GBS

Cryptography

- An introduction to encryption -

Principles of cryptography and general encryption methods

Expertise matters

Contents

1 Executive Summary

The amount of communication handled via email has strongly increased in the last couple of years and is still growing. There is hardly a company that does not use email to run its business processes both within the company and with external business partners. In addition to short response times, constant reachability and cost-efficient communication, major issues also include the security of emails and the protection of confidential email contents. Many companies therefore rely on email security solutions that also include encrypting emails. With its Crypt module, iQ.Suite from GBS Software offers a comprehensive and complete policy-based solution to these issues. The present documentation deals with the basic principles of cryptography and provides an overview of the different encryption options and methods available. Details on the Public Key Infrastructure (PKI) and the implementation in iQ.Suite Crypt Pro are provided in two additional whitepapers, which are both available for download on our Whitepaper Download Page.

2 What is Cryptography?

This must thou ken:

Of one make ten,

Pass two, and then

Make square the three,

So rich thou'lt be.

Drop out the four!

From five and six,

Thus says the witch,

Make seven and eight.

So all is straight!

And nine is one,

And ten is none,

This is the witch's one-time-one!

Goethe, Faust I

Is that enigmatic, magic and cryptic? Well, at least it is encrypted. Whether or not Goethe based the witch's poem on true mathematical rules and, if so, how it is to be decrypted, still remains a controversial issue among experts, even though a number of magic squares have been found to date.

Cryptography, in a wider sense also referred to as cryptology, is the (mathematical) science of keeping data secure by way of encryption and decryption methods. A typical and simple example of a cryptographic method is the so-called CAESAR cipher. The method is very simple:

A text is encrypted by substituting the letter A with the letter D, B with E, C with F, etc. The key is the number of positions each letter is shifted within the alphabet, i.e. 3 in our example. Thus encrypted, GROUP would read JURZS.

A cryptographic method is considered secure when it is difficult to decrypt an encrypted message without knowing the key, even though the method itself may be known. In this context, “difficult” means that decryption is impossible by reasonable standards. Thus, the CAESAR cipher is certainly not a secure encryption method, as simple trial and error will quickly reveal the actual text.

3 Attacks on Encrypted Emails

Whatever encryption system is used, potential attackers will always try to exploit the system’s weak points as far as possible. Cryptanalysis is the science of making encrypted data unencrypted without knowing the key. Therefore, attackers are also often called cryptanalysts, as the basic methods used for “legal” cryptanalysis (evaluation of cryptographic power) and “illegal” cryptanalysis (unauthorized decryption of data to retrieve secret information) are identical.

With the encryption methods typically used today and described further down, two communication partners exchange data over an insecure channel. Focusing on security-relevant issues, we will assume that all information on the system used (excluding the key) is available to a potential attacker, so that he has unlimited access to the communication.

Digital signatures can be used as evidence that a document really comes from the sender specified and, as such, that the communication partner is always the same. However, this does not provide a guarantee of the actual identity. To get that kind of guarantee, one has, among other things, to register personally with a Trust Center.

The following distinction is made between attackers:

- Passive attackers – they monitor a communication channel in order to decrypt messages.
- Active attackers - they are also able to manipulate transmissions.

The following weapons against cryptosystems are available to attackers:

- Ciphertext-only: the weakest form of attack. The attacker is assumed to have access only to a set of ciphertexts, but not to information that goes beyond generally known facts.
- Known-plaintext: Attack in which the attacker has samples of both the plaintext and its encrypted version.
- Chosen-plaintext: This attack presumes that the attacker has the capability to choose arbitrary plaintexts to be encrypted and obtain the corresponding ciphertexts, i.e. that he has

access to the encryption device. Though seemingly unrealistic at first sight, this is the typical type of attack against public-key encryption schemes (see Asymmetric Methods – Public-).

- Chosen-ciphertext: An attack on a cryptosystem in which the attacker has the possibility to decipher a ciphertext selected by himself.

The following methods are available to attackers:

- Brute-force methods

  These are generic methods used to determine the secret key. Basically, it means that all possibilities are systematically tried. In such brute-force attack, all keys are used one after the other until the plaintext obtained makes sense, in which case the operation is considered successful. The success of such an attack depends on the length of the key and the computing power available.

  Encryption systems with an insufficient key space do not meet standard security requirements. To be on the safe side, the number of operations needed to determine the key should be in the range of 10^{25}.

  When using Hash Functions[1], which are independent of the key, the attacker attempts to break the encryption scheme by way of a birthday attack[2]. This is an attack is against the collision resistance: the attacker attempts to find any two messages M and M' which produce identical h(M) and h(M') hash values. A hash function should create 160-bit outputs so that an attack requires 2^{80} operations to be successful. This corresponds to the security of an 80-bit key in a symmetric encryption method.

  Concerning public-key methods (see Asymmetric Methods – Public-), the form of attack is not clear. This method tries to break the underlying mathematical problems, also called "hard problems". For instance, judging by recent findings, a key length of 1024 should be used with the RSA algorithm (see Algorithms).

- Statistical methods

  These methods attempt to exploit the statistical structure of the plaintext by analyzing the frequency of characters and groups of characters in the encrypted text and matching the result with expected frequencies in the plaintext.

- Analytical methods

  These methods are specifically designed to break a particular cryptosystem by exploiting the system's weak points. The aim is to find an analysis method that allows to determine the key from the ciphertext (and the plaintext).

[1] Hash function: a function that calculates a fixed-length function value (the hash value) for any input string. These functions are used to generate the electronic counterpart of a fingerprint.

[2] The name goes back on the statistical phenomenon that the number of people in a group needed for the probability that any two of them have their birthday on the same day to be even is surprisingly small (23).

For symmetric encryption, a distinction cam be made between the following analytical methods:

- Differential cryptanalysis – Based on the study of how differences in an input can affect the resultant difference at the output, differential cryptanalysis refers to a set of techniques for tracing differences through the network of transformations, discovering where the cipher exhibits non-random behavior, and exploiting such properties to recover the secret key.
- Linear cryptanalysis – This form of attack is based on finding and exploiting simple ("linear") dependencies between the bits of a plaintext and the encrypted text in order to obtain information on the key.

Modern algorithms usually try to protect themselves against both forms of attack.

4 Mathematical Foundations of Cryptography

The basis of cryptography is modular arithmetic. In school, children learn modular arithmetic in exercises such as "Peter is due at home at 13.00, but he is 12 hours late. At what time did he come home?" In this "clock arithmetic", there is no 25.00, but counting restarts at 1.00. Mathematically, this is 25 modulo 24 and the result is 1, or in mathematical notation:

$$25 \equiv 1 \bmod 24$$

In number theory, two integers *a* and *b* are said to be "congruent modulo *m*" (where *m* is a positive integer), when their difference (*a*-*b*) is integrally divisible by the number *m*. Or in other words, two numbers are congruent modulo *m* if they produce the same remainder when divided by *m*.

The set of all integers congruent to *a* (modulo *m*) is called the remainder class of *a* modulo *m. m* is called modulus.

For modulus *m*, the set of integers {0, 1, 2, …, n-1} is called Z_m. Together with addition modulo *m*, this set forms a mathematical structure formally called group.

The exact mathematical definition is:

A group is a mathematically abstract object consisting of a non-empty set G together with an associative operation ° that is based on a pair of elements from the set.
A number of properties and definitions apply to a group.

One of advantages of modular arithmetic is that the number of integers to be considered is finite. In particular the exponentiation of elements, widely used in cryptography, benefits from this property. Cryptography requires a so-called long integer arithmetic, i.e. an arithmetic with the ability to compute very long numbers that exceed a computer's display range.

Mathematical problems typically associated with asymmetric cryptography include:

- The knapsack problem – Having a backpack (knapsack) with a specific mass capacity as well as an unspecific number of objects of different masses, the question is: Which objects must be selected to fill the backpack in an optimal way?

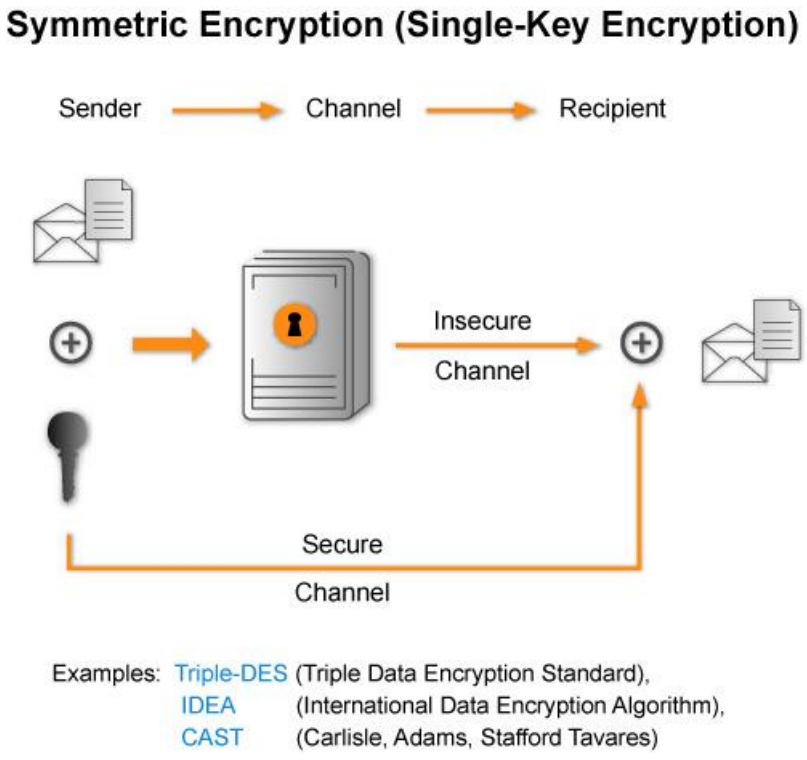
- The integer factorization problem – This problem relates to the fact that while finding and multiplying two large primes (100 decimals or more) together is very easy, it is much more difficult to retrieve them from the large composite number. As the best-known cryptographic method, the RSA algorithm is based on that problem.
- The problem of the discrete algorithm – The discrete algorithm (DL) problem relates to the attempt to compute the inverse of exponentiation, i.e. to calculate the value $x = \log_a y$ for any given $y = g^x$ where g is known and x is “large”.

5 Encryption Methods

5.1 Symmetric Methods – Secret-Key Cryptography

5.1.1 General

Symmetric encryption methods are those where only one key (the private or secret key), is used for both encryption and decryption. The CAESAR cipher mentioned above is a symmetric method. Anyone in possession of this key can both encrypt data and decrypt data that was encrypted with this key. The main problem with this method is how to agree and to exchange the key between the communication partners in a secure way.



This method is typically used when information is to be made available to several users in parallel (at the same time).

In “one-to-one” communication, the management of this method is rather complicated. To ensure that a user can communicate with *n* other users (assuming that, with *n+1* users involved, user A can communicate with *n* people, who in turn can send only to A) while at the same time preventing third parties from decrypting the sent data, *n* keys must be available. This means that each user must have and manage a large number of keys; as many keys as there are communication partners, to be precise.

Now, if communication is extended to *n* users (i.e. every user can communicate with every other user) while still ensuring that no user other than the sender and the recipient of a message can read that message, *n* times *(n-1) / 2* keys are needed with the symmetric encryption method. For 1000 users, this would amount to 499,500 keys.

5.1.2 Classical Cipher Systems

In the classical sense, all cipher systems rely on two simple basic principles, the transposition and the substitution.

With the transposition method, the characters of a plaintext are interchanged according to a specific scheme (permutations). On the other side, substitution means that characters or character strings are replaced with other characters or character strings. Repeatedly alternating and applying transposition and substitution schemes yields a product cipher.

The substitution systems can be classified as follows:

- Mono-alphabetic substitution: Each character or character string of the plaintext alphabet is replaced with a predetermined character or character string from a cipher alphabet B. The CAESAR cipher mentioned above is a typical example of that kind of system.
- Poly-alphabetic substitution: Each character or character string of the plaintext alphabet is replaced with a predetermined character or character string from a set of a cipher alphabets B1,.., Bn. An example of this substitution method is the Vernam cipher (France, 1917) or “One-time Pad” where the key is a random “pad” exactly the same length as the plaintext and used only once. If a “truly random sequence” is selected to generate the key, i.e. by tossing a coin or picking random numbers, the system is 100 percent unbreakable – and this can be mathematically proved.
- Monographic substitution: With this method, individual characters are replaced with other individual characters. The CAESAR cipher is an example of a mono-alphabetic and monographic substitution.
- Polygraphic substitution: With this method, character strings are replaced with other character strings.

5.1.3 Stream and Block Ciphers

As symmetric encryption methods usually require a fixed-length input, longer messages have to be split before they can be encrypted. There are two categories of symmetric algorithms:

1. Stream algorithms or stream ciphers – The plaintext is processed character by characters, where a character may be a bit or a byte. If implemented on the hardware side, these methods are faster than block-oriented ones, as bit-by-bit or byte-by-byte processing is better adapted to a hardware solution.
2. Block algorithms or block ciphers – The plaintext is processed in groups of bits, called blocks. These algorithms are typically used in software-based systems. However, even when using block ciphers, it makes sense that the encryption of a particular block also depends on how preceding blocks were encrypted. This is to make sure that identical plaintext blocks are mapped to different ciphertexts in order to make as hard as possible for attackers to break the encryption.

The chaining of consecutive blocks is also called cryptographic mode, variants of which are ECB (Electronic Codebook), CBC (Cipher Block Chaining), CFB (Cipher Feedback) or OFB (Output Feedback).

The creation of the secret key is performed by way of a bit sequence generated by a random generator, which acts as key.

5.1.4 Symmetric Algorithms

- DES (Data Encryption Standard)

  DES is the classical cryptographic technique. It was developed by IBM and presented in 1976 following a call for tenders put out by the National Bureau of Standards (NBS, today NIST). The DES algorithm is certified every 5 years by National Institute of Standards and Technology (NIST). The last certification dates back to 1999, however based on the Triple DES variant, as DES no longer meets today's security requirements.

  DES is a symmetric block cipher, which encrypts data in blocks of 64 bits. The algorithm transforms a 64-bit block of plaintext into a 64-bit block of ciphertext. The key length is 56 bit. With every eighth bit used as parity bit, the key is expressed as ordinary 64-bit number.

  DES works on a 64-bit block of the plaintext. After an initial permutation, this block is broken into two 32-bit halves. This is followed by 16 rounds of identical operations during which the data is combined with the key. After round 16, the two halves are joined again. The algorithm is completed with the inverse of the initial permutation.

  The DES key space comprises 2^{56} different keys only, some of which are known to be weak. With linear cryptanalysis, it is possible to start a known-plaintext attack[3] with 2^{47} known

[3] **Attacker knows plaintext and corresponding ciphertext or several pairs of text and attempts to find out the key used.**

plaintexts. Using differential cryptanalysis methods, single-DES complexity is reduced from 2^{56} to 2^{47}. Thus, with an effective key length of 56 bit, DES can no longer be considered secure.

- Triple DES

  Triple DES is a variation on DES where the DES algorithm is applied three times with different keys. This exploits the fact that DES is not a mathematical group, so that applying the three keys results in a different key space. Triple encryption with two keys works as follows:

  A block is first encrypted with the first key, then decrypted with the second key and finally re-encrypted with the first key. This technique, called encrypt-decrypt-encrypt (EDE) mode, was modified to improve DES for the X9.17 and ISO 8732 standards. Among others, Triple DES is currently used to compute the new EC-card PIN number and for the HCBI online banking standard.

- AES

  The National Institute of Standards and Technology (NIST) is planning to define a DES successor a new standard for symmetric encryption, the Advanced Encryption Standard (AES).

  AES must meet the following requirements:

  - A symmetric cryptographic method
  - A block cipher
  - A block size of 128 bit
  - A key size of 128, 192 and 256 bit

  The remaining contenders in the final round are MARS, RC6, Rijndael, Serpent and Twofish.

- CAST

  Named after its developers Carlisle Adams and Stafford Tavares, CAST was registered for patent on 23 April 1996. The symmetric block cipher CAST is a Feistel network[4]. The block length is a 64-bit block cipher, with a key length of 40 - 128 bit. Its main advantage compared to DES is the lack of weak keys.

  CAST is used in PGPhone. Northern Telecom, IBM, Tandem and Microsoft also use CAST in their products. CAST is the standard cipher in PGP.

[4] A far from negligible ciphering problem is that the encryption function has to be reversible to enable correct decryption of encrypted text. Feistel ciphers meet this requirement through a specific block cipher design.

- Blowfish

  Developed by the known cryptographer Bruce Schneider, this technique was introduced in December 1993. Blowfish is a symmetric encryption technique with a Feistel network. The block length is a 64-bit block cipher, with a key length of 8 - 448 bit. No attack with practical consequences has been recorded to date. Due to certain modifications and its license-free availability, Blowfish is very widely used. With its high storage requirement, the algorithm is neither suited for smart cards, nor for applications where the key has to be frequently replaced. Blowfish is used in PGPhone and Nautilus, two programs for secure telephony.

- IDEA (International Data Encryption Algorithm)

  The IDEA algorithm was developed by ETH Zürich in early 90s.. The patent to IDEA is held by ASCOM, a Swiss company.

  Due to its key length, IDEA is immune against brute-force attacks and – not less dangerous – cryptanalysis. IDEA is best known as algorithm in the encryption program PGP.

5.2 Asymmetric Methods – Public-Key Cryptography

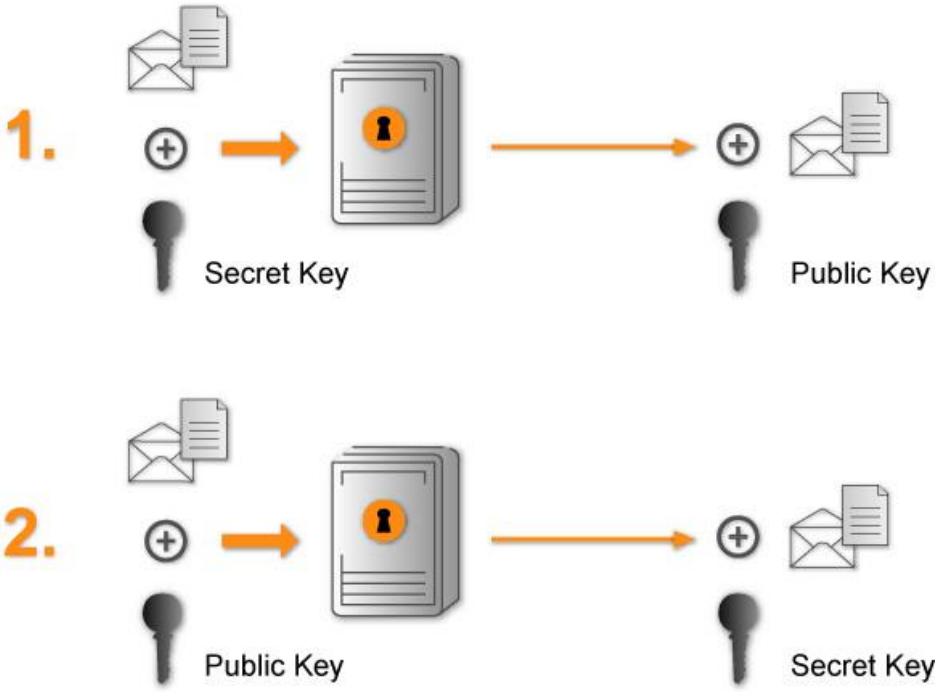
5.2.1 General

Asymmetric methods offer highly efficient options for safeguarding data. Asymmetric encryption methods involve two keys. Mathematically, both keys depend on each other, but none of them can (reasonably) be recovered from the other.

Both keys can be used to encrypt a message, but only the “other” one can be used for decryption. This technique is also called “private and public key cryptography”.

These features make it possible to publish one of the keys – the so-called public key. The private key is stored at a secure place and kept secret at all times. Called public-key methods, such methods are independent of the security or insecurity of the transmission path used. Anyone using public-key methods has his own pair of keys, consisting of a private (secret) key and a corresponding public key.

Options for Asymmetric Encryption



If a communication partner is to receive an encrypted message, his public key is used for encryption. Thus, the recipient is the only person who can decrypt the message (with his private key). The integrity and authenticity of a message, i.e. evidence on the ownership of a specific key, is achieved by encrypting a message with one's private key. By decrypting the message with the corresponding public key, anyone can check whether or not the message really is from the presumed sender. If decryption fails, the message has either been manipulated (data integrity violated) or the sender is not the one he claims to be (authenticity violated).

Encrypting a document with one's own private key validates that document just like a physical signature on paper – that's why this is called the digital signature.

As a consequence, anyone is able to encrypt data, but only the person owning the corresponding counterpart hat is in a position to decrypt the message. Normally, this will be exactly one person, the intended recipient (typically the person who has also provided the public key).

In case *n* persons want to communicate with each other, exactly *n* pairs of asymmetric keys are needed. From these *n x 2* keys, exactly *n*, i.e. the "public keys", are made public.

Within a network, the significant advantage of this method is that the public key of each user has to be published only once. All users should be given access to that key in order to be able to send encrypted information to the owner of the public key.

This also applies to new users, who can also send encrypted information to the owner of a public key of the key without the latter having to make a new key available. This means that a recipient may very well receive encrypted mail from unknown senders.

With symmetric encryption, by contrast, only users who have first exchanged a key can send encrypted messages to each other.

In practice, public key algorithms are not a substitute for symmetric algorithms, as they are much slower. That is why they are used, for instance, to encrypt keys for symmetric methods. This kind of procedure is then called hybrid method (see Hybrid Methods).

5.2.2 Asymmetric Encryption Algorithms

There is a wide variety of asymmetric encryption algorithms, some of which are shortly presented below.

- RSA

  RSA is the best known asymmetric encryption method. Named after and developed by Ron Rivest, Adi Shamir and Leonard Adleman, RSA made its appearance in 1976.

  RSA is an asymmetric algorithm based on the IF problem[5]. To create the two keys, two large primes *p* and *q* are randomly selected. To ensure maximum security, these primes should have approximately the same length. The product $n = p \times q$ is calculated.

  Then a cipher key *e* is randomly selected, where *e* is relative prime to (p-1)(q-1). Then the decryption key *d* is calculated.

  The public key then consists of *e* and *n*, while the private key consists of *d*. Encryption then has the form

  $$c \equiv m^e \bmod n$$

  and decryption the form

  $$m \equiv c^d \bmod n.$$

  To that effect, the message *m* is broken into blocks m_i that are smaller than *n*. The encrypted message *c* will then consist of message blocks c_i of the same size.

  It should be noted that the parameters *p*, *q* and *e* have to selected “as randomly as possible” and that *p* and *q* should be kept secret to make encryption secure.

  The security could neither be proved nor refuted by cryptanalysis. The security is based on the great difficulty of factoring large numbers. The public and private keys depend on a pair of large primes (over 150 digits). Though frequently used, it becomes more and more apparent that 512-bit RSA encryption is no longer secure. But the problem with more powerful RSA encryption (2048 bit or higher) is that it is difficult or impossible to implement in smartcards.

[5] Integer Factoring: Factoring is the problem of determining the prime factors for any given number. The problem underlying the RSA algorithm relates to the case where n=pxq (i.e. n is the product of two primes, p and q).

- DLIES

  DLIES is an asymmetric encryption method involving a key exchange procedure as well as a MAC[6] (also refer to Hash Functions) and based on a Diffie-Hellman problem[7]. Its previous name was DHAES, but that was changed due to a collision of names with the DES successor AES.

  When two partners want to send each other messages, a secret g^{uv} is shared between them. Together with one of the partners' public key g^u, this secret is compressed using a hash function[8]. This hash value is split. One part goes into the MAC. The other, together with the message M, is encrypted by a symmetric algorithm to the ciphertext SYM(M), the second input for the MAC. After applying the MAC, the TAG is produced as output. The data sent from one partner to the other consist of the public key g^u, the ciphertext SYM(M) and the tag.

  The security of DLIES is based on appropriate assumptions about the hardness of the Diffie-Hellman problem and the assumption that the underlying symmetric algorithm, the hash function as well as the MAC are secure.

- EC-IES

  ES-IES is a variation of the DLIES scheme, where elliptic curves are used. It is described in the IEEE standard.

5.3 Hybrid Methods

Hybrid methods are a combination of symmetric and asymmetric methods. They exploit the advantages of both methods, i.e. the speed of symmetric cryptography and the security of asymmetric cryptography. That is why they are often used when large amounts of data have to be encrypted.

The message is symmetrically encrypted with a so-called session key that is used only once. This session key is then asymmetrically encrypted with the recipient's public key and attached to the message. With is private key, the recipient is able to decrypt the key and then the message. For "long" messages, this procedure is more efficient than asymmetric encryption of the entire message. As the session key is used only once, this method is just as secure as asymmetric encryption.

---

[6] A Message Authentication Code (MAC) uses a secret code to add specific redundant information to a message. This information is stored or transmitted together with the message in order to provide authentication of the message, also refer to Authentication Codes (MAC).

[7] Diffie-Hellman (DH): see Key Exchange Protocols.

[8] Hash function: a function that calculates a fixed-length function value (the hash value) for any input string. These functions are used to generate the electronic counterpart of a fingerprint.

6 Protocols and Functions

6.1 Key Exchange Protocols

One of the major problems with symmetric encryption is the fact that both partners must have the same secret key. This secret key needs to be exchanges before any encrypted message is sent.

The following protocols are used to exchange a key while keeping it secret from anyone else:

- Diffie-Hellman key exchange

  Of all key exchange protocols, this is the best known one. It can be used to generate and distribute keys, but nor encryption or decryption. It is based on the discrete algorithm problem. Both partners agree on an appropriate group, in which the DL (discrete logarithm) problem can not be reasonably solved. In addition, they select a fixed group element g, which may be public (as the group itself):

  - Partner A selects a random integer x, computes gx and sends the result to Partner B
  - Partner B selects a random integer y, computes gy and sends the result to Partner A
  - Partner A computes (gy)x = gyx
  - Partner B computes (gx)y = gxy

  Now g^{yx} is the common secret key for symmetric encryption.

  This protocol is considered an asymmetric method, as the values g^x and g^y could be regarded as public keys and x and y as the corresponding private keys.

  The security of the Diffie-Hellman algorithm depends on the security of the DL problem[9] in the selected group. If implemented in Z_P, a large enough value for p should be selected (at least 128 bit).

- EC-DH

  EC-DH is a variation of the Diffie-Hellman key exchange protocol based on elliptic curves.

- MQV

  The Menezes-Qu-Vanstone key exchange protocol is an extension of the Diffie-Hellman protocol. The problem with the latter is that identical keys are generated in each session. Protocols such as MQV were developed to solve that problem and protect oneself against attacks such as “Man-in-the-Middle” attacks[10]. Its advantage is the implicit authentication of

---

[9] Discrete Logarithm (DL), see Mathematical Foundations of Cryptograph

[10] Continuation of server spoofing. In a man-in-the-middle attack, the attacker gets between two communicating computers. From there, he is able to intercept and modify data. If he receives the data at all, the recipient will believe to have received the data from the original sender.

the communication partners. Furthermore, the generation of the session keys involves random elements to ensure that different keys are used in different sessions.

- EC-MQV

  EC-MQV is described in the IEEE standard. It is the variation of the Menezes-Qu-Vanstone protocol based on elliptic curves.

6.2 Hash Functions

In general, it cannot be safely assumed that encrypting a message automatically guarantees its authenticity and integrity.

Hash functions are often used to check the integrity of data and MACs are used to ensure their authenticity.

A hash function compresses data in a specific but virtually irreversible way, unless the term of the one-way function is known. A one-way function is a function f that can be computed quickly but for which it is virtually impossible to determine an argument x for a function value y where f(x)=y.

A cryptographic hash function is an algorithm that creates a manipulation-proof check value (fingerprint) for a given message. The only parameter of the hash function is the message itself, i.e. no key is used. A typical example is the electronic signature, where the process is based on the data‘s hash value and not on the entire data.

To meet today’s requirements, a hash value should be at least 160 bit long.

Other than that, the hash function should have the following properties:

1. For a given M, the hash value h(M) is “easy” to compute.
2. It is virtually impossible to find any two messages M and M’ with identical hash values h(M) and h(M’).

A hash function with these properties is called collision-resistant or collision-free.

The following are possible attacks against hash functions:

1. For a given hash value, finding a message with the same hash value.
2. Finding two messages with identical hash values.

For instance, to attack a 160-bit hash value requires approximately 2^{80} operations to create collisions.

6.2.1 Construction Principles and Algorithms

There are different methods to construct a hash function. A method often used is the following one:

- The message is broken into a sequence of blocks of equal size and appropriate length. Where required, the last block is “padded” (filled) according to specific scheme.
- The algorithm is started with a given fixed initialization value IV (Initial Value).
- The message blocks are processed one after the other by using them as input for a compression function. Each value returned by this function is the initialization value for the next block.
- The hash value is the function’s last output value.

The compression function is applied several times.

This results in a class of hash functions developed for 32-bit architectures. The first one was MD4, followed by MD5, RIPEMD-128, RIPEMD-160, SHA-0 and SHA-1.

With a final output of 128 bit, approx. 2^{64} operations are needed to construct two different messages with the same hash value – or 2^{128}-1 operations to determine a matching message for a given hash value.

6.2.2 Algorithms

- **MD2**

  A hash function developed by Ron Divest in 1989. MD stands for Message Digest and was optimized for 8-bit architectures.

- **MD4**

  Developed by R.L. Rivest in 1990, MD4 is the predecessor if numerous hash functions. However, using MD4 is no longer recommended, as its vulnerability is known and proven by experience.

- **MD5**

  This further development of MD4 was introduced by R.L. Rivest in 1991; it is used, among others, in PGP.

- **RIPEMD-128**

  RIPEMD has a design similar to MD4. For RIPEMD-128, it is not impossible to find collision functions and the birthday attack is also possible, though at high cost. Therefore, it is not recommended to use this algorithm.

- **RIPEMD-160**

  RIPEMD-160 is the further development of RIPEMD-128 with a hash value of 160 bit. There are no known weak points ands this hash function is generally recommended.

- **SHA-0**

  The Secure Hash Algorithm (SHA-0) was designed by the National Security Agency (NSA). It was developed as hash algorithm for the signature algorithm DSA.

- **SHA-1**

  Further development of SHA-0 and generally recommended, as there are no known weak points.

The following table shows an overview of hash functions, including bit length and number of runs within the algorithm.

Name	Bit length	Runs x steps per run
MD4	128	3 x 16
MD5	128	4 x 16
RIPEMD-128	128	4 x 16 twice (parallel)
RIPEMD-160	160	5 x 16 twice (parallel)
SHA-0	160	4 x 20
SHA-1	160	4 x20

6.3 Authentication Codes (MAC)

For the electronic authentication of data, special information is added to a message M. This information is computed from M using cryptographic methods and then saved/transmitted with the message. To ensure that no attacker will be able to modify this redundant information, it has to be protected.

If a secret key is used to determine the redundant information, one speaks of a Message Authentication Code (MAC). Thus, a MAC can also be called a hash function with an additional secret key.

The evidence of a message's integrity is based on the secrecy or integrity of the cryptographic key. The simplest case is a symmetric encryption of the hash value. The recipient needs to know the key, but he can also use this key to generate other messages with the same hash value.

A MAC can be generated from a hash function or a symmetric block cipher.

The following methods are available for MACs:

- MAC based on hash functions – The MAC is created by combining the key, the message and again the key and using the result as input for the hash function. The security of this kind of MAC depends on the secrecy of the key and the hash method used.
- CBC-MAC
  The construction of this MAC is based on very simple possibility, the encryption of a message with a block algorithm in CBS mode. The last cipher block is used as tag. The security of the CBS-MAC depends on the secrecy of the key and the symmetric method used.

6.4 Digital Signatures

Signing a document in electronic (digital) form is the equivalent to physically signing a document on paper signing a document available on paper.

The main issues center around the following questions:

1. Has the document been modified after having been created?
2. Does the document come from the expected sender?

There are legal aspects to be considered. For instance in Germany, the Signature Act and a complementary Signature Decree passed in 1997 set out, among others, the legal framework for counterfeit-proof digital signatures and the preservation of rights of electronic legal traffic participants.

There are also a number of regulatory initiatives under way at EU level concerning digital signatures.

Public key methods can be used to sign documents. When using asymmetric encryption methods, the public key is normally used for encryption and the private key for decryption. For digital signatures, the keys are used the other way around:

- Partner A encrypts the data with his private key and sends the signed document to Partner B.
- Partner B decrypts the document with Partner A's public key to check the signature's authenticity.

However, a drawback for large documents is that asymmetric encryption may take too much time. Therefore, in practice, it is not the entire document that is encrypted, but only the hash value. This is a similar situation as for hybrid methods.

The digital signature now works as follows:

- Partner A encrypts the hash value of his document with his private key; this is the electronic signature.
- Partner A sends the document and the signed hash value to Partner B.
- Partner B uses the same hash function to compute the hash value of the document sent by Partner A. Using Partner A's public key and the algorithm for electronic signatures, he decrypts the signed hash value. If this value matches the hash value computed by himself, the signature (and the document) can be considered authentic.

As any modification is automatically detected, hash functions effectively solve the problem of counterfeiting.

A person's identity is guaranteed by his public key, provided it has been confirmed by a certification authority.

6.4.1 Signature Methods

- **DSA**

  DSA is a variation of the signature algorithms proposed by Schnorr and ElGamal. DSA is based on the Discrete Algorithm problem in finite groups.

  DSA uses the following parameters:

  p Prime, 512 to 1024 bit long (in 64-bit steps)

  q Prime, 160-bit long factor of p-1

  g Generator of order q

  x Private key, any random number smaller than q

  $y = g^x \bmod p$ Public key

  The first three parameter (p, q, g) as well as the public key (y) are publicly known, while the private key (x) must be kept secret. Let us also assume:

  m Message

  H Hash function

  To create and verify signatures with DSA, the following equations are essential:

  (1) $r = (g^k \bmod p) \bmod q$ k, randomly selected

  (2) $sk = H(m) + xr \pmod q$

  The underlying idea is that the owner of the signature is able to determine r and s. The recipient verifies the signature by checking whether or not (2) is true. The ransom number k is to be considered a temporary key that has to be kept secret. Knowledge of this secret is needed to be able to determine s from equation (2), which is necessary to create a correct signature.

With 512 bit, DSA is not powerful enough to ensure the level of security actually needed, but it is so with 1024 bit. The security corresponds to RSA with comparable parameters. The security of DSA is based on two different but related problems: (a) the general DL problem in Z_p (for which sub-exponential forms of attack exist, similarly to the Factoring problem), and (b) the DL problem in the sub-group of order q, created by g. To have a chance of success, the best known attacks need operations in the range of $\sqrt{q}$.

- **RSA**

  RSA cannot only be used for encryption but also for digital signatures. The algorithm is similar to the one for encryption (also refer to Asymmetric Encryption Algorithm), but with roles reversed: The own private key is used to sign a document, while the recipient uses the corresponding public key to check the signature.

  As mentioned before, it is usually not the entire document that is signed, but only the hash value. It has to be made sure that the value produced by the hash function (i.e. to be signed) must have an appropriate. Where necessary, the hash value will have to be “padded” to the required length.

- **EC-DSA**

  EC-DSA is the variation of DSA based on elliptic curves. Rather than working on a Z_p sub-group of order q, the underlying mathematical structure is an elliptic curve E.

  The security of this kind of cryptographic system is based on the difficulty of the DL problem in groups of points on an elliptic curve (EC-DLP).

- **NR**

  NR is a signature method based on the DL problem. The original algorithm supports message recovery, but this option is not foreseen in the protocol’s formulations included in the standards. The algorithm was developed by K. Nyberg and R. Rueppel, hence the acronym NR. It is a variation of the ElGamal signature scheme.

- **EC-NR**

  As for DSA, the original protocol developed by Nyberg and Rueppel can be based on elliptic curves.

About GBS

GROUP Business Software is a leading vendor of solutions and services in the fields of messaging security and workflow for the IBM and Microsoft collaboration platforms. Over 5,000 customers and more than 4 million users worldwide trust in GBS expertise. The company operates in Europe, North America and Asia.

Further information at www.gbs.com