A Survey of Cryptographic Protocols

Mervat Mikhail*[1], Yasmine Abouelsoud*[2] and Galal El Kobrosy*[3]

*1 Assistant lecturer
Engineering Mathematics Department
Alexandria University, Alexandria 21544, EGYPT
mervatmekhaeil@yahoo.com

*2 Assistant professor of Engineering Mathematics
yasmineabosoud@gmail.com

*3 Professor of Engineering Mathematics
elkobrosy@yahoo.com

Abstract

With the recent acceleration in research into cryptography, we consider this a suitable moment to compare different cryptosystems. In this paper, a survey on cryptographic standards and algorithms is presented. First, the concept of cryptography is explained as well as its most common practical problems. Second, classification of cryptographic algorithms according to key management scheme is provided. A literature review of the most famous protocols together with some tables of comparison is presented. The main goal of this survey is to answer the question "*What are the differences between these cryptographic schemes from a practical viewpoint?*" The aim of this paper is to identify the distinguishing features of each. In doing so, we highlight the important questions to be asked when weighing up the benefits and drawbacks of each scheme.

Keywords: cryptographic protocols, symmetric key cryptosystem, public key cryptosystem, stream ciphers; block ciphers, identity-based cryptosystems ID-PKC, certificateless public-key cryptosystem CL-PKC

1 Introduction

Secure communication in a strongly interconnected world has become an impelling need. With the advent in communication technologies and the Internet, new challenges and applications in the field of security have emerged.

Cryptography is the art of keeping messages secure. In addition to providing confidentiality, cryptography is often asked to do other jobs such as authentication, integrity and non-repudiation. *Authentication* means that it should be possible for the receiver of a message to ascertain its origin; an intruder should not be able to masquerade as someone else. *Integrity* means that it should be possible for the receiver of a message to verify that it has not been modified in transit; an intruder should not be able to substitute a false message for a legitimate one. *Non-repudiation* means that a sender should not be able to falsely deny later that he sent a message.

To achieve the previous security goals, some secret piece of information should be shared, which is referred to as a cryptographic key. The problem of distributing the secret keys for cryptographic algorithms is known as key management.

Cryptographers often classify encryption algorithms according to the type of transformation and keys; that is, the key management scheme employed. Each class solves a different set of cryptographic problems. Some classes require that parties first agree on a secret key by secure means that is separate from the normal communication protocol; others do not have this limitation. The algorithms are classified accordingly into: Secret-Key (symmetric key) Cryptosystems (SKC) and Public-Key (asymmetric key) Cryptosystems (PKC). In the former, the sender and receiver both use the same secret key, one could think of a symmetric algorithm as *a safe*. Someone with the key can open the safe, put a document inside, and close it again. Someone else with the key can open the safe and take the document out. While in the latter, the receiver only is in possession of the secret key and publishes the corresponding public key. The public key is obtained by a suitable one-way trapdoor transformation to the secret key. A one-way trapdoor function is a function which is difficult to find its inverse unless given the trapdoor information (key). It is as if someone turned the cryptographic safe into *a mailbox*. Putting mail in the mail box is analogous to encrypting with the public key; anyone can do it. Just open the slot and drop it in. Getting mail out of a mailbox is analogous to decrypting with the private key. Generally, it's hard; you need welding torches. However, if you have the secret (the physical key to the mailbox), it's easy to get mail out of a mailbox.

Symmetric key algorithms are known for their computational efficiency; however, there are several problems in such cryptosystems. First of all, keys must be distributed in secret; that is, there is a need for private channels. Moreover, assuming a separate key is used for each pair of users in a network, the total number of keys increases rapidly as the number of users increase. A network of n users requires n(n -1) /2 keys. Furthermore, since both the sender and receiver share the same secret key, symmetric-key cryptosystems are not suitable for achieving authentication.

Symmetric-key algorithms are further divided into two classes; which are stream ciphers and block ciphers. Stream ciphers are an important class of symmetric key encryption algorithms [1]. They encrypt individual characters of a plaintext message one at a time using an encryption transformation which varies with time. By contrast, block ciphers tend to simultaneously encrypt groups of characters of a plaintext message using a

fixed encryption transformation.

Public-key cryptography solves the key-management problem with symmetric-key cryptosystems. With no prior arrangements, the transmitter can send a secure message to the receiver. An eavesdropper, listening in on the entire exchange, has access to the receiver's public key and a message encrypted in that key, but cannot recover either receiver's private key or the message. Moreover, public key cryptosystems offer a good method for providing authentication. Furthermore, the invention of public key cryptosystems gave rise to a new and remarkable idea, which is the concept of digital signature. The digital signature is the electronic analogue of the handwritten signature. A signer can digitally sign a document with a secret key (Private Key), and generates a signature on that document. The signer then sends the generated signature, a document and its public key to any verifier. Therefore, a verifier can check the validity of the signature with the corresponding public key.

Traditionally, any involved party must register his public key with a central authority, which is known as the Certificate Authority (CA). The CA issues digital certificates providing the link between a user's identity and its public key. Actually, the certificate is nothing but a digital signature generated by the CA for this information.

Another famous type of public-key encryption is the Identity Based Encryption (IBE), in which the public key of a user is some unique information about the identity of the user (e.g. a user's email address) and the private key is generated by a key generation center.

Certificateless public-key encryption (CL-PKE) [2, 3] is a form of public-key encryption that is designed to eliminate the disadvantages of both traditional public-key encryption scheme and identity-based encryption. Unlike public-key encryption, there is no requirement for digital certificates or a public-key infrastructure. Unlike identity-based encryption, the trusted third party need not be given the ability to decrypt ciphertexts intended for users. Certificateless public-key encryption integrates together the benefits of traditional PKI-based public-key encryption and identity-based encryption. They provide security without the need for a public key to be signed by a certificate authority. Also, they remain secure against attacks made by any third party (including a key generation center or a certificate authority)

The rest of the paper is organized as follows. In the next section, symmetric-key cryptosystems are explained together with a comparative study. Then, traditional, certificate-based public key cryptosystems are reviewed in Section 3. Identity-Based Cryptosystems and Certificateless Public-Key Encryption CL-PKE are examined in Section 4 and Section 5. Finally, the last section concludes the paper.

2 Symmetric key cryptosystems

2.1 Stream ciphers

A stream cipher is a type of symmetric key cryptosystem. The idea of stream ciphers was inspired from the famous cipher called the One-time Pad [1]. This cipher is based on XORing the message bits and the key bits as $c_i = m_i \oplus k_i$, where m_i are the message bits and k_i are the corresponding key bits. Feedback shift registers, in particular linear feedback shift registers (LFSRs), are the basic building blocks in most stream ciphers. However, algorithms such as RC4 [4] and SEAL [5] are examples of software-oriented implementations of stream ciphers not based on LFSRs.

The latest classification [6] divides stream ciphers into three main categories *Hardware-based* stream ciphers, *Software-Based* stream ciphers and *Hybrid designs* of stream ciphers.

The classification aims to look at stream ciphers from the implementation perspectives. The in-depth classification of hardware-based stream ciphers include: FCSR/NLFSR-based, clock control based and LFSR-based stream ciphers. On the other hand software-based stream ciphers include: T-function based, block cipher-based, S-box-based and simple logical and arithmetic operations. The last category, the hybrid designs, includes those stream ciphers which depend on the combination of both hardware and software techniques in their constructional designs. A comprehensive classification of stream ciphers is described in **Table 1**.

Table 1 Stream ciphers classifications

<table>
<tr><td colspan="4">Stream ciphers</td></tr>
<tr><td colspan="3">Hardware-based</td><td>Software-Based</td><td>Hybrid Design</td></tr>
<tr><td colspan="3">Shift register</td><td>T-Function</td><td rowspan="5"></td></tr>
<tr><td>LFSR</td><td>NLFSR/ FCSR</td><td>Clock control</td><td>S-Box</td></tr>
<tr><td colspan="2">Shrinking & self Shrinking</td><td>Stop & Go</td><td>Block Cipher</td></tr>
<tr><td colspan="2">Summation</td><td>Cascades</td><td rowspan="2">Simple logical & Arithmetic operations</td></tr>
<tr><td colspan="2">Boolean Function</td><td>ABSG</td></tr>
</table>

Table 2 Best known examples of stream ciphers

Stream Cipher	First published	Classification
RC4	Rivest 1987	Software based stream cipher
FISH	Siemens 1993	Software based stream cipher
PANAMA	Daemen 1998	Hash function and Software based stream cipher
SCREAM	Halevi 2002	S-Box
RABBIT	Boesgaard 2003	Simple logical & Arithmetic operations
SNOW	Pre-2003	Combination of LFSR and a Finite State Machine (FSM)
Grain	Pre-2004	LFSR and NLFSR
Py	Pre-2004	Software based stream cipher
VEST	O'Neil 2007	NLFSR and T function

The best known stream cipher algorithms with some

brief information about creation date, effective key length and complexity are shown in **Table 2**.

2.2 Block ciphers

Block ciphers is the second type of symmetric key ciphers. The simplest techniques for encrypting a block of symbols are substitution and permutation. Substitution replaces a symbol by another, while permutation moves the symbols of a block around. Neither substitution nor permutation work very well by themselves. Frequency analysis, using the relative commonness of letters, pairs of letters, etc., is a strong tool against both. However, a proper combination of simple operations such as $\oplus$, substitution and permutation produces a cryptosystem whose strength is greater than the sum of its component.

An *iterated block cipher* [7] is a cryptosystem on a block of symbols that sequentially repeats an internal function called a *round*. Iteration is a natural way to proceed because that yields an algorithm with a small set of instructions, an important issue for hardware implementations.

In *Feistel ciphers*, the 2*t*-bit input block is split into *t*-bit halves L_0, R_0 and proceed as follows:

In the i^{th} round, the right half of the previous round becomes the new left half

$$L_i \leftarrow R_{i-1}$$

While the new right half R_i is the XOR of the previous left half and a preferably non-linear function of a round sub-key K_i and the previous right half.

$$R_i \leftarrow L_{i-1} \oplus f(R_{i-1}, K_i)$$

The inverse process is pretty similar to the above construction. Working backwards,

$$\mathrm{R_{i-1}} \leftarrow \mathrm{L_i}$$

$$\mathrm{L_{i-1}} \leftarrow \mathrm{R_i} \oplus \mathrm{f(R_{i-1}, K_i)}$$

regardless of the round function f used.

Decryption is actually the algorithm run in reverse with sub-keys used in the opposite order. In order to make decryption a genuine inverse of encryption, the final round of a Feistel cipher switches the ciphertext to $(\mathrm{R_r}, \mathrm{L_r})$.

DES (Data Encryption Standard) is a 16-round Feistel cipher [8]. Encrypting ordinary text in DES begins by grouping the text into 64-bit blocks. An initial permutation is applied in the beginning and its inverse is applied in the end. In DES, the sub-keys selection or key schedule starts by splitting the 56-bit key into two 28-bit halves and then rotating each half one or two bits (one bit in rounds 1,2,9 and 16; two bits otherwise). The two halves are put together and then 48 particular bits are chosen and put into some prescribed order. The rotation ensures that a different subset of key bits is used for each of the sixteen rounds. Due to cryptanalytic attacks against DES, such as linear [9] and differential attacks [10], the need for a new encryption standard arose.

The Advanced Encryption Standard (AES) was announced by the National Institute of Standards and Technology (NIST) [11]. Nowadays, AES has become one of the most popular algorithms used in symmetric key cryptography. It is an iterated block cipher with block size 128 bits. The cipher key is 128, 192 or 256 bits in length. Unlike DES (the predecessor of AES), AES is a substitution-permutation network, that is, a series of linked mathematical operations, not a Feistel network. AES is fast in both software and hardware, is relatively easy to implement and requires little memory. In 2011, Amber Jain started a careful study of specifications, variations of 5 symmetric block cipher algorithms (Blowfish, Camellia, CAST-128, DES and IDEA) [12]. During this investigation, notable design guidelines were collected to reach to a comparison of symmetric block cipher algorithms, which is depicted in **Table 3**.

Block ciphers has many modes of operation which describe how to repeatedly apply a cipher's single-block operation to securely transform amounts of data larger than a block such as ECB (electronic codebook), CBC (cipher block chaining), CFB (cipher feedback), OFB (output feedback). Summary of Block Cipher Modes of operation is provided in [13].

Table 3 Best known examples of Block Ciphers

Property	Blowfish	Camellia	CAST-128	DES
Key length	8–448 bits in steps of 8 bits	128, 192 or 256 bits	40 to 128 bits	56 bit
Variable key length	Yes	Yes	Yes	No
Type	Feistel	Feistel	Feistel	Balanced Feistel
Block Size	64 bits	128 bit	64 bits	64 bit
Avalanche	Yes	Yes	Yes	Yes
Coding effort needed	Comparatively easy	Comparatively easy	Comparatively easy	Comparatively Difficult
Weak keys	Yes	Probably yes	Probably yes	Yes
S box	Yes	Yes	Yes	Yes
Precomputable subkeys	Yes	Yes	Yes	Yes
Rounds	16 (feistel)	18 or 24 (feistel)	12 or 16 (feistel)	16 (feistel)
Current state	Secure	Secure	Insecure	Highly insecure

2.3 Stream ciphers versus block ciphers

In this subsection, the differences between two approaches are summarized.

(a) Idea

Stream ciphers partition the text into small blocks (e.g. 1 bit) and let the encoding of each block depend on many previous blocks. While the block cipher partition

the text into relatively large (e.g. 128 bits) blocks and encode each block separately.

(b) *Key*
In stream ciphers, for each block, a different key is generated. While in block ciphers the same key is used for each block.

(c) Hard ware speed and complexity
Stream ciphers are faster in hardware than block ciphers and have less complex hardware circuits so it is more suitable in hard ware implementation while block ciphers are more suitable in software implementations.

(d) Integrity & authentication
Stream ciphers do not provide integrity protection or authentication while some block ciphers (depending on mode) can provide integrity protection, in addition to confidentiality.

(e) Possible reasons to prefer stream ciphers today
- A smaller footprint in low-end hardware implementations
- Higher encryption speed
- Smaller input/output delay
- Simpler protocols for handling small or variable sized inputs

(f) Possible reasons to prefer block ciphers today
- Availability of standardized schemes
- More versatile building block
- Better understanding of security issues
- Better covered by textbooks and courses

2.4 Strengths and Weaknesses of symmetric-key cryptography

Strengths: Private keys are robustly resistant to brute force attacks. While the one-time pad , which combines plaintext with a random key, holds secure in the face of any attacker regardless of time and computing power, symmetric-key algorithms are generally more difficult to crack than their public-key counterparts. Additionally, secret-key algorithms require less computing power to be created compared to equivalent private keys in public-key cryptography.

Weakness: The biggest obstacle in successfully deploying a symmetric-key algorithm is the necessity for a proper exchange of private keys. This transaction must be completed in a secure manner.

Another problem concerns the compromise of a private key; every participant has an identical private key. As the number of participants in a transaction increases, both the risk of compromise and the consequences of such a compromise increase dramatically. Each additional user adds another potential point of weakness that an attacker could take advantage of. If such an attacker succeeds in gaining control of just one of the private keys in this world, all users, whether hundreds or more of them or only a few, are completely compromised.

3 Traditional Public key cryptosystems

In 1976, Whitfield Diffie and Martin Hellman changed that paradigm of cryptography forever [14]. They described public-key cryptography. They used two different keys one public and the other private. It is computationally hard to deduce the private key from the public key. Anyone with the public key can encrypt a message but not decrypt it. Only the person with the private key can decrypt the message.

Mathematically, the process is based on trap-door one-way functions. Encryption is the easy direction. Instruments for encryption are the public key and the message; anyone can encrypt a message. Decryption is the hard direction. It's made hard enough that people with Cray computers and thousands (even millions) of years couldn't decrypt the message without the private key. With that secret, decryption is as easy as encryption.

3.1 Hard Computational Problems

Assume G is a multiplicative cyclic group (large prime order subgroups of groups $\mathbf{Z}_p^{*)}$ and g is a generator of G, then from the definition of cyclic groups, we know every element h in G can be written as g^x for some x

- *Discrete Logarithm Problem (DLP)*
  Given g, h= g^x , what is the value of x ?
- *Computational Diffie-Hellman Problem (CDHP)*
  Given an element g and the values of g^x and g^y, what is the value of g^{xy} ?
- *The Integer-Factorization (IF) Problem*
  Given a positive integer n, find its prime factors decomposition such that one can write $n = {q_1}^{e_1} {q_2}^{e_2} \dots {q_k}^{e_k}$ where $q_i's$ are pair wise distinct primes and $e_i \geq 1$.

The most two famous public key encryption algorithms are RSA and ElGamal. RSA cryptosystem was invented by Rivest, Shamir and Adelman [15], whose security relies on the hardness of the integer factorization problem. ElGamal cryptosystem was invented by Taher ElGamal [16] and its security relies on the hardness of the discrete logarithm problem over finite fields.

3.2 Strengths and Weaknesses of public-key cryptography

Strengths: The asymmetric nature of public-key cryptography allows it a sizable advantage over symmetric-key algorithms. The unique private and public keys provided to each user allow them to conduct secure exchanges of information without first needing to devise some way to secretly swap keys.

Weakness: Keys in public-key cryptography, due to their unique nature, are more computationally costly than their counterparts in secret-key cryptography. Symmetric keys must be many times longer compared to keys in secret-cryptography in order to ensure equivalent security.

Keys in asymmetric cryptography are also more vulnerable to brute force attacks than in secret-key cryptography. There exist algorithms for public-key cryptography that allow attackers to crack private keys faster than a brute force method would require. The widely used and pioneering RSA algorithm is indeed susceptible to attacks in less than brute force time. While generating longer keys in other algorithms will usually prevent a brute force attack from succeeding in any meaningful length of time, these computations

become more intensive. These longer keys can still vary in effectiveness, depending on the computing power available to an attacker.

Public-key cryptography is also vulnerable to various attacks, such as the man-in-the-middle attack. In this situation, a malicious third party intercepts a public key on its way to one of the parties involved. The third party can then instead pass along his or her own public key with a message claiming to be from the original sender. An attacker can use this process at every step of an exchange in order to successfully impersonate each member of the conversation without any other parties becoming aware of this deception.

4 Identity based cryptosystems

The concept of identity-based cryptosystems is due to Shamir [17]. Such a scheme has the property that a user's public key is an easily calculated function of his identity, while a user's private key can be calculated for him by a trusted authority, called private key generator (PKG). The ID-based public key cryptosystem can be an alternative for certificate-based public key infrastructure (PKI), especially when efficient key management and moderate security are required. The public key distribution problem is eliminated by making each user's public key derivable from some known aspect of his identity, such as his email address. The first ID-based encryption was proposed by Boneh and Franklin [18] in 2001 that uses bilinear pairing as well as Cha-Cheon's efficient ID-based signature scheme [19].

4.1 Structure of Identity-Based Cryptosystems

An identity-based encryption scheme is specified by four randomized algorithms: Setup, Extract, Encrypt and Decrypt.

Setup: It takes a security parameter k and returns the system parameters *params* and a master key *mk*. Intuitively, the system parameters will be publicly known, while the master key will be known only to the Private Key Generator (PKG).

***Extract* (*Key Generation*):** It receives as input the system parameters, the master secret key mk and an arbitrary user identifier string $ID \in \{0,1\}^*$. It returns a private key d_{ID}, which is then delivered to the user through a private channel. Here, ID is an arbitrary string that will be used as a public key and d_{ID} is the corresponding private decryption key. The Extract algorithm extracts a private key from the given public key.

Encrypt: It takes as input *params*, recipient's identifier ID and a message m. It returns a ciphertext σ.

Decrypt: Its inputs include $params$, a ciphertext σ and a private key d_{ID}. It returns the decrypted text m. Again, these algorithms must satisfy the standard consistency constraint, namely when d_{ID} is the private key properly generated by the Extract algorithm when it is given ID as the public key.

4.2 Advantages and disadvantages of ID-based encryption

Advantages: It makes maintaining authenticated public key directories unnecessary. Instead, a directory for authenticated public parameters of PKG's is required which is less burdensome than maintaining a public key directory since there are substantially fewer PKGs than total users. In particular, if everyone uses a single PKG, then everyone in the system can communicate securely and users need not perform online lookup of public keys or public parameters

Disadvantages:

(a) the PKG knows the receiver's private key, i.e. key escrow is inherent in the system which for some applications may be a serious problem

(b) the receiver has to authenticate himself to its PKG in the same way as he would authenticate himself to a certifying authority (CA)

(c) the receiver's PKG requires a secure channel to send the receiver his private key

(d) the receiver has to publish his PKG's public parameters and the sender must obtain these parameters before sending an encrypted message to the receiver

5 Certificateless public key cryptosystems

It is a variant of ID-based cryptography intended to prevent the key escrow problem. Ordinarily, keys are generated by a certificate authority or a key generation center (KGC) who is given complete power and is implicitly trusted. To prevent a complete breakdown of the system in the case of a compromised KGC, the key generation process is split between the KGC and the user. The KGC first generates a key pair, where the private key is now the partial private key of the system. The remainder of the key is a random value generated by the user, and is never revealed to anyone, not even the KGC. All cryptographic operations by the user are performed by using a complete private key which involves both the KGC's partial key, and the user's random secret value. One disadvantage of this is that the identity information no longer forms the entire public key.

To encrypt a message to another user, three pieces of information are needed: 1) the other user's public key and 2) identity, and also 3) the third party's public information. To decrypt, a user just needs to use their private key.

These are realized by having two separate public/private key pairs:

(a) A standard public/private key pair generated by the receiver. The private key is called *secret value* to stay clear from confusion with the full private key of the receiver. The public key is made public but inevitably is not certified by a certificate authority.

(b) An identity-based public/private key pair comprising of the receiver's digital identifier, and the associated identity-based private key provided by a key generation center. This private key is called *partial private key*.

To encrypt a plaintext, the sender utilizes the receiver's digital identifier and the receiver's public key. To decrypt a ciphertext, the receiver uses the secret value generated by him and the partial private key provided by the key generation center.

Certificateless cryptography had a really fast evolution, with several schemes being introduced for encryption [2] and digital signature [2]. Also, a few alternative security models for certificateless encryption have been

presented that are, to a great extent, weaker than the original model of Al-Riyami–Paterson [2]. In 2008, Dent reviewed almost all the security models for certificateless encryption [20]. The notion of a certificateless public-key encryption scheme was first introduced by Al-Riyami and Paterson [2, 3].

There are three different architectures for CL-PKC:

(a) **AP Formulation**: In the original Al-Riyami and Paterson (AP) formulation [21, 22], the receiver can generate their public key at any time. This means that the receiver can publish their public key before receiving their partial private key from the key generation centre.

(b) **BSS Formulation**: In the Baek, Safavi-Naini and Susilo (BSS) formulation [23], the receiver can only generate their public key after receiving the partial private key. The partial private key is obtained via a single secure message from the key generation centre.

(c) **LK-Formulation**: In the Lai and Kou (LK) formulation [24], the receiver can only generate their public key after completing a protocol with the key generation centre.

6 Conclusion

In this paper, the different types of cryptosystems available to date have been reviewed. We have provided several comparison tables between different cryptographic concepts and algorithms, in addition to comparing the strengths and weaknesses of different schemes. Since there is no absolutely perfect encryption scheme that suits all situations, a comparative study is very important for most researchers who want to know the most appropriate encryption scheme for use in their work.

References

[1] A. Menezes, P. van Oorschot and S. Vanstone, Handbook of Applied Cryptography, CRC Press, Inc., (1997).

[2] Sattam S. Al-Riyami and Kenneth Paterson, "Certificateless public key cryptography", Springer Berlin / Heidelberg, (2003).

[3] Sattam S. Al-Riyami., Cryptographic schemes based on elliptic curve pairings, *PhD thesis, Royal Holloway, University of London*, (2004).

[4] R.L. Rivest., "The RC4 Encryption Algorithm", RSA Data Security Inc., (1992).

[5] P. Rogaway and D. Coppersmith, "A software-optimized encryption algorithm", Springer-Verlag, Berlin, (1994), pp. 56-63.

[6] K. Suwais, A. Samsudin, "New Classification of Existing Stream Ciphers", Computational Intelligence and modern heuristics, (2010).

[7] S. Landau, Standing the Test of Time: The Data Encryption Standard, Notices of the AMS, vol. 47(3), (2000), pp. 341-349.

[8] United States Department of Commerce, National Bureau of Standards, Federal Information Processing (FIPS), Publication no. 46, Data Encryption Standard, (1977).

[9] M. Matsui, "Linear Cryptanalysis Method for DES Cipher", In Advances in Cryptology-EUROCRYPT'93, LNCS 765, Springer-Verlag, (1994), pp. 386-397.

[10] E. Biham and A. Shamir, "Differential Cryptanalysis of Des-like Cryptosystems", Journal of Cryptology, vol. 4(1), (1991), pp. 3-72.

[11] National Institute of Standards and Technology (NIST), U.S. Federal Information Processing Publication (FIPS PUB 197), The Advanced Encryption Standard, November 26, (2001).

[12] Amber Jain, Investigation of Symmetric Block Cipher Algorithms, master dissertation, (2011).

[13] Bruce Schneier , Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, (1996).

[14] W. Diffie and M.E. Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, vol. 22, (1976), pp. 644-654.

[15] R.L. Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", Communications to the ACM, vol. 21, (1978), pp. 120-126.

[16] T. ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory, vol. 31, (1985), pp. 469-472.

[17] A. Shamir., "Identity-based Cryptosystems and Signature Schemes", Proc. Crypto '84, LNCS, Vol. 196, Springer, (1985), pp. 47-53.

[18] D. Boneh and M. Franklin, "Identity Based Encryption from the Weil Pairing", In Advances in Cryptology- CRYPTO 2001, LNCS 2139, Springer, (2001).

[19] J.C. Cha and J.H. Cheon, "An Identity-Based Signature from Gap Diffie-Hellman Groups", In Proceedings of PKC'03, LNCS 2567, Springer-Verlag, (2003), pp. 18-30.

[20] Alexander w. dent, Benoît Libert & Kenneth G. Paterson, "Certificateless Encryption Schemes Strongly Secure in the Standard Model In Public Key Cryptography", *Springer. ISBN 978-3-540-78439-5*, (2008).

[21] S. Al-Riyami, Cryptographic schemes based on, elliptic curve pairings, *PhD thesis, Royal Holloway,University of London,* (2004).

[22] S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography", Springer-Verlag, (2003).

[23] J. Baek, R. Safavi-Naini and W. Susilo, "Certificateless public key encryption without pairing", *Springer-Verlag*, (2005).

[24] J. Lai and K. Kou, "Self-generated-certificate public key encryption without pairing", *Springer-Verlag*, (2007).

Received on December 30, 2013
Accepted on January 31, 2014