Protocols for Secure Electronic Commerce

Mostafa Hashem Sherif

Algorithms and Architectures for Security

Publication details
Mostafa Hashem Sherif
Published online on: 25 May 2016

Accessed on: 15 Dec 2018

PLEASE SCROLL DOWN FOR DOCUMENT

Full terms and conditions of use: https://www.routledgehandbooks.com/legal-notices/terms

This Document PDF may be used for research, teaching and private study purposes. Any substantial or systematic reproductions, re-distribution, re-selling, loan or sub-licensing, systematic supply or distribution in any form to anyone is expressly forbidden.

The publisher does not give any warranty express or implied or make any representation that the contents will be complete or accurate or up to date. The publisher shall not be liable for an loss, actions, claims, proceedings, demand or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of this material.
The security of electronic commerce (e-commerce) transactions covers the security of access to the service, the correct identification and authentication of participants (to provide them the services that they have subscribed to), the integrity of the exchanges, and, if needed, their confidentiality. It may be necessary to preserve evidences to resolve disputes and litigation. All these protective measures may counter users’ expectations regarding anonymity and nontraceability of transactions.

This chapter contains a short review of the architectures and algorithms used to secure electronic commerce. In particular, the chapter deals with the following themes: definition of security services in open networks, security functions and their possible location in the various layers of the distribution network, mechanisms to implement security services, certification of the participants, and the management of encryption keys. Some potential threats to security are highlighted particularly as they relate implementation flaws.

This chapter has three appendices. Appendices 3A and 3B contain a general overview of the symmetric and public key encryption algorithms, respectively. Described in Appendix 3C are the main operations of the Digital Signature Algorithm (DSA) of the American National Standards Institute (ANSI) X9.30:1 and the Elliptic Curve Digital Signature Algorithm (ECDSA) of ANSI X9.62, first published in 1998 and revised in 2005.

### 3.1 Security of Open Financial Networks

Commercial transactions depend on the participants’ trust in their mutual integrity, in the quality of the exchanged goods, and in the systems for payment transfer or for purchase delivery. Because the exchanges associated with electronic commerce take place mostly at a distance, the climate of trust that is conducive must be established without the participants meeting in person and even if they use dematerialized forms of money or even digital currencies. The security of the communication networks involved is indispensable: those that link the merchant and the buyer, those that link the participants with their banks, and those linking the banks together.

The network architecture must be capable to withstand potential faults without important service degradation, and the physical protection of the network must be insured against fires, earthquakes, flooding, vandalism, or terrorism. This protection will primarily cover the network equipment (switches, trunks, information systems) but can also be extended to user-end terminals. However, the procedures to ensure such protection are beyond the scope of this chapter.

Recommendations X.800 (1991) from the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) categorizes the specific informational threats into two main categories: passive and active attacks. Passive attacks consist in the following:

1. Interception of the identity of one or more of the participants by a third party with a mischievous intent.
2. Data interception through clandestine monitoring of the exchanges by an outsider or an unauthorized user.

Active attacks take several forms such as the following:

- Replay of a previous message in its entirety or in part.
- Accidental or criminal manipulation of the content of an exchange by a third party by substitution, insertion, deletion, or any unauthorized reorganization of the user’s data $d$.
- Users’ repudiation or denial of their participation in part or in all of a communication exchange.
- Misrouting of messages from one user to another (the objective of the security service would be to mitigate the consequences of such an error as well).
- Analysis of the traffic and examination of the parameters related to a communication among users (i.e., absence or presence, frequency, direction, sequence, type, volume), in which this analysis would be made more difficult by encryption.
- Masquerade, whereby one entity pretends to be another entity.
• Denial of service and the impossibility of accessing the resources usually available to authorized users following the breakdown of communication, link congestion, or the delay imposed on time-critical operations.

Based on the preceding threats, the objectives of security measures in electronic commerce are as follows:

• Prevent an outsider other than the participants from reading or manipulating the contents or the sequences of the exchanged messages without being detected. In particular, that third party must not be allowed to play back old messages, replace blocks of information, or insert messages from multiple distinct exchanges without detection.

• Impede the falsification of Payment Instructions or the generation of spurious messages by users with dubious intentions. For example, dishonest merchants or processing centers must not be capable of reutilizing information about their clients’ bank accounts to generate fraudulent orders. They should not be able to initiate the processing of Payment Instructions without expediting the corresponding purchases. At the same time, the merchants will be protected from excessive revocation of payments or malicious denials of orders.

• Satisfy the legal requirements on, for example, payment revocation, conflict resolution, consumer protection, privacy protection, and the exploitation of data collected on clients for commercial purposes.

• Ensure reliable access to the e-commerce service, according to the terms of the contract.

• For a given service, provide the same level of service to all customers, irrespective of their location and the environmental variables.

The International Organization for Standardization (ISO) standard ISO 7498-2:1989 (ITU-T Recommendation X.800, 1991) describes a reference model for the service securities in open networks. This model will be the framework for the discussion in the next section.

3.2 OSI Model for Cryptographic Security

3.2.1 OSI Reference Model

It is well known that the Open Systems Interconnection (OSI) reference model of data networks establishes a structure for exchanges in seven layers (ISO/IEC 7498-1:1994):

1. The physical layer is where the electrical, mechanical, and functional properties of the interfaces are defined (signal levels, rates, structures, etc.).

2. The link layer defines the methods for orderly and error-free transmission between two network nodes.

3. The network layer is where the functions for routing, multiplexing of packets, flow control, and network supervision are defined.

4. The transport layer is responsible for the reliable transport of the traffic between the two network endpoints as well as the assembly and disassembly of the messages.

5. The session layer handles the conversation between the processes at the two endpoints.

6. The presentation layer manages the differences in syntax among the various representations of information at both endpoints by putting the data into a standardized format.

7. The application layer ensures that two application processes cooperate to carry out the desired information processing at the two endpoints.

The following section provides details about some cryptographic security functions that have been assigned to each layer.

3.2.2 Security Services: Definitions and Location

Security services for exchanges used in e-commerce employ mathematical functions to reshuffle the original message into an unreadable form before it is transmitted. After the message is received, the authenticated recipient must restore the text to its original status. The security consists of six services (Baldwin and Chang, 1997):

1. Confidentiality, that is, the exchanged messages are not divulged to a nonauthorized third party. In some applications, the confidentiality of addresses may be needed as well to prevent the analysis of traffic patterns and the derivation of side information that could be used.

2. Integrity of the data, that is, proof that the message was not altered after it was expedited and before the moment it was received. This service guarantees that the received data are exactly what were transmitted by the sender and that they were not corrupted, either intentionally or by error in transit in the network. Data integrity is also needed for network management data such as configuration files, accounting, and audit information.

3. Identification, that is, the verification of a pre-established relation between a characteristic
Algorithms and Architectures for Security

63

(e.g., a password or cryptographic key) and an entity. This allows control of access to the network resources or to the offered services based on the privileges associated with a given identity. One entity may possess several distinct identifiers. Furthermore, some protection against denial-of-service attacks can be achieved using access control.

4. **Authentication** of the participants (users, network elements, and network element systems), which is the corroboration of the identity that an entity claims with the guarantee of a trusted third party. Authentication is necessary to ensure nonrepudiation of users as well of network elements.

5. **Access control** to ensure that only the authorized participants whose identities have been duly authenticated can gain access to the protected resources.

6. **Nonrepudiation** is the service that offers an irrefutable proof of the integrity of the data and of their origin in a way that can be verified by a third party, for example, the nonrepudiation that the sender sent the message or that a receiver received the message. This service may also be called authentication of the origin of the data.

The implementation of the security services can be made over one or more layers of the OSI model. The choice of the layer depends on the several considerations as explained in the following text.

If the protection has to be accorded to all the traffic flow in a uniform manner, the intervention has to be at the physical or the link layers. The only cryptographic service that is available at this level is confidentiality by encrypting the data or similar means (frequency hopping, spread spectrum, etc.). The protection of the traffic at the physical layer covers all the flow, not only user data but also the information related to network administration: alarms, synchronization, update of routing table, and so on. The disadvantage of the protection at this level is that a successful attack will destabilize the whole security structure because the same key is utilized for all transmissions. At the link layer, encryption can be end to end, based on the source/destination, provided that the same technology is used all the way through.

Network layer encription achieves selective bulk protection that covers all the communications associated with a particular subnetwork from one end system to another end system. Security at the network layer is also needed to secure the communication among the network elements, particularly for link state protocols, where updates to the routing tables are automatically generated based on received information then flooded to the rest of the network.

For selective protection with recovery after a fault, or if the network is not reliable, the security services will be applied at the transport layer. The services of this layer apply end to end either singly or in combination. These services are authentication (whether simple by passwords or strong by signature mechanisms or certificates), access control, confidentiality, and integrity.

If more granular protection is required or if the nonrepudiation service has to be ensured, the encryption will be at the application layer. It is at this level that most of the security protocols for commercial systems operate, which frees them from a dependency on the lower layers. All security services are available.

It should be noted that there are no services at the session layer. In contrast, the services offered at the presentation layer are confidentiality, which can be selective such as by a given data field, authentication, integrity (in whole or in part), and nonrepudiation with a proof of origin or proof of delivery.

As an example, the Secure Sockets Layer (SSL)/Transport Layer Security (TLS) protocols are widely used to secure the connection between a client and a server. With respect to the OSI reference model, SSL/TLS lie between the transport layer and the application layer and will be presented in Chapter 5.

In some cases, it may be sufficient for an attacker to discover that a communication is taking place among partners and then attempt to guess, for example:

- The characteristics of the goods or services exchanged
- The conditions for acquisition such as delivery intervals, conditions, and means of settlement
- The financial settlement

The establishment of an enciphered channel or “tunnel” between two points at the network layer can constitute a shield against such types of attack. It should be noticed, however, that other clues, such as the relative time to execute the cryptographic operations, or the variations in the electric consumption or the electromagnetic radiation, can permit an analysis of the encrypted traffic and ultimately lead to breaking of the encryption algorithms (Messerges et al., 1999).

### 3.3 Security Services at the Link Layer

Internet Engineering Task Force (IETF) Request for Comment (RFC) 1661 (1994) defines the link-layer protocol Point-to-Point Protocol (PPP) to carry traffic between two entities identified with their respective IP addresses. The Layer 2 Tunneling Protocol (L2TP) defined in IETF
RFC 2661 (1999) extends the PPP operation by separating the processing of Internet Protocol (IP) packets within the PPP frames from that of the traffic flowing between the two ends at the link layer. This distinction allows a remote client to connect to a network access server (NAS) in a private (corporate) network through the public Internet, as follows. The client encapsulates PPP frames in an L2TP tunnel, prepends the appropriate L2TP header, and then transports the new IP packet using the User Datagram Protocol (UDP). The IP addresses in the new IP header are assigned by the local Internet Service Provider (ISP) at the local access point.

![Figure 3.1](image.png)

**Figure 3.1**
Layer 2 tunneling with L2TP.

Although L2TP does not provide any security services, it is possible to use Internet Protocol Security (IPSec) to secure the layer 2 tunnel because L2TP runs over IP. This is shown in the following section.

### 3.4 Security Services at the Network Layer

The security services at this layer are offered from one end of the network to the other end. They include network access control, authentication of the users and/or hosts, and authentication and integrity of the exchanges. These services are transparent to applications and end users, and their responsibilities fall on the administrators of network elements.

Authentication at the network layer can be simple or strong. *Simple* authentication uses a name and password pair (the password may be a one-time password), while *strong* authentication utilizes digital signatures or the exchange of certificates issued by a recognized certification authority (CA). The use of strong authentication requires the presence of encryption keys at all network nodes, which imposes the physical protection of all these nodes.

IPSec is a protocol suite defined to secure communications at the network layer between two peers. The most recent road map to the IPSec documentation is available in IETF RFC 6071 (2011). The overall security architecture of IPSec-v2 is described in IETF RFC 2401; the architecture of IPSec-v3 is described in RFC 4301 (2005).

IPSec offers authentication, confidentiality, and key management and is not tied to specific cryptographic algorithms. The Authentication Header (AH) protocol defined in IETF RFCs 4302 (2005) and 7321 (2014) provides authentication and integrity services for the payload as well as the routing information in the original IP header. The Encapsulating Security Payload (ESP) protocol is described in IETF RFCs 4303 (2005) and 7321 (2014) that define IPSec-v3. ESP focuses on the confidentiality of the original payload and the authentication of the encrypted data as well as the ESP header. Both IPSec protocols provide some protection against replay attacks with the help a monotonically increasing sequence number that is 64 bits long. The key exchange is performed with the Internet Key Exchange (IKE) version 2, the latest version of which is defined in RFCs 7296 (2014) and 7427 (2015).

IPSec operates in one of two modes: the transport mode and the tunnel mode. In the transport mode, the protection covers the payload and the transport header only, while the tunnel mode protects the whole packet, including the IP addresses. The transport mode secures the communication between two hosts, while the tunnel mode is useful when one or both ends of the connection are a trusted entity, such as a firewall, which provides the security services to an originating device. The tunnel mode is also employed when a router provides the security services to the traffic that it is forwarding (Doraiswamy and Harkins, 1999). Both modes are used to secure virtual private networks with IPSec as shown in Figure 3.2. The AH protocol is used for the transport mode only, while the ESP is applicable to both modes.

Illustrated in Figure 3.3 is the encapsulation in both cases. In this figure, the IPSec header represents either...
FIGURE 3.2
Securing virtual private networks with IPSec.

FIGURE 3.3
Encapsulation for IPSec modes.
Protocols for Secure Electronic Commerce

Thus, routing information associated with the private or corporate network can be encrypted after the establishment of a Transmission Control Protocol (TCP) tunnel between the firewall at the originating side and the one at the destination side. Note that ESP with no encryption (i.e., with a NULL algorithm) is equivalent to the AH protocol.

In verifying the integrity, the contents of fields in the IP header that change in transit (e.g., the “time to live”) are considered to be zero. With respect to transmission overheads, the length of the AH is at least 12 octets (a multiple of 4 octets for IPv4 and of 6 octets for IPv6). Similarly, the length of the ESP header is 8 octets. However, the total overhead for ESP includes 4 octets for the initialization vector (if it is included in the payload field), as well as an ESP trailer of at least 6 octets that comprise a padding and authentication data.

The protection of L2TP layer 2 tunnels with the IPSec protocol suite is described in IETF RFC 3193 (2001). When both IPSec and L2TP are used together, various headers are organized as shown in Figure 3.4.

IPSec AH (RFC 4302) and IPSec ESP (RFC 4303) define an antireplay mechanism using a sliding window that limits how far out of order a packet can be, relative to the authenticated packet with the highest sequence number. A received packet with a sequence number outside that window is dropped. In contrast, the window is advanced each time a packet is received with a sequence number within the acceptable range. RFCs 4302 and 4303 define minimum window sizes of 32 and 64 octets.

3.5 Security Services at the Application Layer

The majority of the security protocols for e-commerce operate at the application layer, which makes them independent of the lower layers. The whole gamut of security services is now available:

1. Confidentiality, total or selective by field or by traffic flow
2. Data integrity
3. Peer entity authentication
4. Peer entity authentication of the origin
5. Access control
6. Nonrepudiation of transmission with proof of origin
7. Nonrepudiation of reception with proof of reception

The Secure shell (SSH®), for example, provides security at the application layer and allows a user to log on, execute commands, and transfer files securely.

Additional security mechanisms are specific to a particular usage or to the end-user application at hand. For example, several additional parameters are considered to secure electronic payments, such as the ceiling of allowed expenses or withdrawals within a predefined time interval. Fraud detection and management depends on the surveillance of the following (Sabatier, 1997, p. 85):

- Activities at the points of sale (merchant terminals, vending machines, etc.)
- Short-term events
- Long-term trends, such as the behavior of a subpopulation, within a geographic area and in a specific time interval

In these cases, audit management takes into account the choice of events to collect and/or register, the validation of an audit trail, the definition of the alarm thresholds for suspected security violations, and so on.

The rights of intellectual property to dematerialized articles sold online pose an intellectual and technical challenge. The aim is to prevent the illegal reproduction of what is easily reproducible using “watermarks” incorporated in the product. The means used differ depending on whether the products protected are ephemeral (such as news), consumer oriented (such as films, music, books, articles, or images) or for production (such as enterprise software).

* Secure Shell and SSH are registered trademarks of SSH Communications Security, Ltd. of Finland.
Algorithms and Architectures for Security

Next, we give an overview of the mechanisms used to implement security service. The objective is to present sufficient background for understanding the applications and not to give an exhaustive review. More comprehensive discussions of the mathematics of cryptography are available elsewhere (Schneier, 1996; Menezes et al., 1997; Ferguson et al., 2010; Paar and Pelzl, 2010).

3.6 Message Confidentiality

Confidentiality guarantees that information will be communicated solely to the parties that are authorized for its reception. Concealment is achieved with the help of encryption algorithms. There are two types of encryption: symmetric encryption, where the operations of message obfuscation and revelation use the same secret key, and public key encryption, where the encryption key is secret and the revelation key is public.

3.6.1 Symmetric Cryptography

Symmetric cryptography is the tool that is employed in classical systems. The key that the sender of a secret message utilizes to encrypt the message is the same as the one that the legitimate receiver uses to decrypt the message. Obviously, key exchange among the partners has to occur before the communication, and this exchange takes place through other secured channels. The operation is illustrated in Figure 3.5.

Let $M$ be the message to be encrypted with a symmetric key $K$ in the encryption process $E$. The result will be the ciphertext $C$ such that

$$E[K(M)] = C$$

The decryption process $D$ is the inverse function of $E$ that restores the clear text:

$$D(C) = M$$

There are two main categories of symmetric encryption algorithms: block encryption algorithms and stream cipher algorithms. Block encryption acts by transforming a block of data of fixed size, generally 64 bits, in encrypted blocks of the same size. Stream ciphers convert the clear text one bit at a time by combining the stream of bits in the clear text with the stream of bits from the encryption key using an exclusive OR (XOR).

Table 3.1 presents in alphabetical order the main algorithms for symmetric encryption used in e-commerce applications.

Fortezza is the Cryptographic Application Programming Interface (CAPI) that the National Security Agency (NSA) defined for security applications running on PCMCIA (Personal Computer Memory Card International Association) cards. SKIPJACK algorithm is used for encryption, and the Key Exchange Algorithm (KEA) is the algorithm for key exchange. The experimental specifications of IETF RFC 2773 (2000) describe the use of SKIPJACK and KEA for securing file transfers.

The main drawback of symmetric cryptography systems is that both parties must obtain, one way or another, the unique encryption key. This is possible without too much trouble within a closed organization; on open networks, however, the exchange can be intercepted. Public key cryptography, proposed in 1976 by Diffie and Hellman, is one solution to the problem of key exchange.

3.6.2 Public Key Cryptography

Algorithms of public key cryptography introduce a pair of keys for each participant, a private key $SK$ and a public key $PK$. The keys are constructed in such a way that it is practically impossible to reconstitute the private key with the knowledge of the public key.

![FIGURE 3.5 Symmetric encryption.](image-url)
Consider two users, $A$ and $B$, each having a pair of keys $(PK_A, SK_A)$ and $(PK_B, SK_B)$, respectively. Thus,

1. To send a secret message $x$ to $B$, $A$ encrypts it with $B$'s public key and then transmits the encrypted message to $B$. This is represented by

$$e = PK_B(x)$$

2. $B$ recovers the information using his or her private key $SK_B$. It should be noted that only $B$ possesses $SK_B$, which can be used to identify $B$. The decryption operation can be represented by

$$x = SK_B(e)$$

3. $B$ can respond to $A$ by sending a new secret message $x'$ encrypted with the public key $PK_A$ of $A$:

$$e' = PK_A(x')$$

4. $A$ obtains $x'$ by decrypting $e'$:

$$x' = SK_B(e')$$

The diagram in Figure 3.6 summarizes these exchanges.

**FIGURE 3.6**
It is worth noting that the preceding exchange can be used to verify the identity of each participant. More precisely, A and B are identified by the possession of the decryption key, SK_A or SK_B, respectively. A can determine if B possesses the private decryption key SK_B if the initial message x is included in the returned message x’ that B sends. This indicates to A that the communication has been made with the entity that possesses SK_B. B can also confirm the identity of A in a similar way.

The de facto standard for public key encryption is the algorithm RSA invented by Rivest et al. (1978). In many new applications, however, elliptic curve cryptography (ECC) offers significant advantages as described in Appendix 3B.

3.7 Data Integrity

The objective of the integrity service is to eliminate all possibilities of nonauthorized modification of messages during their transit from the sender to the receiver. The traditional form to achieve this security is to stamp the letter envelope with the wax seal of the sender. Transposing this concept to electronic transactions, the seal will be a sequence of bits associated univocally with the document to be protected. This sequence of bits will constitute a unique and unfalsifiable “fingerprint” that will accompany the document sent to the destination. The receiver will then recalculate the value of the fingerprint from the received document and compare the value obtained with the value that was sent. Any difference will indicate that the message integrity has been violated.

The fingerprint can be made to depend on the message content only by applying a hash function. A hash function converts a sequence of characters of any length into a chain of characters of a fixed length, L, usually smaller than the original length, called hash value. However, if the hash algorithm is known, any entity can calculate the hash value from the message using the hash function. For security purposes, the hash value depends on the message content and the sender’s private key in the case of a public key encryption algorithm or a secret key that only the sender and the receiver know in the case of a symmetric encryption algorithm. In the first case, anyone who knows the hash function can calculate the fingerprint with the public key of the sender; in the second case, only the intended receiver will be able to verify the integrity. It should be noted that lack of integrity can be used to break confidentiality. For example, the confidentiality of some algorithms may be broken through attacks on the initialization vectors.

The hash value has many names: compression, contraction, message digest, fingerprint, cryptographic checksum, message integrity check (MIC), and so on (Schneier, 1996, p. 31).

3.7.1 Verification of the Integrity with a One-Way Hash Function

A one-way hash function is a function that can be calculated relatively easily in one direction but with considerable difficulty in the inverse direction. A one-way hash function is sometimes called a compression function or a contraction function.

To verify the integrity of a message whose fingerprint has been calculated with the hash function H(), this function should also be a one-way function; that is, it should meet the following properties:

1. Absence of collisions: In other words, the probability of obtaining the same hash value with two different texts should be almost null. Thus, for a given message x, the probability of finding a different message x’ such that H(x) = H(x’), is extremely small. For the collision probability to be negligible, the size of the hash value L should be sufficiently large.

2. Impossibility of inversion: Given the fingerprint h of a message x, it is practically impossible to calculate x such that H(x) = h.

3. A widespread among the output values: This is because a small difference between two messages should yield a large difference between their fingerprints. Thus, any slight modification in the original text should, on average, affect half of the bits of the fingerprint.

Consider the message X. It will have been divided into n blocks, each consisting of B bits. If needed, padding bits would be appended to the message, according to a defined scheme, so that the length of each block reaches the necessary B bits. The operations for cryptographic hashing are described using a compression function f() according to the following recursive relationship:

\[ h_i = f(h_{i-1}, x_i), \quad i = 1, \ldots, n \]

In this equation, h_0 is the vector that contains an initial value of L bits and x = [x_1, x_2, ..., x_n] is the message subdivided into n vectors of B bits each. The hash algorithms that are commonly used in e-commerce are listed in Table 3.2 in alphabetical order.

For the MD5 and SHA-1 hashing algorithms, the message is divided into blocks of 512 bits. The padding consists, in appending to the last block, a binary “1,” then as many “0” bits as necessary for the size of the last block with padding to be 448 bits. Next, a suffix of 8 octets is added to contain the length of the initial message (before padding) coded over 64 bits,
which brings the total size of the last block to 512 bits of 64 octets.

For a hash algorithm with an output of \( n \) bits, there are \( N = 2^n \) hashes, which are assumed to be equally probable. Based on the generalization of the birthday problem, the event that there are no collisions after \( q \) hashes has a probability given by (Barthélemy et al., 2005, pp. 390–391; Paar and Pelzl, 2010, pp. 299–303):

\[
\prod_{i=1}^{q}(1 - \frac{i}{N}) \approx \prod_{i=1}^{q} e^{-\frac{i}{N}} = e^{-\sum_{i=1}^{q} \frac{1}{N}}
\]

\[
= e^{-\frac{q(q-1)}{2N}}
\]

Therefore, the probability of at least one collision is given by

\[
1 - e^{-\frac{q(q-1)}{2N}}
\]

Now, put the probability of at least one collision to be 50%. Solving for \( q \) by taking the natural logarithm of both sides, we get

\[
q(q-1) = 2N\ln(2)
\]

Since \( q \) is large, \( (q-1) = q \), therefore,

\[
q^2 = 2N\ln(2),
\]

i.e.,

\[
q = \sqrt{2N\ln(2)} = 1.18\sqrt{N} \approx \sqrt{N} \text{ or } 2^n
\]

Thus, the number of messages to hash to find a collision is roughly the square root of the number of possible output values. The so-called birthday attack on cryptographic hash functions of 256 bits indicates that the probability of a collision exceeds 50% after around \( 2^{128} \) hashes.

In 1994, two researchers, van Oorschot and Wiener, were able to detect collisions in the output of MD5 (van Oorschot and Wiener, 1994), which explains its gradual replacement with SHA-1. In 2007, Stevens et al. (2009) exploited the vulnerability of MD5 to collisions to construct two X.509 certificates for SSL/TLS traffic with the identical MD5 signature but different public keys and belonging to different entities without the knowledge and/or assistance of the relevant certification authority. Knowledge that rogue SSL/TLS certificates can be easily forged accelerated the migration of certification authorities away from MD5.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Name</th>
<th>Signature Size (L) in Bits</th>
<th>Block Size (B) in Bits</th>
<th>Standardization</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR/DFP</td>
<td>Hashing algorithms of German banks</td>
<td></td>
<td></td>
<td>German Banking Standards</td>
</tr>
<tr>
<td>DSMR</td>
<td>Digital signature scheme giving message recovery</td>
<td></td>
<td></td>
<td>ISO/IEC 9796</td>
</tr>
<tr>
<td>MCCP</td>
<td>Banking key management by means of public key algorithms using the RSA cryptosystem; signature construction by means of a separate signature</td>
<td></td>
<td></td>
<td>ISO/IEC 1116-2</td>
</tr>
<tr>
<td>MD4</td>
<td>Message digest algorithm</td>
<td>128</td>
<td>512</td>
<td>No, but described in RFC 1320</td>
</tr>
<tr>
<td>MD5</td>
<td>Message digest algorithm</td>
<td>128</td>
<td>512</td>
<td>No, but described in RFC 1321</td>
</tr>
<tr>
<td>NVB7.1, NVBAK</td>
<td>Hashing functions used by Dutch banks</td>
<td></td>
<td></td>
<td>Dutch Banking Standard, published in 1992</td>
</tr>
<tr>
<td>RIPEMD</td>
<td>Extension of MD4, developed during the European project RIPE (Menezes et al., 1997, p. 380)</td>
<td>128</td>
<td>512</td>
<td>ISO/IEC 10118-3</td>
</tr>
<tr>
<td>RIPEMD-128</td>
<td>Dedicated hash function #2</td>
<td>128</td>
<td>512</td>
<td>FIPS 180</td>
</tr>
<tr>
<td>RIPEMD-160</td>
<td>Improved version of RIPEMD (Dobbertin et al., 1996)</td>
<td>160</td>
<td>512</td>
<td>ISO/IEC 10118-3</td>
</tr>
<tr>
<td>SHA</td>
<td>Secure Hash Algorithm (replaced by SHA-1)</td>
<td>160</td>
<td>512</td>
<td>FIPS 180-1 (National Institute of Standards and Technology, 1995), FIPS 180-4 (National Institute of Standards and Technology, 2012b)</td>
</tr>
<tr>
<td>SHA1 (SHA-1)</td>
<td>Dedicated Hash Function #3 (revision and correction of the Secure Hash Algorithm)</td>
<td>160</td>
<td>512</td>
<td>FIPS 180-4</td>
</tr>
<tr>
<td>SHA-2</td>
<td></td>
<td>224, 256</td>
<td>512</td>
<td>FIPS 180</td>
</tr>
<tr>
<td></td>
<td></td>
<td>384, 512</td>
<td>1024</td>
<td>FIPS 180-4</td>
</tr>
</tbody>
</table>
### Table 3.3
Comparison of the Hash Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Output Size (Bits)</th>
<th>Block Size (Bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD5</td>
<td>128</td>
<td>512</td>
</tr>
<tr>
<td>SHA-1</td>
<td>160</td>
<td>512</td>
</tr>
<tr>
<td>SHA-2</td>
<td>SHA-224</td>
<td>224</td>
</tr>
<tr>
<td></td>
<td>SHA-256</td>
<td>256</td>
</tr>
<tr>
<td></td>
<td>SHA-384</td>
<td>384</td>
</tr>
<tr>
<td></td>
<td>SHA-512</td>
<td>512</td>
</tr>
<tr>
<td></td>
<td>SHA-512/224</td>
<td>224</td>
</tr>
<tr>
<td></td>
<td>SHA-512/256</td>
<td>256</td>
</tr>
<tr>
<td>SHA-3</td>
<td>SHA3-224</td>
<td>224</td>
</tr>
<tr>
<td></td>
<td>SHA3-256</td>
<td>256</td>
</tr>
<tr>
<td></td>
<td>SHA3-384</td>
<td>384</td>
</tr>
<tr>
<td></td>
<td>SHA3-512</td>
<td>512</td>
</tr>
<tr>
<td></td>
<td>SHAKE128</td>
<td>Arbitrary</td>
</tr>
<tr>
<td></td>
<td>SHAKE256</td>
<td>1088</td>
</tr>
</tbody>
</table>


Federal Information Processing Standard (FIPS) 180-2 is the standard that contains the specifications of both SHA-1 and SHA-2. SHA-2 is a set of cryptographic hash functions (SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256) published in 2001 by the National Institute of Standards and Technology (NIST) as a U.S. Federal Information Processing Standard (FIPS). SHA-2 includes a significant number of changes from its predecessor, SHA-1.

In 2007, NIST announced a competition for a new hash standard, SHA-3, to which 200 cryptographers from around the world submitted 64 proposals. In December 2008, 51 candidates were retained in the first round. In July 2009, the candidate list was reduced to 14 entries. Five finalists were selected in December 2010. After a final round of evaluations, NIST selected Keccak as SHA-3 in October 2012. Keccak was developed by a team of four researchers from STMicroelectronics, a European semiconductor company, and is optimized for hardware but can be implemented in software as well. It uses an entirely different technique from previous cryptographic algorithms, which use a compression function to process fixed-length blocks of data. Keccak applies a permutation process to extract a fingerprint from the data, a technique given the name of a “sponge function” (Harbert, 2012).

Table 3.3 provides a comparison of the parameters of the various hash algorithms.

#### 3.7.2 Verification of the Integrity with Public Key Cryptography

An encryption algorithm with a public key is called “permutable” if the decryption and encryption operations can be inverted, that is, if

\[ M = PK_X(SK_X(M)) \]

In the case of encryption with a permutable public key algorithm, an information element \( M \) that is encrypted by the private key \( SK_X \) of an entity \( X \) can be read by any user possessing the corresponding public key \( PK_X \).

A sender can, therefore, sign a document by encrypting it with a private key reserved for the signature operation to produce the seal that accompanies the message. Any person who knows the corresponding public key will be able to decipher the seal and verify that it corresponds to the received message.

Another way of producing the signature with public key cryptography is to encrypt the fingerprint of the document. This is because the encryption of a long document using a public key algorithm imposes substantial computations and introduces excessive delays. Therefore, it is beneficial to use a digest of the initial message before applying the encryption. This digest is produced by applying a one-way hash function to calculate the fingerprint that is then encrypted with the sender’s private key. At the destination, the receiver recomputes the fingerprint. With the public key of the sender, the receiver will be able to decrypt the fingerprint to verify if the received hash value is identical to the computed hash value. If both are identical, the signature is valid.

The block diagram in Figure 3.7 represents the verification of the integrity with public key encryption of the hash. In this figure, \( h \) represents the hash function, \( O \) represents the encryption with the public key, and \( O^{-1} \) represents the decryption function to extract the hash for comparison with the recalculated hash at the receiving end.

The public key algorithms, which are frequently used to calculate digital signatures, are listed in Table 3.4.

Even though this method allows the verification of the message integrity, it does not guarantee that the identity of the sender is authentic.

In the case of a public key, a signature produced from a message with the signer’s private key and then verified with the signer’s corresponding public key is sometimes called a “signature scheme with appendix” (IETF RFC 3447, 2003).

#### 3.7.3 Blind Signature

A blind signature is a special procedure for a notary to sign a message using the RSA algorithm for public key cryptography without revealing the content (Chaum, 1983, 1989). One possible utilization of this technique is to time-stamp digital payments.

Consider a debtor who would like to have a payment blindly signed by a bank. The bank has a public key \( e \), a private key \( d \), and a public modulo \( N \). The debtor
chooses a random number $k$ between 1 and $N$ and keeps this number secret.

The payment $p$ is “enveloped” by applying

$$(pk^e) \mod N$$

before sending the message to the bank. The bank signs it with its private key so that

$$(pk^e)^d \mod N = p^d \mod N$$

and returns the payment to the debtor. The debtor can now extract the signed note by dividing the number by $k$. To verify that the note received from the bank is the one that has been sent, the debtor can raise it to the $e$ power because (as will be shown in Appendix 3B)

$$(p^d)^e \mod N \equiv p \mod N$$

The various payment protocols for digital money take advantage of blind signatures to satisfy the conditions of anonymity.

### 3.7.4 Verification of the Integrity with Symmetric Cryptography

The message authentication code (MAC) is the result of a one-way hash function that depends on a secret key. This mechanism guarantees simultaneously the integrity of the message content and the authentication of the sender. (As previously mentioned, some authors call the MAC the “integrity check value” or the “cryptographic checksum.”)

The most obvious way of constructing a MAC is to encrypt the hash value with a symmetric block encryption algorithm. The MAC is then affixed to the initial message, and the whole is sent to the receiver. The receiver recomputes the hash value by applying the same hash function on the received message and compares the result obtained with the decrypted MAC value. The equality of both results confirms the data integrity.

The block diagram in Figure 3.8 depicts the operations where $h$ represents the hash function, $C$ the encryption function, and $D$ the decryption function.

Another variant of this method is to append the secret key to the message that will be condensed with the hash functions.
It is also possible to perform the computations with the compression function $f()$ and use as an initial value the vector of the secret key, $k$, of length $L$ bits in the following recursion:

$$k_i = f(k_{i-1}, x_i), \quad i = 1, \ldots, n$$

where $x = \{x_1, x_2, \ldots, x_n\}$ is the message subdivided into $n$ vectors, each of $B$ bits. The MAC is the value of the final output $k_n$.

The procedure that several U.S. and international standards advocate, for example, ANSI X9.9 (1986) for the authentication of banking messages and ISO 8731-1 (1987) and ISO/International Electrotechnical Commission (IEC) 9797-2 (2002) for implementing a one-way hash function, is to encrypt the message with a symmetric block encryption algorithm either in the cipher block chaining (CBC) or in the cipher feedback (CFB) modes. The MAC is the last encrypted block, which is encrypted one more time in the same CBC or CFB mode.

The following key hashing method augments the speed of computation in software implementation and increases the protection, even when the one-way hash algorithm experiences some rare collisions (Bellare et al., 1996).

Consider the message $X$ subdivided into $n$ vectors of $B$ bits each and two keys ($k_1$ and $k_2$), each of $L$ bits. The padding bits are added to the end of the initial message according to a determined pattern. The hashing operations can thus be described with the help of two compression functions $f_1()$ and $f_2()$:

$$k_1^i = f_1(k_{i-1}^i, x_i)$$

$$k_2^i = f_2(k_{i-1}^i, k_i)$$

where $k_0^1$ and $k_0^2$ are the initial values of $k_1$ and $k_2$, respectively.

$$x = x_1, x_2, \ldots, x_n$$

The result that this method yields is denoted as the Nested Message Authentication Code (NMAC). It is, in effect, constructed by applying compression functions in sequence, the first on the padded initial message and the second on the product of the first operation after padding.

The disadvantage of this method is that it requires access to the source code of the compression functions to change the initial values. In addition, it requires the usage of two secret keys. This explains the current popularity of the keyed hashed message authentication code (HMAC), which is described in IETF RFC 2104 (1997) and NIST FIPS 198-1 (2008). This method uses one single key $k$ of $L$ bits.

Assuming that the function $H()$ represents the initial hash function, the value of the HMAC is computed in the following manner:

$$\text{HMAC}_k(x) = H[k \oplus \text{opad} \parallel H[k \oplus \text{ipad}, x0]]$$

In this construction, $k$ is the vector of minimum length of $L$ bits, which after padding with a series of 0 bits will reach a total length of $B$ bits. The variables $\text{opad}$ and $\text{ipad}$ are constants for outer padding and inner padding, respectively. The variable $\text{opad}$ is formed with the octet 0x36 repeated as many times as needed to constitute a block of $B$ bits. The variable $\text{ipad}$ is the octet 0x5C repeated as many times. For MD5 and SHA-1, SHA-256, and SHA-512, the number of repetitions is 64. Finally, the symbols $\parallel$ and $\oplus$ in the previous equation.
denote, respectively, the concatenation and exclusive OR operations.

It should be noted that with the following representation

\[ k^1 = f_1(k \oplus ipad) \]
\[ k^2 = f_2(k \oplus opad) \]

the HMAC becomes the same as the nested MAC.

### 3.8 Identification of the Participants

Identification is the process of ascertaining the identity of a participant (whether a person or a machine) by relying on uniquely distinguishing feature. This contrasts with authentication, which is the confirmation that the distinctive identifier indeed corresponds to the declared user.

Authentication and identification of a communicating entity take place simultaneously when one party sends to the other in private a secret that is only shared between them, for example, a password or a secret encryption key. Another possibility is to pose a series of challenges that only the legitimate user is supposed to be capable of answering.

A digital signature is the usual means of identification because it associates a party (a user or a machine) with a shared secret. Other methods of simultaneous identification and authentication of human users exploit distinctive physiological and behavioral characteristics such as fingerprints, voiceprints, the shape of the retina, the form of the hand, signature, gait, and so on. The biometric identifiers used in electronic commerce are elaborated upon the following section.

### 3.9 Biometric Identification

Biometry systems use pattern recognition algorithms on specific physiological and/or behavioral characteristics of individuals. Biometric identification systems recognize the identity of an individual by matching the features extracted from a biometric image with an entry in a database of templates. Identification is used mostly in forensic and law enforcement applications. Verification systems, in contrast, authenticate a person's identity by comparing a newly captured biometric image with that of person's biometric template stored in the system storage or in that user's credential (or identity card) (Maltoni et al., 2003, p. 3). Verification is the operation that controls access to resources such as a bank account or a payment system.

Biometrics was reserved until recently for forensic and military applications but is now used in many civilian applications. Biometric systems present several advantages over traditional security methods based on what the person knows (password or personal identification number [PIN]) or possesses (card, pass, or physical key). In traditional systems, the user must have several keys or cards, one for each facility, and remember different passwords to access each system. These keys or cards can be damaged, lost, or stolen, and long passwords can be easily forgotten, particularly if they are difficult and/or changed at regular intervals. When a physical card or key or a password is compromised, it is not possible to distinguish between a legitimate user and one who has acquired access illegally. Finally, the use of biological attributes for identification and authentication bypasses some of the problems associated with cryptography (e.g., key management).

There are two main categories of biometric features. The first category relates to behavioral patterns and acquired skills such as speech, handwriting, or keystroke patterns. In contrast, the second category comprises physiological characteristics such as facial features, iris morphology, retinal texture, hand geometry, or fingerprints. Methods based on gait, odor, or genetics using deoxyribonucleic acid (DNA) have limited applications for electronic payment systems. Methods using vascular patterns, palm prints, palm veins, or ear features have not been applied on a wide scale in electronic commerce.

Biometric verification consists of five steps: image acquisition during the registration phase, features extraction, storage, matching, and decision-making. The digital image of the person under examination originates from a sensor in the acquisition device (e.g., a scanner or a microphone). This image is processed to extract a compact profile that should be unique to that person. This profile or signature is then archived in a reference database that can be centralized or distributed according to the architecture of the system. In most cases, registration must be done in person in the presence of an operator to record the necessary biometric template.

The accuracy of a biometric system is typically measured in terms of a false acceptance rate (an impostor is accepted) and false rejection rate (a legitimate user is denied access). These rates are interdependent and are adjusted according to the required level of security. Other measures distinguish between the decision error rates and the matching errors, in the cases that the system allows multiple attempts or includes multiple templates for the same user (Faundez-Zanuy, 2006).
The choice of a particular system depends on several factors, such as the following:

1. The accuracy and reliability of the identification or verification. The result should not be affected by the environment or by aging.
2. Cost of installation, maintenance, and operation.
3. Scale of applicability of the technique; for example, handwriting recognition is not useful for illiterate people.
4. The ease of use.
5. The reproducibility of the results; in general, physiological characteristics are more reproducible than behavioral characteristics.
6. Resistance to counterfeit and attacks.

ISO/IEC 19794 consists of several parts specifying the data formats for biometric data as follows:

1. Part 2: Finger minutiae data
2. Part 3: Finger pattern spectral data
3. Part 4: Finger image data
4. Part 5: Face image data
5. Part 6: Iris image data
6. Part 7: Signature time series data
7. Part 8: Finger pattern skeletal data
8. Part 9: Vascular image data
9. Part 10: Hand geometry silhouette data

Part 11 concerns dynamic data collected during a manual signature and is currently being prepared.

### 3.9.1 Fingerprint Recognition

Fingerprinting is a method of identification and identity verification based on the ridge patterns of the fingerprints. The method is based on the belief that these ridge patterns are unique to each individual and that, in normal circumstances, they remain stable over an individual’s life. In the past, fingerprints were collected by swiping the finger tips in a special ink and then pressing them over a paper to record a negative image. Fingerprint examiners would then look for specific features or minutiae in that image as distinguishing characteristics that identify an individual. Today, fingerprints are captured electronically by placing the finger on a small scanner using optical, electric, thermal, or optoelectronic transducers (Pierson, 2007, pp. 82–87).

Optical transducers are charge-coupled devices (CCDs) that measure the reflections of a light source on the finger and focused through a prism. Electric transducers measure the fluctuations in the capacitance between the user’s fingers and sensors on the surface of a special mouse or home button. Another electric technique measures the changes in the electric field between a resin plate on which the finger rests and the derma with a low-tension alternating current injected into the finger pulps. Thermal techniques rely on a transducer tracking the temperature gradient between the ridges and the minutiae. Optoelectronic methods employ a layer of polymers to record the image of the fingerprint on a polymer layer that converts the image into a proportional electric current. Finally, ultrasound sensors are more capable of penetrating the epidermis to get an image of the pattern underneath it but are not yet ready for mass-market applications.

During the enrollment phase, the image of the user’s fingerprint is recorded and then processed to extract the features that form the reference template that describes the user’s minutiae. The template must consist of stable and reliable indices, insensitive to defects in the image that could have been introduced by dirty fingers, wounds, or deformities. Depending on the application, the template may be stored in a central database or recorded locally on a magnetic card or an integrated circuit card issued to the individual. ISO/IEC 19794-2 (2011) defines three standard data formats for the data elements of the templates.

During verification, fingerprints may be used alone or to supplement another identifier such as a password, a PIN, or a badge. The system then processes the new image to extract the relevant features for the matcher to conduct a one-to-one comparison with the user’s template retrieved from storage.

There are two well-known categories of fingerprint recognition algorithms. Most algorithms are minutiae based and measure a variety of features, such as ridge endings and bifurcations (when two lines split), and compare them to the stored minutiae templates. Image-based methods use other characteristics such as orientation, ridge shape, and text texture. The latter approach is particularly useful when the establishment of a reliable minutiae map is made difficult, for example, because of a bad quality imprint or fingerprint damage due to a high degree of oxidative stress and/or immune activation.

The verification algorithm must be insensitive to potential translations, rotations, and distortions. The degree of similarity between the features extracted from the captured image and those in the stored template is described in terms of an index that varies from 0% to 100% or 1 to 5. The traditional measure was a count of the number of minutiae that corresponded, typically 12–17 (Ratha et al., 2001; Pierson, 2007, pp. 168–169).

NIST and the Federal Bureau of Investigation (FBI) collaborated to produce a large database of fingerprints gathered from crime scenes with their corresponding minutiae to train and evaluate new algorithms for
automatic fingerprint recognition. NIST also provides a Biometric Image Software (NBIS) for biometric processing and analysis, free of charge, and with no licensing requirements (a previous version was called NIST Fingerprint Image Software or NFIS). It includes a fingerprint image quality algorithm, NFIQ, that analyzes that fingerprint image and assigns it a quality value of 1 (highest quality) to 5 (lowest quality).

NIST has conducted several evaluations of fingerprint technologies. In the latest evaluations in 2014, several databases were used from the federal government and law enforcement with different fingerprint combinations varying from a single index up to 10 fingers, impressions on flats, and plain and rolled impressions. (The rolled impressions are obtained during a roll from side to side to capture the full width.) It was found that, for a false-positive identification rate of $10^{-3}$, the false-negative identification rates (i.e., the miss rates) were 1.9% for a single index finger to 0.10 for 10 fingers plain and rolled, that is, more fingers improved accuracy. The most accurate results were achieved with 10 fingers, with searches in datasets of 3 and 5 million subjects (Watson et al., 2014).

There are now several large-scale applications using fingerprints as biometrics. In 2002, the International Civil Aviation Organization (ICAO) adopted the following biometrics for machine-readable travel documents: facial recognition, fingerprint, and iris recognition. The templates are recorded as 2D PDF417 barcodes, or encoded in magnetic stripes, integrated circuits cards with contacts or contactless, or on optical memory. In 2003, the International Labor Organization (ILO) added two finger minutia–based biometric templates to the international identity document of seafarers encoded in the PDF417 stacked code format of ISO/IEC 15438 (International Labor Organization, 2006). In 2009, India launched the Aadhar national identity scheme using, the fingerprints of the 10 digits and iris biometrics to assign each citizen a unique 12-digit identification number (Unique Identification Authority of India, 2009).

Scanned fingerprints are also used in many payment applications. A typical approach used in payment is to use the fingerprint captured by placing the finger on the reader for identification then entering a PIN to authorize the payment. Citibank launched a biometric credit card in Singapore in 2006 using equipment from Pay By Touch. This technology was also used by several grocery chains in the United States and in the United Kingdom. In 2013, Apple incorporated a fingerprint scanner in its iPhone 5S under the brand name Touch ID. The percentage of false rejects in commercial systems reaches about 3%, and the rate of false acceptance is less than one per million.

Fingerprint spoofing has a long tradition but was more difficult with analog techniques. Since 2001, it was shown that automatic finger recognition with digital images is vulnerable to reconstructions of fake fingers made of gelatin (“gummy fingers”) or of silicon rubber filled with conductive carbon (“conductive silicon fingers”). The patterns imprinted on the artificial fingers can be extracted from the negative imprints of real fingers, or from latent fingerprints lifted with a specialized toolkit, digitizing them with a scanner and then using the scanned image to imprint the artificial finger or from high-resolution photos of the true fingerprints (Matsumoto et al., 2002; Maltoni et al., 2003, pp. 286–291; Pierson, 2007, pp. 116–120; Galbally et al., 2010, 2011). A more recent technique is to use minutiae templates to reconstruct a digital image that is similar to the original fingerprint (Galbally et al., 2010). One advantage of thermal and capacitive sensors is that the quality of fake fingerprint images using gummy fingers is low, which could help in identifying spoofing scenarios (Galbally et al., 2011). However, the Chaos Computer Club in Germany announced less than a month after the introduction of Apple’s Touch ID, which uses a capacitive transducer, that the system could be fooled with a fingerprint photographed on a glass surface (Chaos Computer Club, 2013; Musil, 2013).

It is worth noting that these results infirm the assumption made by the ILO in 2006 in allowing visible biometric data that “it is sufficiently difficult to reconstruct from the biometric data that will be stored in the barcode either an actual fingerprint (…) or a fraudulent device that could be used to misrepresent seafarer intent or presence” (International Labor Organization, 2006, § 5.1, p. 9).

It should be noted that some subjects do not have fingerprints or have badly damaged fingerprints.

Finally, it should be noted that the individuality of fingerprints remains an assumption. Most human experts and automatic fingerprint recognition algorithms declare that fingerprints have a common source if they are “sufficiently” similar in a given representation scheme based on a similarity metric (Maltoni et al., 2003, chapter 8). This has important implications in forensic investigation but probably will be less significant in payment applications.

### 3.9.2 Iris Recognition

The iris is the colored area between the white of the eye and the pupil, with a texture that is an individual characteristic that remains constant for many years. The digital image of the iris texture can be encoded as an IrisCode and used for the identification of individuals with accuracy and an error probability of the order of 1 in 1.2 million. It is even possible to distinguish among identical twins and to separate the two irises of
the same person (Flom and Safir, 1987; Daugman, 1994, 1999; Wildes, 1997). The inspection is less invasive than a retinal scan.

This technique was developed by John Daugman and was patented by Iridian Technologies—previously known as IriScan, Inc, a company formed by two ophthalmologists and a computer scientist. This patent expired in 2004 in the United States and in 2007 in the rest of world, leaving the field open to new entrants. Since 2006, it is a division of Viisage Technology. A major supplier of the technology is currently the Lithuanian company Neurotechnology.

During image acquisition, the person merely faces a camera connected to computer at a distance of about 1 m. Iris scanning software can also be downloaded to mobile phones and smartphones. Some precautions need to be respected during image capture, particularly to avoid reflections by ensuring uniform lighting. Contact lenses produce a regular structure in the processed image.

Image acquisition can be accomplished in special kiosks or from desktop cameras. The latter are cheaper because they were proven difficult for some users that they must look into a hole to locate an illuminated ring. ISO/IEC 19794-6 (2011) defines the format of the captured data.

Iris recognition is typically used as a secondary identifier in addition to fingerprint imaging. Other potential applications include the identification of users of automatic bank teller machines, the control of access either to a physical building or equipment or to network resources. It is being evaluated to increase the speed of passenger processing at airports. As mentioned earlier, India is collecting iris images together with the 10 fingerprints and an image of the face for its new national identity scheme (Unique Identification Authority of India, 2009).

It was shown, however, that the accuracy of the current generation of commercial handheld iris recognition devices degrades significantly in tactical environments (Faddis et al., 2013). Furthermore, it was demonstrated that the templates or IrisCodes have sufficient information to allow the reconstruction of synthetic images that closely match the images collected from real subjects and then deceive commercial iris recognition systems. The success rate of the synthetic image ranges from 75% to 95%, depending on the level set for the false acceptance rate. The attack is also successful that even the reconstructed image of the iris does not use the stored templates that the system used for recognition (Galbally et al., 2012). This creates the possibility of stealing someone’s identity through their iris image.

### 3.9.3 Face Recognition

Police and border control agencies have been using facial recognition to scan passports. This has been done on the basis of a template that encodes information derived from proprietary mathematical representations of features in the face image, such as the distance between the eyes, the gap between the nostrils, the dimension of the mouth. U.S. government initiatives, starting with the Face Recognition Technology (FERET) program in September 1993, have provided the necessary stimulus to improve the speed and the accuracy of the technology. ISO/IEC 19794-5 (2011) defines the format for the captured data.

Some consumer applications, such as Apple’s iPhoto or Google’s Picasa, have included facial recognition to identify previously defined faces in photo albums. Another consumer application is Churchix (http://churchix.com, last accessed January 26, 2016), designed to help church administrators and event managers track their members attendance by comparison with a database of reference photos. Nowadays, almost all mobile phones have a photographic camera and newer programs that allow people to take a picture of a person and then search the Internet for possible matches to that face. Polar Rose, a company from Malmö, Sweden, which produced such software, was acquired by Apple in 2010. There are also tools to help search for pictures on Facebook. Other applications allow users to identify a person in a photo and then search the web for other pictures in which that person appears (Palmer, 2010).

Facial recognition, however, is not very accurate in less controlled situations. The accuracy is sensitive to accessories such as sunglasses, beards or mustaches, grins, or head tilts (yaw) by 15°. It can be combined with other biometrics such as fingerprints (Yang, 2010; de Oliveira and Motta 2011) or keystrokes (Popovic et al., 2014) to increase the overall system accuracy.

An international contest for face verification algorithms was organized in conjunction with the 2004 International Conference on Biometric Authentication (ICBA). The National Institute of Standards and Technology (NIST) has tracked the progress of face recognition technologies in a series of benchmark evaluations of prototype algorithms from universities and the research industrial laboratories in 1997, 2002, 2006, 2010, and 2013. In recent years, many participants were major commercial suppliers from the United States, Japan, Germany, France, Lithuania, and Chinese academics. The 2010 results showed that an algorithm from NEC was the most accurate with chances of identifying an unknown subject from a database of 1.6 million records about 92%, the search lasting about 0.4 second. The search duration, however, increased linearly with the size of the enrolled population, while the speed of less accurate algorithms increased more slowly with the size of the database (Grother et al., 2010). In general, the differences between the best and worst performances can be significant.
The National Police Academy in Japan followed NIST study with a focused investigation of NEC algorithm to evaluate the influence of several factors such as (1) the effect of age, due to the time lapse between the recorded picture and the subject; (2) the shooting angle, that is, the yaw; (3) the effect of face expression; and (4) of accessories such as a cap, sunglasses, beards, and mustaches. The results confirmed that from 2001 to 2010, the top of the line face recognition algorithms improved significantly. The overall recognition rate was 95% (compared to 48% in 2001). In particular, 98% of the images without glasses were classified correctly (compared to 73% in 2001). Similar improvements were observed for the effects of the yaw (provided that the angle is less than 30°) of the expression, and of the various accessories, such as spectacles (Horiuchi and Hada, 2013). It should be noted that the results were obtained with pictures taken under good lighting conditions, which is not always the case for police investigations, which often start with low-quality images, taken at bad angles and under poor lights and/or with small face sizes.

NIST (2013) study used reasonable quality law enforcement mug shot images, poor-quality web cam images collected in similar detention operations and moderate quality visa application images. The range thus covered high quality in identity documents (passports, visa applications, driver licenses) to poorly controlled surveillance situations. The 2013 study confirmed that NEC had the most accurate set of algorithms, with an error rate of less than half of the second place algorithm. It was followed by those supplied by Morpho (a Safran company), which merged its algorithms with those acquired from L1 Identity Solutions in 2011. Other leading suppliers are Toshiba, Cognitec Systems, and 3M/Cogent. Algorithms with lesser accuracy (error rate higher than 12%) are those from Neurotechnology, Zuhai Yisheng, HP, Decatur, and Ayonix. Some Chinese universities provided high-accuracy algorithms as well. The ranking of performance across algorithms, however, must be weighed by application-specific requirements; some algorithms are more suited to recognition of difficult webcam images (Grother and Ngan, 2014).

3.9.4 Voice Recognition

Depending on the context, voice recognition systems have one of two functions:

1. **Speaker identification**: This is a one-to-many comparison to establish the identity of the speaker. A digital vocal print, newly acquired from the end user, is compared with a set of stored acoustic references to determine the person from its utterances.

2. **Speaker verification**: This case consists in verifying that the voice imprint matches the acoustic references of the person that the speaker pretends to be.

The size of the voice template that characterizes an individual varies depending on the compression algorithm and the duration of the record.

Speaker verification is implemented in payment systems as follows. The voice template that characterizes a subject is extracted during registration through the pronunciation of one or several passwords or passphrases. During the verification phase, the speaker utters one of these passwords back. The system then attempts to match the features extracted from new voice sample with the previously recorded voice templates. After identity verification, the person is allowed access.

A bad sound quality can cause failures. In remote applications, this quality depends on several factors such as the type of the telephone handset, ambient noise (particularly in the case of hands-free telephony), and the type of connection (wire line or wireless). The voice quality is also affected by a person’s health, stress, emotions, and so on.

Some actors have the capability of mimicking other voices. Furthermore, there are commercial personalized text-to-speech systems that produce voice automation prompts using familiar voices (Burg, 2014). Speech synthesis algorithms require about 10–40 hours of professionally recorded material with the voice to be mimicked. Using archival recordings, they could also bring voices back from the dead. An easier method to defraud the system would be playback recordings of authentic commands. This is why automatic speaker recognition systems must be supplemented with other means of identification.

3.9.5 Signature Recognition

Handwritten signatures have long been established as the most widespread means of identity verification particularly for administrative and financial institutions. They are a behavioral biometrics that changes over time and is affected by many physical and emotional conditions. ISO/IEC 19794-7 specifies two formats for the time series describing the captured signatures. One format is for general use and the other is more compact for use with smart cards.

Systems for automatic signature recognition rely on the so-called permanent characteristics of an individual handwriting to match the acquired signature with a prerecorded sample of the handwriting of the person whose identity is to be verified. It goes without saying that handwritten signatures do not work in areas with high illiteracy and are not persistent before 16 years of age (Blanco-Gonzalo et al., 2013).
Signature recognition can be static or dynamic. In static verification, also called offline verification, the signature is compared with an archived image of the signature of the person to be authenticated. For dynamic signature recognition, also called online verification, signing can be with a stylus on a digitizing tablet or the fingertip on a pad connected to a computer, or on the screen of a tablet or a mobile phone. The algorithms are based on techniques such as dynamic time warping to compare two time series even with their different timescales. The parameters under consideration are derived from the written text and various movement descriptors such as the pressure exercised on the pad or the screen, the speed and direction of the movement, the accelerations and decelerations, the angles of the letters, and the azimuth and the altitude of the pen with respect to the plan of the pad. The static approach is more susceptible to forgeries because the programs have yet to include all the heuristics that graphologists take into consideration based on their experience. There are enormous differences in the way people from different countries sign, and there are specific approaches for non-Latin scripts (Impedovo and Pirlo, 2008).

Even though users prefer stylus-based devices to finger-based systems, an initial evaluation showed their performances are comparable and even that finger signing on an Apple iPad2 can exceed that of stylus-based devices, but there are no obvious explanations (Blanco-Gonzalo et al., 2013).

### 3.9.6 Keystroke Recognition

Keystroke recognition is a technique based on an individual's typing style in terms of rhythm, speed, duration and pressure of keystrokes, and so on. It is based on the premise that each person's typing style is distinct enough to verify a claimed identity. Keystroke measures are based on several repetitions of a known sequence of characters (e.g., the login and the password) (Obaidat and Sadoun, 1999; Dowland et al., 2002).

There are no standards for the data collected, and the user profile is based on a small dataset, which could result in high error rates. Typically, the samples are collected using structured texts at the initiation of a session to complement a classic login approach. The keystrokes are monitored unobtrusively as the person is keying in information. This approach, however, does not protect against session hijacking, that is, seizing control of the session of a legitimate user after a valid access. There is an active research to include continuous verification of the user's identity using free text, that is, what users type during their normal interaction with a computing device (Ahmed and Traore, 2014; Popovici et al., 2014). Another possibility to reduce error rate and to avoid session hijack is to combine keystroke dynamics with face recognition (Giot et al., 2010).

One important security risk of keystroke monitoring is that any software positioned at an input device can also leak sensitive information, such as passwords (Shah et al., 2006).

### 3.9.7 Hand Geometry

In the past several years, hand geometry recognition has been used in large-scale commercial applications to control access to enterprises, customs, hospitals, military bases, prisons, and so on.

The principle is that some features related to the shape of the human hand, such as the length of the fingers, are relatively stable and peculiar to an individual. The image acquisition requires the subject's cooperation. The user positions the hand with outstretched fingers flat on a plate facing the lens of a digital camera. The fingers are spread, resting against guiding pins soldered on the plate. This plate is surrounded by mirrors on three sides to capture the frontal and side-view images of the hand. The time for taking one picture is about 1.2 seconds. Several pictures (three to five) are taken, and the average is stored in memory as a reference to the individual. One commercial implementation uses a 3D model with 90 input parameters to describe the hand geometry with a template of 9 octets.

### 3.9.8 Retinal Recognition

The retina is a special tissue of the eye that responds to light pulses by generating proportional electrical discharges to the optical nerve. It is supplied by a network of blood vessels according to a configuration that is characteristic of each individual and that is stable throughout life. The retina can even distinguish among twins. A retinal map can be drawn by recording the reflections of light pulses by generating proportional electrical discharges to the optical nerve. It is supplied by a network of blood vessels according to a configuration that is characteristic of each individual and that is stable throughout life. The retina can even distinguish among twins. A retinal map can be drawn by recording the reflections of low-intensity infrared beam with the help of a charge-coupled device (CCD) to form a descriptor of 35 octets.

The equipment used is relatively large and costly, and image acquisition requires the cooperation of the subject. It entails looking through an eyepiece and concentration on an object while a low-power laser beam is injected into the eye. As a consequence, the technique is restricted to access control to high-security areas: military installations, nuclear plants, high-security prisons, bank vaults, and network operation centers. Currently, this technique is not suitable for remote payment systems or for large-scale deployment.

### 3.9.9 Additional Standards

FIPS 201 (National Institute of Standards and Technology, 2013) defines the procedures for personal identity verification for federal employees and contractors, including
prot of identity, registration, identity card issuance and reissuance, chains of trust, and identity card usage. The biometrics used are fingerprints, iris images, and facial images. NIST Special Publication (SP) 800-76-2 contains the technical specifications for the biometric data used in the identification cards. In particular, the biometric data use the Common Biometric Exchange File Format (CBEFF) of ANSI INCITS 398 (Podio et al., 2014).

ISO/IEC 7816-11 (2004) specifies commands and data objects used in identity verification with a biometric means stored in integrated circuit cards (smart cards). ITU-T Recommendation X.1084 (2008) defines the Handshake protocol between the client (card reader or the terminal) and the verifier, which is an extension of TLS.

ANSI INCITS 358, published in 2002 and revised in 2013, contains BioAPI 2.0, a standard interface that allows a software application to communicate with and utilize the services of one or more biometric technologies. (ISO/IEC 19784-1 [2006] is the corresponding international standard). ITU-T Recommendation X.1083 (ISO/IEC 24708) specifies the BioAPI Interworking Protocol (BIP), that is, the syntax, semantics, and encodings of set of messages (“BIP messages”) that enable a BioAPI-conforming application to request biometric operations in BioAPI-conforming biometric service providers. It also specifies extensions to the architecture and the BioAPI framework (specified in ISO/IEC 19784-1) that support the creation, processing, sending, and reception of BIP messages.

BioAPI 2.0 is the culmination of several U.S. initiatives starting in 1995 with the Biometric Consortium (BC). The BC was charted to be the focal point for the U.S. government on research, development, testing, evaluation, and application of biometric-based systems for personal identification/verification. In parallel, the U.S. Department of Defense (DOD) initiated a program to develop a Human Authentication–Application Program Interface (HA–API). Both activities were merged, and version 1.0 of the BioAPI was published in March 2000. The BioAPI specification was also the basis of the ANSI 358-2002[R2012], first published in 2002 and revised in 2007 and 2013.

ANSI X9.84 (2010) describes the security framework for using biometrics for the authentication of individuals in financial services. The scope of the specification is the management of the biometric data throughout the life cycle: collection, storage, retrieval, and destruction. Finally, the Fast Identity Online Alliance (FIDO) was established in July 2012 to develop technical specifications that define scalable and interoperable mechanisms to authenticate users by hand gesture, a biometric identifier, or pressing a button.

3.9.10 Summary and Evaluation

There is a distinction between the flawless image of biometric technology presented by promoters and futuristic authors and the actual performance in terms of robustness and reliability. For example, the hypothesis that biometric traits are individual and that they are sufficiently stable in the long term is just an empirical observation. Systems that perform well in many applications have yet to scale up to cases involving tens of millions of users. False match rates, which may be acceptable for specific tasks in small-scale applications, can be too costly or even dangerous in applications designed to provide high level of security. For some techniques, the biometrics images obtained during data acquisition vary significantly from one session to another.

Attacks on biometric systems can be grouped into two main categories: those that are common to all information systems and those that are specific to biometrics. The first category of attacks comprises the following:

1. Attacks on the communication channels to intercept the exchanges, for example, to snoop on the user image of the biometrics in question or the features extracted from that image. The stolen information can be replayed as if it came from a legitimate user.
2. Attacks of the various hardware and software modules, such as physical tampering with the sensors or inserting malware at the feature extraction module or the matcher module.
3. Attack on the database where the templates are stored.

Attacks specific to biometrics constitute the second category. The automatic recognition system is deceived with the use of fake fingers or reconstructed iris patterns.

In total, there are eight places in a generic biometric system where attacks may occur (Ratha et al., 2001), these are identified in the following list:

1. Fake biometrics are presented to the sensor such as a fake finger, a copy of a signature, or a face mask.
2. Replay attacks to bypass the sensor with previously digitized biometric signals.
3. Override of the feature extraction process with a malware attack to allow the attacker to control the features to be processed.
4. A more difficult attack involves tampering with the template extracted from the input signal before it reaches the matcher. This is the case if they are not collocated and are connected over a long-haul network. Another condition for this attack to succeed is that the method for coding the features into a template is known.
5. An attack on the matcher so that it produces preselected results.
3.10 Authentication of the Participants

The purpose of authentication of participants is to reduce, if not eliminate, the risk that intruders might masquerade under legitimate appearances to pursue unauthorized operations.

As stated previously, when the participants utilize a symmetric encryption algorithm, they are the only ones who share a secret key. As a consequence, the utilization of this algorithm, in theory, guarantees the confidentiality of the messages, the correct identification of the correspondents, and their authentication. The key distribution servers also act as authentication servers (ASs), and the good functioning of the system depends on the capability of all participants to protect the encryption key.

In contrast, when the participants utilize a public key algorithm, a user is considered authentic when that user can prove that he or she holds the private key that corresponds with the public key that is attributed to the user. A certificate issued by a certification authority indicates that it certifies the association of the public key (and therefore the corresponding private key) with the recognized identity. In this manner, identification and authentication proceed in two different ways, the identity with the digital signature and the authentication with a certificate. Without such a guarantee, a hostile user could create a pair of private/public keys and then distribute the public key as if it was that of the legitimate user.

Although the same public key of a participant could equally serve to encrypt the message that is addressed to that participant (confidentiality service) and to verify the electronic signature of the documents that the participant transmits (integrity and identification services), in practice, a different public key is used for each set of services.

According to the authentication framework defined by ITU-T Recommendations X.500 (2001) and X.811 (1995), simple authentication may be achieved by one of several means:

1. Name and password in the clear
2. Name, password, and a random number or a time stamp, with integrity verification through a hash function
3. Name, password, a random number, and a time stamp, with integrity verification using a hash function

Strong authentication requires a certification infrastructure that includes the following entities:

1. Certification authorities to back the users’ public keys with “sealed” certificates (i.e., signed with the private key of the certification authority) after verification of the physical identity of the owner of each public key.
2. A database of authentication data (directory) that contains all the data relative to the private encryption keys, such as their value, the
Protocols for Secure Electronic Commerce

duration of validity, and the identity of the owners. Any user should be able to query such database to obtain the public key of the correspondent or to verify the validity of the certificate that the correspondent would present.

3. A naming or registering authority that may be distinct from the certification authority. Its principal role is to define and assign unique distinguished names to the different participants.

The certificate guarantees the correspondence between a given public key and the entity, the unique distinguished name of which is contained in the certificate. This certificate is sealed with the private key of the certification authority. When the certificate owner signs documents with the private signature key, the partners can verify the validity of the signature with the help of the corresponding public key that is contained in the certificate. Similarly, to send a confidential message to a certified entity, it is sufficient to query the directory for the public key of that entity and then use that key to encrypt messages that only the holder of the associated private key would be able to decipher.

3.11 Access Control

Access control is the process by which only authorized entities are allowed access to the resources as defined in the access control policy. It is used to counter the threat of unauthorized operations such as unauthorized use, disclosure, modification, destruction of protected data, or denial of service to legitimate users. ITU-T Recommendation X.812 (1995) defines the framework for access control in open networks. Accordingly, access control can be exercised with the help of a supporting authentication mechanism at one or more of the following layers: the network layer, the transport layer, or the application layer. Depending on the layer, the corresponding authentication credentials may be X.509 certificates, Kerberos tickets, simple identity, and password pairs.

There are two types of access control mechanisms: identity based and role based. Identity-based access control uses the authenticated identity of an entity to determine and enforce its access rights. In contrast, for role-based access control, access privileges depend on the job function and its context. Thus, additional factors may be considered in the definition of the access policy, for example, the strength of the encryption algorithm, the type of operation requested, or the time of day. Role-based access control provides an indirect means of bestowal of privileges through three distinct phases: the definition of roles, the assignment of privileges to roles, and the distribution of roles among users. This facilitates the maintenance of access control policies because it is sufficient to change the definition of roles to allow global updates without revising the distribution from top to bottom.

At the network layer, access control in IP networks is based on packet filtering using the protocol information in the packet header, specifically the source and destination IP addresses and the source and destination port numbers. Access control is achieved through “line interruption” by a certified intermediary or a firewall that intercepts and examines all exchanges before allowing them to proceed. The intermediary is thus located between the client and the server, as indicated in Figure 3.9. Furthermore, the firewall can be charged with other security services, such as encrypting the traffic for confidentiality at the network level or integrity verification using digital signatures. It can also inspect incoming and outgoing exchanges before forwarding them to enforce the security policies of a given administrative domain. However, the intervention of the trusted third party must be transparent to the client.

The success of packet filtering is vulnerable to packet spoofing if the address information is not protected and if individual packets are treated independently of the other packets of the same flow. As a remedy, the firewall can include a proxy server or an application-level gateway that implements a subset of application-specific functions. The proxy is capable of inspecting all packets in light of previous exchanges of the same flow before allowing their passage in accordance with
the security policy in place. Thus, by filtering incoming and outgoing electronic mail, file transfers, exchanges of web applications, and so on, application gateways can block nonauthorized operations and protect against malicious codes such as viruses. This is called a stateful inspection. The filter uses a list of keywords, the size and nature of the attachments, the message text, and so on. Configuring the gateway is a delicate undertaking because the intervention of the gateway should not prevent daily operation.

A third approach is to centralize the management of the access control for a large number of clients and users with different privileges with a dedicated server. Several protocols have been defined to regulate the exchanges among network elements and Access Control Servers. RFC 6929 (2013) specifies Remote Authentication Dial in User Service (RADIUS) for client authentication, authorization, and for collecting accounting information of the calls. In RFC 1492 (1993), Cisco described a protocol called Terminal Access Controller Access System (TACACS), which was later updated in TACACS+. Both RADIUS and TACACS+ require a secret key between each network element and the server. Depicted in Figure 3.10 is the operation of RADIUS in terms of a client/server architecture. The RADIUS client resides within the Access Control Server while server relies on an X.509 directory through the protocol Lightweight Directory Access Protocol (LDAP). Both X.509 and LDAP will be presented later in this chapter.

Note that both server-to-client authentication and user-to-client authentication are outside the scope of RADIUS. Also, because RADIUS does not include provisions for congestion control, large networks may suffer degraded performance and data loss. It should be noted that there are some known vulnerabilities in RADIUS or in its implementations (Hill, 2001).

Commercial systems implement two basic approaches for end-user authentication: one-time password and challenge response (Forrester et al., 1998). In a typical one-time password system, each user has a device that generates a number periodically (usually every minute) using the current time, the card serial number, and a secret key held in the device. The generated number is the user’s one-time password. This procedure requires that the time reference of the Access Control Server be synchronized with the card so that the server can regenerate an identical number.

In challenge–response systems, the user enters a personal identification number to activate handheld authenticators (HHA) and then to initiate a connection to an Access Control Server. The Access Control Server, in turn, provides the user with a random number (a challenge), and the user enters this number into a handheld device to generate a unique response. This response depends on both the challenge and some secret keys shared between the user’s device and the server. It is returned to the Access Control Server to compare with the expected response and decide accordingly.

Two-factor authentication is also used in many services, particularly in mobile commerce domain. The site to be accessed sends a text to the user that is requesting access with a new code and a password each time the user attempts to login.

3.12 Denial of Service

Denial-of-service attacks prevent normal network usage by blocking the access of legitimate users to the network resources they are entitled to, by overwhelming...
the hosts with additional or superfluous tasks to prevent them from responding to legitimate requests or to slow their response time below satisfactory limits.

In a sense, denial of service results from the failure of access control. Nevertheless, these attacks are inherently associated with IP networks for two reasons: network control data and user data share the same physical and logical bandwidths; and IP is a connectionless protocol where the concept of admission control does not apply. As a consequence, when the network size exceeds a few hundred nodes, network control traffic (due, e.g., to the exchange of routing tables) may, under some circumstances, occupy a significant portion of the available bandwidth. Further, inopportune or ill-intentioned user packets may able to bring down a network element (e.g., a router), thereby affecting not only all endpoints that rely on this network element for connectivity but also all other network elements that depend on it to update their view of the network status. Finally, in distributed denial-of-service (DDOS) attacks, a sufficient number of compromised hosts may send useless packets toward a victim around the same time, thereby affecting the victim’s resources or bandwidth or both (Moore et al., 2001; Chang, 2002).

As a point of comparison, the public switched telephone network uses an architecture called common channel signaling (CCS) whereby user data and network control data travel on totally separate networks and facilities. It is worth noting that CCS was introduced to protect against fraud. In the old architecture, called Channel-Associated Signaling (CAS), the network data and the user data used separate logical channels on the same physical support.

Denial-of-service attacks work in two principal ways: forcing a protocol state machine a deadlock situation and overwhelming the processing capacity of the receiving station.

One of the classical examples of protocol deadlocks is the SYN flooding attack that perturbs the functioning of the TCP protocol (Schuba et al., 1997). The handshake in TCP is a three-way exchange: a connection request with the SYN packet, an acknowledgment of that request with SYN/ACK packet, and finally, a confirmation from the requesting client to solve a puzzle in the form of simple cryptographic problems before the allocated resources needed to establish a connection. To avoid replay attacks, these problems are formulated using the current time, a server secret, and additional information from the client request (Juels and Brainard, 1999). This approach, however, requires programs for solving puzzles specific to each application that are incorporated in the client browser.

A similar approach is to require the sender to do some hash calculation before sending a message to make spamming uneconomic in the form of a proof of work (Warren, 2012). The difficulty of the proof of work is made proportional to the size of the message, and each message is time-stamped to protect the network against denial-of-service attacks by flooding old messages.
The aforementioned solutions require a complete overhaul of the Internet architecture. Yet, in their absence, electronic commerce is vulnerable to interferences from botnets that can be thwarted with ad hoc solutions only. A botnet is a virtual network constituted by millions of terminals infected with a specialized virus or worm called a bot. The bot does not destroy data on the infected terminal nor does it affect any of the typical services; all what it does is to give the bot master remote control. Each client of botnet can be instructed to send messages surreptitiously in spamming campaigns or in distributed denial-of-service attacks. The bot master is also responsible for the bot maintenance: fixing errors, ensuring that the bot remains undetected, and changing the IP address of the master server periodically to prevent tracing. Once all is in place, the bot master offers the botnet services in an online auction to the highest bidder, which can be a political opponent or a commercial competitor of the website under attack.

3.13 Nonrepudiation

Nonrepudiation is a service that prevents a person who has accomplished an act from denying it later, in part or as a whole. Nonrepudiation is a legal concept to be defined through legislation. The role of informatics is to supply the necessary technical means to support the service offer according to the law. The building blocks of nonrepudiation include the electronic signature of documents, the intervention of a third party as a witness, time stamping, and sequence numbers. Among the mechanisms for nonrepudiation are a security token sealed with the secret key of the verifier that accompanies the transaction record, time stamping, and sequence numbers. Depending on the system design, the security token sealed with the verifier’s secret key can be stored in a tamper-resistant cryptographic module. The generation and verification of the evidence often require the intervention of one or more entities external to parties to the transaction such as a notary, a verifier, and an adjudicator of disputes.

ITU-T Recommendation X.813 (1996) defines a general framework for nonrepudiation in open systems. Accordingly, the service comprises the following measures:

- Generation of the evidence
- Recording of the evidence
- Verification of the evidence generated
- Retrieval and reverification of the evidence

There are two types of nonrepudiation services:

1. **Nonrepudiation at the origin.** This service protects the receiver by preventing the sender from denying having sent the message.

2. **Nonrepudiation at the destination.** This service plays the inverse role of the preceding function. It protects the sender by demonstrating that the addressee has received the message.

Threats to nonrepudiation include compromise of keys or unauthorized modification or destruction of evidence. In public key cryptography, each user is the sole and unique owner of the private key. Thus, unless the whole system has been penetrated, a given user cannot repudiate the messages that are accompanied by his or her electronic signature. In contrast, nonrepudiation is not readily achieved in systems that use symmetric cryptography. A user can deny having sent the message by alleging that the receiver has compromised the shared secret or that the key distribution server has been successfully attacked. A trusted third party would have to verify each transaction to be able to testify in cases of contention.

Nonrepudiation at the destination can be obtained using the same mechanisms but in the reverse direction.

3.13.1 Time Stamping and Sequence Numbers

Time stamping of messages establishes a link between each message and the date of its transmission. This permits the tracing of exchanges and prevents attacks by replaying old messages. If clock synchronization of both parties is difficult, a trusted third party can intervene as a notary and use its own clock as reference.

The intervention of the “notary” can be in either of the following mode:

- Offline to fulfill functions such as certification, key distribution, and verification if required, without intervening in the transaction.
- Online as an intermediary in the exchanges or as an observer collecting the proofs that might be required to resolve contentions. This is a similar role to that of a trusted third party of the network layer (firewall) or at the application layer (proxy) but with a different set of responsibilities.

Let’s assume that a trusted third party combines the functions of the notary, the verifier, and the adjudicator. Each entity encrypts its messages with the secret key that has been established with the trusted third party
before sending the message. The trusted third party decrypts the message with the help of this shared secret with the intervening party, time-stamps it, and then reencrypts it with the key shared with the other party. This approach requires the establishment of a secret key between each entity and the trusted third party that acts as a delivery messenger. Notice, however, that the time-stamping procedures have not been normalized and each system has its own protocol.

Detection of duplication, replay, as well as the addition, suppression, or loss of messages is achieved with the use of a sequence number before encryption. Another mechanism is to add a random number to the message before encryption. All these means give the addressee the ability to verify that the exchanges genuinely took place during the time interval that the time stamp defines.

### 3.14 Secure Management of Cryptographic Keys

Key management is a process that continues throughout the life cycle of the keys to thwart unauthorized disclosures, modifications, substitutions, reuse of revoked or expired keys, or unauthorized utilization. Security at this level is a recursive problem because the same security properties that are required in the cryptographic system must be satisfied in turn by the key management system.

The secure management of cryptographic keys relates to key production, storage, distribution, utilization, withdrawal from circulation, deletion, and archiving (Fumer and Landrock, 1993). These aspects are crucial to the security of any cryptographic system. SP 800-57 (National Institute of Standards and Technology, 2012c) is a three-part recommendation from NIST that provides guidance on the management. Each part is tailored to a specific audience: Part 1 is for system developers and system administrators, Part 2 is aimed at system or application owners, while Part 3 is more general and targets system installers, end users, as well as people making purchasing decisions.

#### 3.14.1 Production and Storage

Key production must be done in a random manner and at regular intervals depending on the degree of security required.

Protection of the stored keys has a physical aspect and a logical aspect. Physical protection consists of storing the keys in safes or in secured buildings with controlled access, whereas logical protection is achieved with encryption.

In the case of symmetric encryption algorithms, only the secret key is stored. For public key algorithms, storage encompasses the user’s private and public keys, the user’s certificate, and a copy of the public key of the certification authority. The certificates and the keys may be stored on the hard disk of the certification authority, but there is some risk of possible attacks or of loss due to hardware failure. In cases of microprocessor cards, the information related to security, such as the certificate and the keys, are inserted during card personalization. Access to this information is then controlled with a confidential code.

#### 3.14.2 Distribution

The security policy defines the manner in which keys are distributed to entitled entities. Manual distribution by mail or special dispatch (sealed envelopes, tamper-resistant module) is a slow and costly operation that should only be used for the distribution of the root key of the system. This is the key that the key distributor utilizes to send to each participant their keys.

An automatic key distribution system must satisfy all the following criteria of security:

- Confidentiality
- Identification of the participant
- Data integrity: by giving a proof that the key has not been altered during transmission or that it was not replaced by a fake key
- Authentication of the participants
- Nonrepudiation

Automatic distribution can be either point to point or point to multipoint. The Diffie–Hellman key exchange method (Diffie and Hellman, 1976) allows the two partners to construct a master key with elements that have been previously exchanged in the clear. A symmetric session key is formed next on the basis of the data encrypted with this master key or with a key derived from it and exchanged during the identification phase. IKE is a common automated key management mechanism designed specifically for use with IPsec.

To distribute keys to several customers, an authentication server can also play the role of a trusted third party and distribute the secret keys to the different parties. These keys will be used to protect the confidentiality of the messages carrying the information on the key pairs.

#### 3.14.3 Utilization, Withdrawal, and Replacement

The unauthorized duplication of a legitimate key is a threat to the security of key distribution. To prevent this type of attack, a unique parameter can be concatenated
to the key, such as a time stamp or a sequence number that increases monotonically (up to a certain modulo).

The risk that a key is compromised increases proportionately with time and with usage. Therefore, keys have to be replaced regularly without causing service interruption. A common solution that does not impose a significant load is to distribute the session keys on the same communication channels used for user data. For example, in the SSL/TLS protocol, the initial exchanges provide the necessary elements to form keys that would be valid throughout the session at hand. These elements flow encrypted with a secondary key, called a key encryption key, to keep their confidentiality.

Key distribution services have the authority to revoke a key before its date of expiration after a key loss or because of the user’s misbehavior.

### 3.14.4 Key Revocation

If a user loses the right to employ a private key, if this key is accidentally revealed, or, more seriously, if the private key of a certification authority has been broken, all the associated certificates must be revoked without delay. Furthermore, these revocations have to be communicated to all the verifying entities in the shortest possible time. Similarly, the use of the revoked key by a hostile user should not be allowed. Nevertheless, the user will not be able to repudiate all the documents already signed and sent before the revocation of the key pair.

### 3.14.5 Deletion, Backup, and Archiving

Key deletion implies the destruction of all memory registers and magnetic or optical media that contain either the key or the elements needed for its reconstruction.

Backup applies only to encryption keys and not to signature keys; otherwise, the entire structure for nonrepudiation would be put into question.

The keys utilized for nonrepudiation services must be preserved in secure archives to accommodate legal delays that may extend for up to 30 years. These keys must be easily recoverable in case of need, for example, in response to a court order. This means that the storage applications must include mechanisms to prevent unrecoverable errors from affecting the ciphertext.

### 3.14.6 A Comparison between Symmetric and Public Key Cryptography

Systems based on symmetric key algorithms pose the problem of ensuring the confidentiality of key distribution. This translates into the use of a separate secure distribution channel that is preestablished between the participants. Furthermore, each entity must have as many keys as the number of participants with whom it will enter into contact. Clearly, the management of symmetric keys increases exponentially with the number of participants.

Public key algorithms avoid such difficulties because each entity owns only one pair of private and public keys. Unfortunately, the computations for public key procedures are more intense than those for symmetric cryptography. The use of public key cryptography to ensure confidentiality is only possible when the messages are short, even though data compression before encryption with the public key often succeeds in speeding up the computations. Thus, public key cryptography can complement symmetric cryptography to ensure the safe distribution of the secret key, particularly when safer means such as direct encounter of the participants, or the intervention of a trusted third party, are not feasible. Thus, a new symmetric key could be distributed at the start of each new session and, in extreme cases, at the start of each new exchange.

### 3.15 Exchange of Secret Keys: Kerberos

Kerberos is the most widely known system for the automatic exchange of keys using symmetric encryption. Its name is from the three-headed dogs that, according to Greek mythology, guarded the gates of Hell.

Kerberos comprises the services of online identification and authentication as well as access control using symmetric cryptography (Neuman and Ts’o, 1994). It allows management access to the resources of open network from nonsecure machines such as the management of student access to the resources of a university computing center (files, printers, etc.). Kerberos has been the default authentication option since Windows 2000. Since Windows Vista and Windows Server 2008, Microsoft’s implementation of the Kerberos authentication protocol enables the use of Advanced Encryption Standard (AES) 128 and AES 256 encryption with the Kerberos authentication protocol.

The development of Kerberos started in 1978 within the Athena project at the Massachusetts Institute of Technology (MIT), financed by the Digital Equipment Corporation (DEC) and IBM. Version 5 of the Kerberos protocol was published in 1994 and remains in use. RFC 4120 (2005) provides an overview and the protocol specifications. Release 1.13.12 is the latest edition and was published in May 2015.

The system is built around a Kerberos key distribution center that enjoys the total trust of all participants.
with whom they all have already established symmetric encryption keys. Symmetric keys are attributed to individual users for each of their account when they register in person. Initially, the algorithm used for symmetric encryption is the Data Encryption Standard (DES), but AES was later added in 2005 as per RFC 3962. Finally, in 2014, DES and other weak cryptographic algorithms were deprecated (RFC 6649, 2012).

The key distribution center consists of an authentication server (AS) and a ticket-granting server (TGS). The AS controls access to the TGS, which in turn, controls access to specific resources. Every server shares a secret key with every other server. Finally, during the registration of the users in person, a secret key is established with the AS for each user’s account. With this arrangement, a client has access to multiple resources during a session with one successful authentication, instead of repeating the authentication process for each resource. The operation is explained as follows.

After identifying the end user with the help of a login and password pair, the authentication server (AS) sends the client a session symmetric encryption key to encrypt data exchanges between the client and the TGS. The session key is encrypted with the symmetric encryption key shared between the user and the authentication server. The key is also contained in the session ticket that is encrypted with the key preestablished between the TGS and the AS.

The session ticket, also called a ticket-granting ticket, is valid for a short period, typically a few hours. During this time period, it can be used to request access to a specific service; which is why it is also called an initial ticket.

The client presents the TGS with two items of identification: the session ticket and an authentication title that are encrypted with the session key. The TGS compares the data in both items to verify the client authenticity and its access privileges before granting access to the specific server requested.

Figure 3.11 depicts the interactions among the four entities: the client, the AS, the TGS, and the desired merchant server or resource S.

The exchanges are now explained.

3.15.1 Message (1): Request of a Session Ticket

A client C that desires to access a specific server S first requests an entrance ticket to the session from the Kerberos AS. To do so, the client sends message consisting of an identifier (e.g., a login and a password), the identifier of the server S to be addressed, a time stamp \(H_1\), and a random number \(R_{nd}\), both to prevent replay attacks.

3.15.2 Message (2): Acquisition of a Session Ticket

The Kerberos authentication server responds by sending a message formed of two parts: the first contains a session key \(K_{CGS}\) and the number \(R_{nd}\) that were in the first message, both coded with the client’s secret key \(K_C\), and the second includes the session ticket \(T_{CGS}\) destined for the TGS and encrypted by the latter’s secret key between itself and the Kerberos authentication server.

The session ticket includes several pieces of information, such as the client name \(C\), its network address \(A_d\), the time stamp \(H_1\), the period of validity of the ticket \(V_{al}\), and the session key \(K_{CGS}\). All these items, with the exception of the server identity TGS, are encrypted with the long-term key \(K_{TGS}\) that the TGS shares with the AS. Thus,

\[T_{CGS} = TGS, K_{TGS} \{C, A_d, H_1, Val, K_{CGS}\}\]

and the message sent to the client is

\[K_C \{K_{CGS}, R_{nd}\}, T_{CGS}\]

where \(K[x]\) indicates encryption of the message \(x\) with the shared secret key \(K\). The client decrypts the message with its secret key \(K_C\) to recover the session key \(K_{CGS}\) and the random number. The client verifies that the random number received is the same as was sent as a protection from replay attacks. The time stamp \(H_1\) is also used to protect from replay attacks. Although the
client will not be able to read the session ticket because it is encrypted with $K_{TGS}$; it can extract it and relay it in the server.

By default, the session ticket $T_{CTGS}$ is valid for 8 hours. During this time, the client can obtain several service tickets to different services without the need for a new authentication.

3.15.3 Message (3): Request of a Service Ticket

The client constructs an authentication title $Auth$ that contains its identity $C$, its network address $Ad_c$, the service requested $S$, a new time stamp $H_2$, and another random number $Rnd_3$ and then encrypts it with the session key $K_{CTGS}$. The encrypted authentication title can be represented in the following form:

$$Auth = K_{CTGS} \{ C, Ad_c, S, H_2, Rnd_3 \}$$

The request of the service ticket consists of the encrypted authentication title and the session ticket $T_{CTGS}$:

$$Service\ request = Auth, T_{CTGS}$$

3.15.4 Message (4): Acquisition of the Service Ticket

The TGS decrypts the ticket content with its secret key $K_{TGS}$; deduces the shared session key $K_{CTGS}$ and extracts the data related to the client’s service request. With knowledge of the session key, the server can decrypt the authentication title and compare it with the data it has supplied. This comparison gives formal proof that the client is the entity that was given the ticket. The time stamps confirm that the message was not an old message that has been replayed. Next, the TGS returns a service ticket for accessing the specific server $S$.

The exchanges described by Messages (3) and (4) can be repeated for all other servers available to the user as long as the validity of the session ticket has not expired.

The message from the TGS has two parts: part one contains a service key $K_{CS}$ between the client and the server $S$ and the number $Rnd_3$; both encoded with shared secret key $K_{CTGS}$ and part two contains the service ticket $T_{CS}$ destined to the server $S$ and encrypted by the service key $K_{STGS}$ shared between the server $S$ and the TGS.

As earlier, the service ticket destined for the server $S$ includes several pieces of information, such as the identity of the server $S$, the client name $C$, its network address $Ad_c$, a time stamp $H_2$, the period of validity of the ticket $Val$, and, if confidentiality is desired, a service key $K_{CS}$. All these items, with the exception of the server identity $S$, are encrypted with the long-term key $K_{STGS}$ that the TGS shares with the specific server. Thus, the client decrypts the message with the shared secret key $K_{CTGS}$ to recover the service key $K_{CS}$ and the random number. The client verifies that the random number received is the same as was sent as a protection from replay attacks.

3.15.5 Message (5): Service Request

The client constructs a new authentication title $Auth_2$ that contains its identity $C$, its network address $Ad_c$, a new time stamp $H_3$, and another random number $Rnd_4$ and then encrypts it with the service key $K_{CS}$. The encrypted authentication title can be represented as follows:

$$Auth_2 = K_{CS} \{ C, Ad_c, H_3, Rnd_4 \}$$

The request of the service consists of the encrypted new authentication title and the service ticket $T_{CS}$:

$$Service\ request = Auth_2, T_{CS}$$

3.15.6 Message (6): Optional Response of the Server

The server decrypts the content of the service ticket with the key $K_{STGS}$ it shares with the TGS to derive the service key $K_{CS}$ and the data related to the client. With knowledge of the service key, the server can verify the authenticity of the client. The time stamps confirm that the message is not a replay of old messages. If the client has requested the server to authenticate itself, it will return the random number, $Rnd_4$, encrypted by the service key $K_{CS}$. Without knowledge of the secret key $K_{CS}$, the server would have not been able to extract the service key $K_{CS}$.

The preceding description shows that Kerberos is mostly suitable for networks administered by a single-administrative entity. In particular, the Kerberos key distribution center fulfills the following roles:

- It maintains a database of all secret keys (except of the key between the client and the server, $K_{CS}$). These keys have a long lifetime.
- It keeps a record of users’ login identities, passwords and access privileges. To fulfill this role, it may need access to an X.509 directory.
- It produces and distributes encryption keys and ticket-granting tickets to be used for a session.
3.16 Public Key Kerberos

The utilization of a central depot for all symmetric keys increases the potential of traffic congestion due to the simultaneous arrival of many requests. In addition, centralization threatens the whole security infrastructure, because a successful penetration of the storage could put all the keys in danger (Sirbu and Chuang, 1997). Finally, the management of the symmetric keys (distribution and update) becomes a formidable task when the number of users increases.

The public key version of Kerberos simplifies key management, because the server authenticates the client directly using the session ticket and the client’s certificate sealed by the Kerberos certification authority. The session ticket itself is sealed with the client’s private key and then encrypted with the server public key. Thus, the service request to the server can be described as follows:

\[ \text{Service request} = S, PK_s \{ Tauth, Kr, Auth \} \]

with

\[ \text{Auth} = C, \text{certificate}, [Kr, S, PK_c, Tauth]SK_c \]

where

- \( Tauth \) is the initial time for authentication
- \( Kr \) is a one-time random number that the server will use as a symmetric key to encrypt its answer
- \( \ldots \) represents encryption with the server public key, \( PK_s \)
- \( \ldots \) represents the seal computed with the client’s private key, \( SK_c \)

This architecture improves speed and security.

3.16.1 Where to Find Kerberos?

The official web page for Kerberos is located at http://web.mit.edu/kerberos/www/index.html. A Frequently Asked Questions (FAQ) file on Kerberos can be consulted at the following address: ftp://athena-dist.mit.edu/pub/kerberos/KERBEROS.FAQ. Tung (1999) is a good compendium of information on Kerberos.

The Swedish Institute of Computer Science is distributing a free version of Kerberos, called Heidmal. This version was written by Johan Danielsson and Assar Westerlund. The differences between Heidmal and the MIT APIs are listed at http://web.mit.edu/kerberos/krb5-1.13/doc/appdev/h5l_mit_apidiff.html.

In general, commercial vendors offer the same Kerberos code that is available at MIT for enterprise solutions. Their main function is to provide support for installation and maintenance of the code and may include administration tools, more frequent updates and bug fixes, and prebuilt (and guaranteed to work) binaries. They may also provide integration with various smart cards to provide more secure and movable user authentication. A partial list of commercial implementations is available at http://web.ornl.gov/~jar/commerce.htm.

3.17 Exchange of Public Keys

3.17.1 The Diffie–Hellman Exchange

The Diffie–Hellman algorithm is the first algorithm for the exchange of public keys. It exploits the difficulty in calculating discrete algorithms in a finite field, as compared with the calculation of exponentials in the same field. The technique was first published in 1976 and entered in the public domain in March 1997.

The key exchange comprises the following steps:

1. The two parties agree on two random large integers, \( p \) and \( g \), such that \( g \) is a prime with respect to \( p \). These two numbers do not have to be necessarily hidden, but their choice can have a substantial impact on the strength of the security achieved.

2. \( A \) chooses a large random integer \( x \) and sends to \( B \) the result of the computation:

\[ X = g^x \mod p \]

3. \( B \) chooses another large random integer \( y \) and sends to \( A \) the result of the computation:

\[ Y = g^y \mod p \]

4. \( A \) computes

\[ k = Y^x \mod p = g^{xy} \mod p \]

5. Similarly, \( B \) computes

\[ k = X^y \mod p = g^{xy} \mod p \]

The value \( k \) is the secret key that both correspondents have exchanged.

The aforementioned protocol does not protect from man-in-the-middle attacks. A secure variant is as follows (Ferguson et al., 2010, pp. 183–193):
1. The two parties agree on \((p, q, g)\) such that
   
   a. \(p\) and \(q\) are prime numbers, typically \(2^{255} < q < 2^{256}\), that is, \(q\) is a 256-bit prime and \(p = Nq + 1\), for some even \(N\)
   
   b. \(g = \alpha^x \mod p\), with \(\alpha \in \mathbb{Z}_p^*, \mathbb{Z}_p^* = 1, \ldots, p - 1\) is the finite field modulo \(p\), such that \(g \neq 1\) and \(g^q = 1 \mod p\)
   
   2. \(A\) chooses a large random integer \(x \in [1, \ldots, q - 1]\) and sends \(B\) the result of the computation:
      \[X = g^x \mod p\]
   
   3. \(B\) verifies that
      
      a. \(X \in [2, \ldots, p - 1]\)
      
      b. \(q\) is a divisor of \((p - 1)\)
      
      c. \(g \neq 1\) and \(g^q = 1 \mod p\)
      
      d. \(X^q = 1\)
   
   4. Next, it chooses a large random integer \(y \in [1, \ldots, q - 1]\) and sends \(A\) the result of the computation:
      \[Y = g^y \mod p\]
   
   5. \(A\) verifies that \(Y \in [2, \ldots, p - 1]\) and that \(Y^q = 1\) and then computes
      \[k = Y^x \mod p = g^{xy} \mod p\]
   
   6. Similarly, \(B\) computes
      \[k = Y^x \mod p = g^{xy} \mod p\]

SSL/TLS uses the method called ephemeral Diffie–Hellman, where the exchange is short-lived, thereby achieving perfect forward secrecy, that is, that a key cannot be recovered after its deletion.

The Key Exchange Algorithm (KEA) was developed in the United States by the National Security Agency (NSA) based on the Diffie–Hellman scheme. All calculations in KEA are based on a prime modulus of 1024 bits with a key of 1024 bits and an exponent of 160 bits.

### 3.17.2 Internet Security Association and Key Management Protocol

IETF RFC 4306 (2005) combines contents that were described over several separate documents. One of these is Internet Security Association and Key Management Protocol (ISAKMP), a generic framework to negotiate point-to-point security associations and to exchange key and authentication data among two parties. In ISAKMP, the term security association has two meanings. It is used to describe the secure channel established between two communicating entities. It can also be used to define a specific instance of the secure channel, that is, the services, mechanisms, protocol and protocol-specific set of parameters associated with the encryption algorithms, the authentication mechanisms, the key establishment and exchange protocols, and the network addresses.

ISAKMP specifies the formats of messages to be exchanged and their building blocks (payloads). A fixed header precedes a variable number of payloads chained together to form a message. This provides a uniform management layer for security at all layers of the ISO protocol stack, thereby reducing the amount of duplication within each security protocol. This centralization of the management of security associations has several advantages. It reduces connect setup time, improves the reliability of software, and allows for future evolution when improved security mechanisms are developed, particularly if new attacks against current security associations are discovered.

To avoid subtle mistakes that can render a key exchange protocol vulnerable to attacks, ISAKMP includes five default exchange types. Each exchange specifies the content and the ordering of the messages during communications between the peers.

although ISAKMP can run over TCP or UDP, many implementations use UDP on port 500. Because the transport with UDP is unreliable, reliability is built into ISAKMP.

The header includes, among other information, two 8 octet “cookies”—also called “syncookies”—which constitute an anticlogging mechanism, because of their role against TCP SYN flooding. Each side generates a cookie specific to the two parties and assigns it to the remote peer entity. The cookie is constructed, for example, by hashing the IP source and destination addresses, the UDP source and destination ports and a locally generated secret random value. ISAKMP recommends including the data and the time in this secret value. The concatenation of the two cookies identifies the security association and gives some protection against the replay of old packets or SYN flooding attacks. The protection against SYN flooding assumes that the attacker will not intercept the SYN/ACK packets sent to the spoofed addresses used in the attack. As was explained earlier, the arrival of unsolicited SYN/ACK packets at a host that is accessible to the victim will elicit the transmission of an RST packet, thereby telling the victim to free the allocated resources so that the host whose address has been spoofed will respond by resetting the connection (Juels and Brainard, 1999; Simpson, 1999).
The negotiation in ISAKMP comprises two phases: the establishment of a secure channel between the two communicating entities and the negotiation of security associations on the secure channel. For example, in the case of IPSec, Phase I negotiation is to define a key exchange protocol, such as the Internet Key Exchange (IKE) and its attributes. Phase II negotiation concerns the actual cryptographic algorithms to achieve IPSec functionality.

IKE is an authenticated exchange of keys consistent with ISAKMP. IKE is a hybrid protocol that combines the aspects of the Oakley Key Determination Protocol and of SKEME. Oakley utilizes the Diffie–Hellman key exchange mechanism with signed temporary keys to establish the session keys between the host machines and the network routers (Cheng, 2001). SKEME is an authenticated key exchange that uses public key encryption for anonymity and nonrepudiation and provides means for quick refreshment (Krawczyk, 1996). IKE is the default key exchange protocol for IPSec. The protocol was first specified in 1998, and the latest revision of IKEv2 is defined in RFC 7296 (2014) and RFC 7427 (2015).

None of the data used for key generation is stored, and a key cannot be recovered after deletion, thereby achieving perfect forward secrecy. The price is a heavy cryptographic load, which becomes more important the shorter the duration of the exchanges. Therefore, to minimize the risks from denial-of-service attacks, ISAKMP postpones the computationally intensive steps until authentication is established.

ISAKMP is implemented in OpenBSD, Solaris, Linux, and Microsoft Windows and in some IBM products. Cisco routers implement ISAKMP for VPN negotiation using the cryptographic library from Cylink Corporation. A denial-of-service vulnerability of this implementation was discovered and fixed in 2012 (Cisco, 2012).

### 3.18 Certificate Management

When a server receives a request signed with a public key algorithm, it must first authenticate the declared identity that is associated with the key. Next, it will verify if the authenticated entity is allowed to perform the requested action. Both verifications rely on a certificate that a certification authority has signed. As a consequence, certification and certificate management are the cornerstones of e-commerce on open networks.

Certification can be decentralized or centralized. Decentralized certification utilizes PGP (Pretty Good Privacy) (Garfinkel, 1995) or OpenPGP. Decentralization is popular among those concerned about privacy because each user determines the credence accorded to a public key and assigns a confidence level in the certificate that the owner of this public key has issued. Similarly, a user can recommend a new party to members of the same circle of trust. This mode of operation also eliminates the vulnerability to attacks on a central point and prevents the potential abuse of a single authority. However, the users have to manage the certificates by themselves (update, revocation, etc.). Because that load increases exponentially with the number of participants, this mode of operation is impractical for large-scale operations such as online commerce.

In a centralized certification, a root certification authority issues the certificates to subordinate or intermediate certification authorities, which in turn certify other secondary authorities or end entities. This is denoted as X.509 certification, using the name of the relevant recommendation from the ITU-T. ITU-T Recommendation X.509 is identical to ISO/IEC 9594-1, a joint standard from the ISO and the IEC. It was initially approved in November 1988, and its seventh edition was published in October 2012. It is one of a series of joint ITU-T and ISO/IEC specifications that describe the architecture and operations of public key infrastructures (PKI). Some wireless communication systems such as IEEE 802.16 use X.509 certificates and RSA public key encryption to perform key exchanges.

X.500 (ISO/IEC 9594-1) provides a general view of the architecture of the directory, its access capabilities, and the services it supports.

X.501 (ISO/IEC 9594-2) presents the different information and administrative models used in the directory.

X.509 (ISO/IEC 9594-8) defines the base specifications for public key certificates such as using identity certificates and attribute certificates.

X.511 (ISO/IEC 9594-3) defines the abstract services of the directory (search, creation, deletion, error messages, etc.).

X.518 (ISO/IEC 9594-4) for searches and referrals in a distributed directory system using the Directory System Protocol (DSP).

X.519 (ISO/IEC 9594-5) specifies four protocols. The Directory Access Protocol (DAP) provides a directory user agent (DUA) at the client side access to retrieve or modify information in the directory. The Directory System Protocol (DSP) provides for the chaining of requests to directory system agents (DSA) that constitute a distributed directory. The Directory Information Shadowing Protocol (DISP) provides for the shadowing of information held on DSA to another DSA. Finally, the Directory Operational Binding Management Protocol (DOP) provides for the establishment, modification, and termination of bindings between pairs of DSAs.

X.520 (ISO/IEC 9594-6) and X.521 (ISO/IEC 9594-7) specify selected attribute types (keywords) and selected object classes to ensure compatibility among implementations.

The relationship among these different protocols is shown in Figure 3.12.

A simplified version of DAP, the Lightweight Directory Access Protocol (LDAP), is an application process that is part of the directory that responds to requests conforming to the LDAP protocol. The LDAP server may have the information stored in its local database or may forward the request to another DSA that understands the LDAP protocol. The latest specification of LDAP is defined in IETF RFCs 4511, 4512, and 4513 (2006). The main simplifications are as follows:

1. LDAP carried directly over the TCP/IP stack, thereby avoiding some of the OSI protocols at the application layer.
2. It uses simplified information models and object classes.
3. Being restricted to the client side, LDAP does not address what happens on the server side, for example, the duplication of the directory or the communication among servers.
4. Some directory queries are not supported.
5. Finally, Version 3 of LDAP (LDAPv3) does not mandate the strong authentication mechanisms of X.509. Strong authentication is achieved on a session basis for the TLS protocol.

IETF RFC 4513 (2006) specifies a minimum subset of security functions common to all implementations of LSAPv3 that use the SASL (Simple Authentication and Security Layer) mechanism defined in IETF RFC 4422 (2006). SASL adds authentication services and, optionally, integrity and confidentiality. Simple authentication is based on the name/password pair, concatenated with a random number and/or a time stamp with integrity protection using MD5.

FIGURE 3.12
Communication protocols among the components of the X.500 directory system.
3.18.1 Basic Operation

After receiving over an open network a request encrypted using public key cryptography, a server has to accomplish the following tasks before answering the request:

1. Read the certificate presented.
2. Verify the signature by the certification authority.
3. Extract the requester public key from the certificate.
4. Verify the requester signature on the request message.
5. Verify the certificate validity by comparison with the Certificate Revocation List (CRL).
6. Establish a certification path between the public key certificate to be validated and an authority recognized by the relying party, for example, the root authority. That certification path—or chain of trust—starts from an end entity and ends at the authority that validates the path (the root certification authority or the trust anchor as explained later).
7. Extract the name of the requester.
8. Determine the privileges that the requester enjoys.

The certificate permits the accomplishment of Tasks 1 through 7 of the preceding list. In the case of payments, the last step consists of verifying the financial data relating to the requester, in particular, whether the account mentioned has sufficient funds. In the general case, the problem is much more complex, especially if the set of possible queries is large. The most direct method is to assign a key to each privilege, which increases the complexity of key management.

The Certificate Management Protocol (CMP) of IETF RFC 4210 (2005) specifies the interactions between the various components of a public key infrastructure for the management of X.509 certificates (request, creation, revocation, etc.). The Online Certificate Status Protocol (OCSP) of IETF RFC 6960 (2013) specifies the data exchanges between an application seeking the status of one or more certificates and the server providing the corresponding status. This functionality is desirable to sending a CRL to save bandwidth. IETF RFC 2585 (1999) describes how to use the File Transfer Protocol (ftp) and HTTP to obtain certificates and certification revocation lists from their respective repositories.

During online verification of the certificates, end entities or registration authorities submit their public key using the Certification Signing Request of PKCS #10 as defined in IETF RFC 2986 (2000).

3.18.2 Description of an X.509 Certificate

An X.509 certificate is a record of the information needed to verify the identity of an entity. This record includes the distinguished name of the user, which is a unique name that ties the certificate owner with its public key. The certificate contains additional fields to locate its owner’s identity more precisely. The certificate is signed with the private key of the certification authority.

There are three versions of X.509 certificates, Versions 1, 2, and 3, the default being Version 1. X.509 Versions 2 and 3 certificates have an extension field that allows the addition of new fields while maintaining compatibility with the previous versions. Version 1 pieces of information are listed in Table 3.5.

Usually, a separate key is used for each security function (signature, identification, encryption, etc.) so that, depending on the function, the same entity may hold several certificates from the same authority.

There are two primary types of public key certificates: end entity certificates and certification authority (CA) certificates. The subject to an end entity public key certificate is not allowed to issue other public key certificates, while the subject of a CA certificate is another

| TABLE 3.5  |
| Content of a Version 1 X.509 Certificate |
| Field Name          | Description                                                                 |
| Version             | Version of the X.509 (2001) certificate                                      |
| Serial number       | Certificate serial number assigned by the certification authority          |
| Signature           | Identifier of the algorithm and hash function used to sign the certificate   |
| Issuer              | Distinguished name of the certification authority                           |
| Validity            | Duration of the validity of the certificate                                 |
| Subject             | References for the entity whose public is certified, such as the distinguished name, unique identifier (optional), and so on |
| Subject public key info | Information concerning the algorithm that this public key is an instance of and the public key itself |
certification authority. CA certificates fall into the following categories:

- Self-issued certificates, where the issuer and the subject are the same certification authority.
- Self-signed certificates, where the private key that the certification authority used for signing is the same as the public key that the certificate certifies. This is typically used to advertise the public key or any other information that the certification authority wishes to make available.
- Cross-certificates, where the issuer is one certification authority and the subject is another certification authority. In a strict hierarchy, the issuer authorizes the subject certification authority to issue certificates, whereas in a distributed trust model, one certification authority recognizes the other.

Cross-certifications are essential for business partners to validate each other credentials. For cross-certification, policies and policy constraints must be similar or equivalent on both sides. This requires a mixture of technical, political, and managerial steps. On a technical level, NIST Special Publication 800-15 provides the minimum interoperability specifications for PKI components (MISPC) (Burr et al., 1997).

### 3.18.3 Attribute Certificates

X.509 defines a third type of public key certificates called *attribute* certificates that are digitally signed by an attribute authority to bind certain prerogatives, such as access control, to identity separately from the authentication of the identity information. More than one attribute certificate can be associated with the same identity.

Attribute certificates are managed by a *Privilege Management Infrastructure* (PMI). This is the infrastructure that supports a comprehensive authorization service in relation to a public key infrastructure. The PKI and PMI are separate logical and/or physical infrastructures and may be established independently but they are related. When a single entity acts as both a certification authority and an attribute authority, different keys are used for each kind of certificates. With a hierarchical role-based access control (RBAC), higher levels inherit the permissions accorded to their subordinates.

The Source of authority (SOA) is the ultimate authority to assign a set of privileges. It plays a role similar to the root certification authority and can be certified by that authority. The SOA may also authorize the further delegation of these privileges, in part or in full, along a delegation path. There may be restrictions on the power of delegation capability, for example, the length of the delegation path can be bonded, and the scope of privileges allowed can be restricted downstream. To validate the delegation path, each attribute authority along the path must be checked to verify that it was duly authorized to delegate its privileges.

Although it is quite possible to use public key identity certificates to define what the holder of the certificate may be entitled to, a separate attribute certificate may be useful in some cases, for example:

1. The authority for privilege assignment is distinct from the certification authority.
2. A variety of authorities will be defining access privileges to the same subject.
3. The same subject may have different access permissions depending on the role that individual plays.
4. There is the possibility of delegation of privileges, in full or in part.
5. The duration of validity of the privilege is shorter than that of the public key certificate.

Conversely, the public key identity certificate may suffice for assigning privileges, whenever the following occur:

1. The same physical entity combines the roles of certification authority and attribute authority.
2. The expiration of the privileges coincides with that of the public key certificate.
3. Delegation of privileges is not permitted or if permitted, all privileges are delegated at once.

### 3.18.4 Certification Path

The idea behind X.509 is to allow each user to retrieve the public key of certified correspondents so that they can proceed with the necessary verifications. It is sufficient therefore to request the closest certification authority to send the public key of the communicating entity in a certificate sealed with the digital signature of that authority. This authority, in turn, relays the request to its own certifying authority, and this permits an escalation through the chain of authorities, or certification path, until reaching the top of the certification pyramid (the root authority, RA). Figure 3.13 is a depiction of this recursive verification.

Armed with the public key of the destination entity, the sender can include a secret encrypted with the public key of the correspondent and corroborate that the partner is the one whose identity is declared. This is because, without the private key associated with the
key used in the encryption, the destination will not be able to extract the secret. Obviously, for the two parties to authenticate themselves mutually, both users have to construct the certification path back to a common certification authority.

Thus, a certification path is formed by a continuous series of certification authorities between two users. This series is constructed with the help of the information contained in the directory by going back to a common point of confidence. In Figure 3.13, authorities C, B, and
A are the intermediate certification authorities. From the end entity’s perspective, the root authority and an intermediate certification authority perform the same function, that is, they are functionally equivalent. However, the reliability of the system requires that each authority of the chain ensures that the information in the certification is correct. In other words, the security requires that none of the intermediate certification authorities are deficient or compromised.

It should be noted that the various intermediate authorities do not have to reside in the same country neither in the country of the root authority. This can be also different from the country where the data are stored. End users are therefore vulnerable to the different privacy laws in various jurisdictions. Moreover, governments in any country along the certification path have the possibility of initiating man-in-the-middle attacks by forcing the CAs in their jurisdiction to issue false certificates that can be used to intercept users data (Soghoian and Stamm, 2010).

Registration relates to the approval and rejection of certificate applications and the request of revocation or renewal of certificates and is different from key management and the certificate management. In some systems, the registration authority is different from the certification authority, for example, the human resources (HR) of a company while the certification is handled by the IT department. In such a case, the HR will have its own intermediate certification authority.

The tree structure of the certification path can be hierarchical or nonhierarchical as explained next.

### 3.18.5 Hierarchical Certification Path

According to a notational convention used in earlier versions of X.509, a certificate is denoted by

\[ \text{card}_{\text{authority}}(\text{entity}) \]

Thus,

\[ X_1(\langle X_2 \rangle) \]

indicates the certificate for entity \( X_2 \) that authority \( X_1 \) has issued, while

\[ X_1(\langle X_2 \rangle)X_2(\langle X_3 \rangle)\ldots X_n(\langle X_{n+1} \rangle) \]

represents the certification path connecting the end entity \( X_{n+1} \) to authority \( X_1 \). In other words, this notation is functionally equivalent to \( X_1(\langle X_{n+1} \rangle) \), which is the certificate that authority \( X_1 \) would have issued to the end entity \( X_{n+1} \). By constructing this path, another end entity would be able to retrieve the public key of end entity \( X_{n+1} \), if that other end entity knows \( X_1p \), the public key of authority \( X_1 \). This operation is represented by

\[ X_1p \cdot X_1(\langle X_2 \rangle) \]

where \( \cdot \) is an infix operator, whose left operand is the public key, \( X_1p \), of authority \( X_1 \), and whose right operand is the certificate \( X_1(\langle X_2 \rangle) \) delivered to \( X_2 \) by that same certification authority. This result is the public key of entity \( X_2 \).

In the example depicted in Figure 3.14, assume that user \( A \) wants to construct the certification path toward another user \( B \). \( A \) can retrieve the public key of authority \( W \) with the certificate signed by \( X \). At the same time, with the help of the certificate of \( V \) that \( W \) has issued, it is possible to extract the public key of \( V \). In this manner, \( A \) would be able to obtain the chain of certificates:

\[ X(\langle W \rangle), W(\langle V \rangle), V(\langle Y \rangle), Y(\langle Z \rangle), Z(\langle B \rangle) \]

This itinerary, represented by \( A \rightarrow B \), is the forward certification path that allows \( A \) to extract the public key \( Bp \) of \( B \), by an application of the operation \( \cdot \) in the following manner:

\[ Bp = Xp \cdot (A \rightarrow B) \]

\[ = Xp \cdot X(\langle W \rangle)W(\langle V \rangle)V(\langle Y \rangle)Y(\langle Z \rangle)Z(\langle B \rangle) \]

In general, the end entity \( A \) also has to acquire the certificates for the return certification path \( B \rightarrow A \), to send them to its partner:

\[ Z(\langle Y \rangle), Y(\langle V \rangle), V(\langle W \rangle), W(\langle X \rangle), X(\langle A \rangle) \]

When the end entity \( B \) receives these certificates from \( A \), it can unwrap the certificates with its private key to extract the public key of \( A \), \( Ap \):

\[ Ap = Zp \cdot (B \rightarrow A) \]

\[ = Zp \cdot Z(\langle Y \rangle)V(\langle V \rangle)W(\langle W \rangle)X(\langle X \rangle) \]

As was previously mentioned, such a system does not necessarily impose a unique hierarchy worldwide. In the case of electronic payments, two banks or the fiscal authorities of two countries can mutually certify each other. In the preceding example, the intermediate certification authorities \( X \) and \( Z \) are cross-certified.
If A wants to verify the authenticity of B, it is sufficient to obtain
\[ X(Z), Z(B) \]
to form the forward certification path and
\[ Z(X) \]
to construct the reverse certification path. This permits the clients of the two banks to be satisfied with the certificates supplied by their respective banks.

A root certification authority may require all its intermediary authorities to keep an audit trail of all exchanges and events, such as key generation, request for certification, validation, suspension, or revocation of certificates.

Finally, it should be noted that cross-certification applies to public key certificates and not to attribute certificates.

### 3.18.6 Distributed Trust Model

If certification authorities are not organized hierarchically, the end entities themselves would have to construct the certification path. In practice, the number of operations to be carried out can be reduced with various strategies, for example:

1. Two end entities served by the same certification authority have the same certification path, and can exchange their certificates directly. This is the case for the end entities C and A in Figure 3.15.
2. If one end entity is constantly in touch with users that a particular authority has certified, that end entity could store the forward and return certification paths in memory. This would reduce the effort for obtaining the other users’ certificates to a query into the directory.
3. Two end entities that have each other’s certificates mutually authenticate themselves without querying the directory. This reverse certification is based on the confidence that each end entity has in its certification authority.

Later revisions of X.509 have introduced a new entity called a Trust Anchor. This is an entity that a party relies on to validate certificates along the certification path. In complex environments, the trust anchor may be different from the root authority, for example, if the length of the certification path is restricted for efficiency.

### 3.18.7 Certificate Classes

In general, service providers offer several classes of certificates according to the strength of the link between the certificate and the owner’s identity. Each class of
certificates has its own root authority and possibly registration authorities.

Consider, for example, a three-class categorization. In this case, Class 1 certificates confirm that the distinguished name that the user presents is unique and unambiguous within the certification authority's domain and that it corresponds to valid e-mail address. They are typically used for domain registration. Class 1 certificates are used for modest enhancement of security through confidentiality and integrity verification. They cannot be used to verify an identity or to support nonrepudiation services.

Class 2 certificates are also restricted to individuals. They indicate that the information that the user has submitted during the registration process is consistent with the information available in business records or in “well-known” consumer databases. In North America, one such reference database is maintained by Equifax. Class 3 certificates are given to individuals and to organizations. To obtain a certificate of this class, an individual has to be physically present with their public key in possession before an authority to confirm the identity of the applicant with a formal proof of identity (passport, identity card, electricity or telephone bill, etc.) and the association of that identity with the given public key. If the individual is to be certified as a duly authorized representative of an organization, then the necessary verifications have to be made. Similarly, an enterprise will have to prove its legal existence. The authorities will have to verify these documents by querying the databases for enterprises and by confirming the collected data by telephone or by mail. Class 3 certificates have many business applications.

### 3.18.8 Certificate Revocation

The correspondence between a public key and an identity lasts only for a period of time. Therefore, certification authorities must refer to revocation lists that contain certificates that have expired or have been revoked. These lists are continuously updated. Table 3.6 shows the format of the revocation list that Version 1 of X.509 has defined. In the third revision of X.509, other optional entries, such as the date of the certificate revocation and the reason for revocation, were added.

The CPS describes the circumstances under which certification of end users and the various intermediate authorities can be revoked and defines who can request that revocation. To inform all the entities of the PKI, CRLs are published at regular intervals—or when the certificate of an authority is revoked—with the digital signature of the certification authority to ensure their integrity. Among other information, the CRL indicates the issuer’s name, the date of issue, the date of the next scheduled CRL, the revoked certificates serial numbers, and the specific times and reasons for revocation.

In principle, each certification authority has to maintain at least two revocation lists: a dated list of the certificates that it has issued and revoked and a dated list of all the certificates that the authorities that it recognizes have revoked. The root certification authority and each of its delegate authorities must be able to access these lists to verify the instantaneous state of all the certificates to be treated within the authentication system.

Revocation can be periodic or exceptional. When a certificate expires, the certification authority withdraws it from the directory (but retains a copy in a special directory, to be able to arbitrate any conflict that might arise in the future). Replacement certificates have to be ready and supplied to the owner to ensure the continuity of the service.

The root authority (or one of its delegated authorities) may cancel a certificate before its expiration date, for example, if the certificate owner’s private key was compromised or if there was abuse in usage. In the case of secure payments, the notion of solvency, that is, that the user has available the necessary funds, is obviously one of the essential considerations.

Processing of the revocation lists must be done quickly to alert users and, in certain countries, the authorities, particularly if the revocation is before the expiration date. Perfect synchronization among the various authorities must be attained to avoid questioning the validity of documents signed or encrypted before the withdrawal of the corresponding certificates.

Users must also be able to access the various revocation lists; this is not always possible because current client programs do not query these lists.

### 3.18.9 Archival

Following certification expiration or its revocation, the records associated with a certificate are retained for at least the specific time periods. Table 3.7 shows the current intervals for Symatech. Thus, archival of Class 1
certificates lasts for at least 5 years after expiration of the certificate or its revocation, while the duration for Class 2 and 3 certificates are 10.5 years each (Symantec Corporation, 2015).

### 3.18.10 Recovery

Certification authorities implement procedures to recover from computing failures, corruption of data, such as when a user’s private key is compromised, and natural or man-made disasters.

A disaster recovery plan addresses the gradual restoration of information services and business functions. Minimal operations can be recovered within 24 hours. They include certificate issuance or revocation, publication of revocation information, and recovery of key information for enterprises customers.

### 3.18.11 Banking Applications

A bank can certify its own clients to allow them access to their back account across the Internet. Once access has been given, the operation will continue as if the client were in front of an automatic teller machine. The interoperability of bank certificates can be achieved with interbank agreements, analogous to those that have permitted the interoperability of bank cards. Each financial institution certifies its own clients and is assured that the other institutions will honor that certificate.

As the main victims of fraud, financial institutions have partnered to establish their own certification infrastructures. In the United States, several banks including Bank of America, Chase Manhattan, Citigroup, and Deutsche Bank have formed IdenTrust (in 2000). The main purpose was to enable a trusted business-to-business e-commerce marketplace with financial institutions as the key trust providers. The institution changed its name into IdenTrust (http://www.identrust.com) in 2006 and was acquired by HID in 2014. It is one of two vendors accredited by the General Services Administration (GSA) to issue digital certificates.

At the same time, about 800 European institutions have joined forces to form a Global Trust Authority (GTA) as a nonprofit organization whose mission is to put in place an infrastructure of trust that can be used, by all sectors, to conduct cross-border e-business. In 2009, the project was suspended (IDABC, 2009).

### 3.19 Authentication

#### 3.19.1 Procedures for Strong Authentication

Having obtained the certification path and the other side’s authenticated public key, X.509 defines three procedures for authentication, one-way or unidirectional authentication, two-way or bidirectional authentication; and three-way or tridirectional authentication.

##### 3.19.1.1 One-Way Authentication

One-way authentication takes place through the transfer of information from User $A$ to User $B$ according to the following steps:

- $A$ generates a random number $R_A$ used to detect replay attacks.
- $A$ constructs an authentication token $M = (T^A, R_A, I_B, d)$ where $T^A$ represents the time stamp of $A$ (date and time) and $I_B$ is the identity of $B$. $T^A$ comprises two chronological indications, for example, the generation time of the token and its expiration date, and $d$ is an arbitrary data. For additional security, the message can be encrypted with the public key of $B$.
- $A$ sends to $B$ the message:

$$B \rightarrow A, A\{ (T^A, R_A, I_B, d) \}$$

where $B \rightarrow A$ is the certification path

$A[M]$ represents the message $M$ encrypted with the private key of $A$

$B$ carries on the following operations:

- Obtain the public key of $A$, $A_p$, from $B \rightarrow A$, after verifying that the certificate of $A$ has not expired.
- Recover the signature by decrypting the message $A[M]$ with $A_p$. $B$ then verifies that this signature is identical to the message hash, thereby ascertaining simultaneously the signature and the integrity of the signed message.
- Verify that $B$ is the intended recipient.
- Verify that the time stamp is current.
• Optionally, verify that $R^A$ has not been previously used.

These exchanges prove the following:

• The authenticity of $A$, that is, the authentication token was generated by $A$
• The authenticity of $B$, that is, the authentication token was intended for $B$
• The integrity of the identification token
• The originality of the identification token, that is, it has not been previously utilized

### 3.19.1.2 Two-Way Authentication

The procedure for two-way authentication adds to the previous unidirectional exchanges similar exchanges but in the reverse direction. Thus, the following applies:

• $B$ generates another random number $R^B$.
• $B$ constructs the message $M' = (T^B, R^B, I_A, R^A, d)$, where $T^B$ represents the time stamp of $B$ (date and time), $I_A$ is the identity of $A$, and $R^A$ is the random number received from $A$. $T^B$ consists of one or two chronological indications as previously described. For security, the message can be encrypted with the public key of $A$.
• $B$ sends to $A$ the message:

$$B[(T^B, R^B, I_A, R^A, d)]$$

where $B[M']$ represents the message $M'$ encrypted with the private key of $B$.

• $A$ carries out the following operations:
  • Extracts the public key of $B$ from the certification path and uses it to decrypt $B[M']$ and recovers the signature of the message that $B$ has produced; $A$ verifies next that the signature is the same as the hashed message, thereby ascertaining the integrity of the signed information.
  • Verifies that $A$ is the intended recipient
  • Checks the time stamp to verify that the message is current
  • As an option, verifies that $R^B$ has not been previously used

### 3.19.1.3 Three-Way Authentication

Protocols for three-way authentication introduce a third exchange from $A$ to $B$. The advantage is the avoidance of time stamping and, as a consequence, of a trusted third party. The steps are the same as for two-way identification but with $T^A = T^B = 0$. Then:

• $A$ verifies that the value of the received $R^A$ is the same that was sent to $B$.
• $A$ sends to $B$ the message:

$$A\{R^B, I_B\}$$

encrypted with the private key of $A$.

• $B$ performs the following operations:
  • Verifies the signature and the integrity of the received information
  • Verifies that the received value of $R^B$ is the same as was sent

### 3.20 Security Cracks

We have reviewed in this chapter the main concepts of how cryptographic algorithms may be used to support multiple security services. On a system level, however, the security of a system depends on many factors such as

1. The rigorosity of the authentication criteria
2. The degree of trust place in the root authority and/or intermediate certification authorities
3. The strength of the end entity’s credentials (e.g., passport, birth certificate, driver’s license)
4. The strength of the cryptographic algorithms used
5. The strength of the key establishment protocols
6. The care with which end entities (i.e., users) protect their keys

This is why the design of security systems that provide the desired degree of security requires high skill and expertise and attention to details. For example, if the workstations are not properly configured, users can be denied access or signed e-mails may appear invalid (Department of Defense Public Key Enabling and Public Key Infrastructure Program Management Office, 2010).
3.20.1 Problems with Certificates

Certification is essential for authenticating participants to prevent intruders from impersonating any side to spy on the encrypted exchanges. When an entity produces a certificate signed by a certification authority, this means that the CA attests that the information in the certificate is correct. As a corollary, the entry for that entity in the directory that the certification authority maintains has the following properties:

1. It establishes a relationship between the entity and a pair of public and private cryptographic keys.
2. It associates a unique distinguished name in the directory with the entity.
3. It establishes that at a certain time, the authority was able to guarantee the correspondence between that unique distinguished name and the pair of keys.

Each certification authority describes its practices and policies in a Certification Practice Statement (CPS). The CPS covers the obligations and liabilities, including liability caps, of various entities, their obligations. For example, one obligation is to maintain their cryptographic technology current and to protect the integrity of their physical and logical operations including key management.

There is no standard CPS, but IETF RFC 3647 (2003) offers guidance on how to write such a certification statement. The accreditation criteria of entities are also not standardized, and there is no code of conduct for certification authorities. Each operator defines its conduct, rights, and obligations and operates at its own discretion and is not obliged to justify its refusal to accredit an individual or an entity. There are also no standard criteria to evaluate the performance of a certification authority, so that browser vendors are left to their own discretion as to which certification authorities they should trust. There are almost no laws to prevent a PKI operator from cashing in on the data collected on individuals and their purchasing habits by passing the information to all those that might be interested (merchants, secret services, political adversaries, etc.). Thus, should the certification authorities fail to perform their role correctly, willingly, by accident or negligently, the whole security edifice is called into question.

The American Institute of Certified Public Accountants and the Canadian Institute of Chartered Accountants have developed a program to evaluate the risks of conducting commerce through electronic means. The CPA WebTrustSM is a seal that is supposed to indicate that a site is subject to quarterly audit on the procedures to protect the integrity of the transactions and the confidentiality of the information. However, there are limits to what audits can uncover as shown in the notorious case of DigiNotar.

DigiNotar was a certification authority agreed and audited by the Dutch government as part of its PKI program. In August 2011, however, following an investigation by the Dutch government, DigiNotar was forced to admit that more than 500 SSL/TLS certificates were stolen from compromised DigiNotar’s servers and were used to create fake DigiNotar certificates. In particular, the fake certificate for google.com was used to spy on to the TLS/SSL sessions of some 300,000 Iranians using gmail accounts. DigiNotar detected and revoked some of the fraudulent certificates without notifying the browsers’ manufacturers such as Apple, Google, Microsoft, Mozilla, and Opera forcing them to issue updates to block access to sites secured with DigiNotar certificates. Finally, in September 2011, The Dutch Independent Post and Telecommunications Authority (OPTA) revoked DigiNotar as a certification authority (Schoen, 2010; Keizer, 2011; Nightingale, 2011).

In the case of DigiNotar, a fundamental problem was shown in the way certificates are used in consumer applications. Browsers maintain a list of trusted certification authorities or rely on the list that the operating system provides. Each of these authorities has the power to certify additional certification authorities. Thus, a browser ends up trusting hundreds of certificate authorities some which do not necessarily have the same policies. Because the authorities on the chain of trust may be distributed over several countries, different government agencies may legally compel any of these certification authorities to issue false certificates to intercept and hijack individuals’ secure channels of communication. In fact, there are some commercial covert surveillance devices that operate on that principle (Soghoian and Stamm, 2010).

Browsers currently contact the CAs to verify that a certificate that a server presents has not been revoked, but they currently do not track changes in the certificate, for example, by comparing hashes (but there are Firefox add-ons that perform this function). In principle, browser suppliers work with certification authorities to respond and contain breaches by blocking fraudulent certificates. Experience has shown, however, that certification authorities do not always notify the various parties of security breaches promptly.

3.20.2 Underground Markets for Passwords

The spread of online electronic commerce has been correlated to the number of times individuals have to login to different systems and applications. In many cases, the back-end authentication systems are different
so that users have to manage an increasing number of passwords, particularly that many systems force the users to replace their passwords periodically. In a typical large application, such as for a bank, the password database may include around 100 or 200 m password. Increasingly, users face the problem of creating and remembering multiple user names and passwords and often end up reusing passwords.

One of the consequences of all these factors is the rise in cybercrime, fuelled by underground markets of stolen passwords and of tools ("bots") to attempt automatic logins in many websites trying the passwords in a file until access is achieved.

### 3.20.3 Encryption Loopholes

Encryption is a tool to prevent undesirable access to a secret message. While the theoretical properties of the cryptographic algorithms are important, how the fundamentals of cryptography implemented and used are essential. Brute-force attacks where the assailant systematically tries all possible encryption keys until getting the one that will reveal the plaintext. Table 3.8 provides the estimated time for successful brute-force attacks with exhaustive searches on symmetric encryption algorithms with different key lengths for the current state of technology (Paar and Pelzl, 2010, p. 12).

As a consequence, a long key is a necessary but not sufficient condition for secure symmetric encryption. Cryptanalysis focuses on finding design errors, implementation flaws or operational deficiencies to break the encryption and retrieve the messages, even without knowledge of the encryption key. GSM, IEEE 802.11b, IS-41, and so on are known to have faulty or deliberately weakened protection schemes.

The most common types of cryptological attacks include the following (Ferguson et al., 2010, pp. 31–36):

1. Attacks on the encrypted text assuming that the clear text has a known given structure, for example, the systematic presence of a header with a known format (this is the case of e-mail messages) or the repetition of known keywords.

2. Attacks starting with chosen plaintexts that are encrypted with the unknown key so as to deduce the key itself.

3. Attacks by replaying old legitimate messages to evade the defense mechanisms and to short-circuit the encryption.

4. Attacks by interception of the messages (man-in-the-middle attacks) where the interceptor inserts its eavesdrop at an intermediate point between the two parties. After intercepting an exchange of a secret key, for example, the interceptor will be able to decipher the exchanged messages while the participants think they are communicating in complete security. The attacker may also be able to inject fake messages that would be treated as legitimate by the two parties.

5. Attacks by measuring the length of encryption times, of electromagnetic emissions, and so on to deduce the complexity of the operations, and hence their form.

6. Attacks on the network itself, for example, corruption of the DNS, to direct the traffic to a spurious site.

In some cases, the physical protection of the whole cryptographic system (cables, computers, smart cards, etc.) may be needed. For example, bending of an optical fiber results in the dispersion of 1%–10% of the signal power; therefore, well-placed acoustic-optic devices can capture the diffraction pattern for later analysis.

A catalog of the causes of vulnerability includes the following (Schneier, 1997, 1998a; Fu et al., 2001):

1. Nonverification of partial computations.

2. Use of defective random number generators, because the keys and the session variables depend on a good supply source for nonpredictable bits.

3. Improper reutilization of random parameters.

4. Misuse of a hash function in a way that increases the chances for collisions.

5. Structural weakness of the telecommunications network.

6. Nonsystematic destruction of the clear text after encryption and the keys used in encryption.

7. Retention of the password or the keys in the virtual memory.

8. No checking of correct range of operation; this is particularly the case when buffer overflows can cause security flaws.

| Table 3.8 |
|---|---|
| **Estimated Time for Successful Brute-Force Attacks on Symmetric Encryption for Different Key Lengths with Current Technology** | |
| Key Length in Bits | Estimated Time with Current Technology |
| 56–64 | A few days |
| 112–128 | Several years |
| 256 | Several decades |
9. Misuse of a protocol can lead to an authenticator traveling in plaintext. For example, IETF RFC 2109 (1997)—now obsolete—specified that when the authenticator is stored in a cookie, the server has to set the Secure flag in the cookie header so that the client waits before returning the cookie until a secure connection has been established with SSL/TLS. It was found that some web servers neglected to set this flag thereby negating that protection. The authenticator can also leak if the client software continues to use even after the authentication is successful.

A new type of vulnerability was recently discovered and given the name of cross-app resource access attacks (XARA). Operating systems such as Apple’s OS X and iOS attempt to isolate applications from each other even when the same user is running (“sandboxing”) to prevent a malicious or compromised program from harming the others. To talk to each other, applications use interprocess communication channels. Apple’s implementation of sandboxing has several flaws that can be exploited to enable a malicious app, though vetted by the Apple Store, to gain unauthorized access to the sensitive data of other apps (e.g., passwords) and utilize their resources surreptitiously (Xing et al., 2015). For example, Apple’s credential management service, called the keychain, allows each application to record the user’s credentials (the user’s passwords, secret keys, and certificates) but does not offer an easy way to determine the owner of an existing keychain item and/or to authenticate that owner before granting access to that item. Taking advantage of this flaw, a malicious app can secretly allocate keychain attributes for apps that have not yet been installed in the hope that the user will install them later. In that case, the malicious app will have full access to the credentials. If the target application is already installed, the malicious app can delete the corresponding item of keychain and replace it with a new access control list under the attacker’s control. So, when the target application updates the user’s credentials, they will be divulged to the attacker. In June 2015, Apple was still working on a fix for these security flaws (Goodin, 2015).

It is also possible to take advantage of other implementation details that are not directly related to encryption. For example, when a program deletes a file, most commercial operating systems merely eliminate the corresponding entry in the index file. This allows recovery of the file, at least partially, with off-the-shelf software. The only means by which to guarantee total elimination of the data is to rewrite systematically each of the bits that the deleted file was using. Similarly, the use of the virtual memory in commercial systems exposes another vulnerability because the secret document may be momentarily in the clear on the disk.

With regard to systems for e-commerce that are for the general public must be easily accessible and affordably priced. As a consequence, many compromises will be made to improve response time and the ease of use. However, if one starts from the principle that, sooner or later, any system is susceptible to unexpected attacks with unanticipated consequences, it is important the system make it possible to detect attacks and to accumulate proofs that are accepted by law enforcement personnel and the courts. The main point is to have an accurate definition of the type of expected threats and possible attacks. Such a realistic evaluation of threats and risks permits a precise understanding of what should be protected, against whom, and for how long.

3.20.4 Phishing, Spoofing, and Pharming

Phishing, spoofing, and pharming have been used to trick users to voluntarily reveal their credentials. These terms are neologisms that describe various tricks played on users. Although they are often used interchangeably, we attempt here to distinguish among them for the sake of clarity.

Phishing is a deceitful message to trick users into revealing their credentials (bank account numbers, passwords, payment card details, etc.) to unauthorized entities. The term was coined to indicate that fraudsters are “fishing” for online banking details of customers through e-mail. The sender can impersonate reputable companies such as banks, financial intermediaries or banks and exploits the whole gamut of human emotions and credulity, ranging from obligation toward friends in distress in foreign lands, fear of arrest warrants to greed and ignorance. For example, messages purportedly from a bank would alert the recipient that there is a security problem with the recipient’s account that requires immediate attention and then ask the recipient to click on a link that seems legitimate under any false pretense (to restore a compromised account, to verify identity, etc.). However, by using HTML Forms in the body of the text or JavaScript event handler, the link seen in the e-mail is different from the actual link destination. Once the link is clicked, a window opens that contain the real site or a fake copy of the bank’s website would ask for their account details or authentication data. When the user types the credential, they are captured and used to siphon the user’s account (Drake et al., 2004).

After convincing the recipient that the e-mail is credible and originated from a trusted institution, phishing exploits the whole gamut of human emotions and credulity to persuade the recipient to divulge personal and financial information under the guise of verifying
account, paying taxes, assisting a friend mugged in a foreign land, and so on.

Alternatively, a malware (malicious software) may be injected into the target device to record all keyboard inputs and e-mail the collected data periodically to what is called a mail drop (Schneier, 2004; Levin et al., 2012). An extreme form of this malware is called ransomware, where hackers seize control of the target system unless a ransom is paid. This happened in June 2015 to the police force of Tewksbury, Massachusetts (a suburb of Boston), which was only able to access its data back after paying a ransom in Bitcoin.

In February 2015, the Russian antivirus firm Kaspersky Lab issued a report detailing how a group calling themselves Carbanak was able to penetrate bank networks using phishing e-mails to steal over $500 million from banks in several countries and their private customers. In some instances, cash machines were instructed to dispense their contents to associates. In other cases, they altered databases to increase fraudulently balances on existing accounts and then pocket the difference without the knowledge of the account owner (Economist, 2015b).

Finally, pharming is the systematic exploitation of a vulnerability of the DNS servers due, for instance, to a coding error or to an implanted malware. As a result, instead of translating web names into their corresponding IP address, spurious IP addresses associated with a fake site are used. In other words, the effects of spoofing and pharming are the same, but in spoofing the victim is a willingly albeit gullible participant while in pharming the victim is unaware of the redirection.

To conduct such a campaign, an attacker registers a new domain with an Internet domain registrar using contact details similar to those of the target company as obtained with a whois lookup on the real domain. The attacker then creates a DNS record that points to the newly registered domain that will post a fake landing page to give the e-mail recipients the impression of genuineness so that they enter willingly their credentials. The deceptive site may even be a compromised website of another victim to which the attacker has uploaded the phishing kit. Some kits are freely available on the Internet, while others are distributed within closed communities in a hacking forum.

The root cause of all these problems is the lack of authentication in electronic mail exchanges. E-mail was originally intended for the communication among collaborators who knew each other and not for the users distributed among more than 70 million domains to communicate without any prior screened. Only partial solutions have been offered thus far.

RFC 4871 (2007) defines an authentication framework, called DomainKeys Identified Mail (DKIM), which is a synthesis of two proposals from Cisco and Yahoo! to associate a domain name identity to a message through public key cryptography. This can be applied at the message transfer agents (MTA) located in the various network nodes or at the e-mail clients, that is, readers or more formally mail user agents (MUA). The sending MTA checks if the source is authorized to send mail using its domain. If the message is authorized, it signs the body of the message and selected header fields and inserts both signatures in the new DKIM-Signature header field. The MTA of the receiving domain verifies the integrity of the message and the message headers.

The signatures are calculated through hashing with SHA-1 or SHA-256 and the encryption with RSA, with key sizes ranging from 512 to 2048 bits. RFC 4871 avoids a public key certification infrastructure and uses the DNS to maintain the public keys of the claimed senders. Accordingly, verifiers have to request the sender public key from the DNS.

The scheme in RFC 4871 does not protect from relay attacks. For example, a spammer can send a message to an accomplice while the original message satisfies all the criteria set in RFC 4871. The accomplice inserts the extra fields needed to forward the message, but the signatures will remain valid, fooling the DKIM tool at the receiving end (see § 8.5 of RFC 4871). The end user will still be unable to determine if the message is a phishing attack or that it contains a malicious attachment. The information, however, may be useful for the receiving MTA to establish that a specific e-mail address is being used for spamming (Selzer, 2013). Also, because the DNS is not fully equipped for key management, an attacker can publish key records in DNS that are intentionally malformed.

The Sender Policy Framework (SPF) of RFC 7208 (2014) is another approach based on IP addresses. Work on the specifications took more than 10 years, but the concept was implemented and deployed before the RFC was published. Here, a domain administrator populates the SPF record of that domain’s entry in the DNS with the IP addresses that are allowed to send e-mail from that domain and publish that SPF data as DNS TXT (type 16) Resource Record per RFC 1035 (1987). Any mail server or spam filter on the user’s machine query the DNS for the domain’s SPF record to verify the source of e-mail messages as they arrive. Only messages whose IP addresses, as specified in the MAIL FROM field, match one of the authorized IP addresses will be accepted. Thus, other mail servers can stop fake mail claiming to originate from that domain. The only benefit for the domain is to protect its reputation. Furthermore, there is no obligation for receiving mail servers to filter their incoming mail based on SPF, which they may avoid to avoid going through multiple DNS lookups.

There are attempts at understanding the problem and tracking the fraudsters down. For example, the
University of Alabama at Birmingham has established in 2007 the Computer Forensics Research Lab to store spam mails in a Spam Data Mine available for investigators. The lab has developed automated and manual techniques to analyze the mail identity, the phishing website, and determine the tool kits used to creating the phishing website. The university’s Phishing Operations work with law enforcement and corporate investigators to identify groups or related phishing sites, particularly when the financial loss is significant (Levin et al., 2012).

Finally, there are websites that help distinguish between legitimate e-mails and traps designed to steal personal information and/or money. PayPal offers a specific address (spoof@paypal.com) where users can inquire if the message purportedly originating from PayPal is authentic.

3.21 Summary

There are two types of attacks: passive and active. Protection can be achieved with suitable mechanisms and appropriate policies. Recently, security has leaped to the forefront in priority because of changes in the regulatory environment and in technology. The fragmentation of operations that were once vertically integrated increased the number of participants in end-to-end information transfer. In virtual private networks, customers are allowed some control of their part of the public infrastructure. Finally, security must be retrofitted in IP networks to protect from the inherent difficulties of having user traffic and network control traffic within the same pipe.

Security mechanisms can be implemented in one or more layers of the OSI model. The choice of the layer depends on the security services to be offered and the coverage of protection.

Confidentiality guarantees that only the authorized parties can read the information transmitted. This is achieved by cryptography, whether symmetric or asymmetric. Symmetric cryptography is faster than asymmetric cryptography but has a limitation in terms of the secure distribution of the shared secret. Asymmetric (or public key) cryptography overcomes this problem; this is why both can be combined. In online systems, public key cryptography is used for sending the shared secret that can be used later for symmetric encryption. Two public key schemes used for sharing the secrets are Diffie–Hellman and RSA. As mentioned earlier, ISAKMP is a generic framework to negotiate point-to-point security and to exchange key and authentication data among two parties.

Data integrity is the service for preventing nonauthorized changes to the message content during transmission. A one-way hash function is used to produce a signature of the message that can be verified to ascertain integrity. Blind signature is a special procedure for signing a message without revealing its content.

The identification of participants depends on whether cryptography is symmetric or asymmetric. In asymmetric schemes, there is a need for authentication using certificates. In the case of human users, biometric features can be used for identification in specific situations. Kerberos is an example of a distributed system for online identification and authentication using symmetric cryptography.

Access control is used to counter the threats of unauthorized operations. There are two types of access control mechanisms: identity based and role based. Both can be managed through certificates defined by ITU-T Recommendation X.509. Denial of service is the consequence of failure of access control. These attacks are inherently associated with IP networks, where network control data and user data share the same physical and logical bandwidths. The best solution is to authenticate all communications by means of trusted certificates. Short of this, defense mechanisms will be specific to the problem at hand.

Nonrepudiation is a service that prevents a person who has accomplished an act from denying it later. This is a legal concept that is defined through legislation. The service comprises the generation of evidence and its recording and subsequent verification. The technical means to ensure nonrepudiation include electronic signature of documents, the intervention of third parties as witnesses, time stamping, and sequence numbering of the transactions.

3A Appendix: Principles of Symmetric Encryption

3A.1 Block Encryption Modes of Operation

The principal modes of operation of block ciphers are: electronic codebook (ECB) mode, cipher block chaining (CBC) mode, cipher feedback (CFB) mode, output feedback (OFB) mode, and counter (CTR) mode (National Institute of Standards and Technology, SP 800-38A, 2001).

The ECB mode is the most obvious, because each clear block is encrypted independently of the other blocks. However, this mode is susceptible to attacks by replay or reordering blocks, without detection. This is the reason this mode is only used to encrypt random data, such as the encryption of keys during authentication.
Some recent examples of the incorrect use of ECB is the Version 1 of Bitmessage, a peer-to-peer messaging system built on Bitcoin, which made it vulnerable (Buterin, 2012; Lerner, 2012; Warren, 2012).

The other three modes use a feedback loop protect against such types of attacks. They also have the additional property that they need an initialization vector to start the computations. This is a dummy initial ciphertext block whose value must be shared with the receiver. It is typically a random number or generated by encrypting a nonce (Ferguson et al., 2010, pp. 66–67). The difference among the three feedback modes resides in the way the clear text is mixed, partially or in its entirety, with the preceding encrypted block.

In the CBC mode, the input to the encryption algorithm is the exclusive OR of next block of plain text and the preceding block of the ciphertext. This is called “chaining” the plaintext blocks, as shown in Figure 3.15. Figure 3.16 depicts the decryption operation. In these figures, $P_i$ represents the $i$th block of the clear message, while $C_i$ is the corresponding encrypted block. Thus, the encrypted block $C_i$ is given by

$$C_i = E_k (P_i \oplus C_{i-1}), \quad i = 0, 1, \ldots$$

where

$E_k()$ represents the encryption with the secret key $K$ and $\oplus$ is the exclusive OR operation.
The starting value $C_i$ is the initialization vector. The initialization vector does not need to be secret but it must be unpredictable and its integrity protected. For example, it can be generated by applying $E_K$ to a nonce (a contraction of “number used once”) or by using a random number generator (National Institute of Standards and Technology, 2001, p. 20). Also, the CBC mode requires the input to be a multiple of the cipher’s block size, so padding may be needed.

The decryption operation, shown in Figure 3.16, is described by

$$P_i = C_{i-1} \oplus D_K(C_i)$$

Any subset of a CBC message will be decrypted correctly. The CBC mode is efficient in the sense that a stream of infinite length can be processed in a constant memory in linear time. It is useful for non-real-time encryption of files and to calculate the signature of a message (or its MAC) as specified for financial and banking transactions in ANSI X9.9 (1986), ANSI X9.19 (1986), ISO 8731-1 (1987), and ISO/IEC 9797-1 (1999). Transport Layer Security (TLS) also uses the CBC mode but, in general, the CFB and OFB modes are often used for the real-time encryption of a character stream, such as in the case of a client connected to a server.

In the CFB mode, the input is processed $s$ bits at a time. A clear text block of $b$ bits is encrypted in units of $s$ bits ($s = 1, 8$, or 64 bits), with $s \leq b$, that is, in $n = [s/b]$ cycles, where $[x]$ is the smallest integer less than $x$. The clear message block, $P_i$, is divided into $n$ segments, $\{P_{i_1}, P_{i_2}, ..., P_{i_n}\}$ of $s$ bits each; with extra bits padded to the trailing end of the data string if needed. In each cycle, an input vector $I_i$ is encrypted into an output $O_i$. The segment $P_i$ is combined through an exclusive OR function, with the most significant $s$ bits of that output $O_i$ to yield the ciphertext $C_i$ of $s$ bits. The ciphertext $C_i$ is fed back and concatenated to the previous input $I_i$ in a shift register, and all the bits of this register are shifted $s$ positions of the left. The $s$ left-most bits of the register are ignored, while the remainder of the register content becomes the new input vector $I_{i+1}$ to be encrypted in the next round. The CFB encryption is described as follows:

$$I_i = \text{LSB}_{b-s}(I_{i-1}) \parallel C_{i-1}$$

$$O_i = E_K(I_i)$$

$$C_i = P_i \oplus \text{MSB}_s(O_i)$$

where

- $I_0$ is the initialization vector
- $\text{LSB}_j(x)$ is the least significant $j$ bits of $x$
- $\text{MSB}_j(x)$ is the most significant $j$ bits
- $E_K()$ represents the encryption with the secret key $K$

The decryption operation is identical to the roles of $P_i$ and $E_i$ transposed, that is,

$$I_i = \text{LSB}_{b-s}(I_{i-1}) \parallel C_{i-1}$$

$$O_i = E_K(I_i)$$

$$P_i = C_i \oplus \text{MSB}_s(O_i)$$

Depicted in Figure 3.17 is the encryption and illustrated in Figure 3.18 is the decryption.

Similar to CBC encryption, the initialization vector need not be secret but is preferably unpredictable and needs to be changed for each message. However, the chaining mechanism causes the ciphertext block $C_i$ to depend on both $P_i$ and the preceding $O_i$. The decryption operation is sensitive to bit errors, because one bit error in the encrypted text affects the decryption of $n$ blocks.

![FIGURE 3.17 Encryption in the CFB mode of a block of $b$ bits and $s$ bits of feedback.](image-url)
If \( s = b \), the shift register can be eliminated and the encryption is done as illustrated in Figure 3.19. Thus, the encrypted block \( C_i \) is given by

\[ C_i = P_i \oplus E_K(C_{i-1}) \]

where \( E_K() \) represents the encryption with the secret key \( K \). The decryption is obtained with another exclusive OR operation as follows:

\[ P_i = C_i \oplus E_K(C_{i-1}) \]

which is shown in Figure 3.20.

The CFB mode can be used to calculate the MAC of a message. This method is also indicated in ANSI X9.9 (1986) for the authentication of banking messages, as well as ANSI X9.19 (1986), ISO 8731-1 (1987), and ISO/IEC 9797-1 (1999).

For the ECB, CBC, and CFB modes, the plaintext must be a sequence of one or more complete data blocks. If the data string to be encrypted does not satisfy this property, some extra bits, called padding, are appended to the plaintext. The padding bits must be selected so that they can be removed unambiguously at the receiving end. For example, it can consist of an octet with value 128 and then as many octets as required to complete the block (Ferguson et al., 2010, p. 64).

In the OFB mode, the input to the block cipher is not the clear text but a pseudo-random stream, called the key stream, which is derived from the ciphertext. The clear text message itself is not an input to the block cipher. This encryption scheme is called a “stream cipher.” The encryption of the OFB mode is described as follows:

\[ I_i = O_{i-1} \]
\[ O_i = E_K(I_i) \]
\[ C_i = P_i \oplus MSB_i(O_i) \]

where \( I_0 \) is the initialization vector. Each initialization vector must be used once, otherwise confidentiality may be compromised. In a stream cipher, there is no padding so if the last output block is a partial block of \( u \) bits, only the most significant \( u \) bits of that block are used in the exclusive OR operation. The lack of padding reduces the overhead, which is especially important with small messages.
Protocols for Secure Electronic Commerce

The decryption process is exactly the same operation as encryption and is described as

\[ I_i = O_{i-1} \]

\[ O_i = E_k (I) \]

\[ P_i = C_i \oplus MSB_s (O_i) \]

This is illustrated in Figures 3.21 and 3.22 for the encryption and decryption, respectively.

In the OFB mode, the input to the decryption depends on the preceding output only (i.e., it does not include the previous ciphertext), so errors are not propagated. This makes it suitable to situations where transmission is noisy. In this case, a single bit error in the ciphertext affects only one bit in the recovered text, provided that the values in the shift registers at both ends remain identical to maintain synchronization. Thus, any system that incorporates the OFB mode must be able to detect synchronization loss and have a mechanism to reinitialize the shift registers on both sides with the same value.

In the case where \( s = b \), the encryption is illustrated in Figure 3.23 and is described by

\[ O_i = E_k (O_{i-1}) \]

\[ C_i = P_i \oplus O_i \]

The decryption is described by

\[ O_i = E_k (O_{i-1}) \]

\[ P_i = C_i \oplus O_i \]

and is depicted in Figure 3.24.
For security reasons, the OFB mode with $s = b$, that is, with the feedback size equal to the block size, is recommended (National Institute of Standards and Technology, 2001b; Barthélemy et al., 2005, p. 98).

Finally, the Counter (CTR) mode uses a set of input blocks, called counters, instead of the initialization vector. A cipher is applied to the input blocks to produce a sequence of output blocks that are used to produce the ciphertext. Given a sequence of counters, $T_1, T_2, \ldots, T_n$, the encryption operation is defined in SP 800-38A as follows (National Institute of Standards and Technology, 2001b):

$$O_i = E_K(T_i)$$

$$C_i = P_i \oplus O_i$$
Each counter in the sequence must be different from every other, that is, never repeat, for any given key. According to Appendix B of SP 800-38A, the counter block can be either a simple sequential counter that is incremented started from an initial string (i.e., $T_i = T_i + j \mod 2^k$) or a combination of a nonce for each message and a counter. For a 128-bit block size, a typical approach is to use $T_i$ as the concatenation of a 48-bit message number, a 16-bit of additional nonce data and 64 bits for the block counter $i$ so that one key can be used for encrypting at most $2^{48}$ different messages (Ferguson et al., 2010, p. 70). Because the nonce must be changed for every message, the plain text message is limited to $(2^{64} \times 128)/8 = 2^{68}$ octets. Should the last output block be a partial block of $u$ bits, only the most significant $u$ bits of that block are used in the exclusive OR operation, that is, there is no need for padding.

The decryption operation is as follows:

$$O_i = E_k (T_i)$$
$$P_i = C_i \oplus O_i$$

In both CTR encryption and CTR decryption, the output blocks $O_i$ can be calculated in parallel and before the plaintext or the ciphertext are available. Also, any message block $P_i$ can be recovered independently from any other message blocks, provided that the corresponding counter block is known; this allows parallel encryption, which makes suitable for high-speed data transmission. Another advantage is that the last block can be arbitrary and no padding is needed.

The CCM mode (Counter with CBC-MAC) is a derived mode of operation that provides both authentication and confidentiality for cryptographic block ciphers with a block length of 128 bits, such as the Advanced Encryption Standard (AES). In this way, the CCM mode avoids the need to use two systems: a MAC for authentication and a block cipher encryption for privacy. This results in a lower computational cost compared to the application of separate algorithm for both.

The inputs to the encryption are a nonce $N_i$, some additional data $A$, for example, network packet header, and the plaintext $P$ of length $m$. The length of $N$, $k$, $< k$, the length of the block $k$, and is a multiple of 8 between 56 and 112 bits while the tag length $k$ is a multiple of 16 between 32 and 128 bits. The data and the plaintext are authenticated while the CTR mode encryption is applied to the plaintext only. First, a tag $T$ of length $k$ is calculated as

$$T = CBC - MAC(N\|A\|P)$$

CBC-MAC acts on blocks of bit length $k$, (typically 128 bits) and so if the last block has $j$ bits, after the last full block has been encrypted, the ciphertext is encrypted again and the left-most $j$ bits of the encrypted ciphertext is exclusive ORed with the short block of the message to generate the tag $T$. Next, the tag $T$ and the message $P$ are encrypted with the CTR mode to form a ciphertext $c$ of length $m + k$. The nonce $N$ must not have been used in a previous CCM encryption during the lifetime of the key. The counters blocks $CTR_i$ are generated from the nonce $N$ using a function $\pi$ as follows:

$$CTR_i = \pi(i, N, P, A)$$

This mode was developed to avoid the intellectual property issues around the use of another mode, the Offset Codebook (OCB) mode proposed for the IEEE 802.11i standard. The OCB mode, however, is still optional in that standard. The CCM mode is described in RFC 3610 (2003).

### 3A.2 Examples of Symmetric Block Encryption Algorithms

#### 3A.2.1 DES and Triple DES

DES, also known as the Data Encryption Algorithm (DEA), was widely used in the commercial world for applications such as the encryption of financial documents, the management of cryptographic keys, and the authentication of electronic transactions. The algorithm was developed by IBM and then adopted as a U.S. standard in 1977. It was published in FIPS 81, then adopted by ANSI in ANSI X3.92 under the name of Data Encryption Algorithm (DEA). However, by 2008, commercial hardware costing less than $15,000 could break DES keys in less than a day on average.

DES operates by encrypting blocks of 64 bits of clear text to produce blocks of 64 bits of ciphertext. The key length is 64 bits, with 8 bits for parity control, which gives an effective length of 56 bits. The encryption and decryption are based on the same algorithm with some minor differences in the generation of subkeys.

In 2005, the National Institute of Standards and Technology (NIST) finally withdrew the DES standard following the development of the AES (Advanced Encryption Standard).

The vulnerability of DES to an exhaustive attack forced the search for an interim solution until a replacement algorithm could be developed and deployed. Given the considerable investment in the software and hardware implementations of DES, triple DES, also known as TDEA (Triple Data Encryption Algorithm),
Algorithm), is based on the use of DES three successive times (Schneier, 1996, pp. 359–360). The operation of Triple DES with two different 56-bit keys is represented in Figure 3.25.

The use of three stages doubles the effective length of the key to 112 bits. The operations “encryption–decryption–encryption” aim at preserving compatibility with DES, because if the same key is used in all operations, the first two cancel each other.

Three independent 56-bit keys are highly recommended in federal applications (National Institute of Standards and Technology, 2012c, p. 37). In this case, the operation becomes as illustrated in Figure 3.26.

3A.2.2 AES

The Advanced Encryption Standard (AES) is the symmetric encryption algorithm that replaced DES. It is published by NIST as FIPS 197 (National Institute of Standards and Technology, 2001a) and is based on the algorithm Rijndael as developed by Joan Daemen of Proton World International and Vincent Rijmen from the
Catholic University of Leuven (Katholieke Universiteit Leuven). It is a block code with blocks of 128, 192, or 256 bits. The corresponding key lengths are 128, 192, and 256 bits, respectively. It is based on finding a solution of an algebraic equation of the form

\[ y^2 = (x^3 + ax + b) \mod n \]

The selection in October 2000 came after two rounds of testing following NIST’s invitation for submission to cryptographers from around the world. In the first round, 15 algorithms were retained for evaluation. In the second round of evaluation, five finalists were retained: RC6, MARS, Rijndael, Serpent, and Twofish. All the second round algorithms showed a good margin of security. The criteria used to separate them are related to algorithmic performance: speed of computation in software and hardware implementations (including specialized chips), suitability to smart cards (low memory requirements), and so on. Results from the evaluation and the rational for the selection have been documented in a public report by NIST (Nechvatal et al., 2000).

SAFER+ (Secure And Fast Encryption Routine) was one of the candidate block ciphers, based on a nonproprietary algorithm developed by James Massey for Cylink Corporation (Schneier, 1996, pp. 339–341). An enhanced version of that algorithm SAFER-128 is used in Bluetooth networks for confidentiality and mutual authentication.

AES CGM (Galois/Counter Mode) and AES CCM (Counter with Cipher Block Chaining—Message Authentication Code Mode) are two modes of operation of AES with key lengths of 128, 192, or 256 bits (National Institute of Standards and Technology, 2001b, 2007). These modes provideAuthenticated Encryption with Associated Data (AEAD), where the plaintext is simultaneously encrypted and its integrity protected. The input may be of any length but the ciphered output is generally larger than the input because of the integrity check value. CCM is described in RFC 5116 (2008). AES-CGM provides an effective defense against side-channel attacks but at the expense of performance.

As of now, there are no known attacks that would break a correct implementation of encryption with AES. AES is free of royalties and highly secure, and today, it is often the first choice for symmetric encryptions.

The AES competition is generally viewed as having provided a tremendous boost to the cryptographic research community’s understanding of block ciphers, and a tremendous increase in confidence in the security of some block ciphers.

3A.2.3 RC4

The stream cipher RC4 was originally designed by Ron Rivest and became public in 1994. It uses a variable key length ranging from 8 to 2048 bits. It has the advantage of being extremely fast when implemented in software; as a consequence, it is commonly used in cryptosystems such as SSL, TLS, the wireless security protocols WEP (Wired Equivalent Privacy) of IEEE 802.11, WPA (Wi-Fi Protected Access) of IEEE 802.11i and some Kerberos encryption modes used in Microsoft Windows.

The statistical distribution of single octets as well as in the entire RC4-encrypted stream shows deviations from a uniform distribution. In other words, there are strong biases toward particular values at specific positions and some bit patterns occur in the output stream more frequently than others. In 2013, it was shown how to use the 65,536 single octet statistical biases in the initial 256 octets of the RC4 ciphertext to mount an attack on SSL/TLS via Bayesian analysis (AlFardan et al., 2013c). The attack, however, requires access to between \(2^{28}\) and \(2^{32}\) copies of the same data encrypted using different keys. This could be done with a browser infected with a JavaScript malware to make these many connections to a server and, accordingly, give the attacker enough data. This attack may not be always practical, but it is quite possible that other attacks are known but not revealed, which may explain why RC4 was never approved as a Federal Information Processing Standard (FIPS).

3A.2.4 New European Schemes for Signature, Integrity, and Encryption

The following algorithms have been selected by the New European Schemes for Signature, Integrity, and Encryption (NESSIE) project:

1. MISTY: This is a block encryption algorithm. The block size is 64 bits and the key is 128 bits.
2. AES with 128 bits.
3. Camellia: The block length is fixed at 128 bits but the key size can be 128, 192, or 256 bits.
4. SHACAL-2: The block size is 256 bits and the key is 512 bits.

Camellia is also used in Japanese government systems and is included in the specifications of the TV-Anytime Forum for high-capacity storage in consumer systems.

3A.2.5 eSTREAM

In 2004, ECRYPT, a Network of Excellence funded by the European Union, announced the eSTREAM competition to select new stream ciphers suitable for widespread adoption. This call attracted 34 submissions. Following hundreds of security and performance evaluations, the eSTREAM committee selected a portfolio containing several stream ciphers.
3A.2.6 IDEA
IDEA was invented by Xuejia Lai and James Massey circa 1991. The algorithm takes blocks of 64 bits of the clear text, divides them into subblocks of 16 bits each, and encrypts them with a key 128 bits long. The same algorithm is used for encryption and decryption. IDEA is clearly superior to DES but has not been a commercial success. The patent is held by a Swiss company Ascom-Tech AG and is not subject to U.S. export control.

3A.2.7 SKIPJACK
SKIPJACK is an algorithm developed by the NSA for several single chip processors such as Clipper, Capstone, and Fortezza. Clipper is a tamper-resistant very large-scale integration (VLSI) chip used to encrypt voice conversations. Capstone provides the cryptographic functions needed for secure e-commerce and is used in Fortezza applications. SKIPJACK is an iterative block cipher with a block size of 64 bits and a key of 80 bits. It can be used in any of the four modes ECB, CBC, CFB (with a feedback of 8, 16, 32, or 64 bits), and OFB with a feedback of 64 bits.

3B Appendix: Principles of Public Key Encryption
The most popular algorithms for public cryptography are those of Rivest, Shamir, and Adleman (RSA) (1978), Rabin (1979), and ElGamal (1985). Nevertheless, the overwhelming majority of proposed systems in commercial systems are based on the RSA algorithm.

It should be noted that RSADSI was founded in 1982 to commercialize the RSA algorithm for public key cryptography. However, its exclusive rights ended with the expiration of the patent on September 20, 2000.

3B.1 RSA
Consider two odd prime numbers \( p \) and \( q \) whose product \( N = p \times q \). \( N \) is the modulus used in the computation that is public, while the values \( p \) and \( q \) are kept secret.

Let \( \phi(N) \) be the Euler totient function of \( N \). By definition, \( \phi(N) \) is the number of elements formed by the complete set of residues that are relatively prime to \( N \). This set is called the reduced set of residues modulo \( N \).

If \( N \) is a prime, \( \phi(N) = N - 1 \). However, because \( N = p \times q \) by construction, while \( p \) and \( q \) are primes, then

\[
\phi(N) = (p - 1)(q - 1)
\]

According to Fermat’s little theorem, if \( m \) is a prime and \( a \) is not a multiple of \( m \), the

\[
a^{m-1} \equiv 1 \mod m
\]

Euler generalized this theorem in the form

\[
a^{\phi(N)} \equiv 1 \mod N
\]

Choose the integers \( e, d \) both less than \( \phi(N) \) such that the greatest common divisor of \( (e, \phi(N)) = 1 \) and \( e \times d \equiv 1 \mod (\phi(N)) = 1 \mod ((p - 1)(q - 1)) \).

Let \( X, Y \) be two numbers less than \( N \)

\[
Y = X^e \mod N \quad \text{with } 0 \leq X < N \\
X = Y^d \mod N \quad \text{with } 0 \leq Y < N
\]

because, by applying Fermat’s little theorem,

\[
Y^d \mod N = (X^e)^d \mod N = X^{ed} \mod N = X^{\phi(N)} \equiv 1 \mod N
\]

To start the process, a block of data is interpreted as an integer. To do so, the total block is considered as an ordered sequence of bits (of length, say, \( \lambda \)). The integer is considered to be the sum of the bits by giving the first bit the weight of \( 2^{\lambda-1} \), the second bit the weight of \( 2^{\lambda-2} \), and so on until the last bit that will have the weight of \( 2^0 = 1 \).

The block size must be such that the largest number does not exceed the modulo \( N \). Incomplete blocks must be completed by padding bits with either 1 or 0 bits. Further padding blocks may also be added.

The public key of the algorithm \( PK \) is the number \( e \), along with \( n \), while the secret key \( SK \) is the number \( d \). RSA achieves its security from the difficulty of factoring \( N \). The number of bits of \( N \) is considered to be the key size of the RSA algorithm. The selection of the primes \( p \) and \( q \) must make this factorization as difficult as possible.

Once the keys have been generated, it is preferred that, for reasons of security, the values of \( p \) and \( q \) and all intermediate values such as the product \( (p - 1)(q - 1) \) be deleted. Nevertheless, the preservation of the values of \( p \) and \( q \) locally can double or even quadruple the speed of decryption.

3B.1.1 Chosen-Ciphertext Attacks
It is known that plain RSA is susceptible to a chosen-ciphertext attack (Davida, 1982). An attacker who wishes...
to find the decryption of a message \( m \equiv c^d \mod N \) of a ciphertext \( c \) chooses a random integer \( s \) and asks for the decryption of \( c' \equiv s^e c \mod N \). The answer is \((s^e c')^d \mod N = 1 \times c^d \mod N = c^d \mod N\).

It is also known that a number of attacks are based on the fact that around \( \frac{1}{4} \) of the bits of the private key are leaked, such as by timing and power analysis attacks. Also, the strength of cryptosystem depends on the choice of the primes (Pellegrini et al., 2010).

### 3B.1.2 Practical Considerations

To increase the speed of signature verification, suggested values for the exponent \( e \) of the public key are 3 or \( 2^{16} + 1 \) (65,537) (Menezes et al., 1997, p. 437). Other variants designed to speed up decryption and signing are discussed in Boneh and Shacham (2002).

For short-term confidentiality, the modulus \( N \) should be at least 768 bits. For long-term confidentiality (5–10 years), at least 1024 bits should be used. Currently, it is believed that confidentiality with a key of 2048 bits would last about 15 years.

In practice, RSA is used with padding schemes to avoid some of the weaknesses of the RSA algorithm as described earlier. Padding techniques such as Optimal Asymmetric Encryption Padding (OAEP) are standardized as discussed in the next section.

### 3B.2 Public Key Cryptography Standards

The public key cryptography standards (PKCS) are business standards developed by RSA Laboratories in collaboration with many other companies working in the area of cryptography. They have been used in many aspects of public key cryptography, which are based on the RSA algorithm. At the time of writing this section, their number has reached 15.

PKCS #1 (RFC 2437 [1998]; RFC 3447 [2003]) defines the mechanisms for data encryption and signature using the RSA algorithm. These procedures are then utilized for constructing the signatures and electronic envelopes described in PKCS #7. Following the presentation of an adaptive chosen-ciphertext attack on PKCS #1 (Bleichenbacher, 1998), more secure encodings are available with PKCS #1v1.5 and PKSC #1v2.1, the latter of which is defined in RFC 3447. In particular, PKCS #1v2.1 defines an encryption scheme based on the Optimal Asymmetric Encryption Padding (OAEP) of Bellare and Rogaway (1994). PKCS #2 and #4 are incorporated in PKCS #1.

PKCS #3 defines the key exchange protocol using the Diffie–Hellman algorithm.

PKCS #5 describes a method for encrypting an information using a secret key derived from a password. For hashing, the method utilizes either MD2 or MD5 to compute the key starting with the password and then encrypts the key with DES in the CBC mode.

PKCS #6 defines a syntax for X.509 certificates.

PKCS #7 (RFC 2315, 1998 defines the syntax of a message encrypted using the basic encoding rules [BER] of ASN.1 [Abstract Syntax Notation 1]) (Steeman, 1993) of ITU-T Recommendation X.209 (1988). These messages are formed with the help of six content types:

1. **Data**, for clear data
2. **SignedData**, for signed data
3. **EnvelopedData**, for clear data with numeric envelopes
4. **SignedAndEnvelopedData**, for data that are signed and enveloped
5. **DigestedData**, for digests
6. **EncryptedData**, for encrypted data

The secure messaging protocol, S/MIME (Secure Multipurpose Internet Mail Extensions) and the messages of the SET protocol, designed to secure bank card payments over the Internet, utilize the PKSC #7 specifications.

PKCS #8 describes a format for sending information related to private keys.

PKCS #9 defines the optional attributes that could be added to other protocols of the series. The following items are considered: the certificates of PKCS #6, the electronically signed messages of PKCS #7, and the information on private keys as defined in PKCS #8.

PKCS #10 (RFC 2986, 2000) describes the syntax for certification requests to a certification authority. The certification request must contain details on the identity of the candidate for certification, the distinguished name of the candidate, his or her public key, and optionally, a list of supplementary attributes, a signature of the preceding information to verify the public key, and an identifier of the algorithm used for the signature so that the authority could proceed with the necessary verifications. The version adopted by the IETF is called CMS (Cryptographic message syntax).

PKCS #11 defines a cryptographic interface called Cryptoki (Cryptographic Token Interface Standard) between portable devices such as smart cards or PCMCIA cards and the security layers.

PKCS #12 describes a syntax for the storage and transport of public keys, certificates, and other users’ secrets. In enterprise networks, key pairs are transmitted to individuals via password protected PKCS #12 files.

PKCS #13 describes a cryptographic system using elliptic curves.

PKCS #15 describes a format to allow the portability of cryptographic credentials such as keys, certificates,
passwords, PINs among application and among portable devices such as smart cards.

The specifications of PKCS #1, #7, and #10 are not IETF standards because they mandate the utilization of algorithms that RSADSI does not offer free of charge. Also note that in PKCS #11 and #15, the word *token* is used to indicate a portable device capable of storing persistent data.

### 3B.3 PGP and OpenPGP

Pretty Good Privacy is considered to be the commercial system whose security is closest to the military grade. It is described in one of the IETF documents, namely, RFC 1991 (1996). PGP consists of six functions:

1. Public key exchange using RSA with MD5 hashing
2. Data compression with ZIP, which reduces the file size and redundancies before encryption (Reduction of the size augments the speed for both processing and transmission, while reduction of the redundancies makes cryptanalysis more difficult.)
3. Message encryption with IDEA
4. Encryption of the user’s secret key using the digest of a sentence instead of a password
5. ASCII “armor” to protect the binary message for any mutilations that might be caused by Internet messaging systems (This armor is constructed by dividing the bits of three consecutive octets into four groups of 6 bits each and then by coding each group using a 7-bit character according to a given table. A checksum is then added to detect potential errors.)
6. Message segmentation

The IETF did not adopt PGP as a standard because it incorporates proprietary protocols. OpenPGP is based on PGP but avoids these intellectual property issues. It is specified in RFCs 4880 (2007), RFC 6637 (2012) describes how to use elliptic curve cryptography (ECC) with OpenPGP.

OpenPGP is currently the most widely used e-mail encryption standard. Companies and organizations that implement OpenPGP have formed the OpenPGP Alliance to promote it and to ensure interoperability. The Free Software Foundation has developed its own OpenPGP conformant program called GNU Privacy Guard (abbreviated GnuPG or GPG). GnuPG is freely available together with all source code under the GNU General Public License (GPL).

### 3B.4 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a public key cryptosystem where the computations take place on an elliptic curve. Elliptic curves have been applied in factoring integers, in proving primality, in coding theory and in cryptography (Menezes, 1993). Variants of the Diffie–Hellman and DSA algorithms on elliptic curves are the Elliptic Curve Diffie–Hellman algorithm (ECDH) and the Elliptic Curve Digital Signature Algorithm (ECDSA), respectively. They are used to create digital signatures and to establish keys for symmetric cryptography. Diffie–Hellman and ECDH are comparable in speed, but RSA is much slower. The advantage of elliptic curve cryptography is that key lengths are shorter than for existing public key schemes that provide equivalent security. For example, the level of security of 1024-bit RSA can be achieved will elliptic curves with a key size in the range of 171–180 bits (Wiener, 1998). This is an important factor in wireless communications and whenever bandwidth is a scarce resource.

Elliptic curves are defined over the finite field of the integers modulo a primary number \( p \) [the Galois field \( \text{GF}(p) \)] or that of binary polynomials \( \text{GF}(2^m) \). The key size is the size of the prime number or the binary polynomial in bits. Cryptosystems over \( \text{GF}(2^m) \) appear to be slower than over \( \text{GF}(p) \) but there is no consensus on that point. Their main advantage, however, is that addition over \( \text{GF}(2^m) \) does not require integer multiplications, which reduces the cost of the integrated circuits implementing the computations.

NIST has standardized a list of 15 elliptic curves of varying key lengths (NIST, 1999). Ten of these curves are for binary fields and five are for prime fields. These curves provide confidentiality equivalent to symmetric encryption with keys of length 80, 112, 128, 192, and 256 bits and beyond.

Table 3.9 shows the comparison of the key lengths of RSA and elliptic cryptography for the same level of

<table>
<thead>
<tr>
<th>Reduction Factor</th>
<th>RSA</th>
<th>Elliptic Curve</th>
<th>RSA/ECC</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>106</td>
<td></td>
<td>5:1</td>
</tr>
<tr>
<td>1,024</td>
<td>160</td>
<td></td>
<td>7:1</td>
</tr>
<tr>
<td>2,048</td>
<td>211</td>
<td></td>
<td>10:1</td>
</tr>
<tr>
<td>5,120</td>
<td>320</td>
<td></td>
<td>16:1</td>
</tr>
<tr>
<td>21,000</td>
<td>600</td>
<td></td>
<td>35:1</td>
</tr>
</tbody>
</table>

TABLE 3.10
NIST Recommended Key Lengths in Bits

<table>
<thead>
<tr>
<th>Symmetric Key Size in Bits</th>
<th>RSA and Diffie–Hellman Cost: Elliptic Curve Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>1,024</td>
</tr>
<tr>
<td>112</td>
<td>2,048</td>
</tr>
<tr>
<td>128</td>
<td>3,072</td>
</tr>
<tr>
<td>192</td>
<td>7,680</td>
</tr>
<tr>
<td>256</td>
<td>15,360</td>
</tr>
</tbody>
</table>

Source: Barker, E. et al., National Institute of Standards and Technology (NIST), Recommendation for Key Management – Part 1: General, (Revision 3), NIST Special Publication SP 800-57, July 2012c.

security measured in terms of effort to break the system (Menezes, 1993).

Table 3.10 gives the key sizes recommended by the National Institute of Standards and Technology for equivalent security using symmetric encryption algorithms (e.g., AES, DES or SKIPJACK) or public key encryption with RSA, Diffie–Hellman, and elliptic curves for equivalent security.

Thus, for the same level of security per the currently known attacks, elliptic curve–based systems can be implemented with smaller parameters. This computational efficiency is compared to Diffie–Hellman in Table 3.11 for several key sizes.

Another related aspect is the channel overhead for key exchanges and digital signatures on a communications link. In channel-constrained environments or when computing power, memory and battery life of devices are critical such as in wireless communication, elliptic curves offer a much better solution than RSA or Diffie–Hellman. As a result, many companies in wireless communications link. In channel-constrained environments or when computing power, memory and battery life of devices are critical such as in wireless communication, elliptic curves offer a much better solution than RSA or Diffie–Hellman. As a result, many companies in wireless communications have embraced elliptic curve cryptography. Furthermore, many nations (e.g., the United States, the United Kingdom, and Canada) have adopted elliptic curve cryptography in the new generation of equipment to protect classified information.

One disadvantage of ECC is that it increases the size of the encrypted message significantly more than RSA encryption. Furthermore, the ECC algorithm is more complex and more difficult to implement than RSA, which increases the likelihood of implementation errors, thereby reducing the security of the algorithm. Another problem is that various aspects of elliptic curve cryptography have been patented notable the Canadian company Certicom, which holds over 130 patents related to elliptic curves and public key cryptography in general.

TABLE 3.11
Relative Computation Costs of Diffie–Hellman and Elliptic Curves

<table>
<thead>
<tr>
<th>Symmetric Key Size in Bits</th>
<th>Diffie–Hellman Cost: Elliptic Curve Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>3:1</td>
</tr>
<tr>
<td>112</td>
<td>6:1</td>
</tr>
<tr>
<td>128</td>
<td>10:1</td>
</tr>
<tr>
<td>192</td>
<td>32:1</td>
</tr>
<tr>
<td>256</td>
<td>64:1</td>
</tr>
</tbody>
</table>


3C Appendix: Principles of the Digital Signature Algorithm and the Elliptic Curve Digital Signature Algorithm

According to the Digital Signature Algorithm (DSA) defined in ANSI X9.30-1 (1997), the signature of a message $M$ is the pair of numbers $r$ and $s$ computed as follows:

$$r = (g^k \mod p)$$

and

$$s = (k^{-1} \times [\text{SHA}(M) + x r]) \mod q$$

The following are given:

- $p$ and $q$ are primes such that $2^{511} < p < 2^{1024}$, $2^{159} < q < 2^{160}$, and $q$ is a prime divisor of $(p − 1)$, that is, $(p − 1) = mq$ for some integer $m$.
- $g^q = h^{(p−1)/q} \mod p$ is a generator polynomial modulo $p$ of order $q$. The variable $h$ is an integer $1 < h < (p−1)$ such that $h^{(p−1)/q} \mod p > 1$. By Fermat’s little theorem, $g^q = h^{(p−1)} \mod p$ because $g < p$. Thus, each time the exponent is a multiple of $q$, the result will be equal to $1 \mod p$.
- $x$ is a random integer such that $0 < x < q$.
- $k$ is a random integer in the interval $0 < k < q$ and is different for each signature.
- $k^{-1}$ is the multiplicative inverse of $k$ modulo $q$, that is, $(k^{-1} \times k) \mod q = 1$.
- SHA() is the SHA-1 hash function.
- $x$ is the private key of the sender, while the public key is $(p, q, g, y)$ with $y = g^x \mod p$. 
The DSA signature consists of the pair of integers \((r, s)\). To verify the signature, the verifier computes

\[
\begin{align*}
    w &= s^{-1} \mod q \\
    u_1 &= SHA(M)w \mod q \\
    u_2 &= r \cdot w \mod q \\
    v &= (g^w \cdot y^{u_2} \mod p) \mod q
\end{align*}
\]

The signature is valid if \(v = r\). To show this, we have

\[
\begin{align*}
    v &= \left( g^{SHA(M)w} \cdot y^{u_2w} \mod p \right) \mod q \\
    &= \left( g^{SHA(M)} \cdot y^{w} \right)^w \mod p \\
    &= \left( g^{x \cdot w} \right)^w \mod p \\
    &= \left( g^w \right)^w \mod p \\
    &= (g \mod p)^w \mod q \\
    &= (g^w \mod p) \mod q,
\end{align*}
\]

since the generator is of order \(q\) by construction

\[= r\]

The strength of the algorithm is heavily dependent on the choice of the random numbers.

The random variable \(k\) is also transmitted with the signature. This means that if verifiers also know the signer’s private key, they will be able to pass additional information through the channel established through the value of \(k\).

The Elliptic Curve Digital Signature Algorithm (ECDSA) of ANSI X9.62:2005 is used for digital signing, while ECDH can be used to secure online key exchange. Typical key sizes are in the range of 160–200 bits. The ECDSA keys are generated as follows (Paar and Pelzl, 2010, pp. 283–284):

- Use an elliptic curve \(E\) with modulus \(p\), coefficients \(a\) and \(b\), and a point \(A\) that generates a cyclic group of prime order \(q\), that is, \(A^q = 1 \mod p\) and all the elements of this cyclic group can be generated with \(A\).
- Choose a random integer \(y\) such that \(0 < y < q\) and compute \(B = yA\).
- The private key is \(y\) and the public key is \((p, a, b, q, A, B)\).

The ECDSA signature for a message \(M\) is computed as follows:

- Choose an integer \(k\) randomly with \(0 < k < q\).
- Compute \(R = kA\) and assign the \(x\)-coordinate of \(R\) to the variable \(r\).
- Compute \(s = (H(M) + y \cdot r)k^{-1} \mod q\), with \(H()\) a hash function.

The signature verification proceeds as follows:

- Compute \(w = s^{-1} \mod q\).
- Compute \(u_1 = w \cdot H(x) \mod q\) and \(u_2 = w \cdot r \mod q\).
- Compute \(P = u_1A + u_2B\).
- The signature is valid if and only if \(x_P\) the \(x\)-coordinate of point \(P\) satisfies the condition.

\[x_p = r \mod q.\]

Questions

1. What are the major security vulnerabilities in a client/server communication?
2. What are the services needed to secure data exchanges in e-commerce?
3. Compare the tunnel mode and transport mode of IPSec.
4. Why is the Authentication header (AH) less used with IPSec than the Encapsulating Security Payload (ESP) protocol?
5. What factors affect the strength of encryption?
6. Discuss some potential applications for blind signatures.
7. Discuss the difference between biometric identification and biometric verification.
8. Compare and contrast the following biometrics: fingerprint, face recognition, iris recognition, voice recognition, and hand geometry.
9. Discuss some of the vulnerabilities of biometric identification systems.
10. What is needed to offer nonrepudiation services?
11. What conditions favor denial-of-service attacks?
12. Which of the following items is not in a digital public key certificate?
   (a) the subject’s public key
   (b) the digital signature of the certification authority
   (c) the subject’s private key
   (d) the digital certificate serial number
13. What is cross-certification and why is it used? What caveats need to be taken considered for it to work correctly?

14. What is the Sender Policy Framework (SPF)? Describe its advantages and drawbacks.

15. What is the DomainKeys Identified Mail (DKIM)? Describe its advantages and drawbacks.

16. Using the case of AES as a starting point, define a process to select a new encryption algorithm.

17. Compare public key encryption and symmetric encryption in terms of advantages and disadvantages. How to combine the strong points of both?

18. What are the reasons of the current interest in elliptic curve cryptography (ECC)?

19. Explain why the length of the encryption key cannot be used as the sole measure for the strength of an encryption.

20. Speculate on the reasons that led to the declassification of the SKIPJACK algorithm.

21. What are the problems facing cross-certification? How are financial institutions attempting to solve them?

22. What is the counter mode of block cipher encryption? What are its advantages?

23. With the symbols as explained in Appendix 3C, show that a valid ECDSA signature satisfies the condition \( r = x_p \mod q \).

24. List the advantages and disadvantages of controlling the export of strong encryption algorithms from the United States.