Digital Signature Algorithm

Gunjan Jain
Sr. Associate Technical Writer
Globallogic India Pvt Ltd, Noida
gnjn86@gmail.com

Abstract- People have traditionally used signatures as a means of informing others that the signature has read and understood a document. Digital signature in a document is bound to that document in such a way that altering the signed document or moving the signature to a different document invalidates the signature. This security eliminates the need for paper copies of documents and can speed the processes involving documents that require signatures. Digital Signatures are messages that identify and authenticate a particular person as the source of the electronic message, and indicate such person's approval of the information contained in the electronic message. Emerging applications like electronic commerce and secure communications over open networks have made clear the fundamental role of public key cryptosystem as unique security solutions. On the other hand, these solutions clearly expose the fact, that the protection of private keys is a security bottleneck in these sensitive applications. This problem is further worsened in the cases where a single and unchanged private key must be kept secret for very long time (such is the case of certification authority keys, and e-cash keys). They help users to achieve basic security building blocks such as identification, authentication, and integrity.

Keywords- Digital signature; PKI; CA; DSA algorithm.

I. INTRODUCTION

The Digital Signature Standard, created by the NIST, specifies DSA as the algorithm for digital signatures and SHA-1 for hashing. DSA is for signatures only and is not an encryption algorithm, although Schneier describes encryption mechanisms (ElGamel encryption and RSA encryption) based on DSA. DSA is a public key algorithm; the secret key operates on the message hash generated by SHA-1; to verify a signature, one recomputed the hash of the message, uses the public key to decrypt the signature and then compare the results.

The key size is variable from 512 to 1024 bits which is adequate for current computing capabilities as long as you use more than 768 bits. Signature creation is roughly the same speed as with RSA, but is 10 to 40 times (Schneier) as slow for verification. However, these numbers depend partially on the assumptions made by the bench marker. Since verification is more frequently done than creation, this is an issue worth noting.

The only known cracks (forgery) are easily circumvented by avoiding the particular module (prime factor of p - 1 where p is the public key) that lead to weak signatures. Schneier states that DSS is less susceptible to attacks than RSA; the difference is that RSA depends on a secret prime while DSA depends on a public prime -- the verifier can check that the prime number is not a fake chosen to allow forgery. It is possible to implement the DSA algorithm such that a "subliminal channel" is created that can expose key data and lead to forgeable signatures so one is warned not to used unexamined code. A Digital Signature is a checksum which depends on the time period during which it was produced. It depends on all the bits of a transmitted message, and also on a secret key, but which can be checked without knowledge of the secret key. A major difference between handwritten and digital signatures is that a digital signature cannot be a constant; it must be a function of the document that it signs. If this were not the case then a signature, could be attached to any document. Furthermore, a signature must be a function of the entire document; changing even a single bit should produce a different signature.

A digital signature algorithm authenticates the integrity of the signed data and the identity of the signatory. A digital signature algorithm may also be used in proving to a third party that data was actually signed by the generator of the signature. Is intended for use in electronic mail, electronic data interchange, software distribution, and other applications that require data integrity assurance and data origin authentication.The wireless protocols, like HiperLAN and WAP have specified security layers and the digital signature algorithm have been applied for the authentication purposes.

II. II. DIGITAL SIGNATURE

The term digital signature encompasses a great many variety of "signatures". Electronic signatures are simply an electronic confirmation of identity. This definition is deliberately broad enough to encompass all forms of electronic identification, from biometric signatures such as iris scans and fingerprints to non-biometric signatures, such as

digital signatures. Electronic signatures can be further subdivided into the highly secure and the insecure.

Digital signature must serve the same essential functions that we expect of documents signed by handwritten signatures, namely integrity, non-repudiation, authentication and confidentiality. In the digital realm, integrity means ensuring that a communication has not been altered in the course of transmission. It is concerned with the accuracy and completeness of the communication. The recipient of an electronic communication must be confident of a communication's integrity before she can rely on and act on the communication. Integrity is critical to e-commerce transactions, especially where contracts are formed electronically.

The process of digitally signing starts by taking a mathematical summary (called a *hash code*) of the check. This hash code is a uniquely-identifying digital fingerprint of the check. If even a single bit of the check changes, the hash code will dramatically change. The next step in creating a digital signature is to sign the hash code with your private key. This signed hash code is then appended to the check. How is this a signature? Well, the recipient of your check can verify the hash code sent by you, using your public key. At the same time, a new hash code can be created from the received check and compared with the original signed hash code. If the hash codes match, then the recipient has verified that the check has not been altered. The recipient also knows that only you could have sent the check because *only you have the private key that signed the original hash code.*

III. DIGITAL SIGNATURE TECHOLOGY

Digital signatures enable people to sign digital documents by providing the properties of a handwritten signature. They must fulfill the five compelling attributes of handwritten signatures as listed by (Schneier, 1996). He stated that the handwritten signatures are authentic, unforgivable, not reusable, unalterable, and cannot be repudiated. In the case of handwritten signatures, both the signature and the document are physical things, which makes it difficult for the "signer" to claim the signature is not their own. In order to provide a secure electronic signature scheme, these attributes must be satisfied. Electronic signature technologies include PINs, user identifications and passwords, digital signatures, digitized signatures, and hardware and biometric tokens. Therefore, it is important to distinguish between electronic and digital signatures. Digital signatures are a subset of electronic signature technologies that utilize keys and cryptographic algorithms for signing documents.

Digital signatures can be generated using various techniques; however, the only digital signature standard approved by National Institute for Standards and Technology (NIST) employs public key cryptography combined with a one-way hash function. This infrastructure, commonly referred to as the Public Key Infrastructure (PKI), requires each user to have a public-private key pair where the public key is available to the world while the private key is only known by the user. Figure 1 illustrates the use of PKI for generating digital signatures. The following is an example of a digital signature scenario. Bob (sender) wants to send Alice (receiver) a text message with a digital signature. First, Bob creates the text message to be signed and generates a hashed message using a message digest function (e.g., MD5, SHA1, etc.). A message digest function is a mathematical function that generates a 162-bit hash of the original message; this hash cannot be used to regenerate the original message. Therefore, the hashed message is secure and unique. Once Bob has the hashed message, he uses the public key digital signature algorithm and his private key to sign the hash to generate a digital signature for the specific document. Once Alice receives the digital signature, and the corresponding text message, she will need to calculate two separate values. First the hashed message of the received text is calculated using the same hashing algorithm. Then, once she has the hash value, she can now use the decryption algorithm with Bob's public key and digital signature to retrieve the signed hash. If she can decrypt the digital signature, this implies that Bob's private key was used to encrypt the hashed message. The final step for Alice is to compare the hash she calculated with the hash she retrieved from the decryption process. If these two hashed messages match, this implies that she received the original message Bob signed (thus preserving message integrity).

Key generation and distribution are the biggest challenges in deploying PKI. The solution is to use a trusted central authority – called a Certification Authority (CA) in PKI. CA is a trusted entity that accepts certificate applications from entities, authenticates applications, issues certificates to users and devices in a PKI, and maintains and provides status information about the certificates. If a CA is managing a large, geographically dispersed population, it may use Local Registration Authorities (LRAs), who provide direct physical contacts with subjects. These LRAs are especially required if the CA is issuing a high level of assurance for its certificates. Currently, there are four levels of assurance defined in the evolving government standard (PEC Solutions, 2000): Rudimentary; Basic; Medium; and High.

Traditionally, PKI architectures fall into one of three configurations: a single CA, a hierarchy of CA's, or a mesh of CA's. Each of the configurations

is determined by the fundamental attributes of the PKI: the number of CA's in the PKI, where users of the PKI place their trust (known as a user's trust point), and the trust relationships between CA's within a multi-CA PKI (Polk and Hastings, 2000). The most basic PKI architecture is one that contains a single CA, which provides the PKI services (certificates, certificate status information, etc.) for all the users of the PKI. All the users of the PKI place their trust in the sole CA of the architecture. Isolated CA's can be combined to form larger PKIs in two basic ways: using superior-subordinate relationships, or peer-to-peer relationships. In the former, which is called a hierarchical PKI, all users trust a "root" CA. There is single point of trust. The latter, a mesh PKI, connects CA's with a peer-to-peer relationship. A PKI constructed of peer-to-peer CA relationships is called a "web of trust". The Bridge Certification Authority (BCA) architecture was designed to address the shortcomings of the two basic PKI architectures, and to link PKIs that implement different architectures. Unlike a mesh PKI CA, the BCA does not issue certificates directly to users.

IV. DIGITAL SIGNATURE ALGORITHM

A digital signature is computed using a set of parameters and authenticates the integrity of the signed data and the identity of the signatory. An algorithm provides the capability to generate and verify signature. Signature generation makes use of a private key to generate a digital signature. Signature verification makes use of a public key, which corresponds to, but is not the same as, the private key. Each user possesses a private and public key pair. Public keys are assumed to be known to the public in general. Private keys are never shared. Anyone can verify the signature of a user by employing that user public key. Only the possessor of the user private key can perform signature generation.

A hash function is used in the signature generation process to obtain a condensed version of data, called a message digest. The message digest is then input to the digital signature algorithm to generate the digital signature. The digital signature is sent to the intended verifier along with the message. The verifier of the message and signature verifies the signature by using the sender's public key.

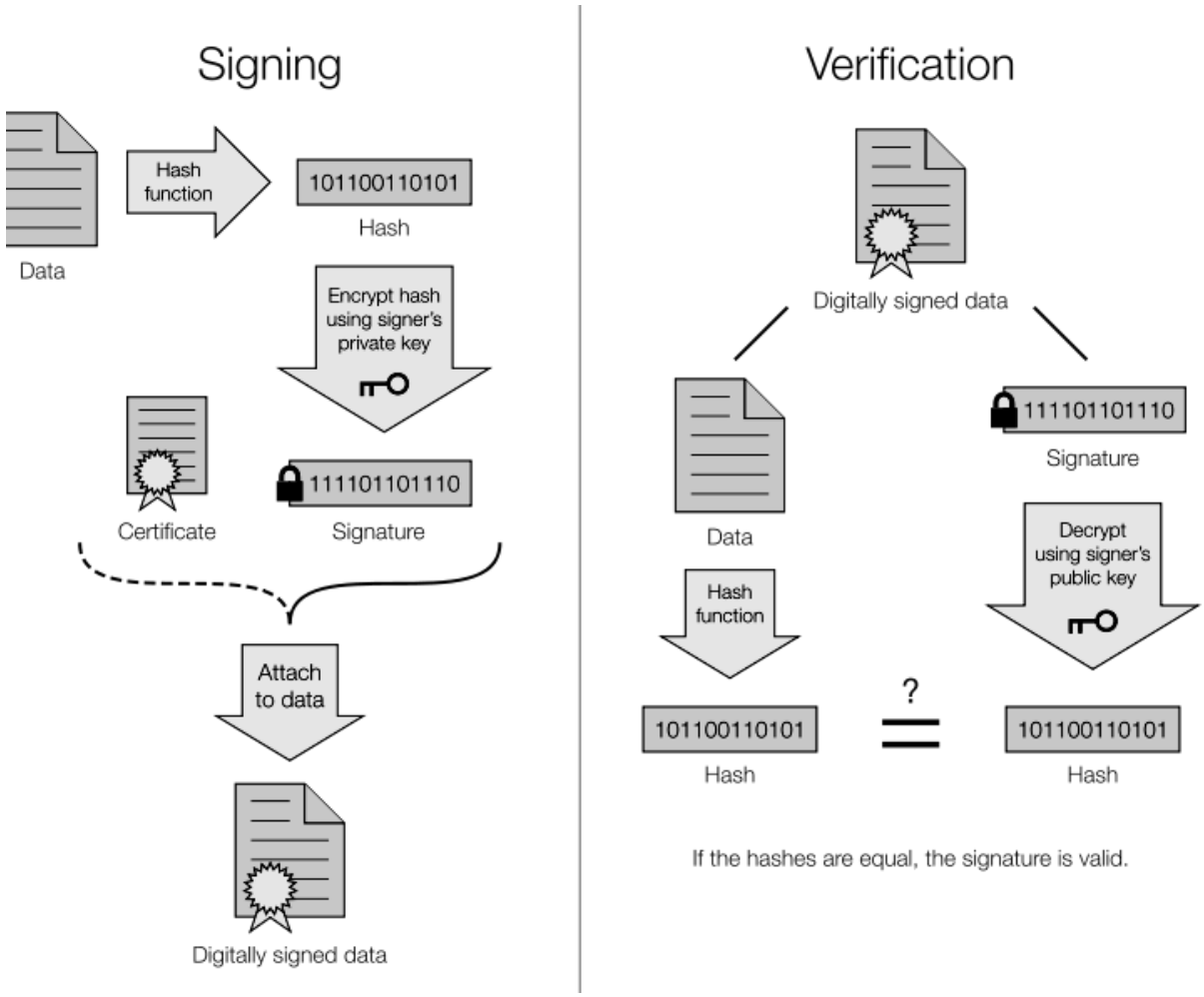

Fig.1 A Digital Signature Scheme

The same hash function must also be used in the verification process. The hash function is specified in a separate standard, the Secure Hash Standard, FIPS 180-1, and FIPS approved digital signature algorithms must be implemented with the Secure Hash Standard. Similar procedures may be used to generate and verify signatures for stored as well as transmitted data. The Digital Signature Standard (DSS) uses three algorithms for digital signature generation and verification. The Digital Signature Algorithm (DSA), the RSA digital signature algorithm as defined in ANSI X9.31 and Elliptic Curve digital signature algorithm (ECDSA) as define in ANSI.

V. DSA DESCRIPTION

A. DSA Parameters

A DSA digital signature is computed using a set of domain parameters, a private key x, a per message secret number k, data to be signed, and a hash function.

These parameters are defined as follows:

- p a prime modulus, where $2^{L-1} < p < 2^{L}$, and L is the bit length of p.
- q a prime divisor of $(p - 1)$, where $2N–1 < q < 2\ N$, and N is the bit length of q.
- g a generator of the subgroup of order q mod p, such that $1 < g < p$.
- x the private key that must remain secret; x is a randomly or pseudo randomly generated integer, such that $0 < x < q$, i.e., x is in the range $[1, q–1]$.
- y the public key, where $y = g^{x} \bmod p$.
- k a secret number that is unique to each message; k is a randomly or pseudo randomly generated integer, such that $0 < k < q$, i.e., k is in the range $[1, q–1]$.

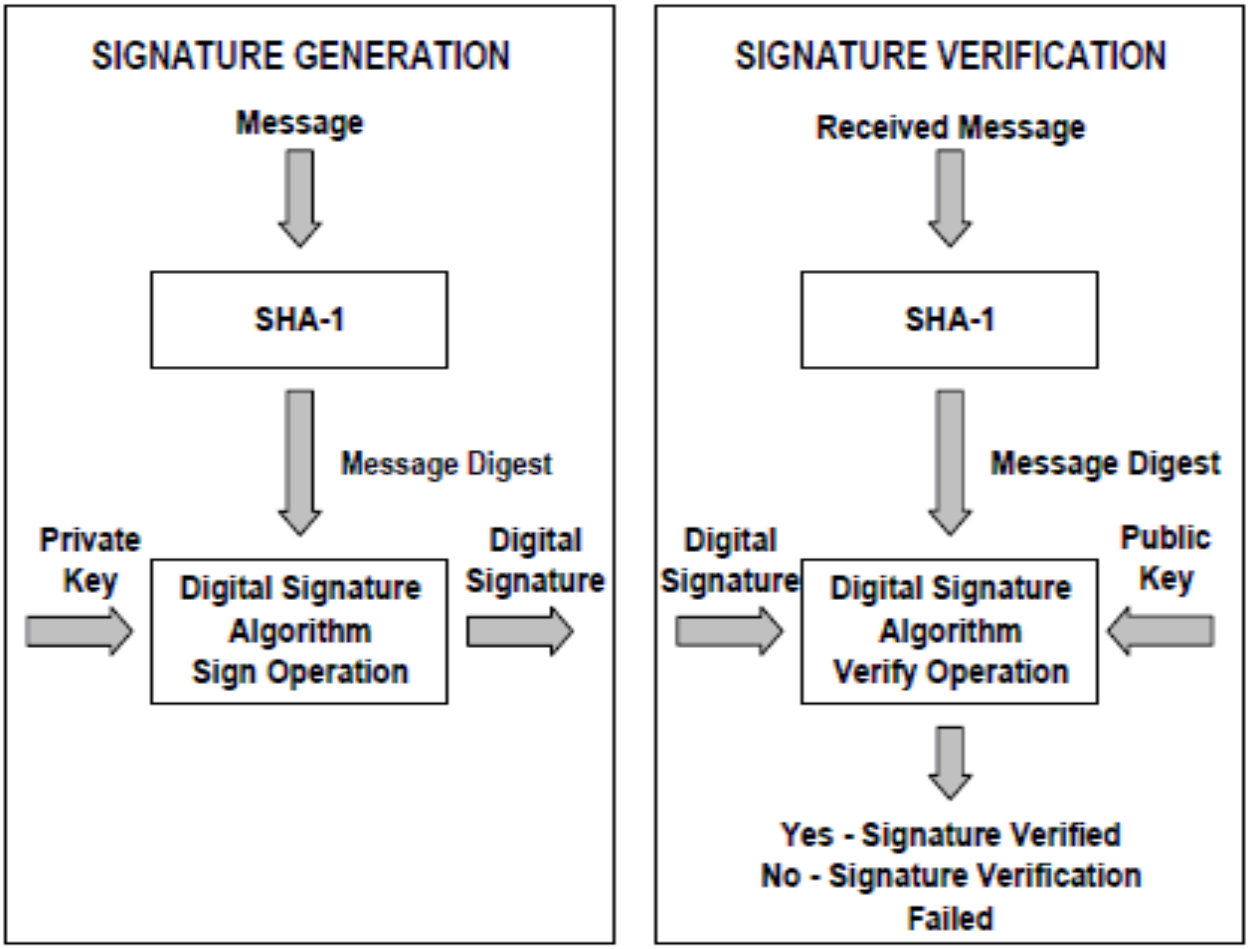

Fig.2 A Digital Signature Algorithm

B. Key Pairs

Each signatory has a key pair: a private key *x* and a public key *y* that are mathematically related
to each other. The private key will be used for only a fixed period of time in which digital signatures may be generated; the public key may continue to be used as long as digital signatures that were generated using the associated private key need to be verified.

Key generation has two phases:-

The first phase is a choice of algorithm parameters which may be shared between different users of the system:

- Choose an approved cryptographic hash function H.
- Decide on a key length L and N. This is the primary measure of the cryptographic strength of the key.
- Choose an N-bit prime q. N must be less than or equal to the hash output length.
- Choose an L-bit prime modulus p such that p–1 is a multiple of q.
- Choose g, a number whose multiplicative order modulo p is q. This may be done by setting g = h^ (p-1) /q mod p for some arbitrary h (1 < h < p-1), and trying again with a different h if the result comes out as 1.

The second phase computes private and public keys for a single user:

- Choose x by some random method, where 0 < x < q.
- Calculate y = g^x mod p .
- Public key is (p, q, g, y). Private key is x.

C. Signing

Let *N* be the bit length of *q*. Let **min** (*N*, *outlen*) denote the minimum of the positive integers *N* and *outlen*, where *outlen* is the bit length of the hash function output block.

The signature of a message *M* consists of the pair of numbers *r* and *s* that is computed according
to the following equations:

- $r = (g^k \bmod p) \bmod q$.
- *z* = the leftmost **min**(*N*, *outlen*) bits of **Hash**(*M*).
- $s = (k^{-1}(z + xr)) \bmod q$.

When computing *s*, the string *z* obtained from **Hash**(*M*) will be converted to an integer**.Ref.**[5]

D. DSA Signature Verification and Validation

Signature verification may be performed by any party using the signatory's public key. A signatory may wish to verify that the computed signature is correct, perhaps before sending the signed message to the intended recipient. The intended recipient verifies the signature to determine its authenticity.

The signature verification process is as follows:

1. The verifier **shall** check that $0 < r' < q$ and $0 < s' < q$; if either condition is violated, the signature **shall** be rejected as invalid.

2. If the two conditions in step 1 are satisfied, the verifier computes the following:
   $w = (s')^{-1} \bmod q$.
   z = the leftmost **min**(N, *outlen*) bits of **Hash**(M').
   $u1 = (zw) \bmod q$.
   $u2 = ((r')w) \bmod q$.
   $v = (((g)u1\ (y)u2) \bmod p) \bmod q$.
3. A technique is provided in Appendix C.1 for deriving $(s')^{-1}$ (i.e., the multiplicative inverse of $s' \bmod q$).The string z obtained from Hash(M′) shall be converted to an integer.

   If $v = r'$, then the signature is verified. For a proof that $v = r'$ when $M' = M$, $r' = r$, and $s' = s$,

4. If v does not equal r', then the message or the signature may have been modified, there may have been an error in the signatory's generation process, or an imposter may have attempted to forge the signature. The signature will be considered invalid. No inference can be made as to whether the data is valid, only that when using the public key to verify the signature, the signature is incorrect for that data.

VI. CONCLUSION

Digital signatures utilizing the public key cryptography system have every potential to achieve the same level of legal recognition as handwritten signatures. However, the main obstacle at present is in the functional element of non-repudiation. This element, unlike the other three elements of handwritten signatures discussed, cannot be achieved by technology alone. Assistance is required from the law to help it attain the functional element of non-repudiation. Once non-repudiation has been achieved, then and only then, can electronic commerce be expected to be successfully taken up. A certification authority in turn can be validated by higher certification authorities, thus creating a certificate chain. Hence, the trustworthiness of a certification authority may depend on its reputation in traditional business transactions, or, it may be a subscriber of a higher certification authority, and use the certificate of the higher certification authority to reassure subscribers and relying parties that it is not a bogus certification authority. The certification authority at

the pinnacle of the certification authority hierarchy is known as a root certification authority and it issues root certificates. The root certification authority self-authenticates for purposes of determining the validity of the certificates.

REFERENCES

[1] S. W. Changchien and M. S. Hwang, "A batch verifying and detecting multiple RSA digital signatures," *International Journal of Computational and Numerical Analysis and Applications*, vol. 2, no. 3, pp. 303-307, 2002.

[2] Dorothy E. R. Denning, *Cryptography and Data Se curity*. Massachusetts: Addison-Wesley, 1982.

[3] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms" *IEEE Transactions on Information Theory*, vol. IT-31, pp. 469-472, July 1985.

[4] L. Harn, "Batch verifying multiple DSA-type digital signatures" *Electronics Letters*, vol. 34, no. 9, pp. 870-871, 1998.

[5] L. Harn, "Batch verifying multiple RSA digital signatures" *Electronics Letters*, vol. 34, no. 12, pp. 1219- 1220, 1998.

[6] M. S. Hwang, I. C. Lin, and K. F. Hwang, "Cryptanalysis of the batch verifying multiple RSA digital signatures," *Informatica*, vol. 11, no. 1, pp. 15-19, 2000.

[7] M. S. Hwang, C. C. Chang, and K. F. Hwang, "An ElGamal-like cryptosystem for enciphering large messages," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp. 445-446, 2002.

[8] M. S. Hwang, C. C. Lee, and Y. C. Lai, "Traceability on RSA-based partially signature with low com putation," *Applied Mathematics and Computation*, vol. 145, no. 2-3, pp. 465{468, 2003.

[9] M. S. Hwang, C. C. Lee, and Eric J. L. Lu, "Cryptanalysis of the batch verifying multiple DSA-type digital signatures," *Pakistan Journal of Applied Sciences*, vol. 1, no. 3, pp. 287-288, 2001.

[10] M. S. Hwang, C. C. Lee, and Y. L. Tang, "Two simple batch verifying multiple digital signatures," in *The Third International Conference on Information and Communication Security (ICICS2001)*, pp. 13-16, Xian, China, 2001.

[11] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120{126, 1978.

[12] Z. Shao, "Batch verifying multiple DSA-type digital signatures," *Computer Networks*, vol. 37, no. 3-4, pp. 383-389, 2001.

[13] S. F. Tzeng, C. Y. Yang, and M. S. Hwang, "A new digital signature scheme based on factoring and discrete logarithms," *International Journal of Computer Mathematics*, vol. 81, no. 1, pp. 9-14, 2004.

[14] Bellare M, Miner S K. A Forward-secure Digital Signature Scheme[C] Proc. of Advances in Cryptology-CRYPTO. 1999:431-448.

[15] Krawczyk H. Simple Forward-Secure Signatures from any Signature Scheme[C]. Proc. of the 7th ACM Conference on Computer and Communication Security. 2000-10: 1-4.

[16] Malkin T, Micciancio D, Miner S. Effcient Generic Forward-secure Signatures with an Unbounded Number of Time Periods[C]. Proc. Of Advances in Cryptology-EUROCRYPT. 2002.

[17] J2EE Programme Guide *by Subrahmanyam Allamaraju (America), translated by Shuqi Ma* [M]. Beijing: Electronic Industry Press, 2002.

[18] W. Stallings, *Cryptography and Network Security*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.

[19] M. Bishop, *Introduction to Computer Security.* Reading, MA: Addison-Wesley, 2005.

[20] J. Feghhi and P. Williams, *Digital Certificates: Applied Internet Security* 1st ed. Reading, MA: Addison-Wesley, 1999.

[21] Department of Veterans Affairs (2003) Public Key Infrastructure Project, *Department of Veterans Affairs*, http://www.va.gov/proj/vapki/default.htm

[22] National Institute of Standards and Technology (NIST), Digital Signature Standard, FIPS PUB 186-2, http://csrc.nist.gov/publications/fips/fips186-2/fips186-2.pdf

[23] Bruce Schneier, Applied Cryptography – Protocols, Algorithms and Source Code in C, Second Edition, John Wiley and Sons, New York, 1996.