Public key cryptography

A. M. Odlyzko
AT&T Bell Laboratories
Murray Hill, New Jersey 07974
amo@research.att.com

ABSTRACT

Public key cryptography is an important development of the last two decades. It is exciting on a purely intellectual level, as it provides capabilities that at first glance might seem impossible. For example, it enables two people to determine who earns more without allowing either to learn the other's salary. On a more serious level, public key cryptography solves several important problems, especially those of key management and digital signatures, that are vital for information processing. This article explains what public key cryptography is, and what its benefits and limitations are.

Public key cryptography

A. M. Odlyzko
AT&T Bell Laboratories
Murray Hill, New Jersey 07974
amo@research.att.com

1. Introduction

Public key cryptography was invented to provide information security for civilian systems more easily than was possible with traditional methods. Conventional cryptosystems can provide security but at substantial cost. An extreme example is that of the Vernam cipher, invented at AT&T in 1917. If Alice and Bob wish to communicate secretly, they agree ahead of time on a string of bits $x_1, x_2, \ldots$. If Alice wishes to convey to Bob a message that is represented by bits $m_1, m_2, \ldots, m_k$, she transmits to him the bits $c_1 = m_1 \oplus x_1$, $c_2 = m_2 \oplus x_2, \ldots, c_k = m_k \oplus x_k$, where $x \oplus y$ is the exclusive-or of bits x and y. Bob, who receives $c_1, \ldots, c_k$, recovers the original message $m_1, \ldots, m_k$ by the operation $m_j = c_j \oplus x_j$. If the x_j are truly random, and are never used more than once (so that the second message from Alice to Bob, or Bob's reply to the first message from Alice, would use bits $x_{k+1}, x_{k+2}, \ldots$) then this cipher (the "one-time pad") is unbreakable, the only cryptosystem that has been proved to be unbreakable. The problem with the one-time pad is that it requires huge numbers of the random bits x_j, one for each bit that Alice and Bob might wish to transmit. Moreover, those bits have to be created and conveyed securely to just Alice and Bob, without allowing anyone else to learn what they are. This can be done in cases requiring extreme security (the Washington-Moscow hot-line is reputedly encrypted with the one-time pad), but is not adequate for the civilian marketplace, where the volume of transmitted information is huge.

There are conventional cryptosystems that do not need large numbers of random key bits. The best known and most widely used is the U. S. Data Encryption Standard (DES). DES uses a 56-bit key. Therefore if Alice and Bob wish to communicate using DES, they do not need to generate beforehand as many random bits x_j as they feel they might need to transmit. Instead, they only need to agree on a 56-bit DES key. When DES was first proposed as a standard, there were suspicions that it might contain trapdoors that would enable government agencies to decrypt transmissions easily. These concerns have been allayed by research done over the last two decades, and the general concensus is that DES is a strong system for its key size. However, it is felt that a 56-bit key is in general too short. Advances in technology have made exhaustive key search of the 2^{56} possible keys feasible, so that for about $1,000,000 one can build a parallel machine that would typically require only four hours to find a key. For many applications this level of security is not adequate, especially since key search machines are becoming faster and cheaper to build. Exhaustive key attacks can be thwarted by using ciphers stronger than DES. One such system is triple-DES, which consists of three encryptions with the basic DES, controlled by two keys, for an effective key size of 112 bits ([Sch, Sim]).

DES, triple-DES, and other similar systems provide levels of security that can be estimated by skilled cryptographers. The key sizes are moderate, so that the main disadvantage of the Vernam cipher is avoided. However, there is still the key distribution problem. For Alice and Bob to communicate using DES, say, they need to have a 56-bit key that nobody else knows. With n people or computers that might need to communicate, the number of keys that are necessary is $n(n-1)/2$. Since there are already over 20 million users of the Internet, to allow any two to communicate in secret would require 200 trillion keys, and each user would have to keep a file of 20 million keys, one for each potential correspondent. This is a major defect of the conventional cryptosystems, and was the main motivator for the invention of public key cryptography.

The area where conventional cryptosystems are most deficient is digital signatures. While key management can often be handled using classical cryptographic methods (as will be explained in Section 3), there is no effective way to authenticate digital documents, which can be copied freely, without using public key methods. With the rapid spread of electronic transactions of all sorts, this is a serious problem.

2. Public key cryptosystems

Public key cryptography was invented in the 1970s by W. Diffie, R. Merkle, and M. Hellman at Stanford. The first practical public key system was the Diffie-Hellman key exchange system, presented in Sidebar 1. If Alice and Bob wish to communicate in secret, they can use the Diffie-Hellman technique to establish a secret key through an exchange of public messages. This secret key can then be used to encrypt the conversation using a conventional cryptosystem such as DES. If suitable precautions are observed, an eavesdropper who knows the entire exchange of messages will not be able to recover the key and will thus not be capable of intercepting the communication. The Diffie-Hellman scheme is perhaps the most commonly used public key system, and is incorporated in the AT&T Telephone Security Device (along with some special enhancements).

The most famous public key cryptosystem is the RSA algorithm, invented by R. Rivest, A. Shamir, and L. Adleman at MIT shortly after the Diffie-Hellman method was announced. It is described in Sidebar 2. It enables people who have not had a chance to establish a common secret key to communicate privately. RSA also provides numerous other capabilities. Key exchange is simple to implement with RSA. If Alice and Bob wish to establish a secret key for use with DES or other conventional cryptosystems, Alice can simply select a secret key and send it to Bob encrypted with Bob's public key. Perhaps most important is the digital signature capability of RSA, illustrated in Sidebar 3.

There are many other public key cryptosystems. For example, there are digital signature schemes that are based on discrete logarithms (as in the Diffie-Hellman scheme), and not on RSA. The proposed U.S. Digital Signature Standard (DSS) is of this type. There are also various systems with additional capabilities. There exist so-called identity-based cryptosystems in which users obtain certificates from a central authority that encode their basic identification information, limits of validity, and so on, and which enable any two participants in the system to generate a common secret key while simultaneously verifying each other's identity without having to access any database of public information. For more detailed information in this area, see [Pom, Sch, Sim].

3. Limitations of public key cryptosystems

Public key cryptosystems are already widely used, and are likely to become even more widespread. However, they do have limitations that prevent them from being used as universally as their earliest proponents expected. The primary limitation is that of the computational burden they impose. Almost all the public key cryptosystems that are regarded as secure are based on number theoretic techniques that involve multiplication of large integers. Intensive research over the last two decades has increased the sizes of the numbers that are needed to provide security. For example, Rivest, Shamir, and Adleman published a challenge in 1977 using a version of the RSA system that relied on 129-digit integers. At that time they fully expected this problem to remain unbroken at least until the end of this century. However, a large distributed computation involving idle time on hundreds of computers around the world and improved algorithms succeeded recently in solving this challenge problem. For information on the methods used in these and related attacks and guidelines on recommended sizes of keys, see [Od]. The computational requirements of public key cryptography are not as much of a barrier to its use as was the case a decade ago, when special hardware appeared necessary. (AT&T even produced a special modular multiplication chip for this purpose.) Today's microprocessors are fast enough to carry out the necessary computations. Still, conventional cryptosystems are usually 10 to 1000 times faster than public key ones. Therefore encryption of messages is invariably done using conventional cryptosystems. Public key schemes are used only for the special tasks where their unique capabilities are needed, such as key exchange, authentication, and digital signatures.

Another reason public key schemes are not used more widely is that many of their capabilities can be obtained from conventional cryptosystems. For example, the Introduction explained the key management problem; if there are n users in a system, then $n(n-1)/2$ keys are needed to allow any two to communicate, and every user has to store $n-1$ keys. If public key systems are used, then only n keys are needed, as only a single key for each user has to be stored, and this key does not have to be safeguarded, as it can (and should) be placed in a publicly accessible database (which has to be secure against unauthorized modifications, though). (An identity-based cryptosystem can sometimes even eliminate the need for this database.) However, in many situations an almost equally satisfactory solution can be constructed with conventional cryptosystems. If there is a trusted center in the system, then each user only needs a single secret key that is shared with the center. If Alice wishes to communicate with Bob, she can send a message to the center, encrypted with the key she and the center share, requesting that a key be generated for the Alice-Bob conversation. The center creates such a key and sends it to Alice (encrypted with the key the center and Alice share) and to Bob (this time encrypted with the key that Bob and the center share). Afterwards Alice and Bob can communicate using the key that the center provided. They obtain not only privacy of the communication, but also assurance of each other's authenticity. The disadvantage of this approach is that the center has to be reachable at all times, and has to be trustworthy (since it possesses the means to listen in on all conversations in the system). Public key cryptography provides ways to solve these problems. However, these drawbacks of conventional cryptosystems are often perceived as not very significant, or else worth the advantage of not having to implement more cumbersome public key schemes. For example, the IS-54 authentication system for North American digital cellular systems (which has since been adapted for other wireless schemes as well), which was designed by Jim Reeds and Phil Treventi of AT&T Bell Laboratories, uses a hierarchy of shared secrets and challenge-response techniques to verify the identity of mobile units without the use of public key cryptosystems. A public key system would have lowered the need for efficient communication between different cellular operators, and would have prevented some denials of service (which might occur as a result of communication overloads or breakdowns). However, since there are relatively few cellular operators, and the Reeds-Treventi system requires only occasional communication between them, the decision made by the industry standards group was not to use public key cryptography in the immediate future. Public key cryptosystems are under consideration for other wireless systems.

Another reason why public key cryptosystems are not used more widely involves patent licensing issues. Most of the basic public key algorithms are patented, and several of the key ones are controlled by a private company, Public Key Partners. Many corporations, including AT&T, Apple, Lotus, and Microsoft, have licenses to these patents. However, there are still many unresolved issues, especially those concerning the proposed U.S. federal Digital Signature Standard (DSS). The U.S. government has publicly stated that it is committed to a no-fee access policy to this standard, but no agreement with Public Key Partners (which claims that DSS infringes on its patents) has been reached as yet.

4. Conclusions

Public key cryptosystems are valuable security tools. They offer essentially the only way to provide digital signatures, and are often the preferred method for authentication or key distribution. However, they should not be used for encryption of general traffic, and can often be dispensed with in networks that have a trusted central authority.

Acknowledgements. The author thanks Joan Feigenbaum, David Maher, and Michael Reiter for their comments.

References

[Od] A. M. Odlyzko, Discrete logarithms and smooth polynomials, in *Finite Fields: Theory, Applications and Algorithms*, G. L. Mullen and P. Shine, eds., Amer. Math. Soc., 1994, in press.

[Pom] C. Pomerance, ed., *Cryptology and Computational Number Theory*, Amer. Math. Soc., Proc. Symp. Appl. Math. #42, 1990.

[Sch] B. Schneier, *Applied Cryptography*, Wiley, 1994.

[Sim] G. Simmons, ed., *Contemporary Cryptology*, IEEE Press, 1991.

Sidebar 1: The Diffie-Hellman key exchange method

Suppose that Alice and Bob wish to establish a secret key that only they will possess. To do this, they agree on a large prime p and an integer g. (These numbers p and g might be the same for many people. We will say more about their choice below.) Then Alice chooses a random integer a, $1 \leq a \leq p-2$, and Bob chooses a random integer b, $1 \leq b \leq p-2$. The integers a and b are kept secret. Alice computes A,

$$A \equiv g^a \pmod{p}, \quad 1 \leq A \leq p-1 \ ,$$

and Bob computes B,

$$B \equiv g^b \pmod{p}, \quad 1 \leq B \leq p-1 \ ,$$

where $x \equiv y \pmod{p}$ means that the remainders obtained by dividing x and y by p are the same. The computations of A and B can be carried out fast with no intermediate results larger than p^2. Then Alice transmits A to Bob over an open channel, and Bob transmits B to Alice. Next, Alice computes

$$X \equiv B^a \pmod{p}, \quad 1 \leq X \leq p-1 \ ,$$

and Bob computes

$$Y \equiv A^b \pmod{p}, \quad 1 \leq Y \leq p-1 \ .$$

Since $B \equiv g^b \pmod{p}$ and $A \equiv g^a \pmod{p}$, we find that $X \equiv g^{ba} \equiv g^{ab} \equiv Y \pmod{p}$, and therefore $X = Y$. Hence Alice and Bob do obtain the same integer $X = Y$, which can then be used to derive a key for a conventional cryptosystem.

An eavesdropper, Eve, who listens to the conversation between Alice and Bob, sees A and B. However, to derive X from A and B, it appears (although this has never been proved) that Eve has to compute either a or b. Computing either of these numbers is an instance of the discrete logarithm problem, and appears very hard in general. Some precautions have to be observed (the prime p has to be large, $p-1$ has to have at least one large prime factor, the multiplicative order of g modulo p has to be large, ...). This protocol is widely used.

Sidebar 2: The RSA cryptosystem

The RSA cryptosystem relies for its security on the difficulty of factoring an integer into primes. If Alice wishes to allow secret messages to be sent to her, she chooses two large primes p and q, and forms $n = p \cdot q$. She then selects a random integer e, $1 < e < n$, such that e has no integer divisors > 1 in common with $p - 1$ and $q - 1$. She then publishes the pair (n, e) as her public key, but keeps p and q secret. To send a message to Alice, Bob transforms it into blocks of integers, each of $\leq \log_2 n$ bits. If a particular block is regarded as the binary representation of an integer m, $0 \leq m < n$, then Bob computes

$$c \equiv m^e \pmod{n}, \quad 0 \leq c < n ,$$

using the same consecutive squaring method as in the Diffie-Hellman method, and transmits c to Alice.

To decrypt the transmitted message c, Alice uses a procedure similar to the encrypting one, namely

$$m \equiv c^d \pmod{n} ,$$

where d is her secret decryption exponent. If the factors p and q are known, d can be computed easily from e, since we need $ed \equiv 1 \pmod{p-1}$ and $ed \equiv 1 \pmod{q-1}$.

There is no known way to break the RSA system without finding the prime factors p and q of n.

Sidebar 3: Digital signatures

There are methods for creating digital signatures using conventional cryptosystems, but they are clumsy. In contrast, public key systems provide a very elegant solution to this problem. The U.S. Digital Signature Standard, proposed by NIST, is based on discrete logarithms, as in the Diffie-Hellman system. Here we show a solution based on the RSA cryptosystem, which is described in Sidebar 2. Suppose that Alice's public key consists of (n, e). To sign a message m, when m is an integer in the range $0 \le m < n$, Alice attaches to it the integer

$$x \equiv m^d \ (\mathrm{mod}\ n) \ , \quad 0 \le x < n \ ,$$

where d is Alice's secret decoding exponent. Since Alice knows d, this is something she and she alone can do. Bob, to verify Alice's signature of m, computes

$$y \equiv x^e \ (\mathrm{mod}\ n) \ , \quad 0 \le y < n \ .$$

Since n and e are public, Bob can perform this operation. The property of the RSA cryptosystem discussed in Sidebar 2 guarantees that $y = m$. This proves to Bob that x was indeed generated by Alice since there is no known way to generate x from m without knowledge of the secret integer d.

The digital signature scheme described above can be used to show some of the pitfalls of using cryptosystems. It is possible for a system to fail even if the basic algorithm is secure. (This can happen to both public key and conventional cryptosystems.) For example, Alice should use separate keys (n, e) and (n', e') for encryption of information (which is sent to her) and for digital signatures (which she generates). To see the reason for this, suppose Alice uses a single key (n, e). Suppose that Eve, the eavesdropper, overhears Bob sending to Alice the message $c \equiv m^e \ (\mathrm{mod}\ n)$. All that Eve has to do to obtain m is to persuade Alice to sign c, since the signature of c is $c^d \equiv m \ (\mathrm{mod}\ n)$. Thus Alice's use of the same modulus and exponent in two different cryptosystems allows Eve to break them with Alice's unwitting cooperation. This is a case of protocol failure, and is one of the main vulnerabilities to be guarded against.