Mobile Security and Privacy

8

The phenomenal growth of the Internet has given rise to a variety of network applications and services that are pervading our daily life at a staggering pace. This trend is being boosted by myriad mobile devices that essentially make it possible to access network resources anywhere, anytime. In parallel, security and privacy issues have surfaced in almost every aspect of the mobile computing paradigm, from wireless communication security to network denial of service (DoS) attacks, to secure network protocols, and to mobile privacy. Furthermore, the inherent characteristics of mobile computing have imposed greater challenges on mobile security and privacy solutions than on general wired network security approaches.

This chapter explores a wide range of mobile security and privacy issues, presents a big picture of this broad area, and offers some insight into the fundamental security problems surrounding the design of secured mobile wireless systems and applications. The chapter begins with a security primer summarizing a set of basic network security concepts and security schemes, followed by an in-depth coverage of security issues in cellular networks, wireless LAN, Bluetooth, and other emerging mobile wireless systems. When presenting each topic, we introduce technical aspects of each problem and discuss some proposed approaches for solving them. When possible, we then outline some real-world solutions to the underlying problems. Readers will be able to quickly obtain a solid understanding of key mobile security and the related privacy issues.

The security issues surrounding mobile wireless networks and applications can be categorized as follows:

Message confidentiality
Message integrity
Message authentication
Nonrepudiation
Access control

When discussing differences between security and privacy, we consider this list to be comprised of security problems, whereas identity and location anonymity are topics relevant to mobile privacy.

8.1 SECURITY PRIMER

Let us first consider a typical scenario in a mobile computing paradigm, where it is possible to use a mobile device (e.g., cell phone, PDA, smart phone, laptop computer) to access a network service using a variety of wireless communication technologies, such as a wireless local area network (LAN) or cdma2000. This operation involves utilizing some type of hardware (i.e., the mobile device being used), one or more wireless network devices, a back-end wired or wireless network infrastructure, and software, such as the application and supporting mobile operating system of the mobile device, operational and management software on wireless devices, and application software on destination servers. The scenario becomes much more complicated when group communication is being performed. Nevertheless, the fundamental question is how we can secure the entire communication environment. This problem can be approached from several different perspectives:

- *End user's perspective*—An end user may use the mobile device for many purposes, including online shopping, online banking, and personal communication with friends and colleagues, or the end user may utilize such services as online maps, weather forecasts, or online gaming. Because in many cases sensitive information is sent back and forth, the end user's major concerns are likely to include data confidentiality and integrity, as well as authenticity of the other party with which the user is connected.

- *Service provider's perspective*—A service provider has to provide a secure network infrastructure for various mobile applications and services that directly interface to end users. This implies secured communication over wireless networks and wired networks. The service provider and the end user have to authenticate each other, and the computing platform should guarantee that no information will be divulged during the communication between them. The service provider also has to protect the network infrastructure against attacks.

- *Employer's perspective*—Enterprise networks must be able to ensure the security of corporate assets. This is particularly crucial when the enterprise network provides both wired and wireless access. A well-defined, highly secured wired enterprise network may be completely open to attackers if a wireless access extension to the enterprise network is not secured. For example, a rogue access point in an enterprise network may essentially provide a means to bypass corporate firewalls and directly access network resources.

Many technical notions, terms, and technologies have been introduced to address security problems in common network environments. Table 8.1 provides a brief summary of this terminology.

Depending on the nature of security problems encountered in the mobile wireless world, they can be addressed in one or more layers of the network protocol stack. Radio modulation techniques such as FHSS (Frequency Hopping Spread

Table 8.1 Security Terminology

Term	**Description**
Encryption	The transformation of some information (*cleartext* or *plaintext*) into a form (*ciphertext*) that is only readable by intended recipients who hold some decryption keys
Confidentiality	A security function that ensures that no one except the intended recipient who holds some key is able to obtain the message being transferred between the sender and the recipient
Integrity	A security function that allows the intended recipient to detect any modification to a message from a sender performed by a third party
Authentication	A security function that enables verification of the identity of a person, a data object, or a system
Nonrepudiation	A security function that ensures that a message sender cannot deny a message it sends previously
Cryptography	Mathematical foundations of security mechanisms facilitating the four security functions: confidentiality, integrity, authentication, and nonrepudiation
Secret key cryptography	A type of cryptographic mechanism that enables the sender and the intended recipient to use the same shared key for security functions
Public key/private key cryptography	Another type of cryptographic mechanism in which two keys are used by an entity—a public key that is made available to anyone and a private key derived from the public key and known only to the owner and sometimes some trusted parties
Symmetric key encryption	An encryption mechanism that allows the sender and recipient to use the same secret shared key to encrypt and decrypt a message; also called *secret key encryption*
Asymmetric key encryption	An encryption mechanism in which the message sender uses the intended recipient's public key to encrypt a message and the recipient uses his or her private key to decrypt it
Cipher	The mathematical algorithm that is used to encrypt cleartext
Message digest	Fixed-size output of a one-way hash function applied to a message of arbitrary size
Message authentication code (MAC)	A code of a message that is computed based on the message and a secret key such that the intended recipient who holds the secret key can verify the integrity of the message
Hash MAC (HMAC)	A MAC that is computed using a one-way cryptographic hash function such as MD5 and SHA-1 and a key

(*Continued*)

Table 8.1 (*Continued*)

Term	Description
Digital signature	A code that is computed based on the message or a hash code of the message and the private key of the sender such that anyone can verify the integrity of the message using the sender's public key; the sender "signs" the message (digital signature is the public key equivalent of MAC)
Digital certificate	A form of electronic certificate document issued by a generally trusted certificate authority (CA) to certify someone's public key; a digital certificate, signed by the CA, contains the owner's identity, the owner's certified public key, the name of the issuer (the CA that issued the digital certificate), certificate expiration date, and some other information; a CA's public key is often distributed with software packages such as web browsers and e-mail software
Public key infrastructure (PKI)	A public-key-based architecture that uses digital certificate signed by a CA to create, manage, distribute, and verify public keys and their associated identity information
Pretty good privacy (PGP)	A technique developed by Phil Zimmermann that uses asymmetric key encryption for e-mail encryption and authentication between two entities
Authorization	The process of granting and denying specific services to an entity based on its identity and established policy

Spectrum) can be used to provide wireless signal transmission security at the physical layer. Link encryption is often used in wireless networks where an access point or master serves as the gateway for everyone. Internet protocol security (IPSec) is an example of a network layer security mechanism. End-to-end security can be addressed at the transport layer. Applications usually have to deal with user authentication and access control. This chapter focuses on security solutions at the data link layer and above which invariably leverage cryptographic principles as building blocks.

A cryptographic system is the realization of a cryptographic scheme or mechanism that can be integrated into a general computer or network system to provide specific security services. The two types of a cryptographic system are *symmetric key systems* and *asymmetric public key systems*. Symmetric key systems such as the Data Encryption Standard (DES) and Advanced Encryption Standard (AES) use the same *secret key* for encryption and decryption, thus requiring a secured way to distribute the key; for example, the Diffie–Hellman key exchange protocol (explained later in Section 8.1.4) specifies a method for symmetric key distribution. In contrast, public key systems use two different keys for encryption

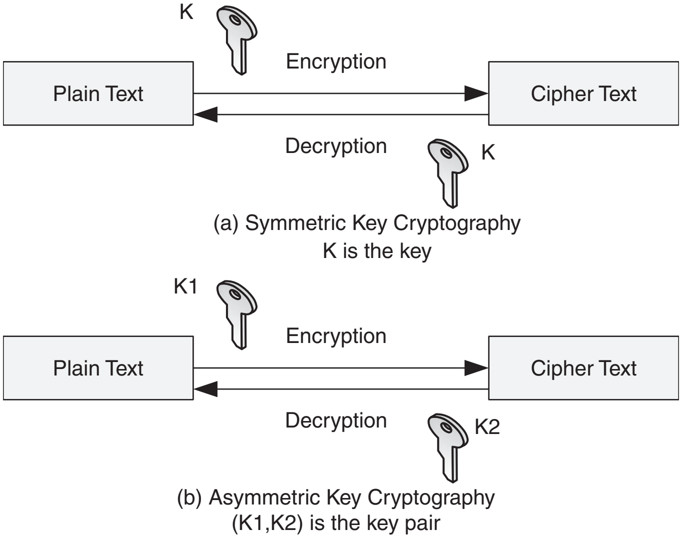


FIGURE 8.1

Symmetric cryptography and asymmetric cryptography.

and decryption: a *public key*, which is known to the public, and a corresponding *private key*, which is known only to the owner of the key pair. The public/private key pair generation algorithm ensures that it is mathematically impossible to deduce the private key based on a public key. An important characteristic of public key cryptographic systems is that the two keys are mathematically related in such a way that data encrypted by a public key can only be decrypted using the corresponding private key, and vice versa. Figure 8.1 depicts both symmetric key cryptography and asymmetric public key cryptography. Public key systems essentially provide a foundation for various security solutions to the problems listed earlier. The basic idea of these approaches is that a message from a sender can be encrypted using its private key and the recipient can verify that the message is, in fact, from the sender (sender authentication). Conversely, by using the recipient's public key to encrypt data, the sender can be assured that only the intended recipient is able to decrypt the scrambled data (recipient authentication). As discussed below, very often a public/private key pair is used in combination with other techniques to provide secure communication during a session. In order to ensure public key authenticity while it is being distributed in a network, the public key infrastructure (PKI) can be used (explained later in Section 8.1.3).

Public key cryptography was first proposed in 1976 by Whitfield Diffie and Martin Hellman as an encryption scheme. Public key cryptographic systems have been widely used to provide confidentiality and authentication between senders and recipients and to secure transmission of some negotiated secret such as a session key between them. In the latter case, the cryptographic system is a hybrid system combining both asymmetric cryptography and symmetric cryptography. Popular public key cryptographic systems include RSA and elliptic curve cryptography (ECC).

8.1.1 Ciphers and Message Confidentiality

The first issue in message security is to encrypt the message such that no one except the intended recipient is able to recover the message content. In the context of symmetric key cryptography, this is often done by a block cipher using some secret key. A block cipher takes a fixed length of information (for example, a 128-bit block of cleartext) and uses a secret key to produce ciphertext, usually of the same length as the cleartext block. A block cipher also supplies a decryption function that takes the cipher text and the secret key and then produces the original cleartext. For messages that are larger than block size, a cipher may employ a particular mode to deal with the message. A mode defines the way a cipher is applied to cleartext. An important concept in data encryption is the well-known Kerckhoffs' principle, which states that an encryption scheme should be secure even if the algorithm used is known to the public. This means that an attacker is well aware of the algorithm and the ciphertext of a message but not the secret key.

Asymmetric encryption algorithms use public/private key pairs for encryption and decryption, thus they do not require the two parties involved to share the same secret key. A good cipher should make it computationally difficult for an attacker to decrypt a message without knowing the key (i.e., the shared secret key or the private key being used for encryption). Popular symmetric block ciphers are DES/Triple-DES and AES, whereas well-known asymmetric ciphers include RSA and ECC. Generally, asymmetric ciphers are much slower than symmetric ones in terms of encryption speed. In addition to the common ciphers introduced below, a number of technology-specific ciphers such as the A5 algorithm are used in global system for mobile (GSM)/general packet radio service (GPRS) systems. Following is a brief introduction to these ciphers:

- Data Encryption Standard (DES) and Triple-DES—DES uses a 56-bit secret key to encrypt message blocks of 64 bits. There are 16 identical stages of processing, called *rounds*, and an initial and final permutation. The Feistel function determines how data are processed throughout those rounds using carefully generated subkeys for each round. DES has been a federal standard of data encryption for years but was finally superseded by AES in 2002, due to its weakness of using short 56-bit keys. In fact, as a result of the fast advancement in computing power, DES has been broken by brute force attacks in one to two days with the help of some powerful computers. Triple-DES is a relatively improved DES in that it uses three DES operations sequentially to compute the ciphertext. It performs a DES encryption, then a DES decryption, and then a DES encryption again. Triple-DES is generally considered a better cipher than DES. Its main drawback is computation overhead incurred by the three DES procedures.

- Advanced Encryption Standard (AES)—AES has a fixed block size of 128 bits and a key size of 128, 192, or 256 bits. A data block is organized into a 4 × 4 array, or *state*. AES may require 10 to 14 rounds of computation, depending on the key size. Because many operations in a single round can be performed in parallel, AES is comparatively easier to implement in both hardware and software and can be done much faster than DES. The real name of the cipher is Rijndael, a combination of the two designer's names: Joan Daemen and Vincent Rijmen. Rijndael was chosen by the National Institute of Standards and Technology (NIST) to be the government standard. As of this writing, no attack has broken AES.
- Blowfish and Twofish—Blowfish is yet another block cipher developed by Bruce Schneier in 1993. It uses a key up to 448 bits over blocks of 64 bits. Blowfish has 16 rounds following the Feistel function. Blowfish is generally regarded as a compact and fast replacement of DES. Twofish specifies block size of 128 bits and uses a key size up to 256 bits. Twofish also made it to the final list of the AES contest but lost to Rijndael. There is no reported successful attack over Blowfish and Twofish.

Other well-known block ciphers are CAST-128, CAST-256, RC5, and RC6, among others. It is important to remember that, with regard to data encryption on mobile devices, computational overhead becomes a much more severe problem than on desktop computers; hence, while choosing a cipher to encrypt packets in a wireless network, those ciphers with low overhead such as RC5 will be advantageous.

In addition to block ciphers, another type of cipher is the stream cipher. Unlike block ciphers, a stream cipher encrypts one bit or one byte at a time. The two types of stream ciphers are synchronous and self-synchronizing ciphers. Synchronous stream ciphers require a key to produce a keystream, which in turn is used to compute the ciphertext. The computation is done by XORing (exclusive OR operation) the keystream with the cleartext. Decryption follows in the same manner. Self-synchronizing stream ciphers do not require a key. Instead, they use some bits of the previous ciphertext to produce the keystream. Stream ciphers are primarily used to secure network data transmission where the cleartext is a stream of bits rather than a static data block.

RC4 is the most widely used stream cipher, although it has been shown that RC4 is not always secure. RC4 was designed by Ron Rivest of RSA Security in 1987. RC4 (Rivest Cipher 4) is one of the four ciphers that Rivest developed. In RC4, a variable-length key is first used to perform a permutation of one byte according to a key scheduling algorithm. The result, along with two index pointers, is fed into a pseudo-random generation algorithm (PRGA) to produce the keystream, which will be XORed with the cleartext to obtain the cipher. RC4 has been found to have serious vulnerability in the key scheduling algorithm that in some special cases may enable an attacker to recover the encryption key [1]. This weakness has been leveraged by some researchers to break wireless equivalent privacy (WEP) encryption, the security mechanism of IEEE 802.11b wireless LAN, which uses RC4 for data encryption. Details regarding this WEP vulnerability are provided in Section 8.3.

Most commercial security software supports a list of block or stream ciphers from which users can choose. A well-known opensource cipher implementation is the *libcrypto* library in the OpenSSL package (http://www.openssl.org/). Both Java and Microsoft .Net provide a package of these ciphers. In addition, they are also supported in the mobile platforms J2ME and .Net Compact Framework. Cryptographic schemes discussed in the rest of this section, such as hashing algorithms, digital signatures, and digital certificates, are generally supported by these libraries.

8.1.2 Cryptographic Hash Algorithms and Message Integrity

Aside from message confidentiality, another security problem is how to ensure message integrity—that is, how to protect data from being modified between the two parties. One-way hashing was introduced for this purpose. Simply put, a one-way hash algorithm, sometimes referred to as a *message digest algorithm*, makes sure that any modification to a message can be detected. A cryptographic hash algorithm or message digest algorithm in this regard must possess the following security properties:

- *Fixed-length output*—Given any size of message, it must produce a fixed size result, which is the hash code.
- *One-way*—Given a message m and a hash algorithm h, it is easy to compute $h(m)$; however, given a hash code x and hash algorithm h, it is computationally impossible to find m such that $h(m) = x$.
- *Collision resistance*—Because a hash algorithm is effectively a mapping between a large code space to a considerably smaller code space, collisions are bound to happen, meaning that brute force attacks are theoretically possible. The challenge is how to find collisions within a reasonable amount of time, given a state-of-the-art computing facility. The two types of collision resistance are strong collision resistance and weak collision resistance. Strong collision resistance means it is computationally impossible to find two different messages that can be hashed into the same code, whereas weak collision resistance means it is impossible to find a message that can be hashed into the same hash code of another given message.

Depending on how a hash algorithm operates, the two types of cryptographic hash algorithms are keyed and keyless. Keyed hash algorithms take a message and a key to compute the hash code, while keyless hash algorithms simply use the message to compute the hash code. Keyless hash algorithms are used to detect modifications to a message, assuming that the hash code of the original message is correctly transmitted to the recipient. Because of the collision resistance property, any change to the transmitted message can be detected immediately; however a problem arises when an attacker modifies the intercepted message, generates a hash code, and sends the tampered message and its hash code to the recipient. In this case, a hash code produced by a keyless hash algorithm fails to ensure message integrity. Message authentication code (MAC) algorithms solve this problem by including a key (either a symmetric secret key or the private key of the sender) in the computation of the hash code; thus, attackers are unaware that the key cannot generate the correct hash code for a modified message. Hash algorithms can also be used in digital signatures (introduced in the next section). Following is a list of widely used cryptographic hash algorithms:

- *Message digests 4 and 5 (MD4 and MD5)*—MD5 splits a message into blocks of 512 bits and then performs four rounds of hashing to produce a 128-bit hash code. MD4 is a weaker hash algorithm that only performs three rounds of hashing. In August 2004, collisions for MD5 were announced by Wang et al. [2]. Their attack technique was reported to take only an hour; on a fairly powerful computer they were able to find an alternative message for a given message, yet both created the same hash code, proving that MD5 is vulnerable to a weak collision attack. Using the same technique, they also devised a method to manually attack MD4 and two other hash algorithms, HAVAL-128 and RIPEMD. MD5 is still widely used in existing systems, ranging from digital signature to file checksum; however, neither MD4 nor MD5 should be considered for future systems due to the collision problem, especially for systems utilizing MD5 to generate digital signatures and digital certificates.
- *Secure hash algorithm 1 (SHA-1)*—SHA-0 was initially proposed in 1993 as a hashing standard by the National Security Agency (NSA) and was standardized by NIST. Later, in 1995, SHA-0 was replaced by SHA-1 after the NSA found a weakness in SHA-0. The weakness was also discovered by Chabaud and Joux. Based on MD4, SHA-1 works on blocks of 512 bits and produces a 160-bit hash code. SHA-1 adds an additional circular shift operation that appears to have been specifically intended to address the weaknesses found in SHA-0. The 160-bit hash code of SHA-1 may not be sufficiently strong against brute force attacks. It has been reported that the same team of Chinese researchers who broke MD5 has found a way to significantly reduce the computational complexity of discovering collisions in SHA-1. As it turns out, the problem of SHA-1 is the hash code size. NIST published three SHA hash algorithms that produce larger hash codes: SHA-256, SHA-384, and SHA-512. These hash algorithms are able to generate hash codes of 256 bits, 384 bits, and 512 bits, respectively. Not surprisingly, they are significantly slower than SHA-1.
- *RACE integrity primitives evaluation message digest –160 (RIPEMD)*—RIPEMD-160 was developed in 1996 by Dobbertin et al. It is an improved version of the original RIPEMP, which was developed in the framework of the EU project RIPE (RACE Integrity Primitives Evaluation, 1988–1992). There are also variants of RIPEMD supporting hash code length of 128 bits, 160 bits, 256 bits, and 320 bits. RIPEMD collisions were reported in 2004 [2], and RIPEMD is not used as often as SHA-1.
- *Message digest and MAC (Message Authentication Code)*—Message digest ensures that if someone in the middle alters a message, the recipient will

> detect it. On the sender side, the sender will hash a message or a file (for checksum computation) to be downloaded using a one-way hashing algorithm (such as MD5 or SHA-1, described above), attach the result (the message digest) to the message, and send it out. Upon receiving the message, the recipient will apply the same hash algorithm to the received message body and compare the result with the received message digest. If they match, the message has been transmitted intact; otherwise, the message has been changed in some way on its way to the recipient, and the recipient may simply reject the message.

If an attacker forges a hash code of a modified message, the hashing algorithm may utilize a cryptographic key as part of the input in addition to the message being transmitted. More generally, a MAC that is computed based on the message and a cryptographic key can be used to guarantee message integrity. If the computation is done using a hash algorithm, such a technique is referred to as HMAC, which essentially uses a keyless hash algorithm and a key to implement the algorithm of a keyed hash algorithm. Well-known HMAC algorithms include HMAC-MD5, HMAC-SHA1, and HMAC-RIPEMD. MAC can also be computed using symmetric block ciphers such as DES; for example, a message can be encrypted using the DES CBC (Cipher Block Chaining) mode. The ciphertext can then be used as MAC. Furthermore, to prevent tampering of the message digest itself, the sender can encrypt the message digest using its own private key so the recipient, with the sender's public key at hand, can be assured that this message has come from the sender. This scheme is referred to as *digital signature* and will be discussed in the next subsection.

As a last note, an attacker may launch a message reply attack by simply resending a number of legitimate messages previously captured. The recipient may be fooled by such legitimate messages. To counteract these attacks, the sender can use a sequence number for each message that is contained in the integrity-protected part of the message. The sequence number keeps increasing so replayed messages will not be accepted.

8.1.3 Authentication

Common authentication mechanisms are digital signature, digital certificate, and PKI, which are described in the following text.

Digital Signature

Digital signature is designed to assure recipients that the senders of messages are really who they claim to be and the messages have not been modified along the way. Similar to a signature in the real world, the sender digitally signs a message, and the receipt is able to verify the authenticity of the message by looking at the digital signature. In other words, digital signature offers authentication of the sender and message integrity.

Digital signing and verification between two parties are conducted as shown in Figure 8.2. The sender:

- Prepares cleartext to send (e.g., an e-mail or a packet).
- Hashes the data using a cryptographic hash algorithm to generate a message digest; hashing is not reversible.
- Encrypts the message digest with the sender's private key, which generates the digital signature that uniquely identifies the sender.
- Appends the digital signature to the original cleartext and sends it to the recipient. Of course, the cleartext can be encrypted using symmetric or asymmetric ciphers.

The recipient:

- Uses the sender's public key to decrypt the digital signature; the result is used in the next step.
- Hashes the received message body with the same algorithm used by the sender.
- Compares the decrypted message digest with the computation result from the previous step; if they are the same, the message must be originated from the sender, and the message has not been altered.

Now let's see if an attacker can impersonate the sender. Without the sender's private key, the attacker has no way to create a valid digital signature for the message because on the recipient side, after the message is hashed, the result will never be

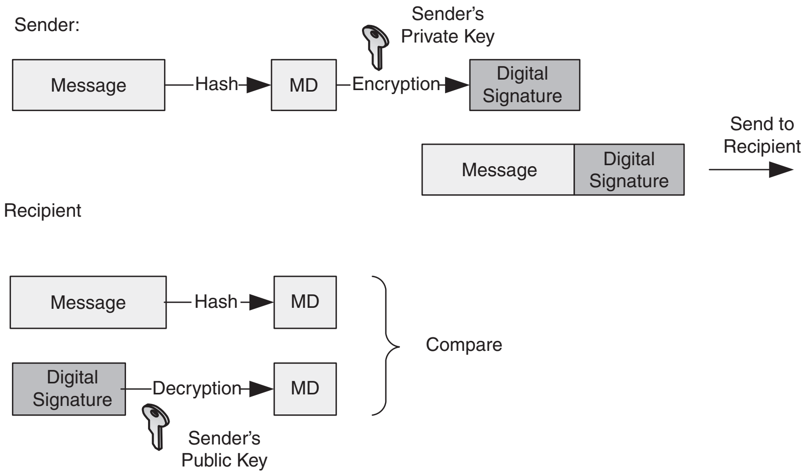


FIGURE 8.2

Digital signature.

the same as the result after decryption of the digital signature. On the other hand, an attacker who chooses to tamper with the sender's message body will also fail, as the hash code of the received message will become inconsistent with that carried in the digital signature.

PKI and Digital Certificate

Asymmetric cryptographic systems (introduced above) assume that a party knows the other's public key. A problem with public authenticity is how someone holding the public key of someone else can be sure that the key does, indeed, belong to that person. What if the distribution of public keys is not at all secure? For example, an attacker could generate and publish bogus public keys of some victims.

The general architecture to address this issue is public key infrastructure (PKI). In a PKI system, the certificate authority (CA) has a public key but its private key is not known to everyone in the system. A single CA PKI is depicted in Figure 8.3(a). To join the PKI system, a user must generate his or her own public/private key pair and ask the CA to certify the public key. The CA will then verify the identity and the associated public key. The CA then signs a digital document stating that the public key really does belong to the person in question. This digital document is a *digital certificate* and should be sent to a recipient whenever the person is about to communicate with some party with public key encryption or digital

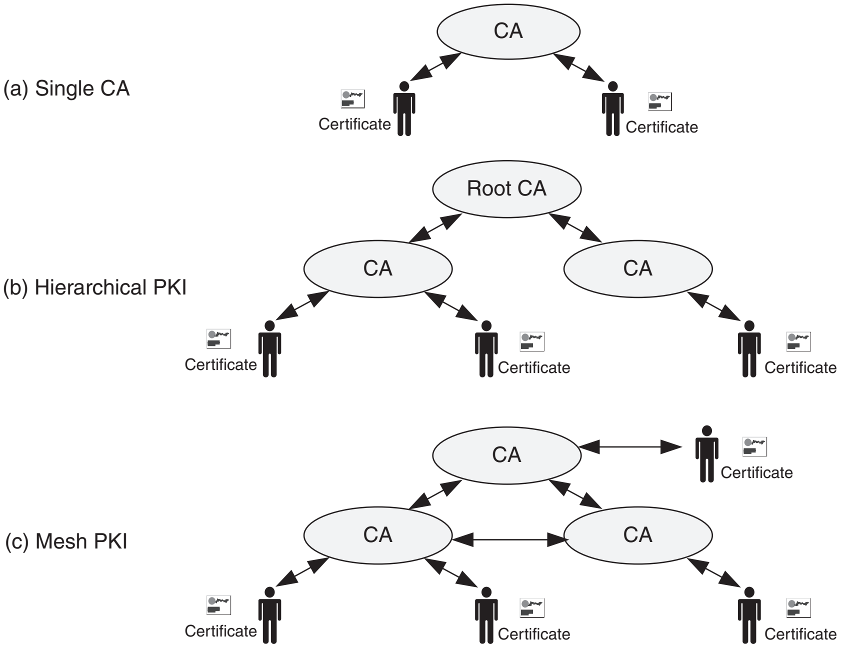


FIGURE 8.3

PKI architecture.

signing. Because everyone in the PKI system knows the public key of the CA, they can check the authenticity of the certificate and thus the public key of the sender. The certificate usually contains the owner's identity, a signature of the CA, and an expiration date. Table 8.2 shows common fields in a digital certificate. The X.509 standard defines the format of a digital certificate.

In reality, a PKI system is organized into multiple levels in a hierarchy to distribute certificate generation and verification among a number of CAs, as shown in Figure 8.3(b). On the top of the tree is the root CA, who is trusted by every user and every other CA. In effect, a chain-of-trust relationship can be established regardless of which low-level CA a user selects, as those CAs can always find a common high-level CA within the hierarchy. Verification is done in the same way as DNS (Domain Name Server) resolution. For example, in a two-level CA system, the public key certificate of a user consists of two parts: (1) a message issued by a high-level CA to certify a low-level CA and (2) a message issued by the low-level CA who will eventually certify the public key of the user. This forms a trust chain of two CAs, and path validation can be conducted. Thus, any party who elects to receive a user certificate (as well as the certificate of the CA certifying the user certificate) must first compute the public key of the low-level CA serving that user and then obtain the user certificate. As the number of levels increases, certificate verification requires more computation. A variance of hierarchical PKI is a trust list architecture, in which some high-level CAs maintain a list of trusted CAs in another hierarchy. A trust chain is therefore established with the trust list instead of a root CA.

A third PKI architecture, mesh PKI, is shown in Figure 8.3(c). There is no publicly trusted root CA in a mesh PKI. A CA in a mesh PKI may choose to trust a subset

Table 8.2 Field in a Digital Certificate

Field	Description
Version	Version number
Serial number	Unique ID of the certificate
Certificate signature algorithm	Encryption and hashing algorithms used to create the signature in the certificate
Issuer	ID of the issuing CA
Validity	Duration for which the certificate is valid
Subject	Owner information
Subject public key info	Subject's public key algorithm (RSA, for example) and public key
Extensions	Additional information regarding the certificate
Certificate Signature Value	Signature of the CA

of other CAs. Users always trust the CA issuing the certificates. Path validation of a user certificate may involve a means to discover the path itself. A bridge CA can be used to link a hierarchical PKI to a mesh PKI. It is not a root CA trusted by everyone; rather, it serves as a common intermediate CA in a trust chain.

8.1.4 Key Management

Key management refers to the process of creating, distributing, and verifying cryptographic keys. It determines how an entity binds to a key. Here, we introduce the Diffie–Hellman (DH) key exchange protocol, RSA, and ECC.

Diffie–Hellman Key Exchange Protocol

The DH key exchange protocol provides a means for two parties to agree on the same secret key over an insecure communication channel. In its simplest form, each party send to the other a number that is computed with a chosen secret number respectively. The same secret key is thus determined based on the number received from the other party; however, if the two numbers are transmitted over an insecure channel, it is computationally difficult for any third party to recover the secret key. The DH key exchange protocol uses a pair of publicly available numbers (p and g) along with the user's random variables for the computation of a secret number. In this case, p is a large prime number and g is an integer less than p, where p and g satisfy the following property: For any number n between 1 and $p - 1$ inclusive, there is a number m such that $n = g^m \bmod p$. Each of the two parties engaging in the DH key exchange protocol will first generate a private random variable. Let's say the variables are a and b. Each party proceeds to compute $g^a \bmod p$ and $g^b \bmod p$ and they exchange results. Then, the shared secret key (k) can be obtained by computing $k = [(g^b) \bmod p]^a \bmod p$ and $[(g^a) \bmod p]^b \bmod p$ at each party. Note that $[(g^b) \bmod p]^a \bmod p = [(g^a) \bmod p]^{\mathrm{b}} \bmod p = (g^{ab}) \bmod p$. No one other than the two communicating parties will know a and b, so it is not computationally feasible to compute k using p, q, and the two public values $g^a \bmod p$ and $g^b \bmod p$.

Note that, although both sides are able to agree on a secret key, there is no way for each of them to be sure that the other side is indeed the person with whom they want to communicate, meaning that no authentication is being performed during the key exchange process. This opens up the protocol to a man-in-the-middle attack, in which an attacker is able to read and modify all messages between the two parties. Digital signature can be applied in this case to prevent man-in-the-middle attacks.

RSA

Designed by Ron Rivest, Adi Shamir, and Len Adleman [3], RSA is a public key algorithm that provides both digital signature and public key encryption. RSA is the public key algorithm used in pretty good privacy (PGP). Key generation in RSA is based on the fact that factoring very large numbers is computationally impossible.

RSA keys are typically 1024 to 2048 bits long, much larger than the largest factored number ever. A message is encrypted using the public key of the recipient. To decrypt the ciphertext, one must know the private key corresponding to that public key. Given the public key and the cipher text, an attacker must factor a large number in the public key into two prime numbers so as to deduce the private key. In addition to message encryption, RSA also provides a digital signature that allows senders to sign a message digest using their private keys. Thus, no one is able to forge a message from the sender unless he or she knows the private key. RSA was patented in the United States in 1983; the patent expired in 2000.

Elliptic Curve Cryptography

An alternative to RSA, elliptic curve cryptography (ECC) is another approach to public key cryptography. It was independently proposed by Victor Miller and Neal Koblitz in the mid-1980s. ECC is based on the property of elliptic curve in algebraic geometrics. An elliptic curve is defined by a set of points (x, y) over a two-dimensional space such that $y^2[+x \cdot y] = x^3 + a \cdot x^2 + b$, where the term in the square bracket can be optional. ECC allows one to choose a secret number as a private key, which is then used to choose a point on a nonsecret elliptic curve. A nice property of an elliptic curve is that it enables both parties to compute a secret key solely based on its private key (the number chosen) and the other's public key. The secret key specific to these two parties is a product of those two private keys and a public base point. A third party cannot easily derive the secret key. NIST has published a recommendation of five different symmetric key sizes (80, 112, 128, 192, 256). ECC is generally used as an asymmetric scheme that allows for smaller key sizes than RSA. The drawback of ECC is the computation overhead associated with the elliptic curve.

Key management in symmetric cryptographic systems poses a different problem. Using stream ciphers, communication between two parties can be encrypted with a secret key only known to the two parties. Naturally it would be better to allow the two parties to frequently change the secret key to reduce the risk of message replay attacks and cipher breaks. For example, the two parties may agree on a new secret key for each new session between them. This secret key is referred to as a session key. In a network environment where many nodes have to communicate with others, a session key can be issued by a trusted third party every time two nodes are about to communicate. This simple scheme requires a node to have only one secret key shared with the trusted third party, relieving it from maintaining a secret key for every other node in the network. An example of such systems is Kerberos (http://web.mit.edu/kerberos/www/).

As a last note in the authentication section, GSP/GPRS systems employ a technology-specific authentication mechanism (the A3 algorithm) for authentication between a base station and a mobile station. The A3 algorithm, along with the A5 encryption algorithm and A8 key management algorithm, are introduced in Section 8.2.

8.1.5 Nonrepudiation

Nonrepudiation refers to a security function of a system that produces evidence to prove that an operation has been performed by an entity. For example, a message recipient should hold a piece of electronic documentation for the message such that the sender cannot deny message transmission. Conversely, the sender must be able to show that the recipient did indeed receive the message. Nonrepudiation of origin proves that the message was sent, and nonrepudiation of delivery proves that the message was received.

Nonrepudiation is generally considered a facet of the security function in electronic transaction settings, as neither sender nor recipient can repudiate a transaction after it is committed. A digital signature appended to a message sent by a sender or an acknowledgement generated by the recipient can be used to provide nonrepudiation. In this case, the digital signature serves as the evidence for nonrepudiation of origin and delivery. Because only the owner of the digital signature knows his or her private key, that person cannot deny transmission of any messages signed by his or her digital signature. One-time passwords are another scheme to realize the nonrepudiation function.

8.1.6 Network Security Protocols

We have discussed security schemes for message confidentiality, message integrity, and message authentication. Those schemes are generally used to secure a communication channel between two parties. Another level of authentication is concerned with user authentication (i.e., verifying the identity of an entity to prevent unintended data access or impersonation). Recall that cryptographic keys are invariably used in those message-centric security mechanisms. Now, let us assume that point-to-point communication channels are secured and look at a network consisting of more than two nodes in which user authentication is associated with proper authorization with respect to data access. For example, in a typical setting, a user elects to log in to a system (a group of machines) in order to read or write a file physically stored somewhere in the system. A user must be authenticated against some security tokens managed at a log-in server before the desired access is granted.

Password

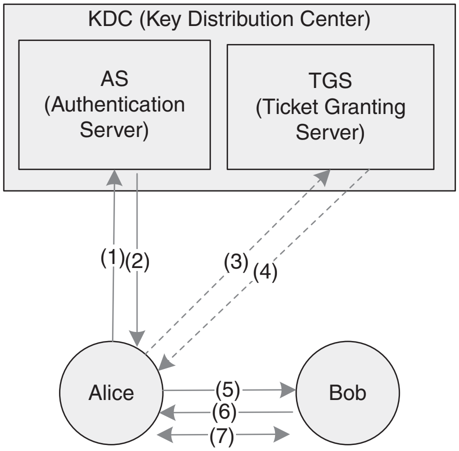
Each account in a multiple-user system is assigned a password. Users can change their passwords but must obey some password creation guidelines to avoid the use of passwords that are too simple. For any network security protocols, cleartext passwords should never be sent over a network. This is the reason why the once popular Telnet protocol has been abandoned in today's networks. On a log-in server, users' passwords are usually hashed. A good password policy should force a user to change passwords once in a while. In addition, other means of human identity can be used to replace passwords. Recent developments in biometrics suggest that fingerprints, voices, faces, and irises can be utilized to identify humans with much better security. The term *biometrics* refers to systems and techniques that make use of features of a person's body for verification and identification. Features of a person include fingerprint, facial pattern, hand geometry, iris, retina, voice pattern, and signature.

Challenge and Response

For challenge and response schemes, the log-in server of a system sends a random message (the challenge) to a user who is willing to authenticate the system. The user applies a security function to the challenge and sends the result back to the log-in server, which performs the same security function and compares its results with those from the user. The challenge and response scheme can be applied in various settings. For example, it can be used to implement a one-time password, in which each password becomes invalid right after it is used. Additionally, the security function itself can be the secret. In effect, no password is transmitted over the network, and the messages subject to interception are different every time, thereby reducing the likelihood of success of an eavesdropping attack.

Kerberos

Kerberos (http://web.mit.edu/kerberos/www/) is a secret key-based network authentication protocol (Figure 8.4). The name *Kerberos* comes from Greek mythology (Kerberos was the three-headed dog that guarded the entrance to Hades).



(1) Alice is authenticated by AS

(2) AS generates TGT (Ticket Granting Ticket)

(3) Alice sends TGT and her identity encrypted by the session key to TGS

(4) TGS verifies Alice's request and then generates a new session key for communication between Alice and Bob, and a service ticket for Alice to pass to Bob

(5) Alice sends the service ticket to BOB and identity encrypted by the new session key

(6) Bob verifies Alice's identity with the service ticket

(7) Alice and Bob start to communicate using the new session key

FIGURE 8.4

Kerberos (version 5).

Kerberos can be viewed as a distributed authentication service that allows a computer program (a client) running on behalf of a principal (a user) to prove its identity to a verifier (a server). In the heart of Kerberos is the key distribution center (KDC), which consists of two logically independent components: an authentication server (AS) and a ticket-granting server (TGS). A user (Alice) who wants to communicate with another user (Bob) must first be authenticated by the AS. To do this, Alice must use her secret key (e.g., her password) to encrypt a challenge sent from the AS. The AS generates a ticket-granting ticket (TGT), which is comprised of (1) a session key encrypted by Alice's secret key (password) for the upcoming communication between Alice and the TGS and (2) a secured temporal credential used to identify Alice's request to the TGS encrypted by the TGS' secret key (which is unknown to Alice) and the session key. Alice then sends the TGT along with an authenticator (i.e., Alice's identity encrypted by the session key of Alice and the TGS) to the TGS. It is the TGS that eventually generates a session key for the upcoming communication between Alice and Bob, after verifying the data in the ticket and the authenticator. At the same time, a service ticket (encrypted by Bob's secret key) for Alice to pass to Bob is also generated.

Finally, Alice sends the service ticket and a corresponding authenticator (her identity encrypted by their session key) to Bob, who verifies if the identity in the service ticket and Alice's authenticator match. If yes, Bob and Alice can begin to communicate with the session key. If Bob is a log-in server of a network system such as in a Windows domain, Kerberos is used to authenticate a user to access shared resources in the network. Kerberos relies on time-stamps and lifespan parameters to prevent message replay attacks. This requires clock synchronization among the participating machines.

Internet Protocol Security

Internet protocol security (IPSec) is a suite of protocols and mechanisms that collectively provides message confidentiality, message integrity, and message authentication at the IP layer. Depending on whether end systems support IPSec, an IP packet can be delivered in one of the two modes: transport mode or tunnel mode. In transport mode, the IP payload is secured in terms of message integrity and authentication when the authentication header (AH) protocol is used or confidentiality when the encapsulating security payload (ESP) protocol is used; but the IP header is not protected. In tunnel mode, a new IP header is used, followed by the encrypted IP packet.

IPSec provides ESP and AH protocols for message security, as shown in Figure 8.5. The AH protocol determines how an IPSec system uses the AH header, which is a hash code of all immutable fields in the IP packet, for message integrity and origin authentication. In contrast, the ESP protocol provides message confidentiality in addition to message integrity and authentication. In transport mode, an ESP header is inserted after the original IP header, and the original IP payload is encrypted. In tunnel mode, a new IP header is used for tunneling, followed by an ESP header and by the encrypted IP packet. In both cases, message integrity and

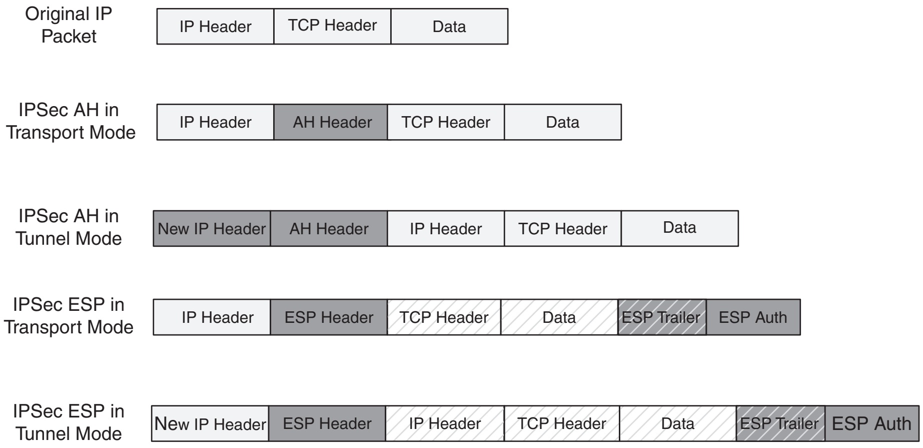


FIGURE 8.5

IPSec.

authentication are provided by an ESP authentication field appended to the end of the packet.

In addition, the Internet key exchange (IKE) protocol is supplied for symmetric key management. The IKE protocol is a hybrid of three key management protocols: Internet Security Association and Key Management Protocol (ISAKMP), Oakley, and SKEME (Versatile Secure Key Exchange Mechanism for Internet). These protocols work together to allow dynamic negotiation of cryptographic keys using the DH key exchange algorithm. IPSec is widely used to implement virtual private networks (VPNs), which enable secure access to a remote network via the public Internet.

Secure Socket Layer

Unlike IPSec, which works at the IP layer, secure socket layer (SSL) and its successor, transport layer security (TLS), are network security protocols at the transport layer. SSL and TLS support any connection-oriented application layer protocols such as HTTP (HyperText Transport Protocol), LDAP (Lightweight Directory Access Protocol), IMAP (Internet Message Access Protocol), and NNTP (Network News Transport Protocol). In reality, SSL and TLS are mainly used in conjunction with HTTPS protocol to secure communication between a web server and a web client, and TLS is being increasingly used with other application protocols such as POP3 (Post Office Protocol 3) and SMTP (Simple Mail Transfer Protocol). The default HTTPS port number on an SSL-enabled web server is 443. SSL requires a server certificate such that the server can be authenticated by a client or a browser according to an RSA public/private key encryption scheme. Subsequent web traffic is encrypted with a 128-bit or longer session key generated by a symmetric cipher such as DES, 3DES, RC2, or RC4. SSL can also be used to authenticate the client, in which case the client must obtain a public/private key pair and a digital certificate.

8.1.7 General Considerations of Mobile Security and Privacy

Mobile security and privacy essentially possess sets of unique characteristics that separate them from wired network security, such as open-air transmission of wireless signals, comparatively low computing power of mobile devices, high error rate of wireless signal transmission, security management for mobility, and location-sensitive security concerns. The need for security is much stronger than in wired networks, yet to build a secure mobile wireless system one must address a variety of constraints unique to the mobile wireless environment. Some security solutions such as those cryptographic ciphers and security network protocols may not be applicable to a mobile computing environment. For example, computationally intensive ciphers may not work on mobile devices, and in many cases the stable network connection required by many network authentication schemes is not always available.

Even if some security mechanisms can be ported to a mobile wireless system, they must be enhanced with sophisticated components so as to provide confidentiality, integrity, authentication, and nonrepudiation in highly varying mobile wireless settings. In addition, many network protocols are designed without security in mind and must be augmented with security considerations. For example, routing protocols in *ad hoc* networks must offer some security to prevent eavesdropping and message tampering. The following is a list of threats in mobile wireless networks:

- *Loss and theft of mobile devices*—Every year hundreds of thousands of mobile devices are lost in airports, hotels, restaurants, etc. This is probably by far the most serious threat to enterprise data and individual privacy.
- *Channel eavesdropping*—An attacker may capture messages transmitted in a wireless channel without being detected.
- *Identity masquerading*—An attacker may impersonate a legitimate user or service provider.
- *Message replay*—An attacker may capture a series of messages between two parties and send them to someone later.
- *Man-in-the-middle attack*—An attacker may intercept and modify messages being sent between two parties or inject new messages without being detected.
- *Wireless signal jamming and interference*—An attacker may use powerful antennas to transmit noisy signals with appropriate modulation in order to disrupt the normal operation of radio receivers.

- *Denial of service*—An attacker may use rogue access points, mobile stations, or specific frequency jamming devices to generate a huge amount of network traffic toward a target computer.
- *War-driving and unauthorized access*—An attacker may use special radio equipment to pinpoint unsecured wireless access points in an area while driving around. Those unsecured wireless LANs, many of which are linked to corporate networks, are wide open to unauthorized users.
- *Virus and wireless spamming*—Small, malicious programs may propagate among mobile device users via short message service (SMS) messages or the wireless Internet. SMS spamming could be another big issue, as subscribers have to pay for that.

In the following sections, security issues in specific wireless networks are discussed in detail.

8.2 CELLULAR NETWORK SECURITY

As more mobile applications are being delivered to cell phone users, the security mechanism employed by underlying traditional cellular systems must be redesigned to adapt to various new network settings. Moreover, because of the extensive use of cell phones and smart phones, a security breach or a network attack may have an enormous impact on every aspect of the modern society, far beyond the scope of the Internet. Emerging cellular systems have provided the means to secure wireless data transmission and e-commerce transactions, in addition to providing a more general authentication, authorization, and accounting (AAA) solution.

8.2.1 Secure Wireless Transmission

Data transmission in a cellular network can be categorized as user traffic or signaling traffic. Four security issues with regard to cellular traffic are user traffic confidentiality, signaling traffic confidentiality, user identity authentication, and user identity anonymity. For user traffic between subscribers and the back-end system, encryption is necessary to ensure confidentiality. Aside from radio layer frequency hopping modulation and code-division multiple access (CDMA) coding schemes, GSM/GPRS, CDMA, universal mobile telecommunications system (UMTS), and cdma2000 all employ some encryption for user traffic to achieve end-to-end security. For user authentication, authentication schemes generally utilize some sort of identity module on the cell phone. We use GPRS and CDMA as examples to show how end-to-end security is implemented in cellular networks. GSM/GPRS (later Third Generation Partnership Project, or 3GPP) defined an encryption protocol based on an A5 algorithm (GEA3 for GPRS), an authentication protocol based on an A3 algorithm, and a cryptographic key management protocol based on an A8 algorithm. Table 8.3 provides a summary of these algorithms.

In a GSM/GPRS network, a subscriber is identified by a unique international mobile subscriber identity (IMSI) stored in the subscriber identity module (SIM) along with the handset (the phone). The SIM also has a secret key (K_i) associated with the IMSI. On the network side, IMSI and its K_i are stored in an authentication center (AuC). The subscriber authentication is carried out in a challenge-and-response fashion, whereby a random number as a challenge is generated and sent to the mobile station by a serving GPRS service node (SGSN). The mobile station uses its K_i to produce a code as the response according to the A3 algorithm. The encryption key (K_c) is derived from the same random number and K_i by the A8 algorithm. On the mobile station, this is performed by the SIM module. Data and voice traffic are encrypted using K_c by applying the A5 algorithm, a stream cipher. It is said that the K_c is 40 bits long, but no official document reveals its actual length. When the mobile station moves around, a temporary mobile subscriber identity (TMSI) is issued by the network to track the mobile station. Whenever a mobile station changes its associated mobile switching center (MSC), it will obtain a new TMSI that is only valid within the location area of the MSC in charge. A TMSI is encrypted with K_c as part of a TMSI reallocation request message and sent to the mobile station. After applying the A5 algorithm with K_c, the mobile station then confirms reception of the TMSI by replying with a TMSI reallocation confirmation message. Thus, the TMSI reallocation process is again a challenge-and-response scheme.

The universal mobile telecommunications system (UMTS)/wideband CDMA (WCDMA) improved GPRS mobile security by introducing large cipher keys of 128

Table 8.3 GSM/GPRS (3GPP) Security Algorithms

Type	Algorithm	Description
Key management	A8	Uses a 128-bit RAND and a 128-bit K_i to produce a 64-bit K_c.
Challenge and response authentication	A3	Uses 1280-bit RAND (the challenge) and a 128-bit authentication key K_i (allocated during user subscription) to produce 32-bit expected response SRES. Implemented on MS SIM and HLR or AuC.
Symmetric encryption	A5	Uses 22-bit COUNT (TDMA frame number) and 64-bit cipher key K_c to produce 140-bit cipher blocks on both BSS and MS for encryption and decryption, respectively.

bits and providing data integrity. Signaling messages and data messages are protected by a KASUMI block cipher protocol that uses the 128-bit cipher key. The same algorithm generates a 64-bit message authentication code to ensure data integrity. Unlike proprietary algorithms used in GSM/GPRS, KASUMI is publicly available for cryptanalytic review.

The Third Generation Partnership Project (3GPP) has formed a working group TSG SA (i.e., Technical Specification Group: Services and Systems Aspects) WG3 Security responsible for the investigation of security issues, and setting up security requirements and frameworks for overall 3GPP systems. The SA WG3 has published a number of technical specifications (TSs) and technical reports (TRs) of security issues ranging from 3G security threats to cryptographic algorithm requirements and specific algorithms to 3GPP and wireless LAN Internet security.

cdma2000 1x uses a 64-bit authentication key (A Key) and an electronic serial number (ESN) to derive two encryption keys for signaling messages and data messages, respectively. The encryption algorithm is AES. cdma2000 1x EVDO uses a 128-bit A Key derived from a DH key exchange. The authentication protocol in cdma2000 networks is cellular authentication and voice encryption (CAVE). Table 8.4 shows a summary of algorithms in CDMA networks. The 128-bit SSD generated by the CAVE algorithm has two equal-length parts: SSD_A and SSD_B. Using

Table 8.4 CDMA Security Algorithms

Type	Algorithm	Description
Key Management	Cellular authentication and voice encryption (CAVE)	Uses a 64-bit reprogrammable authentication key (A Key, allocated with the handset); the electronic serial number (ESN), and a home location register (HLR)/authentication center (AC)-generated random number are used to derive a 128-bit subkey called the shared secret data (SSD), known to the mobile station and its MSC.
Challenge and Response Authentication	CAVE	Uses SSD- and MSC-generated random number (the challenge) to produce an 18-bit authentication signature and a key to replace a well-known value used for voice encoding.
Symmetric Encryption	Cellular message encryption algorithm (CMEA), ORYX, and AES	Uses 64-bit CMEA key derived from part of SSD for signaling message encryption. ORYX is a stream cipher for data messages.

CAVE with SSD_A and a random number (the challenge) generated by the MSC, a mobile station is able to generate an 18-bit authentication signature (the response) and send it to the base station. A mobile station also uses SSD_B to generate a secret key that will be used to scramble the voice. In addition, using the CAVE algorithm, a mobile station can also generate a 64-bit CEMA key and a 32-bit data key. The CEMA key is used to encrypt signaling traffic, and the data key is used to encrypt and decrypt data traffic.

Authentication, authorization, and accounting (AAA) are an integral part of 3G cellular systems. In cdma2000, AAA functionalities are provided by home AAA servers and visited AAA servers along with other mobile IP components. The packet data service node (PDSN) (foreign agent) in a visited network forwards usage data of a mobile station to the home AAA, possibly through a broker AAA. In UMTS, the CN has a home agent and an AAA server. Using serving GPRS service nodes (SGSNs) and gateway GPRS support nodes (GGSNs) as gateways, a mobile station's visited AAA server can communicate with its home AAA server for usage updates, roughly the same procedure as for location updates with the exception that the data being transmitted are related to AAA functions.

8.2.2 Secure Wireless Transaction

Mobile applications are primarily deployed in a heterogeneous network environment in which wireless and wired networks, secured enterprise networks, and wide open home wireless networks coexist and interconnect. One cannot count solely on wireless communication security even though the underlying wireless network is highly secure. Higher layer (network layer or beyond) security mechanisms are invariably required when user traffic is exposed to the unsecured Internet or wireless networks fail to provide the desired security functions. In the following, the wireless transport layer security (WTLS) and WAP (Wireless Application Protocol) identification module (WIM) of WAP and IPSec or SSL VPNs are introduced as the most widely used security protocols on today's mobile devices. Note that they can also be used in other wireless networks, such as wireless LANs and Bluetooth.

Wireless Transport Layer Security

Wireless transport layer security (WTLS), as defined in WAP 2.0, provides message confidentiality, message integrity, and unidirectional or mutual authentication at the transport layer. It is logically identical to SSL/TLS but has been adapted to the wireless environment. Message encryption is performed using RC4, DES, and triple-DES or 3-DES. Message integrity is guaranteed using HMAC. Authentication is based on PKI, using RSA, ECC, or DH. A WAP server (also called WAP gateway) uses a WTLS certificate, a particular form of an X.509 certificate. A WAP client may also use a digital certificate obtained from a CA for authentication, although it is uncommon. The following is a description of session establishment for the case when the WAP server must be authenticated (class 2 service of WTLS).

When a client and a server begin a handshake, they first exchange two random numbers in the "hello" messages. When the public key of the server has been verified with a certificate, the client sends a pre-master secret key encrypted by the server's public key. This pre-master secret key and the random numbers exchanged will be used on both sides to compute a 160-bit master secret key. For data encryption, an encryption key block is calculated based on the master secret key, a sequence number, random numbers exchanged, and a string indicating the party of the calculation. This key block will be eventually used to derive encryption keys for an algorithm such as RC4, DES, or triple-DES that has been negotiated during the "hello" message exchange.

WTLS specifies keyed hashing algorithms such as SHA-1 and MD5 for the computation of MAC over compressed data. For mobile devices with limited computing power, a light overhead SHA_XOR_40 algorithm is also provided in an earlier version of WTLS. The key used during MAC computation, also known as the MAC secret, is also derived from the encryption key block. In order to make denial of service attacks more difficult to accomplish, the WTLS specification suggests that a WAP server should not allow an attacker to break up an existing connection or session by sending a single message in plaintext from a forged address.

Figure 8.6 depicts the WTLS architecture. At its heart is the record protocol, which interfaces with the wireless datagram protocol (WDP) and the wireless transport protocol (WTP) and is responsible for data encryption and integrity verification. The handshake protocol defines the negotiation of cryptographic parameters such as algorithms, authentication schemes, and compression methods. When the negotiation is complete, the change cipher spec protocol is performed, indicating that the party is ready to use the negotiated mechanism. After that application, data can be exchanged according to the application data protocol.

An earlier version of WTLS is not secure, as researchers have found some security problems [4]. For example, in WTLS predicable initialization vectors (IVs) may lead to encryption key breach, and the SHA_XOR_40 algorithm does not provide

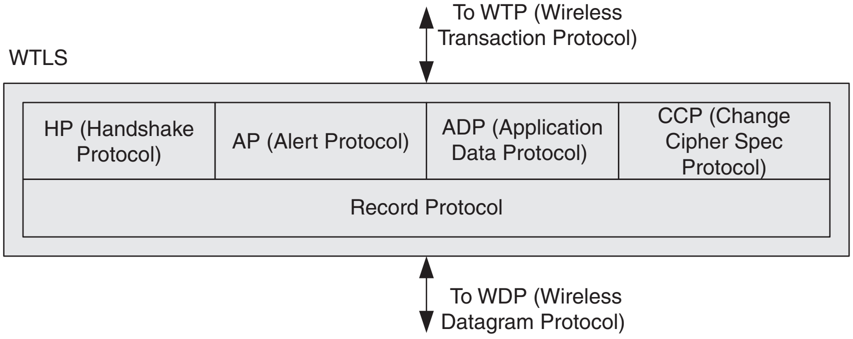

To WTP (Wireless Transaction Protocol)
WTLS
HP (Handshake Protocol)
AP (Alert Protocol)
ADP (Application Data Protocol)
CCP (Change Cipher Spec Protocol)
Record Protocol
To WDP (Wireless Datagram Protocol)

FIGURE 8.6

WTLS architecture.

message integrity if stream ciphers are used. In light of these problems, the latest version of WTLS (version 06-Apr-2001) has made significant changes; for example, the SHA_XOR_40 algorithm has been removed.

WAP Identification Module

In order to seamlessly integrate WAP into an e-commerce environment, a WAP client must be authenticated with respect to mobile device identity. A tamperproof WIM module can be embedded into a WAP client device for this purpose. It could be a component of the SIM card or an external smart card containing the following information: a public/private key pair of the device for signing and another pair for authentication, manufacturer's certificates, and user certificates or their URLs. A WIM module implements the WTLS class 3 service, allowing the WAP client associated with it to be authenticated. This class of service specifies that, in addition to server authentication during the handshake, the client must generate a digital signature using one of its public/private key pairs stored in the WIM module, enabling nonrepudiation of client messages.

As a similar wireless web platform, iMode also provides SSL-based server authentication, message encryption, and integrity. Because iMode is a proprietary architecture, details of its security mechanisms are not publicly available. Other wireless web platforms have been developed by Japanese companies, such as EZWeb (KDDI) and J-Sky (J-Phone). Although internals of those systems are not revealed to the public, it is commonly believed that they offer the same set of security services based on SSL or TLS.

IPSec/SSL VPNs

IPSec/SSL VPNs are widely used in mobile wireless networks to allow for secure remote network access. These protocols are transparent to the underlying radio technologies used for wireless communication. As long as a network is IP based, theoretically IPSec will work without a problem, although in reality there are some problems with respect to the nature of wireless transmission and mobility. For example, a VPN tunnel may be interrupted during handoff. Unlike IPSec VPNs, which provide secure access to a network, SSL VPNs enables secure remote access to an application inside a network.

Mobile VPN is particularly useful when a mobile device is used by a salesperson, field engineer, or other type of mobile worker wishing to remotely access an enterprise network via the Internet. A mobile VPN, based on either IPSec or SSL, could solve the problem. Aside from a VPN client on the mobile device, a VPN gateway must be set up for client authentication and data encryption/decryption. A problem with using VPN is related to U.S. export control on cryptography, which basically imposes strict control over the export of cryptographic software and hardware for national security considerations. Strong cryptographic systems such as 128-bit key VPNs are not allowed to be exported unless certain licenses have been obtained. Worldwide corporate networks are at risk when VPN clients in overseas offices use 40-bit encryption.

Aside from these two protocols, smart phones running an advanced operating system such as Windows Smartphone allow for normal SSL to be used within a mobile web browser. It is expected that higher layer security protocols will be directly ported onto relatively powerful mobile devices such as smart phones.

8.3 WIRELESS LAN SECURITY*

Because more cell phones and smart phones are being equipped with Wi-Fi interfaces, related security problems of IEEE 802.11 wireless LANs have become a hot topic, especially after numerous serious vulnerabilities of wired equivalent privacy (WEP), the security mechanism of 802.11, were discovered. Understandably, when the 802.11 wireless LAN standard was developed, security was apparently not a top priority. The "wired equivalence" design rationale essentially led to some earlier versions of wireless LAN security solutions that clearly failed to deliver security functions they were supposed to provide. Many Wi-Fi products in use are based on these flawed protocols and mechanisms. Fortunately, the IEEE 802.11 working group has offered several new standards with enhanced security. Wireless LAN products often incorporate enhanced security as an option in addition to wide-open configurations. For example, WAP has been required in all new Wi-Fi certified products since 2004, and WPA2 (for 802.11i) was required for Wi-Fi certification beginning in 2005.

Security risks in wireless LANs include eavesdropping, unauthorized access, masquerading, man-in-the-middle attacks, denial of service (DoS), and rogue access points:

- *Eavesdropping*—Eavesdropping is highly possible because the coverage of wireless signals is quite difficult to determine, and anyone within the range with an appropriate interface will be able to pick up the signal and intercept ongoing data transmissions at will. Weak encrypted signals can be cracked with modest effort. Powerful tools such as AirSnort and Kismet made wireless eavesdropping on unsecured wireless LANs much easier.
- *Unauthorized access*—Unauthorized access happens when a home or enterprise wireless LAN operates in default configuration mode, which permits anyone to use its Internet access as well as other resources shared in the network.
- *Masquerading*—Many wireless LANs use wireless adaptor's MAC address (physical address) as filters. Thus, attackers may masquerade themselves by spoofing MAC addresses. This can be done in conjunction with eavesdropping.

---

*Here, we use the most popular wireless LAN standard (IEEE 802.11) for discussion.

- *Man-in-the-middle attacks*—Wireless LANs are designed to allow an access point to authenticate a station but not the other way around; hence, a station cannot be sure that the access point in question is what it claims to be. Attackers may pretend to be an access point sitting between a station and a real access point to intercept, modify, and forge packets.
- *Denial of service (DoS)*—DoS is very common on the wired Internet. Many machines are organized to attack a single website, making it unable to service legitimate users. In wireless LANs, attackers may use rogue APs, their own stations, or other non-802.11 spectrum jammers to send a large amount of forged 802.11 management or control frames or broad-spectrum noise. The IEEE 802.11 MAC protocol also has been shown to be vulnerable to DoS attack [5].
- *Rogue access points*—Due to the ease of network setup and configuration, one may quickly build a small insecure wireless LAN and make it work instantly by connecting it to the wired back-end; hence, the entire wired network may become insecure because of the rogue wireless LAN.

8.3.1 Common 802.11 Security Myths

In practice, wireless LANs are often wide open without any access control or simply employ a MAC-based (here MAC refers to the adapter's physical address) access control list (ACL) to authenticate legitimate mobile stations. An ACL is essentially a list of MAC addresses that are permitted to access the network. Those data frames not originating from legitimate MAC addresses will be rejected by the access point without going through further authentication. As in a wired LAN, a MAC address in a frame header is always transmitted in cleartext regardless of the encryption method in use, allowing anyone to gather a list of MAC addresses of stations associated with an access point. An attacker can forge data frames that use those authorized MAC addresses and gain access to the network; therefore, contrary to common belief, the MAC base access control solution does not solve the problem.

Another common security myth associated with 802.11 is the use of an extended service set identifier (ESSID). Because an ESSID identifies an access point, many believe that by disabling beacon messages containing the ESSID of an access point an attacker will not be able to determine the ESSID and thus cannot associate to the access point. In fact, this does not prevent an attacker from getting the ESSID because it is still sent in probe messages when a client associates to an access point; also, many wireless LANs use default, well-known ESSIDs.

Given the fact that a large number of wireless LAN access points are being used and there is no effective way to prevent wireless LAN signals from traveling far, it is tempting to get free access to adjacent wireless LANs while walking, driving, or even flying by using appropriate wireless LAN equipment. In an effort to detect wireless LANs in a regional area, some people have been intensively engaged in activities known as war walking, war driving, and war flying [6]. In all cases, a PDA or a laptop computer with a wireless LAN interface and a global positioning system (GPS) receiver, a handy software tool such as Net Stumbler (http://www.netstumbler.com/) or Air Magnet (http://www.airmagnet.com/), and an optional high-gain antenna are all it takes to produce a so-called wireless access point (WAP) map of access points, either secured (using WEP/WPA/WPA2 or higher layer security measured) or unsecured. With a powerful antenna, a war driver could be many miles away from the physical location of a wireless LAN yet still manage to pick up its signals. Figure 8.7 is a Wi-Fi map of Seattle made by students at the University of Washington (http://depts.washington.edu/wifimap). The dots in the figure represent 802.11 access points (secured and unsecured) within reach of the war drivers. Unsecured wireless LANs detected by war driving not only offer war drivers a free ride on the Internet but also invite attackers to obtain remote access to a network without being filtered by firewalls or detected by intrusion-detection systems.

8.3.2 WEP Vulnerability

The service set identifier (SSID)-based access control indeed does not offer any security functions. Besides, it is common sense that wireless communication should be encrypted and properly authenticated. WEP is the first security mechanism for wireless LANs. A shared secret key of 40 or 104 bits is used by all participating stations within a BSS (Basic Service Set) bounded to the same access point. The encryption algorithm is RC4. For every packet sent between a station and the associated access point, a 32-bit integrity check value (ICV) is computed according to a CRC-32 algorithm. Then RC4 uses a 64-bit key to encrypt the data and the

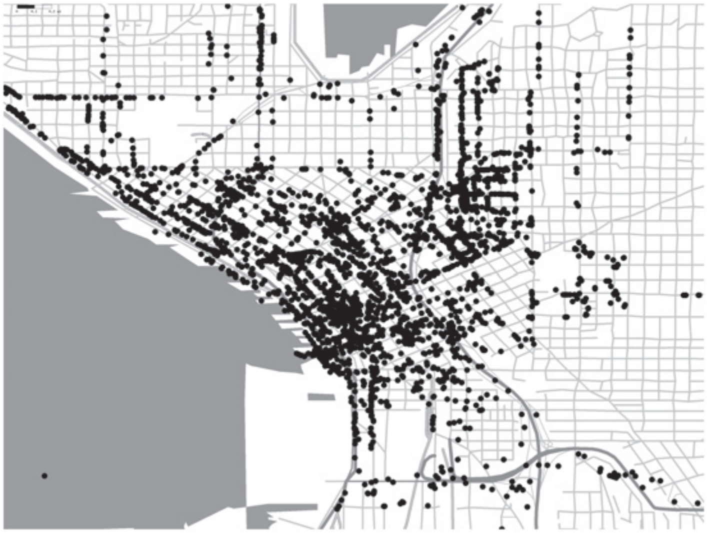
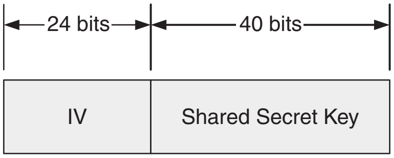


FIGURE 8.7

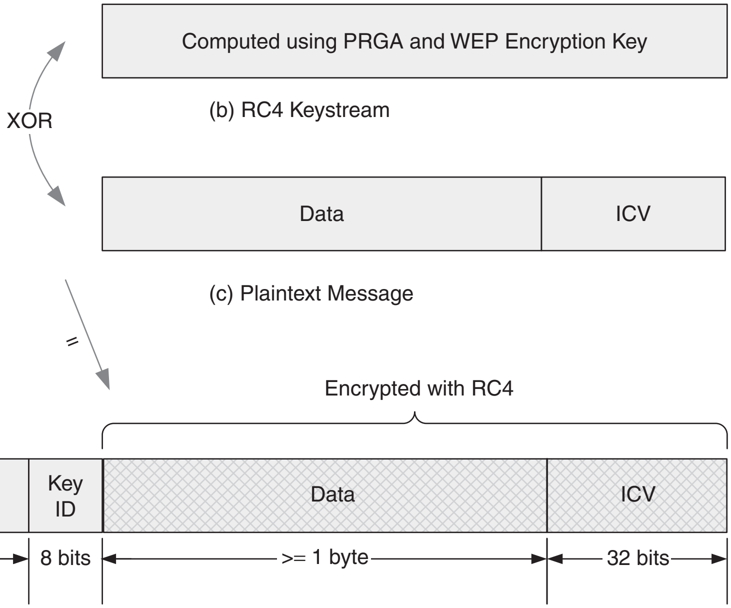
A Wi-Fi map of Seattle, WA (Courtesy of University of Washington).

ICV. The encryption key is composed of a 24-bit randomly generated initialization vector (IV) and a 40-bit shared secret key, as shown in Figure 8.8(a). Using the 64-bit encryption key, the pseudo-random generation algorithm (PRGA) of RC4 computes a keystream, which will be XORed with a plaintext message (see Figures 8.8(b) and (c)). To let the other party know the IV, it is added to the encrypted payload data as part of the packet (the ciphertext), as shown in Figure 8.8(d).

Wired equivalent privacy (WEP) is known to have numerous security problems. The first problem is the lack of key management, such as the DH key exchange protocol. The secret key must be distributed by other means of communication and is subject to social engineering attacks, where attackers trick legitimate users of a system in order to obtain passwords, addresses, or other sensitive information. As the network grows, more stations must be informed of the same secret key, and



(a) WEP Encryption Key



(b) RC4 Keystream

(c) Plaintext Message

(d) An Encrypted Message

FIGURE 8.8

WEP encryption.

it would be quite cumbersome to change the secret key for security reasons. Very often the secret key is not secret any more after some time.

The second problem with WEP is the 24-bit IV. Because each packet transmitted has an IV, it is possible that the same IV will be used again after some time. (The code space of IV will exhaust after 2^{24} packets have been sent.) On the other hand, RC4 has been found by Fluhrer et al. [1] to have a severe weakness in its key scheduling algorithm; when an encryption key is constructed by the above-mentioned method, an attacker will be able to derive the 40-bit secret part of the encryption key by analyzing those packets that share the same encryption key (secret key + IV) [1]. This attack is referred to as the FMS attack. It has been shown that a WEP key can be cracked in a matter of several hours.

The third problem with WEP is the CRC-32 algorithm used to calculate the ICV [7]. CRC in itself is a simple mechanism for detecting random errors; it was not designed to detect deliberate data falsification. In fact, it has been shown that it is possible to modify the encrypted payload of an 802.11b message without disrupting the checksum (ICV). Furthermore, the CRC-32 algorithm does not involve any keying function, such as HMAC. Thus, an attacker who knows a keystream that corresponds to an IV can safely inject forged packets into the BSS.

8.3.3 802.11 Authentication Vulnerabilities

The IEEE 802.11 wireless LAN specification defines two authentication modes: open and shared key authentication. The default open authentication imposes no authentication on a station that wants to communicate with the access point. In the shared key mode, a challenge-and-response scheme is used. Upon receiving an authentication request from a station identified by its MAC address, the access point responds with a 128-byte randomly generated challenge text in cleartext. The station then encrypts the challenge text with a shared key using RC4 and sends the result back to the access point. The access point uses the same shared key to decrypt the response. If the decrypted value matches the challenge text, the station is authenticated and can proceed to send and receive messages in the BSS; otherwise, the station is rejected.

As mentioned earlier, the problem of this authentication mechanism stems from RC4, stated in a paper by Fluhrer et al. [8]. An attacker who obtains a large number of challenge-and-response authentication sequences corresponding to WEP encryption keys (the same IV) can easily deduce the keystreams produced by RC4 by leveraging those weaknesses described in the previous section. From that point, the attacker can authenticate himself to the access point by correctly responding to any challenge texts using the keystream without knowing the shared secret key. Even worse, with the keystream, the cleartext of those messages being analyzed can be revealed by simply XORing the ciphertext against the keystream, exactly the same operation that the associated access point should perform. Using tools such as WEPCrack (http://wepcrack.sourceforge.net/) or AirSnort (http://airsnort.shmoo.com/), it would not take long to crack a WEP key.

8.3.4 802.1X, WPA, and 802.11i

To address the security issues of WEP, one method suggested is to build security overlay on top of the insecure wireless LAN. VPN is often used in practice. 802.11i, which was complete in 2004, was designed to address wireless LAN security issues. 802.1X is a security standard for a more general LAN environment. The Wi-Fi protected access (WPA) protocol has been developed by the Wi-Fi Alliance as an interim solution for 820.11i; hence, 802.11i includes WPA features and some new features, such as AES, CCMP (see discussion below), preauthentication, and key caching for fast handoff.

The IEEE 802.1X standard enables port-based mutual authentication and flexible key management in an IEEE 802 local area network. It does not specify a single authentication method but uses the extensible authentication protocol (EAP) as the underlying authentication framework to support various authentication methods such as smart cards, one-time passwords, and certificates. When an unauthenticated supplicant (a client) attempts to connect to an authenticator (a wireless access point), the authenticator opens a port for the supplicant to pass only EAP authentication messages to the back-end authenticator server, which could be, for example, a remote dial-in user service (RADIUS) server. Initially designed for authentication and authorization of dial-in modem access, RADIUS as a protocol (standardized in RFC 2058) has been augmented to facilitate any form of secure remote access with respect to authentication, authorization, and accounting. The supplicant submits its identity to the authentication server, which makes the decision as to whether or not the supplicant should be granted access to the LAN. The authentication server will send either "accepted" or "rejected" to the authenticator. If the result is "accepted," the authenticator will change the client's port to an authorized state, meaning that the port can be used to pass any other additional traffic. As shown in Figure 8.9, 802.1X can be integrated with an existing AAA infrastructure such as RADIUS to provide user-based centralized authentication.

Wireless protected access (WPA) is an interim solution to wireless LAN security that is required by the Wi-Fi Alliance. WPA is backward compatible with WEP in place on widely deployed wireless LAN devices; WPA only requires software or firmware upgrades to existing systems. Each station using WPA will use a different 128-bit encryption key for RC4 data encryption, which can be "refreshed" frequently. The protocol enabling these features is temporal key integrity protocol (TKIP). Key elements of TKIP are listed as follows [9]:

- *Michael*—Michael is a message integrity code (MIC) algorithm that uses a 64-bit key, called the MIC key, to produce a 64-bit tag (a MAC) for a packet, in addition to ICV. Michael is designed to impose dramatically lowered computational overhead on a mobile station than are other MAC algorithms.
- *Per-packet key mixing*—TKIP employs a key mixing function that takes the base WEP key, source MAC address, and packet sequence number as inputs

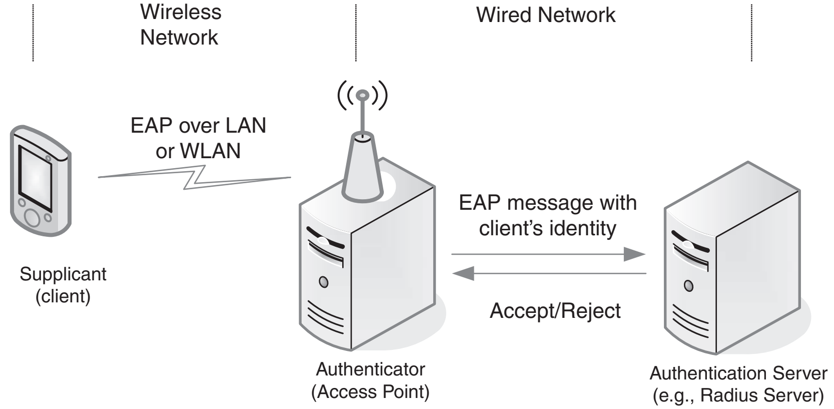

Wireless Network
Wired Network
EAP over LAN or WLAN
EAP message with client's identity
Accept/Reject
Supplicant (client)
Authenticator (Access Point)
Authentication Server (e.g., Radius Server)

FIGURE 8.9

802.1X in a wireless LAN setting.

and produces a new 128-bit WEP key for each individual packet. The mixing function is carried out in two phases to reduce computational overhead.

- *Packet sequencing*—Each packet has a 48-bit sequence number, which will be further used to compute the encryption key. This feature defeats message replay attacks.

WPA adopted IEEE 802.1X to provide both authentication and key management. For enterprise networks where a separate AAA server such as a RADIUS is in place, WPA can be integrated with the AAA server for authentication and key distribution. In WPA and WPA2-Enterprise (WPA2 is the product certification available through the Wi-Fi Alliance for 802.11i compatible products. Both WPA and WPA2 have two authentication modes: Enterprise and Personal), the AAA server authenticates individual users and then delivers per-session pairwise master keys (PMKs). In WPA and WPA2-Personal, all stations and the AP have the same pre-shared secret key (PSK) used for both group authentication and PMK. In both cases, the PMK is not used for encryption; it is mixed with the station's MAC and an IV to derive a pairwise temporal key (PTK), which in turn will be used to deduce the AES encryption key.

The encryption key and MIC key used by TKIP are derived from a master key generated by 802.1X. Frequent key changes enabled by 802.1X allow the encryption key and MIC key used by the TKIP to be refreshed every once in a while, thus reducing the risk of key breach due to eavesdropping. Created by the Wi-Fi Alliance, WPA is supported by a large number of device vendors.

Because WPA serves as a quick patch to WEP, it effectively makes it more difficult to compromise a wireless LAN. The downside of WPA is that it is rather complicated to implement, which could give rise to more security risks. It is also not efficient to introduce an additional MIC key other than the encryption key and use both ICV and MIC for message integrity. Unlike WPA, 802.11i is designed to provide a long-term solution to 802.11 security. Like WPA, it employs 802.1X as the underlying authentication mechanism. Other key features of 802.11i are:

- Countermode–CBC–MAC protocol (CCMP)—Like TKIP, CCMP provides message confidentiality and integrity but uses AES as the cipher instead of RC4. The cipher block chaining message authentication code (CBC–MAC) protects both header and data integrity. The 128-bit encryption key is also used for computation of the 64-bit MAC. The IV is still 48 bits.
- *Pairwise key hierarchy*—802.11i does not compute an encryption key for each packet; instead, the same PMK generated by the 802.1X authentication procedure is used for all packets during an association. PMK is first used to derive a PTK by the access point and the station after proper handshakes between them. The AES encryption key is further deduced from PTK.
- *Key caching and preauthentication*—A user's credentials are kept on the authentication server; thus, when the user leaves and returns shortly, it is not necessary to prompt the user for log-in information; the reauthentication is done transparently. Preauthentication enables a station to be authenticated to an AP before moving to it. Both schemes are designed to speed up authentication in supporting fast handoff.

It should be noted TKIP is also part of 802.11i, but it should only be considered as a short-term solution. Table 8.5 presents a comparison of the three security protocols for 802.11 wireless LAN.

Table 8.5 802.11 Security Protocols Comparison

	WEP	WPA	802.11i
Stage	Initial security mechanism; insecure	Intermediate solution (a snapshot of 802.11i taken in 2002)	Long-term solution (completed in 2004) (WPA2 certifies 802.11i products)
Encryption Algorithm	RC4	Enhanced RC4	AES
Key Length	40 bits	128 bits refreshable	128 bits
Key Management	None	802.1X EAP	802.1X EAP
Message Integrity	CRC-32	Michael (including header)	CCMP (including header)
Logical Equivalence	None	802.1X + TKIP + RC4	802.1X + CCMP + AES

8.4 BLUETOOTH SECURITY

As a simple personal area network (PAN) solution, Bluetooth has become the *de facto* standard interface on cell phones and PDAs. People use Bluetooth to transmit files between a mobile device and a desktop computer or between two Bluetooth-enabled devices. Bluetooth earphones enable voice over Bluetooth channels within a short range. Even though the Bluetooth signal can travel only a very limited distance (usually less than 10 m), there are still security issues with respect to data confidentiality and authentication. The Bluetooth SIG (Special Interest Group) has incorporated a security architecture into the official Bluetooth specification.

8.4.1 Bluetooth Security Architecture

Recall that the Bluetooth specification defines a number of "profiles" for different types of typical usages, such as a dial-up networking profile, fax profile, headset profile, LAN access profile, file transfer profile, and synchronization profile. Each profile has been specified with a set of protocols suitable for those applications falling into the profile. In providing security for various applications, the Bluetooth SIG has defined a number of profile security policies, each of which specifies recommended baseband security options and protocols for different usage models and profiles. Aside from frequency hopping, the basic Bluetooth baseband security mechanisms are listed below:

- *Challenge-and-response authentication*—If device A wishes to be authenticated by device B, device B will send a 128-bit random number (RAND) to device A upon being requested to do so by device A, which uses a 128-bit secret authentication key (link key), RAND, and its 48-bit device address (BD_ADDR) to compute a response according to an algorithm called E_1. When the response is received at device B, device B performs the same computation and compares the result with the response. If they match, then device B is authenticated. Bluetooth devices in a piconet of multiple devices use a shared link key for mutual authentication between two devices. The same link key is also used to derive the encryption key.
- *Per-packet encryption using* E_0—Bluetooth devices may use an encryption key of length 4 to 128 bits, subject to an individual country's regulations. The encryption key is generated by an E_3 algorithm each time the device enters encryption mode. Because communication is always between a slave and a master, the master should initiate the encryption sequence by sending a RAND to the slave. On the slave side, it performs the E_0 algorithm that takes the encryption key, the device address of the master, current clock value, and RAND to compute a keystream. The keystream is then XORed with the packet payload to produce to ciphertext.

In a Bluetooth piconet, a session refers to the period of time a device stays in a piconet. A link key can be either a semipermanent key or a temporary key, depending on the application.

A semipermanent link key allows a device to use the same link key to connect to other devices in a piconet after a session is over. This is useful when some devices must communicate frequently once in a while. A temporary link key is valid only within a session and will be discarded when the session is over. For different scenarios, four different types of link keys are defined. Below is a summary of these keys and when they should be used:

- *Combination link* key is used for each new pair of Bluetooth devices if they decided to use this type of link key. The procedure to establish a combination link key between two devices is called pairing, in which both devices generate a random number and use it to produce a key. They then exchange those random numbers and compute the combination key. It is used for multiple connections from a single device.
- *Unit link* key is specific to a single device and is stored in nonvolatile memory. It is used in installation or when the device is first activated and is never changed afterwards. A device can use another device's unit key as a link key. Which link key should be used is determined during initialization. It is used for communication between two trusted devices.
- *Master link* key is a temporary link key generated by a master device to replace the current link key. It is used for point-to-multipoint communication such as a master broadcasting to its slaves.
- *Initialization link* key is generated using a shared PIN code and device address. The PIN code must be entered to both devices. It is used only to protect initialization parameter transmission when no other keys are available during Bluetooth pairing.

Bluetooth security profile policies have provided general recommendations as to what protocols and algorithms as well as keys should be used in different settings. For specific applications, however, care must be taken to ensure that desired security functions or countermeasures to possible attacks are implemented.

8.4.2 Bluetooth Weakness and Attacks

The use of a PIN code during pairing presents some security risks [10]. The length of a PIN can be between 8 and 128 bits. It could come with the device or can be selected by the user. Prior to link key exchange, an initialization key will first be computed, which in turn uses the PIN code. An attacker may make an exhaustive search over all possible PINs up to a specific length. To verify its guess, the attacker only needs to eavesdrop on the communication channel between two victims to capture random numbers in cleartext and perform the initialization key algorithm. When the PIN code is obtained, the attacker can compute the initialization key and the link key. Eventually, the encryption key can also be obtained, and the communication between those two devices is completely compromised. For this reason, longer PIN codes are strongly suggested by the Bluetooth SIG. An even better countermeasure to PIN attacks is to conduct initialization of two devices in a private and closely secured environment where no wireless communication can be eavesdropped.

The nature of the Bluetooth technology allows mobile device manufacturers to choose a set of configurations optimized for a specific application model. Although this does offer some flexibility to mobile device manufacturers and effectively promotes the technology, it also results in security risks to some extent because in some cases security mechanisms are not well implemented or not taken into account, even if the security building blocks are clearly specified in Bluetooth specification. The five types of attacks targeting Bluetooth implementation problems are:

- Bluesnarfing
- Bluebugging
- Bluejacking
- Back-door attack
- Virus and battery draining attack

In a Bluesnarfing attack, an attacker uses modified Bluetooth equipment and directional antennae to capture data from some Bluetooth devices that could be a mile away. The weakness being leveraged in this case is a default insecure mode enabled by some mobile device manufacturers (see below for details). After successful Bluesnarfing, everything on the device is exposed to the attacker.

In a Bluebugging attack, an attacker may remotely control a Bluetooth device, intercepting or rerouting communication without a trace. Bluesnarfing and Bluebugging attacks are mainly targeting cell phones with a Bluetooth interface. They usually require the victim devices to be in "discoverable" mode; that is, the device will respond to discovery queries sent from other Bluetooth devices. It turns out that many cell phones are in this mode by default, which makes them susceptible to these attacks. Worse, a brute force MAC address scan could possibly discover those devices that are not in "discoverable" mode, aided by tools such as RedFang and Bluesniff (http://bluesniff.shmoo.com/); thus, Bluetooth war walking or war driving (i.e., the activity of discovering Bluetooth devices in the proximity) are also possible using these tools.

Bluejacking involves sending unsolicited messages to a Bluetooth cell phone utilizing a security vulnerability in the Bluetooth handshake protocol when two devices are pairing for mutual authentication. During the handshake, the other party's device name will be displayed. Thus, by manipulating the device name, an attacker can send anonymous messages or broadcast messages (proximity spamming) among visible devices. Contrary to public perception, Bluejacking does not imply hijacking of a Bluetooth device. Personal data on a device remain secure and the device is still under the total control of the user, but it does make the victim worry about the security of the device because unwanted messages from someone are being displayed on the device.

A back-door attack allows an attacker to take advantage of a secretly established trusted "pairing" relationship such that the target Bluetooth device can be remotely monitored and controlled without the user's notice. Not only can personal data such as phone books, business cards, calendar, pictures, and e-mail be downloaded from the target device, but also all services available on the device, such as the cellular network connection, built-in camera, audio recorder, or music player, may be accessed and surreptitiously controlled.

The insecure "discoverable" mode of Bluetooth provides a vehicle for mobile virus and worm propagation. Although today's mobile operating systems have imposed strict security mechanisms whereby users are prompted when any installation of programs is about to occur, most people do not even bother to read the warning message and simply click "OK." Worms such as the Cabir worm have certainly demonstrated that cell phones can easily be infected by a mobile virus. Several variants of the Cabir worms have spread among smart phones running Symbian OS with Bluetooth configured in the "discoverable" mode. As a worm, the program tries to propagate by scanning for vulnerable cell phones using Bluetooth and then sends itself to those victims. A side effect of this worm is that the device's battery drains quickly while the worm is constantly scanning for other devices. Other forms of battery draining attacks use some properly powered attacking Bluetooth device to query a victim repetitively, effectively disabling the device after some time. Code signing is a defending technique against these threats. Only those programs developed by trust vendors will be registered and digitally signed, so users have the chance to reject any unsigned downloaded code.

It is clear that, in fighting with mobile viruses, users have to bear the responsibility to be alert to any suspicious programming. Mobile antivirus software may also help users detect any possible infections.

8.5 *AD HOC* NETWORK SECURITY

Security issues in mobile *ad hoc* networks encompass a much broader range of challenges, in addition to secured routing at the network layer. As communication in a MANET involves one-hop link layer protocols between two directly connected nodes and multihop packet routing protocols across a set of nodes, the security mechanism in MANET should also take into account both the link layer and network layer accordingly, assuming the wireless physical layer is properly secured.

8.5.1 Link Layer *Ad Hoc* Security

For a MANET application, end-to-end security service can be provided by authentication and encryption, which in turn rely on lower layer security protocols to function. IEEE 802.11 WEP is an example of a link layer security mechanism that unfortunately fails to protect one-hop communication between a mobile station and an access point. As discussed earlier, 802.11i has been designed to address the problem of WEP. Specifically, in the distribution coordination function (DCF) mode, when a node senses the channel and finds out it is used by other transmissions, it will initiate a binary exponential back-off procedure waiting until the next try. This scheme does not guarantee any fairness over channel access. In fact, it favors the last node among contending nodes. Therefore, one heavily loaded node may keep occupying the channel whereas a lightly loaded node may have to back off many times. Modifications to the back-off scheme have been proposed, mainly to penalize those misbehavior nodes with a large back-off value.

The principle idea of protocols is to add security extensions to traditional *ad hoc* routing protocols. Note that secured *ad hoc* routing protocols can be categorized as "proactive" security services that are based on node authentication and message confidentiality [11] and the assumption that a node will forward messages according to its routing table or routing mechanism. When a node is compromised and does not forward messages as expected, "reactive" schemes such as ACK (Acknowledgement)-based malicious node detection and coordinated rating are needed.

8.5.2 Key Management

Node authentication in MANET is much more complicated than in a fixed network because of the nature of transient network organization and dynamically changing network topology. Indeed, there is hardly a centralized trusted authority in MANET. And, even when there is, it may not constantly be accessible to every node in the network. Thus, a PKI-based authentication scheme is not directly applicable to MANET. To provide authentication among mobile nodes in such a distributed environment, threshold cryptography can be used.

Threshold cryptography essentially distributes cryptographic functions of an individual node to each node in a group, thus eliminating central authority. It is based on the idea that, even if some individual nodes may be compromised, the majority of a group can always be trusted. In its simplest form, in the context of CA, each node in a group of n nodes holds a distinct piece of the group's private key, and any t nodes can work together to perform the security function as a whole for the group, but any $t - 1$ nodes cannot. This scheme can be used to distribute the security function (i.e., providing a certificate for a node's public key) of a single CA over a number of servers [12]. Each server (a fairly stable node in an *ad hoc* network) holds a share of the private key (k). It computes a public key corresponding to its private key share. The public key (K) corresponding to the private key (k) is known to each server. To sign a digital certificate, each server generates a partial digital signature using its private key share. A combiner (a server that directly interfaces a service requester) needs to gather t such partial signatures in order to produce a signed digital certificate. Hence, compromised servers will not affect the digital signature service provided by these servers as a whole because they can only generate at most $t - 1$ partial digital signatures. A combiner also verifies the combination using its public key. Tampered partial signatures from compromised servers will be detected by the combiner.

Constructing partial signatures for CA certification is highly computationally intensive and cannot be performed on mobile devices with inherent resource constraints. To adapt threshold cryptography to MANET, a scheme that combines ID-based cryptography and threshold cryptography has been introduced [13]. An ID-based cryptosystem provides public/private key encryption using node ID to derive the effective public key of each node. An ID-based encryption scheme consists of four algorithms as follows:

- *Setup* takes an input security parameter and returns a master public/private key pair for the system. Every node in the system knows the master public key but not the private key.
- *Encrypt* takes the master public key, the identity of the recipient, and a plaintext message and returns a ciphertext. Note that in normal encryption, the recipient's public key and the plaintext are fed into a cipher.
- *Extract* takes the master private key and an ID (an identity string, such as a MAC address) and produces a personal private key to the identity. Every node must obtain its private key from a private key generation (PKG) service.
- *Decrypt* takes the master public key, a cipher text, and a personal private key and returns the plaintext.

It is obvious that an attacker cannot decrypt an intercepted message without knowing the master private key or personal private key of the node to which the message is headed. The combined key management approach aims at leveraging ID-based public/private key pair generation to reduce computational overhead. It works as follows. First, the initial participating nodes decide on a set of security parameters, such as threshold t, and their identities. Then a threshold PKG is performed by these initial nodes to compute a master public/private key pair in a distributed fashion. The master public key is known to everyone. The personal private key of each node is generated based on the node's identity conforming to t-out-of-n threshold cryptography such that fewer than t nodes cannot recover the master private key. Nodes joining the system later must communicate with at least t nodes serving the PKG to obtain t shares of the personal private keys (not the private keys) and compute the personal private key. Because node ID is commonly available in a message header, this approach does not require any specific public key propagation mechanism, as is the case in the CA approach. Another way to introduce low-overhead asymmetric cryptography to mobile devices is using ECC. To this end, some ECC-based distributed key generation schemes have been proposed, such as that of Boneh–Franklin [14].

8.5.3 Wireless Sensor Network Security

Wireless sensor networks have been used in a number of application scenarios, including wild habitat monitoring, lighting and temperature control of a building, and glacial monitoring. More wireless sensor applications that closely relate to our daily life are on the way. As a consequence, security problems of wireless sensor networks have surfaced in response to concerns that potential data interception or tampering could result in serious damage to a system.

The principle challenge of security in a wireless sensor network is the seriously constrained sensor hardware that cannot facilitate generally used security mechanisms on regular desktop computers. Below is a summary of the hardware capability of a Smart Dust node developed at the University of California, Berkeley (see Table 8.6) [15]. It is worth noting that new wireless sensor modules tend to have significantly improved hardware components as a result of the rapid advancements in wireless sensor technology but still lag behind regular desktop computers and even PDAs.

A wireless sensor is generally expected to operate for years without battery replacement, thus reducing power consumption is always a key design objective. Even if it is possible to incorporate powerful processors and communication capabilities into sensor nodes, their power consumption may exceed what a small battery can support. Consequently, given such a hardware configuration, it would be impractical to use traditional security mechanisms in a wireless sensor network, as they usually require a large amount of memory during operation and impose significant communication and computing overhead on the sensor nodes; for example, asymmetric digital signatures for authentication are too expensive for sensor nodes because they may drain a battery too quickly. One way to provide authentication is to employ symmetric key cryptographic systems between sensor nodes, each sharing a secret key with the central trusted base station. To establish a new key, two nodes use the base station as a trusted third party to set up a secured communication channel.

Table 8.6 Characteristics of Prototype Smart Dust Nodes

CPU	**8-Bit, 4 MHz**
Storage	8-KB instruction flash 512-bytes RAM 512-bytes EEPROM
Communication	916-MHz radio
Bandwidth	10 Kbps
Operating system	TinyOS
Operating system code space	3500 bytes
Available code space	4500 bytes

Another security problem in this domain is secured routing in both static wireless sensor networks and future mobile wireless sensor networks. *Ad hoc* routing protocols such as DSR (Dynamic Source Routing) or AODV (Ad hoc On-Demand Distance Vector) are again unsuitable for wireless sensor networks because of the communication overhead and requirement for state maintenance at each node. In addition, message routing in a wireless sensor network often follows a pattern of many-to-one, meaning that many sensor nodes communicate back to a base station, and, as opposed to routing in *ad hoc* networks of mobile devices, in-networking processing (intermediate nodes processing messages being forwarded) for data aggregation makes secured routing in a wireless sensor network more challenging. Commonly used end-to-end security mechanisms cannot be applied in this case because the contents of messages are subject to modification. Karlof and Wagner [16] compiled a list of attacks on sensor network routing:

- Spoofed, altered, or replayed routing information
- Selective forwarding (a sensor node does not forward messages faithfully)
- Sinkhole attacks (a compromised node spoofs messages to attract traffic from adjacent nodes according to the routing algorithm)
- Sybil attacks (a compromised node pretends to be multiple nodes, thereby confusing routing algorithms and resulting in potential identity theft)
- Wormholes (multiple compromised nodes can collude to establish out-of-band channels, effectively disrupting network topology)
- "Hello" flood attacks (a node simply broadcasts bogus "hello" messages or overheard messages in the network hoping to manipulate topology)
- Acknowledgment spoofing (a node sends spoofed link layer acknowledges to senders of overheard messages)

Among these types of attacks, bogus routing information, Sybil attacks, "hello" floods, and acknowledgment spoofing can be defeated by employing link layer encryption and authentication along with identity verification and authenticated broadcast. Multipath routing can be used to defeat selected forwarding attacks. In order to provide symmetric key cryptography, the network must ensure that no other nodes can impersonate the trusted base station, as each node will obtain a symmetric key from the trusted base station, which also initiates an authenticated broadcast to perform a query. Asymmetric authentication is needed to make sure compromised nodes cannot perform authenticated broadcasts. One way to achieve this is to use delayed disclosure of a series of keys derived from a one-way symmetric key chain [15], which requires the base station and nodes to be loosely synchronized. The base station uses a secret key (K_n) as the last key in the key chain and computes K_{n-1} using a one-way function F: $K_{n-1} = F(K_n)$. Then, it uses $K_1, K_2, \ldots$ during a specific time period subsequent to computing the MAC

(Message Authentication Code) of packets sent within that time slot. Nodes receiving those packets can verify the integrity as well as authenticity of those packets later when the base station discloses keys in the same order as they were used to compute the MAC.

8.6 MOBILE PRIVACY

As in wired networks, security issues in a mobile computing environment are closely related to privacy issues. Generally, the notion of privacy encompasses two types of problems. One is data privacy: the protection of sensitive user information that by all means should be secured during transmission or in storage, such as a credit card number being transmitted over a secure socket layer (SSL) connection, or Social Security numbers stored in a database on disks and tapes. These problems also fall into the mobile security domain, and various security mechanisms to ensure data privacy have already been discussed in this chapter. The second type of privacy issue—namely, privacy services—is primarily concerned with adjustable privacy exposure and enabling mechanisms. The key challenge to this type of security issue is the conflict between more pervasive mobile applications utilizing the sensitive information of users and the need for privacy protection in a computing environment of many such applications. Three approaches have been proposed to offer general privacy-related services in a pervasive mobile computing environment [17]:

- *Increasing awareness of potential privacy breach*—Let the system notify the user whenever sensitive information is being revealed to an external service or system that the user cannot control. For example, a smart phone user should be notified when the user's location and identity are being tracked by a location-based service.
- *Maintaining an audit trail*—The system keeps an audit log of all privacy-related information exposure, interactions, and data exchanges. This does not prevent privacy violation but at least provides some record of what information has been exposed, how, and when.
- *Intelligent alert*—In some cases a user's privacy is exposed not by the system but by an adversary; for example, a smart phone user engaging in a Bluetooth data transmission may be detected by someone nearby and the identity of the user may be revealed because of the data transmission. In this case, ideally the system should be able to detect such a privacy breach even if it is directly involved.

Mobile privacy is more complicated than mobile security because you cannot just draw a line between what information can be used or shared and what cannot, whereas in security we know that a set of security functions should be implemented in a system. Moreover, legislation is very often involved, because privacy is indeed surrounded by sensitive legal issues. Because a system may surreptitiously detect, use, expose, distribute, or abuse people's privacy-related information, it seems quite reasonable to regulate the use of personal data by credit card companies, telecoms, banks, etc. For example, laws pertaining to privacy include the Privacy Act of 1974, Electronic Communications Privacy Act (1986), and No Electronic Theft (NET) Act (1997). Also, in response to security challenges on the web, W3C has been working on a project called P3P (Platform for Privacy Preference Project, http://www.w3.org/P3P), aimed at developing a framework of protocol for an international privacy policy.

Technologically, the fundamental challenge in this domain is to provide more new services to improve productivity and the user's experience while still guaranteeing a minimal, satisfactory level of privacy exposure. Below we introduce two major mechanisms in this field: identity privacy and location privacy.

8.6.1 Identity and Anonymity

Anonymity in the context of computing refers to a service that prevents the disclosure of the identity of someone who is engaged in network communication or interaction to a system. Anonymity is not always a priority; in many cases, we are not particularly concerned that when we surf the web our travels on the Internet are being logged by nearly all web servers. In some cases, however, anonymity is required, such as:

- Users do not want to be censored when accessing some websites. Information censorship is largely due to political reasons.
- Users do not want to reveal information about operations such as file sharing being performed with a computer or a cell phone.
- Users do not want to expose personal information to an untrustworthy online community or they simply do not want to be traced in a network.
- Users want to remain anonymous to prevent identity theft.

Many people think that the Internet offers anonymity. This was reflected by a now-famous cartoon in an issue of *The New Yorker* magazine published in 1993. It showed a dog sitting at a computer, talking to another one: "On the Internet, no one knows you are a dog." Unfortunately, without using a specially designed privacy-enhancing system, nearly every action of a user, as well as the user's identity, is traceable as long as interested parties such as law enforcement agencies, Internet service providers (ISPs), and network administration authorities consider it worth the time and money to do so. In the mobile wireless world, we are well aware that every phone call is logged and can be tapped, and technically every bit can be traced back to the sender. Thus, the challenge of mobile privacy lies in the fact that a mobile system must provide both privacy and accountability.

When there is a direct logical mapping between the user's identity and network location such as IP address, cell phone number, or processor identifier, the goal of anonymity is thus reduced to protecting the network location from being exposed to unintended parties. A simple solution is to introduce a proxy between the user and the place of interest (a website, for example) to hide the network location of the user. An example of such a proxy-based anonymity system is Anonymizer (http://www.anonymizer.com). A user always goes through the proxy (the anonymizer) in order to reach the destination using a URL such as http://www.anonymizer.com:8080/www.yahoo.com. The proxy acts on behalf of the user when visiting a website. Although a user's identity is hidden from the visited server, this approach does not protect the anonymity of the server. To this end, a proxy for the server could be a solution, which hides the real URL of the server and simply exposes a cryptic URL to users. Rewebber (http://www.rewebber.de/) is an example of such a system. In both user–proxy and server–proxy setups, a user has to trust the proxy, and communication between a user and the proxy is not protected in terms of privacy. For this reason, cryptographic mechanisms are introduced in some systems such as Mix-Net and Onion Ring.

A mix network is a set of router nodes (mixes) that allow for anonymous message transfer using a layered public key encryption. They have been used to maintain privacy during e-mailing, web surfing, electronic voting, and electronic payment. The idea of mix networks as a solution to e-mail privacy was first proposed by David Chaum in 1981. In its initial design, a computer (called a *mix*) processes each e-mail (or any type of data item) before it is delivered. A sender may choose intermediate mix nodes to form a path across a mix network, or the mix network can enforce a path for every message. The latter approach is referred to as a *cascade*. Figure 8.10 depicts the logical architecture of a cascade. Depending on the number of mix nodes a message will traverse, a message (M) is encrypted first using the public key of the recipient (K_a) and a random number (R_1). The result is then appended with the address of the recipient and further encrypted using the public key of the last mix node along the path (K_n). Each intermediate mix node decrypts the message using its private key and forwards the result to the next mix node. Output messages at a mix node are also permutated to disguise the order of

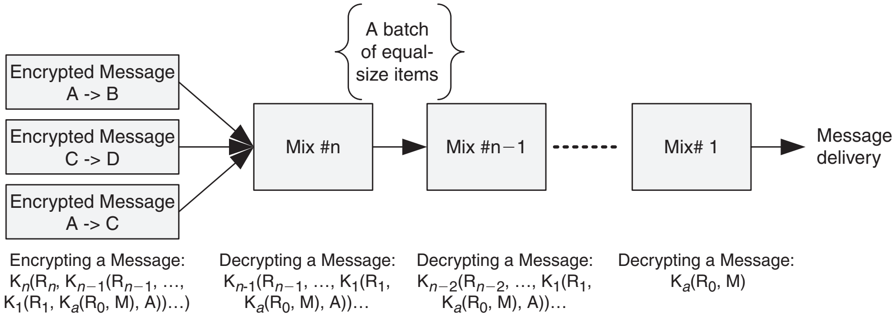

A batch of equal-size items
Encrypted Message A -> B
Encrypted Message C -> D
Encrypted Message A -> C
Mix #n
Mix #n−1
Mix# 1
Message delivery
Encrypting a Message: $K_n(R_n, K_{n-1}(R_{n-1}, \ldots, K_1(R_1, K_a(R_0, M), A))\ldots)$
Decrypting a Message: $K_{n\text{-}1}(R_{n-1}, \ldots, K_1(R_1, K_a(R_0, M), A))\ldots$
Decrypting a Message: $K_{n-2}(R_{n-2}, \ldots, K_1(R_1, K_a(R_0, M), A))\ldots$
Decrypting a Message: $K_a(R_0, M)$

FIGURE 8.10

A cascade mix network.

arrival. In effect, no single mix node knows both the sender address and recipient address. Input messages to a mix node are reordered; therefore, correspondence between items in its input and those in its output at a mix node is protected. The downside of a mix network is that it requires mix nodes to trust each other, meaning that everyone will perform normally. Later improvements to the approach have employed credit-based mix node selection and threshold cryptography to relax this requirement while still ensuring message anonymity.

A similar idea—namely, Onion Routing (http://www.onion-router.net)—is designed to use a collection of widely distributed routers (Tor nodes) to create random paths for the sender such that no individual server knows the complete path. Before sending data over an anonymous path, the first Tor router adds a layer of encryption for each subsequent one in the path. As the message traverses the network, each Tor router removes one layer of encryption.

Unlike Mix-Net and Onion Ring operating at the network layer, Crowds [18] is an application layer protocol designed for web traffic anonymity, utilizing a crowd of proxies to hide the network location of a message. The basic idea is to blend a user's traffic with that of many others such that it is not possible to trace a single web request or reply back to the sender. Any user willing to participate in the crowd could be a proxy in the crowd. The user's traffic will first be forwarded to the crowd along a probabilistic virtual path before going to the public Internet.

Freenet is an example of a peer-to-peer-based anonymity network (http://freenet.sourceforge.net/). It allows anybody to publish and read information with complete anonymity. Freenet achieves this by pooling the nodes' storage for data replication services while the true origin or destination of the data remains completely anonymous. In Freenet, shared files are mapped into a key space. Aside from locally stored files, a node maintains a local key routing table allowing the node to forward a query message from one neighbor (the predecessor) to another appropriate neighboring node on behalf of that predecessor, in case the key in question is not locally served. Note that, unlike IP routing, where the source IP address is always forwarded hop-by-hop as part of the IP header, query routing messages in Freenet do not carry the request's identity along the path. Therefore, requester (a node querying a file) anonymity is preserved because a node forwarding or replying to a query does not know the requester's identity (a node ID in Freenet). In order to maintain the inserter's anonymity (or, more precisely, key anonymity, as a file is identified by a routable key), Freenet employs a variation of the mix network approach for inserter (a node that shares a file in the network) anonymity: Messages between a sender and a recipient must go through a chain of prerouting nodes, each acting as a mix to impose public key-based encryption over links along the chain. After going through the mix network, a message is disguised as if it is originated from the last mix. Then the message is sent to Freenet for normal routing.

In the context of mobile wireless services, identity anonymity is sometimes necessary in mobile payment, mobile trading, and information sharing. Considering the amount of web traffic in current mobile Internet and wireless network applications, an application layer anonymity system is preferable to network layer solutions. For example, we can use a set of WAP proxies acting like a crowd to mix WAP traffic from many mobile users. Alternatively, depending on the requirements of a specific mobile service, a service anonymizer can be introduced as part of the back-end system by a mobile service provider. The predominant task of a service anonymizer in an m-commerce environment is hiding one user's real identity from the other during a transaction. In addition, a service anonymizer can be integrated with an authentication system. An example of such systems used for mobile micropayment is described in Hu et al. [19].

8.6.2 Location Privacy

A particularly significant class of privacy issues is location privacy in a mobile wireless environment. Location privacy refers to the capability of a mobile application or service to prevent unintended parties from obtaining a person's current or past location. The fact that more location-based services, including GPS, Wi-Fi, radiofrequency identification (RFID), and wireless sensor network technologies, will have the capability to monitor a user's location has led to increasing concerns as to how to protect the location information from unintended access. Here, we focus on a subsystem in a location-aware system that enables location privacy.

There are three categories of problems surrounding location privacy for a mobile system, each solving the problem from a different viewpoint:

- *Category I*—Location information security (secure location data gathering and transmission with respect to privacy requirement)
- *Category II*—Identity pseudonym (applying identity anonymity schemes to location service)
- *Category III*—Location information policy (building interactive social and legal privacy aware framework)

The first category, location information security, is mainly concerned with the formatting and secure transmission of location information in order to protect user privacy. The IEEE Work Group Geographic Location/Privacy (Geopriv) [20] has provided a location privacy framework that is independent of the underlying location determination mechanism. The framework defines a location object that conveys location information and possibly privacy rules to which Geopriv security mechanisms and privacy rules are to be applied. Geopriv recommends the use of security mechanisms of the location object itself, such as MAC (Message Authentication Code) and encryption as part of the location object. In addition, secure transport of location objects should be used whenever possible in protocols carrying location objects to ensure appropriate distribution, protection, usage, retention, and storage of location objects based on the rules that apply to those location objects. One example of such a privacy-preserving communication protocol is Mist [21]. This approach is based on an overlay network in the form of a hierarchy of Mist routers that perform limited PKI-secured handle-based routing to hide the location of a connection (here, location is the addresses of the source and the destination). A handle is an ID that uniquely identifies an upward Mist router in the hierarchy. Intermediate Mist routers are unaware of the endpoints of a connection (source and destination addresses). The protocol effectively prevents insiders, system administrators, and the system itself from tracking a user's location without affecting normal secured communication.

The second category, identity pseudonym, hides the user's identity by making network traffic anonymous in a location-based application. A broad range of anonymity techniques used in wired network applications could be adopted to location-based applications. For example, Beresford and Stajano [22] have designed a privacy-protecting framework based on frequently changing pseudonyms, thereby effectively mapping the problem of location privacy onto that of anonymous communication. An anonymizing proxy is introduced to leverage the idea of mix networks in the general anonymity service domain to delay and reorder messages when users exit mix zones.

The last category of solutions, location information policy, focuses on building a framework of privacy policies and mechanisms that allows users to interact with location-based applications to control location information release with respect to corresponding privacy policies. Privacy solutions in this category in essence rely on respect and social and legal norms to enforce privacy. The most notable effort in this direction includes the Privacy Preference Project (P3P) [23] and pawS [24], which provide an industry standard of privacy policies that websites can use to announce their specific privacy practices. The goals of P3P include simplifying the process of reading privacy policies, minimizing latency delays, and making policies conforming to the law. The P3P architecture consists of user agents, privacy reference files, and privacy policies. User agents can be part of a web browser or a browser plug-in. A user agent automatically fetches the P3P policies of a website when the user visits the site and checks these policies against the user's predetermined preferences. A policy reference file is used to collect the P3P policies of certain regions of a website (such as a web page), portions of a website, or the entire website. P3P employs an XML encoding scheme for P3P policies. pawS [24] is a similar approach. Both P3P and pawS are specifically designed to address privacy issues on the Web. A more general approach utilizing the same basic idea has been proposed to protect privacy when arbitrary location-based applications request a user's location [25].

8.7 CONCLUSION

Mobile security and privacy are by all means interrelated issues that must be addressed as a whole. Because of the potentially pervasive nature of future mobile computing applications, people are far more concerned with these issues than common security risks in a wired network environment. A mobile wireless system must take security and privacy into account at the very beginning of the design phase and utilize appropriate security service building blocks to provide data confidentiality, integrity, authentication, and nonrepudiation, as well as efficient access control. Different mobile wireless systems and applications may employ a set of security mechanism at different layers, due to the intrinsic restrictions of the underlying network and mobile devices. In this chapter, we have explored security issues in some widely deployed mobile wireless systems, such as cellular networks, wireless LAN, and Bluetooth. We also introduced interesting yet challenging security issues in emerging mobile *ad hoc* networks. 3G cellular networks by design provide strong low layer security for mobile applications and services. On the other hand, wireless LAN is an excellent example of bad design strategy to demonstrate that security has to be considered a high priority when it comes to designing a mobile wireless system. The well-known WEP vulnerabilities have largely hindered the widespread implementation of 802.11 wireless LAN in business organizations and government agencies. The IEEE 802.11 working group has designed a new standard, called 802.11i, to address these weaknesses. Bluetooth security concerns grew significantly after researchers demonstrated that they could use Bluetooth equipment to hack into a Bluetooth cell phone up to a mile away. Although this particular security problem is merely an implementation issue rather than a serious protocol design issue, some researchers have pointed out several weaknesses in the official Bluetooth specifications that may lead to personal information breaches and device compromise. The Bluetooth specification was largely based on the assumption that within its limited signal range of <10 m security was not a significant problem. This turned out to be a false assumption. Security services in *ad hoc* networks lead to new challenges due to the absence of a fixed network infrastructure in MANET.

Problems such as secured routing, link layer security, and key management were examined. We introduced two problems in the domain of mobile privacy: anonymity and location privacy. Anonymity is a critical problem because people are seeking technological ways to ensure freedom over the Internet. Location privacy is particularly important to mobile users who wish to take advantage of emerging location-based services but do not want to be traced for whatever reasons. Technical, social, and legal solutions have been proposed to address this problem to some extent.

Aside from wireless network security mechanisms, many of the security and privacy problems discussed in this chapter are closely related to requirements of the underlying mobile applications and services such as location-based services, mobile commerce, and instant messaging.

FURTHER READING

3GPP SA3 Security Working Group, http://www.3gpp.org/TB/SA/SA3/SA3.htm (3GPP technical specifications).

3GPP2's Security Working Group (3GPP2 TSG-S Working Group 4), http://www.3gpp2.org/Public_html/specs/tsgs.cfm.

AES/Rijndael, csrc.nist.gov/CryptoToolkit/aes/rijndael/.

EEF DES Cracker, http://www.eff.org/Privacy/Crypto/Crypto_misc/DESCracker/.

For a list of practical ways to protect a Wi-Fi network, see http://www.wi-fi.org/OpenSection/secure.asp? TID = 2 (the site also introduces WPA2, a Wi-Fi certified security solution based on 802.11i).

IEEE AAA Working Group, http://www.ietf.org/html.charters/aaa-charter.html.

IETF Geographic Location/Privacy Working Group, http://www.ietf.org/html.charters/geopriv charter.html; Geopriv Requirement, http://www.ietf.org/rfc/rfc3693.txt.

IETF Internet Key Exchange, http://www.ietf.org/rfc/rfc2409.txt.

IETF IPSec Working Group, http://www.ietf.org/html.charters/ipsec-charter.html.

IETF PKI (X.509) Working Group, http://www.ietf.org/html.charters/pkix-charter.html.

REFERENCES

[1] S. R. Fluhrer, I. Mantin, and A. Shamir, "Weaknesses in the key scheduling algorithm of RC4," in *Proc. of the Eighth Annual Workshop on Selected Areas in Cryptography*, August 2001.

[2] X. Wang, D. Feng, X. Lai, and H. Yu, "Collisions for Hash Functions MD4, MD5, HAVAL-128, and RIPEMD," in *Proc. the 24th Annual International Cryptology Conference (Crypto'04)*, Santa Barbara, CA, 2004.

[3] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM.*, 21(2):120–126, 1978.

[4] M.-J. Saarinen, "Attacks against the WAP WTLS Protocol," in *Proc. Proceedings of the IFIP TC6/TC11 Joint Working Conference on Secure Information Networks: Communications and Multimedia Security*, Leuven, Belgium, 1999.

[5] H. Berghel, "Wireless infidelity I: War driving," *Commun. ACM.*, 47(9):21–26, 2004.

[6] S. Fluhrer, I. Mantin, and A. Shamir, "Weaknesses in the key schedule algorithm of RC4," in *Proc. the 4th Annual Workshop on Selected Areas of Cryptography*, August 2001.

[7] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting mobile communications: The insecurity of 802.11," in *Proc. the 7th Annual International Conference on Mobile Computing and Networking (MOBICOM'01)*, Rome, Italy, 2001.

[8] N. Cam-Winget, R. Housley, D. Wagner, and J. Walker, "Security flaws in 802.11 data link protocols," *Commun. ACM*, 46(5):35–39, 2003.

[9] M. Jakobsson and S. Wetzel, "Security weakness in Bluetooth," in *Proc. RSA Conference 2001*, San Francisco, CA, 2001.

[10] V. Gupta, S. Krishnamurthy, and M. Faloutsos, "Denial of Service Attacks at the MAC Layer in Wireless Ad Hoc Networks," in *Proc. IEEE Military Communications Conference (MILCOM)*, Anaheim, CA, 2002.

[11] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security in mobile *ad hoc* networks: Challenges and solutions," *IEEE Wireless Commun.*, 11(1):38–47, 2004.

[12] L. Zhou and Z. J. Haas, "Securing *ad hoc* networks," *IEEE Network Mag.*, 13(6):24–30, 1999.

[13] A. Khalili, J. Katz, and W. A. Arbaugh, "Toward secure key distribution in truly *ad hoc* networks," in *Proc. 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*, Orlando, FL, January 27–31, 2003.

[14] D. Boheh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM J. Computing.*, 32(3):586–615, 2003.

[15] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar, "SPINS: Security Protocols for Sensor Networks," in *Proc. the 7th Annual International Conference on Mobile Computing and Networking (MOBICOM'01)*, Rome, Italy, 2001.

[16] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," in *Proc. of the First IEEE International Workshop on Sensor Network Protocols and Applications*, Anchorage, AK, May 2003.

[17] M. Satyanarayanan, "Privacy: The Achilles heels of pervasive computing?," *IEEE Pervasive Comput.*, 2(1):2–3, 2003.

[18] M. K. Reiter and A. D. Rubin, "Anonymous web transactions with crowds," *Commun. ACM.*, 42(2):32–48, 1999.

[19] Z.-Y. Hu, Y.-W. Liu, X. Hu, and J.-H. Li, "Anonymous Micro-payments Authentication (AMA) in Mobile Data Networks," in *Proc. the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, Hong Kong, 2004.

[20] IEEE Geographic Location/Privacy, http://www.ietf.org/html.charters/geopriv-charter.html, 2004.

[21] J. Al-Muhtadi, R. Campbell, A. Kapadia, M. D. Mickunas, and S. Yi, "Routing through the mist: Privacy preserving communication in ubiquitous computing environment," in *Proc. of the 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, Vienna, Austria, July 2002.

[22] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Comput.*, 2(1):46–55, 2003.

[23] Platform for Privacy Preference 1.0 (P3P1.0) specification, World Wide Web Consortium, http://www.w3.org/TR/2002/REC-P3P-20020416/, 2002.

[24] M. Langheinrich, "Privacy by design: Principles of privacy-aware ubiquitous systems," in G. D. Abowd, B. Brumitt, and S. A. Shafer (Eds.), *Proc. of the 3rd International Conference on Ubiquitous Computing*, Atlanta, GA: Springer–Verlag, 2001.

[25] G. Myles, A. Friday, and N. Davies, "Preserving privacy in environments with location-based applications," *IEEE Pervasive Comput.*, 2(1):56–64, 2003.