CS 3002 Information Security

Fall 2023

1. Explain key concepts of information security such as design principles, cryptography, risk management,(1)

2. Discuss legal, ethical, and professional issues in information security (6)
3. Analyze real world scenarios, model them using security measures, and apply various security and risk management tools for achieving information security and privacy (2)
4. Identify appropriate techniques to tackle and solve problems of real life in the discipline of information security (3)
5. Understand issues related to ethics in the field of information security(8)

ISO/IEC 27001: 2013

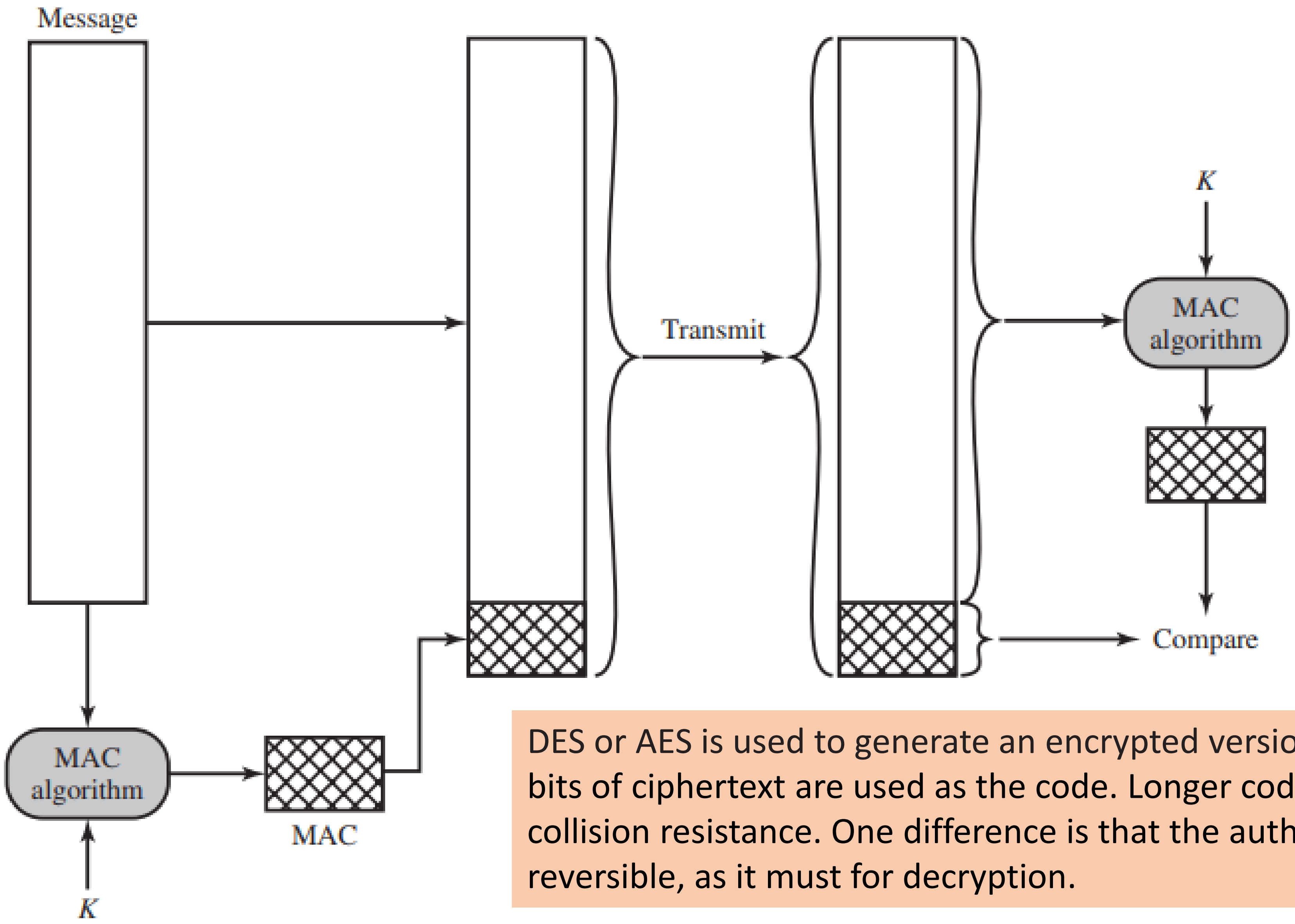
Week # 4

Dr. Nadeem Kafi Khan

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, 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: $MAC_M = F(K_{AB}, M)$.[3] 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 (see Figure 2.3).

[3]Because messages may be any size and the message authentication code is a small fixed size, there must theoretically be many messages that result in the same MAC. However, it should be infeasible in practice to find pairs of such messages with the same MAC. This is known as collision resistance.

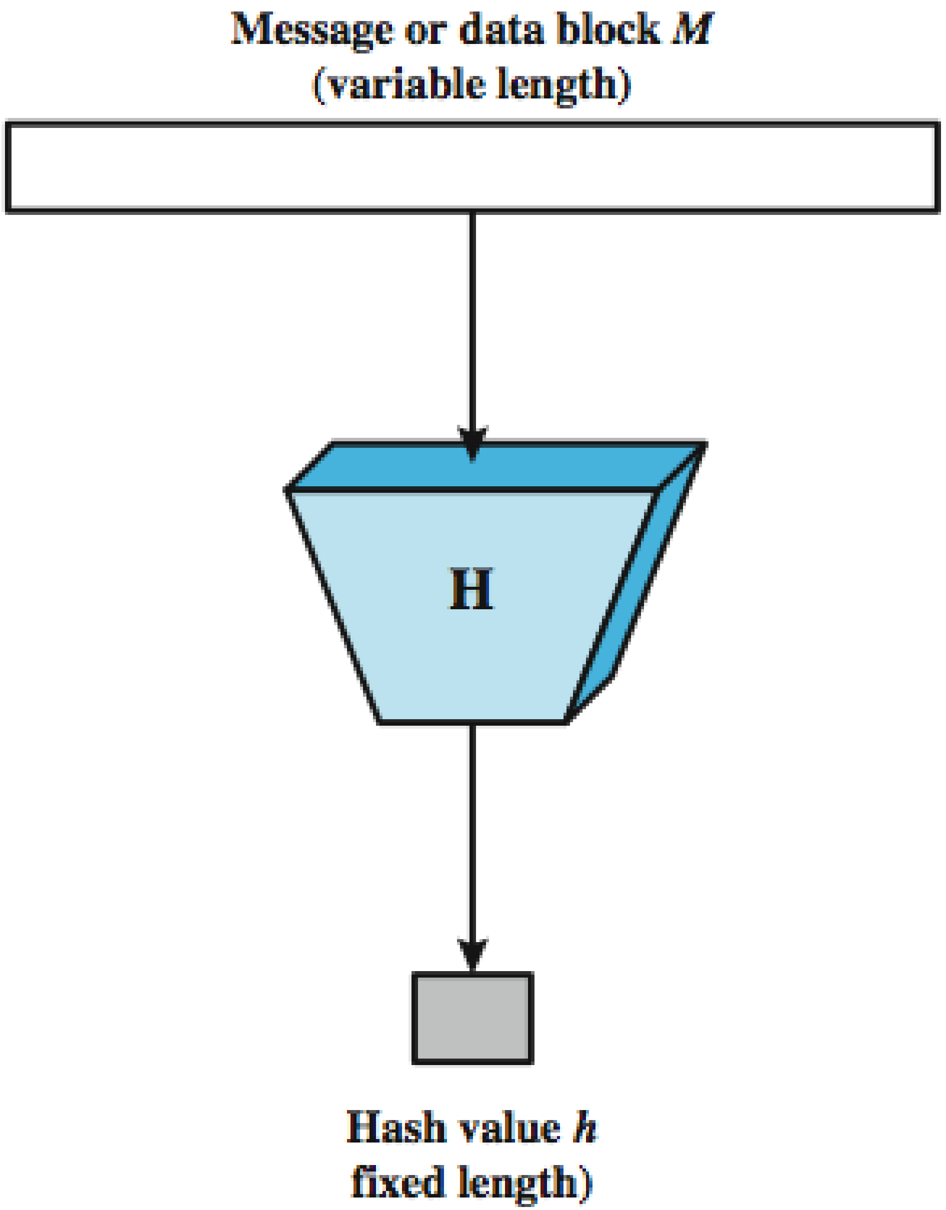


DES or AES is used to generate an encrypted version of the message, and some of the bits of ciphertext are used as the code. Longer codes are used to provide sufficient collision resistance. One difference is that the authentication algorithm need not be reversible, as it must for decryption.

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

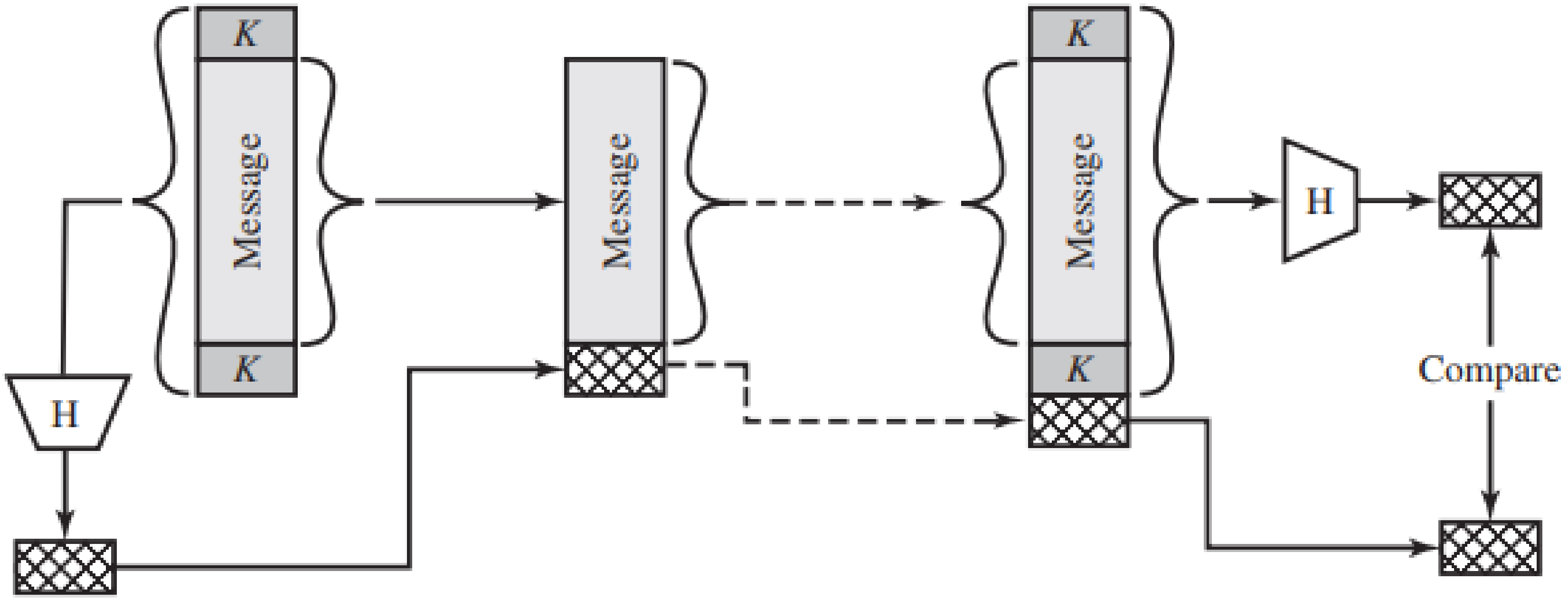
ONE-WAY HASH FUNCTION An alternative to the message authentication code is the one-way hash function. As with the message authentication code, a hash function accepts a variable-size message *M* as input and produces a fixed-size message digest H(*M*) as output (see Figure 2.4). Typically, the message 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. 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 take a secret key as input. Figure 2.5 illustrates three ways in which the message can be authenticated using a hash function.


Message or data block M
(variable length)
H
Hash value h
fixed length)

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

- 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.
- Encryption hardware costs are nonnegligible. 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.
- 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.
- An encryption algorithm may be protected by a patent.



(c) Using secret value Figure 2.5 **Message Authentication Using a One-Way Hash Function**

This technique, known as a keyed hash MAC, 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. In the approach illustrated in Figure 2.5c, when A has a message to send to B, it calculates the hash function over the concatenation of the secret key and the message: $MD_M = \mathrm{H}(K \| M \| K)$.[5] It then sends $[M \| MD_M]$ to B. Because B possesses K, it can recompute $\mathrm{H}(K \| M \| K)$ 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.

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]

Various constructions of Hash Functions

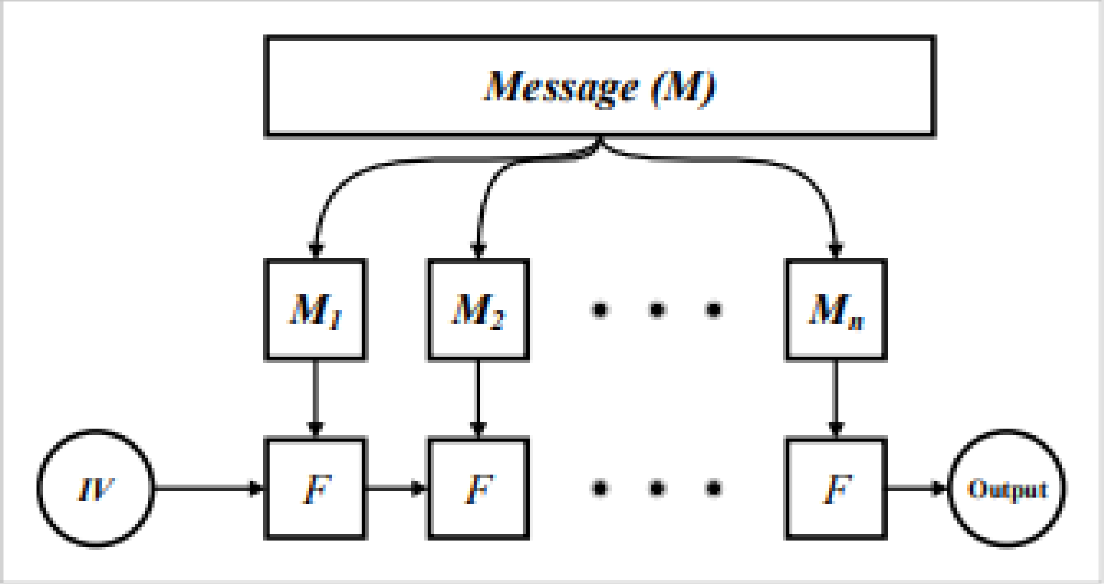


Fig. 1. The general structure of the Merkle-Damgård construction

Figure 1 shows the general structure of the Merkle-Damgard construction:

- a message (M) is divided into blocks and processed sequentially using a compression function (F).
- This structure requires **initial values (IV)** that are used to process the first block.
- The output of each block is used to process the next block.
- The final output is produced after processing the last block of the message.

In cryptography, an initialization vector (IV) or starting variable is **an input to a cryptographic primitive being used to provide the initial state**. The IV is typically required to be random or pseudorandom, but sometimes an IV only needs to be unpredictable or unique.

MD4 and MD5 formed the base for the following hash functions that come after, which have the same Merkle Damgard construction. We list examples of these functions as follows: **SHA-0, SHA-1, SHA-2 (SHA-224, SHA-256, SHA-384, and SHA-512), RIPEMD 160**.

F. Security Requirements

A hash function is considered as secure if it supports the security requirements of Pre-image, 2^{nd}-Pre-image, Collision resistance, MAC, and PRF [27]–[29].

1) **Pre-image**: Which is based on the problem of finding a pre-image M for a given hash h. Pre, which stands for *pre-image resistance*, requires that given a randomly chosen hash function f and a uniformly randomly chosen hash h it is hard to find a pre-image $M \in f^{-1}(h)$.

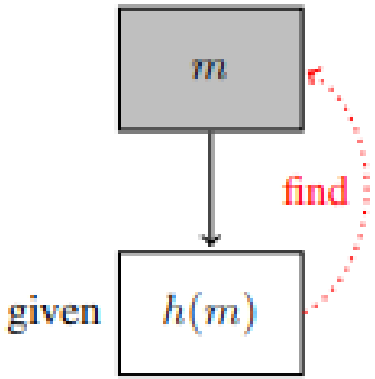


Fig. 6. Pre-image

2) **Sec-Pre-image**: Which is based on the problem of finding another message M' that hash the same hash h as a given message M. *Sec-Pre-image* requires that the function f and message M to be uniformly and randomly chosen.

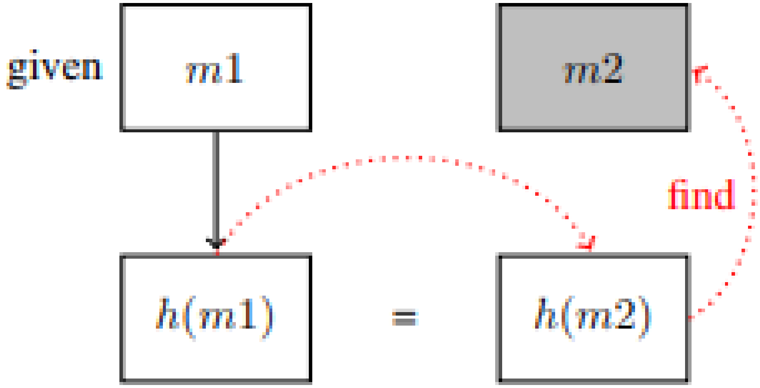


Fig. 7. Second Pre-image

3) **Collision**: Which is the property that finding two messages M and M' that have the same hash h for the same chosen f. The hash function f must be strong enough to resist any possibility of a collision. Unfortunately, this property was the first to be broken.

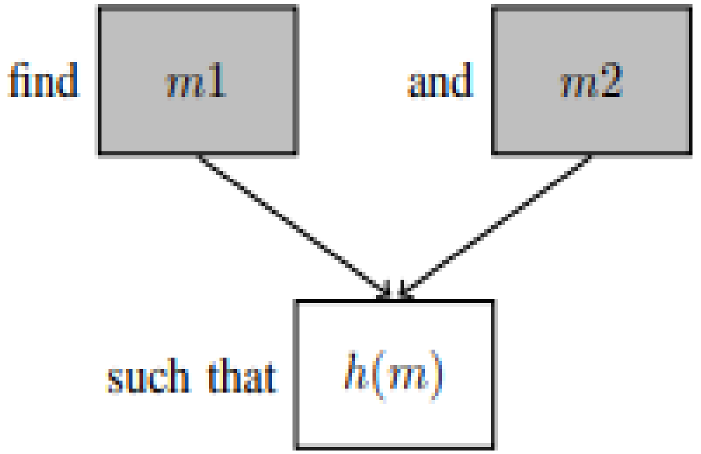


Fig. 8. Collision

4) **MAC**: Which stands for Message Authentication Code, is the property that canvasses the hash function f as a keyed hash function (f_k) where k is a randomly chosen secret key that is associated with f. The MAC property requires that the dilemma of finding a message M and the hash h with the presence of secret key k, is difficult to achieve.

5) **PRF**: Which stands for Pseudo-Random Function, is the property that the problem of determining the actual function f_k for a randomly chosen k with a randomly chosen hash h must be difficult.

A PRF is an efficient (i.e. computable in polynomial time), deterministic function that maps two distinct sets (domain and range) and looks like a truly random function.

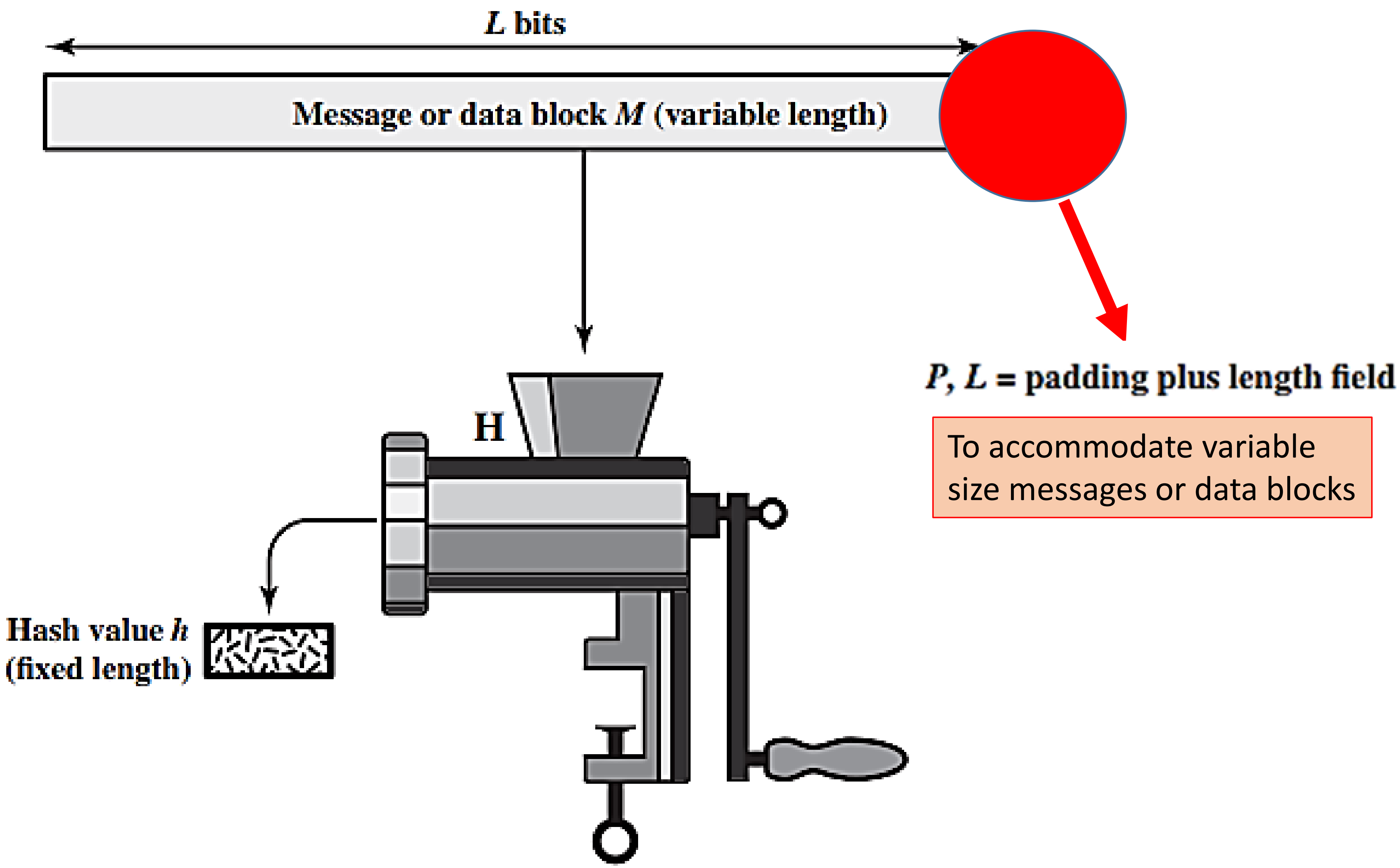
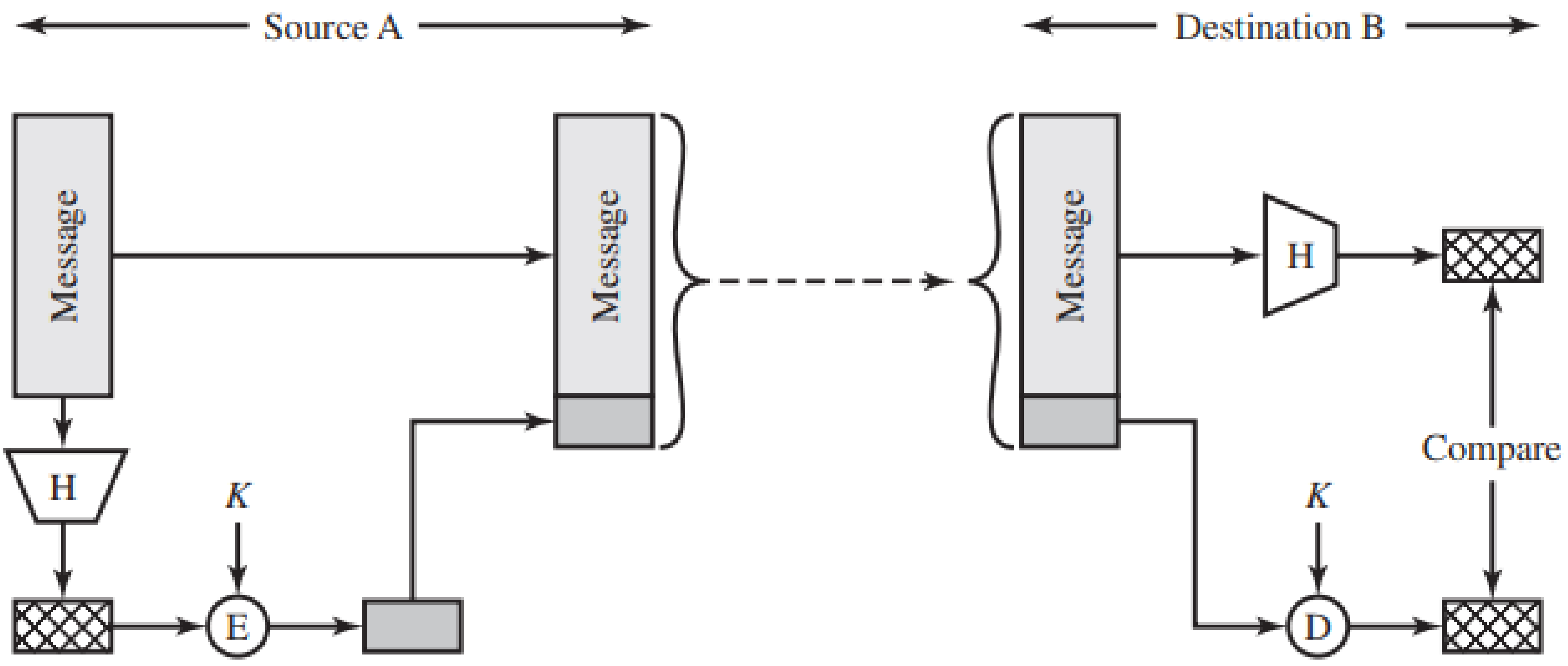


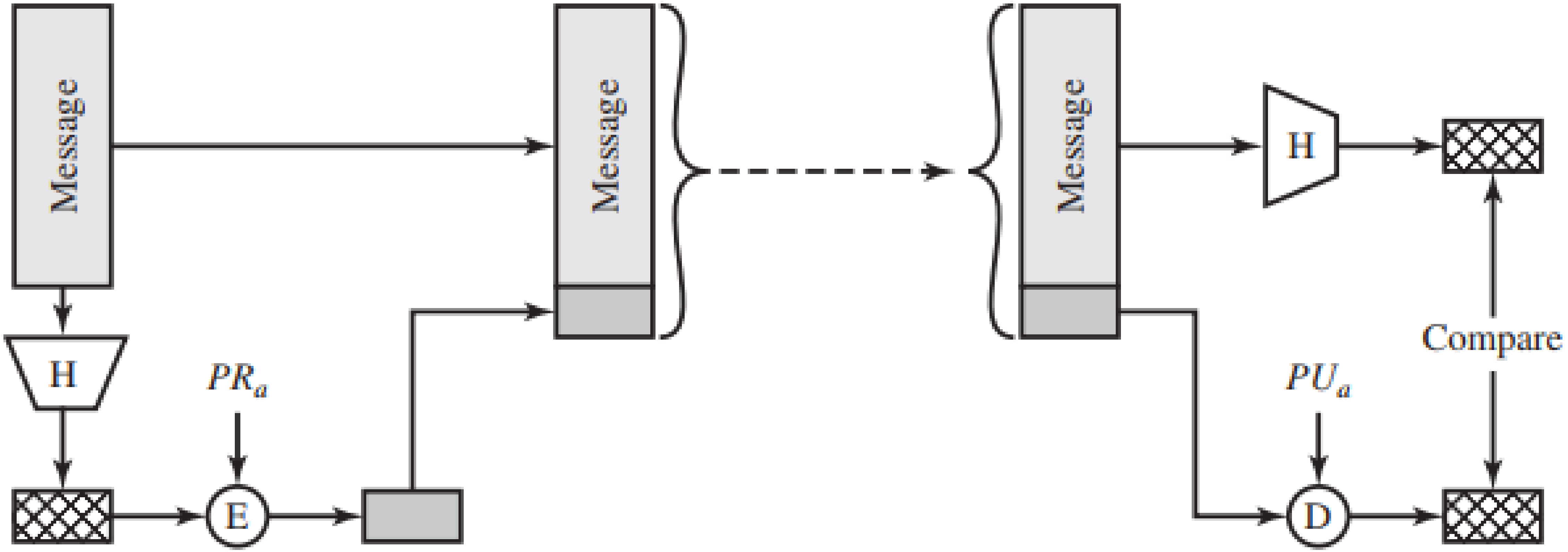
Figure 2.4 Cryptographic Hash Function; $h = H(M)$



(a) Using symmetric encryption

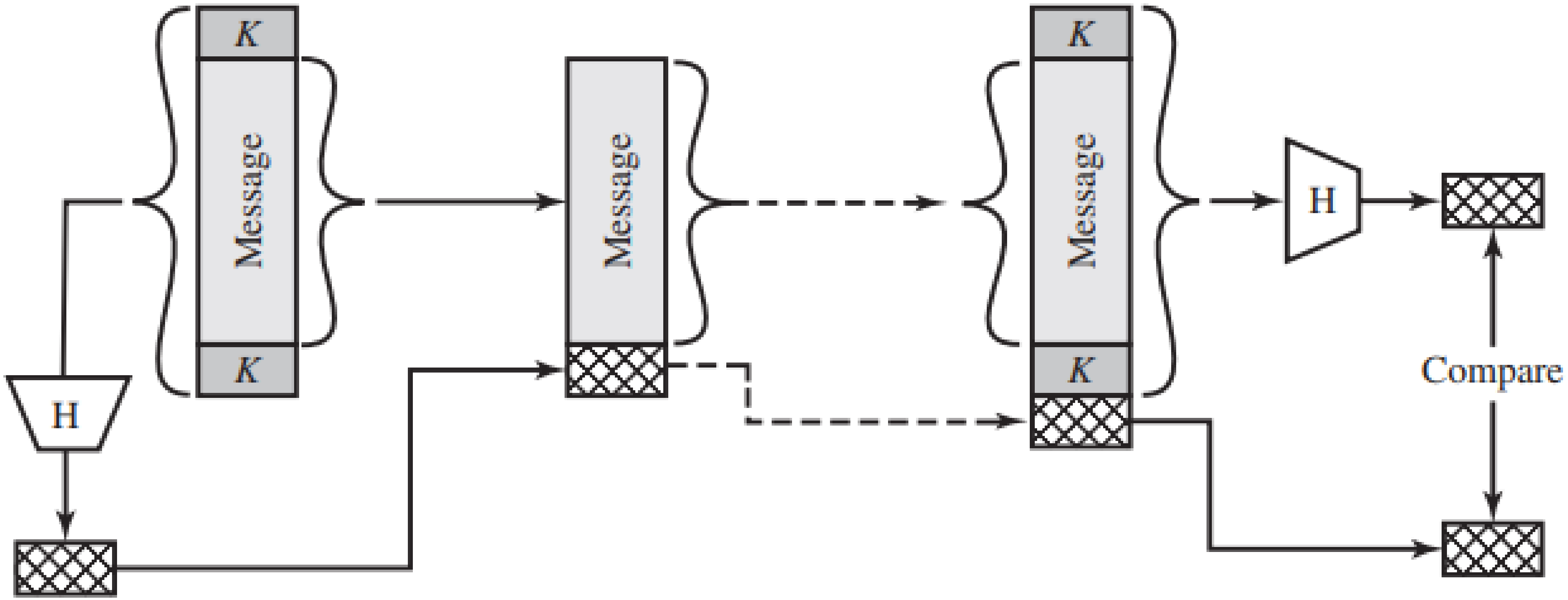
Figure 2.5 Message Authentication Using a One-Way Hash Function

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.



(b) Using public-key encryption

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



(c) Using secret value

Figure 2.5 Message Authentication Using a One-Way Hash Function

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.

2.3 PUBLIC-KEY ENCRYPTION

Of equal importance to symmetric encryption is public-key encryption, which finds use in message authentication and key distribution.

Public-Key Encryption Structure

Public-key encryption, first publicly proposed by Diffie and Hellman in 1976 [DIFF76], is the first truly revolutionary advance in encryption in literally thousands of years. Public-key algorithms are based on mathematical functions rather than on simple operations on bit patterns, such as are used in symmetric encryption algorithms. More important, public-key cryptography is **asymmetric**, involving the use of two separate keys, in contrast to symmetric encryption, which uses only one key. The use of two keys has profound consequences in the areas of confidentiality, key distribution, and authentication.

Public-Key Encryption Structure

Before proceeding, we should first mention several common misconceptions concerning public-key encryption. One is that public-key encryption is more secure from cryptanalysis than symmetric encryption. In fact, the security of any encryption scheme depends on (1) the length of the key and (2) the computational work involved in breaking a cipher. There is nothing in principle about either symmetric or public-key encryption that makes one superior to another from the point of view of resisting cryptanalysis. A second misconception is that public-key encryption is a general-purpose technique that has made symmetric encryption obsolete. On the contrary, because of the computational overhead of current public-key encryption schemes, there seems no foreseeable likelihood that symmetric encryption will be abandoned. Finally, there is a feeling that key distribution is trivial when using public-key encryption, compared to the rather cumbersome handshaking involved with key distribution centers for symmetric encryption. For public-key key distribution, some form of protocol is needed, often involving a central agent, and the procedures involved are no simpler or any more efficient than those required for symmetric encryption.

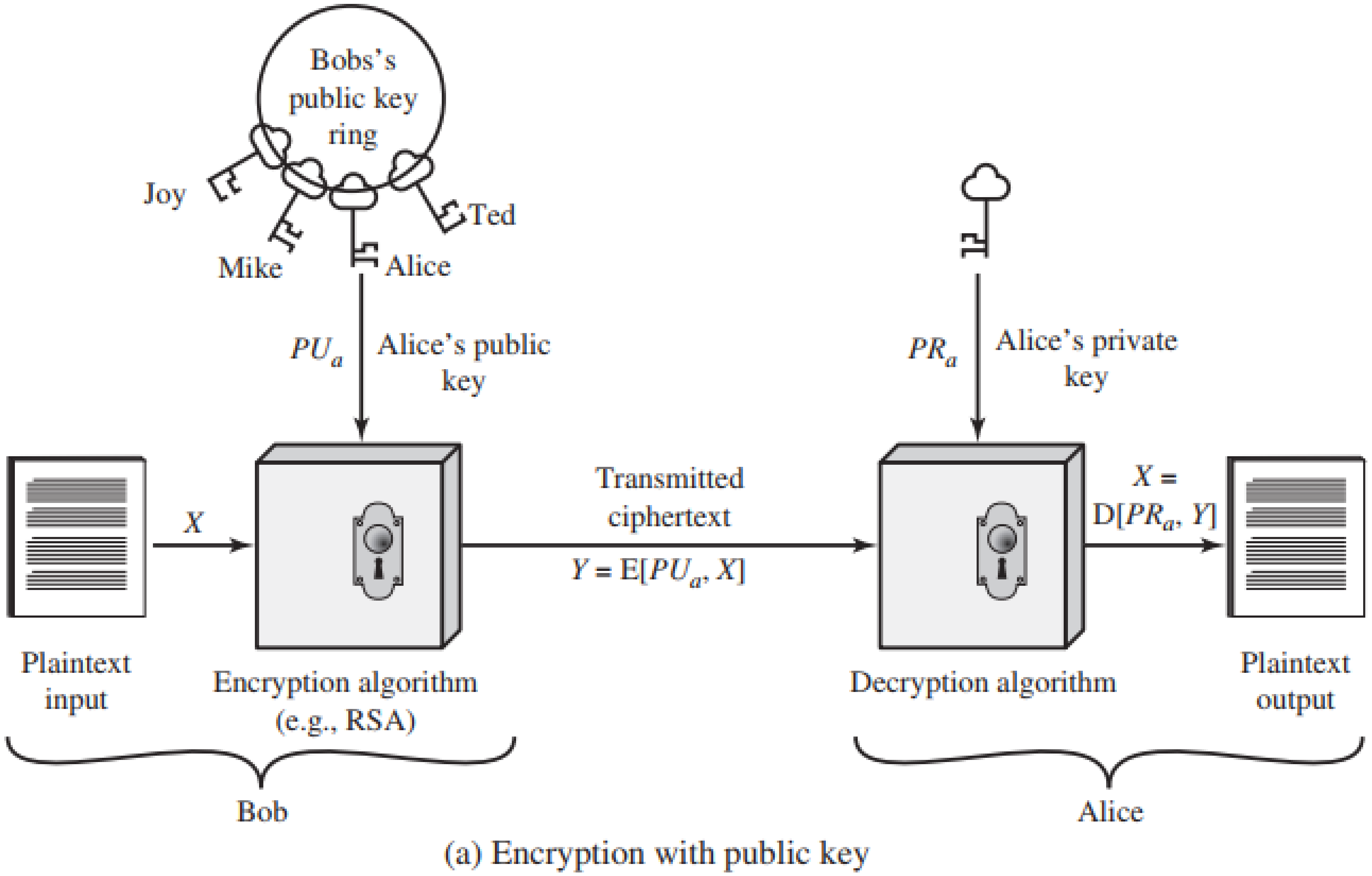
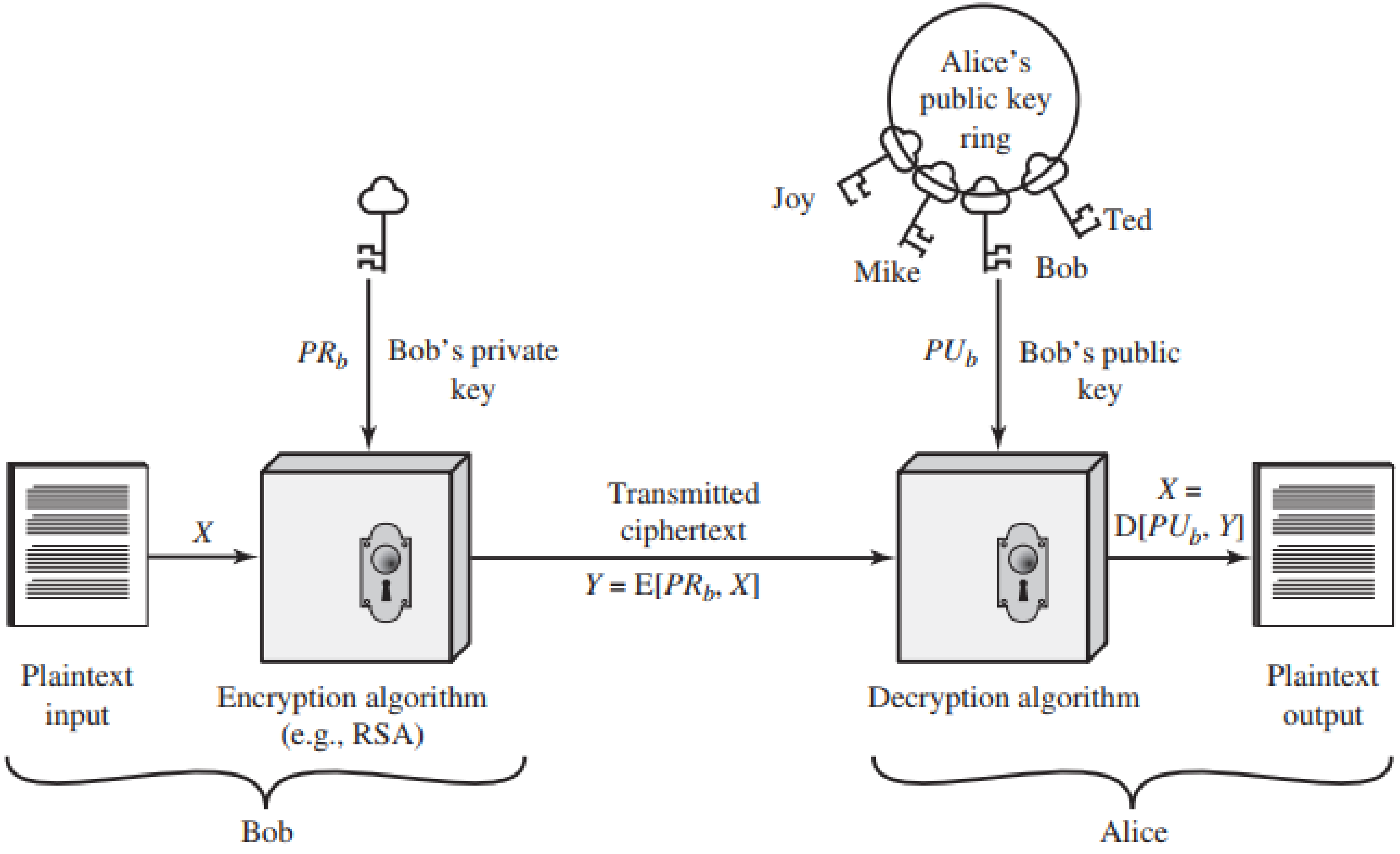


Figure 2.6 **Public-Key Cryptography**



(b) Encryption with private key

Figure 2.6 Public-Key Cryptography

Table 2.3 Applications for Public-Key Cryptosystems

Algorithm	Digital Signature	Symmetric Key Distribution	Encryption of Secret Keys
RSA	Yes	Yes	Yes
Diffie–Hellman	No	Yes	No
DSS	Yes	No	No
Elliptic Curve	Yes	Yes	Yes

* Digital Signature Standard (DSS)

Requirements for Public-Key Cryptography

The cryptosystem illustrated in Figure 2.6 depends on a cryptographic algorithm based on two related keys. Diffie and Hellman postulated this system without demonstrating that such algorithms exist. However, they did lay out the conditions that such algorithms must fulfill [DIFF76]:

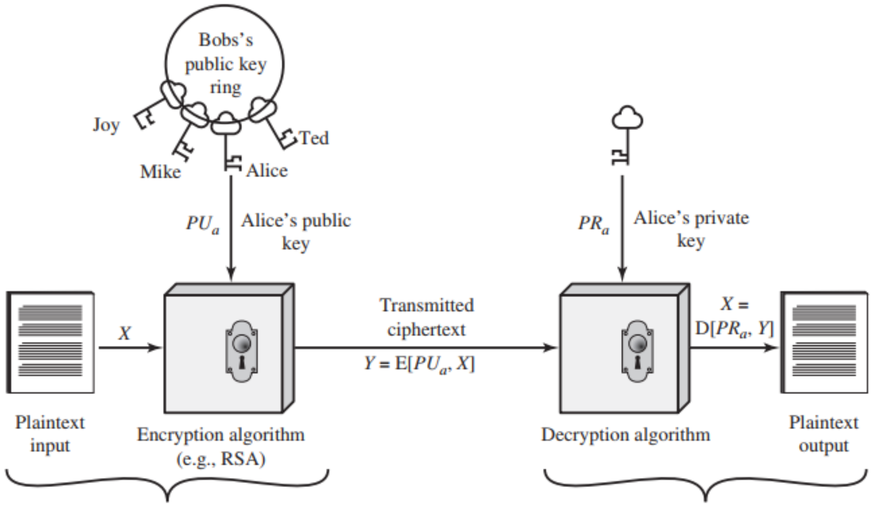
1. It is computationally easy for a party B to generate a pair (public key PU_b, private key PR_b).
2. It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext:

$$C = \mathrm{E}(PU_b, M)$$

3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message:

$$M = \mathrm{D}(PR_b, C) = \mathrm{D}[PR_b, \mathrm{E}(PU_b, M)]$$

4. It is computationally infeasible for an opponent, knowing the public key, PU_b, to determine the private key, PR_b.
5. It is computationally infeasible for an opponent, knowing the public key, PU_b, and a ciphertext, C, to recover the original message, M.



(a) Encryption with public key

Asymmetric Encryption Algorithms

RSA One of the first public-key schemes was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and first published in 1978 [RIVE78]. The RSA scheme has since reigned supreme as the most widely accepted and implemented approach to public-key encryption. RSA is a block cipher in which the plaintext and ciphertext are integers between 0 and $n - 1$ for some n.

used a public-key size (length of n) of 129 decimal digits, or around 428 bits. This result does not invalidate the use of RSA; it simply means that larger key sizes must be used. Currently, a 1024-bit key size (about 300 decimal digits) is considered strong enough for virtually all applications.

FACT 1. **Prime generation is easy:** It's easy to find a random prime number of a given size.

FACT 1, FACT 2 and Conjecture # 3 are not part of syllabus

FACT 2. **Multiplication is easy:** Given p and q, it's easy to find their product, $n = pq$.

CONJECTURE 3. **Factoring is hard:** Given such an n, it appears to be quite hard to recover the prime factors p and q.

2.4 DIGITAL SIGNATURES AND KEY MANAGEMENT

As mentioned in Section 2.3, public-key algorithms are used in a variety of applications. In broad terms, these applications fall into two categories: digital signatures, and various techniques to do with key management and distribution.

With respect to key management and distribution, there are at least three distinct aspects to the use of public-key encryption in this regard:

- The secure distribution of public keys
- The use of public-key encryption to distribute secret keys
- The use of public-key encryption to create temporary keys for message encryption

Digital Signature

Public-key encryption can be used for authentication with a technique known as the digital signature. NIST FIPS PUB 186-4 [*Digital Signature Standard (DSS)*, July 2013] defines a digital signature as follows: The result of a cryptographic transformation of data that, when properly implemented, provides a mechanism for verifying origin authentication, data integrity and signatory non-repudiation.

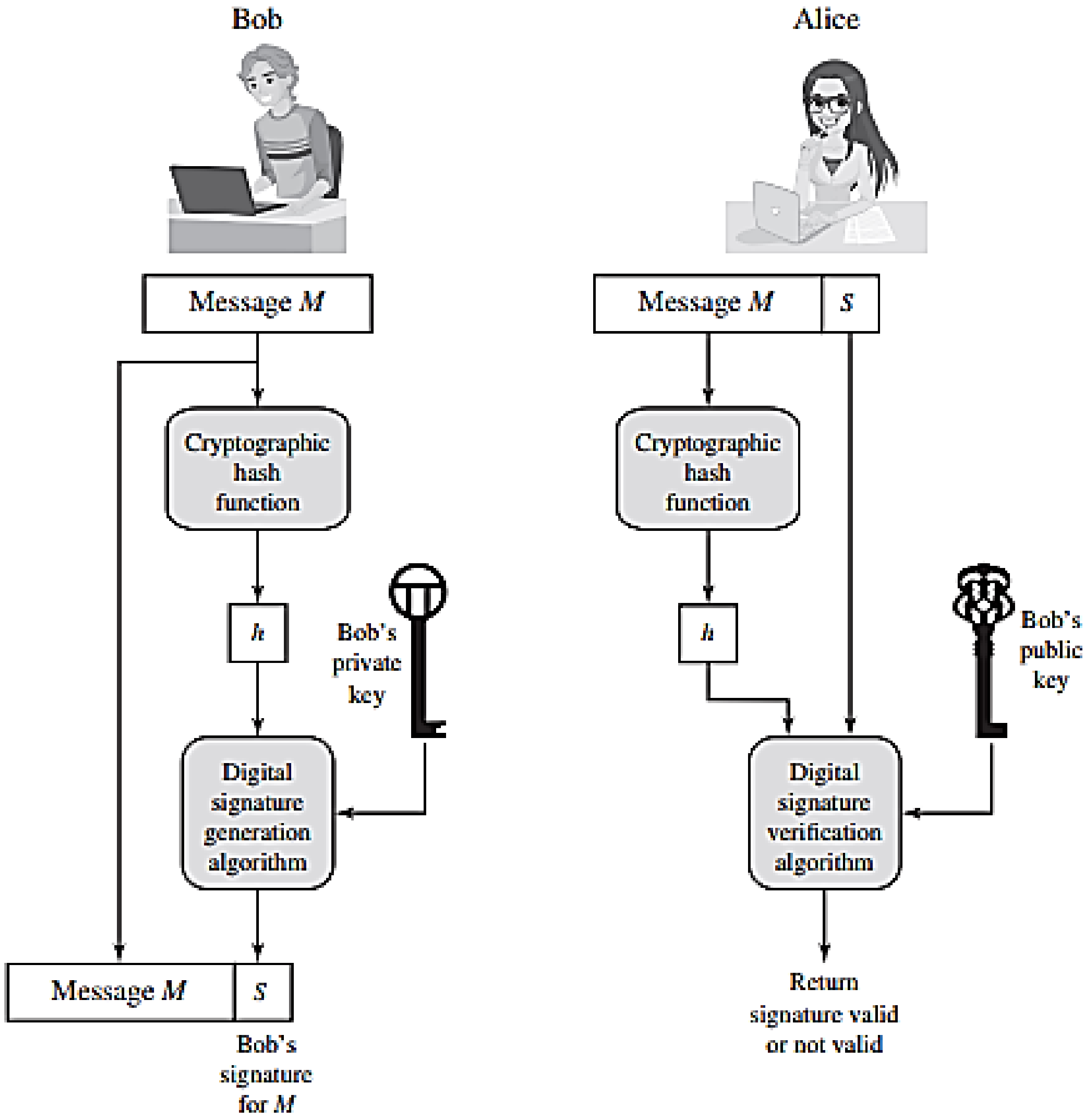
Thus, a digital signature is a data-dependent bit pattern, generated by an agent as a function of a file, message, or other form of data block. Another agent can access the data block and its associated signature and verify (1) the data block has been signed by the alleged signer, and (2) the data block has not been altered since the signing. Further, the signer cannot repudiate the signature.

Digital Signature

FIPS 186-4 specifies the use of one of three digital signature algorithms:

- **Digital Signature Algorithm (DSA):** The original NIST-approved algorithm, which is based on the difficulty of computing discrete logarithms.
- **RSA Digital Signature Algorithm:** Based on the RSA public-key algorithm.
- **Elliptic Curve Digital Signature Algorithm (ECDSA):** Based on elliptic-curve cryptography.

Digital Signature



(a) Bob signs a message

(b) Alice verifies the signature

Figure 2.7 Simplified Depiction of Essential Elements of Digital Signature Process

Public-Key Certificates

of the public key.

Anyone can forge public announcement. That is, some user could pretend to be Bob and send a public key to another participant or broadcast such a public key. Until such time as Bob discovers the forgery and alerts other participants, the forger is able to read all encrypted messages intended for Bob and can use the forged keys for authentication.

Public-Key Certificates

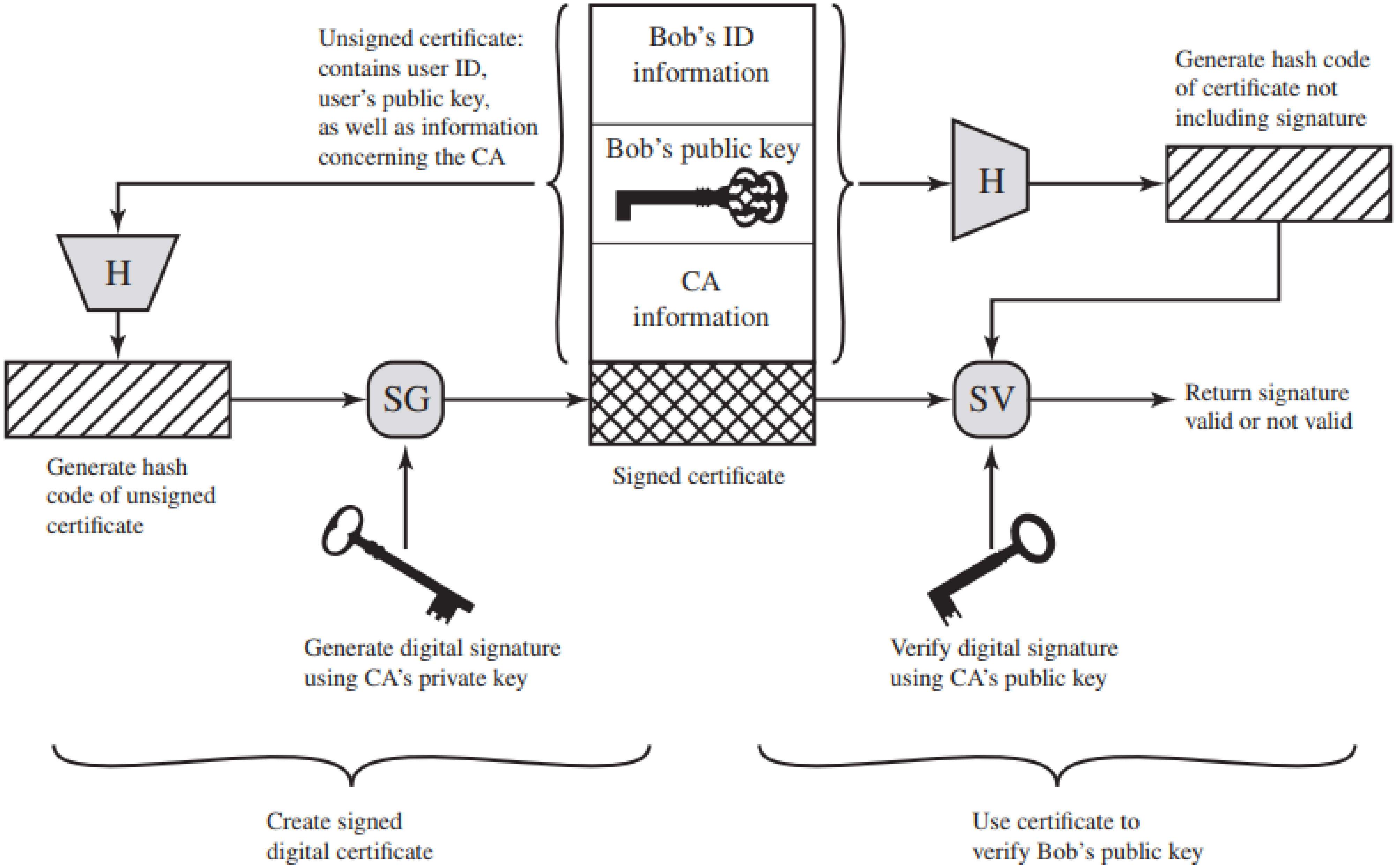


Figure 2.8 Public-Key Certificate Use

2.7 KEY TERMS, REVIEW QUESTIONS, AND PROBLEMS

Key Terms

Advanced Encryption Standard (AES)	collision resistant	Decryption
asymmetric encryption	confidentiality	Diffie–Hellman key exchange
authentication	cryptanalysis	digital signature
brute-force attack	Data Encryption Standard (DES)	Digital Signature Standard (DSS)
ciphertext	data integrity	elliptic curve cryptography
encryption	preimage resistant	second preimage resistant
hash function	private key	secret key
keystream	pseudorandom number	secure hash algorithm (SHA)
message authentication	public key	secure hash function
message authentication code (MAC)	public-key certificate	strong collision resistant
modes of operation	public-key encryption	symmetric encryption
one-way hash function	random number	triple DES
plaintext	RSA	weak collision resistant

Types of Cryptanalytic attacks

- **Known-Plaintext Analysis (KPA) :**
In this type of attack, some plaintext-ciphertext pairs are already known. Attacker maps them in order to find the encryption key. This attack is easier to use as a lot of information is already available.
- **Chosen-Plaintext Analysis (CPA) :**
In this type of attack, the attacker chooses random plaintexts and obtains the corresponding ciphertexts and tries to find the encryption key. Its very simple to implement like KPA but the success rate is quite low.
- **Ciphertext-Only Analysis (COA) :**
In this type of attack, only some cipher-text is known and the attacker tries to find the corresponding encryption key and plaintext. Its the hardest to implement but is the most probable attack as only ciphertext is required.
- **Man-In-The-Middle (MITM) attack :**
In this type of attack, attacker intercepts the message/key between two communicating parties through a secured channel.

- **Adaptive Chosen-Plaintext Analysis (ACPA) :**
This attack is similar CPA. Here, the attacker requests the cipher texts of additional plaintexts after they have ciphertexts for some texts.

CHAPTER 3

User Authentication

Table 3.1 Identification and Authentication Security Requirements (NIST SP 800-171)

	Basic Security Requirements:
1	Identify information system users, processes acting on behalf of users, or devices.
2	Authenticate (or verify) the identities of those users, processes, or devices, as a prerequisite to allowing access to organizational information systems.
	Derived Security Requirements:
3	Use multifactor authentication for local and network access to privileged accounts and for network access to non-privileged accounts.
4	Employ replay-resistant authentication mechanisms for network access to privileged and non-privileged accounts.
5	Prevent reuse of identifiers for a defined period.
6	Disable identifiers after a defined period of inactivity.
7	Enforce a minimum password complexity and change of characters when new passwords are created.
8	Prohibit password reuse for a specified number of generations.
9	Allow temporary password use for system logons with an immediate change to a permanent password.
10	Store and transmit only cryptographically-protected passwords.
11	Obscure feedback of authentication information.

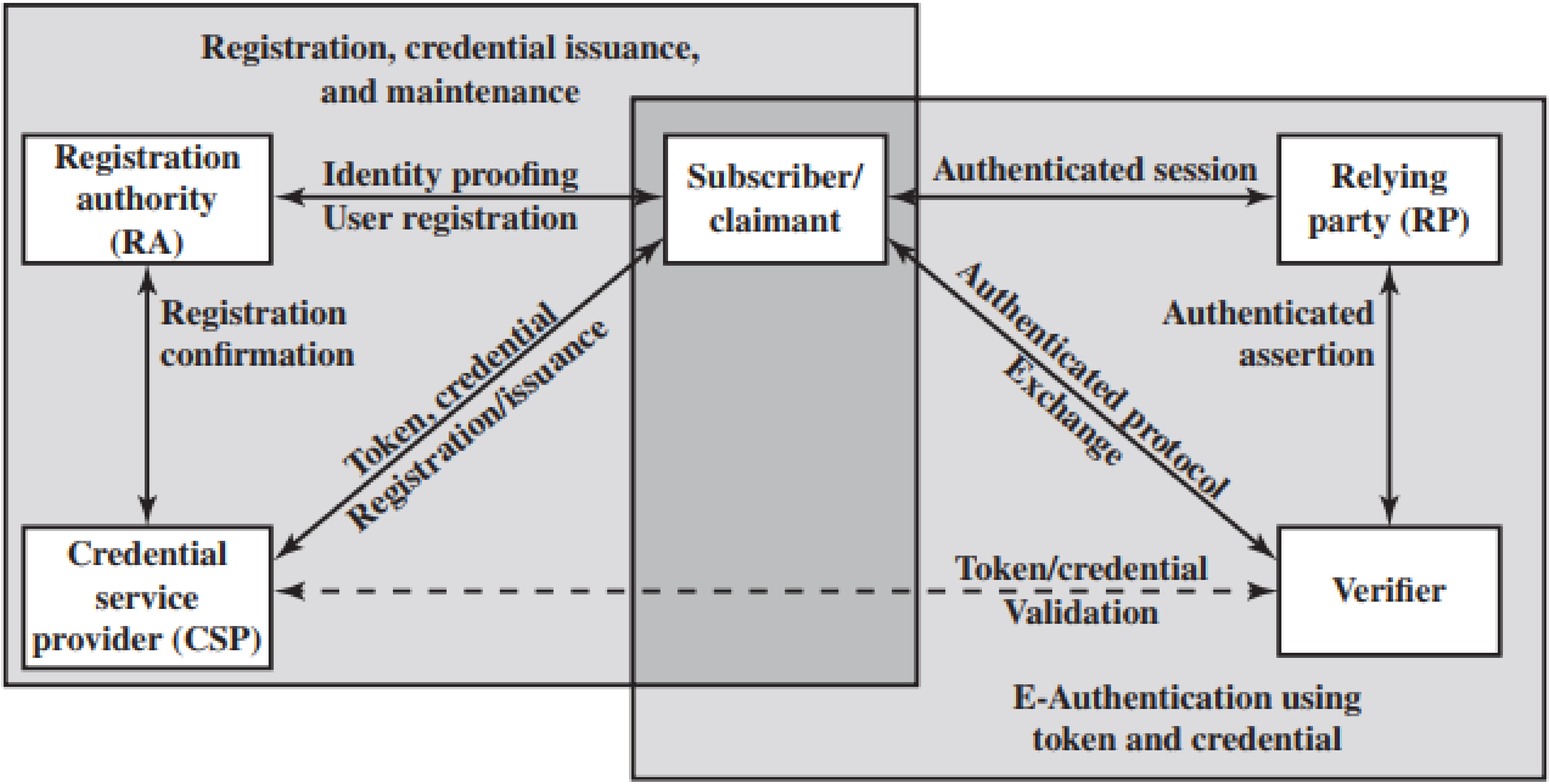


Figure 3.1 The NIST SP 800-63-3 E-Authentication Architectural Model

Means of Authentication

There are four general means of authenticating a user's identity, which can be used alone or in combination:

- **Something the individual knows:** Examples include a password, a personal identification number (PIN), or answers to a prearranged set of questions.
- **Something the individual possesses:** Examples include electronic keycards, smart cards, and physical keys. This type of authenticator is referred to as a *token*.
- **Something the individual is (static biometrics):** Examples include recognition by fingerprint, retina, and face.
- **Something the individual does (dynamic biometrics):** Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm.