Digital signature and correspodence between ship and shipping company with digital documents certificated by digital signature

Pantelakis Panagiotis
Athens, Greece
panagiotis.pantelakis@gmail.com

Pantelis Apostolos
Athens, Greece
panta1973@yahoo.com

***Abstract*—This electronic document is to present how works the digital signature and where to use a shipping company this function for shipping documents and the correspondence between ship and shipping company**

***Index Terms*—public, private, key, certificate, digital, signature, shipping, document.**

I. Introduction

In our days, maritime shipping undergoes rapid digitalization. This applies to safety and security reporting, mandatory ship documents, electronic port clearance as well as commercial and operational information exchanges. The move from paper and voice based communication to digital and automated information exchanges creates new requirements to authentication of document originator, verifiable integrity of messages as well as confidentiality when this is needed. Papers with stamps and signatures in sealed envelopes currently provide these mechanisms. In the future, new digital solutions are needed to maintain and increase the trust and accountability between parties.

Ships move internationally, and frequently encounter ports or port organizations that they have never met before. New security mechanisms must be internationally applied so that the required trust can be established without prior exchanges of user codes, passwords or similar user authentication data.

II. Chronology

Cryptography or cryptology, from Greek word “κρυπτός” which means "hidden, secret", is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it.

Cryptography is closely related to the disciplines of cryptology and crypt analysis. Cryptography includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. However, in today's computer centric world, cryptography is most often associated with scrambling plain text (ordinary text, sometimes referred to as clear text) into cipher text (a process called encryption), then back again (known as decryption). Individuals who practice this field are known as cryptographers.

The use of simple codes to protect information can be traced back to the fifth century BC. The first known use of a modern cipher was by Julius Caesar (100 BC to 44 BC), who did not trust his messengers when communicating with his governors and officers. For this reason, he created a system in which each character in his messages was replaced by a character three positions ahead of it in the Roman alphabet.

III. Introduction to Cryptography

As time has progressed, the methods by which information is protected have become more complex and more secure. Encryption can be used to provide high levels of security to network communication, e-mail, files stored on hard drives or floppy disks, and other information that requires protection.

Modern cryptography concerns itself with the following five objectives:

- Confidentiality (the information cannot be understood by anyone for whom it was unintended)
- Integrity (the information cannot be altered in storage or transit between sender and intended receiver without the alteration being detected)
- Non-repudiation (the creator/sender of the information cannot deny at a later stage his or her intentions in the creation or transmission of the information)
- Authentication (the sender and receiver can confirm each others identity and the origin/destination of the information)
- Key exchange: The method by which crypto keys are shared between sender and receiver.

A. Cryptographic Algorithms

There are several ways of classifying cryptographic algorithms. For purposes of this paper, they will be categorized based on the number of keys that are employed for encryption and decryption, and further defined by their application and use. The three types of algorithms that will be discussed are (Figure 1):

- Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption; also called symmetric encryption. Primarily used for privacy and confidentiality.

- Public Key Cryptography (PKC): Uses one key for encryption and another for decryption; also called asymmetric encryption. Primarily used for authentication, non-repudiation, and key exchange.
- Hash Functions: Uses a mathematical transformation to irreversibly "encrypt" information, providing a digital fingerprint. Primarily used for message integrity.

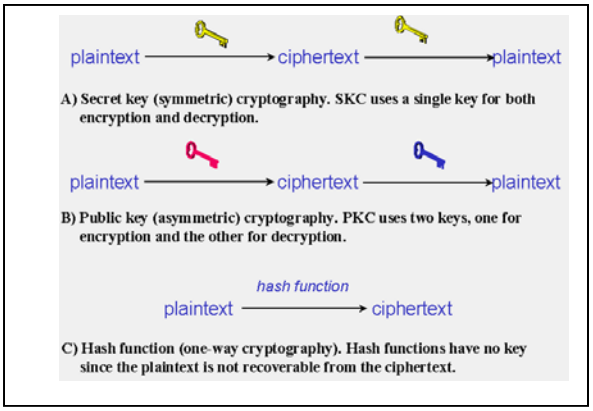


Fig. 1. Three types of cryptography: secret key, public key, and hash function.

1) *Secret Key Cryptography*

Secret key cryptography methods employ a single key for both encryption and decryption. As shown in Figure 1A, the sender uses the key to encrypt the plaintext and sends the ciphertext to the receiver. The receiver applies the same key to decrypt the message and recover the plaintext. Because a single key is used for both functions, secret key cryptography is also called symmetric encryption.

With this form of cryptography, it is obvious that the key must be known to both the sender and the receiver; that, in fact, is the secret. The biggest difficulty with this approach, of course, is the distribution of the key (more on that later in the discussion of public key cryptography).

Secret key cryptography schemes are generally categorized as being either stream ciphers or block ciphers.

Stream ciphers operate on a single bit (byte or computer word) at a time and implement some form of feedback mechanism so that the key is constantly changing.

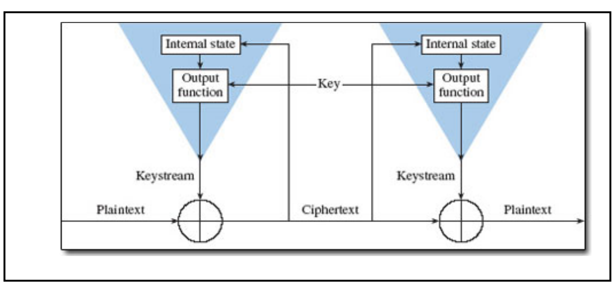


Fig. 2. Self-synchronizing stream cipher.

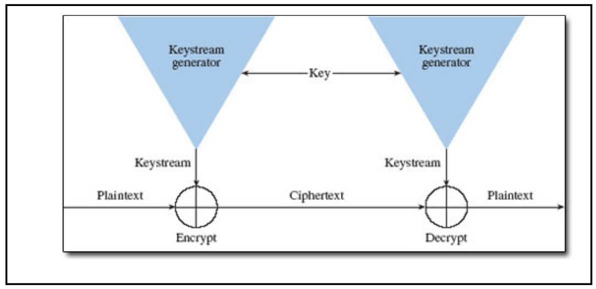


Fig. 3. Types of stream ciphers

Stream ciphers come in several flavors but two are worth mentioning here (Figure 2 and 3). Self-synchronizing stream cipherscalculate each bit in the keystream as a function of the previous n bits in the keystream. It is termed "self-synchronizing" because the decryption process can stay synchronized with the encryption process merely by knowing how far into the n-bit keystream it is. One problem is error propagation; a garbled bit in transmission will result in n garbled bits at the receiving side. Synchronous stream ciphers generate the keystream in a fashion independent of the message stream but by using the same keystream generation function at sender and receiver. While stream ciphers do not propagate transmission errors, they are, by their nature, periodic so that the keystream will eventually repeat.

A block cipher is so-called because the scheme encrypts one block of data at a time using the same key on each block. In general, the same plaintext block will always encrypt to the same ciphertext when using the same key in a block cipher whereas the same plaintext will encrypt to different ciphertext in a stream cipher. The most common construct for block encryption algorithms is the Feistel cipher, named for cryptographer Horst Feistel (IBM).

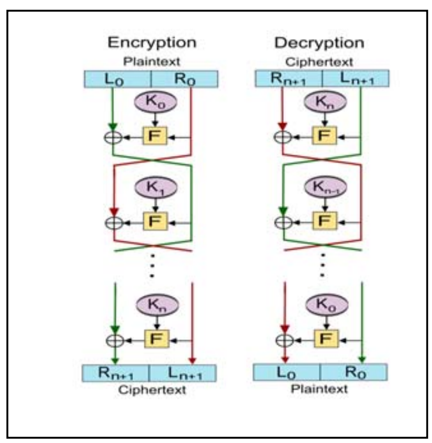


Fig. 4. Feistel cipher

As shown in Figure 4, a Feistel cipher combines elements of substitution, permutation (transposition), and key expansion; these features create a large amount of "confusion and diffusion" (per Claude Shannon) in the cipher. One advantage of the Feistel design is that the encryption and decryption stages are similar, sometimes identical, requiring only a reversal of the key operation, thus dramatically reducing the size of the code (software) or circuitry (hardware) necessary to implement the cipher. One of Feistel's early papers describing this operation is "Cryptography and Computer Privacy" (Scientific American, May 1973, 228(5), 15-23).

2) Public Key Cryptography

Public key cryptography has been said to be the most significant new development in cryptography in the last 300-400 years. Modern PKC was first described publicly by Stanford University professor Martin Hellman and graduate student Whitfield Diffie in 1976. Their paper described a two-key crypto system in which two parties could engage in a secure communication over a non-secure communications channel without having to share a secret key. Asymmetric algorithms are used for digital signatures and also for secure key exchange on unsafe communication channels.

PKC depends upon the existence of so-called one-way functions, or mathematical functions that are easy to compute whereas their inverse function is relatively difficult to compute. Let me give you two simple examples:

- Multiplication vs. factorization: Suppose you have two prime numbers, 3 and 7, and you need to calculate the product; it should take almost no time to calculate that value, which is 21. Now suppose, instead, that you have a number that is a product of two primes, 21, and you need to determine those prime factors. You will eventually come up with the solution but whereas calculating the product took milliseconds, factoring will take longer. The problem becomes much harder if we start with primes that have, say, 400 digits or so, because the product will have ~800 digits.

- Exponentiation vs. logarithms: Suppose you take the number 3 to the 6th power; again, it is relatively easy to calculate 36 = 729. But if you start with the number 729 and need to determine the two integers, x and y so that logx 729 = y, it will take longer to find the two values.

While the examples above are trivial, they do represent two of the functional pairs that are used with PKC; namely, the ease of multiplication and exponentiation versus the relative difficulty of factoring and calculating logarithms, respectively. The mathematical "trick" in PKC is to find a trap door in the one-way function so that the inverse calculation becomes easy given knowledge of some item of information.

Generic PKC employs two keys that are mathematically related although knowledge of one key does not allow someone to easily determine the other key. One key is used to encrypt the plaintext and the other key is used to decrypt the ciphertext. The important point here is that it does not matter which key is applied first, but that both keys are required for the process to work (Figure 4). Because a pair of keys is required, this approach is also called asymmetric cryptography.

Public key cryptography algorithms that are in use today for key exchange or digital signatures include:

- RSA: The first, and still most common, PKC implementation, named for the three MIT mathematicians who developed it — Ronald Rivest, Adi Shamir, and Leonard Adleman. RSA today is used in hundreds of software products and can be used for key exchange, digital signatures, or encryption of small blocks of data. RSA uses a variable size encryption block and a variable size key. The key-pair is derived from a very large number, n, that is the product of two prime numbers chosen according to special rules; these primes may be 100 or more digits in length each, yielding an n with roughly twice as many digits as the prime factors. The public key information includes n and a derivative of one of the factors of n; an attacker cannot determine the prime factors of n (and, therefore, the private key) from this information alone and that is what makes the RSA algorithm so secure. (Some descriptions of PKC erroneously state that RSA's safety is due to the difficulty in factoring large prime numbers. In fact, large prime numbers, like small prime numbers, only have two factors!) The ability for computers to factor large numbers, and therefore attack schemes such as RSA, is rapidly improving and systems today can find the prime factors of numbers with more than 200 digits. Nevertheless, if a large number is created from two prime factors that are roughly the same size, there is no known factorization algorithm that will solve the problem in a reasonable amount of time; a 2005 test to factor a 200-digit number took 1.5 years and over 50 years of compute time (see the Wikipedia article on integer factorization.) Regardless, one presumed protection of RSA is that users can easily increase the key size to always stay ahead of the computer processing curve. As an aside, the patent for RSA expired in September 2000 which does not appear to have affected RSA's popularity one way or the other.

- Diffie-Hellman: After the RSA algorithm was published, Diffie and Hellman came up with their own algorithm. D-H is used for secret-key key exchange only, and not for authentication or digital signatures.

- Digital Signature Algorithm (DSA): The algorithm specified in NIST's Digital Signature Standard (DSS), provides digital signature capability for the authentication of messages. Described in FIPS 186-4.

- ElGamal: Designed by Taher Elgamal, a PKC system similar to Diffie-Hellman and used for key exchange.

- Elliptic Curve Cryptography (ECC): A PKC algorithm based upon elliptic curves. ECC can offer levels of security with small keys comparable to RSA and other PKC methods. It was designed for devices with limited compute power and/or memory, such as smartcards and PDAs. More detail about ECC can be found below in Section 5.8. Other references include the Elliptic Curve Cryptography page and the Online ECC Tutorial page, both from Certicom. See also RFC 6090 for a review of fundamental ECC algorithms and The Elliptic Curve Digital Signature Algorithm (ECDSA) for details about the use of ECC for digital signatures.

- Public Key Cryptography Standards (PKCS): A set of interoperable standards and guidelines for public key cryptography, designed by RSA Data Security Inc.

PKCS #1: RSA Cryptography Standard (Also RFC 8017)
PKCS #2: Incorporated into PKCS #1.
PKCS #3: Diffie-Hellman Key-Agreement Standard
PKCS #4: Incorporated into PKCS #1.
PKCS #5: Password-Based Cryptography Standard (PKCS #5 V2.1 is also RFC 8018)
PKCS #6: Extended-Certificate Syntax Standard (being phased out in favor of X.509v3)
PKCS #7: Cryptographic Message Syntax Standard (Also RFC 2315)
PKCS #8: Private-Key Information Syntax Standard (Also RFC 5208)
PKCS #9: Selected Attribute Types (Also RFC 2985)
PKCS #10: Certification Request Syntax Standard (Also RFC 2986)
PKCS #11: Cryptographic Token Interface Standard
PKCS #12: Personal Information Exchange Syntax Standard (Also RFC 7292)
PKCS #13: Elliptic Curve Cryptography Standard
PKCS #14: Pseudorandom Number Generation Standard is no longer available
PKCS #15: Cryptographic Token Information Format Standard

- Cramer-Shoup: A public key cryptosystem proposed by R. Cramer and V. Shoup of IBM in 1998.

- Key Exchange Algorithm (KEA): A variation on Diffie-Hellman; proposed as the key exchange method for the NIST/NSA Capstone project.

- LUC: A public key cryptosystem designed by P.J. Smith and based on Lucas sequences. Can be used for encryption and signatures, using integer factoring.

- McEliece: A public key cryptosystem based on algebraic coding theory

3) Hash Functions

Hash functions, also called message digests and one-way encryption, are algorithms that, in essence, use no key (Figure 1C). Instead, a fixed-length hash value is computed based upon the plaintext that makes it impossible for either the contents or length of the plaintext to be recovered. Hash algorithms are typically used to provide a digital fingerprint of a file's contents, often used to ensure that the file has not been altered by an intruder or virus. Hash functions are also commonly employed by many operating systems to encrypt passwords. Hash functions, then, provide a measure of the integrity of a file.

Hashes are one-way encryption. You cannot take a hash and "decrypt" it to find the original string that created it. Suppose that you want to crack someone's password, where the hash of the password is stored on the server. Indeed, all you then need is a string that produces the correct hash and you're in! However, you cannot prove that you have discovered the user's password, only a "duplicate key."

Hash algorithms that are in common use today include:

Message Digest (MD) algorithms: A series of byte-oriented algorithms that produce a 128-bit hash value from an arbitrary-length message.

- MD2 (RFC 1319): Designed for systems with limited memory, such as smart cards. (MD2 has been relegated to historical status, per RFC 6149.)
- MD4 (RFC 1320): Developed by Rivest, similar to MD2 but designed specifically for fast processing in software. (MD4 has been relegated to historical status, per RFC 6150.)
- MD5 (RFC 1321): Also developed by Rivest after potential weaknesses were reported in MD4; this scheme is similar to MD4 but is slower because more manipulation is made to the original data. MD5 has been implemented in a large number of products although several weaknesses in the algorithm were demonstrated by German cryptographer Hans Dobbertin in 1996 ("Cryptanalysis of MD5 Compress").

Secure Hash Algorithm (SHA): Algorithm for NIST's Secure Hash Standard (SHS), described in FIPS 180-4 The status of NIST hash algorithms can be found on their " Policy on Hash Functions" page.

- SHA-1 produces a 160-bit hash value and was originally published as FIPS PUB 180-1 and RFC 3174. SHA-1 was deprecated by NIST as of the end of 2013 although it is still widely used.
- SHA-2, originally described in FIPS PUB 180-2 and eventually replaced by FIPS PUB 180-3 (and FIPS PUB 180-4), comprises five algorithms in the SHS: SHA-1 plus SHA-224, SHA-256, SHA-384, and SHA-512 which can produce hash values that are 224, 256, 384, or 512 bits in length, respectively. SHA-2 recommends use of SHA-1, SHA-224, and SHA-256 for messages less than 264 bits in length, and employs a 512 bit block size; SHA-384 and SHA-512 are recommended for messages less than 2128 bits in length, and employs a 1,024 bit block size. FIPS PUB 180-4 also introduces the concept of a truncated hash in SHA-512/t, a generic name referring to a hash value based upon the SHA-512 algorithm that has been truncated to t bits; SHA-512/224 and SHA-512/256 are specifically described. SHA-224, -256, -384, and -512 are also described in RFC 4634.
- SHA-3 is the current SHS algorithm. Although there had not been any successful attacks on SHA-2, NIST decided that having an alternative to SHA-2 using a different algorithm would be prudent. In 2007, they launched a SHA-3 Competition to find that alternative; a list of submissions can be found at The SHA-3 Zoo. In 2012, NIST announced that after reviewing 64 submissions, the winner was Keccak (pronounced "catch-ack"), a family of hash algorithms based on sponge functions. The NIST version can support hash output sizes of 256 and 512 bits.

- RIPEMD: A series of message digests that initially came from the RIPE (RACE Integrity Primitives Evaluation) project. RIPEMD-160 was designed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel, and optimized for 32-bit processors to replace the then-current 128-bit hash functions. Other versions include RIPEMD-256, RIPEMD-320, and RIPEMD-128.
- HAVAL (HAsh of VAriable Length): Designed by Y. Zheng, J. Pieprzyk and J. Seberry, a hash algorithm with many levels of security. HAVAL can create hash values that are 128, 160, 192, 224, or 256 bits in length. More details can be found in a AUSCRYPT '92 paper.
- Whirlpool: Designed by V. Rijmen (co-inventor of Rijndael) and P.S.L.M. Barreto, Whirlpool is one of two hash functions endorsed by the New European Schemes for Signatures, Integrity, and Encryption (NESSIE) competition (the other being SHA). Whirlpool operates on messages less than 2256 bits in length and produces a message digest of 512 bits. The design of this hash function is very different than that of MD5 and SHA-1, making it immune to the same attacks as on those hashes.
- Tiger: Designed by Ross Anderson and Eli Biham, Tiger is designed to be secure, run efficiently on 64-bit processors, and easily replace MD4, MD5, SHA and SHA-1 in other applications. Tiger/192 produces a 192-bit output and is compatible with 64-bit architectures; Tiger/128 and Tiger/160 produce a hash of length 128 and 160 bits, respectively, to provide compatibility with the other hash functions mentioned above.
- eD2k: Named for the EDonkey2000 Network (eD2K), the eD2k hash is a root hash of an MD4 hash list of a given file. A root hash is used on peer-to-peer file transfer networks, where a file is broken into chunks; each chunk has its own MD4 hash associated with it and the server maintains a file that contains the hash list of all of the chunks. The root hash is the hash of the hash list file.

A digression on hash collisions, Hash functions are sometimes misunderstood and some sources claim that no two files can have the same hash value. This is, in theory if not in fact, incorrect. Consider a hash function that provides a 128-bit hash value. There are, then, 2128possible hash values. But there are an infinite number of possible files and ∞ >> 2128. Therefore, there have to be multiple files — in fact, there have to be an infinite number of files! — That has the same 128-bit hash value. (Now, while even this is theoretically correct, it is not true in practice because hash algorithms are designed to work with a limited message size, as mentioned above. For example, SHA-1, SHA-224, and SHA-256 produce hash values that are 160, 224, and 256 bits in length, respectively, and limit the message length to less than 264 bits; SHA-384 and all SHA-256 variants limit the message length to less than 2128 bits. Nevertheless, hopefully you get my point.)

The difficulty is not necessarily in finding two files with the same hash, but in finding a second file that has the same hash value as a given first file. Consider this example. A human head has, generally, no more than ~150,000 hairs. Since there are more than 7 billion people on earth, we know that there are a lot of people with the same number of hairs on their heads. Finding two people with the same number of hairs, then, would be relatively simple. The harder problem is choosing one person (say, you, the reader) and then finding another person who has the same number of hairs on their head as you have on yours.

This is somewhat similar to the Birthday Problem. We know from probability that if you choose a random group of ~23 people, the probability is about 50% that two will share a birthday (the probability goes up to 99.9% with a group of 70 people). However, if you randomly select one person in a group of 23 and try to find a match to that person, the probability is only about 6% of finding a match; you'd need a group of 253 for a 50% probability of a shared birthday to one of the people chosen at random (and a group of more than 4,000 to obtain a 99.9% probability).

What is hard to do, then, is to try to create a file that matches a given hash value so as to force a hash value collision — which is the reason that hash functions are used extensively for information security and computer forensics applications. Alas, researchers in 2004 found that practical collision attacks could be launched on MD5, SHA-1, and other hash algorithms.

Certain extensions of hash functions are used for a variety of information security and digital forensics applications, such as:

- Hash libraries, aka hashsets, are sets of hash values corresponding to known files. A hashset containing the hash values of all files known to be a part of a given operating system, for example, could form a set of known good files, and could be ignored in an investigation for malware or other suspicious file, whereas as hash library of known child pornographic images could form a set of known bad files and be the target of such an investigation.
- Rolling hashes refer to a set of hash values that are computed based upon a fixed-length "sliding window" through the input. As an example, a hash value might be computed on bytes 1-10 of a file, then on bytes 2-11, 3-12, 4-13, etc.
- Fuzzy hashes are an area of intense research and represent hash values that represent two inputs that are similar. Fuzzy hashes are used to detect documents, images, or other files that are close to each other with respect to content. See "Fuzzy Hashing" (PDF) by Jesse Kornblum for a good treatment of this topic.

B. *What is a Certificate*

A certificate is a piece of information that proves the identity of a public-key's owner. Like a passport, a certificate provides recognized proof of a person's (or entity) identity.

Certificates are signed and delivered securely by a trusted third party entity called a Certificate Authority (CA).

CAs, a type of Trust Service Provider, are third-party organizations that have been widely accepted as reliable for ensuring key security and that can provide the necessary digital certificates. Both the entity sending the document and the recipient signing it must agree to use a given CA.

As long as Bob and Alice trust this third party, the CA, they can be assured that the keys belong to the persons they claim to be. A certificate contains among other things:

- The CA's identity
- The owner's identity
- The owner's public-key
- The certificate expiry date
- The CA's signature of that certificate
- Other information that is beyond the scope of this article.

With a certificate instead of a public-key, a recipient can now verify a few things about the issuer to make sure that the certificate is valid and belongs to the person claiming its ownership:

- Compare the owner's identity
- Verify that the certificate is still valid
- Verify that the certificate has been signed by a trusted CA
- Verify the issuer's certificate signature, hence making sure it has not been altered.

Bob can now verify Alice's certificate and be assured that it is Alice's private-key that has been used to sign the message. Alice must be careful with her private-key and must not divulge how to get to it; by doing so, she is enforcing one aspect of the non-repudiation feature associated with her digital signature. As will be seen in section 3.3, there is more to consider for effective non-repudiation support.

Note that certificates are signed by a CA, which means that they cannot be altered. In turn, the CA signature can be verified using that CA's certificate.

1) Certificate validation added to the process

When Alice encrypts a message for Bob, she uses Bob's certificate.

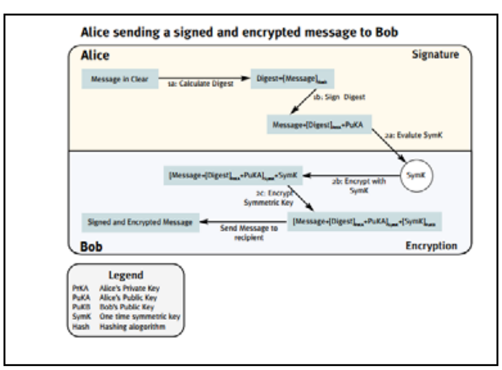


Fig. 5. Signature and Encryption details with keys

Prior to using the public-key included in Bob's certificate as shown in figure 4, some additional steps are performed to validate Bob's certificate:

- Validity period of Bob's certificate
- The certificate belongs to Bob
- Bob's certificate has not been altered
- Bob's certificate has been signed by a trusted CA

Additional steps would be required to validate the CA's certificate in the case where Alice does not trust Bob's CA. These steps are identical to the ones requires to validate Bob's certificate. In the example below, it is assumed that both Bob and Alice trust that CA.

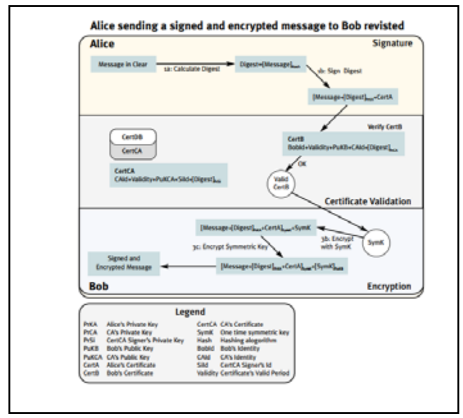


Fig. 6. Signature and Encryption details with certificates

In the Figure 5 above, a Certificate validation step is added to what is shown in Figure 4. Only the fields required for the validation of a certificate are displayed.

- Alice wants to make sure that the PuKB included in CertB belongs to Bob and is still valid.
- She checks the Id field and finds BobId, which is Bob's identity. In fact, the only thing she really knows is that this certificate appears to belong to Bob.
- She then checks the validity fields and finds that the current date and time is within the validity period. So far the certificate seems to belong to Bob and to be valid.
- The ultimate verification takes place by verifying CertB's signature using the CA's public key (PuKCA found in CertCA) 10. If CertB signature is ok, this means that:

a) Bob's certificate has been signed by the CA in which Alice and Bob has put all their trust.

b) Bob's certificate integrity is proven and has not been altered in any way.

c) Bob's identity is assured and the public-key included in the certificate is still valid and belongs to Bob. Therefore, Alice can encrypt the message and be assured that only Bob will be able to read it.

Similar steps will be performed by Bob on Alice's certificate before verifying Alice's signature.

C. *The Public Key Infrastructure*

A Public Key Infrastructure (PKI) is a combination of software and procedures providing a means for managing keys and certificates, and using them efficiently. Just recall the complexity of the operations described earlier in this article for having a feel on the absolute necessity to provide users with appropriate software support for encryption and digital signature. But nothing has been said yet about management.

1) *Key and certificate management*

Key and certificate management is the set of operations required to create and maintain keys and certificates. The following is the list of the major points being addressed in a managed PKI:

a) Key and certificate creation: How to generate key pairs? How to issue certificates to the users? A PKI must offer software support for key pair generation as well as certificate requests. In addition, procedures must be put in place to verify the user identity prior to allowing him to request a certificate.

b) Private-key protection: How will the user protect his private-key against misuse by other malicious users? Certificates are widely accessible because they are used for either encryption or signature verification. Private-keys require some reasonable level of protection because they are used either for decryption or for digital signature. A strong password mechanism must be part of the features of an effective PKI.

c) Certificate revocation: How to handle the situation where a user's private-key has been compromised? Similarly, how to handle the situation where an employee leaves the company? How to know whether or not a certificate has been revoked? A PKI must provide a means by which a certificate can be revoked. Once revoked, this certificate must be included in a revocation list that is available to all users. A mechanism must be provided to verify that revocation list and refuse to use a revoked certificate.

d) Key backup and recovery: What happens to encrypted files when a user loses his private-key? Without key backup, all messages and files that have been encrypted with his public-key can no longer be decrypted and are lost forever. A PKI must offer private-key backup and a private-key recovery mechanism such that the user can get back his private-key to be able to get access to his files11.

e) Key and certificate update: What happens when a certificate reaches or is near its expiry date? Keys and certificates have a finite lifetime. A PKI must offer a mechanism to at least update the expiry date for that certificate. Good practice though is to update the user's keys and certificates. The key and certificate update can be automatic in which case the end user gets notified that his keys have been updated, or can require that the user performs an action during or before his keys and certificates expire; if this case, the PKI must inform the user that this action is required prior the expiry time of his keys and certificates.

f) Key history management: After several key updates, how will a user decide which privatekey to use to decrypt files? Each key update operation generates new key pairs. Files that have been encrypted with previous public-keys can only be decrypted with their associated private-keys. Without key history management, the user would have to make decision on the key to use for decrypting files12. 11

g) Certificate access: How will a user, who wants to send a message to several recipients, get their certificates? A PKI must offer an easy and convenient way to make these certificates available. The use of an LDAP directory is commonly used for that purpose.

2) *Support for non repudiation of digital signature*

One important point that has to be clarified is non-repudiation of digital signature. This notion refers to the fact that a user cannot deny having signed a given message. This implies that the user who signed the message is the only one who has access to the private-key used for signing. However, as we have seen above, in a managed PKI, private-keys are kept by the CA for key recovery purposes. Therefore, both the user and the CA know that private-key, which means that both can (in theory) use that key for signing a message. A user can then deny having signed that message.

In order to avoid this situation and provide non-repudiation support, there must be a second key pair used for signature/verification purposes only. No backup is made for the signing private-keys, and only the user has access to it. In the case where the user loses his password, he loses his signing key as well. At key recovery time, the encryption/decryption key pair is given back to the user and a new signature/verification key pair is generated. This causes no problem since each time a user signs a document, the associated verification certificate is appended to it, so the document signature can always be verified at any time.

IV. DIGITAL SIGNATURE

Digital signatures are like electronic "fingerprints." In the form of a coded message, the digital signature securely associates a signer with a document in a recorded transaction. Digital signatures use a standard, accepted format, called Public Key Infrastructure (PKI), to provide the highest levels of security and universal acceptance. They are a specific signature technology implementation of electronic signature (eSignature).

A. *Importance of Digital Signature*

Out of all cryptographic primitives, the digital signature using public key cryptography is considered as very important and useful tool to achieve information security.

Apart from ability to provide non-repudiation of message, the digital signature also provides message authentication and data integrity. Let us briefly see how this is achieved by the digital signature

- Message authentication − When the verifier validates the digital signature using public key of a sender, he is assured that signature has been created only by sender who possess the corresponding secret private key and no one else.

- Data Integrity − In case an attacker has access to the data and modifies it, the digital signature verification at receiver end fails. The hash of modified data and the output provided by the verification algorithm will not match. Hence, receiver can safely deny the message assuming that data integrity has been breached.

- Non-repudiation − Since it is assumed that only the signer has the knowledge of the signature key, he can only create unique signature on a given data. Thus the receiver can present data and the digital signature to a third party as evidence if any dispute arises in the future.

By adding public-key encryption to digital signature scheme, we can create a cryptosystem that can provide the four essential elements of security namely − Privacy, Authentication, Integrity, and Non-repudiation.

B. Model of Digital Signature

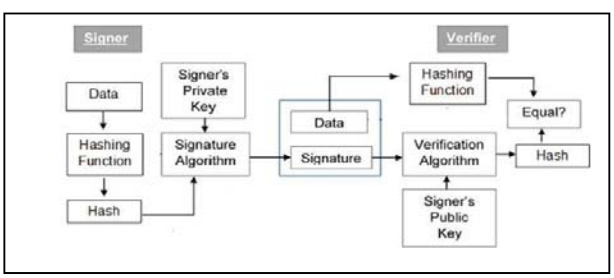
Digital signatures, like handwritten signatures, are unique to each signer. Digital signature solution providers, follow a specific protocol, called PKI. PKI requires the provider to use a mathematical algorithm to generate two long numbers, called keys. One key is public, and one key is private.

When a signer electronically signs a document, the signature is created using the signer's private key, which is always securely kept by the signer. The mathematical algorithm acts like a cipher, creating data matching the signed document, called a hash, and encrypting that data. The resulting encrypted data is the digital signature. The signature is also marked with the time that the document was signed. If the document changes after signing, the digital signature is invalidated.

To protect the integrity of the signature, PKI requires that the keys be created, conducted, and saved in a secure manner, and often requires the services of a reliable Certificate Authority (CA). Digital signature providers, meet PKI requirements for safe digital signing.

The model of digital signature scheme is depicted in the following illustration

Fig. 7. Digital signature scheme



The following points explain the entire process in detail:

- Each person adopting this scheme has a public-private key pair.

- Generally, the key pairs used for encryption/decryption and signing/verifying are different. The private key used for signing is referred to as the signature key and the public key as the verification key.

- Signer feeds data to the hash function and generates hash of data.

- Hash value and signature key are then fed to the signature algorithm which produces the digital signature on given hash. Signature is appended to the data and then both are sent to the verifier.

- Verifier feeds the digital signature and the verification key into the verification algorithm. The verification algorithm gives some value as output.

- Verifier also runs same hash function on received data to generate hash value.

- For verification, this hash value and output of verification algorithm are compared. Based on the comparison result, verifier decides whether the digital signature is valid.

- Since digital signature is created by 'private' key of signer and no one else can have this key; the signer cannot repudiate signing the data in future.

It should be noticed that instead of signing data directly by signing algorithm, usually a hash of data is created. Since the hash of data is a unique representation of data, it is sufficient to sign the hash in place of data. The most important reason of using hash instead of data directly for signing is efficiency of the scheme.

Let us assume RSA is used as the signing algorithm. As discussed in public key encryption chapter, the encryption/signing process using RSA involves modular exponentiation.

Signing large data through modular exponentiation is computationally expensive and time consuming. The hash of the data is a relatively small digest of the data, hence signing a hash is more efficient than signing the entire data.

C. Digital Certificate

A digital certificate is an electronic document issued by a Certificate Authority (CA). It contains the public key for a digital signature and specifies the identity associated with the key, such as the name of an organization. The certificate is used to confirm that the public key belongs to the specific organization. The CA acts as the guarantor. Digital certificates must be issued by a trusted authority and are only valid for a specified time. They are required in order to create a digital signature.

D. Difference Between a Digital Signature and an Electronic Signature

The broad category of electronic signatures (eSignatures) encompasses many types of electronic signatures. The category includes digital signatures, which are a specific technology implementation of electronic signatures. Both digital signatures and other eSignature solutions allow you to sign documents and authenticate the signer. However, there are differences in purpose, technical implementation, geographical use, and legal and cultural acceptance of digital signatures versus other types of eSignatures.

In particular, the use of digital signature technology for eSignatures varies significantly between countries that follow open, technology-neutral eSignature laws, including the United States, United Kingdom, Canada, and Australia, and those that follow tiered eSignature models that prefer locally defined standards that are based on digital signature technology, including many countries in the European Union, South America, and Asia. In addition, some industries also support specific standards that are based on digital signature technology.

E. Creation of a Digital Signature

eSignature providers that offer solutions based on digital signature technology, make it easy to digitally sign documents. They provide an interface for sending and signing documents online and work with the appropriate Certificate Authorities to provide trusted digital certificates.

Depending upon the Certificate Authority you are using, you may be required to supply specific information. There also may be restrictions and limitations on whom you send documents to for signing and the order in which you send them. When you receive a document for signing via email, you must authenticate as per the Certificate Authority's requirements and then "sign" the document by filling out a form online.

F. Reasons of Using a Digital Signature

Many industries and geographical regions have established eSignature standards that are based on digital signature technology, as well as specific certified CAs, for business documents. Following these local standards based on PKI technology and working with a trusted certificate authority can ensure the enforceability and acceptance of an e-signature solution in each local market. By using the PKI methodology, digital signatures utilize an international, well-understood, standards-based technology that also helps to prevent forgery or changes to the document after signing.

G. eSignatures Legality Based on Digital Dignature Technology

The EU passed the EU Directive for Electronic Signatures in 1999, and the United States passed the Electronic Signatures in Global and National Commerce Act (ESIGN) in 2000. Both acts made electronically signed contracts and documents legally binding, like paper-based contracts. Since then, the legality of electronic signatures has been upheld many times.

By now, most countries have adopted legislation and regulations modeled after the United States or the European Union, with a preference in many regions for the E.U. model of locally managed, digital signature technology-based eSignatures. In addition, many companies have improved compliance with the regulations established by their industries (e.g., FDA 21 CFR Part 11 in the Life Sciences industry), which has been achieved by using digital signature technology. These country- and industry-specific regulations are continuously evolving, a key example being the Electronic identification and trust services (eIDAS) regulation that was recently adopted in the European Union.

H. Example of a Digital Signature Work

Consider a scenario where Alice has to digitally sign a file or an email and send it to Bob (Figure 8):

- Alice selects the file to be digitally signed or clicks on 'sign' in her email application
- The hash value of the file content or the message is calculated by Alice's computer
- This hash value is encrypted with Alice's Signing Key (which is a Private Key) to create the Digital Signature.
- Now, the original file or email message along with its Digital Signature are sent to Bob.
- After Bob receives the signed message, the associated application (such as email application) identifies that the message has been signed. Bob's computer then proceeds to:
- Decrypt the Digital Signature using Alice's Public Key
- Calculate the hash of the original message
- Compare the (a) hash it has computed from the received message with the (b) decrypted hash received with Alice's message.
- Any difference in the hash values would reveal tampering of the message.

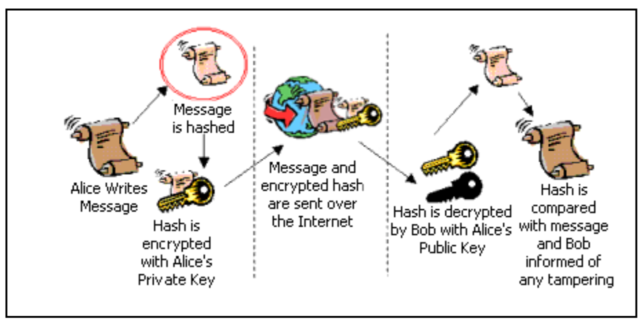


Fig. 8. Example of a digital sign

V. Digital Certificates – Certification – Certificate Authorities

A. Digital Certificates

From more we can conceive relatively easily that the operation of asymmetrical cryptographic system (Public Key) is efficient, when the sender knows in advance the Public Key of recipient. This make limits to the operations and the possibilities of safe communication. On one side, the user is not possible to assemble and upgrades all the Public Keys of users with which he wishes to communicate in his calculating system, on the other side it is not functional before a communication to be requested the Public Key of recipient by the sender. This process waste time from the communication and does not ensure the guarantee of identity of the user who is found .on the other side of connection and impersonates the legal recipient.

As likely solution in the first question if it is possible to be created a big online data base which are maintained all the Public Keys of the users. In any case where some user (sender) wants to send a secure message in some other user (recipient), then it will be connected in this base of data and receives the Public Key of the user or the organism that he wants to communicate.

The creation of this big online data base, even if it appears feasible theoretically, faces serious practical problems as in maintenance, in functionalism at the support of hundreds of millions of Public Keys as well as the dangerous monopolistic force that will acquire the organism that it will be developed.

In the second question, on which is based the main reflection of reader "Why is not possible a non certificated user to create his own Public Key and certificate itself as other sender " is presented a more practical solution, which is based on widely acquaintances and secure third entities. According to this solution in order to be avoided a problem of fake Public Key, the name and the Public Key of each user they are registered in a small document which is cryptographic signed from a widely official and confidential authority. The signature that is given and connects the name of user with his Public Key from the beginning, gives the secure against in the counterfeiting. This document is named certificate.

Each user that receives this certificate can:

- Be confirmed for the original signature of entrusting authority, therefore it confirms that the Public Key and the identity that are connected with this it has not been modified.

- Know that the information was encrypted only with the Public Key by the particular sender.

- Be confirmed each signature, as well as the integrity of each document where it was signed by the particular sender

The confidential factor which signs the certificates is named Certification Authority – CA.

So, the user maintains the Public Keys of wide famous and secure Certification Authorities in stead of the Public keys of particular users. The sender receives from the recipient's certificate, which has been signed by a Certification Authority. The sender is certified, by the certificate, for the identity of the recipient and receives the Public Key which with encrypt the message.

The certificates are separated in various categories depending on the use and the information that provide.

Examples of certificates are following:

- Server Certificates, a server is required to present a certificate as part of the initial connection setup.

- Client Certificate, are used to authenticate the client connecting to a TLS service, for instance to provide access control.

- Email Certificate, senders need to discover which public key to use for any given recipient, so they get this information from an email certificate.

- Intermediate Certificate is a certificate used to sign other certificates. An intermediate certificate must be signed by another intermediate certificate, or a root certificate.

These are some of the most common certificates which are defined by X.509. Because X.509 is very general, the format is further constrained by profiles defined for certain use cases, such as Public Key Infrastructure X.509 as defined in RFC 5280.

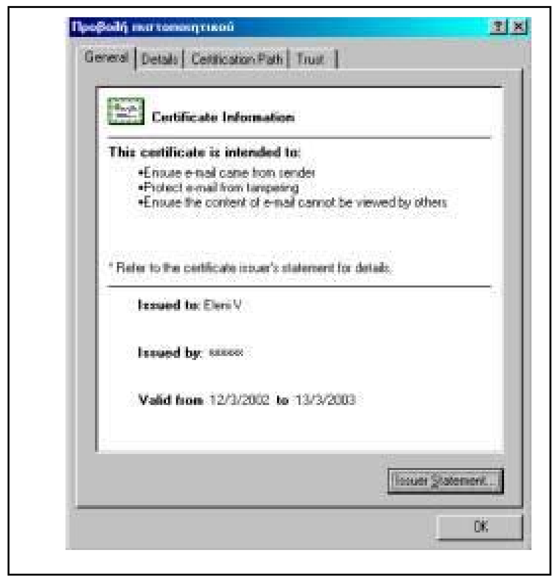


Fig. 9. Example of a digital certificate

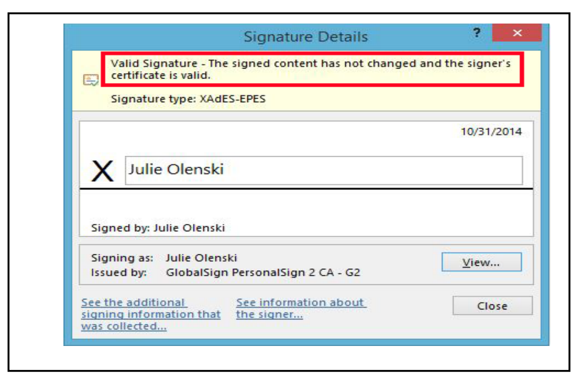


Fig. 10. Example of a digital signature details

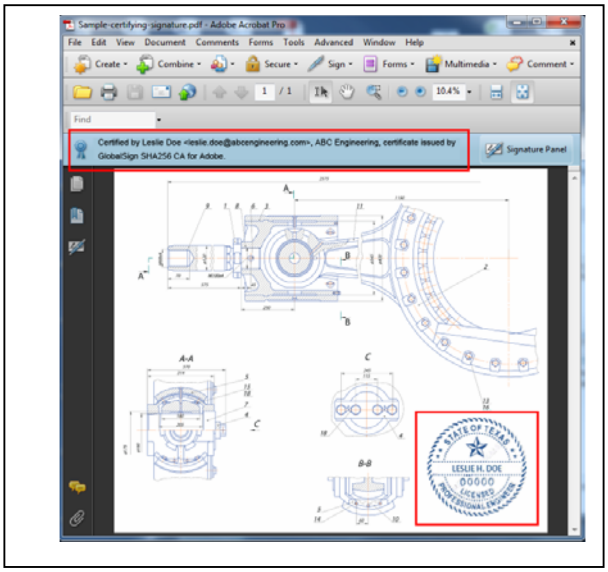


Fig. 11. *Digital signature in Microsoft Word*

B. *Certificate Authorities*

In cryptography, a certificate authority or certification authority (CA) is an entity that issues digital certificates. A digital certificate certifies the ownership of a public key by the named subject of the certificate. This allows others (relying parties) to rely upon signatures or on assertions made about the private key that corresponds to the certified public key. A CA acts as a trusted third party—trusted both by the subject (owner) of the certificate and by the party relying upon the certificate. The format of these certificates is specified by the X.509 standard.

For the issue of certificate in the sender, the Certificate Authority (CA) confirms the identity of the sender. This ratification becomes from an authority, this is Registration Authority - RA.

After the approval of the sender's identity, the Registration Authority creates a unique name (username) of the sender. This username, which contains the name that gave the sender - user, confirms and distinguishes between the users which are supported by this certificate, this particular user from some else who is resembled. The Certificate Authority after all creates the certificate in order to connect this username of the sender with the Public Key which is connected with the sender's Private Key.

Initially, a pair of Public and Private Key is created. The user supplements an application of certification, which contains the Public Key and the essential information of the identity and then sends to Certificate Authority.

Usually it takes time to the verification of elements from the Certificate Authority. The verification of elements is a subjective process for each Certificate Authority and depends on the principles and the rules of each one. Afterwards the verification of elements, the Certificate Authority creates a certificate that contains the Public Key of the user with elements from his identity (name, email address, address in the Web - URL etc.). Then, the Certificate Authority produces a summary message from the certificate and signs the result of this summary with its own Private Key. In this way was created a signed certificate which the Certificate Authority returns to the user.

The sender, in order to send a secure message, asks from the recipient to present a signed certificate. The sender decrypt the signed certificate with the known Public Key of the Certificate Authority and thus verifies that the information that provides the certificate (Public Key, elements of identity etc.) they have not been degraded by the moment that the certificate signed. Then, it is henceforth certain that the Public Key that it will be used belongs to the particular recipient.

Initially, the user contact with the Certificate Authority through software parcels (browsers or cryptographic parcels), which contains the most signed testimonial certain known Certification Authorities they called "Root Certification Authorities". With the legal buy of software, the user ensures also the legal Public Keys from certain Certification Authorities. One of the advantages of the Certification Authorities is that, except the certification of the user, is possible to certify another Certification Authority. This sequential process develops a "chain testimonial" which develops a hierarchy of Certification Authorities in a tree form. The start of this hierarchy is a widely known and reliable Authority (Root Certification Authority). This authority signs certificates which belong in inferior Certification Authorities and represent individual or inferior production of certificates. This hierarchy can have in the depth a lot of Certification Authorities. Because, each testimonial of Certification Authority is fully signed by other Certification Authority and this up to the root of hierarchy, the user of certificate can check all the tree structure from the particular Certification Authority up to a widely famous and confidential authority (Root Certification Authority).

The hierarchy of Certification Authority is more common in internal networks (intranet) as well as in protocols of digital payments as the SET.

Each Certification Authority has the capability to contain the secure parameters of all Certificate Authorities hierarchal, gives also the guarantee of identity of user that provides an individual Certification Authority.

The Certification Authority in order secure all the previous guarantees can place various restrictions in the certificates as, the time interval at which the certificate will be in effect (for example the certificates that are given by the most commercial Certification Authorities they are in effect for one year), as well as it can cancel or recall certificates. This can happen when the user is released by the Organism of Certification or if exists loss or alteration of the User's or even Certification Authority's Private Key.

The Certification Authority can publish lists of retraction of certificates (Certificate Revocation List – CRL) in order to a user (recipient) be informed relatively before makes acceptable the Public Key of the sender. The recipient has the possibility, through the computer's software, to be confirmed for the sender's validity testimonial at the period of the certification.

The certificates can be published with an application by a user or through the process of Computer. The companies' servers that execute web works contain certificates server.

These server's certificates are used in order to help main server in the certification of user - customer and support the exchange of encrypted information through the network. Also, it supports the safeguarding of information as passwords and the credit cards numbers.

C. SEARCH A USER CERTIFICATION

The certificate of each sender – user should be available in each user who wishes to communicate with. In the case where the sender signs different types of documents as emails or is certified in web works, then is possible to include the certificate with the signature However, the recipient may want to send an encrypted message to the sender (using his Public Key) before the recipient receives from the sender the certificate.

In order to be ensured the wide availability of certificates, the Certification Authority publishes all the certificates in a list.

The list is a service which allows the users to receive information for other users. In this list also are published and lists of retraction certificates, as well as these should be widely available.

Because of that technology in particular sector is not enough developed, there are various conflicts in the technical committees for the use of lists. In this case, the Lightweight Directory Access Protocol (LDAP) constitutes the main model for the access in lists.

All this kind of lists have suitable structure that is required for big lists, with segregation between the different organisms and applications (with the system of the Public Key Structure), with the best organized storage information, the information are assembled or distributed by the suitable way, as well as with the process of guarantee of information to be effected.

D. LEGAL FRAMWORK – THE HELLENIC TELECOMMUNICATION & POST COMMISSION

At the daily validity of any transaction is required the signature dealing. The signature in a text, which proof that the person who signs the content of the text knows, recognizes, accepts this text. The signatory cannot deny the signed text, except of cases infringing behavior (counterfeiting, fraud ect). A signed text has legal substance and ratifies the transaction.

The Presidential Degree (P.D) 150/2001 is the implementation of Directive 99/93/EC of the European Parliament and of the Council on a Community framework for electronic signatures, determined that the frame which a digital signature is recognized legally as by one's own hand. This means that under specific conditions, the persons that contract in an electronic transaction and sign electronic, after cannot deny.

Moreover, the Presidential Decree, except the following:

Determine the terms that should be in effect to digital certificates in order to be authenticated certificate. Also, the terms and conditions for certification – service – provider, in order to provide recognized certificates.

It establishes a legal framework for electronic signatures and certain certification-services in order to ensure the proper functioning of the internal market.

Establishes the conditions of legal recognition in the EU of authenticated certificates which is published from Certification – service – Providers who are based in countries except EU, and other relative forecasts concerning international aspects

Establishes the frame of responsibility for the Certification – Service - Providers

Establishes to the Hellenic Telecommunications & Post Commission (EETT) concrete competences

The competences of EETT, as arise from PD 150/2001, are the following:

The permission of Voluntary accreditation, setting out rights and obligations specific to the provision of certification services, to be granted upon request by the certification-service-provider concerned, by the public or private body charged with the elaboration of, and supervision of compliance with, such rights and obligations, where the certification-service-provider is not entitled to exercise the rights stemming from the permission until it has received the decision by the body.

The monitoring and the control installed in Greece of certification-service-provider, as well as institutions of accreditation and control of conformity signatures to Annex III of PD. 150/2001 (provided that the EETT assigns such duties in other institutions) (article 4 par. 8).

The conformity of secure signature-creation-devices with the requirements laid down in Annex III Presidential Decree 150/2001 (article 4 par. 2) shall be determined by appropriate public or private bodies designated by Member States

The imposition of fines to certificate-service-providers, who acts as accrediting, without to be (article 4 par 9)

EETT shall maintain a registry of all Certification Service Providers established in Greece in electronic form or/and printed form in order to report to the European Committee on the names and the addresses of them (articles 8 par.2 and 3).

EETT with No. 248/71 Regulation for the Provision of Electronic Signature Certification Services (Government Gazette 603/B/16-5-2002) issues a Regulation for the Provision of Electronic Signature Certification Services, which is as follows

Issues pertaining to the provision of electronic signature certification services

In particular, issues pertaining to Qualified Certificates

The supervision and inspection of the electronic signature Certification Service Providers established in Greece that issue

Qualified or Non-Qualified Certificates or provide other electronic signature related certification services.

VI. DIGITAL DOCUMENTS CORRESPODENCE

A. DOCUMENTS DIGITALIZATION

All the documents which are signed digital and dispatched electronic should be in digital form. For this reason, if the documents are in hard copy, should be digitalized.

Digitalization is the process of transformation of elements as document, text, picture, object or signal from proportional in digital form for their import in the computer, so that they can be stored and become object of treatment of the user.

A digital document need less space of the printing one, allow to quick search in the information that includes, while it is easier accessible. The filings to digital documents are much more easily. Thus, a large amount of records that have been created in the past can be easily accessed. The digitalization is implemented by specialized functions and documents scanning. The information is incorporated immediately in the daily flow of work on that way. The Scan to Email offers, for example, the opportunity documents to be sent direct from corporate scanner in the desirable recipients. The Scan to USB constitutes another example, where the information is stored immediately in USB sticks, save time and processes.

Another useful function is Scan to Server, where the digital documents are stored directly from the scanner in selected network file. This process of digitalization is offered a direct and wider access in all the corporate documents by certified users. The digital filing save the documents in a secure space, where they cannot be lost under normal circumstances (in this contribute the process of backup).

B. SHIPPING DOCUMENTS AND BOOKS-USE OF THE DIGITAL SIGNATURE AND DIGITAL CORRESPODENCE

Generally with the term Shipping documents or documents of boat (Ship's papers) are all the essential books and certificates which are required to observe and have a boat, because of law or the shipping tactic and practice.

The Shipping documents, according to the article of 46 Code of Public Maritime Law, are the following:

- Certificate of Registry
- Tonnage Certificate
- Ship Security Certificate
- Load Line Certificate
- Crew List
- Log-Book
- Engine Logbook
- Crime Sheet
- Oil Record Book – ORB
- Record of Equipment for the Passenger Ship Safety Certificate

Also, other shipping documents from the shipping tactic and practice are:

- Decontamination Certificate
- Certificate of fire safety (fire extinguishers and systems of firefighting)
- Ship Sanitation Control Certificate
- Customs Book through Canal passage
- Permission of Departure (previous port)
- Insurance policy of danger of pollution
- Charter Party

The volume of all above shipping documents is particularly big. More concretely, according to the IMO document FAL 2/Circ 87 MEPC/Circ 426 MSC/Circ 1151 (17/12/2004) for Revised List of Certificates and Documents Required be Carried Being Board Ships, are required at least 67 different certificates only on the conformity with the provisions of conventions of organism.

Moreover, The marine transport creates dangers as for the safety as for the maintenance of good situation of these documents. The shipping documents are signed and checked by a very big number of persons (personnel of boat and company, inspectors Port state, state of flag, classification societies, ect) under pressure of time and in big latitude, where the boat operates. All these documents occupy precious space and weight on the boat, while the safety of so much big volume of documents does not follow the modern tasks for the green environment and friendly practices (reduction of litter, recycling, etc). All the above documents are very often vulnerable in forgeries.

For this reason, the ensured exchange of data, the single form of documents, the speed of mailing and reception data and the friendly use to environment will be provided through the digital distribution of documents between the boat and the shipping company and further more between the ship and public or private bodies.

All countries which register ships and boat with their flag should implement the digital distribution of documents.

Everybody should follow the example of Liberia, which is advanced, since 24 Sep 2008, in the publication of digital certificates and documents that concern the Bunker Civil Liability Convention, Radio Licenses, Civil Liability Convention, Vessel's port of registration, Administration or Recognized Organization issuing the Company's Document of Compliance and/or Safety Management Certificate, Administration or Recognized Security Organization (RSO) issuing the vessel International Ship Security Certificate with respect to the ISPS Code.

The digital signature and the seal on documents and certificates are considered valid as prototypes. The digital certificates and documents can be transmitted digital and should be protected from falsification or alteration. The issue and expiry date can be written or typed while the issuing process by authorized inspectors or Liberian surveyors. Also, all the colored printed certificate documents can be used valid as prototype.

VII. END

The use of digital signature and the digital distribution of documents, specifically in the sector of Shipping, will constitute one of the most important achievement, because it will give the possibility in shipping enterprises to collaborate in world level with thousands other companies direct, economically and safe the total of correspondence that became by fax or the post. The main result is the minimum of the operation cost, the digital filing of the documents. The digital correspondence of documents will be happening instantly everywhere in worldwide.

REFERENCES

[1] https://www.instantssl.com/https-tutorials/digital-signature.html

[2] https://www.cgi.com/filew/white-papers/cgi_whpr_35_pki_e.pdf

[3] https://en.wikipedia.org/wiki/Digital_signature

[4] https://www.docusign.com/how-it-works/electronic-signature/digital-signature/digital-signature-faq

[5] https://en.wikipedia.org/wiki/Public_key_certificate#Common_fields

[6] https://en.wikipedia.org/wiki/Public_key_certificate

[7] https://en.wikipedia.org/wiki/Public_key_infrastructure

[8] https://en.wikipedia.org/wiki/Digital_Signature_Algorithm

[9] https://www.tutorialspoint.com/cryptography/cryptography_digital_signatures.htm

[10] http://searchsecurity.techtarget.com/definition/digital-signature

[11] http://www.productivity501.com/digital-signatures-encryption/4710/

[12] https://www.symantec.com/connect/articles/introduction-encryption

[13] https://gpgtools.tenderapp.com/kb/how-to/introduction-to-cryptography

[14] http://www.articsoftpgp.com/encryption.htm

[15] https://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/crypto.html

[16] https://www.globalsign.com/en/blog/glossary-of-cryptographic-algorithms/

[17] https://www.storagecraft.com/blog/5-common-encryption-algorithms/

[18] http://searchsoftwarequality.techtarget.com/definition/cryptography

[19] https://www.garykessler.net/library/crypto.html

[20] https://en.wikipedia.org/wiki/Key_(cryptography)

[21] https://www.europeanpaymentscouncil.eu/sites/default/files/kb/file/2017-12/EPC342-08%20v7.0%20Guidelines%20on%20cryptographic%20algorithms%20usage%20and%20key%20management.pdf

[22] https://csrc.nist.gov/csrc/media/projects/key-management/documents/transitions/transitioning_cryptoalgos_070209.pdf

[23] http://www.eett.gr/opencms/opencms/EETT/

[24] http://www.wikipedia.gr

[25] www.dione.lib.unibi.gr

[26] www.netweek.gr

[27] Notes Nikolaos Bardis, 2014-2015

[28] Nikolaoq G Bardis, etc:Accelerated Modular Multiplication Algorithm of Large word Length Numbers with a Fixed Module, 2010

[29] Nikolaos G. Bardis, etc:Fast subscriber indentification based on the zero knowledge principle for multimedia content distribution (2010)

[30] Nikolaos G. Bardis, etc:design and development of a secure military communication based on AES prototype crypto algorithm and advanced key management scheme, ACM DL.

[31] Malerbas M. "Shipping Documents" Puclishis Stamoulis, Athens 1999

[32] http://www.eett.gr/opencms/opencms/EETT/Electronic_Communications/DigitalSignatures/IntroEsign.html