American Journal of Applied Sciences, 2012, 9 (11), 1849-1861
ISSN 1546-9239

doi:10.3844/ajassp.2012.1849.1861 Published Online 9 (11) 2012 (http://www.thescipub.com/ajas.toc)

PERFORMANCE EVALUATION OF MULTICRYPT ENCRYPTION MECHANISM

[1]John Prakash Arockiasamy,
[2]Lydia Elizabeth Benjamin and [1]Rhymend Uthariaraj Vaidyanathan

[1]Ramanujan Computing Centre, Anna University, Chennai, Tamil Nadu, India
[2]Department of Information Technology, Anna University, Chennai, Tamil Nadu, India

Received 2012-07-27, Revised 2012-08-28; Accepted 2012-09-04

ABSTRACT

Multicast communication allows a single message packet to be routed to multiple nodes simultaneously. Membership in a multicast group is dynamic, allowing nodes to enter and leave the multicast session. Besides the benefits, multicast communication presents the challenge of securing the communication. In order to preserve confidentiality the general encryption mechanism used for point to point communications are used. A specific encryption mechanism rather a general one is needed to suit the multicast communication requirements wherein the life time of a secret key is very short and requires a frequent change. Moreover, the next generation wireless networks have very limited resources and need a light weight security mechanism. The proposed cipher, Multicrypt, is similar to the One Time Pad and Hill cipher based on a sub band coding scheme using the principle of Orthogonal Vectors. The proposed cipher is based on the assumption of Computational Diffie Hellman problem and insolvability of Hadamard conjecture. It is designed to have multiple keys to decrypt the message like asymmetric cryptosystem so that a (key) compromise of a member would not lead to compromise of the entire system, less computational and communicational overheads, less storage complexity and there is no need for state-full members. This study also presents an extensive security analysis and the performance analysis with RSA, a public key encryption mechanism used to establish session keys. With the help of security analysis the study proves that brute force attack does not compromise the system. Multicrypt cryptosystem has the capability of dynamically adding and revoking members. The performance of Multicrypt is relatively better in terms of key setup time, encryption time, decryption time, encryption throughput and decryption throughput than RSA in the simulated setup. The proposed cipher is also proved to be secure against IND-CPA and IND-CCA attacks.

Keywords: Key Management, Multicast Encryption, Multicast Security, Orthogonal Matrices, Hadamard Matrices, Encryption Mechanism, Cryptosystem

1. INTRODUCTION

A multicast encryption scheme provides confidentiality for multicast data-ensuring that any parties other than the intended recipients should not be able to access the message. The basic security requirements are guaranteed by using cryptographic mechanism. As group members move in and out of the group, in order to preserve confidentiality, cryptographic keys are used. The cryptographic methods designed for point-to-point communication are been tailored to cater to the requirements of multicast communication. But unlike point-to-point communication, multicast communication environment is very dynamic in nature. In such an environment, the secret key used to preserve forward and backward secrecy (Canetti *et al*., 1999) of the data has to be renewed each time a member either leaves or joins the group.

Most of the existing work use one of two approaches (Rafaeli and Hutchinson, 2003; Steiner *et al*., 1996; Manz *et al*., 2010; Begum, 2011). In the first kind of approach, symmetric key encryption is used and the data is encrypted with a Traffic Encryption Key (TEK) that is known only to the multicast group members. Managing the keys is a problem in this approach. The TEK is changed when members join or leave the group to provide forward and backward secrecy. This process is known as re-keying. Among the efficient solutions, the Logical Key Hierarchy (LKH) (or Key Graph) (Wong *et al*., 2000) has individual and auxiliary keys organized into a hierarchy and each group member is assigned to a leaf and holds all the keys from its leaf to the root. The root key is shared by all group members and used as the TEK. New TEK is distributed by encrypting it with keys that deleted members do not have.

Corresponding Author: John Prakash Arockiasamy, Ramanujan Computing Centre, Anna University, Chennai, Tamil Nadu, India

So O(logn) is the best known storage (for both centre and members) and communication complexity the LKH based schemes achieved, where is the size of the multicast group. The problem with this approach is that revoking a single user involves changing the keys for all others and the receivers must be state-full to receive the latest TEK.

The second approach uses asymmetric key cryptosystem (Boneh and Franklin,1999; Boneh *et al*., 2005) and allows the receivers to be stateless. This includes the work in cryptography such as traitor tracing broadcast encryption, initiated by Fiat and Naor (1994). It's based on encryption schemes where a cipher text can be decrypted by multiple parties with different keys. The scheme requires O(tlogt tlogn) keys per user and the transmission of $O(t^2 \log^2 t \log n)$ messages where t is the number of revoked users. Boneh and Franklin (1999) proposed a scheme based on Reed-Solomon codes and the representation problem for discrete logs. There is a line of work (Tzeng and Tzeng, 2001; Kim *et al*., 2003) classified as Asymmetric Threshold Decryption-based (ATD-based) multicast encryption in which a private key is shared using a (t+1, n+t)-threshold scheme and the shares are distributed asymmetrically. Namely the centre is given shares and each user is given one share. The centre broadcasts a cipher text together with partial decryptions. Any member with a valid share of the private key can produce another decryption share and recover the message. With such schemes, user only has to store a key of constant length. Both the message complexity and sender storage is O(t), independent of the group size. The Encryption scheme described by Harkins *et al*. (2005) using finite frames and Hadamard arrays is a cipher similar to one-time pad and McEliece cipher based on sub band coding scheme. The encryption mechanism is an approximation to the one-time pad encryption scheme. The cipher is for a general communication security. The cipher uses finite frames and Hadamard arrays as key. The linearity exhibited by the cipher enables a chosen plain text attack.

This study proposes a cryptosystem, namely, Multicrypt, which is close in algebraic structure to Harkins *et al*. (2005) encryption scheme, extending our earlier work (Prakash and Uthariaraj, 2008; 2009). The Multicrypt presented here has a modified Authentication, Encryption, Decryption algorithms such it is more efficient in terms of computational complexity and security than our earlier work. Unlike our previous work, in this study each encryption does not require exponentiation, decryption makes use of the multiplicative inverse which can be computed prior, authentication procedure is simplified and security analysis are more rigorous. The proposed Multicrypt operates with multiple keys like asymmetric cryptosystems but provides mechanism for member revocation and addition without rekeying. This property of Multicrypt will help any key management protocol to reduce the overheads involved in rekeying dramatically in terms of computation and communication.

The motivation of Multicrypt Cryptosystem is that in symmetric schemes, more nodes hold the same (group) key increasing the risk of being compromised. Furthermore, the symmetric schemes expect state-full members. If a member misses a rekey message then it will be excluded from the service eventually. The asymmetric key cryptosystem overcomes these disadvantages but with increased computational and communicational complexities. Moreover, the decoupling of the group dynamics and overheads through rekeying is necessary.

A lot of work has been done to modify the cryptographic methods designed for point-to-point communication systems with sole aim of reducing the overheads involved during rekeying. But hardly, any work on designing a cryptographic method for multicast communication without re-keying has been done.

The objective behind the construction of a provably secure multicast cryptosystem is the following:

- A provably secure encryption mechanism robust against brute force, IND-CPA and IND-CCA attacks
- Multiple keys to decrypt the message like asymmetric cryptosystem so that a (key) compromise of a member would not lead to compromise of the entire system
- Less computational and communicational overheads during dynamic user revocation and addition
- Less storage complexity
- No need for state-full members

The definitions and nomenclature used in this study are presented next, followed by description of the proposed Multicrypt Cryptosystem with the security analysis, then the performance analysis of the proposed mechanism is presented and conclusion summarises the principle, contributions and performance of the proposed mechanism.

2. DEFINITIONS AND NOMENCLATURE

2.1. Definition 1 (Multicrypt Cryptosystem)

A Multicrypt Cryptosystem denoted by M = (K. R.ε. D.) consists of the following procedures.

2.2. Procedure R

A probabilistic algorithm to compute the secret initialization data for a new user subscribing to the system. The procedure, R, gets m_i as input associated with the user and returns the user's secret key Γ_i where, $\Gamma = \{\Gamma_1, \Gamma_2,\ldots,\Gamma_n\}$ and $\Gamma_i \in \Gamma$.

2.3. Key Generation K

A probabilistic polynomial-time (in k) algorithm which takes a security parameter 1^k, (initial) the number

of group members n, users to be revoked as input and generates the encryption key K. The execution of the algorithm K to obtain a K is denoted as K ← K.

2.4. Encryption ε

A probabilistic polynomial-time algorithm that, on inputs K, the encryption key and a string msg ∈ $\{0, 1\}^k$ produces an output c∈ {0,1}' U {⊥} called the cipher text. c←ε_k (msg) is denoted for the operation of executing ε on K and msg while c denote the cipher text returned.

2.5. Decryption D

A deterministic polynomial-time algorithm D takes a key $\Gamma_i \in \Gamma$ and a cipher text c ∈ {0, 1}* to return the msg∈ $\{0, 1\}^k$ U {⊥}. The operation of executing D on Γ_i and c is denoted as msg ←$D_{\Gamma i}$ (c).

Key Generation K and Authentication Procedure R should be executed together by the Core (an entity which controls the generation and distribution of cryptographic keys to the members in a multicast session) of with a set of n members for initial group setup. Addition of members is done by the R while revocation of the members which is a trivial operation (as explained later) is performed by the K algorithm itself. The Multicrypt algorithm is based on the Principle of Orthogonality defined as

2.6. Definition 2 [Principle of Orthogonality]

Two vectors X, Y∈R^n are orthogonal or perpendicular if X,Y = 0. Moreover X_1, …, X_p∈R^n are mutually orthogonal if X_i X_j = 0 whenever i ≠ j. A set of mutually orthogonal vectors is called an orthogonal set. Mutually orthogonal unit vectors {v_1,..v_p ∈ R^n} are said to be orthonormal. Alternatively, {v_1, v_2…,v_p} is called an orthonormal set.

2.7. Definition 3 [Hadamard Matrices]

A square n×n matrix H with elements ±1 that satisfies $H \times H^T = nI_n$ is called a Hadamard matrix of order n.

The nomenclature used in this chapter to describe Multicrypt Cryptosystem is described in **Table 1**.

3. MULTICRYPT-ALGORITHM DESCRIPTION

3.1. Key Generation K

The key generation process is an important process in the Multicrypt encryption scheme. The Hadamard matrices are used as one of the components of the key. These matrices exhibit good orthogonal properties. The Principle of Orthogonality enables cross correlation values to be zero which is exploited in the Multicrypt encryption scheme. Three potential schemes for key generation are presented here.

Table 1. Nomenclature used to describe multicrypt

n	Number of members in the group
m^j_i (1≤I≤n)	The i^{th} member of the j^{th} group
$\{msg\}_k$	Encryption of the message msg using secret key K
c_k^{-z}	Decryption of the cipher text using secret key K
i→m_i{Data}$_k$	l sends data to m_1 encrypted with key K
KU_m	Public key of member m
KR_m	Private key of member
K	Encryption key of the zone under consideration
Γ_i	Sub-key of m_i
A × B	Scalar product of A and B
A . B	Vector dot product of A and B
A + B	Vector Addition ‖
A‖B	Concatenation of A and B

3.2. Scheme 1

The Hadamard matrices defined in Definition 3 can be easily constructed from PN sequences. PN sequences are sequence of 1's and 0's where the numbers look like statistically independent and uniformly distributed.

3.3. Construction of Hadamard Matrices

If an (N+1)×(N+1) array is formed whose rows are each of the PN sequences, formed by same primitive polynomial, by replacing 1's with -1's and 0's with 1's of each sequence along with adding an initial row of length N and an initial column of length (N+1) with all 1's, the resultant array is a 2n×2n Hadamard matrix:

$$V = \{R_1, ..., R_N\}$$

3.4. Scheme 2

The Hadamard matrix can be generated by choosing p hadamard arrays HA_1, HA_2,…,HA_p each of size, say, $e_i \times e_i$ for 1≤i≤p, where each e_i is either 2,4, or 8. Then constructing e_1 e_2…e_p-sized matrix HA_M by the tensor product of these matrices p Eq. 1 (Steiner *et al*., 1996; Harkins *et al*., 2005):

$$V = HA_M = \bigotimes_{i=1}^{p} HA_i = HA_1 \otimes HA_2 \otimes HA_3 ..... \otimes HA_p$$
$$V = \{v_1, ...v_N\} \quad (1)$$

3.5. Scheme 3

The Gram-Schmidt Orthogonalization is a procedure for replacing linearly independent vectors X_1,…X_p, with mutually orthogonal vectors Y_1….Y_p such that Span $\{Y_1,…,Y_p\}$= Span$\{Y_1,…,Y_p\}$. The algorithm inductively generates Y_1…. Y_p in such a way that for each k = 1…. span $\{Y_1….Y_p\}$ = span$\{Y_1….Y_p\}$:

$$V = \{y_1, ..., y_p\}$$

In all of the above schemes, V is the set containing vectors which satisfy the Principles of Orthogonality. The key generation algorithm generates two keys,

namely master key and sub-keys. The master key denoted by K is computed and used by the Core and sub-keys denoted by Γ_i are computed mutually between Core and the user through the authentication procedure. The master key is the sum of the sub-keys given by:

$$K = \Gamma_1 + \Gamma_2 + \cdots .. + \Gamma_n$$

where, $\Gamma_i = v_i . g^{N_i N_c} | \Gamma_i \in \Gamma$ and $v_i \in V$. The set of sub-keys (Γ) is computed by the authentication procedure given below.

3.6. Authentication Procedure R

It is a probabilistic algorithm to compute the secret initialization of data for a new user subscribing to the system. The authentication procedure R receives as input N_i and N_c, which are random nonce associated with the user and Core respectively. The authentication procedure returns the user's secret key Γ_i where, $\Gamma = \{\Gamma_1, \Gamma_2, \ldots \Gamma_n\}$ and $\Gamma_i \in \Gamma$. Let $V = \{v_1, v_2, \ldots v_n\}$ be the set of orthogonal vectors generated from key generation process. Let q be a large prime number and g be the primitive root of q and N_i, $N_c < q$.

Throughout this study, $g^N \bmod 9$ is denoted as g^N for simplicity. Let $G = \{0,1,\ldots 9\text{-}1\}$. Then each multicast subscriber registers with the multicast service provider as given in Algorithm 1.

This algorithm is a modified key establishment protocol described by Boyd *et al.* (2006). The authentication process uses random oracles which can be instantiated by any proved agreed upon candidate one way function like MD5.

3.7. Encryption ε_K

A probabilistic polynomial-time algorithm that on input, the encryption key and a string $msg \in \{0, 1\}^k$ produces an output $c \in \{0,1\}' \cup \{\perp\}$ called the cipher text. $c \leftarrow \varepsilon_K[msg]$, is denoted for the operation of executing on K and msg while c denotes the cipher text returned. The encryption mechanism is described in Algorithm 2.

Algorithm 1 Authentication procedure R

Member	Core
1. $m_i^j : N_i \in_R \{0,1\}^k$	
2. $m_i^j \rightarrow Core_j : \{m_i^j, Core_j, g^{N_i}\}_{KU_{Core_j}}$	
	3. $Core_j : N_c \in_R \{0,1\}^k$
	4. $Core_j : SID_{Core_j} = g^{N_i} \| g^{N_C}$
	5. $Core_j : M = H_1(m_i^j \| Core_j \| SID_{Core_j} \| g^{N_i N_C})$
	6. $Core_j : S_{K_i} = H_0(m_i^j \| Core_j \| SID_{Core_j} \| g^{N_i N_C})$
	7. $Core_j : DeleteN_c$
	8. $Core_j \rightarrow m_j^i : \{g^{N_c}, \{Core_j, m_j^i SID_{Core_j}\}_M\}$
9. $m_i^j : SID_{m_i^j} = g^{N_i} \| g^{N_C}$	
10. $m_i^j : M = H_1(m_i^j \| Core_j \| SID_{m_i^j} \| g^{N_i N_C})$	
11. $m_i^j : Verify\{1, Core_j, m_i^j, SID_{m_i^j}\}_M$	
12. $m_i^j : S_{K_i} = H_0(m_i^j \| Core_j \| SID_{m_i^j} \| g^{N_i N_C})$	
13. $m_j^i : DeleteN_i$	
14. $m_j^i :\rightarrow Core_j : \{2, m_i^j Core_j SIDm_j^i\}_M$	
	15. $Core_j : Verify\{2, m_i^j, Core_j, SID_{m_i^j}\}_M$
16. $m_i^j : ACCEPTED$	
	17. $Core_j : ACCEPTED$
	18. $Core_j \rightarrow m_i^j : \{v_i, Core_j, m_i^j, T\}_{S_{K_i}}$
	19. $Core_j : \Gamma_i = v_i \times g^{N_i N_c}$

Algorithm 2 Encryption Procedure $c \leftarrow \varepsilon_K[msg]$

1. w = OAEP_ Encode (msg)
2. A = K*w
3. Choose r∈ G
4. B = v_j*r
5. c = A + B

Algorithm 3 Decryption Procedure (msg←$D_{\Gamma i}$(c))

1. K $R_{mi} = g^{NiNc}$
2. w = $(c, v_i^T)^* KR_{m_i}^{-1}$
3. msg = OAEP_Decode (w)

The following steps describe the encryption function ε_K to encrypt msg:

- The message is encoded with Optimal Asymmetric Encryption Padding (OAEP+) described by Shoup (2001) is used to obtain w. Given a plain text msg, the padding algorithm (OAEP) randomly chooses $r' \in \{0,1\}^{k_0}, u \in \{0,1\}^{n+k_1}, t \in \{0,1\}^{k_0}, w \in \{0,1\}^{k}, y \in \{0,1\}^{k_0}$ and then computes:

$$u = (G(r) \oplus m) \| H'(r' \| m)$$

$$t = H(u) \oplus r'$$

$$w = u \| t$$

- The scalar multiplication of w and vector K gives A where, K is the key
- r ∈ G is chosen randomly and v_i×r is computed. The resultant is then scalar multiplied with v_j to obtain B
- The cipher text c is obtained by the vector addition of A and B

3.8. Decryption D

A deterministic polynomial-time algorithm D takes a key $\Gamma_i = \{v_i, S_{K_i}\}$ and a cipher text c ∈ = {0,1}' to return some msg ∈ $\{0,1\}^k$ U {⊥}. The operation of executing D on Γ_i and c is denoted as msg ← $D_{\Gamma i}$ (c). To decrypt c, with decryption function $D_{\Gamma i}$ the algorithm is as described in Algorithm 3.

The following steps describe the decryption procedure:

- The decryption function finds the transpose v_i^T where $v_i \in V$
- To obtain w, compute w = $(c.v_i^T)^* KR_{m_i}^{-1}$, where $KR_{m_i}^{-1}$ is multiplicative inverse of $KR_{m_i} \in G$.
- To recover the message from w, the decryption algorithm uses the decoding or reverse pad of OAEP+ scheme.
- Compute u, t, r' as follows:

$$u = w[0 \ldots n + k_1 - 1]$$

$$t = w[n + k_1 \ldots k_1]$$

$$r' = H(u) \oplus t$$

- If $c = H'(r' \| m)$, then the algorithm outputs the clear text msg. Otherwise, the algorithm rejects the cipher text and does not output a cipher text

The following theorem proves that the decryption algorithm described above provide correct decryption.

3.9. Theorem 1

The scheme is said to provide correct decryption if for any key $\Gamma_i \in \Gamma$ and any message msg ∈ $\{0, 1\}^k$

$$Pr[c \leftarrow E_K(w) : c = \perp \text{ or } D_{\Gamma_i}(c) = msg] = 1$$

3.10. Proof

Consider the user and Core common secret derived during authentication procedure given by $R_{m_i} = g^{x_i s}$, the multiplicative inverse of KR_{m_i} denoted by $KR_{m_i}^{-1}$, the sub-key of a user i given by $\Gamma_i = v_i.g^{x_i s}$ and the master key given by K = Γ_1 + …. . Γ_n The decryption provided by the decryption algorithm is correct because:

$$D_{\Gamma_i}(E_K(w)) = D_{\Gamma_i}(c) = (c.v_i^T)^* KR_{m_i}^{-1} = (K.w + v_j.r).v_i^T)^* KR_{m_i}^{-1}$$

Expanding the above equation:

$$D_{\Gamma_i}(E_K(w)) = \left(\left(\left(v_1.g^{x_1 s} + \ldots_{\ldots} + v_{n-1}.g^{(x_{n-1} s)}\right) w + v_j.r\right).v_i^T\right)^* KR_{m_i}^{-1}$$

By the definition of Principle of Orthogonality the above equation can be simplified as:

$$D_{\Gamma_i}(E_K(w)) = g^{x_i s} * KR_{m_i}^{-1} * w = w$$

The decryption algorithm uses the reverse padding procedure then to recover the message msg from the given cipher text. Thus, for any msg∈G and valid user secret key Γ_i, the decryption algorithm will output msg with probability equal to 1.

3.11. Dynamic Key Addition and Revocation

Any member can be dynamically revoked and added with trivial computations. Members have to be revoked during multicast communication session in the event of a voluntary member leave or compelled member leave. In that case, the leaving member's key should be revoked without affecting the state of the other active user secret keys. Let l be the leaving member whose membership has to be revoked. The member revocation is done as follows Eq. 2:

$$K = \Gamma_1 + \ldots \Gamma_l + \ldots + \Gamma_n - \Gamma_l \quad (2)$$

Similarly, dynamically adding a member (Γ_{n+1}) during a transaction can be done without affecting the functioning of the other active members as Eq. 3:

$$K = \Gamma_1 + \ldots + \Gamma_n + \Gamma_{n+1} \quad (3)$$

3.12. Property 1

A traitor t that redistributes his user secret key Γ_t to unauthorized members can be traced.

3.13. Proof

Assume that the user t is a traitor, re-distributing his secret key for unauthorized access. Then the pirated decoder's would be $\Gamma_t = v_t.g^{ts}$ the t's user secret key. Given the pirated decoder, the identity of the traitor can be traced as follows:

- The Core knows the public key of every user g^{x_i} which was obtained and verified during the process of authentication procedure execution.
- $\forall v_i \in V, \text{compute } \Gamma' = \frac{\Gamma_t.v_i}{\left(g^{x_i}\right)^s}$
- $\text{if } (\Gamma' = 1) \text{ then } t \leftarrow i$
- i^{th} user is the traitor

Hence the traitor can be traced.

3.14. Property 2

Any member can be dynamically revoked with trivial computations.

3.15. Proof

Members have to be revoked in the event of a voluntary member leave or compelled member leave (traitor). In that case, the leaving member's key should be revoked without affecting the state of the other active user secret keys. Let l be the leaving member whose membership has to be revoked. The member revocation is done as follows:

- K is giver by $K = \Gamma_1 + \cdots \ldots + \Gamma_i + \ldots + \Gamma_n$
- To revoke a member l, delete I's secret key Γ_i from K. The process is given by:

$$K = \Gamma_1 + \cdots .. + \Gamma_i + \ldots + \Gamma_n - \Gamma_i$$

Any revoked member cannot decrypt the message subsequently maintaining forward secrecy. Assume Γ_i is the user secret key that was revoked and $c = K.\ m + v_j.\ g^{sy}$ the cipher text obtained after revocation. Then process of decryption using the revoked key Γ_l is given by:

$$D_{\Gamma_i}\left(E_K(w)\right) = D_{\Gamma_i}(c) = \left(c.v_i^T\right) * KR_{m_i}^{-1} = (K.w + v_j.r).v_l^T) * KR_{m_i}^{-1}$$

Then by the principle of Orthogonality the above equation can be simplified as:

$$D_{\Gamma i}(c) = 0$$

Hence any revoked user cannot decrypt the messages correctly after revocation.

4. SECURITY AND PERFORMANCE ANALYSIS

The security analysis proves that Multicrypt is secure against IND-CPA and IND-CCA with the help of standard formal security models.

4.1. Theorem 2

Multicrypt encryption function is a one-way function and the following hold:

- The function is easy to compute. Namely, there exists a PPT algorithm A which on input msg returns $c \leftarrow \varepsilon_K(msg)$ in time polynomial in |msg|
- The function is hard to invert. Namely, for all PPT algorithms there exists a negligible function $\in$ (.) such that Eq. 4:

$$\Pr\left[\begin{matrix} msg \leftarrow \{0,1\}^k; c = \in_K (msg); msg' \\ = A\left(1^k, c\right); E(msg') = c \end{matrix}\right] \leq \in (k) \tag{4}$$

4.2. Proof

The Multicrypt encryption scheme given by $c = K*w + v_j*r$ consists of scalar multiplication, vector addition and one exponentiation which could be done in polynomial time with an algorithm which on input msg returns c = f (msg). To prove the sec case, consider a polynomially bounded adversary A having access to $\left(q, g, g^{N_A}, g^{N_B}, c, |msg|\right)$. Then the task of the adversary is to find $msg' = msg \text{ or } \varepsilon_K[msg'] = c$. In order to compute msg the adversary should compute any user secret key $\Gamma_i = \left(v_i \times KR_{mi}\right)$. The task of the adversary then is to:

- Find the generated orthogonal matrix (V) or find a vector $v_i \in V$. Let $\Pr\left[(V' = V)|c\right]$ be the probability that finds the orthogonal matrix or a vector given the cipher text (c)
- Find $h = g^{N_A N_B} \in \Gamma \text{ given} \left(q, g, g^{N_A}, g^{N_B}, c\right) \text{for } i = 1, \ldots, n.$

Let $\Pr[h' = g^{N_A N_B} | (q, g, g^{N_A}, g^{N_B}, c)]$ be the probability that A finds $h \in \Gamma$. Therefore, equation (4) can be written as Eq. 5:

$$\begin{aligned} &\Pr\left[msg \leftarrow \{0,1\}^k; c = \varepsilon_K(msg); msg' = A\left(1^k, c\right); \varepsilon(msg') = c\right] \\ &= \left(\Pr\left[(V' = V)|c\right].\Pr\left[h' = g^{N_A N_S} | \left(p.g.g^{N_A}.g^{N_S}.c\right)\right]\right) \end{aligned} \tag{5}$$

Multicrypt is similar to a sub band coding scheme. As in sub-band coding scheme, encoding a message twice results in two different cipher texts, encrypting a message twice results in two different cipher texts. The numerical experiment carried out by Harkins *et al*. (2005) shows that

a brute force attacks on is infeasible. Moreover, the experiments and mathematical proofs by Harkins *et al*. (2005) show that the garbage or random value r added during the encryption process can control the accuracy an adversary would need to make a guess of V.

Further, two Hadamard matrices are considered equivalent if one can be obtained from the other by negating rows or columns, or by interchanging rows or columns. Up to equivalence, there is a unique Hadamard matrix of orders 1, 2, 4, 8 and 12. There are 5 in equivalent matrices of order 16, 3 of order 20, 60 of order 24 and 487 of order 28. Millions of in equivalent matrices are known for orders 32, 36 and 40. Using a coarser notion of equivalence that also allows transposition, there are 4 in equivalent matrices of order 16, 3 of order 20, 36 of order 24 and 294 of order 28. Therefore, if the orthogonal matrix (V) is carefully chosen as proposed by Koukouvinos and Simos (2011) then the attacker needs to try all possible key values to find the matrix. Due to the randomness introduced in the encryption process, an exhaustive key search does not give all possible plain text messages. The same plaintext message when encrypted twice will result in two different cipher texts. Therefore, it becomes hard to perform an exhaustive key search. Assuming the key is carefully chosen and the key space is sufficiently large then it can be written that Eq. 6:

$$\Pr\left[(V' = V) \mid c\right] \leq \in (k) \tag{6}$$

The problem of finding $g^{N_A N_B}$ given $(q, g, g^{N_A}, g^{N_B}, c)$ is equivalent to the problem of Computational Diffie-Hellman (CDH). Assuming, CDH problem is intractable and one-way. The adversary has negligible probability of finding h' =h. Therefore, the probability can be written as Eq. 7:

$$\Pr\left[\left(h' = g^{N_A N_S}\right) \mid \left(q.g.g^{N_A}.g^{N_S}.c\right)\right] \leq \in (k) \tag{7}$$

Hence, we get Eq. 8:

$$\Pr\left[\begin{matrix} msg \leftarrow \{0,1\}^k; c = \varepsilon_k(msg); msg' \\ = A\left(1^k.c\right); \varepsilon(msg') = c \end{matrix}\right] \leq \varepsilon(k) \tag{8}$$

4.3. Corollary 1

If ε_k is a one way function, then with OAEP+ the encryption scheme is IND-CCA and IND-CCA2 secure (Shoup, 2001)

Then from Theorem 2 and Corollary 1, Multicrypt is IND-CCA and IND-CCA2 secure.

4.4. Alternate Security Analysis

Bellare and Rogaway (1994) gave the first formal model of security for the analysis of authentication and key agreement protocols. It is a game-based definition, in which the adversary is allowed to interact with a set of oracles that model communicating parties in a network and where the adversary's goal is to distinguish whether the challenge it is given is a correctly shared key or is a randomly generated value. This study also provided the first computational proof of security for a cryptographic protocol. By following this approach namely, the Bellare and Rogaway or BR models, another formal security proof is presented here to prove that the proposed cryptosystem is IND-CPA and IND-CCA secure.

4.5. Theorem 3

Let $\mu \in \{\text{IND-CPA, IND-CCA}\}$ and a multicast encryption scheme is resilient against any attack of type u, only if $Adv^{u}_{MA}(k)$, of any polynomial time adversary A is a negligible function of $\in (k)$:

$$Adv^{u}_{MA}(k) = \left|\Pr(b' = b) = \frac{1}{2}\right|$$

4.6. Proof

Indistinguishability under chosen plain text attack or left-right indistinguishability under chosen plain text attack is to consider an adversary not in possession of the secret key, chooses two messages of same length. Then one of the messages is encrypted and the cipher text is given to the adversary. The working of the left-right encryption oracle is as given in Algorithm 4. The scheme is considered secure if the adversary has negligible advantage in guessing which one of the two messages was encrypted.

The problem for the adversary is to find to which oracle it is interacting. The adversary can make polynomial queries to the oracle as the adversary is polynomial bounded. An adversary, is constructed which is given a left-right encryption oracle $\varepsilon_k(LR(m_o,m_1,b))$ that takes as input two messages and return the encryption of either the left or the right message in the pair, depending on the value of the bit b. The bit $b \in \{0,1\}$ is chosen random. The adversary construction is shown in Algorithm 5.

Algorithm 4 Left or Right Encryption Oracle

Oracle $\varepsilon_k\left(LR\left(msg_0, msg_1, b\right)\right)$

1. $b \in_U \{0,1\}$ and $\left\{msg_0, msg_1\right\} \in \{0,1\}$
2. if $msg_0 = msg_1$ then return $\perp$
3. $C \leftarrow_k \left(msg_b\right)$
4. return

Algorithm 5 Adversary Construction for IND-CPA Attacks

Adversary $A^{\varepsilon_K}(LR(....b))$

1. $m_1 \leftarrow 0^{2n}; m_0 \leftarrow 0^n \| 1^n$
2. $c_1c_2 \leftarrow \varepsilon_k(LR(m_0, m_1, b))$
3. if $c_1 = c_2$?1:0

Algorithm 6 Experiment $Exp_M^{(IND-CCA)^b}(A)$

Experiment $Exp_M^{(IND-CCA)^b}(A)$

1. $K \leftarrow K$
2. $b \leftarrow A^{\varepsilon_K(LR(.,.,b)),D_{\Gamma_i}(.)}$
3. if A querid $D_\Gamma(.)$ on a ciphertext previously retuned by ε_K (LR(....b))? 0:b

when, b = 1, the oracle returns $c_1c_2 = \varepsilon_K(0^n)\|\varepsilon_K(0^n)$. that $c_1 = c_2$ due to the randomness introduced in the algorithm. Moreover, ε_K is a random permutation and the adversary algorithm would return 0 similarly, when b = 0, the oracle returns $c_1c_2 = \varepsilon_K(0^n)\|\varepsilon_K(1^n)$. From the description of the same multicrypt encryption algorithm it can be observed that $c_1 = c_2$ due to the randomness in the algorithm. The adversary algorithm would return 0. Therefore:

$$Pr\left[Exp_M^{(IND-CPA)^1}(A)=1\right] = Pr\left[Exp_M^{(IND-CPA)^0}(A)=1\right]$$

Therefore, the adversary making a guess b^r of the value b becomes hard as it can be observed that adversary algorithm will return 0 for both left and right oracle functioning. In other words, $Pr(b^r = b) = \frac{1}{2^s}$ Therefore:

$$Adv_{MA}^{IND-CPA}(k) = |Pr(b^r = b) - \frac{1}{2}| = \in(k)$$

Alternatively:

$$Adv_{MA}^{IND-CPA}(k) = |Pr\left[Exp_M^{(IND-CPA)^1}(A)=1\right] - Pr\left[Exp_M^{(IND-CPA)^0}(A)=1\right]| = \in(k)$$

Hence, A's IND-CPA advantage is zero. Indistinguishability under chosen cipher text attack is a stronger type of attack. In this type of attack, an adversary has access to decryption oracle as well. A decryption oracle can be assumed as any user with valid key offering decryption service. As in the case of chosen plain text attacks, the adversary is given the left or right encryption oracle described Algorithm 4. The experiment is as given in Algorithm 6.

Algorithm 7 Adversary $A^{\varepsilon_K(LR(.,b))D_{\Gamma_i}(.)}$

Adversary $A^{\varepsilon_K(LR(.,b)),D_{\Gamma_i}(.)}$

1. $msg_0 \leftarrow 0^l; msg_1 \leftarrow 1^l$
2. $c \leftarrow E_K(LR(msg_0, msg_1, b))$
3. $c' \leftarrow cÅ1^l$
4. $msg \leftarrow D_{\Gamma_i}(c')$
5. if msg = msg_0?1:0

The adversary goal is to guess the value of b correctly. The adversary construction for indistinguishability under chosen cipher text attack is as given in Algorithm 7.

The adversary queries with the message (msg_0, msg_1) each one block long and it's returned a cipher text c. It flips the bits of c to get c' and then submits the cipher text c' to the decryption oracle. When b = 1, let c be the cipher text that was returned from the encryption oracle. Then, ε_k (msg_1) $= K \times w_1 + v_j \times r \oplus 1^l$. Now the decryption part, msg = $Dr_i(c') \neq msg_0$ or msg_1. A close observation of the encryption and decryption process would confirm the claim that in any case msg $\neq msg_1$. It can thus be written that:

$$Pr\left[Exp_M^{(IND-CPA)^1}(A)=1\right] = Pr\left[Exp_M^{(IND-CPA)^0}(A)=1\right]$$

Therefore:

$$Adv_M^{IND-CPA}(k) = Pr\left[Exp_M^{(IND-CPA)^1}(A)=1\right] - Pr\left[Exp_M^{(IND-CPA)^0}(A)=1\right]$$
$$Adv_M^{IND-CPA}(k) = \in(k)$$

Hence the Multicrypt encryption scheme is resilient against any attack of type μ and $Adv_M^{IND-CPA}(k))$ of any polynomial time adversary A is a negligible function $\in$ (k).

4.7. Performance Analysis

In this subsection, the performance of Multicrypt is analyzed. **Table 2** gives the time complexity of Multicrypt. Simulations were carried out in which the average time taken by Multicrypt for key setup, encryption and decryption were calculated. The results obtained by varying key size and data size are plotted in **Fig. 1-6** respectively. Also, the simulation compares Multicrypt with RSA, a standard public key encryption mechanism.

The operating characteristics are:

4.8. Key Length

The length of the two keys, the master key and the sub-keys are the same. Therefore, the length of the key is equal to the dimension of the vector v_i. Therefore, the length of the key is denoted as l_k bits.

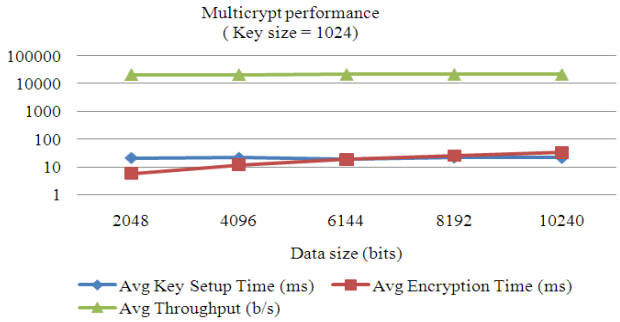

Fig. 1. Multicrypt Performance for varied Data sizes

Table 2. Time complexity analysis

Plain text length	$\mathcal{O}(n_p)$ bits
Cipher text length	$\mathcal{O}(l_k)$ bits
Encryption complexity	$\mathcal{O}(2l_k)$ operation
Decryption complexity	$\mathcal{O}(l_k+1)$ operation
Key length	$\mathcal{O}(l_k)$ bits
Revocation complexity	$\mathcal{O}(1)$
Keys per user	$\mathcal{O}(1)$
Keys per Core	0(|user∈ Core|)

4.9. Plain Text Length

The plain text or the message m can take values from the group setup (G.×.g.q). Therefore, the range of message is g<msg<q. If the size of q is n_q bits then the size of msg would also be n_q bits long and it's of $O(n_p)$.

4.10. Cipher Text Length

Cipher text c is given by c = K*msg + v_j*r. As msg and r are scalars, the length of the cipher text is equal to the dimension of K. If the vector K is of dimension l_k, the cipher text is of $O(l_k)$.

4.11. Encryption Complexity

The encryption complexity is determined by counting the number of operations performed during the process. The encryption process consists of two scalar multiplication and one vector addition. Therefore, one scalar multiplication involves multiplying, vector dimension times the scalar. The number of multiplication is equal to the dimension of the vector, say l_k, then scalar multiplications constitutes to $2l_k$ multiplications. The number of additions performed in adding two vectors is equal to the dimension of the vector. Therefore, the complexity is given by $O(2l_k)$ assuming addition as trivial operation.

4.12. Decryption Complexity

The decryption complexity is determined by counting the number of operations performed during the process. In the decryption process, one vector dot product and scalar multiplication is performed. The complexity then from the above discussions is shown to be of $O(l_k+1)$.

4.13. Simulation

The performance of the Multicrypt cryptosystem was simulated using Java programming language in Intel Core Pentium i5 machine. The average key setup time, encryption time and decryption time were analysed.

4.14. Average Key Setup Time

The average key setup time is the time taken in milliseconds to set up the key for a given key size. The Multicrypt cryptosystem needs to set up the keys as described in the key generation procedure. In the simulation carried out, it's the time taken to generate the sub-keys ($\Gamma_i = v_i \times g^{sy} | \forall i \in \Gamma$) and master key ($K = \Gamma_1 + \ldots + \Gamma_n$).

4.15. Average Encryption \ Decryption Time

The average encryption\decryption time is the time taken in milliseconds to encrypt \decrypt a given message with the master key.

4.16. Average Throughput

Average throughput is the number of plaintexts encrypted and decrypted in bits per sec. In other words, it is the ratio of size of message in bits to the sum of average key setup time, encryption time and decryption time in sec.

4.17. Multicrypt Performance-Varied Data Sizes

The performance of Multicrypt for the parameters indicated above for different data sizes for a fixed key size of 1024 were analyzed and are plotted in **Fig. 1**. The analysis was carried out for varied data sizes ranging from 2048 bits (2 MB) to 10240 bits (10 MB) in steps of 2048 bits (2 MB). The key size is fixed at 1024 bits (the secure standard size for Diffie Hellman Key Exchange setup).

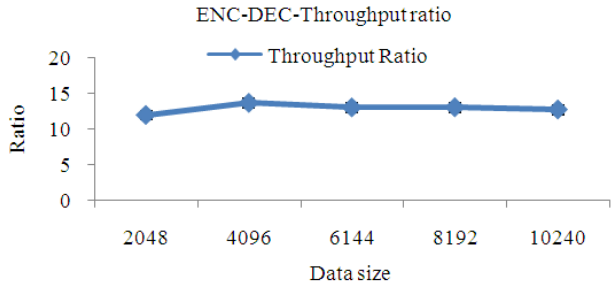

Fig. 2. Average Encryption – Decryption Throughput Ratio

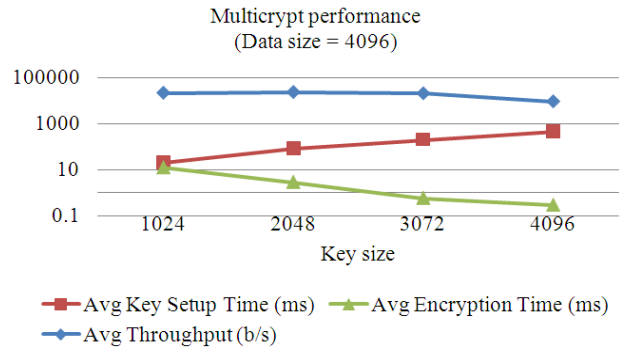

Fig. 3. Multicrypt Performance for varied Key sizes

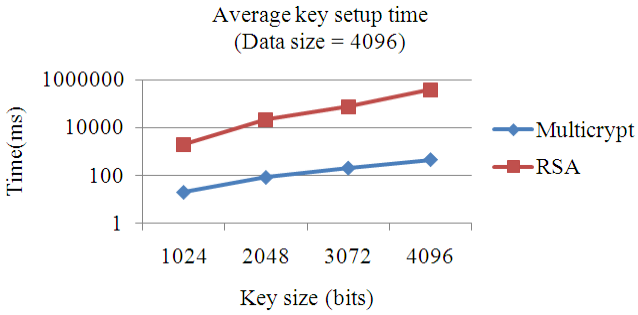

Fig. 4. Average Key Setup Time for varied Key Sizes

From the **Fig. 1** it can be observed that average encryption and decryption time increases with increase in data size. This is due to the fact that message range is between g<m<q.q is the modulus in the group setup and g is the primitive element or the generator. Larger the number of bits for q, larger would be the magnitude of the number that can be represented and larger would be the cardinality of the group.

Therefore, a larger the number of bits for q, a bigger message can be encrypted in one encryption whereas for q with less number of bits (a small value) will require multiple encryptions and decryptions. Moreover, the throughput ratio defined as the ratio between average encryption throughput and average decryption throughput remains constant as shown in **Fig. 2**. Even from **Fig. 1**, it can be observed that the average encryption throughput and average decryption throughput remains constant across varied data sizes.

4.18. Multicrypt Performance-Varied Key Sizes

The analysis were carried out for varied key sizes in bits like 1024, 2048, 3072 and 4096 though the secure recommended standard key size for Diffie Hellman Key Exchange's (DHKE), Computational Diffie Hellman (CDH) assumption to hold is 1024.

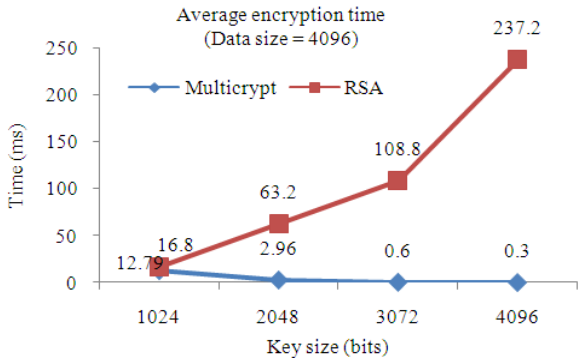

Fig. 5. Average Encryption Time for varied Key sizes

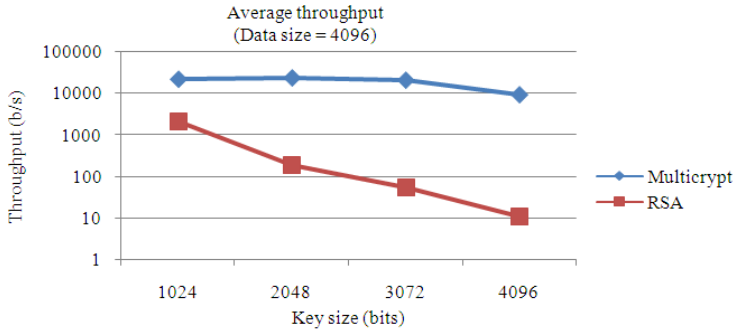

Fig. 6. Average Throughput for varied Key Sizes

From **Fig. 3** it can be seen that the key setup time increases with increase in key size which is the inherent property of DHKE set ups. The average encryption time is relatively less. An important observation would be the decrease in average encryption as the size of key increases. This is due to the fact that not every encryption needs exponentiation. A closer analysis of the encryption algorithm will reveal that they are one time setup and subsequently they are used with trivial and few non-trivial operations. Another reason for this decline of average encryption time with increase of key size is the DHKE group set up. In this kind of setup, as explained before, message can take values only between g<m<q. Therefore, the choice of number of bits for q affects the number of encryptions or the size of message that can be accommodated in one encryption. But larger the size of q, larger would be the key setup time. Since, q being a large prime, the time taken to test a large n-bit number to be a prime or not increases with the number of bits n. That is the reason why the key setup time increases with the number of bits.

4.19. Benchmark Tests

To benchmark the performance of Multicrypt cryptosystem with a standard cryptosystem, RSA was chosen for comparison. The architecture of Multicrypt encryption mechanism is so comprising that it could not be possibly clearly fitted into any of the classification of cryptosystems like symmetric, asymmetric, broadcast and threshold. Elgammal encryption mechanism which could be arguably close to the working of Multicrypt will definitely perform below Multicrypt, since it requires a new key to be generated for every encryption. RSA being an asymmetric cryptosystem was chosen to only benchmark the performance of Multicrypt key setup time, average encryption and decryption throughput. The result of the simulation were analyzed and plotted as given in **Fig. 4-6** respectively.

From **Fig. 4**, it can be observed that the key set up time increases with increase in key size for both RSA and Multicrypt. But relatively, it can be observed that Multicrypt average key set up time is significantly less than RSA. RSA's average key set up time is on the higher side due to its heavy dependence on exponentiation over large numbers. As the key size increases these operations gets costlier in RSA.

Figure 5 and 6 shows the average encryption time and throughput comparison between Multicrypt and RSA respectively. It can be seen from **Fig. 5** that average encryption time decreases with key size for Multicrypt while the same increases for RSA. This is due to fact that an

encryption in RSA involves exponentiation and as the key size increases, time taken for exponentiation operations increases. This increases the average encryption time of RSA and thereby decreasing the average throughput as well which is evident in **Fig. 6**. Average throughput of Multicrypt is significantly larger due to the increase in the size of message that can be accommodated in one encryption as described in previous subsections.

5. CONCLUSION

Multicast Security is very difficult to achieve in real-life and more so in the presence of adversaries in the system. In this study, the design of Multicrypt, a multicast cryptosystem which would secure multicast communication is presented. The key management protocol using Multicrypt will be a future work. The Multicrypt reduces the overhead involved in key establishment process when incorporated into a key management protocol. Support for dynamic key revocation reduces the overheads of rekeying. This enables decoupling of network dynamics from the rekeying. **Table 2** shows that Multicrypt requires less storage space per user and controller. Multicrypt cryptosystem's security was analyzed and was shown to be resilient against IND-CPA and IND-CCA attacks. Multicrypt performance through benchmark tests showed that Multicrypt cryptosystem's average key setup time, average encryption throughput and average decryption throughput are efficient. Therefore, from the results and analysis, the Multicrypt cryptosystem when applied to dynamic environments like ad hoc network, cloud computing environment would provide significant improvement in the reduction of communicational and computational overheads.

6. REFERENCES

Bellare, M. and P. Rogaway, 1994. Entity authentication and key distribution. Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology, (AC' 94) Springer-Verlag Inc., New York, USA., pp: 232-249.

Begum, S.J., 2011. A new scalable and reliable cost effective key agreement protocol for secure group communication. J. Comput. Sci., 7: 328-340. DOI: 10.3844/jcssp.2011.328.340

Boneh, D. and M. Franklin, 1999. An efficient public key traitor tracing scheme. Adv. Cryptol., 1666: 783-783. DOI: 10.1007/3-540-48405-1_22

Boneh, D., C. Gentry and B. Waters, 2005. Collusion resistant broadcast encryption with short ciphertexts and private keys. Proceedings of the 25th Annual International Conference on Advances in Cryptology, (AC' 05), Springer-Verlag Berlin, Heidelberg, pp: 258-275. DOI: 10.1007/11535218_16

Boyd, C., K.K.R. Choo and A. Mathuria, 2006. An extension to bellare and rogaway (1993) model: Resetting compromised long-term keys. Proceedings of the 11th Australasian conference on Information Security and Privacy, (ISP' 06), Springer-Verlag Berlin, pp: 371-382. DOI: 10.1007/11780656_31

Canetti, R., J. Garay, G. Itkis, D. Micciancio and M. Naor *et al*., 1999. Multicast security: A taxonomy and some efficient constructions. Proceedings of IEEE 18th Annual Joint Conference of the IEEE Computer and Communications Societies, Mar. 21-25, IEEE Xplore Press, New York, pp: 708-716. DOI: 10.1109/INFCOM.1999.751457

Fiat, A. and M. Naor, 1994. Broadcast encryption. The The Pennsylvania State University.

Harkins, R., E. Weber and A. Westmeyer, 2005. Encryption schemes using finite frames and hadamard arrays. Exp. Math., 14: 423-433. DOI: 10.1080/10586458.2005.10128935

Prakash, A.J. and V.R. Uthariaraj, 2008. Multicast cryptosystem: A cryptosystem for secure multicast communication. Proceedings of the IFIP International Conference on Network and Parallel Computing IEEE Computer Soceity, Oct. 18-21, IEEE Xplore Press, Shanghai, pp: 119-124. DOI: 10.1109/NPC.2008.73

Prakash, A.J. and V.R. Uthariaraj, 2009. Multicrypt: A provably secure encryption scheme for multicast communication. Proceedings of the 1st International Conference on Network Communications, Dec. 27-29, IEEE Xplore Press, Chennai, pp: 246-253. DOI: 10.1109/NetCoM.2009.80

Kim, C.H., Y.H. Hwang and P.J. Lee, 2003. An Efficient Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. Department of Electronic and Electrical Engineering.

Koukouvinos, C. and D.E. Simos, 2011. Encryption schemes using plotkin arrays. Int. J. Applied Math. Inform. Sci., 5: 547-557.

Manz, D., P. Oman and J.A. Foss, 2010. A framework for group key management protocol assessment independent of view synchrony. J. Comput. Sci., 6: 229-234. DOI: 10.3844/jcssp.2010.229.234

Rafaeli, S. and D. Hutchinson, 2003. A survey of key management for secure group communication. ACM Comp. Surveys, 35: 309-329. DOI: 10.1145/937503.937506

Shoup, V., 2001. OAEP reconsidered. Adv. Cryptol. Lecture Notes Comput. Sci., 2139: 239-259. DOI: 10.1007/3-540-44647-8_15

Steiner, M., G. Tsudik and M. Waidner, 1996. Diffie-Hellman key distribution extended to group communication. Proceedings of the 3rd ACM Conference on Computer and Communications Security, (CCS' 96) ACM Press, New York, USA., pp: 31-37. DOI: 10.1145/238168.238182

Tzeng, W.G. and Z.J. Tzeng, 2001. A public-key traitor tracing scheme with revocation using dynamic shares. Lecture Notes Comput. Sci., 1992: 207-224. DOI: 10.1007/3-540-44586-2_16

Wong, C.K., M. Gouda and S.S. Lam, 2000. Secure group communications using key graphs. IEEE/ACM Trans. Netw., 8: 16-30. DOI: 10.1109/90.836475