Cybersecurity

Professor: F. d'Amore

A.Y. 2022/23

Table of Contents

List of Figures

List of Tables

1 Introduction and terminologies

2 Symmetric Encryption

Secret key cryptography involves the use of a single key. Given a message (the plaintext) and the key, encryption produces unintelligible data which is about the same length as the plaintext was.
Secret key cryptography is sometimes referred to as **conventional cryptography** or **symmetric cryptography**.
Secret key encryption schemes require that both the party that does the encryption and the party that does the decryption share a secret key. We will discuss two types of secret key encryption schemes:

- **Stream Ciphers**: This uses the key as a seed for a pseudorandom number generator, produces a stream of pseudorandom bits, and $\oplus$ s (bitwise exclusive ors) that stream with the data. Since $\oplus$ is its own inverse, the same computation performs both encryption and decryption.
- **Block Ciphers**: This takes as input a secret key and a plaintext block of fixed size (older ciphers used 64-bit blocks, modern ciphers use 128-bit blocks). It produces a ciphertext block the same size as the plaintext block. To encrypt messages larger than the blocksize, the block cipher is used iteratively with algorithms called *modes of operation*. A block cipher also has a decryption operation that does the reverse computation.

So, block ciphers encrypt data in blocks of set lengths, while stream ciphers do not and instead encrypt plaintext one byte at a time. The two encryption approaches, therefore, vary widely in implementation and use cases.
Here we will have some prerequisite concepts and then we will dive into these two encryption algorithms

2.1 Finite Fields

2.2 Per-Round Keys

2.3 S-boxes and Bit Shuffles

2.4 Feistel Cipher

It is very important to make the encryption algorithm revered so that the decryption is possible. One method (used by AES) of having a cipher is to make all components reversible. With DES, the S-boxes are clearly not reversible since they map 6-bit inputs to 4-bit outputs. So instead, DES is designed to be reversible using a clever technique known as a Feistel cipher.

A Feistel cipher builds reversible transformations out of one-way transformations by only working on half the bits of the input value at a time. let us assume a 64-bit input block. (Figure 1) shows both how encryption and decryption work. In a Feistel cipher there is some irreversible component that scrambles the input. We'll call that component the mangler function.

In encryption for round n, the 64-bit input to round n is divided into two 32-bit halves called L_n and R_n. Round n generates as output 32-bit quantities L_{n+1} and R_{n+1}. The concatenation of L_{n+1} and R_{n+1} is the 64-bit output of round n and, if there's another round, the input to round n+1. L_{n+1} is simply R_n. To compute R_{n+1}, do the following. R_n and K_n are input to the mangler function, which takes as input 32 bits of data plus some bits of key to produce a 32-bit output. The 32-bit output of the mangler is XORed with L_n to obtain R_{n+1}.

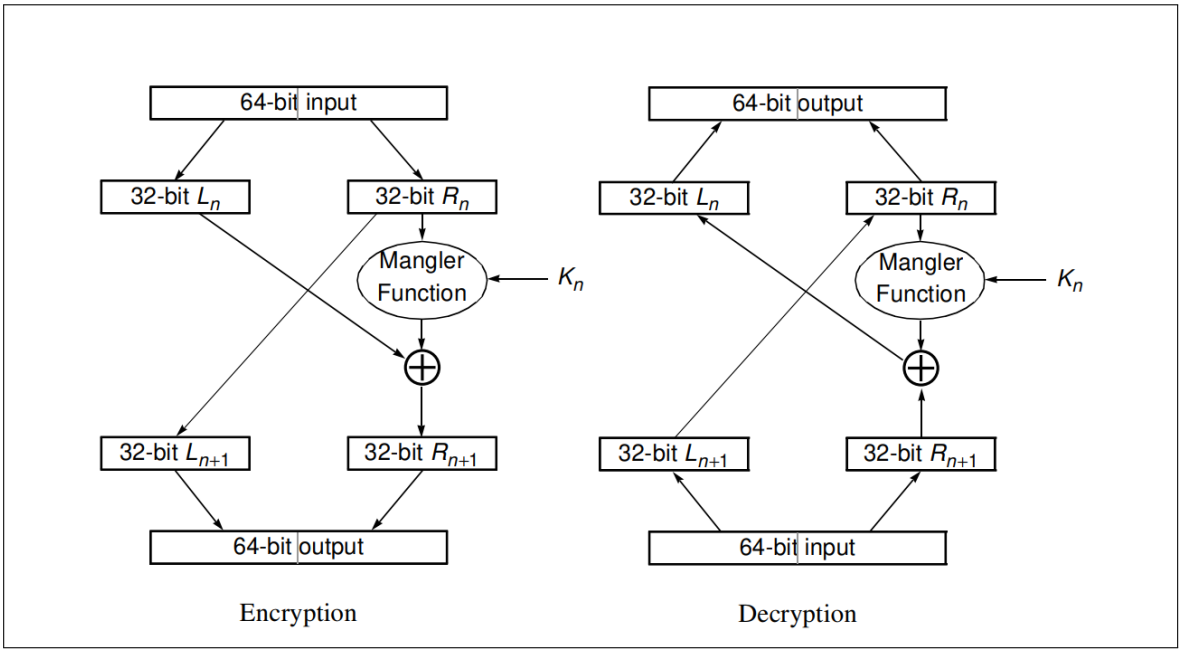

Figure 1: Feistel Cipher

2.5 Stream Ciphers

Idea: try to simulate one-time pad

A stream cipher encrypts a continuous string of binary digits by applying time-varying transformations on plaintext data. Therefore, this type of encryption works bit-by-bit, using keystreams to generate ciphertext for arbitrary lengths of plain text messages. The cipher combines a key (128/256 bits) and a nonce digit (64-128 bits) to produce the **keystream** — a pseudorandom number XORed with the plaintext to produce ciphertext. While the key and the nonce can be reused, the keystream has to be unique for each encryption iteration to ensure security. Stream encryption ciphers achieve this using feedback shift registers to generate a **unique nonce (number used only once) to create the keystream**.

Encryption schemes that use stream ciphers are less likely to propagate system-wide errors since an error in the translation of one bit does not typically affect the entire plaintext block. Stream encryption also occurs in a **linear, continuous manner**, making it simpler and faster to implement. On the other hand, stream ciphers lack diffusion since each plaintext digit is mapped to one ciphertext output. Additionally, they do not validate authenticity, making them vulnerable to insertions. If hackers break the encryption algorithm, they can insert or modify the encrypted message without detection. Stream ciphers are mainly used to encrypt data in applications where the amount of plain text cannot be determined and in low latency use-cases.

2.6 Types of Stream Ciphers

Stream ciphers fall into two categories:

Synchronous Stream Ciphers:

- In a synchronous stream cipher, the keystream block is generated independently of the previous ciphertext and plaintext messages. This means that **each keystream block is generated based only on the key** and does not depend on any previous blocks.
- The most common stream cipher modes use pseudorandom number generators (PRNGs) to create a string of bits, which is combined with the key to form the keystream.
- The keystream is then XORed with the plaintext to generate the ciphertext.

Self-Synchronizing/Asynchronous Stream Ciphers:

- A self-synchronizing stream cipher, also known as ciphertext autokey, **generates the keystream block as a function of both the symmetric key and the fixed-size (N-bits) previous ciphertext block**.
- By altering the ciphertext, the content of the next keystream is changed. This property allows self-synchronizing ciphers to detect active attacks because any modification to the ciphertext will affect the decryption of subsequent blocks, making it easier to detect tampering.
- Asynchronous stream ciphers also provide limited error propagation. If there is a single-digit error in the ciphertext, it can affect at most N bits (the size of the previous ciphertext block) in the next keystream block

.

2.7 Stream Ciphers in practice

popular encryption schemes that use stream ciphers include:

2.7.1 A5/1

2.7.2 Rivest Cipher (RC4)

2.7.3 Salsa20

2.8 Block Ciphers

Block ciphers convert data in plaintext into ciphertext in fixed-size blocks. The block size generally depends on the encryption scheme and is usually in octaves (64-bit or 128-bit blocks). If the plaintext length is not a multiple of 8, the encryption scheme uses **padding** to ensure complete blocks. For instance, to perform 128-bit encryption on a 150-bit plaintext, the encryption scheme provides two blocks, 1 with 128 bits and one with the 22 bits left. 106 Redundant bits are added to the last block to make the entire block equal to the encryption scheme's ciphertext block size.

While Block ciphers use symmetric keys and algorithms to perform data encryption and decryption, they also require an **initialization vector (IV)** to function. An initialization vector is a pseudorandom or random sequence of characters used to encrypt the first block of characters in the plaintext block. The resultant ciphertext for the first block of characters acts as the initialization vector for the subsequent blocks. Therefore, the symmetric cipher produces a unique ciphertext block for each iteration while the IV is transmitted along with the symmetric key and does not require encryption.

Block encryption algorithms offer **high diffusion**; that is, if a single plaintext block were subjected to multiple encryption iterations, it resulted in a unique ciphertext block for each iteration. This makes the encryption scheme relatively tamper-proof since it is difficult for malicious actors to insert symbols into a data block without detection. On the other hand, block ciphers have a **high error propagation rate** since a bit of change in the original plaintext results in entirely different ciphertext blocks.

2.9 Block Ciphers in practice

2.9.1 Data Encryption Standard (DES)

DES is a symmetric block cipher using 64 bit blocks and 56 bits key.
The choice of using 56 bits for keys in DES was a compromise between security

and practicality. While a longer key would provide stronger security, the designers of DES also needed to consider the limitations of computing technology at the time. The inclusion of 8 parity bits reduced the effective key size to 56 bits, which some experts even then considered inadequate for robust security. However, the decision to use a 56-bit key was likely influenced by a balance between security requirements and the feasibility of implementing and processing longer keys with the available hardware during that era.

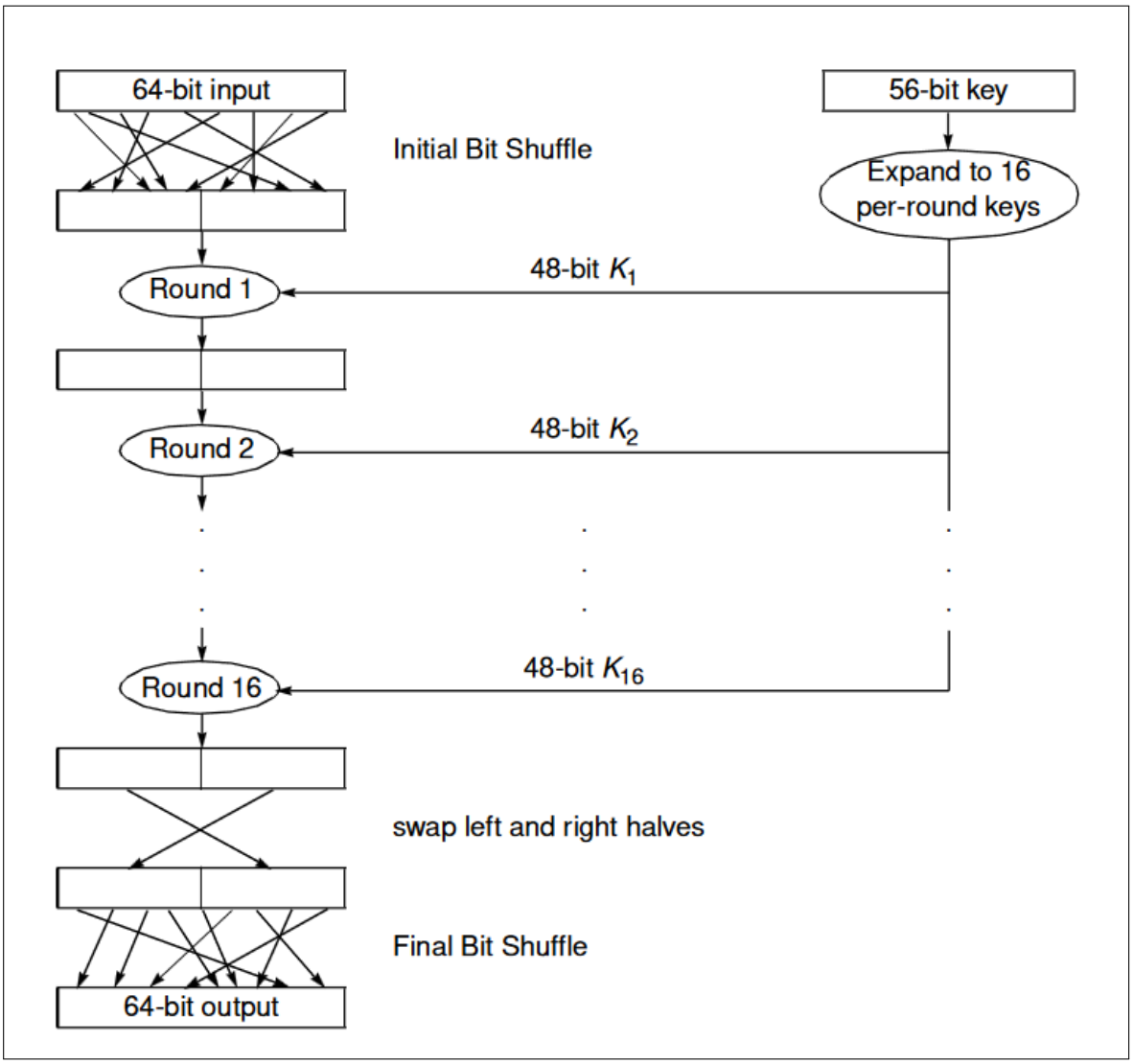

Figure 2: Basic Structure of DE

2.9.2 DES Overview

First, the 64-bit input is subjected to an initial bit-shuffle. Then, 56-bit key is expanded into sixteen 48-bit per-round keys by taking a 48-bit subset of the 56-bit

input key for each of the keys. Each round takes a 64-bit input with a 48-bit key and produces a 64-bit cipher block. After the sixteenth round, the output has its halves swapped and then is subjected to another bit-shuffle which happens to be the inverse of the initial bit-shuffle. This swapping after the final round does not add any cryptographic strength but has a side benefit. As noted in Feistel Ciphers, Swapping the output make the encryption and description identical except for the key schedule

2.9.3 DES Complementary Notes

The Mangler Function

Undesirable Symmetries

DES Variants

Add DES Mangler Function

Add DES Undesirab

Add DES Variantss

2.9.4 Advanced Encryption Standard (AES)

AES is a symmetric block cipher using 128 bit blocks and 128, 192 or 256 bits key with the resulting variants called AES-128, AES-192, and AES-256.

AES is similar to DES in that there is a key expansion algorithm which takes the key as an input and expands it into a bunch of round keys, and the algorithm executes a series of rounds that mangle a plaintext block into a ciphertext block. (Figure 3). With DES each round it takes a 64-bit input with a 48-bit key and outputs 64-bit that (along with the next round key) is fed to the next round. With AES, each round takes a 128-bit input and a 128-bit key and produces a 128-bit output which is the input to the next round. Unlike DES, AES is not Feistel cipher. (The Feistel cipher structure is characterized by the division of the plaintext into two halves and the repeated application of a round function that involves both halves. Read more here: 2.4)

In accordance to do as many rounds as needed to make the exhaustive search cheaper than any other form of cryptanalysis, AES does more round with bigger keys. AES-128 has 10 rounds, AES-196 has 12 rounds and AES-256 has 14 rounds. If AES had a 64-bit version, it would have eight rounds, which would be comparable to the sixteen rounds in DES.

Another diffference between DES and AES is that DES operates on bits where AES

operates on octets (bytes)

In AES, the key (128, 192, or 256 bits) is expanded into a series of 128-bit round keys, where there is one more round key than there are rounds. AES encryption or decryption consists of $\oplus$ing a round key into the intermediate value at the beginning of the process, at the end of the process, and between each pair of rounds.

In terms of Substitution and Permutation and cryptographic transformations, DES relies on predefined tables for S-boxes and P-boxes, which may appear arbitrary without understanding their rationale. In contrast, AES provides simpler mathematical formulas for substitutions and permutations, openly stating the reasons behind their design choices.

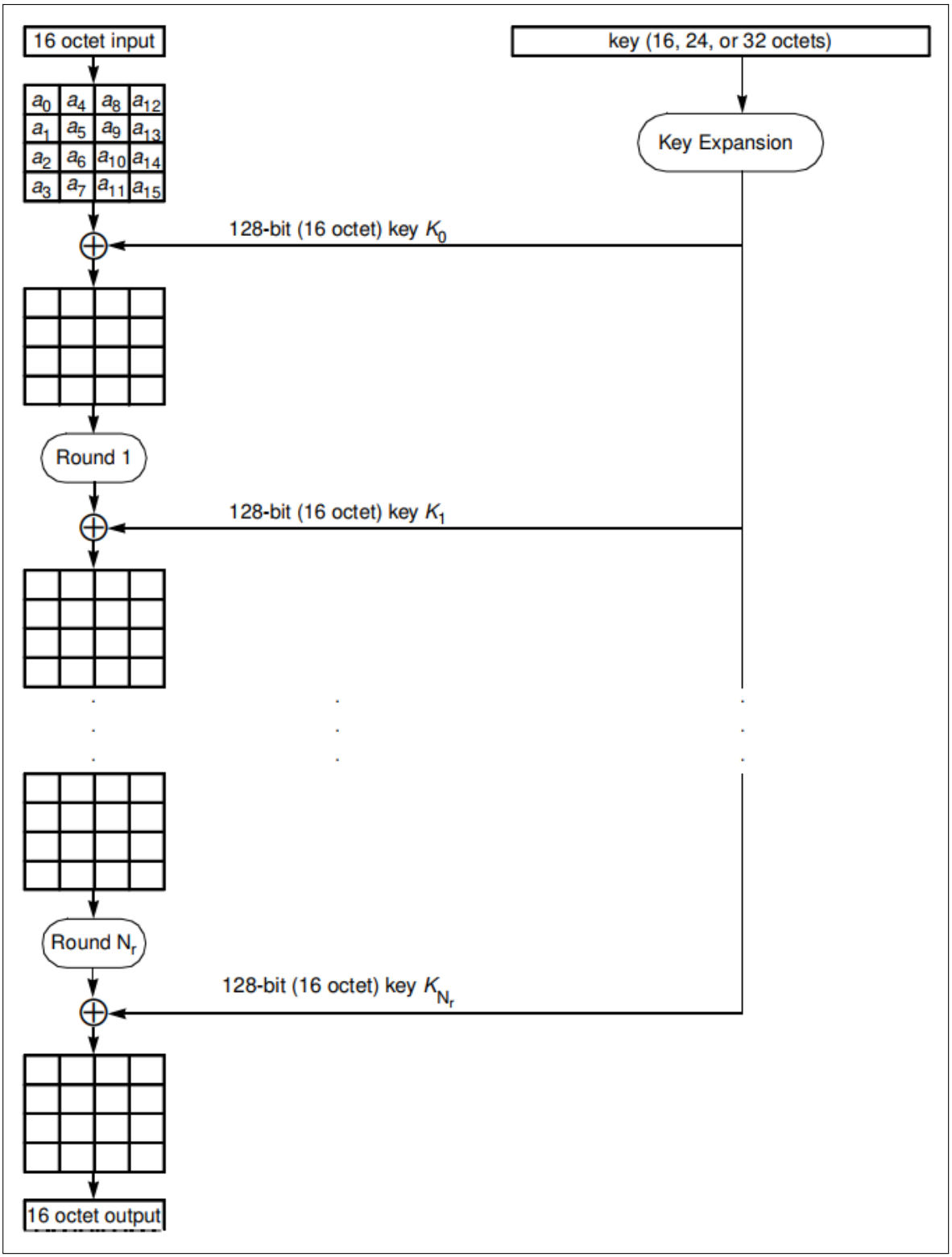

Figure 3: Basic Structure of AE

2.9.5 AES Complementary Notes

(AES's S-box)

Key expansion

Add AES Box

Add AES Key expansion

2.10 Block Ciphers Modes of Operations

Block ciphers operate on fixed-length blocks, typically 64 or 128 bits. The reason for using fixed-length blocks is twofold: first, messages can have varying lengths, and second, encrypting the same plaintext with the same key should always produce the same output.

To address the need for encrypting messages of **arbitrary** lengths, several modes of operation have been invented. These modes allow block ciphers to provide confidentiality for messages of any length. These modes define how the block cipher is applied to multiple blocks and how they are combined to encrypt or decrypt the entire message.

2.10.1 Electronic Cook Book (ECB)

a full 128 bits), and encrypt each block with the secret key (Figure 4). The other side receives the encrypted blocks and decrypts each block in turn to recover the original message (Figure 5).

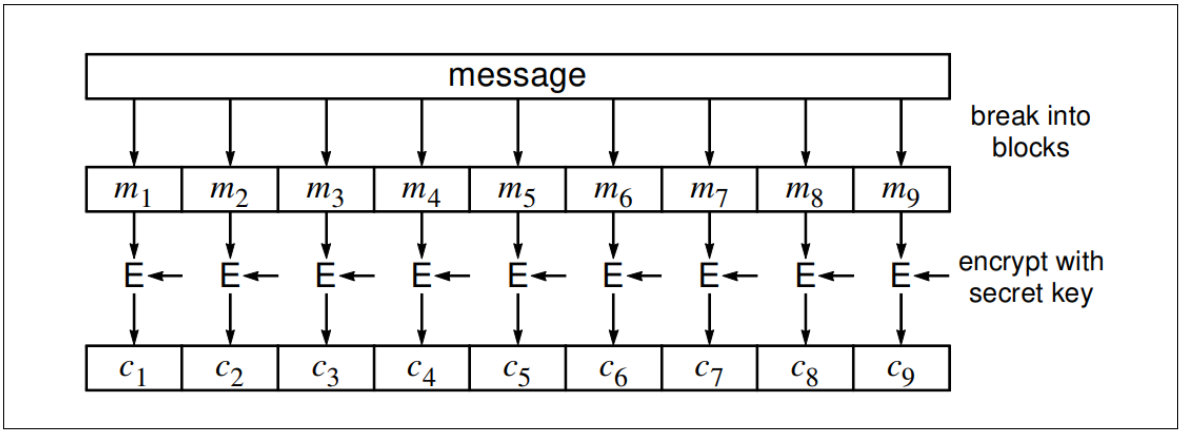

Figure 4: Electronic Code Book Encryption

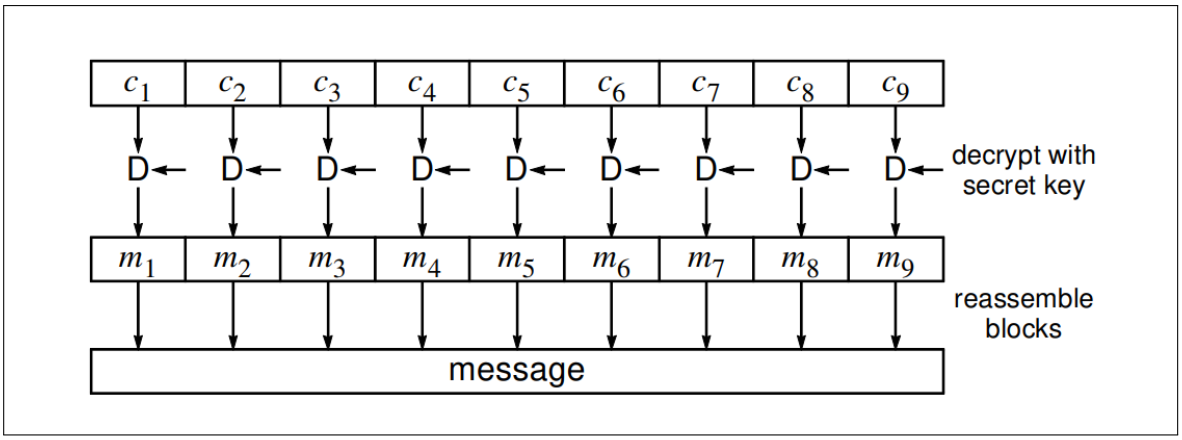

Figure 5: Electronic Code Book Decryption

ECB has two serious flaws. Patterns in the ciphertext, such as repeated blocks, leak information, and nothing prevents someone from rearranging, deleting, modifying, or duplicating blocks.

2.10.2 Cipher Block Chaining (CBC)

2.10.3 Cipher Stealing

2.10.4 Cipher Feedback (CFB)

2.10.5 Output Feedback (OFB)

2.10.6 Counter Mode (CTR)

3 Data Integrity

4 Public Key Cryptography

5 Digital Signatures

6 Cryptographically Secure Pseudo-Random Number Generators

7 Authentication

8 Secret Sharing

9 Access Control

10 Secure Protocols

11 Firewalls

12 Email Security

13 Web Technologies

14 Web Security

15 Web Tracking