



PRIMEUR

Security Today

Security Today



Contents

Introduction	2
Security by any other name.....	2
The need for security.....	2
The explosion of E-commerce.....	3
The importance of integration.....	3
Basic definitions and aims of security.....	4
Cryptography; basic terms and concepts.....	4
The two basic aims of security	5
Security services and encryption algorithms.....	5
The economy of security; tactical and strategic encryption.....	6
Classical cryptography.....	8
Monoalphabetic ciphers; Caesar's security cipher.....	8
Polialphabetic ciphers.....	9
Keys and keyspace	9
Steganography.....	9
Modern cryptography	10
Ciphering machines, Vernam ciphers; cryptography meets the computer.....	10
The emergence of standards: DES and the problem of keys	11
Public key cryptography: RSA	13
The real world: hybrid systems	15
Strong encryption; Key recovery and Key Escrow.....	16
The new role of security.....	18
The irrefutable certainties.....	18
Digital signature.....	18
The weak point of public key cryptography and the certification of keys.....	20
Certification authorities.....	21
PKIs, Public Key Infrastructures	21
Certification infrastructure: 2 diametrically opposed models.....	22
The integration of security in information systems.....	23
Security in a complex information system	23
Link-oriented and end-to-end security	23
Modelling the system to achieve the desired effect.....	23
Security and the Internet.....	25
Cryptographic standards: a minimal glossary	27

Introduction

Security by any other name

The aim of this short document is to provide the reader with a basic knowledge of security and cryptographic concepts. It examines them as they have evolved over the centuries, from the first rudimentary ciphers, right up to the current levels of sophistication (and complexity) necessary for protecting information in the ever-present and global context of electronic communications.

Reading this document will not make you an expert on security (that would require rather more time). It should, nevertheless, provide you with a reference framework that will help you to place the various concepts in their correct perspective and to grasp new concepts. These concepts may be useful to you in your business, allowing you to speak confidently about this topic.

Finally, a word of caution: security has a wealth of meaning. We won't go too far into the various nuances of the term, but we will try to narrow down the discussion to those aspects that be useful to us.

The context we are dealing with here is the world of information technology, and especially the area of data communications. In this context, our data travels across networks that are potentially insecure, public, or subject to outside (possibly malicious) intrusions. What we mean by security in this case is a series of techniques for protecting data, as it is transmitted across unprotected channels. In this specific sense, security becomes synonymous with encryption. The security we require will be implemented through the use of various security techniques.

The need for security

Though cryptography has its origins in the military arena, it soon became an indispensable tool in the commercial world at large. Alongside these areas, the use of cryptography today has been extended to a third, that is the whole area of the rights of the individual. The rights of the individual, and above all the right to privacy, are universally perceived as a fundamental element of modern society. In various ways across different nations, this fact has given rise to laws protecting such rights. As we shall see, cryptography is closely bound up with this area, and is today the "technical approach" for implementing these rights in the context of information technology and telecommunications. We will look at the as yet unresolved debate sparked off by a conflict of interests – on one hand the citizen's need for privacy, on the other, the controls and security required by authority. The fact remains that the whole area of security is undergoing huge development, as its techniques becoming increasingly integrated into our everyday lives.

The explosion of E-commerce

The most important element driving the development of security is without doubt Electronic Commerce. Without digressing too far on the potential growth of E-commerce, we must underline how fundamental security is in this area. For E-commerce, security is not a nice-to-have feature or extra, but is simply a sine qua non. You cannot have an E-commerce application without adequate security. In fact, according to important analysts such as GG, the main obstacle to the development of E-commerce is the difficulty in implementing appropriate security tools for it. Without fear of contradiction, we can state that the growth rates of the security market mirror the growth rates of E-commerce.

The importance of integration

Security is an extremely technical area in which specialists are relatively rare compared to other professionals. Companies are springing up around the globe in response to this need. On the other hand, encryption software is widespread and easily obtained through the Internet; this software is very cheap, or even free, including perhaps the widely known and used PGP.

Let's look at the following question: what is the role of a company such as Primeur in this context? What **ADDED VALUE** can we bring to the market? The following points will help us to formulate an answer:

- Primeur is a specialist in the integration of heterogeneous environments.
- Primeur knows "big" environments well and the complexity of software installed in them. Furthermore, Primeur is particularly competent on the MVS platform.
- Primeur is able to understand the problems of high-end users and to organize solutions based on multi-vendor, and not only internal, technologies.

One single response arises out of these points: **INTEGRATION OF SECURITY IN THE COMPLEX ENVIRONMENTS OF HIGH-END USERS.**

What is specific to us, our added value, our capacity to generate margins is to be found not simply in offering an underlying security technology that is more or less as broad, flexible and comprehensive as that of our competitors. Our value lies in providing software and specific competence in understanding the issues involved in integrating security solutions into such environments.

Basic definitions and aims of security

This chapter provides a series of definitions and basic concepts in order to provide an outline introduction to the world of security.

Cryptography; basic terms and concepts

- Cryptology is the science that studies “secret writing”.
- Cryptography is the part that studies message protection systems.
- Cryptanalysis is the “evil twin” of Cryptography, that is, the part that studies how to succeed at decoding encrypted messages, against the will of the sender.
- Plaintext is an unencrypted message.
- Message encryption or encipherment is the process algorithmic transformation of plaintext to...
- Ciphertext or encrypted text.
- Decipherment is the legitimate (and easy) operation of decoding the encrypted text to plaintext.
- Decryption is the (difficult) activity of decoding by cryptanalysis, that is, in an unauthorised way.
- The key is the secret part of an encryption/decryption operation. The key is usually “small” compared with the message to be encrypted.
- The encryption of a message requires the use of an encryption algorithm and a key. By applying the algorithm to the plaintext message and to the key the encrypted message is obtained: $\text{encrypted_msg} = f(\text{plaintext_msg}, \text{Key})$.
- The secrecy of the encrypted text cannot depend on the secrecy of the used: on the contrary, the algorithm must be WIDELY known; in practice, STANDARD algorithms must be used. The part that makes the text secret must be the key alone. This is known as “Kerckhoff’s Principle”. This fundamental cryptographic principle is based on two premises:
 - An algorithm cannot be kept secret for any length of time.
 - A secret algorithm is not a secure algorithm: only testing and continuing resistance to cryptoanalytical attacks can demonstrate the real security of an algorithm.
- Symmetric systems are those security systems where the same key is used to encrypt and decrypt a message. This is also known as “secret key encryption”.
- Asymmetric systems, on the other hand, are those where the key used to encrypt is different from the key used to decrypt. These systems are of the greatest importance today. The two keys are related to each other, but deriving one from the other without any other information is “extremely difficult”. Such systems are also known as “public key encryption systems”.



- Encryption standards. Encryption standards include those algorithms, protocols, encryption procedures universally recognized, usually by international organizations, both private and public (ANSI, ISO, PKCS etc). In the world of security standards have, if possible, an importance even greater than they have throughout the world of IT generally, on two accounts:
 - A guarantee of interoperability.
 - A superior guarantee of security, given that the standards are public.

The two basic aims of security

Encryption techniques address two basic aims:

- **PRIVACY**: data can be read only by those for whom it is meant:
- **AUTHENTICITY**: data is authentic; it has not been modified; it really comes from the sender; the sender is who he/she claims to be.

In all forms of encryption both these elements are present to one degree or another. Authenticity can assume many different and important shades of meaning in this context.

We should first note the difference between authenticating data and authenticating the sender, and also be aware that one does not imply the other.

- You may go into a bank to withdraw money from your account, and the teller will check your identity. This is a case of **IDENTIFICATION**.
- Alternatively, you may want to cash a cheque made payable to you. In this case not only your identity, but also the authenticity of the cheque itself will be checked: the cheque must not be forged, the sum it contains must be the amount originally written by the payer, the signature must be genuine. In this case, we are talking about **AUTHENTICATION**.

In all forms of encryption both aspects exist explicitly or implicitly. We will look below at the various forms that they take.

Security services and encryption algorithms

The term security “service” is used to indicate the functional objectives to be achieved through security itself. Security algorithms are something else: the transformation of relevant data into an encrypted format.

Privacy or integrity are, for example, typical security services. They may be implemented with DES and SHA algorithms respectively.

A security service may be implemented through the simple application of a single algorithm. (E.g. privacy using DES)

The same service may be implemented using different algorithms. (E.g. privacy once again using DES or T-DES or RC4 etc.)

A security service may require the use of different algorithms (E.g. authentication).

A security service may require the use of a protocol for the interaction of the various parties involved and of the various algorithms used (E.g. PEA).

In general, therefore, there is no simple correspondence between services and algorithms. What is required is a security service; this service may be obtained through one or more algorithms, through one of several algorithms, in one or more steps.

Some security services:

- **PRIVACY:** Privacy is the first and most obvious service offered by security. Only the addressee can read the message.
- **INTEGRITY (No Tampering):** The data that arrives has not been changed, and is exactly the same as the data originally sent.
- **AUTHENTICATION and IDENTIFICATION:**
 - Identification is the process of verifying the identity of someone or something. Identification is what the bank employee is doing when you want to make a withdrawal, by checking your identification or recognizing you by your features. This same process can be carried out electronically, for example when you enter the PIN for an ATM card.
 - Authentication consists in verifying both the **integrity** and the **source** of a message. Authentication is similar to identification but is a wider concept. Verification of identity is not required, only verification of the right of the person or entity in question to perform a specific operation.Moreover, with identification a piece of information is compared with all the available related material. With authentication a single item of information must be the same as another previous single item of information. Identification identifies an entity uniquely; authentication does not necessarily require uniqueness (for example, the knowledge of a password necessary in order to access a joint account)
- **PEER ENTITY AUTHENTICATION (PEA):** Using this service two communicating entities (persons or processes) can be sure of each other's identity.
- **NON-REPUDIATION:** Guaranteeing that a particular transaction took place. The sender cannot later deny having sent the message.

The economy of security; tactical and strategic encryption

A basic guiding principle for all implementations of security systems is economy: the cost of a security system must be less than the value of the data that it protects. This principle tells us that subsystems exist whose data has little or no value, and for which there is no point in planning cryptographic implementations. On the other hand there are subsystems whose value would justify the deployment of large and costly applications.



Another important distinction, likewise based on economics, is that between tactical and strategic cryptography.

In each cryptographic system, the robustness of the actual system, that is, its ability to withstand cryptanalytical attacks, must be balanced against its ease of construction and use; these features are usually in inverse proportion to each other. The guiding criteria here is time consideration: how long must the information remain secret? And how long would it take an intruder to decode it? Some information only lasts for a matter of minutes (For example, the order that an artillery officer may give to fire in a certain position; a stock exchange order to sell immediately). These are cases of information that has great value for a very limited period of time; furthermore, such information must be acted on quickly. These cases lend themselves to the use of lightweight encryption techniques, which are easy to use, but would not hold out for long against cryptanalytical attacks. Other types of information (for example, the contents of your share portfolio) have a much longer shelf life and require (and can accept) the use of more robust (and heavyweight) encryption techniques.



Classical cryptography

All the classical encryption techniques (algorithms), developed empirically over many centuries and formalized in recent times, are based on the separate or combined action of two fundamental operations: transposition and substitution. These two transformations are the conceptual basis of all ciphers, both classical and modern.

By TRANSPOSITION we mean that the symbols that make up a message are “jumbled” in such a way that the final result is unintelligible. Only someone who knows the rule underlying the re-ordering can reconstruct the original message. The encrypted message is basically an anagram of the plaintext, all that has changed is the ordering. The encryption/decryption key is the order in which the symbols were transposed.

PIPPO -> key (43215) -> PPIPO

Another classical example: the text is written, in lines, in a square matrix, and encrypted by reading it in columns. The key is provided by the dimensions of the matrix.

In practice, ciphers based on transposition alone are weak. Even in their most complex versions, they leave themselves open to cryptanalysis. They are however very useful if used in conjunction with substitution ciphers.

In SUBSTITUTION ciphers the order of the symbols remains the same, but their appearance changes. The original symbols are substituted, according to a specific rule, by other symbols. Substitution ciphers have been, and with various changes still are, the fundamental technique of cryptographic algorithms.

In practice, mixed ciphers are used in several layers. For example, a transposition cipher is applied to a substitution cipher. This is the technique used by DES.

Monoalphabetic ciphers; Caesar’s security cipher

The first cipher mentioned in history is attributed to Julius Caesar. The algorithm is banal: each letter is replaced by another at a fixed distance in the alphabet. The distance is the key.

E.g. PIPPO -> key (+3) -> SNSSR

Caesar’s security cipher is obviously terribly weak and would present no challenge at all to a cryptanalyst. Its weakness is threefold:

- The key is too simple
- The number of possible keys is too small
- The relationship between plain and encrypted text is too strong (if you identify just one symbol, you can decode the whole lot)



Polialphabetic ciphers

The inherent weakness of monoalphabetic ciphers is the overly close relationship between plaintext and ciphertext: each symbol corresponds only and exactly to one other symbol. Monoalphabetic ciphers can not therefore escape frequency analysis. For this reason, the strict correspondence between plaintext and ciphertext must be broken.

One solution to this problem is to use several alphabets instead of just one. The alphabet used for encryption will change for each word, or even for each symbol. The key contains information about which alphabet to use.

Polialphabetic ciphers were discovered in renaissance Italy, and survive in various guises right up to our time. The simplest Polialphabetic cipher is the so-called Vigenere cipher. It is based on two alphabets using the Caesar cipher. A KEY determines which alphabet to use for each symbol.

Keys and keyspace

The size of the keyspace is the number of different keys that a particular algorithm permits. The total of possible keys for the Caesar algorithm is 25 (if the alphabet has 26 symbols and identical transformation is excluded). It is obvious that this number is unacceptably low and exposes the algorithm to a brute force attack (trying all possible combinations).

In general, a good cipher should not allow easy brute force attacks, and the keyspace is fundamental in this regard.

Moving to the present, the famous DES has a 56 bit key, which allows about $7 \cdot 10^{16}$ different keys. It seems an enormous number (70 million billion), but it is well known that DES was recently broken using a PC network.

It has been demonstrated that a key as long as the message to be encrypted allows completely secure encryption, in theory as well as in practice. This type of cipher is called a "Vernam cipher". Of course, managing this type of key poses insurmountable problems of a practical nature.

To summarize: the size of the key (and therefore of the keyspace) is vital for the effectiveness of an algorithm; what is secure today may not be tomorrow, in the face of the ever growing and cheaper processing power of computers.

Steganography

Finally, among classical encryption techniques, we should mention steganography, even though it is a rarely used technique. It consists of hiding one message inside another. Examples of this technique include using the first letter of sentences to make up a word or phrase, the use of invisible ink, or the use of a grill that hides some letters, but exposes others.

Modern cryptography

By modern, we mean cryptography created this century and developed following the introduction of long-distance communications, such as radio, telephone etc.

Methods of rapid communication lead to profound changes in the concept of communication itself; therefore, cryptography has adapted to a new role, becoming not only more powerful, flexible, and comprehensive, but also more widespread in everyday use.

Ciphering machines, Vernam ciphers; cryptography meets the computer.

Side by side with the revolution in communications, cryptography is transformed on the technical plain. So-called “ciphering machines” begin to appear, giving rise to “automated” encryption. But it is with the advent of the computer that cryptography finds its natural instrument, and forms an indissoluble union with it. Today security is impossible without the use of a computer.

The first ciphering machine, built by Vernam at the turn of the century, is an apparatus that can read two perforated tapes in input and generate a single perforated tape in output. The first tape represents the plaintext, encoded in some way (at the time Baudot’s 32 character code was used). The second tape, the so-called WORM, is the key. The exclusive OR operation is applied to the two tapes in order to obtain the output tape, that is, the encrypted text.

$$T \text{ XOR } W = E$$

Due to the properties of the XOR operation (it conserves information) the decryption operation is simply:

$$E \text{ XOR } W = T$$

Vernam’s tape system is very interesting because it implements an ideal cipher model, in which the security of the method depends wholly on the secrecy of the key and not on the algorithm used, the fundamental principle underlying all modern cryptography. Furthermore, Vernam’s cipher achieves unbreakable security, even on the theoretical front, as long as :

- The worm is random
- It is as long as the plaintext
- It is used only once

It goes without saying that a key which is as long as the message “destroys” the entire order of the message itself; on the other hand the practical difficulty in handling (storing, sharing) a key of this type is as difficult as managing the actual message, and therefore limits the application of such a method to wholly exceptional cases, here the correspondents are very few and the information is of extraordinary value.



Vernam's cipher is a type of Polialphabetic cipher, where substitution is carried out at the level of the individual bit, rather than at character level. It has the further advantage of being easily calculable and reversible (if the key is known), which makes it very suitable for automatic computation.

Vernam's technique, based on the properties of XOR, and on the concept of the secrecy of the key, remains, with shorter, more usable keys, at the base of a great number of encryption techniques used today. For these reasons, Vernam's cipher is usually thought of as the first modern "security" technique.

Around the same period, in the first decade of the century, calculator machines for automatic encryption were designed and produced (the so-called rotary machines).

Rotor machines are mechanical or electromechanical devices, based on a series of metal disks (the rotors) connected to each other. Each rotor performs the substitution of a character and passes the input to the next rotor. The final output is the enciphered character.

Generally, each rotor has a number of positions different to the others (its CYCLE). The famous ENIGMA, the machine used by the Germans in World War II used four rotors, with a full sentence, therefore using 26 symbols, of 26^4 . Enigma was broken by one of the fathers of computer science, Turing.

From a theoretical point of view a rotor machine is the equivalent of a Vernam tape system with a very long worm; furthermore, it fulfils all of Kerckhoff's security requirements, since the key is provided by the initial position of the rotors; without this key, even having an identical machine is useless.

It was however the advent of the computer that substantially accelerated the evolution of cryptography. The use of the computer, in fact, on one hand allows the creation of extremely complex ciphers, that could not be built using electromechanical means, and on the other hand provides cryptanalysts with a formidable tool for performing detailed analysis on encrypted data, to unearth any inherent weaknesses on which to build a decryption procedure; moreover, the computer is, of course, the perfect instrument for carrying out brute force attacks.

For these reasons the computer asserted itself as the cryptologist's tool even before computer networks were a reality; of course, the arrival of networks has revolutionised, redefined and broadened the role and the use of cryptography.

The emergence of standards: DES and the problem of keys

With the increase in electronic communications for civil and commercial use the need for companies to protect their data and communications has been developing since the 1970s the growth of this need is proportional firstly to the growth in the amount of data that is managed automatically, and secondly, to the growth in electronic communications. On top of this, the vast majority of electronic



communication will travel across channels that are intrinsically insecure, such as telephone, radio waves and the Internet.

In the midst of the proliferation of diverse commercial encryption systems, almost exclusively in the US, it became obvious that their incompatibility could have become a serious obstacle to the requirements of companies to be connected amongst themselves and to central government.

At the start of the 1970s, the National Bureau of Standards made a public call for a cryptographic system strong and robust enough to be proposed as a national standard. The winning system was DES, the first publicly available system, recommended and certified by a civil normative entity.

DES is a 64-bit block cipher (data is processed in packets) with a 56-bit key. It is a compound cipher, made up of successive elementary transformations substitution and transposition. DES is a symmetric system (the same key is used both to encipher and decipher) and was designed for the transformation of data in real time, through Hardware implementations.

Though it has received much criticism over the years, especially due to the possible weakness arising from the short length of the key, but also due to some confusion over the theoretical choices on which the algorithm itself is based (it is of course publicly available), DES has protected, in twenty years of honourable service, the greater part of financial transactions the world over, as well as a great quantity of commercial transactions.

The spread of electronic communications, however, soon plunged traditional encryption systems into crisis; these systems are in fact designed to manage the flow of communication amongst a limited number of known and trusted correspondents, not the mass communication between thousands, or even millions of correspondents. What impedes it is that oldest and most delicate of problems connected with encryption systems: key management.

In a system where all the secrecy is tied to the key, it is fundamental to maintain the secrecy of this key at every stage of its life: generation, storage, distribution, duration. At any of these stages there can be a lack of security that would compromise the entire structure that was built so carefully.

Who generates the keys? How random are the generated keys? How and where are they stored? Who can access the key file, and with what rights? What channels are used to distribute the keys? How secure are these channels? How long should a key last? How is a key recognized as old? These and other questions must be answered promptly and satisfactorily. Without these answers, security simply does not exist.

The problem of keys attains biblical proportions in traditional symmetric encryption once the number of correspondents increases. If each pair of possible correspondents has to share a symmetric key, we can see that the number of keys necessary for n correspondents to communicate is (n^2) , that is $n(n-1)/2$. For a medium-sized company, that needs 1000 correspondents, the number of keys



required is in the order of 500.000. In a big company with 10,000 correspondents this number rises to 50.000.000. On the Internet, with millions of correspondents you would have thousands of billions of keys. And, each key would have to be properly stored, transmitted and renewed.

The problem is further complicated by the rule according to which secret keys are exchanged often, for two reasons: on the one hand, the frequent use of a key generates a big sample on which to try cryptanalysis; on the other, if a key is captured it will compromise only those messages for which it was used.

Particularly important, then, is the problem of transmitting secret keys. Cryptography books usually avoid embarrassment at this point with a phrase such as: "keys must be transmitted over a secure channel, different to the insecure one usually used for exchanging messages". A secure channel may be a human being, a courier, a special electronic channel. These channels are usually costly and of limited capacity. Furthermore, they are never totally secure.

The root of the problem lies in the fact that the secret key is not really secret at all, because it must be known to at least two participants; this is the source of all the difficulty in managing keys and channels.

It is clear that overall this situation in an open context and where participants in the communication are many becomes absolutely unmanageable, because of the dimensions it takes on and the difficulty of the problems that arise. A completely new approach is needed, that goes to the very core and changes the terms of the discussion.

Public key cryptography: RSA

In 1976 Diffie and Helman propose a new and revolutionary cryptographic system. This system proposes an ASYMMETRIC type of encryption; there are two keys, distinct (but not independent) that perform inverse transformations to each other. If K1 is used to encipher, its sister key K2 is used to decipher, and vice versa. A single key is not enough for this process. The relationship between K1 and K2 is unique, but is such that extracting one of the two keys from the other is unimaginable in terms of calculation.

The same Diffie and Helman propose this application of the new discovery:

- Each party in a communication has (at least) one pair of keys.
- One of the two keys is secret and is known to absolutely no one else; it is called the "private" key.
- The other key (the inverse of the previous key), is, on the contrary, made known to everyone; ideally this key, called the "public" key, is published in a directory analogous to the phone book, that may be consulted by anyone.
- Let's suppose we want to encipher a message for user A. We will look for A's public key in a directory and use it for encryption. Only the person with the inverse key, that is A alone, can decipher the message. We have now implemented privacy.



- Suppose now that we encrypt a message with our own private key. Since everyone has access to our public key, everyone can read the message; only we, however, are able to encrypt it, and therefore the message can only originate with us. We have now implemented authenticity. Encryption with the private key is not really to render the message unintelligible, but rather, in a certain sense, to “sign” it”, so that there is no doubt about its origin.
- If we required both types of functionality (privacy and authenticity) we merely have to superimpose two layers of encryption, for example, first privacy (encrypt with the public key) and then authenticity (encrypt with the private key).

Diffie and Helman’s security proposal was implemented in practice, in algorithmic terms, in 1977, by three MIT researchers, Rivest, Shamir and Adleman; their initials form the acronym RSA. RSA therefore refers to a particular implementation of public key cryptography; RSA has become the de facto standard in this area, to the point that the two terms (RSA and public key encryption) are incorrectly used as synonyms. Today, RSA inc. is a private company that is very active in the whole area of modern cryptography and acts as the focal point for the definition of a whole range of standards, called PKCS (Public Key Cryptographic Standard), connected with the world of asymmetric encryption and its most recent developments.

The RSA algorithm exploits the computational complexity of a well-known problem in the theory of numbers, the factorisation of an integer, in order to define a so-called pseudo-non-invertible function, or rather, a function that is very easy to perform in one direction, but computationally almost impossible to perform in the other; it is very easy to multiply two, even very big, numbers; it is extremely difficult to find the prime factors of a very large integer. While this is not the place to provide a detailed explanation of the elegant RSA process, suffice it to say that the difficulty of deriving a private key from its public counterpart is the same as the difficulty of finding the prime factors of a very large integer.

The advantages of RSA

RSA’s first great advantage is a conceptual one. In symmetric encryption the privacy and authenticity of a message are inextricably linked; in particular, authenticity is in some way implicit. In asymmetric encryption, however, the two types of functionality are separated and explicitly linked to a specific key and the encryption operation linked to this key: encryption with the private key produces authenticity; encryption with the public key produces privacy. This mechanism allows great flexibility and lends itself to a wide range of applications, as we shall see further on.

The second great advantage is the drastic simplification, at the root, of the problem of key management. As we have seen, in asymmetric encryption the number of keys required to allow n entities to communicate is proportional to n^2 , whereas, in symmetric encryption each participant needs two keys; therefore, the total number of keys required is simply $2*n$. The growth in the number of keys with the growth in

the number of users is linear, and therefore manageable even when there is a “large” number of users.

Lastly, not only is the number of keys greatly reduced, but the management of these keys is simplified to the point of banality. Every user must have a pair of keys, that he/she generates for himself/herself. The secret key must not be shared with anyone and so the problem of transmitting it does not arise, nor do the problems of secure channels and their management; the secret key really is secret, since it is shared with nobody. The public key, however, is shared with everyone, for example in a catalogue analogous to the telephone directory, to which it can be transmitted using the most convenient method, posing, by definition, no problem regarding its privacy.

Disadvantages

The main disadvantage of public key encryption is its well-known computational load. In order of size, asymmetric encryption is 1,000 slower than symmetric encryption. Furthermore, for the same length of key, asymmetric algorithms are less secure than symmetric algorithms when it comes to brute force attacks, therefore longer keys must be used to attain the same level of security. The typical RSA key today is 1024 bits. A “secure” symmetric key is 128 bits. This means that today asymmetric encryption is suitable only for “small” portions of data, while to encrypt entire files you have to use asymmetric encryption.

Finally, we have seen how the problem of key management has been greatly alleviated by public key encryption. However, the problem has not disappeared entirely due to the fact that you cannot be sure of the origin of a public key. This problem, for which various solutions have been found that we will look at later, has brought up once more, albeit in a less onerous form, the problem of key management and the infrastructures it requires.

The real world: hybrid systems

We have seen how RSA solves a whole series of otherwise insurmountable problems, in such a way that it can assert itself as the first real tool that allows amass application of cryptography. We will now see how it can be applied in reality, providing remedies for its drawbacks, which we may remember are its computational overhead and the problem of the source of public keys. This paragraph looks at solving the first of these problems.

Bearing in mind too that symmetric encryption is the only type currently suitable for anything other than the smallest amounts of data, here is a typical scheme of how two entities, A and B, may exchange a file securely.

- **A** dynamically generates a key **K** for symmetric encryption, for example DES.
- **A** encrypts **K** with **B**'s public key and the file with **K**.



- **A** constructs a message with **K** and the file, encrypted as above and sends it to **B**.

$$msg(A \rightarrow B) : \left\{ \left\{ file \right\}_K, \left\{ K \right\}_{Key_Pub_A} \right\}$$

- **B** deciphers **K** using his own private key ; then proceeds to deciphering the file with **K**.

In this example we should note:

- The problem of encryption performance is solved since the symmetric key is used to protect the file; the asymmetric key is used only to encrypt the symmetric key, a packet of few bytes.
- The problem of symmetric key management does not arise: the symmetric key is generated dynamically when it is needed, and transmitted securely, since it is encrypted; its validity lasts only for a single transmission session, in this way increasing security, as it is not reused in further sessions.
- The technique that allows the key to be sent, protected in an asymmetric way, along with the message that it is protecting is known as a “Digital Envelope”. There are standard digital envelope formats, such as S/MIME or PKCS#7, that allow for interoperability of cryptographic operations in heterogeneous environments.

In reality, mixed techniques are used, that allow the exploitation of the advantages of both symmetric and asymmetric encryption, on one hand, and the canceling of their drawbacks on the other. For the whole mechanism to work (be interoperable) in an open and distributed context requires the **standardization both of the algorithms used and the formats of the data exchanged**.

Strong encryption; Key recovery and Key Escrow

We have seen how the length of the key is a fundamental element constituting the security of a cryptographic algorithm. The length of the key directly determines the size of the keyspace and this, in turn, since the algorithms are necessarily public, is the only defense against a brute force attack.

It was recently proved that DES, with its 56-bit key, is no longer secure to counter malicious attacks by those determined enough, even if they do not have access to huge computer resources, while algorithms with 40-bit keys offer no real protection at all, not even to the most casual of cryptanalysts.

Various studies show that 80 and 90-bit keys are very secure today, and can resist attacks even by organizations who can afford to spend millions of dollars on computer power. 128-bit keys are extremely secure today and it is estimated that they will remain so for the next 20 years. We should bear in mind that DES, which still continues to be useful in the efficient protection of data that does not have enormous value, has been around for more than twenty years.



Strong encryption is a qualitative term, referring to all those cryptographic techniques whose decryption is “difficult” and impracticable without the availability of huge technological and financial resources; examples of what is thought of as “strong” encryption today are 128-bit symmetric encryption and 1028-bit asymmetric encryption. The concept of “strong” encryption is however a concept tied to time limits: what was strong 20 years ago is not today, and so on; this fact has certain important technical implications, for example for the storing of documents.

Strong encryption techniques are available today, for example T-DES (that performs the DES process three times, using two standard DES keys) or RC4 that has a variable key length of up to 128 bits.

Strong techniques may however pose serious problems of a legal and even political nature.

To start with, we should remember that the first application of security is military. For example, strong cryptography is considered a strategic weapon in the US and as such, subject to appropriate limiting controls on exportation.

Furthermore, cryptography can obviously be used by criminal organizations to protect their own communications; it is natural that central governments want to limit and control its use, or to provide mechanisms that allow, where necessary and authorized, the decryption of messages. At this point the rights of the individual to privacy collide with the duty of governments to guarantee security and the rule of law for all their citizens.

The debate on strong encryption touches here on some fundamental themes of modern society; needless to say, this debate has taken various directions in various countries, for example, rather restrictive in France (until recently, encryption between private individuals was simply prohibited!), more liberal in Italy, lively dialectic in the US. The legal aspects connected with the production and use of cryptographic systems are however in a dynamic phase across the globe, and cannot be ignored in acting appropriately in this area.

Lastly, we would like to cite some of the techniques proposed for decipherment (by anyone authorized) of strongly encrypted messages. Key recovery is basically any technique that can reconstruct a key or recover the key used to encipher a particular message. Key Escrow is a specific type of Key recovery. As the term suggests (Escrow = the trusting of third parties), Key Escrow presupposes the existence of a trusted third party that keeps the cryptographic keys; under specific conditions these keys may be sent to the requesting authority; to this end, there is a standard for the Escrow procedure called EES (Escrowed Encryption Standard). In 1994 a legal proposal was presented in the US which would have made EES mandatory for all “strong” techniques; the proposal was never approved due to the lively debates it generated. The problem of balancing the privacy of the individual with the security of citizens remains open and complex.

The new role of security

For centuries the role of encryption was to protect private information from prying eyes; today, and from now on, its role will be presented quite differently. It may in fact provide the technical solution to a crucial problem for the future of the cabled society: allowing the use of electronic documents to the same degree as traditional documents, with legal certainty. The obstacle to be overcome is connected with the intrinsic lack of security of communication channels, which leads to uncertainty about the identity of correspondents and the authenticity of messages. Cryptography may restore certainty to network communications and so contribute to the realization of a civil society based on digital communication. Alongside cryptography that hides, we have then a cryptography that **GUARANTEES**, that provides certainty and authenticity for digital information.

The irrefutable certainties

What are actually the aspects of communication where cryptographic techniques the required tools of guarantee and trust? We will distinguish again between the two basic services provided by cryptography.

The first is privacy, which has been until now the main, if not the only, aim of classical cryptography.

The second is what we have until now called authenticity, referring perhaps incorrectly and certainly generically, to all those services connected with the various **GUARANTEES** that may be obtained through cryptography about the authenticity of messages and of their authors. Guarantee services provide the certainty that the whole process of sending and receiving the message is legitimate and is carried out by whoever has the right to do so. We may divide such services into three main areas:

- **IDENTIFICATION.** Certainty about the identity of correspondents.
- **AUTHORISATION.** Certainty about the legitimacy of the operation.
- **AUTHENTICATION.** Certainty about the authenticity and integrity of the message.

Digital signature

The digital signature is the equivalent, in the world of electronic information, of the hand written signature on traditional documents, but offers greater guarantees of security and has entirely new properties, such as the impossibility of appending it to a blank document and non-repudiability. We will look in detail at how this mechanism is implemented and at the properties it possesses.



Algorithms used for digital signatures

Suppose **A** wants to send **B** a message or document digitally signed. These are the steps that must be taken:

- First of all **A** performs a “hash” function on the entire document. The output of the hash function is the “message digest”. The message digest is, so to speak, the digital fingerprint of the document and is normally much shorter than the document from which it was calculated (several bytes). The purpose of the hash function is to “condense” the document into a short fixed-length piece of data, which can be conveniently encrypted using an asymmetric algorithm.
- As a second step, **A** encrypts the message digest with an asymmetric algorithm using his own private key, thus obtaining what is known as a “MAC”, Message Authentication Code.
- The third step is optional. **A** can, if necessary, encrypt the document (symmetrically), as explained in paragraph 4.4.
- **A** now composes a message containing the MAC and the original document (encrypted or otherwise); this composition is carried out in accordance with one of the digital envelope standards such as PKCS#7; this envelope is sent to **B**.
- **B** receives the message, opens the envelope and, supposing for simplicity’s sake that the document is not encrypted, applies the same hash function to the document, obtaining a new message digest.
- **B** now decipheres the MAC contained in the envelope, using **A**’s public key, recovering the original message digest calculated by **A**.
- Next **B** compares the two message digests; if they are identical, then everything has gone as it should. If not, then there are several explanations:
 - The message is not from **A**
 - The message has been altered in transit
 - There was an error in the transmission of the data

Note that the use of the hash function presents a difficulty: two different messages could produce the same message digest, causing a so-called “collision”; the better the hash function, the less possibility there is of a collision.

Finally, the concept of a timestamp can be introduced into the scheme of the digital signature. Tying the signature in this way to a particular date and time retransmission attacks can also be prevented.

Properties of digital signatures

- The digital signature is separate from the document to which it refers and does not alter it in any way.
- The digital signature is dependant on the document to which it refers and therefore differs from document to document. It identifies the pair (signed document, signing entity) uniquely; therefore it is valid only for the document that was signed and cannot be forged, imitated, copied or duplicated on other documents.
- The digital signature cannot be created separately; it cannot be calculated independently of a document.
- The validity of digital signature, since it is the product of a known calculation, can be verified by anyone in a certain and repeatable way. In other words, there is no need for experts and there is no uncertainty about the authenticity of the signature.
- The digital signature will reveal any alterations to the original text made after the document was signed, providing in this way a “no tampering” or “integrity” service.
- A digital signature cannot be repudiated. Once you have digitally signed a document you cannot deny ownership since the digital signature cannot be copied or forged, and cannot be generated by anyone other than the legitimate owner of the key used for that particular calculation.

The weak point of public key cryptography and the certification of keys

Public key cryptography and therefore all those security systems built on it have one weak point, inherent in the very structure of the system; a real Achilles heel, to quote Phil Zimmermann, author of the famous PGP.

We have seen how the whole structure of asymmetric cryptography rests on the existence of a pair of keys, one of which must be “public”, accessible to all, for example residing in a freely accessible catalogue. Now, in the absence of any check, anyone could deposit a public key in the name of someone else and in this way pretend to be the other person. In other words: how can one be sure that a public key really belongs to whom it supposed to belong?

This problem can only be resolved by leaving the task of certifying public keys, and thus guaranteeing the identity of their owners, to a third party, trusted by everyone. In practice, every public key, before being used, must be “verified” by a suitable Certification Authority or CA that will guarantee its origin and ownership.

This guarantee is implemented through the publication of a CERTIFICATE that guarantees the authenticity of a public key. Technically a certificate is simply the public key in question signed with the private key of the CA that issued it; so the trust you must have before using a public key is replaced by the trust you have in a superior entity: the key is trustworthy because the CA that I trust is trustworthy.

In their most simple form, certificates contain a key and the name associated with that key. Normally, this information is extended to include the “expiration date” of the certificate itself, the name of the CA that issued it and a serial number. Various extensions to this format have been proposed recently allowing the specialization of certificates for various uses. The de facto standard for the format of certificates is the so-called X.509, currently at version 3.

As well as being issued, certificates can be revoked. Each CA keeps a list of certificates (CRL, Certificate Revocation List) revoked before their natural expiry date. A certificate may be revoked for a number of reasons: for example, a key specified in the certificate may have been compromised or the user specified in the certificate may no longer have authorization to use a particular key. Before using a certificate, it is essential to examine the current to make sure it is still valid.

Certification authorities

The role of a CA, within the infrastructure of trust based on asymmetric cryptography is that of a guarantor of the identity of each user to all the others. This is obviously a delicate role that must be undertaken by subjects of proven trustworthiness and above all parties, especially in an open context; in a closed context, a company for example, the CA can be entirely internal, and be part of an infrastructure dedicated to the internal security of the company itself.

Among the various subjects that could perform this service, the current tendency is the services of private commercial companies specialized in this role, such as Verisign. In the context of the Italian public administration the role of CA is performed by AIPA.

The central element of a trust infrastructure constituted in this way is the public key of the CA. It must be made known and accessible to everyone and the CA must be universally worthy of trust; alternatively, a CA can supply its own public key attached to a certificate from a higher level CA. This is how the so-called CA hierarchies are made up. They may contain varying levels at the top of which there must be a CA capable of certifying all the others.

PKIs, Public Key Infrastructures

A PKI consists of a series of protocols, services and standards that support public key cryptography applications. The term PKI is rather recent (and fashionable) and has not unfortunately got one single meaning in all the literature concerning it. Sometimes, PKI may refer to a hierarchy of trust based on certificates; in other contexts, the concept includes services such as encryption and digital signature. A definition shared by many could be that according to which a PKI includes services and protocols for managing public keys, typically through the use of CAs, but not necessarily cryptographic services carried out with these same keys. Some typical services of a PKI, in this meaning of the word, would be:

- Registering public keys and issuing a new certificate for a public key
- Revocation of certificates and managing CRLs

- Distribution of the public keys of registered users
- Determining the validity of a certificate and of the operations that it authorizes
- Services related to certificate requests

Here, the concept of PKI is almost analogous to that of the CA, putting the emphasis more on services provided than on the authority necessary for rendering them. Another way of looking at this concept could be this: the CA is the “Server” part of a PKI, which however includes all the services necessary for managing public key encryption.

There is currently no single PKI, even if efforts at standardization are being undertaken in various parts of the world. The greater part of PKIs are however based on the X.509 standard for certificates.

Certification infrastructure: 2 diametrically opposed models

There are currently two differing and opposing models for regulating the relationships between all the subjects performing a certification role.

The **HIERARCHICAL** model, adopted by the X.509 standard, outlines a hierarchy of CAs of ascending levels. It is a structure that is undoubtedly effective but the details of which must be studied with great care, because the trust of the whole structure may depend on a single certification element. It is not explicitly supported for two CAs (or two users) to mutually recognize each other; mutual certification must take place through the hierarchy.

The **DECENTRALISED** model is that adopted by PGP and its many users, with the concept of a “Web of Trust”: CAs are of no use because each has the right to decide independently who to trust and who not to trust. Each user can, if he wishes, act as a CA to other users, signing with his own private key the public keys those users that he wants to guarantee. Of course this scheme expands to generate a web of trust, that in a way implements a transitivity of trust: if **A** trusts **B**, and **B** trusts **C**, then **A** can trust **C**.

The two models are really opposed and, perhaps, destined to lead separate although contiguous lives. The hierarchical model in companies or other structured environments; the decentralized model in the community of Internet users or groups of individuals connected through common interests and not organized hierarchically.

The integration of security in information systems

Security in a complex information system

When security functionality is introduced into a complex computer system, the architecture of this system cannot be ignored. In a computer system with various users, various applications, and various platforms, each with various levels of system software, inter-connected using middleware, the choices on where to insert security functionality have implications that cannot be overlooked. Even using the same algorithms, procedures and services, these choices have different effects on the level of security achieved and therefore alter the complexity of security features of the system in a FUNCTIONAL way.

Link-oriented and end-to-end security

The first big division is between link-oriented measures and end-to-end measures. In link-oriented measures you want to protect the communication channel and establish a secure communication channel. Such measures provide security for the information that passes on an individual link between two network nodes, without reference to either the source or the ultimate destination of the information. In a network that uses this type of approach, encryption is performed independently for each link. If one link is successfully attacked, the loss of privacy is not propagated to the rest of the network. Since the information is encrypted on the link, and not on the nodes, the nodes must themselves be secure.

This is the case with protocols such as IPSec or PPTP which act at the packet transport level in an IP network. In a different way the same considerations apply for those products that use exits of middleware products such as MQSeries, even though this occurs at a higher level in the network. In both cases, however, the “higher” levels, and highest of all the application layer, are unaware of the underlying processes, which are transparent to them.

Link-oriented measures see the network as series of nodes united by links, each of which can be independently protected. End-to-end measures, on the other hand, see the network as a means for transporting information securely from source to destination. Violation of an intermediate node will not compromise security. Therefore, in end-to-end measures, what one wants to ensure is that security reaches the correct level, typically the application level. The application is therefore no longer transparent to security, but explicitly orders its functionality, thus offering higher guarantees.

Modelling the system to achieve the desired effect

The essential element, therefore, in approaching an integration project in a information system, is the design of the system itself, based on which you can make choices about where to introduce the desired security functionality.



Obviously, there is no single model or level of detail on which to base the design. Everything depends on the functionality that you want to achieve, and above all on the entity that you want to protect.

The following could be an example of this type of activity, in which 3 different levels are distinguished:

USER. At this level is the actual user of the system (the human being) who uses the security services. An ad hoc application is supplied, allowing the user to explicitly sign data using his/her own key. The operations performed by the user allow him and his messages to be authenticated and to consider the level of privacy active up to his level. A digital signature, for example, will be a question of "Signing on the bottom line" to create a signature. Note that no transparency is possible regarding security functions: the user is requested to action explicitly.

PROCEDURE. Or application program. Security operations carried out at this level guarantee certainty on the origin of data, where the originator of such data is an application. This will be obtained by introducing into the actual application the various calls to security functions. Still using the signature example, we will speak in this case of "Signature of the Procedure". It may happen that the user level and procedure level coincide (typical in the case of PC users). In any case, the sender is authenticated as is certainty on the origin of the data, wherever that is.

CHANNEL. Or transport process. In this case, security is introduced at the level of the channel or the software process that interfaces the transmission channel. This type of security is less strict than the previous two, since it intervenes after the data has already been handled by the user and/or the application. In the context of SPAZIO this would be a SPAZIO exit that processes the data before it is sent. In the MQSeries context, it would be an MQSeries channel exit. At this level you can guarantee against interference on the network (no tampering, privacy from external access) and again you are guaranteed of the identity of the originating process (or more simply, of the originating Hardware platform). A secure communication channel has been established. No other assumptions can be made as to the identity of the originator or the authenticity of the data connected to it, apart from making limited (and usually unrealistic) presumptions on the security of the sites involved.

Not also that the term CHANNEL can be ambiguous; it may be identified with the middleware level, the transport level or even the network. Note too that choices made have implications for system performance.

In short, introducing security into a computer system cannot be limited to the supply of "algorithms" or "services" of various levels, but must be tied to a project that analyses appropriately and provides answers regarding the elements listed above.

Security and the Internet

Lastly, we want to dedicate an appropriate paragraph to the theme of security on the Internet, because of the omnipresence of this medium, and because the Internet is now becoming the “insecure channel” par excellence and therefore is the first object on which the various techniques previously described were tested. We shall comment briefly on a series of so-called “security protocols”, each of which implements in one way or another the public key cryptography techniques discussed previously. Notice how the functional and application differences between the various protocols do not depend on the different algorithms or services offered (which are actually always the same), but on the way in which the system interfaces with the network (and on the objects that are to be protected).

■ IPsec, PROTECTION FROM THE BOTTOM

TCP/IP does not include methods for the cryptographical protection of packets. For this, another protocol was designed, the IPsec, based on TCP/IP for reasons of compatibility. With IPsec you can obtain security at the lowest communication level, that is at the packet level. Functionality includes authenticity and encryption of the actual packets. IPsec is adapted to protect an entire network and to create a so-called VPN; it provides a type of security closer to the physical layer, designed therefore to neutralize passive “wire tapping” attacks.

■ PPTP, Point to point tunneling protocol

PPTP is another link-oriented security protocol, used especially in building VPNs, that is, Virtual Private Networks built on the public network (Internet).

■ SSL, NETSCAPE’s attempt

SSL, Secure Socket Layer, is a protocol developed by Netscape for data protection and authenticity in Web communications; it provides services for authentication, no tampering and privacy. Servers and clients can authenticate each other and establish a “secure channel” over the Internet or an Intranet in order to protect a subsequent exchange of data. It supports a wide range of algorithms such as DES, MD5, RC4, IDEA etc. While IPsec operates at the level of individual packets, that is at the level of the network, SSL operates at the transport layer, or more precisely between the application and the data transport layers, at a higher level, therefore, than IPsec, but lower than the application layer. SSL the dialogue between two applications, where IPsec guarantees the whole network. SSL allows authentication of the Web server to which you connect, a very important factor guaranteeing the communication of private information with servers guaranteed by our browsers. Furthermore, SSL allows authentication of the user’s browser through a procedure based on the use of an RSA public key. Through this procedure the user can automatically obtain SSL authentication of the client. SSL currently has a discreet presence in the area of Internet e-commerce.

■ S/MIME

Secure MIME is an extension of the MIME (Multipurpose Internet Mail Extension) standard that allows that allows the expansion of this format to include services of authenticity and privacy. The S/MIME protocol guarantees the security of messages at various stages of transmission, storage, authentication and so on. While SSL and



IPSec, in different ways, protect the communication or the participants in the communication, S/MIME protects data in each phase of its existence; for this type of application S/MIME is the de facto standard.

Cryptographic standards: a minimal glossary

We provide here a (far from complete) annotated list of the main cryptographic standards, ordered by issuing body.

ANSI. American National Standards is an American entity that is subdivided into a plethora of committees and sub-committees. ANSI X9 (in some documents referred to as ANSI X9.n.) in particular deals with cryptography. ANSI standards include the DES (symmetric 56-bit key) and T-DES (three passes with 2 DES keys) algorithms. Also hashing algorithms such as SHA-1 and MDC-2 are ANSI standards. ANSI supports the use of RSA for various security protocols, such as Diffie-Helman or other security for exchanging secret keys.

ITU-T. Formerly known as CCITT, the International Telecommunications Union is a world wide grouping that provides standards for telecommunications products and services. Of interest to us is the **X.509** recommendation that specifies authentication services (**PEA X.509**, to be exact, in 2 variants, with 2 or 3 passes) and especially the format of **certificates**, now at version 3, which is the world de facto standard for this type of data. Note that version 3 of X.509 certificates supports the use of “extension fields”, fields that allow the expansion and customisation of the use of certificates. While this innovation undoubtedly improves the usefulness of certificates themselves, it is an obvious limit to standardization. X.509 provides a standard syntax for Certificate Revocation Lists (**CRLs**). X.509 does not prescribe specific encryption algorithms, although it does describe the RSA standard.

PKCS. Public Key Cryptography standards are a series of standards for public key cryptography, developed by RSA Labs in co-operation with an informal consortium including Microsoft, DEC, Lotus, Sun and others. The PKCS standards are compatible with the ITU X.509 standard and are the de facto guide for all applications of asymmetric encryption. These are the most important:

PKCS#1. Defines the RSA algorithm

PKCS#3. Defines the Diffie-Helman key sharing protocol

PKCS#7. Defines the general syntax for messages including security services such as digital signature and symmetric encryption.

PKCS#10. Defines a syntax for certificate requests.

PKCS#11. Defines a programming interface, independent of the supporting technology, called Cryptoki, for the management of “Cryptographic Tokens”



Here is a summary of the various aspects that are open to standardization and the related PKCS:

Standard	PKCS #								External work
	1	3	5	6	7	8	9	10	
<i>Algorithm-independent syntax:</i>									
digitally signed messages					x		x		
digitally enveloped messages					x				
certification requests							x	x	
certificates									X.509, RFC 1422
extended certificates				x			x		
Certificate revocation lists									X.509, RFC 1422
encrypted private-key info.						x	x		
key agreement messages									[ISO90a], [ISO90b]
<i>Algorithm-specific syntax:</i>									
public keys: RSA	x								
private keys: RSA	x								
<i>Algorithms:</i>									
message digest: MD2, 5									RFCs 1319, 1321
secret-key encryption: DES									RFC 1423, [NIST92a]
public-key encryption: RSA	x								
signature: MD2, 4, 5 w/RSA	x								
password-based encryption			x						
key agreement: D-H		x							

Correspondence between aspects suitable for standardization and PKCS.

ISO. The International Standards Organisation promotes the development of world standards. All ANSI standards have an ISO equivalent which is the standard more frequently referred to in Europe.