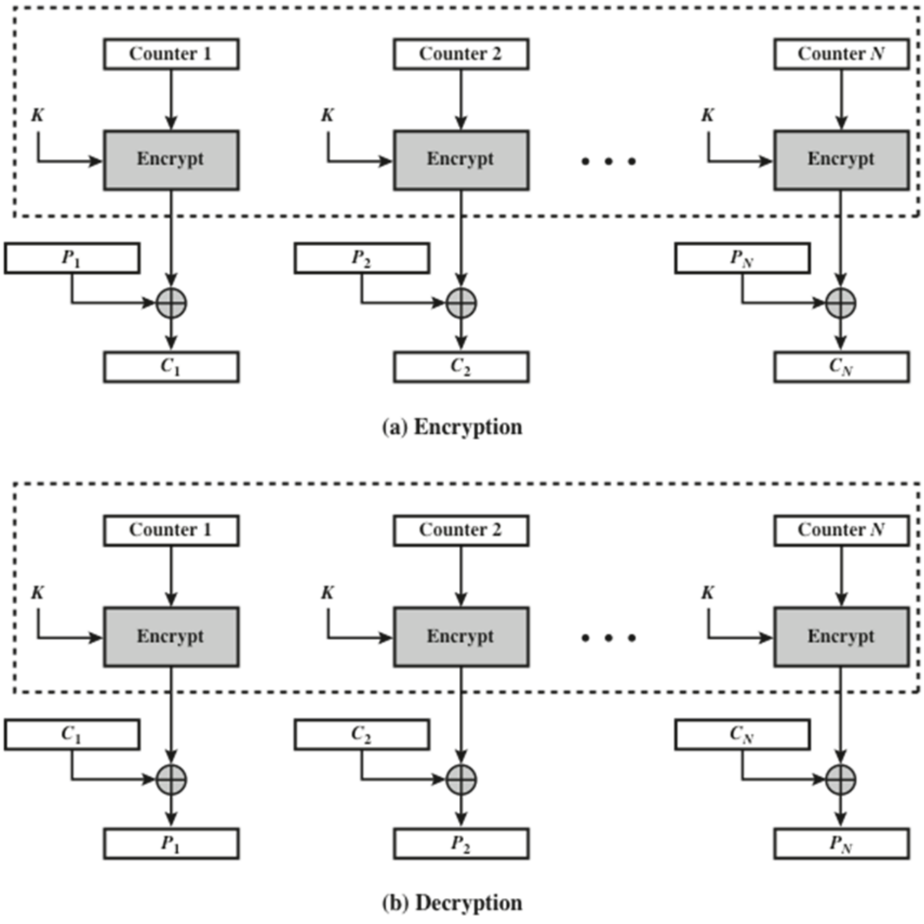


Figure 2.11 Counter (CTR) Mode

Advantage of CTR mode as follows:

- Hardware efficiency: Encryption/decryption can be done in parallel on multiple blocks of plaintext or ciphertext. The throughput is only limited by the amount of parallelism that is achieved.
- Software efficiency: Because of the opportunities for parallel execution, processors that support parallel features can be effectively utilized.
- Pre-processing: The execution of the underlying encryption algorithm does not depend on input of the plaintext or ciphertext --- when the plaintext or ciphertext

input is presented, the only computation is a series of XORs, greatly enhancing throughput.

- Random Access: The i^{th} block of plaintext or ciphertext can be processed in random-access fashion.
- Provable Security: It is at least as secure as the other modes.
- Simplicity: Requires only the implementation of the encryption algorithm and not the decryption algorithm.

References & Useful Resources

- Design and analysis of security protocols course, UAE University, Dr. Khalid Shuaib slides
- William Stallings, Network Security Essentials Applications and Standards, Sixth Edition, 2014, Pearson. ISBN-13: 9780133370430