7 System aspects

At the time of their initial conception, most common network protocols, such as the Transmission Control Protocol (TCP) and the Internet Protocol (IP), were not developed with security concerns in mind. When DARPA launched the first steps towards the packet-switched network that gave birth to the modern Internet, engineering efforts were targeted towards the challenges of guaranteeing reliable communication of information packets across multiple stations from the source to its final destination. The reasons for this are not difficult to identify: the deployed devices were under the control of a few selected institutions, networking and computing technology was not readily available to potential attackers, electronic commerce was a distant goal, and the existing trust among the few users of the primitive network was sufficient to allow all attention to be focused on getting a fully functional computer network up and running.

A few decades later, with the exponential growth in number of users, devices, and connections, issues such as network access, authentication, integrity, and confidentiality became paramount for ensuring that the Internet and, more recently, broadband wireless networks could offer services that are secure and ultimately trusted by users of all ages and professions. By then, however, the layered architecture, in which the fundamental problems of transmission, medium access, routing, reliability, and congestion control are dealt with separately at different layers, was already ingrained in the available network devices and operating systems. To avoid redesigning the entire network architecture subject to the prescribed security guarantees, the solutions adopted essentially resort to adding authentication and encryption to the various layers of the existing protocol stack.

In this chapter, we shall focus our attention on the security of wireless networks as a case study for the implementation of physical-layer security. This class of systems can be deemed an extreme case in that they combine the liabilities inherent to the broadcast property of the wireless medium with the many vulnerabilities of the Internet, with which they share the basic TCP/IP networking architecture and several essential communication protocols. On the basis of a critical overview of existing security mechanisms at each layer of the network architecture, we identify how physical-layer security can be integrated into the system with clear benefits in terms of confidentiality and robustness against active attacks.

7.1 Basic security primitives

Before elaborating on how each networking layer implements the standard security measures against known attacks, we discuss briefly the basic building blocks currently used in almost every security sub-system. For this purpose, we focus on the three main security services: integrity, confidentiality, and authenticity. Other services, such as non-repudiation and access control, often require solutions to be found in the realm of policy rather than among the engineering disciplines.

In most applications of relevance, the ruling paradigm is that of computational security. In other words, the designers of the system implicitly assume that the adversary has limited computing power and is thus unable to break strong ciphers or overcome mathematical problems deemed hard to solve with classical computers. Existing systems are typically secured through a mix of symmetric encryption (confidentiality), hash functions (integrity), and public-key cryptography (secret-key distribution and authentication).

7.1.1 Symmetric encryption

Under the assumption that the legitimate communication partners are in possession of a shared secret key, as illustrated in Figure 1.1, symmetric encryption algorithms (or *ciphers*) are designed to ensure that (a) the plaintext message to be sent can easily be converted into a cryptogram using the secret key, (b) the cryptogram can easily be re-converted into the plaintext with the secret key, and (c) it is very hard, if not impossible, to recover the plaintext from the cryptogram without the key in useful time (a property also referred to as a *one-way trapdoor function*). Useful time means here that an attacker must be unable to break the encryption while there is some advantage in learning the sent message.

Block ciphers achieve this goal by breaking the plaintext into blocks with a fixed number of bits and applying an array of substitutions and permutations that depend on the secret key. State-of-the-art block ciphers employ the principles of *diffusion* and *confusion*. The former ensures that each input plaintext bit influences almost all output ciphertext symbols, whereas the latter implies that it is hard to obtain the key if the ciphertext alone is available for cryptanalysis.

The most prominent examples are the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES), both of which emerged from standardization efforts by the US National Institute of Standards and Technology (NIST). The DES uses a so-called Feistel architecture, which implements diffusion and confusion by repeating the same substitution and permutation operations in multiple identical rounds, albeit with different sub-keys that are derived from the shared secret key. The AES improves over the DES by increasing the block length, implementing substitution and permutation operations that are highly non-linear, and allowing very fast execution. Special *modes of encryption* are used when (a) the size of the plaintext is not a multiple of the defined block length or (b) the same input block appears more than once and the corresponding output ciphertext blocks must be different.

Instead of dividing the input data into blocks, *stream ciphers* produce one encrypted symbol for every input symbol in a continuous fashion. This mode of operation is similar to that of the one-time pad described in Chapter 1. However, rather than mixing each input symbol with perfectly random key symbols, typical stream ciphers use pseudo-random sequences that are functions of the secret key.

Evaluating the security of symmetric encryption mechanisms is widely considered to be a difficult task. In contrast to the case of the one-time pad in information-theoretic security, there are no mathematical proofs for the assured level of secrecy. Standard practices include testing the randomness of the output ciphertext (the closer to perfect randomness the better), counting the number of operations required for a brute-force attack (i.e. trying out all possible keys) and cryptanalysis with known or chosen plaintext–ciphertext pairs. There is also growing consensus that the encryption algorithm should be made public, so that anyone can come up with and publish on efficient attacks against the most widely used ciphers. It follows that the secrecy of the encrypted information must depend solely on the secret key.

7.1.2 Public-key cryptography

The main drawback of symmetric encryption lies in the need for the legitimate communication partners to share a secret key. Under the computational security paradigm, this can be solved in an elegant way by means of *public-key cryptography* (also known as *asymmetric cryptography*). Instead of a secret key, each of the legitimate partners uses a pair of different keys, more specifically a private key, which is not shared, and a public key, which is available to the legitimate receiver and any potential attacker. The encryption algorithm is designed to ensure that the private key and the public key can be used interchangeably, i.e. a cryptogram generated with the public key can be decrypted only with the corresponding private key and vice versa. Thus, if Alice wants to send a confidential message to Bob, she can use his public key to encrypt the message knowing that only he will be able to recover the message, because no one else knows his private key. If the objective is to authenticate the message using a digital signature, Alice can encrypt the message using her private key, so that Bob can verify the sender's authenticity by decrypting with her public key.

The security of public-key cryptography relies on the computational intractability of certain mathematical operations, most notably the prime factorization of large integers or the inversion of the discrete logarithm function. The RSA scheme, named after its creators Rivest, Shamir, and Adleman, puts public-key cryptography into practice by exploiting the properties of exponentiation in a finite field over integers modulo a prime. To generate the public key and the private key, each user must first select two large primes p and q (about 100 digits each) and compute both their product n and its Euler totient function $\phi(n) = (p-1)(q-1)$. The next step is to pick a number e uniformly at random among all $z < \phi(n)$ that are prime relative to $\phi(n)$, and compute also the multiplicative inverse $d < \phi(n)$ of e modulus $\phi(n)$. The public key and the private key are then composed by (e, n) and (d, n), respectively. To encrypt a message block m, satisfying $0 \leqslant m < n$, Alice takes Bob's public key $k_{\text{pu}}^{\text{B}} = (e, n)$ and computes the

cryptogram c according to

$$c = m^e \mod n. \tag{7.1}$$

Using his private key $k_{\text{pr}}^{\text{B}} = (d, n)$, Bob can decrypt m by calculating

$$m = c^d \mod n. \tag{7.2}$$

The correctness of the encryption and decryption mechanisms is ensured by the fact that e and d are multiplicative inverses modulus $\phi(n)$. More specifically, it follows from Euler's theorem that

$$ed = 1 + k\phi(n) \tag{7.3}$$

for some k. Hence,

$$c^d = m^{ed} = m^{1+k\phi(n)} = m^1 = m. \tag{7.4}$$

The RSA scheme and similar public-key cryptography schemes typically use large keys (1024 to 2048 bits), which renders a brute-force attack practically unfeasible with classical computers. It is now widely accepted that, if stable quantum computers can be built with a moderate number of quantum bits, then it will be possible to factorize n and obtain the primes p and q in useful time. This would break hard cryptographic primitives such as those used in public-key cryptography. More recently, elliptic curves have emerged as a group upon the basis of which strong public-key encryption can be obtained with smaller key sizes. However, the vulnerabilities with respect to quantum attacks are similar.

7.1.3 Hash functions

The standard way to ensure the integrity or the message authenticity of the encrypted data is to generate a fixed-length *message digest* using a *one-way hash function*. As their names indicate, the digest is much smaller than the original data and the one-way hash function cannot be reversed. More specifically, it is expected that the hash function satisfies the following properties.

- *Pre-image-resistant.* For any hash value or digest it is hard to find a pre-image that generates that hash value.
- *Second-pre-image-resistant or weakly collision-free.* Given a hash value and the message that originated it, finding another message that generates the same value is hard.
- *Collision-resistant or strongly collision-free.* It is hard to find two messages that generate the same hash value or digest.

The digest typically results from one or more mixing and compression operations on message blocks, which are sometimes implemented using a block cipher and a special encryption mode. Flipping only one bit in the original data should already result in a very different digest with practically no correlation with the previous digest. Examples of hash functions currently in use include the SHA family of hash functions standardized by the NIST. Said cryptographic one-way hash functions should not be confused with the

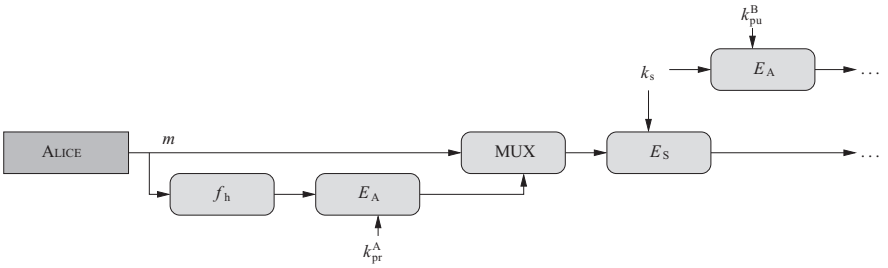

Figure 7.1 Typical security sub-system (sender side).

universal family of hash functions used in information-theoretic secret-key generation, as defined in Chapter 4.

7.1.4 Authentication, integrity, and confidentiality

In principle, the basic mechanisms of public-key cryptography would be sufficient to achieve the fundamental security goals of authentication, integrity, and confidentiality. However, since these mechanisms are very demanding from a computational point of view, most secure systems use a combination of public-key cryptography, symmetric encryption, and one-way hashing.

Figure 7.1 shows a typical solution from the point of view of the sender, Alice. She starts by generating the message digest $f_h(m)$, which she encrypts asymmetrically (E_A) with her private key k_{pr}^{A}. The fact that Alice is the only person in possession of k_{pr}^{A} assures the required sender authentication. Since the message digest is much smaller than the message, the computational overhead that comes with public-key cryptography is not a cause for concern. The authenticated digest, which corresponds to a digital signature, is then multiplexed with the original message. The output is then protected efficiently via symmetric encryption with the session key k_s. To share the session key with Bob securely, Alice uses once again public-key cryptography, this time with Bob's public key k_{pu}^{B}.

On the reception side, illustrated in Figure 7.2, Bob recovers k_s by decrypting it with his private key. He then uses this key to obtain the sent message, to which he applies the known hash function f_h. Using Alice's public key, he can decrypt the sent message digest, and compare it with the result he obtained in the previous step. If the computed hash value is equal to the decrypted message digest, then he can rest assured that the integrity of the message and the authenticity of the sender are guaranteed.

7.1.5 Key-reuse and authentication

Suppose that Alice and Bob share a secret key k. Although in theory it is possible for Alice and Bob to refresh k by generating a new random key k for every message and exchanging k' securely, the overhead incurred in real systems in terms of computation and number of transmissions forces practitioners to compromise and reuse the same key

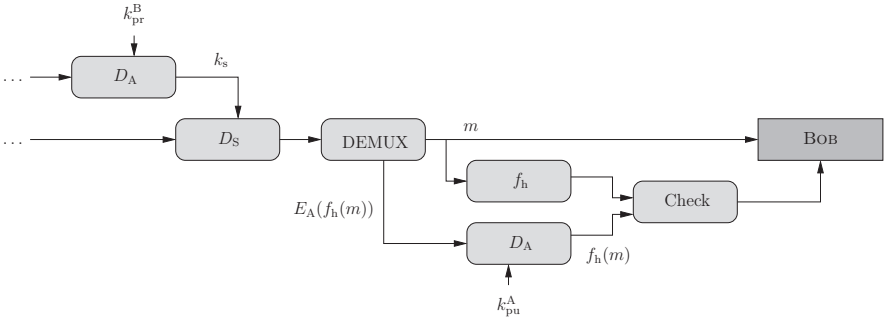

Figure 7.2 Typical security sub-system (reception side).

multiple times. Feasible alternatives include scheduled modifications by means of simple operations (e.g. a shift or an XOR with another key sequence) and re-keying by means of pseudo-random generators, which use the original key as the seed for subsequent keys.

Although changing the key in these ways intuitively makes it harder for the attacker to break the security of the system, from an information-theoretic point of view using functions of the key for encryption does not change the amount of randomness or the uncertainty of the attacker with respect to the protected messages and the secret key.

An active attacker, say Charles, with access to the sent cryptograms will seek not only to break the key but also to impersonate the legitimate sender, either by modifying intercepted messages or by generating new ones using the captured secret key. If we assume that in the worst case the attacker has unlimited computing power, the natural question is then whether Charles can make Bob believe that the faked messages come from Alice.

The scenario described here admits an information-theoretic treatment. Alice wants to send n messages M_i with $i \in [\![1, n]\!]$ in sequence at different points in time. To authenticate the messages, she uses the shared secret key K, and generates one $\mathsf{Y}_i = f(\mathsf{M}_i, \mathsf{K})$ for every M_i to be sent. At time i, Bob must use his knowledge of the key K and all past cryptograms Y_j with $j \in [\![1, i-1]\!]$ to determine whether Y_i is authentic or not. The decision process can be framed as a hypothesis test, in which Y_i was generated either by Alice or by an attacker. A type-I error occurs if Bob rejects the message even though it was actually generated by Alice. On the other hand, Bob incurs a type-II error if he accepts the message when it was generated by the attacker. If we set the probability of type-I errors to zero, i.e. if Bob is expected to accept every authentic message sent by Alice, then the probability P_{SA} that the attack is successful is lower bounded by

$$P_{\mathrm{SA}} \geqslant 2^{-\mathbb{H}(\mathsf{K})/(n+1)}. \tag{7.5}$$

In simple terms, the difficulty in guessing the key is reduced significantly with every re-utilization of the key.

Carter–Wegman universal families of hash functions can be used to ensure unconditional authentication. In other words, it is possible to build digital-signature schemes or authentication tags that cannot be forged or modified even by an attacker with infinite computing power. In precise terms, we say that the scheme is unbreakable with

Table 7.1 The TCP/IP architecture, features, and security mechanisms

Level	Layer	Tasks	Security mechanisms
5	Application	Runs processes and applications	End-to-end cryptography
4	Transport	Reliable communication and congestion control	Secure Sockets Layer (SSL) and Transport Layer Security (TLS)
3	Network	Routing and forwarding	Internet Protocol Security (IPSec)
2	Link	Medium-access control	End-to-end cryptography
1	Physical	Transmission	Spreading against narrow-band jamming

probability p if an attacker who gains access to m and the corresponding authentication tag $f(m)$ has a probability less than or equal to p of guessing the tag of another message m'. To transmit an authenticated message m, the legitimate partners must share a secret key, which consists of the message number i and a function f. The latter is chosen uniformly among a universal family of hash functions that map the message set M to the set of tags T. The fraction of functions in the same class that map m' (different from m) to a particular tag t' is $1/|T|$. Thus, setting $|T| \geqslant 1/p$ ensures that the attacker has a success probability of at most p. To authenticate multiple messages, the sender computes the authentication tag $f(m)$ and then performs a bit-wise XOR of the tag with the message number, which is also known at the receiver. The message number can be used only once. If the authentication tag is k bits long, then the attacker cannot find a message whose tag can be guessed with probability higher than $1/2^k$.

7.2 Security schemes in the layered architecture

The typical architecture of contemporary networks evolved from the Open System Interconnection (OSI) Reference Model to the prevalent five-layer TCP/IP architecture, which is depicted in Table 7.1. The tasks listed for each layer reflect the initial concerns with the reliable communication of information packets from one end of the network to the other. Processes and applications running at the top layer generate data streams, which are segmented and encapsulated into separate packets. Computers and routers then forward the packets across multiple links until they reach the destination. Communication links may exist over different physical media with multiple transmitters, thus requiring specific transmission schemes and channel arbitration among various devices.

Each layer provides the adjacent upper layer with an abstraction of the network. In simple terms, the link layer sees a channel with or without collisions of transmitted packets, the networking layer sees a graph of links with certain rates, the transport layer sees an end-to-end connection with or without packet losses, and the application layer sees a bit-pipe with or without delivery guarantees. It is striking that in their original form none of the layers takes security aspects into consideration.

To fill this gap, the X.800 standard defined a set of fundamental security services based on the original OSI networking model. As summarized in Table 7.2, these services are

Table 7.2 Physical-layer security vis-à-vis the X.800 standard

Category	Service	Objective
Authentication	Peer-entity authentication	Confidence in the identity of communication partners
	Data-origin authentication	Confidence in the source of the received data
Access control	Access control	Prevent unauthorized use of a resource
Data confidentiality	Connection confidentiality	Protect a connection
	Connectionless confidentiality	Protect a single data block
	Selective-field confidentiality	Protect selected fields on a connection or in a data block
	Traffic-flow confidentiality	Prevent information acquisition from traffic observation
Data integrity	Connection integrity with recovery	Detect and correct active attacks on a connection
	Connection integrity without recovery	Detect active attacks on a connection
	Selective-field connection integrity	Detect active attacks on specific fields
	Connectionless integrity	Detect modification and some forms of replay of a single data block
	Selective-field connectionless integrity	Detect active attacks on specific fields of a data block
Non-repudiation	Origin	Prove that a data block was sent by the source
	Destination	Prove that a data block was received by the destination

divided into five categories and fulfill specific security objectives. In the following, we shall demonstrate that some of these services can be enhanced using physical-layer techniques, which at best provide information-theoretic security and at the very least cause significant degradation of the signals observed by the eavesdropper. But first, we set the stage for the integration of physical-layer security into contemporary networks by elaborating on the security mechanisms that have been introduced at each layer to ensure integrity, authentication, and confidentiality.

Application layer

The security mechanisms implemented at the uppermost layer of the protocol stack depend heavily on the application under consideration. A security architecture similar to the one depicted in Figure 7.1 is often implemented to secure email or web-browsing services by means of end-to-end cryptography. Public-key cryptography is used extensively for the purpose of authentication and the sharing of session keys. To ensure the authenticity of public keys, a public-key infrastructure must include trusted third parties, also called certification authorities (CAs). The service they provide consists of managing large directories of registered users and devices, issuing digitally signed certificates

of their public keys upon request. These certificates are then used to establish trusted connections among clients and servers.

With the advent of peer-to-peer systems, in which every node is both a client and a server to other nodes in the network, typical network functions such as packet forwarding and flow control are increasingly assigned to so-called overlay networks, which consist of virtual logical links among processes that run in different computing nodes. This form of virtualization simplifies network management and information dissemination, while allowing the construction of topologies that increase the overall robustness of the networked application. The primary concern in this context relates to active attacks by which information flows are compromised through the injection of erroneous packets. The solution here is to enforce integrity checks by means of cryptographic hash functions.

Transport layer

Transport protocols providing connection-oriented or connectionless services are typically implemented in the operating system. Security extensions, most notably the Secure Sockets Layer (SSL) and the Transport Layer Security (TLS) standard of the Internet Engineering Task Force (IETF), provide message encryption and server authentication, with optional support for client authentication. The system stores a list of trusted CAs, from which certificates can be obtained in order to exchange session keys and establish secured connections. The schemes employed are again very similar to the one shown in Figure 7.1, and the transport headers of the transmitted segments are modified to reflect the security primitives employed.

Network layer

Since the main task of network routers is to store packets and forward them towards the right next hop until the destination is reached, their operation is limited to the network layer and below. Consequently, all security mechanisms are implemented at the level of IP datagrams. The IPSec standard allows for the establishment of a unidirectional network-level security association and the introduction of a so-called authentication header, which includes a connection identifier and the digital signature of the sending node. The encapsulating security payload (ESP) protocol is then responsible for ensuring the confidentiality of the data by means of encryption. Beyond the datagrams that carry application data, it is very important to protect also the control traffic, which ensures that the routing tables are correct and up to date. Successful attacks on link advertisements and other control messages can make it impossible for routers to forward the packets correctly, which eventually leads to a serious disruption of network services.

To protect the network against intrusion attacks, it is common to install firewalls at the gateway nodes. Their goal is to filter out packets that are identified as suspicious on the basis of predefined rules that are under the discretion of network administrators. Beyond enforcing some form of access control, firewalls prevent attackers from learning about the available network resources and how they are mapped to different addresses and ports. In addition, they constitute a first line of defense against denial-of-service

(DoS) attacks in which attackers flood the network with fake connection requests or other forms of signaling.

Link layer

While the network layer is agnostic with respect to the underlying channel, the link layer, which governs the access to the channel by multiple devices, depends naturally on the communication medium of choice. It is thus not surprising that security mechanisms are more often found at the networking layer and above, where the same security subsystem can operate irrespective of the physical characteristics of the link over which communication is taking place.

However, whereas security primitives implemented at the network layer address only the vulnerabilities pertaining to end-to-end connectivity, link-layer security is concerned with the direct connections among devices. In contrast with wired networks, where access to the communication link requires tapping a cable or optical link, wireless networks are easy targets for eavesdroppers and intruders. All that is required is a computer with a wireless interface, both of which have now become very affordable commodities. Thus, security solutions at the link layer of wireless communications systems are aimed at ensuring that only authorized devices can communicate over the channel. One such instance is the Extensible Authentication Protocol (EAP), which defines a security framework over which a device can prove its identity and gain access to a wireless network.

7.3 Practical case studies

The previous overview of the security mechanisms that are typically implemented at the various layers above the physical layer underlines the fact that the individual solutions for each layer are tailored to its specific tasks with little relation to the security primitives implemented elsewhere in the system. A device may, for example, be authenticated over a wireless link and operate IPSec at the network layer, while at the application layer a browser is accessing a web site securely using the end-to-end cryptography implemented by TLS. In some cases, this patchwork of security mechanisms can be redundant, requiring for instance parallel authentication steps at different layers or multiple encryptions. At the same time, some vulnerabilities are left open at other layers. This is well illustrated by the fact that end-to-end cryptography at higher layers is unable to prevent traffic-analysis attacks carried out at the lower layers, such that an opponent can learn about the topology of the network and the type of sessions it is carrying.

Owing to the currently existing trust in the security levels assured by cryptographic mechanisms implemented at higher layers, it is fair to say that most available engineering solutions do not exploit the full potential of the physical layer in increasing the overall security of the system. We start by providing a few motivating examples and then proceed with a system-oriented view of how physical-layer security inspired by

information-theoretic security can be incorporated into the wireless communication networks of today.

Optical communication networks

Communication networks based on optical fibers were among the first communication infrastructures to embrace physical-layer security. Since the information bearing signals are carried along the cable in the form of light with total internal reflection, typical attacks are physical in nature, thereby diverting the light in order to disrupt the communication service, degrade the quality of service, or extract information about the traffic and its content. Although there is very little radiation from a fiber-optic cable that would allow non-intrusive eavesdropping, an attacker with access to a cable can cut the fiber or bend it in a way that allows light to be captured from or released into the fiber. Sophisticated attacks are capable of intercepting selected wavelengths while keeping others intact. Optical jamming can then take different forms, depending on whether the attacker chooses to inject noise, delayed versions of the captured signal (repeat-back jamming), or some other disrupting signal (correlated jamming). Even if the destination is capable of monitoring subtle fluctuations in received power, many of these attacks can go undetected. Physical-layer security solutions for optical communications include robust signaling schemes, coding, and active limitation of the power and bandwidth of input signals. Since the communication rates are extremely high, coding solutions that require complex processing are hard to implement and therefore very expensive.

Spread spectrum

It is fair to say that, much like other classes of communication networks, wireless systems have not been designed with security requirements as their chief concern and main figure of merit. There is, however, one notable exception: spread-spectrum (SS) systems. Since the modulation schemes in this class were invented first and foremost for military applications, their designers aimed at counteracting the possibility of an enemy attacker detecting and jamming the signals sent by the legitimate transmitter. The key idea is to use pseudo-random sequences to spread the original narrow-band signal over a wide band of frequencies, thus lowering the probability of interception and reducing the overall vulnerability to narrow-band jamming. At the receiver the wide-band signal is de-spread back to its original bandwidth, whereas the jamming signal is spread over a wide band of frequencies. This in turn reduces the power of the interference caused by the jammer in the frequency bandwidth of the original signal.

There are two main SS techniques: direct sequence spreading (DSS) and frequency hopping (FH). DSS modulates a signal s_i onto a train of rectangular pulses $p(t)$ of duration T_c, also called chips, whose values are governed by a pseudo-random spreading sequence known both to the transmitter and to the receiver. The outcome of this process,

$$s(t) = \sum_i s_i \, p(t - i\,T_c),$$

is then transmitted to the receiver. To recover the original signal, the receiver must multiply the acquired signal once again by the spreading sequence. The main challenge is

to synchronize the pseudo-random sequence used by the receiver with the one used by the transmitter. It follows that spreading sequences must have a strong autocorrelation. Furthermore, sophisticated search and tracking algorithms are needed in order to synchronize and to keep the synchronization between transmitter and receiver. This in turns limits the attainable bandwidth of DSS signals.

As the name indicates, FH spreads the signal by shifting it to different frequencies that are dictated by a pseudo-random sequence. This operation can be expressed in the complex plane by

$$s(t) = \sum_i \exp(\mathrm{j}(2\pi f_i + \phi_i))p(t - iT_\mathrm{h}),$$

where f_i denotes the frequency shift, ϕ_i is a random phase, and T_h is the so-called hop time. Similarly to DSS, the receiver must acquire and track the pseudo-random FH sequence. The modulation is called fast FH if the number of hops per transmitted symbol is equal to or above one. Otherwise, we speak of slow FH modulation, whereby multiple symbols are transmitted in each hop.

Today, spread-spectrum techniques are used extensively in wireless networks both for military and for civilian use. Low-probability-of-interception (LPI) systems hide the transmissions by exploiting the fact that signals modulated according to pseudo-random spreading sequences are hard to distinguish from white noise. In cellular mobile communication networks, spread-spectrum techniques are used primarily for multiple-user channel accessing, whereby all users transmit simultaneously, albeit using different spreading sequences. Since the spreading sequences are orthogonal to each other, the receiver can recover the signal transmitted by each individual user by multiplying the received signal by the corresponding spreading sequence. Privacy can be enhanced by keeping the spreading sequences secret from potential attackers.

Mobile communication systems

The Global System for Mobile Communications (GSM) standard is widely deployed around the world, offering mobile telephony and short-messaging services over wireless channels. Its security architecture is somewhat unconventional in that authentication is dealt with at the application level while confidentiality is implemented at the physical layer. Data are thus encrypted after having been processed by the channel coding and interleaving blocks, as shown in Figure 7.3. The authentication follows a standard challenge–response protocol that relies on the shared secret embedded in the Subscriber Identity Module (SIM). The encryption stage is based on a shared session key and a specific stream cipher, the so-called A5 algorithm. In addition, GSM uses frequency hopping as a means to combat multipath fading and to randomize co-channel interference. Since frequency hopping is a form of spread-spectrum communication, GSM is inherently robust against narrow-band jamming.

The General Packet Radio Service (GPRS) extends the functionality of GSM to enable data communication at rates up to several tens of kbits/s. In contrast to GSM, the system designers of GPRS opted to place the encryption algorithm at the Logical Link Control

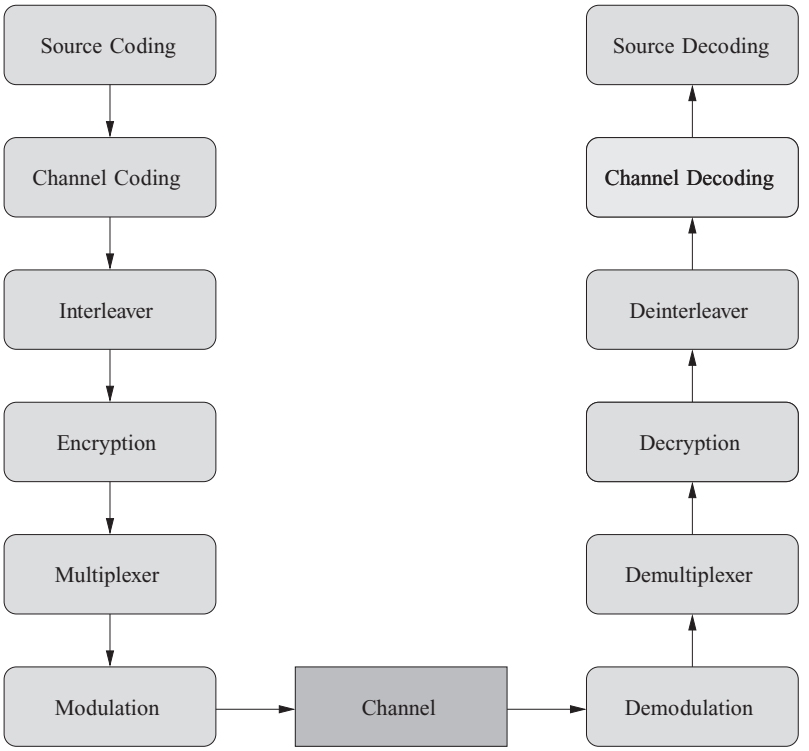

Figure 7.3 GSM architecture.

(LLC) layer instead of at the physical layer. Moreover, the algorithm uses a different stream cipher. More recent standards for mobile broadband communications such as the Universal Mobile Telecommunications System (UMTS) and Long Term Evolution (LTE) also do not employ physical-layer security, except for the aforementioned spread-spectrum techniques. The same is true for the Dedicated Short-Range Communications (DSRCs) for vehicular networks.

RFID and near-field communications

Radio-frequency identification (RFID) tags are emerging as a convenient means to track objects, such as toll cards on highways, passenger luggage in airports, consumer goods in retail, and parcels in shipping businesses. Active, semi-passive, and passive RFID tags all share the same principle of operation: (1) the antenna of the tag receives electromagnetic energy transmitted by the antenna of the RFID reader and (2) the tag sends the EPC back to the reader via the radio channel. The range of transmission can take values up to 100 m. Active tags use a battery to power both their circuitry and their transmissions, whereas passive tags must use some of the power collected by their antenna during reception. Semi-passive tags use battery power for the circuitry only. In a typical system, the tag provides a 96-bit number (also known as an electronic product code, EPC) that

points to an entry in a database with restricted access. An attacker might be able to read the EPC from the tag, but the rest of the information is stored and protected elsewhere.

Near-field communication (NFC) builds on the idea of RFID in order to enable high-rate data transfers at very short ranges. The applications envisioned include making payments for various services or sharing multimedia files among mobile phones. Several prototypes exploit inductive coupling at the physical layer. The NFC forum (an association of more than 130 companies interested in commercializing near-field communication technology) makes the following claim on its web site: *Because the transmission range is so short, NFC-enabled transactions are inherently secure. Also, physical proximity of the device to the reader gives users the reassurance of being in control of the process.*

Physical-layer security is so far absent from NFC and RFID technologies. A common assumption is that an eavesdropper will be farther away from the transmitting device than the legitimate reader. The eavesdropper is thus expected to have a worse signal-to-noise ratio (SNR) than that of the reader, which prevents the leakage of vital information through the electromagnetic waves traversing the channel. Clearly, this assumption is well captured by the wiretap-channel model, which leads us to believe that more secure NFC and RFID systems can be developed using secure channel codes and other physical-layer security techniques.

7.4 Integrating physical-layer security into wireless systems

From the previous sections it is clear that physical-layer security cannot be viewed as a panacea for solving all of the security concerns that exist in today's networks. There is evidence, however, that careful design of the physical layer, promoting these security concerns as fundamental performance criteria, will lead to wireless networks that are arguably more secure than those whose security is rooted on cryptographic primitives alone. We shall now investigate how this can be achieved by applying the information-theoretic security principles described in detail in previous chapters.

Impact on the security architecture

To increase the security of a wireless network in a bottom-up fashion, i.e. from the lowest layer to higher layers, the design of the physical layer must go beyond its traditional role of ensuring virtually error-free transmission across imperfect channels by means of powerful error-correction coding and adequate modulation schemes. Figure 7.4 illustrates two new security functions that can be introduced at the physical layer, either as stand-alone features or in combination with the cryptographic mechanisms at higher layers. One function consists of reducing the number of error-free bits that an eavesdropper can extract from the transmitted signal. This can be achieved in several scenarios of practical interest by using a special class of codes, which we call *secure channel codes*. The other function is concerned with exploiting the randomness of the communications channel to generate secret keys at the physical layer. The keys can then be passed on to

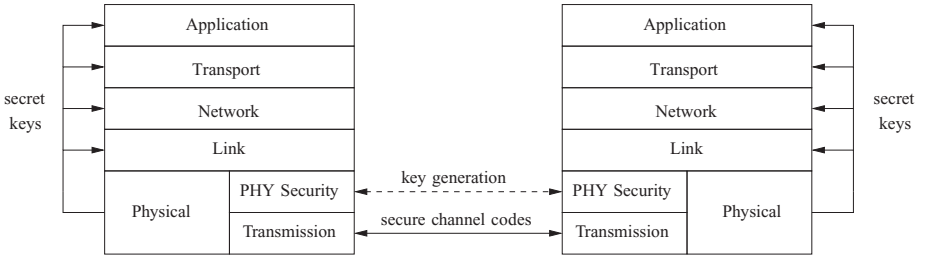

Figure 7.4 Integration of physical-layer security into the network architecture.

higher layers for use with existing cryptographic schemes. The next sections elaborate further on the merits and drawbacks of these two methodologies.

Secure channel codes

The key idea of the first approach presented in the previous section is to substitute typical channel codes and modulation schemes, which ensure reliability but not security, by the class of code constructions described in Chapter 6. These can achieve secrecy and reliable communication at a tolerable price in terms of achievable transmission rate and computational complexity. The codes degrade significantly the amount of information that an eavesdropper is capable of extracting from the observed signals, provided that his SNR is inferior to that of the legitimate receiver over the duration of a transmitted codeword.

While it is fairly straightforward to discover the state of the channel between legitimate communication partners by means of pilot symbols, as explained below, the same is obviously not true for the eavesdropper, who is likely to hide his presence or at the very least conceal his location. We are thus left with the options of (a) transmitting only when the SNR is high (and therefore more likely to exceed the SNR of the eavesdropper), (b) ensuring by physical means that the eavesdropper cannot place an antenna within a certain range, or (c) degrading the eavesdropper's reception by means of jamming the regions where his antenna is assumed to be located, as explained in Chapter 8.

Secure channel codes can be used at the physical layer in a modular fashion, i.e. independently of the security architecture implemented at higher layers. If the sent datagrams are protected by cryptographic primitives, be it through symmetric encryption or public-key cryptography, the use of secure channel codes can strengthen the overall security level significantly by virtue of the fact that the eavesdropper is no longer able to obtain an error-free copy of the cryptogram. Instead, even less sophisticated secure channel codes, which increase the error probability at the eavesdropper rather than the total equivocation (or uncertainty) about the sent information bits, can force the attacker to deal with a large number of erroneous symbols that are virtually indistinguishable from the scrambled bits resulting from cryptographic operations. Therefore, classical tools, such as differential and linear cryptanalysis, no longer apply and must be modified to account for bit errors in the encrypted stream. In some cases, the best that can be achieved is to form a list of possible message sequences. It is to be expected that, in

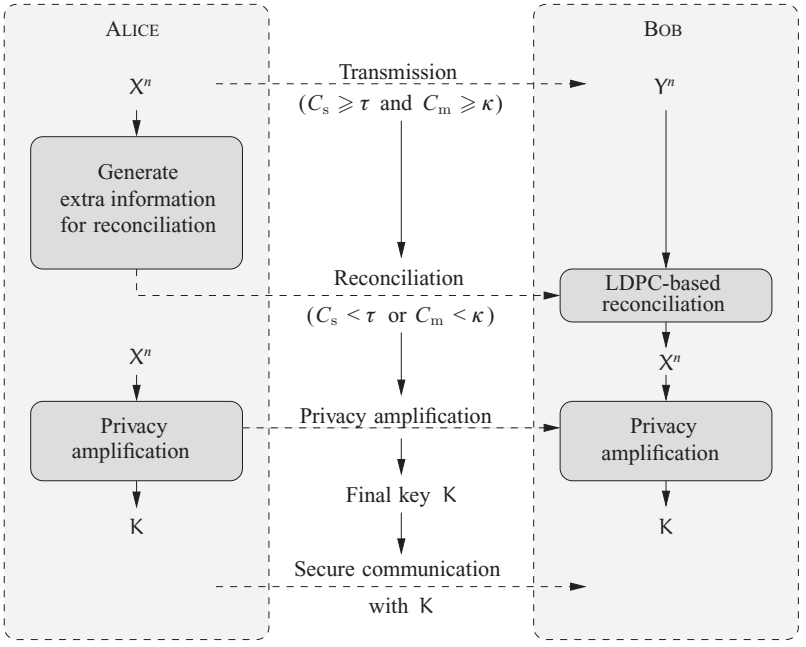

Figure 7.5 Protocol for secret-key agreement.

most cases of interest, the task of the enemy cryptanalyst will become significantly more complex.

When secure channel codes are used in an opportunistic manner depending on current estimates of the SNR, transmissions occur only if the received power is above a certain threshold. Communication rules such as these will obviously affect the transmission schedule and the access to the channel. This suggests joint design of the physical and link layers, as well as an optional adaptation of routing and retransmission at the network and transport layers, respectively.

Insofar as the actual implementation of secure channel codes in real systems is concerned, the main challenges will of course vary from platform to platform. The transmission schemes at the physical layer are typically implemented in a device driver, e.g. for the wireless interface card. With the proliferation of software-defined radios, it is reasonable to assume that security-oriented mechanisms at the physical layer will be easier to integrate into future communication systems.

Secret-key agreement at the physical layer

The second security function that can be implemented at the physical layer is secret-key agreement. As explained in Chapter 4, after sharing some common randomness in the form of correlated symbols, Alice and Bob can agree on a secret key by means of reconciliation and privacy amplification. Figure 7.5 illustrates a wireless security protocol that achieves this purpose. When the estimated secrecy capacity C_s and main channel capacity C_m from Alice to Bob are found to be above the prescribed thresholds,

τ and κ, respectively, Alice transmits Gaussian-shaped symbols X^n to Bob, who observes the channel outputs Y^n. The two sequences of Gaussian symbols are correlated and form the core of common randomness from which Alice and Bob can generate their key. Using a quantizer with a bit mapping, Alice obtains a sequence of bits. To ensure that the two legitimate partners share exactly the same sequence from the observed symbols, Alice sends some side information in the form of a number of low-density parity-check (LDPC) bits (using, for instance, the reconciliation protocol described in Section 6.5.2), which are protected by a powerful channel code. Bob is thus able to receive these bits virtually without error, and feeds them to a belief-propagation decoder for reconciliation. The side information can be sent at any time, even when Eve has a better channel than Alice. To ensure that Eve is unable to extract any information about the key from the noisy symbols and the side information she has access to, Alice and Bob use privacy amplification, thus arriving at a smaller sequence of secret bits, which can be used to communicate securely.

It has been shown that, with a fraction of time dedicated to secret-key generation as small as 1%, such a secret-key agreement protocol can renew a 256-bit encryption key every 25 kbits, i.e. with SNRs of 10 dB and 20 dB for the main and eavesdropper channels, respectively, and, transmitting at an average rate of 2 Mbps, a secret key could be replaced by a new random key every 16 milliseconds. Although these estimates may be optimistic, continuous generation of secret keys at the physical layer thus emerges as a promising solution for the key-reuse problem described.

As in the case of secure channel codes, we are confronted with the need to modify the physical layer, which implies redesigning a wireless interface card and writing a new device driver. Again, software-defined radio may prove valuable in adapting the physical layer for security functionalities. Since common randomness is shared in an opportunistic way, the medium-access control at the link layer, which governs the scheduling of transmissions on the basis of channel state information, will have to be adapted accordingly. This calls for cross-layer design of security protocols, which must ensure also that higher layers are capable of acquiring and using the secret keys generated at the lowest layer. In current systems, this involves altering the communication and security components of the operating system.

The role of channel state information

The objective of physical-layer security is for Alice and Bob to communicate reliability at a certain target rate while leaking the least possible number of information bits to Eve. If the SNRs both of the main channel and of the eavesdropper's channel are known perfectly to Alice, she can compute the secrecy capacity and adjust her transmissions accordingly. When the secrecy capacity turns out to be zero, Alice can decide not to transmit, thus avoiding the likely disclosure of information. On the other hand, when the secrecy capacity is strictly positive, Alice can choose a secure channel code that operates at that rate and achieve information-theoretic security. Alternatively, she can share common randomness with Bob, when the secrecy capacity is positive, and then generate a secret key. Either way, knowledge of the state of both channels is necessary for assuring information-theoretic security.

Estimating the channel between the legitimate users is common practice in the most widespread wireless communication systems. Typically, each burst contains a known training sequence that is transmitted together with the coded bits. In GSM, for example, training bits account for about 20% of the traffic. After detecting the transmitted signals, the receiver can use the training bits and the received signal samples to estimate the channel impulse response. The most common algorithms are the least-squares and linear minimum mean-squared error estimators. Often a feedback channel is available, over which the receiver can inform the transmitter about the state of the channel by sending either the received signal or quantized values of the main channel parameters. In some cases, such as time-division duplex (TDD) systems, in which the coherence time of the channel is sufficiently long, the channels from transmitter to receiver and vice versa are identical. This form of reciprocity obviously simplifies the sharing of channel state information.

In some scenarios, the legitimate communication partners may have partial knowledge of the eavesdropper's channel. This corresponds, for instance, to the situation in which Eve is another active user in the wireless network (e.g. in a TDD environment), so that Alice can estimate the eavesdropper's channel during Eve's transmissions. Naturally, an attacker that aims to intercept or disturb the communication between the legitimate partners cannot be expected to participate in the channel-estimation process or to follow any conventions set by the communication protocol. One possible attack consists of sending low-power signals, which lead Alice and Bob to underestimate the SNR at Eve's receiver. If Eve behaves as a passive attacker, no training signals are available for estimating the state of the eavesdropper's channel. In this case, the best that Alice and Bob can do is to use a conservative estimate based, for example, on the verifiable assumption that Eve is located outside a certain physical perimeter. Studies show that protocols such as the one illustrated in Figure 7.5 are considerably robust against channel-estimation errors.

Authentication requirements

We showed how secure channel codes can be used to increase the levels of confidentiality provided by a wireless network and how secret-key agreement at the physical layer helps solve the key-reuse problem by exploring the randomness of the communications channel. In all cases, it is assumed that the attacker is passive and does not try to impersonate either Alice or Bob. In most scenarios of interest, which do not grant this type of assurance, all communications must be authenticated, otherwise an active attacker is able not only to intercept the datagrams sent over the wireless channel but also to transmit fake signals aimed at confusing Alice and Bob. In the simplest instance, an attacker could disrupt the communications by jamming their transmissions, which can be viewed as a form of DoS. Alternatively, an impersonation attack would allow the attacker to play the role of the man-in-the-middle, for example generating secret keys with Bob while pretending to be Alice, and vice versa, and modifying the exchanged messages without being noticed.

Notice that a small shared secret is sufficient to authenticate the first transmission. Subsequent transmissions can then be authenticated using the secret keys that can be generated at the physical layer through common randomness, reconciliation, and privacy

amplification. Despite our ability to authenticate subsequent transmissions, physical-layer security as it stands does not solve the key-distribution problem and we are still left with the question of how to share the initial secret key, which is critical for bootstrapping the security sub-systems of the wireless communication network.

One way is to rely on the traditional approach using public-key cryptography. More specifically, nodes in the network would rely on a public-key infrastructure (PKI) to obtain authenticated public keys from a trusted certification authority and then use these public keys to share the initial secret key, which allows them to authenticate the first transmission and thus establish a secure link. This approach obviously inherits all the flaws of public-key cryptography, most notably the reliance on computational security and the complexity of establishing a PKI.

If Alice (or Bob) can be trusted to be the first one to establish a connection then it is possible to authenticate transmissions at the physical layer using the impulse response of the channel. The key idea is that Alice sends a known probing signal, which Bob can use to estimate the channel's impulse response and infer the channel state, more specifically the instantaneous fading coefficient. Since it has been shown experimentally that a probing signal sent by a different sender transmitting from a different position is going to generate a different impulse response with overwhelming probability, Bob will be able to detect any attempt by Eve to impersonate Alice after the first transmission. All he needs to do is to compare the observed impulse responses for different transmissions.

7.5 Bibliographical notes

The subject of classical cryptography and computational security is well addressed in a number of textbooks. Stallings gives a general overview of security systems as they exist today [139]. The design of ciphers based on the notions of confusion and diffusion was studied by Shannon in [1]. The principles of public-key cryptography were first published by Diffie and Hellman in [140]. Rivest, Shamir, and Adleman proposed the widely used RSA scheme in [141]. A comprehensive and practically oriented treatment of symmetric encryption and public-key cryptography is given by Schneier in [142]. Another essential reference on these matters is the handbook by Menezes, Van Oorschot, and Vanstone [143]. A survey of physical-layer-security aspects of all-optical communications by Médard, Marquis, Barry, and Finn can be found in [144]. Wegman–Carter authentication schemes are discussed in [66]. An information-theoretic treatment of authentication was presented by Maurer in [145]. Several security schemes presented by Li, Xu, Miller, and Trappe in [146] for wireless networks are based on the impulse response of the channel. Bloch, Barros, Rodrigues, and McLaughlin [105] developed the secret-key agreement protocol described in Section 7.4.