Easing PAIN with Digital Signatures

M. Tariq Banday
P.G. Department of Electronics and Instrumentation Technology
University of Kashmir, India

ABSTRACT

Different forms of encryption techniques are being used to ensure privacy of data transmitted over Internet. Digital signature which uses Public Key Encryption (PKE) ensures privacy of conversation, integrity of data, authenticity of sender and non-repudiation of sender. Several applications on desktops, mobiles and other devices use it to secure data of various applications, services and access to devices. Digital signature exists as a file on some storage device or is embedded in hardware devices. Digital signature certificates are digital signatures that are signed by some third party certifying authority. This paper describes working of digital signature through illustrations and explains various procedures and algorithms involved in signings the data through use of digital signature. It introduces an open source software tool that can be used to study processes, procedures and algorithms used in digital signatures. It also presents information about different types of digital signature certificates currently available, file formats used therein and programming support of digital signature in .NET languages. Further, it presents an excerpt of digital signature scenario in Indian Government.

General Terms

Encryption, Hashing, PAIN, Public Key Encryption, PKI, Authentication, Privacy, Information Security, PKI in India.

Keywords

Digital Signature, Digital Signature Certificate, Public Key, Private Key, Electronic Signature, Privacy and Authentication.

1. INTRODUCTION

With the advent of digital storage and communication technologies the entire spectrum of storage and communication systems has been revolutionized as digital information can be easily stored, copied, changed and transported [1]. These desirable properties of digital information are very useful but owing to easy and almost undetected modification of digital data, they have raised several security concerns. Therefore, digital data is regarded as unreliable in areas where privacy, authentication, and integrity of data are of concern unless some security procedure is attached to it. There are areas like contracts, receipts, approvals and similar others where users have severe and genuine concerns of unauthorized modification or disclosure of data. Hand signatures don’t change this situation, because it is easy to transfer a hand signature from one digitized document to another or to modify a digitized document that is hand signed. The risk of data misuse has increased many folds with the advent of networking and wireless communication as many users can gain access to the data if not secured.

The solution to all these issues is digital signature [2]. A digital signature is not a digitized hand signature, but a special kind of check-sum. Secret information ensures that a digital signature cannot be forged, while public information enables the verification of the signature. Digital signature ensures prevention of unauthorized access to data while ensuring accurate authentication to data without interference. Digital signature is currently used in various application domains [2] that include: i) Government: Filing tax returns online by taxpayers, citizen ID card, issuing forms and licenses, reservations & ticketing, ii) Banking: Inter/ Intra bank messaging systems, corporate Internet banking applications, iii) Financial Services/Broking: Online trading, electronic contract notes, iv) B2B: Online tendering, e-Procurement, v) Healthcare: healthcare management system, electronic medical recording, electronic prescriptions and many others.

This remaining paper is organized as follows: section 2 briefly describes privacy, authentication, integrity and non-repudiation. Section 3 introduces encryption and in section 4 functioning of Public Key Infrastructure (PKI) are illustrated. This section illustrates steps involved in digital signatures and certificates. Section 5 lists various encryption and signings algorithms. Section 6 introduces CrypTool and presents .NET programming language support for digital signatures. It also lists various types of digital signatures, file formats and companies providing digital signatures. Section 7 gives an excerpt of digital signature scenario in Indian Government which is followed by conclusion.

2. PAIN

PAIN which collectively refers to Privacy, Authentication, Integrity and Non-repudiation are four key factors to achieve information security [3]. Privacy also called confidentiality, guarantees non-disclosure of information to unauthorized persons. Authentication ensures that the document or software is genuine. Integrity as a concept means that there is resistance to alteration or substitution of data, and/or that such changes are detected and provable. The information should not be changed except by an authorized agent. Non-repudiation is a security service that prevents a party from falsely denying that it was the source of data that it did indeed create.

2.1 Privacy

It is the right of individuals to control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed. Privacy guarantees the prevention of unauthorized access and manipulation of data. It means that a transaction between businesses cannot be viewed or interfered with by an outside party.

2.2 Authentication

Authentication is the security process that validates the identity of a communicating party. In the simplest implementation, this takes the form of a password. Passwords can be easily compromised through indiscretion and typically do not address who is entering the password. Another variant of authentication is known as strong authentication. In this implementation, authentication is provided by a digital signature which is an encrypted value provided by the entity requesting authentication that can only be decoded by the public key of the signature's owner.

2.3 Integrity
Integrity ensures security against forgery which includes policies to stop distribution of software and data contaminated with viruses, Trojans, Spywares, etc. This usually involves the use of checksums, one-way hashes, or other algorithmic validation of the data. Whether the data might be changed by accident or malice, preventing that change is the foremost concern, and detecting it is second. Integrity can be maintained at many levels, from the hardware all the way to the application logic. At first glance it might seem that authenticity is included in the concept of Integrity. Integrity is more specifically about the content of the data itself.

2.4 Non-repudiation
Non- repudiation means the ability to prove that a transaction originated from a particular party. So that party cannot deny that he performed a certain transaction. A receiver cannot deny that he received a certain message from a sender, and a sender cannot deny that he sent a message to the receiver. It is a security service that prevents a party from falsely denying that it was the source of data that it did indeed create.

3. ENCRYPTION
It is the process of encoding a plain text message so that it cannot be understood by intermediate parties who do not know the key to decrypt it [4]. The purpose of encryption is to keep secrets. It has other uses but encryption was first used to protect messages so that unauthorized person could not decode the message. An Encryption function combines the message and the encryption key to produce an encrypted result. Without prior knowledge of the secret key the result makes no sense. An algorithm used for encryption is called Cipher and the encoded message is called Cipher text. A computing system that implements one or more specific encryption algorithms is called Cryptosystem.

3.1 Secret Key Encryption
The encryption technique that uses same key called secret key or private key for both encoding and decoding the plain message is called Secret Key Encryption [4]. It is also called Symmetric Key Encryption, Shared Key Encryption and Private Key Encryption. The secret key must be kept secret by all parties because it is used to encrypt and also decrypt message. Its disclosure enables anyone to decrypt the encoded information and as such can compromise information security. An algorithm that implements Secret Key Encryption is called Symmetric Algorithm. Classic symmetric encryption [4] can be achieved using following techniques: Caesar, Vigenère, Hill, Monoalphabetic substitution, Playfair, ADFGVX, Byte Addition, Exclusive-OR, Vernam, homophonic substitution, permutation, and Solitaire. Modern Symmetric Encryption [4] can be achieved using following techniques: IDEA, RC2, RC4, DES in ECB mode, DES in CBC mode, Triple-DES in ECB mode, Triple-DES in CBC mode, Rijndael, and AES (self extracting). Secret Key Encryption techniques require sophisticated mechanism to securely distribute key to all parties. However, they are faster than asymmetric key encryption algorithms and play a very vital role in the implementation of Public Key Infrastructure.

3.2 Public Key Encryption
Public Key Encryption [5] is an encryption methodology that uses Asymmetric Key pair (Public Key and Private Key) from which one key is used for encryption and the other is used for decryption. It allows the distribution of an encryption key that does not compromise the secrecy of the decrypting Private Key due to the utilization of a related pair of one-way functions. The Public Key is made public and is distributed widely and freely. The Private Key is never distributed and must be kept secret. Given a key pair data encrypted with the Public Key can only be decrypted by its corresponding Private Key and conversely data encrypted with the Private Key can only be decrypted with the corresponding Public Key. This characteristic is used to implement encryption and digital signature. Some Asymmetric Encryption algorithms are: RSA, DSA, and ECDSA. It offers key advantages like simplified key distribution, digital signature and long-term encryption. Public key authentication is often used when authentication should be performed automatically without user intervention. The systems involved can trade public keys and authentication information without the user interacting with the system [6]. For this reason, public key based authentication and its derivatives like certificate based authentication are frequently used for machine authentication and for establishing anonymous encrypted sessions such as Secure Sockets Layer (SSL), Network Layer protocols, etc. It is also used in e-mail communications, document shearing, etc.

4. FUNCTIONING OF PKI
4.1 Encryption and Decryption
In encryption using Public Key Infrastructure (PKI), the sender encrypts the plain message and transmits it to the recipient using any available mechanism which is decrypted by the recipient. The functioning of encryption and decryption is illustrated in figure 1.

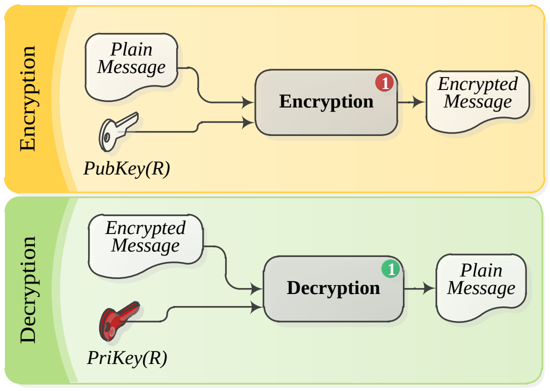

Figure 1: Functioning of Encryption and Decryption

The operations performed in each step are given below:

Process: Encryption

Steps	***Operation***
1	*The Plain message is transformed into encrypted message by encrypting it using the Public Key of the intended recipient.*

Process: Decryption

Steps	***Operation***
1	*The received encrypted message is decrypted into the plain message by using Private Key of the recipient*

The encrypted message can only be transformed to plain text by using Private Key that corresponds to Public Key used for encrypting it. Any attempt to decrypt the encrypted message with wrong Private Key will be unsuccessful, thus ensuring privacy of the message. This scheme however does not ensure authentication, integrity and non-repudiation.

4.2 Signing and Verifying

In signings using PKI, the sender transforms the plain message into signed message and transmits it to the recipient using any available mechanism which is decrypted by the recipient. The functioning of signing and verification is illustrated in figure 2.

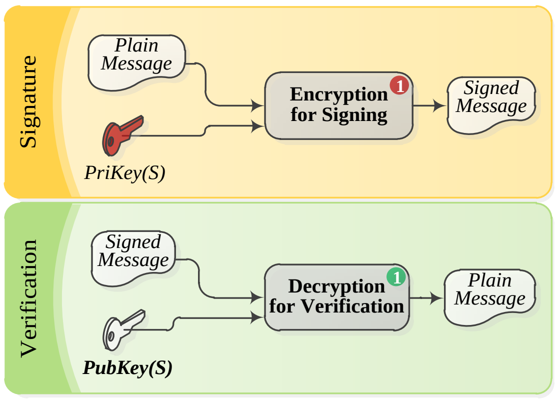

Figure 2: Functioning of Signings and Verification

The operations performed in each step are given below:

Process: Encryption (for Sigining)

Steps	***Operation***
1	*The Plain message is transformed into encrypted form called signed message by encrypting it using the Private Key of the sender.*

Process: Decryption (for Verification)

Steps	***Operation***
1	*The received signed message is decrypted into the plain message by using Public Key of the sender.*

The signed message can only be decrypted to plain message by using Public Key that corresponds to Private Key used for signings it. The Public Key is typically attached to the signed message. Any attempt to decrypt the signed message with wrong Public Key will be unsuccessful, thus ensuring authentication, and non-repudiation. This scheme however, does not ensure privacy and integrity because anybody getting access to the signed message can decrypt it as the Public Key of the sender is available freely and widely.

4.3 Signing and Verifying with Hashing

A one way mathematical function called hashing function [4] can be used to compute a small and fixed length message digest also called figure print or hash or message abstract to ensure faster sigining because sigining a lengthy plain message can be time consuming. The advantages of using a hash function are: i) Hashing is a one – way function and thus it is not possible to compute the original message from its hash, ii) Any change in the message will also change the message abstract – thus changes can be immediately detected, iii) Hash function receives messages of any length and produces hash of fixed length which is smaller than the message itself, and, iv) Hashing algorithms are faster than any symmetric and asymmetric encryption algorithms.

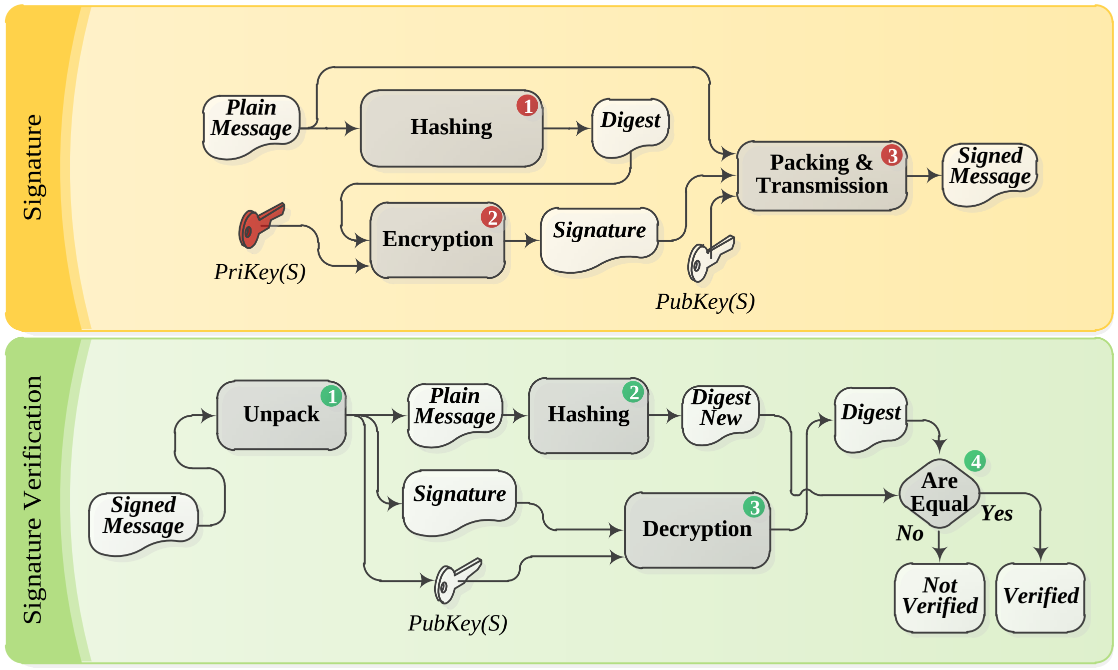

Figure 3: Functioning of Signings and Verification with Hashing

The sender uses a hash function to compute message digest and then encrypts the digest using his Private Key to get message signature. Since the signature is based on digest and not the plain message, it is not possible to reconstruct the message from the signature. Thus this signature is appended to plain message and is transmitted over any transmitting medium. The Public Key is typically attached to the signed message. The recipient separates the plain message from the signature and the Public Key and then using same hashing function computes message digest of the plain message received, the received signature is decrypted into digest using Public Key of the sender. The two digests are compared to determine authenticity of sender. Any change in the plain message will also change the message abstract and thus the changes can be easily detected. The functioning of signing and verification is illustrated in figure 3 and the operations performed in each step are given below:

Process: Sigining

Steps	*Operation*
1	***Hashing**: The sender computes message digest of plain message using some hash function like MD5, etc.*
2	***Encryption**: The message digest is transformed into encrypted form called message signature by encrypting it using the Private Key of the sender.*
3	***Packing:** The plain message, message signature and the Public Key of the sender are packed together to form a single packed unit.*

Process: Verification

Steps	*Operation*
1	***Unpack**: The received packed unit is unpacked into plain message, message signature and the Public Key of the sender.*
2	***Hashing**: The recipient computes the message digest of the plain message received using same algorithm that was used by the sender. This is compared with received message digest in step 4.*
3	***Decryption:** The received message signature is decrypted using the Public Key of the sender to computer the original message digest for comparison with the message digest computed by the recipient.*
4	***Comparison:** Message digest compute from plain message received by the recipient and the message digest obtained after decrypting the received message signature are compared.*

The message digest obtained by decrypting the received signature and digest computed by hashing received plain message are same if plain message has not been modified and correct Public Key is used to decrypt the signature. If the two digests are same, the signature is verified, otherwise not. This scheme achieves authentication, integrity and non-repudiation but not privacy.

4.4 Digital Signature

Encryption and sigining along with hashing can be combined to ensure privacy, integrity, authentication and non- repudiation. The plain message is hashed to compute a message digest which is encrypted using Private Key of the sender to generate the message signature. The plain message, the message signature and the Public Key of the sender are packed together which is transformed into signed and encrypted message using the Public Key of the recipient. The recipient unpacks received message which is the signed and encrypted message after which same hashing function is used to compute message digest of the received message which is compared to the decrypted signature. The functioning of signing and verification is illustrated in figure 4.

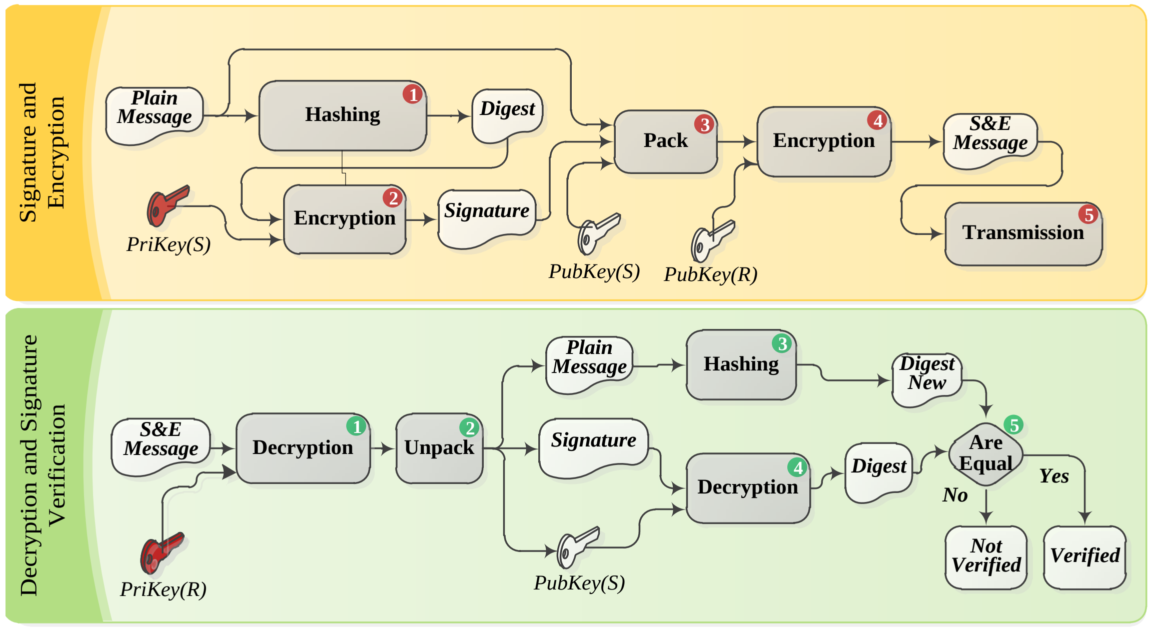

Figure 4: Functioning of Digital Signature

The operations performed in each step are given below:

Process: Sigining and Encryption

Steps	*Operation*
1	***Hashing**: In this step a hashing function is used to compute message digest which is small and unique representation of the message. The purpose of evaluating a digest is to ensure message integrity. The*

digital signature is applied on the message digest which is smaller than the message itself. Further, hashing functions are faster than symmetric and asymmetric encryption algorithms.

2 ***Encryption**: In this step encryption using Private Key of the sender is used to sign the message digest. Signing is performed to obtain non- repudiation. The message digest can be recovered by decrypting the encrypted message digest called message signature using the corresponding public key (public key of sender). Several algorithms have been proposed for signing.*

3 ***Packing:** The plain message, message signature and the Public Key of the sender are packed together to form a single packed unit.*

4 ***Encryption:** The packed message containing the plain message and the signature of the message in the form of encrypted digest along with the public key of the sender is encrypted using the public key of the recipient to form signed and encrypted message.*

Process: Decryption and Verification

Steps ***Operation***

1 ***Decryption:** In this step the received message which is signed and encrypted is decrypted using the private key of the receiver to form a packed message containing plain message, the signature and the public key of the sender.*

2 ***Unpack**: The decrypted message is unpacked into plain text, signature and the public key of the sender.*

3 ***Hashing**: In this step same hashing function that was used by the sender is used to compute message digest from the plain text message obtained after decrypting and unpacking the received message.*

4 ***Decryption:** In this step, the received message signature is decrypted using the received Public Key of the sender to obtain the message digest computed before transmitting the message.*

5 ***Comparison:** Message digest computed from plain message received by the recipient and the message digest obtained after decrypting the received message signature are compared.*

Signed and encrypted messages can only be decrypted by the correct Private Key of the recipient thus ensuring privacy and at the same time hashing and signature verification by digest comparisons offer authenticity, integrity and non- repudiation.

The digital signature can be used to provide assurance that the claimed signatory signed the information. In addition, it may be used to detect whether or not the information was modified after it was signed. These assurances may be obtained whether the data was received in a transmission or retrieved from storage.

4.5 Digital Signature (using Symmetric Key)

Encryption and decryption algorithms using asymmetric keys are too slow to be used for long messages, thus a symmetric key is generated and is used in Digital Signatures to encrypt the packed unit (plain message, the message signature and Public Key of sender). The Symmetric Key is encrypted using the Public Key of the recipient so that it can only be decoded by the intended receiver who can use it to decrypt the packed unit before unpacking it and drawing semantics. The functioning of signing and verification is illustrated in figure 5.

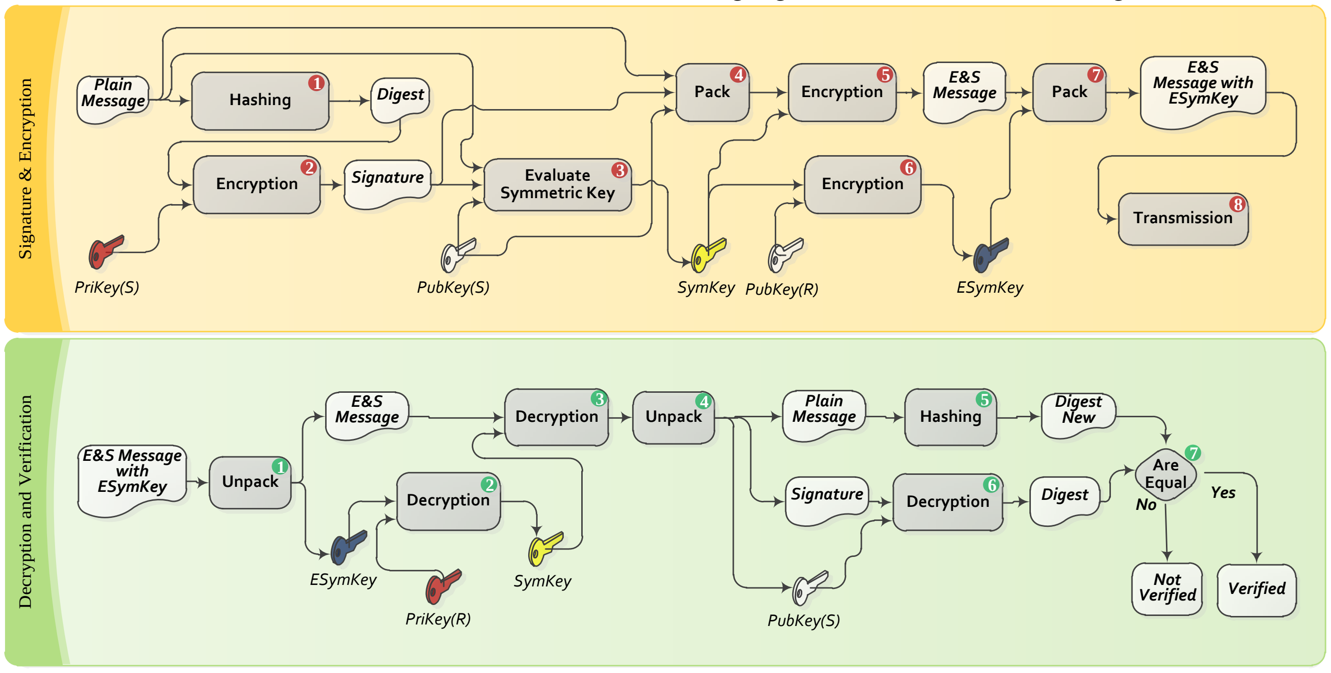

Figure 5: Functioning of Digital Signature (using Symmetric Key)

The operations performed in each step are given below:

Process: Sigining and Encryption

Steps ***Operation***

1 ***Hashing**: This step is same as that for Digital Signature detailed in signed and encryption step 1 of section 4.4.*

2 ***Encryption**: This step is same as that for Digital*

Signature detailed in signed and encryption step 2 of section 4.4.

3	***Symmetric Key Evaluation:*** *A Symmetric Key is calculated using some algorithm based on plain message, message signature and the public key of the sender. Any algorithm may be used for calculating the Symmetric Key.*
4	***Packing:*** *The plain message, message signature and the Public Key of the sender are packed together to form a single packed unit.*
5	***Encryption:*** *The packed message containing the plain message and the signature of the message in the form of encrypted digest along with the Public Key of the sender is encrypted using the Symmetric Key calculated in step 3 to form signed and encrypted message.*
6	***Encryption****: The Symmetric Key is encrypted to form encrypted Symmetric Key using Public Key of the recipient*
7	***Packing:*** *The encrypted and signed message and encrypted Symmetric Key are packed into a single unit called encrypted and signed message with encrypted symmetric key.*

Process: Decryption and Verification

Steps	*Operation*
1	***Unpack****: The received encrypted and signed message with encrypted symmetric key is unpacked in this step.*
2	***Decryption:*** *The encrypted symmetric key is decrypted using the Private Key of the recipient to get the Symmetric Key used for encryption in step 5 of sigining and encryption.*
3	***Decryption:*** *Decryption: In this step the received message which is signed and encrypted is decrypted using the Symmetric Key decrypted in step 2 to form a packed message containing plain message, the signature and the public key of the sender.*
4	***Unpack****: This step is same as for Digital Signature detailed in step 2 of decryption and verification of section 4.4.*
5	***Hashing****: This step is same as for Digital Signature detailed in step 3 of decryption and verification of section 4.4.*
6	***Decrypt:*** *This step is same as for Digital Signature detailed in step 4 of decryption and verification of section 4.4.*
7	***Comparison:*** *This step is same as for Digital Signature detailed in step 5 of decryption and verification of section 4.4.*

The operation of this scheme is similar to that of digital signature without use of Symmetric Key and offers privacy, integrity, authentication and non- repudiation and simultaneously offers additional advantage of faster encryption and decryption. However, additional algorithm is required to evaluate Symmetric Key.

4.6 Digital Signature Certificate

Digital Signature is identity information encrypted with a private key and therefore decryptable with a public key. Digital Certificates are Digital Signatures that have themselves been "SIGNED" using the Digital Signature of some trusted authority, thus creating a "CHAIN" of authentications. In certificate hierarchy, the ultimate authority is called the Root Certifying Authority (Root CA). Root CA's certify the identities of all members so that members who trust the Root CA can trust anyone that they have certified. VeriSign, Thawte and Entrust are examples of Root CA's. In India Controller for Certifying Authorities (CCA) has been authorized by the Government to license and regulate the working of CA. CCA is National Root CA and operates RCAI for certifying the public keys of CA's using it private key. The functioning of signing and verification is illustrated in figure 6 and the operations performed in each step are given below:

Process: Sigining and Encryption

Steps	**Operation**
1	***Hashing****: This step is same as that for Digital Signature detailed in sigining and encryption step 1 of section 4.5.*
2	***Encryption****: This step is same as that for Digital Signature detailed in signed and encryption step 2 of section 4.5.*
3	***Certificate Validation:*** *Validity of the Digital Signature Certificate of the recipient is determined from Certification Authority to ensure that the recipient can later decrypt the message.*
4	***Symmetric Key Evaluation:*** *This step is same as that for Digital Signature detailed in signed and encryption step 3 of section 4.5.*
5	***Packing:*** *This step is same as that for Digital Signature detailed in signed and encryption step 4 of section 4.5.*
6	***Encryption:*** *This step is same as that for Digital Signature detailed in signed and encryption step 5 of section 4.5.*
7	***Encryption****: This step is same as that for Digital Signature detailed in signed and encryption step 6 of section 4.5.*
8	***Packing:*** *This step is same as that for Digital Signature detailed in signed and encryption step 7 of section 4.5.*

Process: Decryption and Verification

Steps	*Operation*
1	***Unpack****: This step is same as that for Digital Signature detailed in decryption and verification step 1 of section 4.5.*
2	***Decryption:*** *This step is same as that for Digital Signature detailed in decryption and verification step 2 of section 4.5.*
3	***Decryption:*** *This step is same as that for Digital Signature detailed in decryption and verification step 3 of section 4.5.*
4	***Unpack****: This step is same as that for Digital Signature detailed in decryption and verification step 4 of section 4.5.*
5	***Certificate Validation:*** *Validity of the Digital Signature Certificate of the sender is determined from third party Certification Authority.*
6	***Hashing****: This step is same as that for Digital Signature detailed in decryption and verification step 5 of section 4.5.*
7	***Decrypt:*** *This step is same as that for Digital Signature detailed in decryption and verification step 6 of section 4.5.*
8	***Comparison:*** *This step is same as that for Digital Signature detailed in decryption and verification step 7 of section 4.5.*

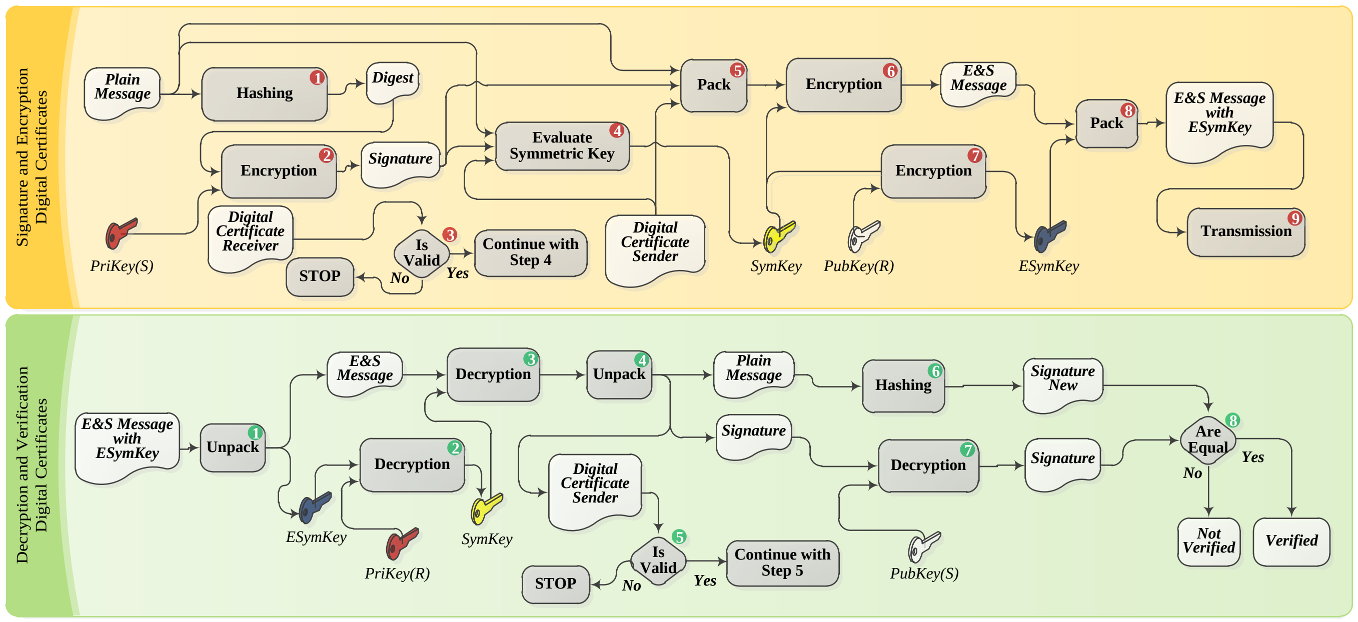

Figure 6: Functioning of Digital Signature Certificates

Digital signature certificates create a chain of authentications, enabling identification of sender and management of complex key and certificate management procedures. It offers information security for a variety of applications and services. It has been adopted as a standard security mechanism by leading companies, standardizing bodies and government legislations of various countries.

5. ALGORITHMS USED IN PKI

5.1 Hash Algorithms

A hash function is a function that maps a bit string of arbitrary length to a fixed length bit string. Approved hash functions are designed to satisfy the following properties: a) One-way: It is computationally infeasible to find any input that maps to any new pre-specified output, and b) Collision resistant: It is computationally infeasible to find any two distinct inputs that map to the same output.

The cryptographic hash algorithms SHA-1, SHA-2 (SHA-224, SHA-256, SHA-384, SHA-512) and MD5 are commonly used functions that take an arbitrary length input message and return a unique fixed length output value known as the message digest. The digest is effectively a compressed but irreversible representation of the entire input message which can be used to provide data integrity checking, since changes in the digest indicate the message has also changed. This property has many useful applications such as the Digital Signature Standard (FIPS 186-3). When used to implement a Hash-based Message Authentication Code (HMAC) to FIPS 198, the hash function is keyed to provide a means of data authentication used in many secure protocols e.g. IPsec, TLS/SSL. The Secure Hash Algorithms (SHA) was developed by the US National Institute of Standards and Technology (NIST) as defined in FIPS 180-3. This describes the original SHA-1 hash as well as the more recent SHA-2 family of algorithms. SHA-1, SHA-224 and SHA-256 are all 32-bit hash algorithms that generate 160-bit, 224-bit, and 256-bit digests respectively. The 64-bit SHA-384 and SHA-512 hash algorithms provide even greater security and generate 384-bit and 512-bit digests. The MD5 hash algorithm (developed by Ron Rivest) is a legacy 32-bit hash algorithm that generates a 128-bit digest and is described in RFC 1321.

Various hash algorithms, their characteristics and hash size are listed in table 1.

Table 1: Hash Algorithms

Name of Algorithm	Type and Characteristics	Hash Size
Secure Hash Algorithm 1 (SHA1) [7]	FIPS approved; other versions (SHA256, SHA384, SHA512) provide longer outputs	160 bit
Message Digest 5 (MD5) [8]	Potential weaknesses is that it can be used as a keyed hash	128 bits
RACE Integrity Primitives Evaluation Message Digest 160 (RIPEMD-160) [9]	Developed as part of the EC's Research and Development in Advanced Communications Technologies in Europe (RACE)	160 bits
TIGER Hash [10]	Designed for efficient operation on 64-bit platforms	192 bits

Towards development of SHA-3 hashing algorithm, National Institute of Science and Technology (NIST) (http://csrc.nist.gov) has announced a public competition (Federal Register Notice to develop a new cryptographic hash algorithm, which converts a variable length message into a short "message digest" that can be used in generating digital signatures, message authentication codes, and many other security applications in the information infrastructure. NIST also plans to host a final SHA-3 Candidate Conference in the spring of 2012 to discuss the public feedback on these candidates, and select the SHA-3 winner later in 2012.

5.2 Digital Signature Algorithms

Digital Signature Algorithm (DSA), Rivest-Shamir-Adleman (RSA) as specified in ANSI X9.31, and Elliptic Curve DSA (ECDSA) as specified in ANSI X9.62 which are captured in FIPS PUB 186-2 (with change notice 1 dated 5 October 2001) are three digital signature algorithms. These algorithms along with their characteristics and minimum key size are listed in table 2. Elliptic Curve Cryptography (ECC) is by far the most efficient algorithm with respect to key size. However, many aspects of elliptic curve technology have been patented by Certicom (www.certicom.com) and therefore licenses may have to be obtained as Certicom claims over 300 patents (and patents pending) various ‘efficient implementations of ECC’ in both hardware and software of elliptic curve technologies. It also holds patents on ECC key agreements, etc. There are open-source elliptic curve libraries that are available for use royalty- and license-free (e.g. libecc, a C++ open source ECC crypto library available at http://libecc.sourceforge.net/). RSA had been patented by RSA but the patents have expired.

Table 2: Digital Signature Algorithms

Name of Algorithm	Type and Characteristics	Min. Key Size
Digital Signature Standard (DSS) [11]	FIPS 186-2 digital signature Digital signature based on SHA1 hash, unencumbered (no patents, no licenses)	1024 bits
RSA Digital Signature [12]	RSA digital signature (FIPS approved) Previously patented digital signature (expired 2000	1024 bits
Elliptic Curve Digital Signature (ECDSA) [13]	Digital signature based on elliptic curve key technology uses smaller keys than other public key technologies but may be encumbered by various	160 bits

6. EXPERIMENTING WITH DIGITAL SIGNATURES

6.1 CrypTool

CrypTool (http://cryptool.org/) is free e-learning, open source software, which enables analysis and application of cryptographic mechanisms. It is available in multiple languages and includes both classis cryptosystem like the Caesar cipher, the ADFGVX cipher, the double-column transposition (permutation), the Enigma encryption algorithm, etc. and modern cryptosystem like the RSA and AES algorithms, hybrid encryption, algorithms based on lattice reduction and elliptic curves, etc. The current version of CrypTool offers analysis and applications of various classic and modern cryptographic algorithms (encryption and decryption, key generation, secure passwords, authentication, secure protocols, etc.), visualization of several algorithms (Caesar, Enigma, RSA, Diffie-Hellman, digital signatures, AES, etc.), cryptanalysis of several algorithms (Vigenère, RSA, AES, etc.), Cryptanalytical measurement methods (entropy, n-grams, autocorrelation, etc.), related auxiliary methods (primality tests, factorization, base64 encoding, etc.), number theory tutorial, accompanying script with additional information about cryptology, etc.

6.2 .NET and Digital Signatures

Availability of cryptographic libraries in the form of Application Programming Interface (API) containing compiled cryptographic functions makes implementation of cryptographic functionalities (including digital signatures and certificates and smart cards) into a desktop or web application very fast and easy.

The .NET Framework Cryptographic Model provides implementations of many standard cryptographic algorithms. [14]. These algorithms are easy to use and have the safest possible default properties. In addition, the .NET Framework cryptography model of object inheritance, stream design, and configuration is extremely extensible.

The .NET Framework security system implements an extensible pattern of derived class inheritance. The hierarchy is as follows: i) Algorithm type class, such as *SymmetricAlgorithm*, *AsymmetricAlgorithm* or *HashAlgorithm*. This level is abstract, ii) Algorithm class that inherits from an algorithm type class; for example, *Aes*, *RC2*, or *ECDiffieHellman*. This level is also abstract, and, iii) Implementation of an algorithm class that inherits from an algorithm class; for example, *AesManaged*, *RC2CryptoServiceProvider*, or *ECDiffieHellmanCng*. This level is fully implemented. In the .NET Framework, the classes in the *System.Security.Cryptography* namespace manage many details of cryptography some of which are wrappers for the unmanaged Microsoft Cryptography API (*CryptoAPI*), while others are purely managed implementation. The following classes implement secret-key encryption algorithms: i) *AesManaged* (introduced in the .NET Framework version 3.5), ii) *DESCryptoServiceProvider*, iii) *HMACSHA1* (This is technically a secret-key algorithm because it represents message authentication code that is calculated by using a cryptographic hash function combined with a secret key, iv) *RC2CryptoServiceProvider*, v) *RijndaelManaged*, and, vi) *TripleDESCryptoServiceProvider*. The following classes implement public-key encryption algorithms: i) *DSACryptoServiceProvider*, ii) *RSACryptoServiceProvider*, iii) *ECDiffieHellman* (base class), iv) *ECDiffieHellmanCng*, v) *ECDiffieHellmanCngPublicKey* (base class), vi)

ECDiffieHellmanKeyDerivationFunction (base class), and, vii) *ECDsaCng*. The following classes implement digital signature algorithms: i) *DSACryptoServiceProvider*, ii) *RSACryptoServiceProvider*, iii) *ECDsa* (base class), and iv) *ECDsaCng*. Hashing algorithms are implemented through the classes *HMACSHA1* , *MACTripleDES* , *MD5CryptoServiceProvider* , *RIPEMD160* , *SHA1Managed* , *SHA256Managed*, *SHA384Managed* , and *SHA512Managed* .

The .NET Framework 3.5 supports the Suite B set of cryptographic algorithms published by the National Security Agency (NSA) (http://www.nsa.gov/ia/programs/suiteb_ cryptography/index.shtml). Suite B set includes Advanced Encryption Standard (AES) algorithm with key sizes of 128, 192, and 256 bits for encryption, Secure Hash Algorithms SHA-1, SHA-256, SHA-384, and SHA-512 for hashing, Elliptic Curve Digital Signature Algorithm (ECDSA), using curves of 256-bit, 384-bit, and 521-bit prime moduli for signing. The NSA documentation specifically defines these curves, and calls them P-256, P-384, and P-521. It also supports Elliptic Curve Diffie-Hellman (ECDH) algorithm, using curves of 256-bit, 384-bit, and 521-bit prime moduli for the key exchange and secret agreement.

Like .NET, Java programming language also supports cryptographic functions and defines Java Cryptography Architecture (JCA) that contains several classes and methods for implementing cryptographic functions [15].

6.3 Obtaining Digital Signature Certificates

Digital signature certificates are provided by several government and corporate companies for different purposes. Currently certificates are available for the purpose of i) Email / document signing (sigining e-mail messages and sigining documents like MS Word and *pdf* documents, ii) Wild Card SSL (securing multiple domains), iii) SSL Server (securing single web domain), iv) SSL Client (certifying a SSL Client), v) Mobile and Desktop Code Sigining (securing applications for mobile phones and desktop or Microsoft Authenticode), vi) Time Stamping (certifying time-stamping servers), and vii) IPSec (securing IP connections). Table 3 list some Root Certification Authorities who provide Digital signature certificates, some of them also provide free certificates for certain periods of time.

Table 3: Companies providing Digital Signature Certificates

Name of the Company	Website Address
Globsign	https://www.globalsign.com
Ascertia	http://www.ascertia.com
Entrust	http://www.entrust.com
VeriSign	https://www.verisign.com
CoSign	http://www.arx.com

Encryption-based access control is also dangerous because data can be irrevocably lost if the private key required to decrypt it is lost. For this reason, most practical systems store a copy of a resource's private key in a key repository that can be accessed by an administrator, and the copy itself is encrypted using another key. PKI has several complex managerial procedures that pertain to keys and certificate management. Several factors that may influence users in procuring certificate from a particular company and of particular type are: i) key and certificate creation algorithms, ii) Private Key protection offered, iii) certificate revocation procedure, iv) key backup and recovery, v) key and certificate update, vi) key history management, vii) method of distribution of certificates, viii) validity period of certificate, ix) cost of certificate, etc.

6.4 File Formats of Digital Signature Certificates

International Telecommunications Union (ITU) published the X.509 [16] recommendation in 1988 which among others defined format for embedding public key and other information into a certificate. A digital signature certificate file is often referred to with obscure acronyms like X.509, PKCS#7, DER or PEM which are actually labels that describe how a certificate is constructed and how it is stored. The data structure of the certificate is defined by standards such asX.509 and PKCS #7 which is turned into a file defined by standards such as DER and PKCS #12. The file name extension, such as *cer* and *p12* identifies the file format of a certificate. A file can contain a certificate and other security items such as entire certificate database, a CRL (Certificate Revocation List) or a private key. The attributes that make up a certificate are defined in a standard called X.509. Certificates are usually stored in a LDAP directory. Converting the certificate attributes and values to a Lightweight Directory Interchange Format (LDIF) format makes importing the file easy.

The PEM format is an ASCII format of a certificate. The certificates in PEM format is most common and usually has extensions such as *.pem*, *.crt*, *.cer* and *.key*. They are Base64 encoded ASCII files and contain "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" statements. Server certificates, intermediate certificates, and private keys can all be put into the PEM format. Apache Server and others use this format for certificates.

The DER format in contrast to PEM format is a binary form of a certificate. It has a file extension of *.cer* or *.der*. All types of certificates and private keys can be encoded in DER format. DER is typically used with Java platforms.

The PKCS#7 or P7B format is usually stored in Base64 ASCII format and has file extensions of *.p7b* or *.p7c*. P7B certificates contain "-----BEGIN PKCS7-----" and "-----END PKCS7-----" statements. A P7B file only contains certificates and chain certificates, not the private key. Platforms including Microsoft Windows and Java Tomcat support P7B files.

The PKCS#12 or PFX format is a binary format that stores server certificates, any intermediate certificates, and the private key in one encrypted file. PFX files usually have extensions such as *.pfx* and *.p12*. Windows machines use PFX files to import and export certificates and private keys.

7. DIGITAL CERTIFICATES IN INDIA

7.1 Excerpt of Indian Government Interoperability Guidelines for Digital Certificates

Digital Certificates Interoperability Guidelines, Release 2.4, updated on 14th June 2011 [17] issued by Controller of Certifying Authorities has issued Organizational and Certificate

Profile Guidelines to CA's. According to these guidelines, the India PKI organization structure consists of Controller of Certifying Authority (CCA) as the apex body and the Root Certifying Authority of India (RCAI) responsible for issuing digital certificates to licensed Certifying Authorities (CA). The CCA has allowed the creation of one level of sub-CAs that are part of same legal entity but issue sub-CA certificate issued by the CA. Guidelines pertaining to Interoperability of various fields and extensions of digital certificates are grouped into following: a) ***Mandatory***: mandatory fields and extinctions must be present in the specified format, b) ***Optional***: these fields are optional but if used must adhere to the guidelines, c) ***Special Purpose***: these fields are to be used under some special circumstances, d) ***Customizable***: these are non-standard extensions having different interpretations, e) ***Prohibited***: these fields must not be included in the certificates, and f) ***Reserved***: these extinctions are reserved for possible future use. Guidelines have been provided for each field and extension in terms of its nature, description, interpretation, usage, compliant standards, type, length and mandatory value. The Compliant international standards for these fields are one or more of the following: RFC 5280, RFC 3279, RFC 4055, RFC 4491, X.520. These guidelines provide certification guidelines for CA's, sub-CA's and the end users. These guidelines make it mandatory to use SHA-256 Hash Algorithm and 2048 bit RSA Key Digital Signature Certificates from 1st January, 2012. Generally digital certificates are issued to persons for the purpose of digital signature. However, some special uses of digital certificate exists for which the certificate fields and extensions vary. These certificates are termed by CCA as special purpose certificates. The special purpose certificates approved by CCA include: SSL certificate (Web server authentication certificate), System Certificate (for machine to machine authentication), Time Stamping Authority Certificate (certificates for the purpose of time stamping), Code Signing (certificates used to sign code), Encryption Certificate (certificates for data encryption/decryption or e-mail protection), and OCSP Responder Certificate. Certain guidelines for application developers have also been provided which also include recommendation of some commercial and Open Source PKI toolkits.

7.2 Controller of Certifying Authority

The Information Technology (IT) Act, 2000 [18] was enacted by the Indian Parliament in June, 2000. It was notified for implementation in October, 2000 with the issuance of Rules under the Act. The purpose of the Act besides others includes promotion of digital signatures for the growth of E-Commerce and E-Governance. For this purpose the IT Act enacted Controller of Certifying Authority (CCA) to govern and regulate digital signature certificates in India.

7.3 Root Certifying Authority of India (RCAI)

The CCA has established the RCAI under section 18(b) of the IT Act to digitally sign the public keys of CAs in the country. The RCAI is operated as per the standards laid down under the Act. The RCAI is operated using SmartTrust software. Authorized CCA personnel initiate and perform Root CA functions in accordance with the Certification Practice Statement of Root Certifying Authority of India. The term Root CA is used to refer to the total CA entity, including the software and its operations.

7.4 Root Certificate

A root certificate is a self-signed certificate. A root certificate, the top-most certificate of the tree, is based on the ITU-T X.509 standard. All certificates below the root certificate inherit the trustworthiness of the root certificate. The root certificate was issued on 13th June 2007 and is valid up to 4th July 2015. This certificate is intended for the following purposes: ensures the identity of a remote computer, proves your identity to a remote computer, protects e-mail messages, ensures software came from software publisher, protects software from alteration after publication, allows data to be signed with the current time, Allows data on disk to be encrypted, allows secure communication on the Internet, and all issuance policies.

7.5 RCAI Licensed Certification Authorities

Controller of certifying authority has currently licensed seven Certifying Authorities which are listed in table 4 who can issue digital signature certificates in India. These can appoint one multiple sub Certifying Authorities.

Table 4: Licensed Certifying Authorities of RCAI

Name of the Certifying Authority	Website Address
Sify Communications	www.safescrypt.com
IDRBT	idrbtca.org.in
National Informatics Centre	https://nicca.nic.in
TCS	www.tcs-ca.tcs.co.in
mtnlTrustline	www.mtnltrustline.com
(n)Code Solutions	www.ncodesolutions.com
e-Mudhra	http://www.e-Mudhra.com

7.6 Types of Certificates currently issued by Licensed Certification Authorities of RCAI

Class-1 Certificates: CA's and sub CA's do not facilitate strong authentication of the identity of the Subscriber; hence are not intended for, and shall not be relied upon, for commercial use where proof of identity is required. They are personal email Certificates that allow secure email messages. These Certificates can be used to: Digitally sign email, Encrypt email, Authenticate to Web Servers.

Class-2 Certificates: CA's and sub CA's are legally valid under the Indian IT Act 2000. Class-2 Certificates are issued as Managed Digital Certificates to employees/ partners/ affiliates/ customers of business and government organizations that are ready to assume the responsibility of verifying the accuracy of the information submitted by their employees/ partners/ affiliates/ customers. The Process of issuance of Class-2 certificate involves an organization who is functioning as an RA or an organization for whom a Sub-CA has been set up. This organization requests the issue of Digital Certificates for employees/ partners/ affiliates/ customers of the organization from TCS-CA. In the case of a Class-2 Certificate, the verification of details supplied with the request for a Digital Certificate is done by these organizations under the TCS-CA trust network.

Class-3 Certificates: issued by CA's and sub CA's are legally valid under the Indian IT Act 2000.These certificates are issued to individuals, companies and government organizations. They can be used both for personal and commercial purposes. They are typically used for electronic commerce applications such as electronic banking, electronic data interchange (EDI), and membership-based on-line services, where security is a major concern. The level of trust created by the Digital Certificate is based on the authentication procedures used by the CA to verify identity and the service guarantees offered by the CA to back up that authentication.

7.7 E-Governance Websites

Some government websites in India are using Digital Signatures for authentication of information as an initiative towards e-governess. A few of them are listed in table 5:

Table 5: Government websites using Digital Signatures

Name of the Organization	Website Address
Ministry of Corporate Affairs, Government of India	http://www.mca.gov.in/
e-Procurement Project of Government of Andhra Pradesh	http://www.eprocurement.gov.in/
Indian Customs and Excise Gateway	http://icegate.gov.in/
Karnataka Government e-Procurement System	https://eproc.karnataka.gov.in
Directorate General of Supplies and Disposal	http://www.dgsnd.gov.in/
Directorate General of Foreign Trade	http://dgft.gov.in/
Income Tax Department	https://incometaxindiaefiling.gov.in

8. CONCLUSION

Various encryption techniques are being used to ensure privacy and authentication of digital information. Digital signatures employs encryption, hashing and digital signature algorithms to ease its users attain all four desired properties namely privacy, integrity, authentication and non-repudiation for information security. There are several possible ways to use digital signatures each having its pros and cons. The most common use is digital signature certificate. Digital signature certificates are issued by several third party certifying authorities some of which are governed by governments. Certificates are issued for different purposes and exist in different file formats. All major programming languages provide cryptographic functions that support implementation of digital signatures and certificates in desktop and web applications. India has taken an initiative at government level to make use of digital signature to implement e-governess and e-commerce in a secure manner.

9. REFERENCES

[1] Gladney, H. 2007. Preserving Digital Information, Springer, ISBN 978-3-642-07239-0.

[2] Katz, J. 2010. Digital Signature, Springer, ISBN 978-0-387-27711-0.

[3] John E. Canava, (2001). Fundamentals of Network Security, Artech House, London, ISBN 1-58053-176-8.

[4] Schneier. B. (1996). Applied Cryptography: Protocols, Algorithms, and Source Code in C, Second Edition, John Wiley & Sons, Inc. ISBn: 0-471-12845-7.

[5] Zheng. Y., Imai. H. and Imai, H. (Ed.). 2007. Public Key Cryptography, Springer, ISBN: 9783540656449.

[6] CGI Group Inc. (2004). Public Key Encryption and Digital Signature: How do they work? CGI Group Inc. http://www.cgi.com/files/white-papers/cgi_whpr_35_pki_e.pdf.

[7] SHA, (1995). Federal Information Processing Standards Publication 180-1, available online at: http://www.itl.nist.gov/fipspubs/fip180-1.htm.

[8] R. Rivest (1992).The MD5 Message-Digest Algorithm, IETF RFC 1321, available online at: http://www.ietf.org/rfc/rfc1321.txt.

[9] RIPE (1995). Integrity Primitives for Secure Information Systems. Final Report of RACE Integrity Primitives Evaluation (RIPE-RACE 1040)," LNCS 1007, Springer-Verlag, 1995.

[10] Ross Anderson and Eli Biham (1996). Tiger: A Fast New Hash Function, Fast Software Encryption, Third International Workshop Proceedings, Springer-Verlag, pp. 89—97.

[11] FIPS (1996). Digital Signature Standard (DSS), FIPS PUB 186-3, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899-890, available online at: http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf.

[12] RSA (2002). RSA Cryptography Standard, RSA Security Inc, available online at: ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf.

[13] ANSI X9.62, (1999). Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), 1999.

[14] MSDN. (2010). .NET Framework Cryptography Model, MSDN Library. http://msdn.microsoft.com/en-us/library/0ss79b2x.aspx.

[15] Hongjie Zhu, Daxing Li. (2008). Research on Digital Signature in Electronic Commerce. In proceedings of the International MultiConference of Engineers and Computer Scientists, 2008 Vol I, IMECS 2008, 19-21 March, 2008, Hong Kong.

[16] ITU. (2008). Information Technology – Open systems interconnection – The Directory: Public-key and attribute certificate frameworks, ITU, http://www.itu.int/rec/T-REC-X.509-200811-I/en.

[17] CCA. (2009). Interoperability Guidelines for Digital Signature Certificates issued under Information Technology Act, CCA India, version 2.4 updated on 14th June 2011, http://cca.gov.in/rw/resource/dsc_guidelines_r2_4.pdf.

[18] IT ACT. (2000), The Information Technology Act, 2000, Government of India, http://www.mit.gov.in/sites/upload_files/dit/files/downloads/itact2000/itbill2000.pdf.