5.2 Modern Encryption Methods

Modern methods of encryption are more secure than the historical methods discussed in the previous section. All the methods discussed in this section are in use today and are considered reasonably secure. In some cases, the algorithm behind these methods requires a sophisticated understanding of mathematics.

Number theory often forms the basis for encryption algorithms. Fortunately, for our purposes, having the exact details of these encryption algorithms is not important; this means that you do not require a strong mathematical background to follow this material. It is more important to have a general understanding of how a particular encryption method works and how secure it is.

5.2.1 Symmetric Encryption

Symmetric encryption refers to the methods where the same key is used to encrypt and decrypt the plaintext.

5.2.1.1 Binary Operations

Part of modern symmetric cryptography ciphers involves using binary operations. Various operations on binary numbers (numbers made of only zeroes and ones) are well known to programmers and programming students. However, for those readers that are not familiar with them, a brief explanation follows. When working with binary numbers, three of the operations are not found in normal math: AND, OR, and XOR operations. Each is illustrated next.

AND

To perform the AND operation, you take two binary numbers and compare them. One place at a time. If both numbers have a “**one”** in both places, then the resultant number is a “**one”**. If not, then the resultant number is a “**zero”**, as you see below:

1 1 0 1

1 0 0 1

---

1 0 0 1

OR

The OR operation checks to see whether there is a “**one”** in either or both numbers in a given place. If so, then the resultant number is “**one”**. If not, the resultant number is “**zero”**, as you see here:

1 1 0 1

1 0 0 1

----

1 1 0 1

XOR

The XOR operation impacts your study of encryption the most. It checks to see whether there is a “**one”** in a number of a given place, but not in both numbers at that place. However, it cannot perform a check in both numbers at that place. If it is in one number but not the other, then the resultant number is “**one”**. If not, the resultant number is “**zero”**, as you see here:

1 1 0 1

1 0 0 1

----

0 1 0 0

XORing has an interesting property, it is reversible. If you XOR the resultant number with the second number, you get back the first number. In addition, if you XOR the resultant number with the first number, you get the second number.

0 1 0 0

1 0 0 1

----

1 1 0 1

Binary encryption using the XOR operation opens the door towards some rather simple encryption. Take any message and convert it to binary numbers and then XOR that with some key. Converting a message to a binary number is a simple two-step process. First, convert a message to its ASCII code, and then convert those codes into binary numbers.

Each letter/number will generate an eight-bit binary number. You can then use a random string of binary numbers of any given length as the key. Simply XOR your message with the key to get the encrypted text, and then XOR it with the key again to retrieve the original message.

This method is easy to use and great for computer science students; however, it does not work well for truly secure communications because the underlying letter and word frequency remains. This exposes valuable clues that even an amateur cryptographer can use to decrypt the message. Yet, it does provide a valuable introduction to the concept of single-key encryption.

Although simply XORing the text is not the method that is typically employed, single-key encryption methods are widely used today. For example, you could simply include a multi-alphabet substitution. This is then XORed with some random bit stream—variations of which do exist in some encryption methods that are currently used today.

Modern cryptography methods, as well as computers, make decryption a rather advanced science. Therefore, encryption must be equally sophisticated in order to have a chance of success.

5.2.1.2 Data Encryption Standard

Data Encryption Standard, or DES as it is often called, was developed by IBM in the early 1970s and made public in 1976. DES uses a symmetric key system, which means the same key is used to encrypt and to decrypt the message. DES uses short keys and relies on complex procedures to protect its information. The actual DES algorithm is quite complex. The basic concept, however, is as follows:

1. The data is divided into 64-bit blocks, and those blocks are then reordered.

2. Reordered data are then manipulated by 16 separate rounds of encryption, involving substitutions, bit-shifting, and logical operations using a 56-bit key.

3. Finally, the data are reordered one last time.

DES uses a 56-bit cipher key applied to a 64-bit block. There is actually a 64-bit key, but one bit of every byte is actually used for error detection, leaving just 56 bits for actual key operations. The problem with DES is the same problem that all symmetric key algorithms have: How do you transmit the key without it becoming

compromised? This issue led to the development of public key encryption.

5.2.1.3 Blowfish

Blowfish is a symmetric block cipher. This means that it uses a single key to both encrypt and decrypt the message and works on “blocks” of the message at a time. It uses a variable-length key ranging from 32 to 448 bits. This flexibility in key size allows you to use it in various situations. Blowfish was designed in 1993 by Bruce Schneier. It has been analysed extensively by the cryptography community and has gained wide acceptance. It is also a non-commercial (that is, free of charge) product, thus making it attractive to budget-conscious organisations.

5.2.1.4 AES (Advanced Encryption Standard)

Advanced Encryption Standard (AES) uses the Rijndael algorithm. The developers of this algorithm have suggested multiple alternative pronunciations for the name, including “reign dahl,” “rain doll,” and “rhine dahl.” This algorithm was developed by two Belgian researchers, Joan Daemen of Proton World International and Vincent Rijmen, a postdoctoral researcher in the Electrical Engineering Department of Katholieke Universiteit Leuven.

AES specifies three key sizes: 128, 192, and 256 bits. By comparison, DES keys are 56 bits long, and Blowfish allows varying lengths up to 448 bits. AES uses a block cipher. This algorithm is widely used, considered very secure, and therefore a good choice for many encryption scenarios.

5.2.2 Public Key Encryption

Public key encryption is essentially the opposite of single-key encryption. With any public key encryption algorithm, one key is used to encrypt a message (called the public key) and another is used to decrypt the message (the private key). You can freely distribute your public key so that anyone can encrypt a message to send to you. However, only you have the private key and only you can decrypt the message. The actual mathematics behind the creation and applications of the keys can be quite complex and beyond the scope of this book. Many public key algorithms are dependent to some extent on large prime numbers, factoring and number theory.

5.2.2.1 RSA

The RSA method is a widely used encryption algorithm. RSA must be discussed whilst conversing about cryptography. This public key method was developed in 1977 by three mathematicians: Ron Rivest, Adi Shamir, and Len Adleman. The name RSA is derived from the first letter of each mathematician's last name.

One significant advantage of RSA is that it is a public key encryption method. This means there are no concerns with distributing the keys for the encryption. However, RSA is much slower than symmetric ciphers. In fact, asymmetric ciphers are slower than symmetric ciphers generally.

The steps to create the key are as follows:

1. Generate two large random primes, **p** and **q**, of approximately equal size.

2. Pick two numbers so that when they are multiplied together the product will be the size you want (that is, 2048 bits, 4096 bits, etc.).

3. Now multiply **p** and **q** to get **n**.

4. Let **n = pq**.

5. Multiply Euler's totient for each of these primes. If you are not familiar with this concept, the Euler's Totient is the total number of co-prime numbers. Two numbers are considered co-prime if they have no common factors.

For example, if the original number is 7, then 5 and 7 would be co-prime. It just so happens that for prime numbers, minus 1 is always the number. For example, 7 has 6 numbers that are co-prime to it (if you think about this a bit you will see that 1, 2, 3, 4, 5, 6 are all co-prime with 7).

6. Let **m = (p – 1)(q – 1).**

7. Select another number; call this number **e**. You want to pick **e** so that it is co-prime to **m**.

8. Find a number **d** that when multiplied by **e** and modulo **m** would yield 1. (Note: Modulo means to divide two numbers and return the remainder. For example, 8 modulo 3 would be 2.)

9. Find **d**, such that **de mod m ≡ 1**.

Now you publish **e** and **n** as the public key and keep **d** and **n** as the secret key. To encrypt you simply take your message raised to the **e** power and modulo **n = Me % n**

To decrypt you take the ciphertext, and raise it to the **d** power modulo **n**:

P = Cd % n

RSA has become a popular encryption method. It is considered quite secure and is often used in situations where a high level of security is needed.

5.2.2.2 Elliptic Curve

The Elliptic Curve algorithm was first described in 1985 by Victor Miller (IBM) and Neil Koblitz (University of Washington).

The security of Elliptic Curve cryptography is based on the fact that finding the discrete logarithm of a random elliptic curve element, is quite difficult to achieve within a public base point. It is also seen as impractical due to its difficulty.

The size of the elliptic curve determines the difficulty of finding the algorithm, and thus the security of the implementation. The level of security afforded by an RSA-based system with a large modulus can be achieved with a much smaller elliptic curve group. There are actually several ECC algorithms. There is an ECC version of Diffie-Hellman, an ECC version of DSA, and many others.

The U.S. National Security Agency has endorsed ECC (Elliptic Curve Cryptography) by including schemes based on it inside Suite B’s set of recommended algorithms. This allows their use for protecting information classified up to top secret with 384-bit keys.

5.2.3 Digital Signatures and Certificates

A digital signature is not used to ensure the confidentiality of a message, but rather to guarantee who sent the message. This is referred to as non-repudiation. Essentially, a digital signature proves who the sender is. Digital signatures are actually rather simple, but clever. They simply reverse the asymmetric encryption process.

Recall that in asymmetric encryption, the public key (which anyone can have access to) is used to encrypt a message to the recipient, and the private key (which is kept secure, and private) can decrypt it. With

a digital signature, the sender encrypts something with his or her private key. If the recipient is able to decrypt that with the sender's public key, then it must have been sent by the person purported to have sent the message.

5.2.3.1 Digital Certificates

Public keys are widely distributed. Obtaining someone's public key is fairly easy to do and is also needed in order to verify a digital signature. As to how public keys are distributed, probably the most common way is through digital certificates. The digital certificate contains a public key and some means to verify whose public key it is.

X.509 is an international standard for the format and information contained in a digital certificate. X.509 is the most used type of digital certificate in the world. It is a digital document that contains a public key signed by a trusted third party, which is known as a certificate authority (CA). The contents of an X.509 certificate are:

- Version
- Certificate holder's public key
- Serial number
- Certificate holder's distinguished name
- Certificate's validity period
- Unique name of certificate issuer
- Digital signature of issuer
- Signature algorithm identifier

A certificate authority issues digital certificates. The primary role of the CA is to digitally sign and publish the public key bound to a given user. It is an entity trusted by one or more users to manage certificates.

A registration authority (RA) is used to take the burden off a CA. This is done by handling verification prior to certificates being issued. RAs act as a proxy between users and CAs. RAs receive a request, authenticate it, and forward it to the CA.

A public key infrastructure (PKI) distributes digital certificates. This network of trusted CA servers serves as the infrastructure for distributing digital certificates that contain public keys. A PKI is an arrangement that binds public keys with respective user identities by means of a CA.

What if a certificate is expired, or revoked? A certificate revocation list (CRL) is a list of certificates that have been revoked for one reason or another. Certificate authorities publish their own certificate revocation lists. A newer method for verifying certificates is Online Certificate Status Protocol (OCSP), a real-time protocol for verifying certificates.

There are several different types of X.509 certificates. They all possess the elements listed at the beginning of this section, but serve different purposes. The most common certificate types are listed below:

- Domain validation certificates are among the most common. These are used to secure communication with a specific domain. This is a low-cost certificate that website administrators use to provide TLS for a given domain.
- Wildcard certificates, as the name suggests, can be used more widely, usually with multiple sub-domains of a given domain. So, rather than having a different X.509 certificate for each sub-domain, you would use a wildcard certificate for all sub-domains.
- Code-signing certificates are X.509 certificates used to digitally sign some type of computer code. These usually require more validation from the person requesting the certificate, before they can be issued.
- Machine/computer certificates are X.509 certificates assigned to a specific machine. These are often used in authentication protocols. For example, in order for the machine to sign into the network, it must authenticate using its machine certificate.
- User certificates are used for individual users. Like machine/computer certificates, these are often used for authentication. The user must present his or her certificate to authenticate before accessing its resource.
- E-mail certificates are used for securing e-mail. Secure Multipurpose Internet Mail Extensions (S/MIME) uses X.509 certificates to secure e-mail communications.
- A Subject Alternative Name (SAN) is not so much a type of certificate as a special field in X.509. It allows you to specify additional items which are protected by this single certificate. These could be additional domains or IP addresses.
- Root certificates are used for root authorities. These are usually self-signed by that authority.

5.2.3.2 PGP Certificates

Pretty Good Privacy (PGP) is not a specific encryption algorithm, but rather a system. It offers digital signatures, asymmetric encryption, and symmetric encryption. It is often found in e-mail clients. PGP was introduced in the early 1990s, and it’s considered to be a very good system.

PGP uses its own certificate format. The main difference, however, is that PGP certificates are self-generated. They are not generated by any certificate authority.