Implementation of Homomorphic Encryption Schemes for Secure Packet Forwarding in Mobile Ad Hoc Networks (MANETs)

Levent Ertaul and Vaidehi,

Department of Mathematics & Computer Science
California State University, EastBay
Hayward, CA, USA

Summary
In this paper we provide a new scheme to securely forward the message in wireless mobile ad hoc networks (MANETs) by using existing homomorphic encryption schemes. This scheme is an alternative for threshold cryptography (TC) in MANETs to securely forward the message. By using homomorphic encryption schemes we remove the computational cost associated with Lagrange Interpolation scheme used in TC and also increase the success rate of the encrypted message at the destination in MANETs. In this paper we determine the computational cost of the homomorphic encryption schemes such as Domingo-Ferrer's new Privacy Homomorphism, Domingo-Ferrer's additive and multiplicative Privacy Homomorphism, Domingo-Ferrer's Privacy Homomorphism allowing field operations on encrypted data and Mixed multiplicative homomorphism (MMH) and suggest a better encryption schemes to be used in MANETs. We also provide the implementation details of the proposed new scheme in MANETs. In addition we provide an alternative new scheme for Domingo-Ferrer's new Privacy Homomorphism and Domingo-Ferrer's Additive and Multiplicative Privacy Homomorphism

Key words:
Security in MANETs, Homomorphic Encryption Schemes, Threshold Cryptography

1. Introduction

In MANETs, mobile nodes communicate directly with each other in a pear to pear manner. Mobile nodes join in, on the fly, and create a network on their own, each node carrying out basic operations like routing and packet forwarding without the help of an established infrastructure. All the available nodes can join the network and carry out network operation. Due to this huge dependency's on the nodes, there are more security problems. In MANETs the nodes are capable of roaming independently. The node with inadequate physical protection can be easily captured, compromised and hijacked. Therefore the nodes in the network must be prepared to work in a mode that trusts no peer [1, 2, 3, 4]

Homomorphic encryption schemes allow operations to be performed on the encrypted data (ciphertext) as if the operation is performed on the plaintext. Homomorphic encryption schemes can have the property of additive, multiplicative and mixed multiplicative homomorphism. In *additive homomorphism*, decrypting the sum of two ciphertext is same as addition of two plaintext represented as $E(x+y) = E(x) + E(y)$. In *multiplicative homomorphism*, decrypting the product of two ciphertext is same as multiplication of the two plaintext. Multiplicative homomorphism is mathematically represented as $E(x*y) = E(x) * E(y)$. In *mixed multiplicative homomorphism*, decrypting the product of one ciphertext plaintext is same as multiplication of two plaintext, represented as $E(x*y) = E(x) * y$. [5, 6, 7, 8]

In this paper we look at the existing homomorphic encryption schemes and also the current security solutions for MANETs. We then propose a new scheme based on homomorphic encryption schemes for secure message forwarding in MANETs as an alternative for TC in MANETs. We implement few of the homomorphic encryption schemes and suggest the best suited homomorphic encryption scheme for MANETs. We implement the proposed new scheme in MANETs and give the implementation issues. We also propose an alternative new scheme which will be used with Domingo-Ferrer's new Privacy Homomorphism and Domingo-Ferrer's additive and Multiplicative Privacy Homomorphism.

The paper is organized as follows. In section 2, we briefly describe the overview of homomorphic encryption schemes. In section 3, we briefly describe the current security solution in MANETs. In section 4, we propose a new scheme which uses homomorphic encryption schemes to securely forward the message in MANETs. In section 5, we give the implementation details of the proposed new scheme in MANETs along with performances details of implemented hommorphic encryption schemes. In section 6, we propose an alternative new scheme in MANETs when specific homomorphic encryption schemes are used. Finally, in section 7, we conclude.

Manuscript received November 5, 2007
Manuscript revised November 20, 2007

2. Encryption Schemes exhibiting the property of homomorphism

In this section we give an overview of four different encryption schemes which exhibit the property of homomorphism.

The cryptosystem ***Mixed Multiplicative Homomorphism*** is introduced in [5]. This cryptosystem uses large number m, where $m = p * q$. Here p and q are large prime numbers, which are kept secret. The set of original plaintext messages is in $Z_p = \{ x | x <= p \}$, $Z_m = \{ x | x < m \}$ has the set of ciphertext messages and $Q_p = \{ a | a \notin Z_p \}$ has the set of encryption clues.

The *encryption algorithm* is performed by choosing a plaintext '$x' \in Z_p$ and a random number 'a' in Q_p such that $x = a \bmod p$. Here p is kept secret. The ciphertext y is calculated as $y = E_p(x) = a \bmod m$.

In *decryption algorithm* the plaintext x is recovered as $x = D_p(y) = y \bmod p$, where p is the secret key.

This cryptosystem has the property of additive, multiplicative and mixed multiplicative homomorphism. The proposed protocol, though exhibits the property of homomorphism is not very secure against known plaintext attacks, but secure against known ciphertext attacks [5].

Domingo-Ferrer's New Privacy homomorphism is introduced in [6] which is a homomorphic encryption scheme not vulnerable to known ciphertext attacks.

Let us look into the protocol in detail. In this protocol n and m are the public parameters. Here $m = p * q$, where p and q are large prime numbers. To increase security, m can be kept secret. The number 'd' represents the split of the plaintext. The secret keys are p, q, x_p, x_q. Here, $x_p \in Z_p$ *and* $x_q \in Z_q$.

Encryption operation is performed by selecting the plaintext $a \in Z_m$. We then split a into secret numbers $a_1, a_2 \ldots a_n$, such that $a = (a_1 + a_2 \ldots + a_i + \ldots a_d) \bmod m$ and $a_i \in Z_m$.

$E_k(a) = (a_1 x_p \bmod p, a_1 x_q \bmod q), (a_2 x^2_p \bmod p, a_2 x^2_q \bmod q) \ldots (a_n x^n_p \bmod p, a_n x^d_q \bmod q)$

Decryption operation is performed by computing scalar product of the i^{th} pair $[\bmod p, \bmod q]$ by $[x^{-i}_p \bmod p, x^{-i}_q \bmod q]$ to get $[a_i \bmod p, a_i \bmod q]$. The pairs are then added up to get $[a \bmod p, a \bmod q]$. Finally, Chinese remainder theorem (CRT) [9] is performed to get $a \bmod m$.

The privacy homomorphism has the property of additive and multiplicative homomorphism. This homomorphism scheme though secure against know ciphertext attacks is not very secure against known plaintext attacks [10].

Domingo-Ferrer's Privacy Homomorphism allowing field operation on encrypted data is introduced in [7]. In this encryption scheme p and p' are large secret primes and $q = pp'$ is public. Qp is defined as $Qp = \{a/b : a,b \in Zp\}$

Encryption operation is performed by selecting a value $x \in Zp$ and a random fraction a/b in Qp, such that $x = ab^{-1} \bmod p$. The ciphertext is computed as $y = Ep(x) = ab^{-1} \bmod q$.

Decryption operation is performed by picking any fraction $A/B \in Qp$ such that $y = AB^{-1} \bmod q$. The key p is used to recover the plaintext x as $x = Dp(y) = AB^{-1} \bmod p$.

This privacy homomorphism has the property of additive, multiplicative and mixed multiplicative homomorphism. The privacy homomorphism is secure against chosen ciphertext attacks but not very secure against known-plaintext attacks [7].

Domingo-Ferrer's Additive and Multiplicative Privacy homomorphism is introduced in [8]. In this protocol the public parameters are $d>2$ and m. m should have many small divisors and there should be many integers less than m that can be inverted modulo m. The secret parameters are $r \in Z_m$ and m' such that $r^{-1} \bmod m$ exists and a small divisor $m > 1$ of m such that $s := \log_m m$ is an integer.

Encryption operation is performed by randomly splitting $a \in \mathbf{Z}_{m'}$ into secret $a_1, \cdots, a_d$ such that $a = (a_1 + a_2 \ldots + a_i + \ldots a_n) \bmod m'$ and $a_i \in \mathbf{Z}_m$. Compute

$E_k(a) = (a_1 r \bmod m, a_2 r^2 \bmod m, \ldots, a_d r^d \bmod m)$

Decryption operation is performed by computing the scalar product of the j-th coordinate by $r^{-j} \bmod m$ to retrieve $a_j \bmod m$. The plaintext a is a obtained by computing, $(a_1 + \ldots + a_j + \ldots + a_d) \bmod m'$.

This privacy homomorphism has the additive, subtractive, multiplicative and division homomorphism. The privacy homomorphism is secure against chosen ciphertext attacks but not secure against chosen plaintext attacks as shown by Wagner [11].

3. Current security solutions in MANETs

Security solutions in MANETs can be grouped as Secure routing and secure data forwarding. Lets have a look at these solutions in detail.

3.1 Secure routing

There are various secure routing protocols suggested for routing packets in MANETs. One such routing protocol is Secure Routing Protocol (SRP) [12, 13]. In SRP, only the end nodes have to be securely associated, with no need for cryptographic operations at the intermediate nodes. SRP provides one or more route replies, whose correctness is verified by the route "geometry" itself, while compromised and invalid routing information is discarded. Another routing protocol is secure link state protocol (SLSP) [14] for MANETs. Its secure neighbor discovery and the use of neighbor lookup

protocol (NLP) strengthen SLSP against attacks that attempt to exhaust network and node resources. Furthermore, SLSP can operate with minimal or no interactions with a key management entity, while the credentials of only a subset of network nodes are necessary for each node to validate the connectivity information provided by its peers.

3.2 Secure Data Forwarding

We look at two major secure message transmission schemes secure message transmission and threshold cryptography.

3.2.1 Secure Message Transmission

Secure routing is the pre-requisite for implementing secure data forwarding. The motivation is to securely forward data in MANETs in the presence of malicious nodes after the route between the source and target is discovered. There are various schemes proposed for secure data forwarding such as data forwarding based on neighbor's rating, implementing currency system in network for packet exchange, and redundantly dividing and routing message over multiple network routes. For example, Secure Message Transmission (SMT) is a secure data forwarding scheme in which first the active paths are discovered between two nodes using secure routing protocol. Based on N active paths, the message is divided into N different parts such that any M parts can be used to recover this message. These N partial messages are then routed on the recognized paths. The destination can recover a message when M or more partial messages are received. Thus, this scheme ensures that the message reaches the destination even if a few packets are dropped in transit. Both the above security solutions are essential to ensure that the MANETs survive even in the presence of malicious nodes. Thus, by implementing the above solutions the nodes can communicate securely without relying on all nodes on only one route. Extending further the concept of dividing the message using SMT protocol, the threshold cryptography can be implemented to redundantly fragment the message into N parts such that using any t parts the message can be recovered [2, 3, 4].

3.2.2 Threshold Cryptography

Threshold cryptography (TC) [2, 3, 4] involves sharing of a key by multiple individuals called shareholders engaged in encryption or decryption. The objective is to have distributed architecture in a hostile environment. Other than sharing keys or working in distributed manner, TC can be implemented to redundantly split the message into n pieces such that with t or more pieces the original message can be recovered. This ensures secure message transmission between two nodes over n multiple paths. Threshold schemes generally involve key generation, encryption, share generation, share verification, and share combining algorithms. Share generation, for data confidentiality and integrity, is the basic requirement of any TC scheme. Threshold models can be broadly divided into single secret sharing threshold e.g. Shamir's *t-out-of-n* scheme based on Lagrange's interpolation and threshold sharing functions e.g. geometric based threshold. These schemes are being used to implement threshold variants of RSA, ElGamal, and ECC [2, 3].

RSA-TC and ECC-TC has been discussed in the papers [2, 3, 4]. It has been shown that RSA-TC using key sharing is unsuitable in resource constrained MANETs due to high storage, computation, and bandwidth requirements [2].

ECC-TC has been shown to be more efficient for resource constrained MANETs [3]. The authors in paper [3] have used variation of ECC such as Diffie-Hellman (DH), Menezes-Vanstone (MV) and Ertaul in MANETs. They have performed various comparison tests in different scenarios between these different ECCs'. ECC-DH split before encryption has been proved to be better for resource constraint sender as the encryption timings are lowest. ECC-MV split before encryption has been proved to be best for decryption at the resource constraint receiver as the decryption time is lowest. The encryption and decryption time of ECC- MV and ECC-DH has been shown to vary significantly for encryption before split and encryption after split. The encryption and decryption time of ECC-Ertaul has been proved to be more moderate for varying key sizes, t and n for both encryption before split and encryption after split. As a result ECC-Ertaul has been suggested as a best variation of ECC for MANETs [3].

In the next section we show how homomorphic encryption scheme can be used as an alternative for TC to securely forward the message in MANETs.

4. Homomorphic encryption schemes for secure data forwarding in MANETs

In ECC based TC there is an overhead of splitting the message using Lagrange Interpolation scheme. In the proposed new scheme keeping the concept of threshold cryptography in mind, we split the messages and encrypt the message using homomorphic encryption scheme removing the overhead of Lagrange Interpolation scheme all together. In our scheme we increase the success rate to 100% as compared to RSA based TC. The Homomorphic encryption schemes used to encrypt the message are Domingo-Ferrer's new privacy homomorphism [6], Domingo-Ferrer's additive and multiplicative privacy homomorphism [8], Domingo-Ferrer's privacy

homomorphism allowing field operation on encrypted data [7] and MMH [5].

We talk about the proposed new scheme to forward the message securely in this section. We show that even if a node is compromised, the node will not be able to determine the sensitive information. If certain number of nodes are compromised and do not send the message, the message can still be recovered by the destination. The message is encrypted with homomorphic encryption schemes [5, 6, 7, 8].

In the proposed new scheme we are not interested in how the path is established from the sender to the receiver. We are only interested in forwarding the message securely on the already established path. We assume that set of disjoint paths have already been established from the sender to receiver by MANETs routing protocols [15, 16, 17, 18, 19, 20]. We also assume that the key has already been established between the sender and receiver by using any of the key distribution schemes [21].

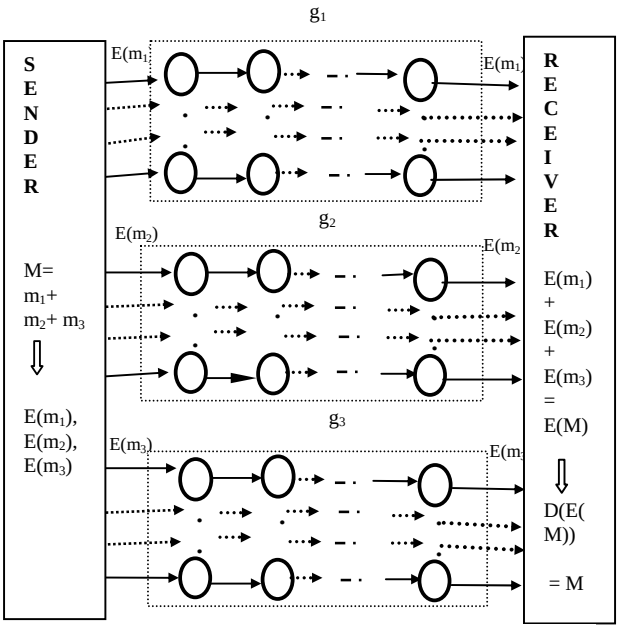


Fig: 1 The proposed new scheme in MANETs

To forward the message securely, the idea is to group the set of *n* disjoint paths from sender to receiver into *g* groups, each group having at least *n/g* active disjoint paths. The message to be forwarded is split into number of messages equal to *g* and encrypted using homomorphic encryption schemes [5, 6, 7, 8]. The encrypted split message is sent to each of the *g* groups, each group having only one encrypted split message. Each node (router) in the group will have the same split message and the node even if compromised will not get the entire message. As Homomorphic encryption schemes are used to encrypt the split message, the receiver can recover the entire encrypted message, by performing addition operation on the encrypted split messages and decryption the entire recovered message. This scheme is illustrated in the *Fig1.*

In MANETs the nodes are always on the move. There will be scenarios where the intermediate node is out of range or may have been killed or out of the MANET all together. In such cases how would the receiver get all the split messages sent by the sender? To ensure that the receiver gets all the split messages, the sender sends the same split messages to more than one disjoint paths. Let us assume that there are *n* disjoint paths and the disjoint paths getting the same split message belongs to one group. Let us assume that there are *g* groups of disjoint path, with each group having atleast *n/g* disjoint paths. The sender splits a message into *g* splits, and sends each split to each group. The receiver recovers the entire message even if at-most *(n/g)-1* disjoint paths are not active. A malicious node cannot recover the entire message as it gets only partial encrypted message. To ensure security the sender does not send more than one split message to the same group of nodes.

5. Implementation of Homomorphic encryption schemes in MANETs

The MANET is simulated using the C programming language [22, 23] in the UNIX environment [24, 25]. The simulation is done on a system having the Intel Pentium-III, 532 MHz CPU and 256 MB system memory running the LINUX kernel –2.6.20-16-generic operating system.

The assumptions during implementation are that there is a sender, receiver and multiple forwarding nodes between them. We assume that set of active disjoint paths have already been established from the sender to receiver by the routing protocols [15, 16, 17, 18, 19, 20]. We also assume that the key for homomorphic encryption scheme has already been established between the sender and receiver by using any of the key distribution schemes [21]. The Homomorphic encryption scheme used to encrypt the message at the sender are Domingo-Ferrer's new privacy homomorphism, Domingo-Ferrer's additive and multiplicative privacy homomorphism, Domingo-Ferrer's privacy homomorphism allowing field operation on encrypted data and MMH.

In our simulation the active disjoint paths getting the same message are grouped as one group. Based on *n* active paths the groups *g* are determined. The sender splits the message and encrypts each split message with the one of the homomorphic encryption schemes. In our network,

n and *g* are fixed to *(10,{2,5,10})*, *(15,{2,7,15})* and *(20,{2,10,20})*. The success rate of our proposed network is computed as,

*(No. of messages recovered by the receiver/No. of messages sent by the sender)*100 … (1)*

The success rate of the network with *n* and *g* fixed to *(10,{2,5,10})*, *(15,{2,7,15})* and *(20,{2,10,20})* is determined by randomly killing the nodes. The nodes are killed randomly by using Exponential distribution provided by the function in GSL library [26].

In our implementation, the sender first splits the message into *g* partial messages where each partial message is sent to one of the *g* groups of the MANETs. Each of the partial messages are associated with a unique *msg split id*. All the *msg split id's* of the partial messages forming the entire message is summed up to set up the *msg split id sum*. The *msg id, msg split id, msg spit id sum* and encrypted partial text is placed in the buffer so that the receiver can recover the entire message from the partial encrypted message. To recover the entire message sent by the sender, the receiver follows two steps. In the first step the receiver adds up all the partial encrypted message whose *msg id's* are same and *msg split id's* sums up to *msg split id sum*. In the second step the receiver decrypts the sum of all partial encrypted messages to recover the entire message. As the same encrypted partial message is sent to all the active paths in the group the receiver is likely to get the same redundant message. The receiver discards the redundant message by discarding the already seen message with the same *msg id* and *msg split id*.

In the next section we look at the buffer structure of the encrypted message.

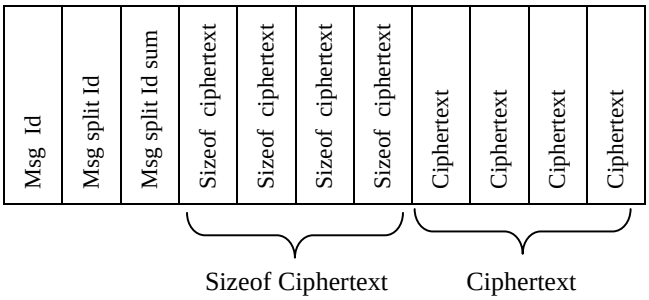
5.1 Buffer structure of the encrypted message

The size of the buffer structure of the encrypted message sent form sender to receiver varies from one homomorphic encryption to another.

5.1.1 Domingo-Ferrer's new privacy homomorphism (DF's new PH)

In DF's new PH the size of the ciphertext increases with the increase in the encryption split *"d"*. So the size of the buffer increases with the increase of the parameter *d* used in encryption.

Table: 1 Buffer structure of message encrypted with DF's new PH with *d=2*



In *Table 1* the *msg Id* field identifies different messages encrypted at the sender. The messages split at the sender is uniquely identified by *msg split Id*. The sum of all the message split id is included in *msg split id sum*. The rest of the buffer is used to contain the size of the cipher data and the ciphertext itself. The size of the ciphertext is essential in recovering the ciphertext by the receiver. The receiver recovers the entire message by adding up all the cipher values with the same *msg id* and whose *msg split id's* adds up to *msg split id sum*.

5.1.2 Domingo-Ferrer's additive and multiplicative privacy homomorphism (DF's additive and multiplicative PH)

In DF's additive and multiplicative PH the size of the ciphertext increases with the increase in the encryption split *"d"*. So the size of the buffer increases with the increase of the parameter *d* used in encryption.

Table: 2 Buffer structure of message encrypted with DF's additive and multiplicative PH with *d=2*

Msg Id	Msg split Id	Msg split Id sum	Sizeof cipher Text	Sizeof cipher Text	Cipher Text	Cipher Text
			Sizeof Ciphertext		Ciphertext	

In *Table 2* the *msg id* field identifies different messages encrypted at the sender. The messages split at the sender is uniquely identified by *msg split id*. The sum of all the msg split id is included in *msg split id sum*. The rest of the buffer is used to contain the size of the ciphertext and the ciphertext itself. The size of the ciphertext is essential in reading the ciphertext from the buffer. The receiver recovers the entire message by adding up all the ciphertexts with the same *msg id* and whose *msg split id's* adds up to *msg split id sum*.

5.1.3 Domingo-Ferrer's privacy homomorphism allowing field operations on encrypted data (DF's field PH)

In DF's field PH the buffer structure is represented in *Table 3.*

Table: 3 Buffer structure of message encrypted with DF's field PH

Msg Id	Msg split Id	Mesg split Id Sum	Size of Cipher Text	Cipher Text

In *Table 3* the *msg id* field identifies different messages encrypted at the sender. The messages split at the sender is uniquely identified by *msg split id.* The sum of all the *msg split id* is included in *msg split id sum*. The rest of the buffer is used to contain the size of the ciphertext and the ciphertext itself. The size of the ciphertext is essential in reading the ciphertext from the buffer. The receiver recovers the entire message by adding up all the cipher values with the same message id and whose *msg split id's* adds up to *msg split id sum.*

5.1.4 Mixed Multiplicative Homomorphism (MMH)

In MMH the buffer structure is represented in *Table 4.*

Table: 4 Buffer structure of message encrypted with MMH

Msg Id	Msg split Id	Msg split id sum	Size of cipher text	Cipher Text

In *Table 4* the *msg id* field identifies different messages encrypted at the sender. The messages split at the sender is uniquely identified by *msg split id*. The sum of all the *msg split id* is included in *msg split id sum*. The rest of the buffer is used to contain the size of the ciphertext and the ciphertext itself. The size of the ciphertext is essential in reading the ciphertext from the buffer. The receiver recovers the entire message by adding up all the cipher values with the same message id and whose *msg split id's* adds up to *msg split id sum.*

5.2 Performance results of Homomorphic encryption schemes

In MANETs the nodes may have low computational power. In such cases we need to find an encryption scheme, which is computational much faster. In our implementation we do various tests to find a relatively faster encryption schemes among DF's new PH, DF's additive and multiplicative PH, DF's field PH and MMH.

In one of our tests we determine the encryption timing of all four encryption schemes by varying the key size to 512, 1024, 2048 bits and keeping the message size fixed to 512 bits. In another test we find the execution timing of all the four encryption schemes by keeping the key size fixed to 512 bits, 1024 bits, 2048 bits and varying message size. The timings are determined over 200 runs.

Fig 2 represents the execution timing of DF's new PH, DF's additive and multiplicative PH, DF's field PH and MMH in micro seconds by varying the key size to 512, 1024, 2048 bits and keeping the message size fixed to 512 bits. From *Fig 2*, it is clear that MMH is much faster than DF's new PH, DF's additive and multiplicative PH and DF's field PH. We also see that the encryption timing of DF's new PH, DF's additive and multiplicative PH and DF's field PH increases with the increase in encryption keys but the encryption timing of MMH remains almost the same with the increase in the encryption key size.

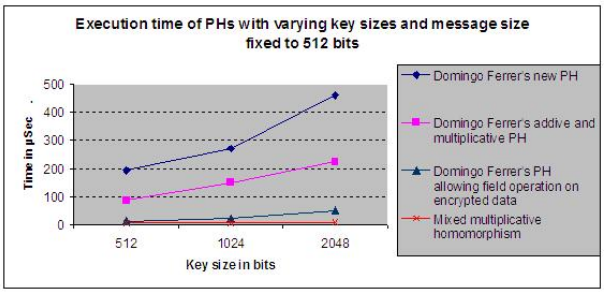


Fig: 2 Execution time of PHs with varying key sizes and message size fixed to 512 bits

Fig 3 represents the execution timing of DF's new PH, DF's additive and multiplicative PH, DF's field PH and MMH in micro seconds by increasing the message size to 100, 250 and 500 bits and by keeping the key size fixed to 512 bits. From *Fig 3*, it is clear that MMH is much faster than DF's new PH, DF's additive and multiplicative PH and DF's field PH. We also see that the encryption timing of DF's new PH and DF's additive and multiplicative PH increases with the increase in message size but the encryption timing of DF's field PH and MMH remains almost the same with the increase in the message size.

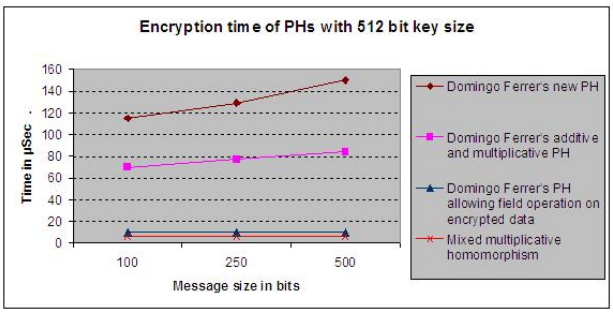


Fig: 3 Execution time of PHs in µSec with 512 bit key size

Fig 4 represents the execution timing of DF's new PH, DF's additive and multiplicative PH, DF's field PH and MMH in micro seconds by increasing the message size to 250, 500 and 1000 bits and by keeping the key size fixed to 1024 bits. From *Fig 4*, it is clear that MMH is much faster than DF's new PH, DF's additive and multiplicative PH and DF's field PH. We also see that the encryption timing of DF's new PH and DF's additive and multiplicative PH increases with the increase in message size but the encryption timing of DF's field PH and MMH remains almost the same with the increase in the message size.

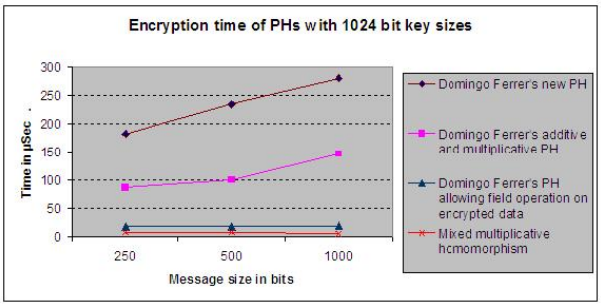


Fig: 4 Execution time of PHs in µSec with 1024 bit key size

Fig 5 represents the execution timing of DF's new PH, DF's additive and multiplicative PH, DF's field PH and MMH in micro seconds by increasing the message size to 500, 1000 and 2000 bits and by keeping the key size fixed to 2048 bits. From *Fig 5,* it is clear that MMH is much faster than DF's new PH, DF's additive and multiplicative PH and DF's field PH. We also see that the encryption timing of DF's new PH and DF's additive and multiplicative PH increases with the increase in message size but the encryption timing of DF's field PH and MMH remains almost the same with the increase in the message size.

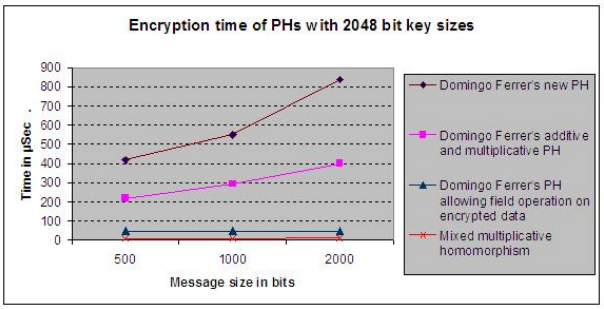


Fig: 5 Execution time of PHs in µSec with 2048 bit key size

From *Fig 2, Fig 3, Fig 4 and Fig 5* it is clear that MMH is much faster than DF's new PH, DF's additive and multiplicative PH and DF's field PH. We also see from *Fig 2* that the encryption timing of DF's new PH, DF's additive and multiplicative PH and DF's field PH increases with the increase in encryption keys but the encryption timing of MMH remains almost the same with the increase in the encryption key size. From *Fig 3, Fig 4* and *Fig 5* we also see that the encryption timing of DF's new PH and DF's additive and multiplicative PH increases with the increase in message size. However the encryption timing of DF's field PH and MMH remains almost the same with the increase in the message size. In determining the encryption timing of DF's new PH and DF's additive and multiplicative PH, the encryption split (*d*) is fixed to the value 2.

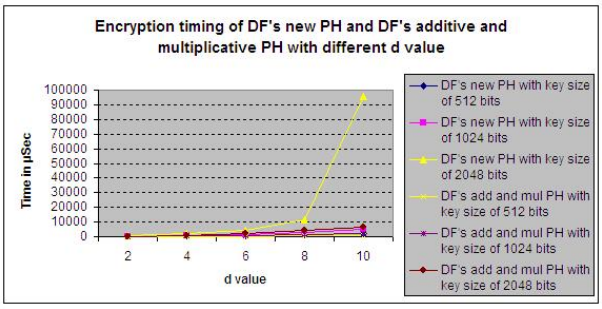


Fig: 6 Execution time of DF's new Ph and DF's additive and multiplicative PH in µSec with varying d

From the *Fig 6* we see that the execution timing in micro seconds increases with the increase in key size and *d* (encryption split) value. Furthermore we also see that the encryption timing for DF's new PH increases dramatically with the *d=10.* We also see that the encryption timing of DF's additive and multiplicative PH is faster than DF's new PH. The encryption scheme is said to be more secure with the increase with the increase in *d* value but with the increase in d value the encryption timing also increases. So we need to determine the value of *d* so that the encryption time is less and provides high security. The *d* value set to *4* seems reasonable for these encryption schemes.

5.3 Performance Results of the proposed new scheme in MANETs

In MANETs the nodes are always on the move and there may be scenarios where the active path may no longer be active and as a result, the receiver may not receive all the packets sent by the sender. The success rate of the network is computed as in *equation 1*. *Fig 7* depicts the success rate of the networks with *n* active paths and *g* groups fixed to *(10,{2,5,10})*, *(15,{2,7,15})* and *(20,{2,10,20}),* by randomly killing the nodes. The nodes in the networks are killed randomly by using Exponential distribution provided by the function in GSL library [26]. The networks with *n* and *g* fixed to *(10,{2,5,10})* defines 3

sets of networks with the first network having *10* active paths, *2* groups and *5* active paths in each group, second network with *10* active paths, *5* groups and *2* active paths in each group and third network with *10* active paths, *10* groups and *1* active path in each group. The networks with *n* and *g* fixed to *(15,{2,7,15})* defines 3 sets of networks with first network having *15* active paths, *2* groups and *7* active paths in one group and *8* active paths in another group, second network with *15* active paths, *7* groups and *3* active paths in one group and *2* active paths in remaining groups and third network with 15 active paths, *15* groups and *1* active path in each group. The networks with *n* and *g* fixed to *(20,{2,10,20})* defines 3 sets of networks with first network having *20* active paths, *2* groups and *10* active paths in each group, second network with *20* active paths, 10 groups and 2 active paths in each group and third network with 10 active paths, 20 groups and 1 active path in each group.

From *Fig 7* it is clear that the success rate increases by reducing the number of groups in the network. This is because by reducing the number of groups in the network we would increase the number of active paths in each group. Just one partial message from each group is enough to recover the entire message. From *Fig 7* we see that the success rate is 100% with *g=2* and *n=10,15,20.* This is because by increasing the number of paths in each group, the probability of one path in each group remaining active is high and with it the probability of recovery of the message at the receiver is also high. The success rate gradually decreases with the gradual increase in the number of groups in the network. With *g=n* we see that success rate is lesser than 50%. Therefore to get the success rate as 100% in the network it is better to reduce the number of groups, thus increasing the number of active paths in each group.

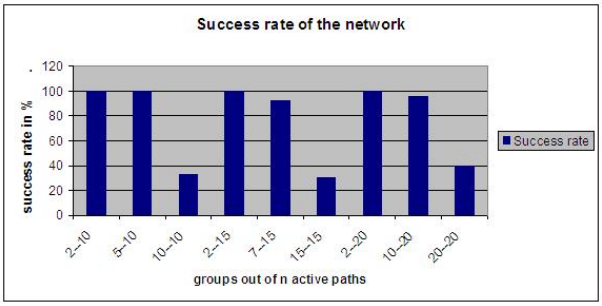


Fig: 7 Success rate of the Network

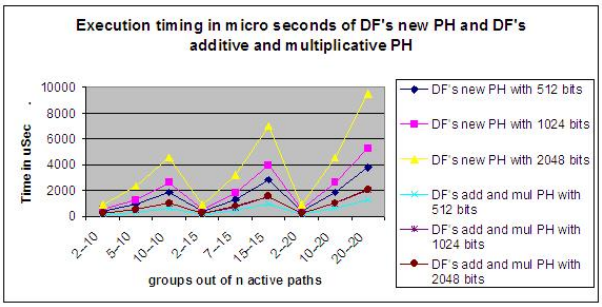


Fig: 8 Encryption timing of DF's new Ph and DF's additive and multiplicative PH in micro Seconds

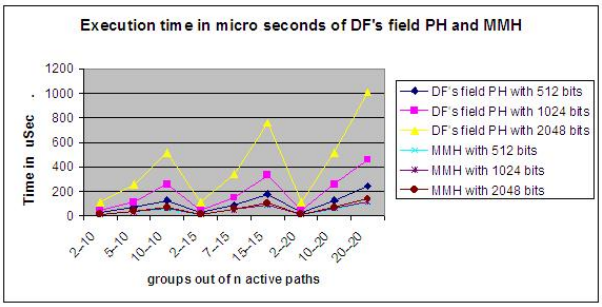


Fig: 9 Encryption timing of DF's field Ph and DF's MMH in micro Seconds

In this proposed new scheme in MANETs the sender splits the message with respect to the value *g*. The sender using the homomorphic encryption scheme then encrypts all the split messages. As the number of splits at the sender is equal to the value *g* the total encryption timing of all the split messages increase with the value *g. Fig 8* and *Fig 9,* represents the total encryption timing of all the split messages. From the *Figures* it is clear that the total encryption timing increase with the value *g.* Also from *Figures* we see that MMH is the fastest encryption scheme, followed by DF's field PH, DF's additive and multiplicative PH and finally DF's new PH.

6. Alternative scheme for DF's new Ph and DF's additive and multiplicative PH

In DF's new PH and DF's additive and multiplicative PH the encrypted message results in *d* partial ciphertexts depending on the papremeter *d*. With the increase in d the size of the ciphertext increases and so does the buffer's structure as shown in *Table 1* and *Table 2.*

In order to keep the buffer stucture and so the packet size almost constant, the sender encrypts the message with either DF's new PH and DF's additive and multiplicative

PH with the parameter *d* set to the value *g*. The sender then sends one of the partial ciphertexts to each of the groups *g* in the network. The receiver recovers the entire message by arranging all the partial ciphertexts in appropriate order to get the entire ciphertand then decrypting the ciphertext to get the message. This scheme is illustrated in the *Fig 10.*

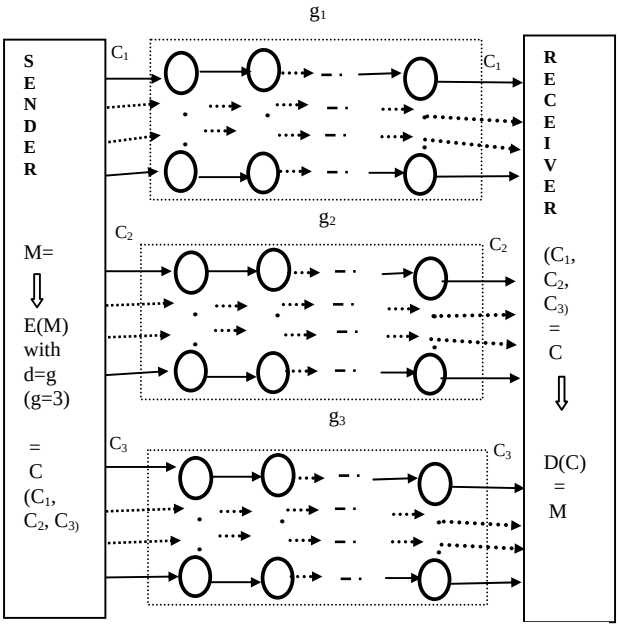


Fig: 10 The Alternative new scheme in MANETs for DF's new Ph and DF's additive and multiplicative PH

In this scheme with the increase in the number of groups *g* in the network the encryption is done only ones by setting the value *d* of the encryption scheme to *g*. But in the original proposed new scheme we do encryptions *g* times for *g* groups. In this alternative proposed new scheme the encryption timing of DF's new PH and DF's additive and multiplicative PH is lesser than the original proposed scheme when the encryption schemes in both the proposed schemes uses the same *d* value. However from *Fig 6* and *Fig 8* we see that the encryption timing of the original proposed new scheme as in *Fig 1* is lesser than the encryption timing of the alternative proposed scheme as in *Fig 10* when the groups *g* in the network increases.

The security issues of both the proposed new schemes are the same. In the alternative proposed new scheme, a single compromised node would not be able to determine the message as the node would get only the partial ciphertext.

7. Conclusion

By using the proposed new scheme in MANETs as an alternative to RSA-TC and ECC-TC, we eliminate the overhead of Lagrange Interpolation Scheme associated with RSA-TC and ECC-TC. Furthermore in our scheme because of the grouped MANETs, if one of the node is compromised the entire message would not be revealed. For the entire message to be recoverd by the attacker, the attacker needs to compromise atleast *g* nodes, one node from each group *g* and know the encryption keys to decrypt the message. The success rate of the proposed new scheme is 100% if there are more number of active paths in each group of the network.

From our implementation results it is clear that MMH is the fastest homomorphic encryption scheme in comparison with DF's new PH, DF's additive and multiplicative PH and DF's field PH. But MMH homomorphic encryption scheme is susceptible to known plaintext attack.

In the propose new scheme the buffer size and so the packet size increases with the increase in the value *d* used in DF's new PH and DF's additive and multiplicative PH. To keep the buffer size constant for DF's new PH and DF's additve and multiplicative PH we propose an alternative new scheme for MANETs. The alternative proposed new scheme is more efficient than the proposed new scheme for MANETs when both schemes uses the same *d* (encryption split) value for DF's new PH and DF's additive and multiplicative PH.

References

[1] P. Papadimitratos, Z.J. Haas"Secure Data Transmission in Mobile Ad Hoc Networks," ACM Workshop on Wireless Security (WiSe 2003), San Diego, CA, September 19, 2003.

[2] L. Ertaul, N. Chavan, " Security of Ad Hoc Networks and Threshold Cryptography", 2005 International Conference on Wireless Networks, Communications, and Mobile Computing, Wirelesscom 2005, MobiWac 2005, June 2005, Maui, Hawaii.

[3] L. Ertaul, N. Chavan, "Elliptic Curve Cryptography based Threshold Cryptography (ECC-TC) Implementation for MANETs", IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.4, pp 48-61 April 2007.

[4] L. Ertaul, W. Lu, "ECC based Threshold Cryptography for Secure Data Forwarding and Secure Key Exchange in Mobile Ad Hoc Networks (MANET) I", Proc. of Networking 2005 International Conference, University of Waterloo, Ontario, CA, May 2005.

[5] Hyungjick Lee, Jim Alves-Foss,Scott Harrison, " The use of Encrypted Functions for Mobile Agent Security",Proceedings of the 37th Hawaii International Conference on System Sciences – 2004.

[6] J. Domingo-Ferrer, “A new Privacy Homomorphism and Applications”, Elsevier North-Holland, Inc, 1996.
[7] J. Domingo-Ferrer and J. Herrera-Joancomarti. “A privacy homomorphism allowing field operations on encrypted data”. I Jornades de Matematica Discreta i Algorismica, Universitat Politecnica de Catalunya, March 1998.
[8] J. Domingo-Ferrer. “A Provably Secure Additive and Multiplicative Privacy Homomorphism”. Information Security Conference, LNCS 2433, pp 471–483, January 2002.
[9] William Stallings “Cryptography and Network Security”, Third Edition, Chinese Remainder Theorem (CRT), pp. 245-247.
[10] Jung Hee Cheon, Hyun Soon Nam,”A Cryptanalysis of the Original Domingo-Ferrer's Algebraic Privacy Homorphism”, http://eprint.iacr.org/2003/221.pdf
[11] D. Wagner, “Cryptanalysis of an algebraic privacy homomorphism”, In proceedings of the 6th information security conference(ISC03), Bristol, UK, October 2003.
[12] P. Papadimitratos and Z.J. Haas, “Secure Routing for Mobile Ad Hoc Networks,” in Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), San Antonio, TX, Jan. 27-31, 2002.
[13] P. Papadimitratos, Z.J. Haas, and P. Samar, “The Secure Routing Protocol (SRP) for Ad Hoc Networks,” Internet Draft, draft-papadimitratos-secure-routing-protocol-00.txt, Dec. 2002.
[14] P. Papadimitratos and Z.J. Haas, “Secure Link State Routing for Mobile Ad Hoc Networks,” in Proceedings of the IEEE CS Workshop on Security and Assurance in Ad hoc Networks, in conjunction with the 2003 International Symposium on Applications and the Internet, Orlando, FL, Jan. 2003.
[15] D. Johnson, D. A. Maltz, and J. Broch. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (Internet-Draft). Mobile Ad-hoc Network (MANET) Working Group, IETF, October 1999.
[16] S. Corson and V. Park. Temporally-Ordered Routing Algorithm (TORA) Version 1 Functional Speci_cation. Mobile Ad-hoc Network (MANET) Working Group, IETF, October 1999.
[17] S. Das, C. E. Perkins and E. M. Royer. Ad Hoc On Demand Distance Vector (AODV) Routing (Internet-Draft). Mobile Ad-hoc Network (MANET) Working Group, IETF, October 1999.
[18] J.J. Garcia-Luna-Aceves and M. Spohn. Source-Tree Routing in Wireless Networks. In Proceedings IEEE ICNP 99: 7th International Conference on Network Protocols, Toronto, Canada, October 31_November 3, 1999.
[19] IETF MANET Working Group Internet Drafts. http://www.ietf.org/ids.by.wg/manet.html.
[20] C.E. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In Proceedings of the SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications , pages 234-244, August 1994.
[21] M.Ramkumar, N. Memon, “An Efficient Key Predistribution Scheme for MANET Security,” submitted to the IEEE Journal on Selected Areas of Communication.
[22] C programming, http://www.cs.cf.ac.uk/Dave/C/
[23] Barry M. Austell-Wolfson, R. Derek Otieno, “Complete Book of C Programming”, Prentice Hall, 1999.
[24] W. Richard Stevens, “UNIX Network Programming Volume 1, Networking APIs: Sockets and XTI”, Second Edition, Prentice Hall, 1998.
[25] W. Richard Stevens, “Unix Network Programming, Volume 2, Interprocess Communication”, Second Edition, Prentice Hall, 1999.
[26] GSL manual, http://www.gnu.org/software/gsl/manual/html_node/Random-Number-Distributions.html

Levent Ertaul received B.Sc. from Anatolia University Turkey, in 1984, M.Sc. from Hacettepe University, Turkey, in 1987, and PhD degree from Sussex University, UK, in 1994. After working as an assistant professor (from 1994) in the Dept. of Electrical & Electronics Engineering, Hacettepe University, he moved to California State University, East Bay in 2002. He is currently a full time Asst. professor at California State University Eastbay, USA in the department of Math & Computer Science. He is actively involved in security projects nationally and internationally. His current research interests are Wireless Security, Ad Hoc Security, Security in WSNs and Cryptography. He has numerous publications in security issues.

Vaidehi received her B.E in Computer Science from the university of Nitte Mahalinga Adhayanthaya Memorial Institute of Technology (NMAMIT) in the year 2003. Currently, she is pursuing M.S in Computer Science at California State University, East Bay. She has publications in International Conferences in the security aspects of Computer Networks and Wireless Sensor Networks.