Information Security Fall 2022

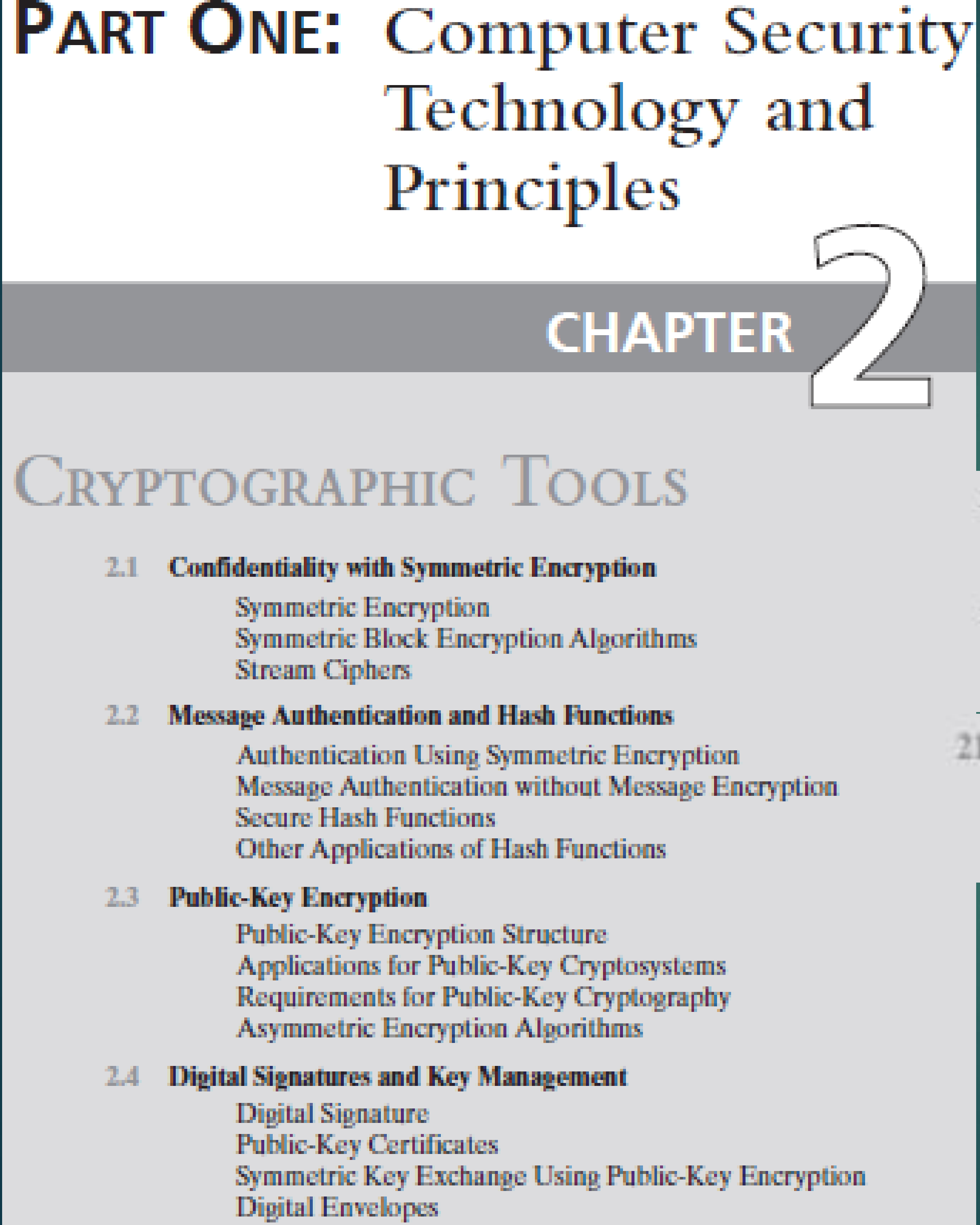
WEEK # 4

LECTURE # 10, 11 AND 12

DR. AQSA ASLAM

Topics from text book:

- Chapter # 2
- Chapter # 20
- Chapter # 21



PART ONE: Computer Security Technology and Principles

CHAPTER 2

CRYPTOGRAPHIC TOOLS

2.1 Confidentiality with Symmetric Encryption

Symmetric Encryption
Symmetric Block Encryption Algorithms
Stream Ciphers

2.2 Message Authentication and Hash Functions

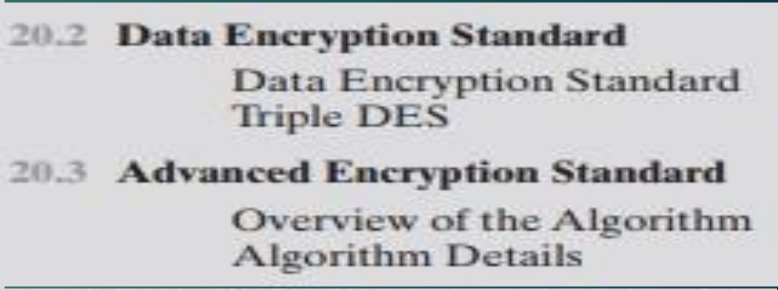
Authentication Using Symmetric Encryption
Message Authentication without Message Encryption
Secure Hash Functions
Other Applications of Hash Functions

2.3 Public-Key Encryption

Public-Key Encryption Structure
Applications for Public-Key Cryptosystems
Requirements for Public-Key Cryptography
Asymmetric Encryption Algorithms

2.4 Digital Signatures and Key Management

Digital Signature
Public-Key Certificates
Symmetric Key Exchange Using Public-Key Encryption
Digital Envelopes

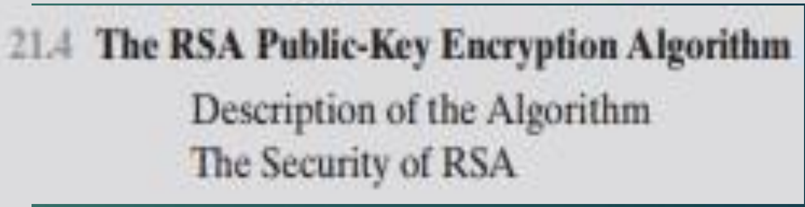


20.2 Data Encryption Standard

Data Encryption Standard
Triple DES

20.3 Advanced Encryption Standard

Overview of the Algorithm
Algorithm Details



21.4 The RSA Public-Key Encryption Algorithm

Description of the Algorithm
The Security of RSA

Message Authentication

Protects against active attacks

Verifies received message is authentic

- Contents have not been altered
- From authentic source
- Timely and in correct sequence

Can use conventional encryption

- Only sender and receiver share a key

Types of authentication

- Message Encryption
- Message authentication code (MAC)
- Hash function

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**

Message Authentication Code

- One authentication technique involves the use of a secret key to generate a small block of data, known as a ***message authentication code (MAC)*** that is appended to the message.
- 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:

    $$MAC_M \;\; F(K_{AB}, M).$$

  - *The message* plus code are transmitted to the intended recipient.
  - The recipient performs the same calculation on the received message, using the same secret key, to generate a new message authentication code.
  - The received code is compared to the calculated code and if the received code matches the calculated code, then:
    - The receiver is assured that the message has not been altered
    - The receiver is assured that the message is from the alleged sender.

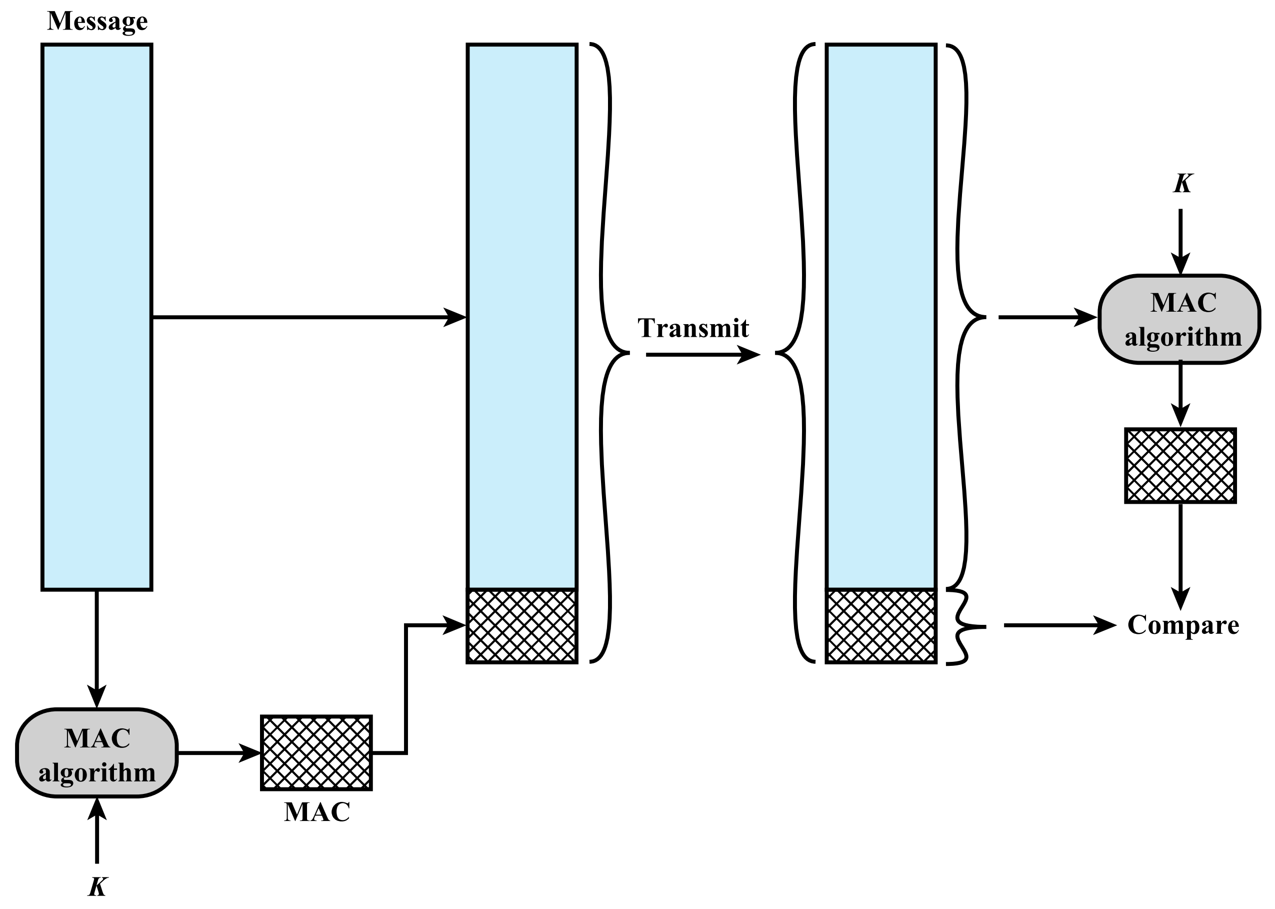


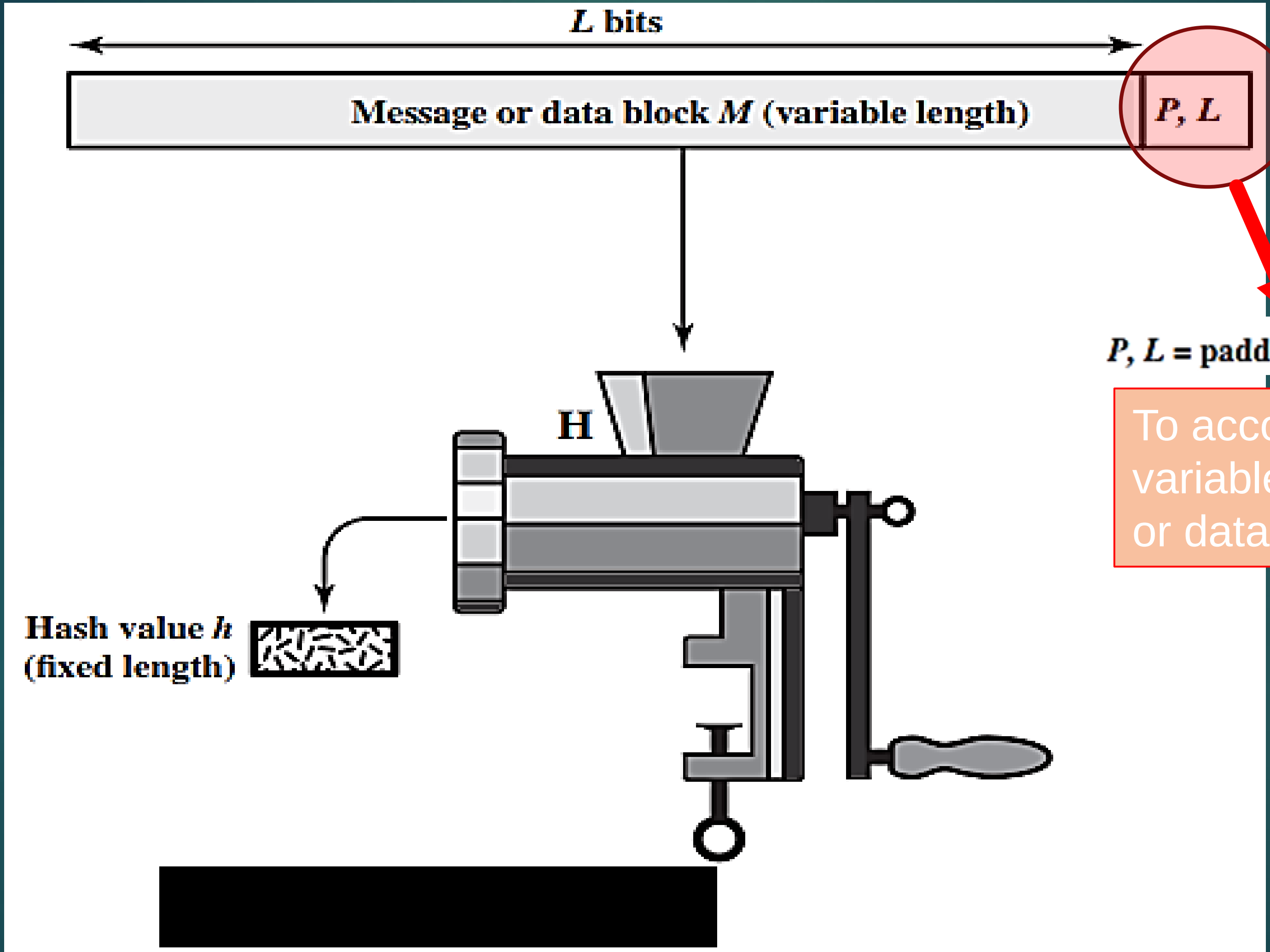
Figure 2.3 Message Authentication Using a Message Authentication Code (MAC).

DES or AES is used to generate an encrypted version of the message, and some of the bits of ciphertext are used as the code. Longer codes are used to provide sufficient collision resistance. One difference is that the authentication algorithm need not be reversible, as it must for decryption.

One-Hash Function

- An alternative to the message authentication code (MAC) 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*
- Unlike the MAC, ***a hash function does not also take a secret key as input.***
  - Typically, the message is padded out to an integer multiple of some fixed length (e.g., 1024 bits) and the padding includes the value of the length of the original message in bits.

One-Hash Function



L bits
Message or data block *M* (variable length)
P, *L*
H
Hash value *h* (fixed length)

[redacted]

P, *L* = padding plus length field

To accommodate variable size messages or data blocks

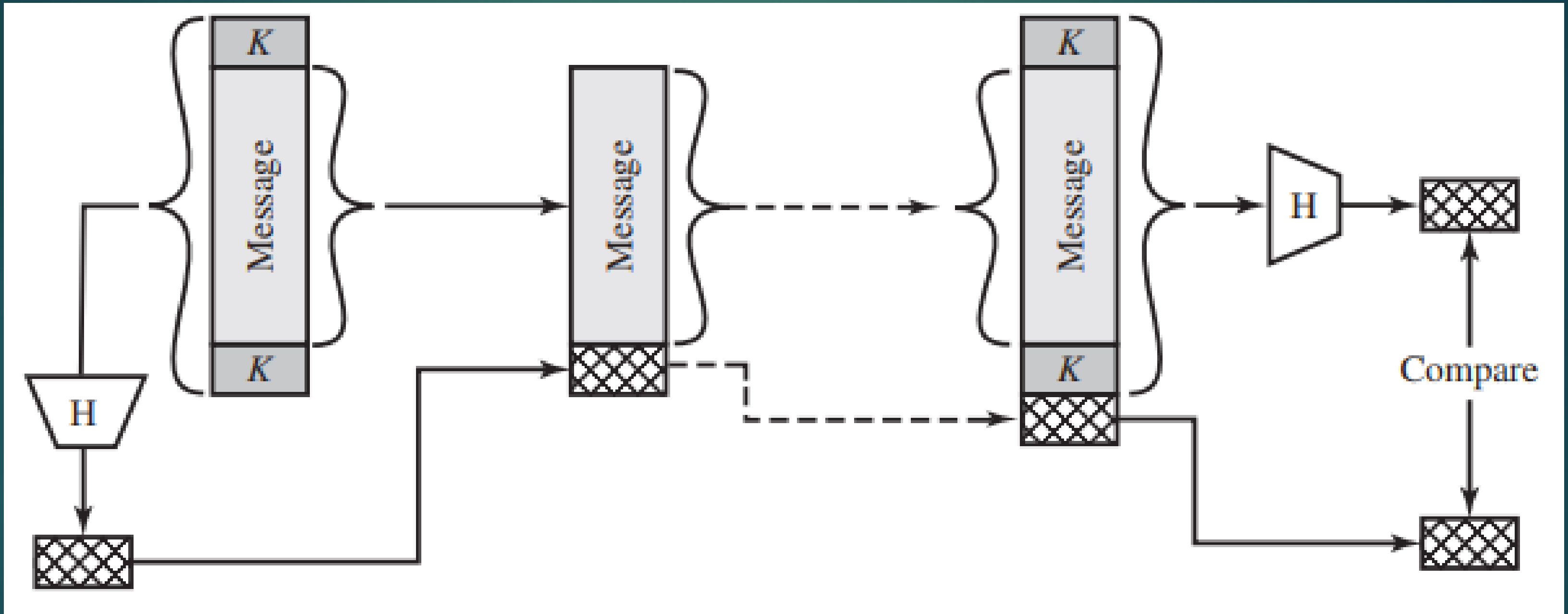
Figure 2.4 Cryptographic Hash Function; $h = H(M)$

One-Way Hash Function

- This technique, known as a ***keyed hash MAC*** (see Figure 2.5c)
- Assumes that two communicating parties, say A and B, share a common secret key K.
  - This secret key is incorporated into the process of generating a hash code.
  - When A has a message to send to B, it calculates the hash function over the concatenation of the secret key and the message:

$$MD_M = H(K \mathbin{||} M \mathbin{||} K)$$

  - It then sends ***[M ii*** MD_M***]*** to B. Because B possesses K, it can re-compute $H(K \mathbin{||} M \mathbin{||} K)$ *and verify* MD_M.
    - Because the secret key itself is not sent, it should not be possible for an attacker to modify an intercepted message.
    - As long as the secret key remains secret, it should not be possible for an attacker to generate a false message.



(c) Using secret value

Three ways in which the message can be authenticated using a hash function (Figure 2.5)

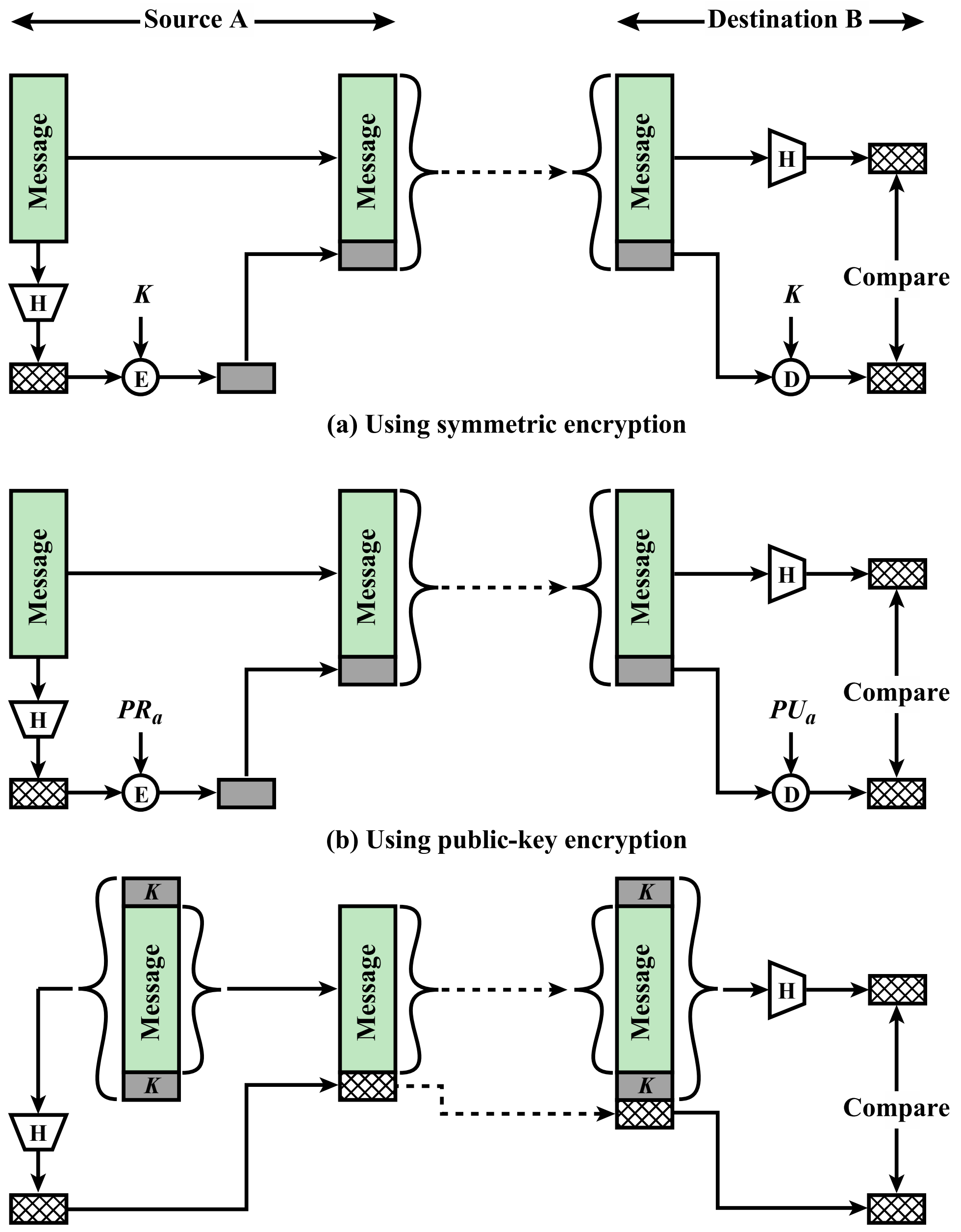
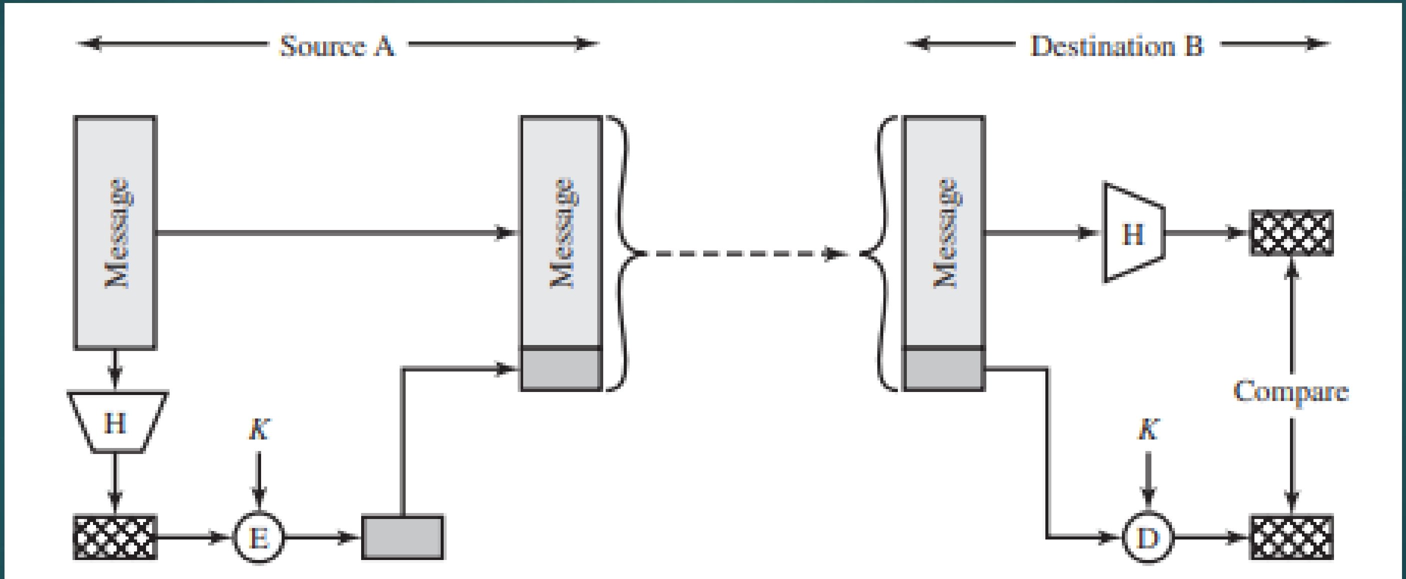


Figure 2.5 Message Authentication Using a One-Way Hash Function.

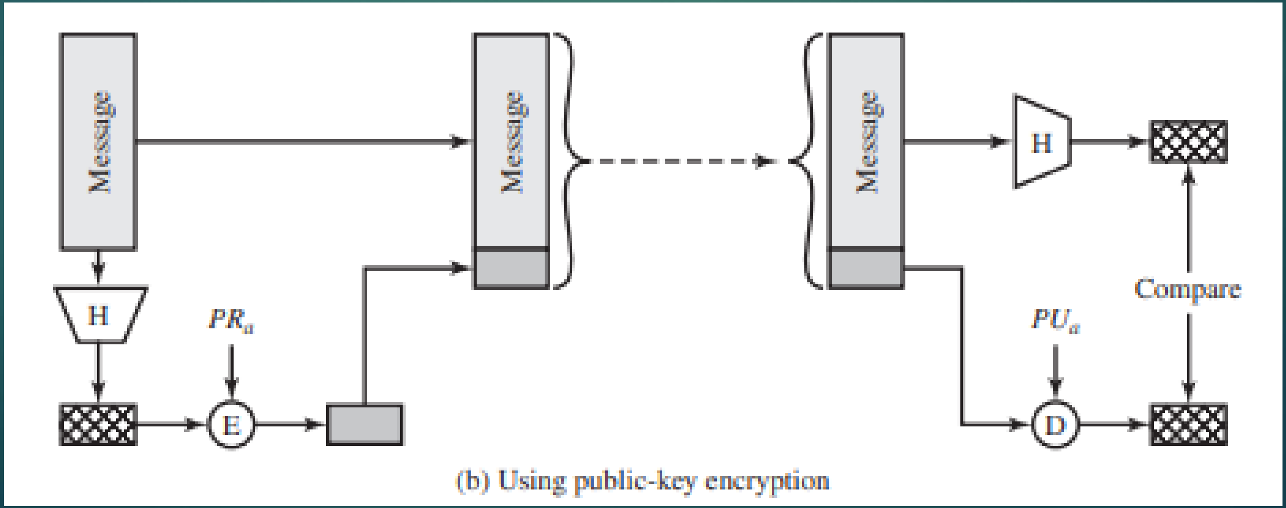
- The message digest can be encrypted using symmetric encryption (see Figure 2.5a)



(a) Using symmetric encryption

- The message digest can also be encrypted using public-key encryption (see Figure 2.5b) ; this is explained in Section 2.3.
- The public key approach has two advantages: It provides a digital signature as well as message authentication, and it does not require the distribution of keys to communicating parties.

Figure 2.5a and b approaches have an advantage over approaches that encrypt the entire message, in that less computation is required.



(b) Using public-key encryption

Reasons to avoid encryption

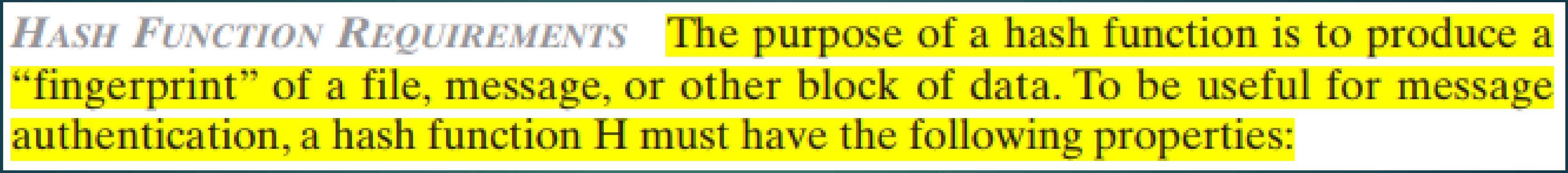
- More common approach is the use of a technique that avoids encryption altogether.
- Several reasons for this interest are pointed out in [TSUD92]:

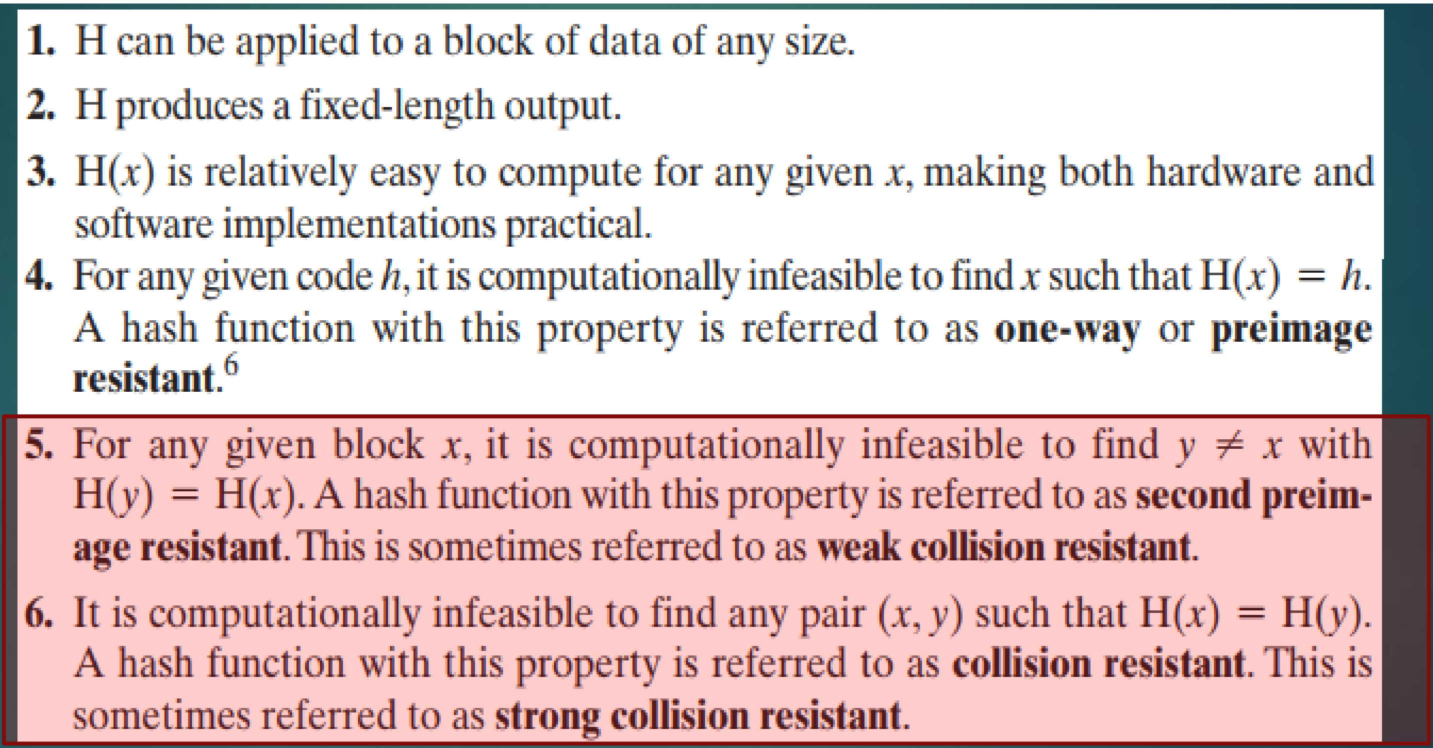
1. Encryption software is quite slow.
    - Even though the amount of data to be encrypted per message is small, there may be a steady stream of messages into and out of a system.
2. Encryption hardware costs are non-negligible.
    - Low-cost chip implementations of DES and AES are available, but the cost adds up if all nodes in a network must have this capability.
3. Encryption hardware is optimized toward large data sizes.
    - For small blocks of data, a high proportion of the time is spent in initialization/invocation overhead.
4. An encryption algorithm may be protected by a patent.

To be useful for message authentication, a hash function H must have the following properties:

- Can be applied to a block of data of any size
- Produces a fixed-length output
- H(x) is relatively easy to compute for any given x
- One-way or pre-image resistant
  - Computationally infeasible to find x such that H(x) = h
- Computationally infeasible to find y ≠ x such that H(y) = H(x)
- Collision resistant or strong collision resistance
  - Computationally infeasible to find any pair (x,y) such that H(x) = H(y)

Detail of previous slide


HASH FUNCTION REQUIREMENTS The purpose of a hash function is to produce a "fingerprint" of a file, message, or other block of data. To be useful for message authentication, a hash function H must have the following properties:


1. H can be applied to a block of data of any size.
2. H produces a fixed-length output.
3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. For any given code h, it is computationally infeasible to find x such that H(x) $= h$. A hash function with this property is referred to as **one-way** or **preimage resistant**.[6]
5. For any given block x, it is computationally infeasible to find $y \neq x$ with H(y) $=$ H(x). A hash function with this property is referred to as **second preimage resistant**. This is sometimes referred to as **weak collision resistant**.
6. It is computationally infeasible to find any pair (x, y) such that H(x) $=$ H(y). A hash function with this property is referred to as **collision resistant**. This is sometimes referred to as **strong collision resistant**.

Other Applications of Hash Functions

- **Passwords:** Chapter 3 will explain a scheme in which a hash of a password is stored by an operating system rather than the password itself. Thus, the actual password is not retrievable by a hacker who gains access to the password file. In simple terms, when a user enters a password, the hash of that password is compared to the stored hash value for verification. This application requires preimage resistance and perhaps second preimage resistance.

- **Intrusion detection:** Store the hash value for a file, H(F), for each file on a system and secure the hash values (e.g., on a write-locked drive or write-once optical disk that is kept secure). One can later determine if a file has been modified by recomputing H(F). An intruder would need to change F without changing H(F). This application requires weak second preimage resistance.