5.2 Public-Key Cryptography

In the cryptosystems we have considered until now, the sender and the receiver select a key K for the enciphering algorithm E_K. This determines the deciphering algorithm D_K. Since the same key is used for both operations,

the two parties must take great care in exchanging the key so that an eavesdropper does not obtain it. The problem of key exchange is one of the most difficult in classical cryptosystems. It is not possible to use the phone or mail system to exchange keys, as these channels are insecure. Couriers are usually the most secure and reliable channel for exchanging keys. For military and diplomatic communications, it is possible to use couriers, but it is expensive and impractical in a commercial setting. In a commercial situation, business transactions are conducted between two or more parties who may not know each other, and who may never do business together again, but still desire complete confidentiality and instant secure communication.

There are additional key management problems in classical cryptosystems. Consider a network of users who wish to communicate with each other, for example, a network of banks or a group of people exchanging e-mail. For complete security, each pair of users must be assigned a different key. For example, with 10 users, we need 45 different keys, and with 100 users, we would need 4950 keys. It is not easy to add a new user to this system, as each user's database of keys needs to be updated to include the new user. Moreover, the large number of keys creates security problems because each user has to safeguard his or her key database.

Classical cryptosystems create problems of trust, as the same key is held by both users in any sender/receiver pair. Suppose Alice and Bob share a secret key. If Carol obtains the key, then she can impersonate Bob and send messages to Alice, who has no way of knowing that the messages she is receiving are not from Bob. Carol can also intercept messages from Alice to Bob and modify them in transit, without Bob ever realizing it. It is also possible for Bob to forge messages, that is, to send a message and later deny that he ever sent the message and claim that his keys were stolen. The problem of forgery is not serious in military/diplomatic settings, where the parties are more likely to have established trust, but it is one of the primary concerns in commercial situations. A bank's customer should not be able to deny making a transaction, but at the same time, the customer should be secure in the belief that the bank is not forging transactions.

We require that a cryptosystem satisfy the following requirements, in addition to being cryptanalytically secure.

Authentication: The recipient of a message should be able to ascertain its origin.

Integrity: The recipient of a message should be able to determine that the message has not been modified in transit.

Non-Repudiability: A sender of a message should not be able to deny later that he sent the message.

These requirements are impossible to achieve in classical one-key cryptosystems. The breakthrough came in 1976 with the invention of Public-Key

Cryptography by Diffie and Hellman, and independently by Merkle. Public-key cryptography was invented to solve the problem of secret-key management. In this system there is no need to exchange keys. Each person uses two keys, an encrypting key and a decrypting key. The encrypting key is published (hence the name public-key) while the decrypting key is kept secret. Anyone wishing to send a message to a person uses that person's public key to encrypt a message. The point here is that the knowledge of the encrypting method does not allow decryption, ensuring that no exchange of secret keys is necessary. One can send confidential information using the published keys, confident that only the intended recipient will be able to decipher it.

It may help the reader to consider an analogy with the mail system. We can send mail to anyone knowing only their address, and once it is in the recipient's mailbox, only the recipient can read it (assuming that the mailbox can only be opened with the recipient's key). The public-key system is stronger in the sense that each message travels in its own safe box, which can only be opened by the legitimate owner. Even the sender cannot open the box to disclose its contents.

In a public-key cryptosystem, each user has two keys. This greatly reduces the number of keys needed to establish a communication network. Each user has a public key, which gives an encrypting function E, and a private key, which gives a decrypting function D. For the scheme to work correctly, we must have

$$D(E(m)) = m \text{ for any message } m.$$

The encrypting function E is known to everyone. Since E is known, an opponent can generate any amount of ciphertext. Hence any such system should be immune to a ciphertext-based attack in which the corresponding plaintext is also known. For the system to be practical, one should be able to compute $E(m)$ and $D(E(m))$ quickly, but for security, it should be impossible to determine D from E in any reasonable amount of time. It is easier to describe the requirements of a public-key system, than to construct the functions E and D, which meet these requirements.

Theoretically, a public-key system can be constructed by a special case of a **one-way function** known as a **trapdoor one-way function**. Intuitively, a one-way function f should be one-to-one, and $f(x)$ easy to compute for any input x, but f^{-1} should be hard to compute. That means that it should be difficult to solve the equation $f(x) = y$ when y is known. A trapdoor one-way function is one for which the equation $f(x) = y$ becomes easy to solve if additional information (the trapdoor) is available. Instead of giving a precise definition of what is meant by *easy to compute* and *hard to compute*, we consider a couple of examples. We should remark that no one has proved the existence of a one-way function. The following two functions are considered the most likely candidates for being one-way functions.

Example 5.2.1. Fix two integers g and n and consider $f(x) = g^x \bmod n$.

The function f is considered easy to compute because the number of steps needed to evaluate f is a polynomial function of the number of digits in the input x, g, and n. The exponential can be computed in at most $\log_2(x)$ squaring operations. The square y^2 of a number y can be computed in $(\log_2 y)^2$ steps. In any case, the number of steps is a polynomial function of the input size, the number of digits. The problem of determining x from the knowledge of $g^x \bmod n$ and g and n is known as the **discrete-log problem**. Despite considerable research, no method is known to evaluate the general discrete-log problem in polynomial time in the number of digits. All known algorithms require time that is polynomial in the input $g^x \bmod n$, not polynomial in the input size. To settle this question, one needs to show that the discrete-log problem is difficult or find an algorithm to compute it in polynomial time in the input size.

Example 5.2.2. Consider the function $f(p, q) = pq$, where p and q are prime numbers. The function is easy to compute because multiplication takes time proportional to the product of the number of digits of p and q, but computing the inverse of f is an extremely difficult problem. There is no efficient method to determine p and q from $n = pq$. The method of factoring by trial division takes time proportional to $\sqrt{p}$, if p is the smaller prime. This is exponential in the number of digits. While trial division is rather crude and cannot factor numbers with more than 18 to 20 digits, modern factoring methods still cannot factor numbers with more than 120 digits. If p and q have 100 digits each, then for practical purposes, this is a one-way function. Again, there is no proof that it is a true one-way function.

The first application of one-way functions seems to have been to the protection of passwords of users in a multi-user computer system. A user needs a password known only to her to be able to access the system. If the computer stores the passwords in a file, then the password is susceptible to attack. The solution is to apply a one-way function to the password and store the result in the password file. To authenticate a user, the computer applies the one-way function to the typed password and compares it to the entry in the password file. If the two are the same, then the user is allowed access. The advantage of this system is that the actual passwords are not stored, and if the function used is one-way, then anyone seeing the password entries will not be able to determine the actual password.

Unlike the previous example, for encryption and decryption, a one-way function is not directly useful. What is needed is a special type of one-way function, the trapdoor one-way function. In Example 5.2.2, if $n = pq$ is known, then p and q can be determined if additional information is available, for example, $\phi(n)$ or $\sigma(n)$. This additional information is the trapdoor. If these functions are used in cryptography, the trapdoor is needed to determine the decrypting algorithm D from the knowledge of E.