===

CRYPTOGRAPHIC TOOLS

An important element in many computer security services and applications is the use of cryptographic algorithms.
We begin with symmetric encryption, which is used in the widest variety of contexts, primarily to provide confidentiality.

2.1 CONFIDENTIALITY WITH SYMMETRIC ENCRYPTION

The universal technique for providing confidentiality for transmitted or stored data is symmetric encryption. This Lecture introduces the basic concept of symmetric encryption. This is followed by an overview of the two most important symmetric encryption algorithms: the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES), which are block encryption algorithms. Finally, this section introduces the concept of symmetric stream encryption algorithms.

Symmetric Encryption

Symmetric encryption, also referred to as conventional encryption or single-key encryption, was the only type of encryption in use prior to the introduction of public-key encryption in the late 1970s.

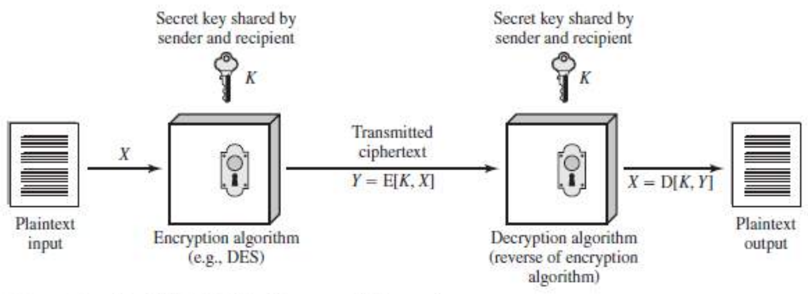

Secret key shared by sender and recipient
K
Secret key shared by sender and recipient
K
X
Transmitted ciphertext
Y = E[K, X]
X = D[K, Y]
Plaintext input
Encryption algorithm (e.g., DES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Figure 2.1 **Simplified Model of Symmetric Encryption**

A symmetric encryption scheme has five ingredients (Figure 2.1):

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

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

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

· **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.

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

There are two requirements for secure use of symmetric encryption:

1. We need a strong encryption algorithm. At a minimum, we would like the algorithm to be such that an adversary who knows the algorithm and has access to one or more ciphertexts would be unable to decipher the ciphertext or figure out the key. This requirement is usually stated in a stronger form:

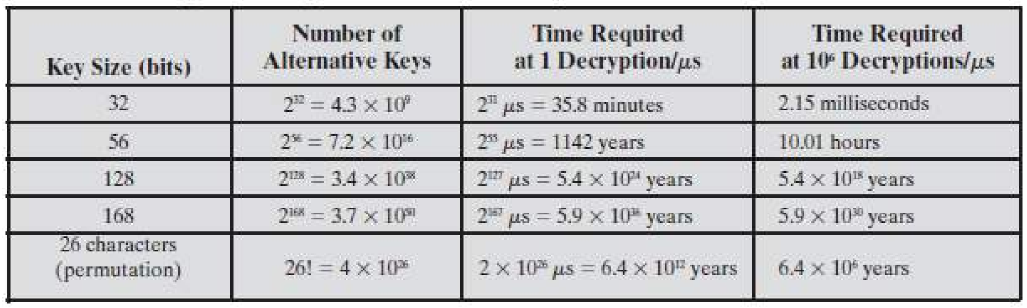
The adversary should be unable to decrypt ciphertext or discover the key even if he or she is in possession حوزة of a number of ciphertexts together with the plaintext that produced each ciphertext.

2. Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure. If someone can discover the key and knows the algorithm, all communication using this key is readable.

There are two general approaches to attacking a symmetric encryption scheme.

The first attack is known as **cryptanalysis**. Cryptanalytic attacks rely on the nature of the algorithm plus some knowledge of the general characteristics of the plaintext or even some sample plaintext-ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to conclude a specific plaintext or to conclude the key being used. If the attack succeeds in deducing the key, the effect is catastrophic: All future and past messages encrypted with that key are compromised.

Table 2.1 Average Time Required for Exhaustive Key Search



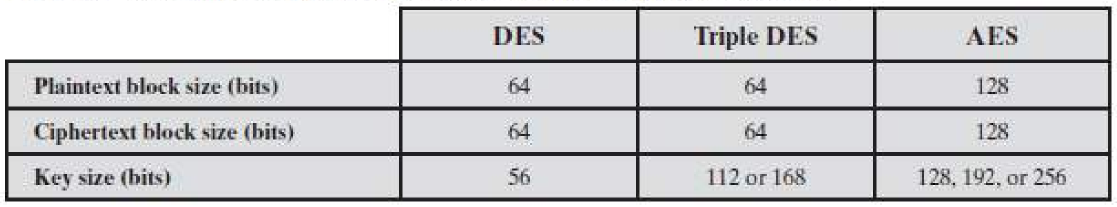
Key Size (bits)	Number of Alternative Keys	Time Required at 1 Decryption/μs	Time Required at 10^6 Decryptions/μs
32	$2^{32} = 4.3 \times 10^9$	2^{31} μs = 35.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	2^{55} μs = 1142 years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} μs $= 5.4 \times 10^{24}$ years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	2^{167} μs $= 5.9 \times 10^{36}$ years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	2×10^{26} μs $= 6.4 \times 10^{12}$ years	6.4×10^6 years

The second method, known as the **brute-force attack** , is to try every possible key on a piece of ciphertext until an intelligible واضح translation into plaintext is obtained. Table 2.1 shows how much time is involved for various key sizes. The table shows results for each key size, assuming that it takes 1 μ s to perform a single decryption, a reasonable order of magnitude for today' s computers. The final column of the table considers the results for a system that can process 1 million keys per microsecond. As one can see, at this performance level, a 56-bit key can no longer be considered computationally secure.

Symmetric Block Encryption Algorithms

The most commonly used symmetric encryption algorithms are block ciphers. A block cipher processes the plaintext input in fixed-size blocks and produces a block of ciphertext of equal size for each plaintext block. The algorithm processes longer plaintext amounts as a series of fixed-size blocks. The most important symmetric algorithms, all of which are block ciphers, are the Data Encryption Standard (DES), triple DES, and the Advanced Encryption Standard (AES); see Table 2.2.

Table 2.2 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

DATA ENCRYPTION STANDARD The most widely used encryption scheme is based on the Data Encryption Standard (DES) adopted in 1977 by the National Bureau of Standards, now the National Institute of Standards and Technology (NIST), as Federal Information Processing Standard 46 (FIPS PUB 46). The algorithm itself is referred to as the Data Encryption Algorithm (DEA). DES takes a plaintext block of 64 bits and a key of 56 bits, to produce a ciphertext block of 64 bits.

Concerns مخاوفabout the strength of DES fall into two categories: concerns about the algorithm itself and concerns about the use of a 56-bit key. The first concern refers to the possibility that cryptanalysis is possible by exploiting the characteristics of the DES algorithm. Over the years, there have been numerous attempts to find and exploit weaknesses in the algorithm. Despite numerous approaches, no one has so far reported a

deadly weakness in DES.

A more serious concern is key length. With a key length of 56 bits, there are 256 possible keys, which is approximately 7.2 $\times 10^{16}$ keys. Thus, on the face of it, a brute force attack appears impractical. Assuming that, on average, half the key space has to be searched, a single machine performing one DES encryption per micro second would take more than a thousand years (see Table 2.1) to break the cipher.

However, the assumption of one encryption per microsecond is overly conventional.

DES finally and finally proved insecure in July 1998, when the Electronic Frontier Foundation (EFF) announced وأعلن that it had broken a DES encryption using a special-purpose "DES cracker" machine that was built for less than $250,000. The attack took less than three days. The EFF has published a detailed description of the machine, enabling others to build their own cracker [EFF98]. And, of course, hardware prices will continue to drop as speeds increase, making DES virtually worthless.

It is important to note that there is more to a key-search attack than simply running through all possible keys.

If the message is just plain text in English, then the result pops out easily, although the task of recognizing English would have to be automated. If the text message has been compressed before encryption, then recognition is more difficult.

And if the message is some more general type of data, such as a numerical file, and this has been compressed, the problem becomes even more difficult to automate.

Thus, to addition the brute-force approach, some degree of knowledge about the expected plaintext is needed, and some means of automatically distinguishing plaintext from garble غربل is also needed.

If we assume that a cracker can perform 1 million decryptions per μ s, which is the rate used in Table 2.1, then a DES code would take about 10 hours to crack.

For example, for a 128-bit key, which is common among modern algorithms, it would take over 10^{18} years to break the code using the EFF cracker. Even if we managed to speed up the cracker by a factor of it would still take over 1 million years to break the code. So a 128-bit key is guaranteed to result in an algorithm that is unbreakable by brute force.

TRIPLE DES triple DES (3DES), which involves repeating the basic DES algorithm three times, using either two or three unique keys, for a key size of 112 or 168 bits.

Triple DES (3DES) has two attractions عامل جذب: First, with its 168-bit key length, it overcomes the vulnerability to brute-force attack of DES. Second, the core encryption algorithm in 3DES is the same as in DES.

If security were the only consideration, then 3DES would be an appropriate choice for a standardized encryption algorithm for decades to come.

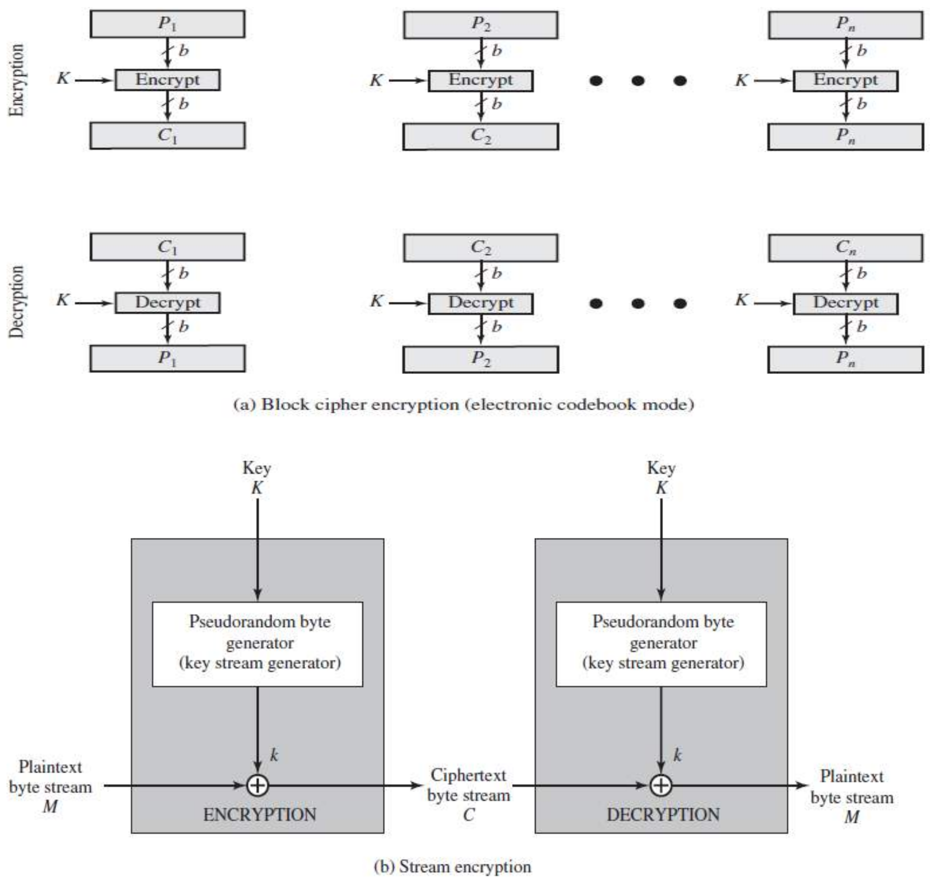
Disadvantage: The principal drawback of 3DES is that the algorithm is relatively slow in software. The original DES was designed for mid-1970s hardware implementation and does not produce efficient software code. 3DES, which requires three times as many calculations as DES, is correspondingly slower. A secondary drawback is that both DES and 3DES use a 64-bit block size. For reasons of both efficiency and security, a larger block size is desirable.

ADVANCED ENCRYPTION STANDARD Because of its drawbacks, 3DES is a replacement, NIST in 1997 issued a call for proposals for a new Advanced Encryption Standard (AES), which should have a security strength equal to or better than 3DES and significantly improved efficiency. In addition to these general requirements, NIST specified that AES must be a symmetric block cipher with a block length of 128 bits and support for key lengths of 128, 192, and 256 bits.

* Evaluation criteria included security, computational efficiency, memory requirements, hardware and software suitability, and flexibility.

PRACTICAL SECURITY ISSUES symmetric encryption is applied to a unit of data. E-mail messages, network packets, database records, and other plaintext sources must be broken up into a series of fixed-length block for encryption by a symmetric block cipher. The simplest approach to multiple-block encryption is known as electronic codebook (ECB) mode, plaintext is handled b bits at a time and each block of plaintext is encrypted using the same key. Typically b = 64 or b =128. Figure 2.3a shows the ECB mode. A plain text of length nb is divided into n b-bit blocks (P1, P2,…,Pn).

Each block is encrypted using the same algorithm and the same encryption key, to produce a sequence of n b -bit blocks of ciphertext (C1, C 2,..,Cn). For lengthy messages, the ECB mode may not be secure.


Encryption
Decryption
P_1
P_2
P_n
K
Encrypt
Decrypt
b
C_1
C_2
C_n
Key
K
Pseudorandom byte generator (key stream generator)
k
Plaintext byte stream M
ENCRYPTION
Ciphertext byte stream C
DECRYPTION
Plaintext byte stream M

(a) Block cipher encryption (electronic codebook mode)

(b) Stream encryption

Figure 2.3 **Types of Symmetric Encryption**

To increase the security of symmetric block encryption for large sequences of data, a number of alternative techniques have been developed, called **modes of operation** . These modes overcome the weaknesses of ECB; each mode has its own particular advantages.

Stream Ciphers

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. Although block ciphers are far more common, there are certain applications in which a stream cipher is more appropriate.

Stream cipher encrypts plaintext one byte at a time, although a stream cipher may be designed to operate on one bit at a time or on units larger than a byte at a time. Figure 2.3b is a representative diagram of stream cipher structure.

In this structure a key is input to a pseudorandom bit generator that produces a stream of 8-bit numbers that are apparently random. A pseudorandom stream is unpredictable without knowledge of the input key and which has an apparently random character .The output of the generator, called a **keystream** , is combined one byte at a time with the plaintext stream using the bitwise exclusive-OR (XOR) operation.

Note:A pseudorandom number generator (PRNG), also known as a deterministic random bit generator (DRBG),[1] is an algorithm for generating a sequence of numbers whose properties approximate the properties of sequences of random numbers. The PRNG-generated sequence is not truly random, because it is completely determined by an initial value, called the PRNG's seed (which may include truly random values). Although sequences that are closer to truly random can be generated using hardware random number generators, pseudorandom number generators are important in practice for their speed in number generation and their reproducibility

With a properly designed pseudorandom number generator, a stream cipher can be as secure as block cipher of comparable key length. The primary advantage of a stream cipher is that stream ciphers are almost always faster and use far less code than do block ciphers. The advantage of a block cipher is that you can reuse keys. 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.

==PART-2

Message Authentication

• protects against active attacks

• verifies received message is authentic

– contents unaltered

– from authentic source

– timely and in correct sequence

• can use conventional encryption

– only sender & receiver have key needed.

Encryption protects against passive attack (eavesdropping). A different requirement

is to protect against active attack (falsification of data and transactions). Protection

against such attacks is known as message or data authentication.

A message, file, document, or other collection of data is said to be authentic when it is genuineحقيقي and came from its assumed source.

*Message or data authentication :is a procedure that allows communicating parties to verify that received or stored messages are authentic.

The two important aspects are to verify that the contents of the message have not been altered and that the source is authentic.

We may also wish to verify a message's timeliness توقيت(it has not been artificially delayed and replayed) and sequence relative to other messages flowing between two parties. All of these concerns come under the category of data integrity.

Authentication Using Symmetric Encryption

It would seem possible to perform authentication simply by the use of symmetric encryption.

*If we assume that only the sender and receiver share a key (which is as it should be), then only the genuine sender would be able to encrypt a message successfully for the other participant, provided the receiver can recognize a valid message**.**

*Furthermore, if the message includes an error-detection code and a sequence number, the receiver is assured that no alterations have been made and that sequencing is proper. If the message also includes a timestamp, the receiver is assured that the message has not been delayed beyond that normally expected for network transit.

In fact, symmetric encryption alone is not a suitable tool for data authentication.

Message Authentication without Message Encryption

we examine several approaches to message authentication that do not rely on message encryption. In all of these approaches, an authentication tag is generated and appended to إلحاق each message for transmission.

Because this approaches do not encrypt the message, message confidentiality is not provided.

*it is possible to combine authentication and confidentiality in a single algorithm by encrypting a message plus its authentication tag.

*situations in which message authentication without confidentiality is preferable:

*الحالات التي تستوجب المصادقة فقط

1. There are a number of applications in which the same message is broadcast to a number of destinations. the message must be broadcast in plaintext with an associated message authentication tag.

2. Another possible scenario is an exchange in which one side has a heavy load and cannot afford the time to decrypt all incoming messages. Authentication is carried out on a selective basis, with messages being chosen at random for checking.

MESSAGE AUTHENTICATION CODE MAC : One authentication technique involves the use of a secret key to generate a small block of data, known as a message authentic-cation code, that is appended to the message.

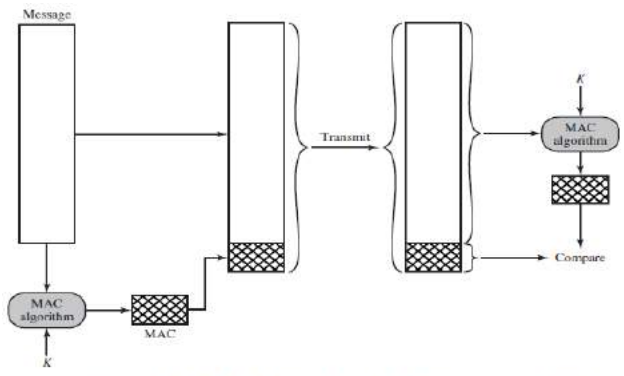


Figure 2.4 Message Authentication Using a Message Authentication Code (MAC) The MAC is a function of an input message and a secret key

authentication code, that is appended to the message. This technique assumes that two communicating parties, say A and B, share a common secret key K_{AB}. When A has a message to send to B, it calculates the message authentication code as a complex function of the message and the key: MAC_M _ $\text{F}(K_{AB}, M)$.

*1-The message plus code are transmitted to the intended recipient.

2-The recipient performs the same calculation on the received message, using the same secret key, to generate a new message authentication code.

3-The received code is compared to the calculated code.

(Figure 2.4). If we assume that only the receiver and the sender know the identity of the secret key, and if the received code matches the calculated code, then:

1. The receiver is assured that the message has not been altered. If an attacker

alters the message but does not alter the code, then the receiver' s calculation of the code will differ from the received code. Because the attacker is assumed not to know the secret key, the attacker cannot alter the code to correspond to the alterations in the message.

2. The receiver is assured that the message is from the alleged sender. Because no one else knows the secret key, no one else could prepare a message with a proper code.

3. If the message includes a sequence number (such as is used with X.25, HDLC, and TCP), then the receiver can be assured of the proper sequence, because an attacker cannot successfully alter the sequence number.

One difference is that the authentication algorithm need not be reversible, as it must for decryption.

ONE-WAY HASH FUNCTION An alternative to the message authentication code is the

one-way hash function. As with the message authentication code, a hash function accepts a variable-size message *M* as input and produces a fixed-size message digest H(*M*) as output (Figure 2.5).

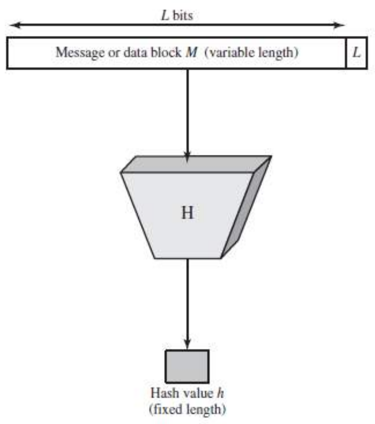


Figure 2.5 **Block Diagram of Secure Hash Function; $h = H(M)$**

The length field is a security measure to increase the difficulty for an attacker to produce an alternative message with the same hash value.

Unlike the MAC, a hash function does not also take a secret key as input.