Wireless Netw (2015) 21:1243–1258
DOI 10.1007/s11276-014-0848-1

Multi-party encryption (MPE): secure communications in delay tolerant networks

Roy Cabaniss · Vimal Kumar · Sanjay Madria

Published online: 16 November 2014

Abstract Mobile ad hoc networks are a subset of delay tolerant networks (DTNs) composed of several mobile devices. These dynamic environments make conventional security algorithms unreliable; nodes located far apart from each other may not have access (available) to each other's public keys or have doubt on the validity of public-keys, making secure message exchange difficult. Furthermore, ad hoc networks are likely to be highly compromised and therefore may be untrusted. Other security methods, such as identity-based encryption and Kerberos, rely on requesting key data from a trusted third party, which can be unavailable or compromised in a DTN like environment. The purpose of this paper is to introduce two security overlay networks capable of delivering messages securely, preventing both eavesdropping and alteration of messages. The first algorithm, Chaining, uses multiple midpoints to re-encrypt the message for the destination node. The second, Fragmenting, separates the message key into pieces that are routed and secured independently from each other. Both techniques improve security in hostile environments; under test conditions, Chaining reduces the number of messages intercepted by 90 %, and Fragmenting by 83 %. This improvement has a performance trade-off, however, reducing the delivery ratio by 63 % in both algorithms.

Keywords Security · Chaining encryption · Delay tolerant networks · Threshold encryption · Fragmented keys

R. Cabaniss · V. Kumar · S. Madria (✉)
Department of Computer Science, Missouri University of Science and Technology, Rolla, MO, USA
e-mail: madrias@mst.edu; rac343@mst.edu

1 Introduction

Secure communication is a base requirement for many wireless computing applications. Thus, any effective delay tolerant networking (DTN) implementation must have a security system capable of routing messages without allowing an adversary to access or modify those messages. With the advent of public-key cryptography, private communication without any direct interaction is now feasible by providing the public key over unsecured networks. To communicate securely, the origin node must verify the key's association with the destination. In conventional communications, this problem is solved by using a trusted key repository to store and verify the keys used. However, this technique is infeasible in dynamic DTN in which the trusted party may be unreachable. This limitation forces nodes to handle their own key distribution and verification [4].

Methods are available that verify a node's identity, ranging from the very low tech (physical contact) to the more sophisticated (measuring the performance traits of the devices) [23]. However, these methods require point-to-point contact and therefore are difficult over any distance. When a node wants to send a message to another in direct contact or has already stored the destination's public key, the problem of secure transmission is trivial. However, in a DTN, validation of a public key [19] may not be feasible. Also, in DTNs connections can be short-lived and message delivery may be delayed. Thus, public keys may be inaccessible or untrusted. In such an environment, a node can encrypt a message with a key of a trusted node or multiple semi-trusted nodes and these nodes will be responsible for secure delivery of messages when connections are established. The intent of multi-party encryption (MPE) is to send a message securely to a destination without having access to the destination's public key for reasons explained above.

This secure message can include security data, allowing conventional communications to be performed afterward the initial secure transmission. This task is performed by routing the message through the keys of other nodes and asking the midpoint to re-encrypt the message for the destination node.

This algorithm can be used to route messages securely through a series of allies (any of whom may or may not be compromised). If a member of the Red army wishes to communicate securely with another Red, he can do so using his public key, which he should already have. If he wants to communicate with a member of the Blue army, however, he may not have the key available or trust the public key. Rather than wait for direct contact with the destination to obtain the key, he can route the message through a series of members to ensure its security. Note that armies use vehicle to vehicle communications or laptops supported by high-capacity batteries, therefore the scheme's higher energy consumption make it useful in such environments. Also, constant communication using MPE is unnecessary, as a secure communication line can be established via an MPE scheme only when required, using conventional encryption afterwards. Another application environment is in a unmanned aerial vehicle (UAV) ad hoc network, which have a much higher battery life. Users can consistently use this scheme to secure communications when public keys may not be accessible.

The objective of this paper is to analyze two MPE schemes: Chaining (Sect. 3.1) and Fragmenting (Sect. 3.2) in a DTN environment for secure delivery of messages. The Chaining algorithm secures messages by sending them through a series of midpoints in the delay tolerant network; each midpoint must decrypt and re-encrypt the message before forwarding it to the final destination. Because the message is always encrypted with multiple layers, no midpoint will have access to the original message. The Fragmenting algorithm functions similarly, sending the message through multiple midpoints in parallel. The message is broken into several fragments. Any node attempting to read the original message must have access to many (although not necessarily all) of the fragments. Simulation results indicate that these schemes increase security significantly. Chaining reduces the number of compromised messages by as much as 90 %, while Fragmenting does so by 83 %. However, there is a trade-off. Chaining reduces the delivery ratio to 62 %, while increasing the average delivery time by a factor of 4. The Fragmenting algorithm performs better than Chaining, reducing the delivery ratio to 81 % and increasing the delivery time by a factor of 3.7. The security can be further improved by increasing the number of links or fragments used by both algorithms. This customization allows the user to determine which security levels and performance penalty, allowing the algorithm to be used in a wide range of applications.

2 Background

A wide variety of advanced encryption methods are available to enable secure communications. Key distribution techniques give nodes access to public keys in the network. Our algorithms approach the problem of secure communications by expanding on these base methods. Normally, when two encryption keys can be used on the same message, they must be encrypted and decrypted in a last-in-first-off order. Commutative encryption is a class of encryption algorithms that can be applied and removed in an arbitrary order. This class of encryption techniques allows multiple encryptions to be applied to a message, thereby increasing the security of the system. Sample commutative encryption techniques include the Shamir algorithm [24], the Massey-Omura algorithm [16], and the El-Gamal re-encryption scheme [9]. Delay tolerant networks (DTNs) are designed to work in unreliable environments in which messages are corrupted or dropped. While this unreliability can be reduced by both message replication and resubmission, these solutions complicate security. Alternative methods can be used to allow a message to be decrypted if portions of the original key are lost. Using Shamir's (k, n) scheme [22], a message can be encrypted with n keys, requiring k shares to decrypt (where k is always less than or equal to n). This technique allows complex security systems to be implemented in unreliable networks.

Onion routing, developed by Syverson et al. [25], allows secure, anonymous communications in a static network. This method is based on establishing secure communications through a series of proxies, each of which only knows their incoming and outgoing proxy. By limiting data, traffic is forwarded without the destination being aware of the source of the messages, and without any of the nodes along the route being aware of the message's contents. The algorithm cannot be implemented in a MANET due to the unavailable or untrusted list of secure nodes; however, it serves as a demonstration of multiple midpoint encryption, which was expanded upon to design the Chaining overlay network.

Node mobility means that proxies may be unavailable or, at the very least, expensive to reach when implementing Tor in a DTN. The designers of the EnPassant [26] scheme expanded on the onion routing scheme with groups of proxies. Both the delivery time and ratio improve because any group members are allowed to act as a proxy, both decrypting and forwarding a data stream. Anonymity is preserved by forcing a message to follow indirect routes. This scheme is functional, but only under certain assumptions regarding the attacker. First, this scheme is very vulnerable to global eavesdropping adversaries; if all messages can be followed through the midpoints, randomizing the route has no benefit as the messages still can be tracked. This is a general weakness among DTN security schemes,

however. A larger issue is that using group keys makes the scheme very vulnerable to Byzantine attacks [27]. A single compromised node jeopardizes the security of the entire group. If a member of each group is compromised, all traffic can be tracked. Despite these vulnerabilities, this scheme prevents traffic analyses fairly well.

El Defrawy et al. [8] proposed a method by which to securely communicate messages in a DTN. With this algorithm, a node lacking the destination's public key can encrypt the message with the public keys of several nodes near the destination, in terms of either physical proximity or contact frequency. While secure, this algorithm has the issue of having to maintain contact information for several nodes in the network. Bhutta et al.'s scheme [2] uses a centralized system for key management which combines the use of PKI and a Kerberos like system. Nodes in a DTN however, may not always have access to a centralized PKI certificate authority and the authentication server of Kerberos. To alleviate such concerns [21] introduced a hierarchical identity based scheme. With identity based encryption, the sender only needs to know the destination's identity, however, it still needs a central server which possesses the master key. Hierarchical Identity based solution proposed in [21] does away with the need of a central server by introducing a logical hierarchy of servers. The drawback therefore, in this scheme is the need of a fixed infrastructure which would facilitate secure communication.

Another algorithm proposed for comparison, the Poor Man's approach, fractures the key into several parts. Each fragment follows a different route to the destination. The random key then is split into several other sub-keys such that all fragments must be accessed to retrieve the original message. Using a bitwise XOR is the easiest method $(K = K_1 \oplus K_2 \oplus K_3 \ldots)$. The encrypted message and all of the fragments are sent directly to the destination, but with a time lag so that each fragment will follow a different route. The purpose is to ensure that only the destination node receives all key fragments. If an adversary can monitor all transmissions (either by global eavesdropping or by being located on the sole path between the nodes), it can retrieve the key as easily as the destination can. This method, expanded on by the MPE algorithms, is simple to implement but is only secure as long as the routing algorithm forwards each fragment independently. Even with these limitations, these approaches to keyless secure communication serve as a foundation for future algorithms.

It should be noted that while at first, because of the similarity in name it appears multi-party computation schemes [6, 7] can provide a solution to the problem, it is not so. Multi party computation schemes require collaboration among all the involved nodes. In the context of this paper and DTNs in general, this may not be always possible as the sender may not always know the public key of the receiver.

The preliminary idea of this paper has appeared in the Ph.D. forum paper [3].

3 Proposed algorithm

The algorithms presented in this section are designed to function in large-scale DTNs. They assume that the message will be processed by untrusted nodes (hence the need for encryption at every step). Each node keeps a subset of the public keys available, referred to as the keychain, maintained by any number of key distribution techniques [5, 12, 29]. The following subsections describe the algorithms designed to use partial keychains to improve message confidentiality.

3.1 Chaining encryption

Chaining encryption forces the algorithm to route through k nodes, known as links, without allowing any links to access the plaintext. The original message is first encrypted with the public key of each link and then routed to the nearest link. At each link, that node's encryption layer is removed. If the link has access to the final destination's public key, then the message is encrypted with it. Otherwise, the link encrypts the message for a node in its keychain, adding that node to the link chain. Only when each layer has been replaced with the final destination's key can the message be forwarded to the endpoint. Once there, it is decrypted k times, each time removing a layer of encryption from the message.

This method is very secure, avoiding pitfalls inherent to the normal key-exchange sequence and Fragmenting method. The only way it can be broken is if all of the links are compromised by an adversary. Another advantage is that it can be either scaled up for high-risk networks or down for more casual security by changing the number of links required. The trade-off is that this algorithm requires a message to travel to numerous midpoints, increasing the message delivery time considerably. Another trade-off is that a link compromised by the adversary can stop message delivery. The compromised link either can refuse to forward a message or can reencrypt it with its own key. The latter can allow the adversary to read the message if all of the link's encryption layers are replaced with the adversary's. If the message uses a compromised node as a link, the message cannot be read by the adversary unless all links are compromised. However, it cannot be read by the destination either, resulting in a dropped message.

For example, Alice, a member of the Red army, sends a secure message to Bob, a member of the Blue army. If Alice possesses Bob's public key, she simply encrypts the message and forwards it. Failing that, she may attempt normal key distribution techniques, asking other members of the Red army (who she assumes are trusted) if they have the destination key. If they do not, she will select two midpoints, Chuck and David (Alice must possess the public key for both), and encrypt the message for both—$enc_{Chuck}(enc_{David}(Msg))$. Because she may not trust them, she layers the encryption to ensure that the message cannot be read by either party. The message is routed to Chuck first simply by virtue of his proximity. Chuck then removes his layer of encryption, leaving $enc_{David}(Msg)$. Chuck cannot read the message, so he obeys the protocol, re-encrypts it with Bob's key, and forwards it to David—$enc_{Bob}(enc_{David}(Msg))$. David cannot read the message either, so he follows the algorithm, removing his layer of encryption and forwarding it to Bob—$enc_{Bob}(enc_{Bob}(Msg))$. Once Bob receives the message, he has no problem removing both layers of encryption and retrieving the original message.

This example of execution changes if one of the midpoints is compromised. If Chuck had been compromised, he would not have encrypted the message for Bob. Even if Chuck cannot read the message, he can refuse to forward it, causing the message delivery to fail. If he wants to read the message, he can re-encrypt it with the adversary key and forward it to David—$enc_{Adv}(enc_{David}(Msg))$. David, in this case, does not possess Bob's public key. Thus, he randomly selects midpoint Eric, forwarding the message on—$enc_{Eric}(enc_{Adv}(Msg))$. If Eric is compromised and working with Chuck, he can remove his layer of encryption and Chuck's adversary key to retrieve the original message. Only through their collaboration can the message be compromised. If Eric is not compromised, he removes his layer of encryption, encrypts the message for Bob, and forwards it—$enc_{Bob}(enc_{Adv}(Msg))$. In this case (when a portion of the chain was compromised), the final intended destination cannot read the message, but neither can the adversary.

The Chaining method, therefore, has three possible outcomes. If all midpoints are uncompromised, the message is delivered successfully and securely. When all members are both compromised and collaborating, the message is compromised. If some are compromised and some not, or if they are not collaborating, the message delivery fails—a midpoint either refuses to forward the message or encrypts it with the wrong key. A detailed look at these relative probabilities is presented in Sect. 4.

3.2 Fragmenting encryption

The trade-off for Chaining's increased security is its significantly increased delivery time. The Fragmenting method, rather than sending messages sequentially through links, will create multiple message fragments and send them simultaneously. Using threshold encryption [11], the message is encrypted into several subkeys. This allows the final destination to decrypt the message even if fragments are compromised by the adversary or dropped. Each fragment is encrypted and forwarded through a single link. Because each fragment is routed through a single midpoint, this technique takes less delivery time than Chaining.

Algorithm 1 Chained Encryption

Notation
k - The number of links through which a message must be routed
$Node_{Origin}$ - Origin of msg
msg_{dest} - Final destination of msg
$plaintext$ - Original message to be sent to msg_{dest}
$Node_A$ - Arbitrary node in MANET
$Keychain_A$ - Set of Public Keys to which $Node_A$ has access
$Enc_A(text)$ - Encrypted form of $text$ using public key of $Node_A$
$Dec_A(text)$ - Decrypted form of $text$ using private key $Node_A$

Trigger - $Node_{Origin}$ wants to send msg to $Node_{dest}$
$msg_{text} \leftarrow plaintext$
for $i = 1 \rightarrow k$ **do**
  $Node_i \leftarrow$ Random Node from $Keychain_{Origin}$
  $msg_{mid} \leftarrow msg_{mid} \cup Node_i$
  $msg_{text} \leftarrow Enc_{Node_i}(msg_{text})$
  /* Message is now encrypted commutatively */
end for
Route to nearest node in msg_{mid}

Trigger - $Node_A$ receives msg
if $Node_A = msg_{dest}$ **and** $msg_{mid} = \emptyset$ **then**
  for $i = 1 \rightarrow k$ **do**
    $msg_{text} \leftarrow Dec_{dest}(msg_{text})$
  end for
  Message has been delivered
else
  if $Node_A \in msg_{mid}$ **then**
    $msg_{mid} \leftarrow msg_{mid} - Node_A$
    $msg_{text} \leftarrow Dec_A(msg_{text})$
    /* Since msg was encrypted commutatively,
      order of node delivery / decryption is irrelevant */
    if $msg_{dest} \in Keychain_A$ **then**
      $msg_{text} \leftarrow Enc_{msg_{dest}}(msg_{text})$
    else
      $Node_i \leftarrow$ Random Node from $Keychain_A$
      $msg_{mid} \leftarrow msg_{mid} \cup Node_i$
      $msg_{text} \leftarrow Enc_{Node_i}(msg_{text})$
    end if
  end if
  Route to nearest node in msg_{mid}
end if

The Fragmenting method uses threshold encryption, which requires a larger keysize to be as secure as the commutative encryption used by the Chaining method. Also, the adversary can read the original message if enough of the fragments are sent through compromised nodes. Additionally, because a copy of the message must be sent with each fragment, the system's energy costs are considerably higher. Section 4 offers detailed information regarding these trade-offs.

A performance weakness in the scheme is that the midpoint is selected randomly from the available keychain rather than from among those related to either the origin or destination of the message. While this selection technique reduces performance by potentially sending messages from one end of the network to the other, it is necessary for security. Any metric that would allow nodes to identify themselves as high-value midpoints would allow adversaries to falsely identify themselves, resulting in a large number of messages routing through compromised nodes. For this reason, midpoints are chosen randomly, rather than using any awareness of the environment.

Algorithm 2 Fragment Encryption

Notation
$Node_{Origin}$ - Origin of msg
$plaintext$ - Original message to be sent to msg_{dest}
$Keychain_A$ - Set of Public Keys to which $Node_A$ has access
Nodes use $Threshold(k,n)$ algorithm
$EncTH_{encKeyset}(text)$ - Encrypts $text$ using Threshold encryption.
$DecTH_{decKeyset}(text)$ - Decrypts $text$ using Threshold encryption.
$decKeyset$ must contain at least k keys of original $encKeyset$
Key_i - Fragment i of a total of n fragments.
$msgFrag$ - Message Fragment. Contains...
$msgFrag_{text}$ - encrypted original message
$msgFrag_{key}$ - One key for encrypted message
$msgFrag_{dest}$ - Final destination of message
$msgFrag_{mid}$ - Midpoint of this fragment
$RcvdKeys$ - Keys received by $Node_{dest}$, initially $\emptyset$

Trigger - $Node_{Origin}$ wants to send message $plaintext$ to msg_{dest}
Generate random keys $\{Key_1, Key_2, ..., Key_n\}$
for $i = 1 \rightarrow n$ **do**
$Node_j \leftarrow$ Random Node from $Keychain_{Origin}$
Generate new message fragment $msgFrag$
/* In message, include encrypted form of original $plaintext$ and Key_i */
$msgFrag_{text} \leftarrow EncTH_{Key_{1..n}}(plainText)$
$msgFrag_{key} \leftarrow Enc_j(Key_i)$
$msgFrag_{dest} \leftarrow msg_{dest}$
$msgFrag_{mid} \leftarrow j$
Send $msgFrag$ to $Node_j$
end for

Trigger - $Node_A$ receives $msgFrag$, $A = msgFrag_{mid}$
if $msgFrag_{dest} \in KeyChain_A$ **then**
$msgFrag_{key} \leftarrow Enc_{dest}(Dec_A(msg_{key}))$
/* At this point, $Enc_{dest}(Dec_A(msg_{key})) = Enc_{dest}(Key_i)$ */
$msgFrag_{mid} \leftarrow \emptyset$
Send $msgFrag$ to $Node_{dest}$
else
$msgFrag_{mid} \leftarrow$ Random Node from $Keychain_A$
$msgFrag_{key} \leftarrow Enc_{mid}(Dec_A(msg_{key}))$
/* At this point, $Enc_{mid}(Dec_A(msg_{key})) = Enc_{mid}(Key_i)$ */
Send $msgFrag$ to $msgFrag_{mid}$
end if

Trigger - $Node_{dest}$ receives $msgFrag$, $msgFrag_{mid} = \emptyset$
$RcvdKeys \leftarrow RcvdKeys \cup Dec_{dest}(msgFrag_{key})$
if $|RcvdKeys| = k$ **then**
/* $Node_{dest}$ has enough keys to decrypt the message */
$plaintext \leftarrow DecTH_{RcvdKeys}(msgFrag_{text})$
Message has been delivered
end if

Consider that Alice again wants to send a message securely to Bob. Lacking the public key, Alice encrypts the message using threshold encryption. Three keys are generated, two of which must be possessed to read the message—$enc_{k_1,k_2,k_3}(Msg)$. The encrypted message is sent to each of the three untrusted midpoints, Chuck, David, and Eric, along with a copy of a single key encrypted with that army's public key—$enc_{Chuck}(k_1), enc_{k_1,k_2,k_3}(Msg)$ is sent to Chuck, and so forth. Each midpoint, upon receiving the message, should decrypt the key, then encrypt it with Bob's public key, and finally forward the message to Bob—$enc_{Bob}(k_1), enc_{k_1,k_2,k_3}(Msg)$. If Chuck has been compromised, he can access a single key that is insufficient for reading the message. This demonstrates the trade-off between security and reliability; by forcing the message to require a larger number of keys in order to be read (such as needing three out of four created keys, for example) the algorithm is more secure. A larger number of midpoints must be compromised by the adversary before it is able to read the original message. This increases security at the cost of preventing the destination from reading the message until it receives more of the keys, thus limiting both its successful delivery ratio and its time to delivery (Table 1).

4 Time and energy analysis

The cost of both implementing and maintaining a security infrastructure is a critical consideration. Thus, this section contains an analysis of both the expected time required to deliver a message and the cost of said delivery for both Chaining and Fragmenting. For comparison, an overview of the null security scheme and the key-request scheme (also referred to as the reflection scheme) also is provided.

Table 1 Variable reference chart

P_{key}	Probability that a node chosen at random has the Public Key for another node chosen at random
$E(T)$	Expected time for the routing algorithm to deliver a message from src to dest
$E_{req}(T)$	Expected time for Key Request scheme to securely deliver msg
$E_{chain}(T)$	Expected time for Chaining scheme to securely deliver msg
$E_{frag}(T)$	Expected time for Fragmenting scheme to securely deliver msg
$E(J)$	Expected energy cost for the routing algorithm to deliver a message from src to dest
$E_{req}(J)$	Expected energy cost for Key Request scheme to deliver msg
$E_{chain}(J)$	Expected energy cost for Chaining scheme to deliver msg
$E_{frag}(J)$	Expected energy cost for Fragmenting scheme to deliver msg

Although the total source-to-destination cost of a message is based on the routing algorithm rather than the security system, the costs will still increase when a message must be re-encrypted and forwarded multiple times.[1] The exception to this is null security, which will only encrypt a message if it already has the key immediately available. Otherwise, the message will be sent in plaintext. The energy cost to transmit this message is simply the cost required to forward a given message based on the routing protocol—$E(J)$. Similarly, the time required to deliver a message, T, is based solely on the routing method used—$E(T)$. Both the distribution and expected values of J and T are undefined because they can change based on the protocol used.

A major factor influencing the efficiency of a security schema is the probability that any given node will have the public key of any other node. Such techniques as caching and distribution can increase this probability but generally have their own security risks [13]. For the purpose of these calculations, the probability that a node will contain another node's key is assumed to be independent of neighboring nodes. Intelligent caching schemes, for instance, are implemented such that if a node does not have a key, nearby nodes are more likely to have them. Naive caching schemes tend to fill the local keyspace with the first keys available, which means that nearby nodes likely will not have the key. As the probability of codependence is a function of the distribution and mobility schemes, for calculation purposes they are assumed to be independent.

4.1 Key request analysis

The key-request scheme begins by determining whether or not the node has the key for the destination in question. If it does, the algorithm simply sends the encrypted message. Otherwise, it sends a key request to the destination, along with the public key. Then, the destination node sends an encrypted, symmetric key back to the source, where the original message is encrypted and sent. The expected energy cost and required transmission time therefore are based on the probability that the source already has the destination's key, represented as P_{key}.

$$E_{req}(J) = P_{key} \times E(J) + (1 - P_{key}) * 3 * E(J)$$
$$E_{req}(T) = P_{key} \times E(T) + (1 - P_{key}) * 3 * E(T)$$

For comparison purposes, consider a large-scale environment in which nodes are capable of carrying 30 % of the total number of public keys. In such a network, 30 % of the messages will be delivered in a single origin-to-destination transmission. The other 70 % will require three such transmissions. Messages thus have an expected delay and cost of 2.4 times that of a single transmission—$E_{req}(J) = .3 * E(J) + .7 * 3 * E(J) = 2.4E(J)$.

4.2 Chaining analysis

Similar to the key-request scheme, the Chaining method begins by determining whether or not the source has the destination's key. If it does not, it selects k midpoints, as described in Algorithm 1. This analysis is based on k being two nodes, although a system with better security will have a higher k. The expected hop count is based on how many nodes the message must visit before it is received by k nodes that have the destination key. The probability that the hop count is equal to the probability of the source having the key for the destination is $P(HC = 1) = P_{key}$. For the hop count to be 3, the first node will not have the key. Both midpoints will, however, and therefore they will not redirect the message at all. The probability of this is $P(HC = 3) = (1 - P_{key}) * P_{key} * P_{key}$. For the hop count to exceed three, either of the midpoints must be forced to redirect the message. The number of redirects is equal to the hop count minus 2, including the source's redirect to the two midpoints (Table 2).

The expected number of node-to-node messages can be derived when the probability of the various hop counts is known. The expected delivery cost and time can be calculated from the hop count.

$$E_{chain}(T) = E(T) \times \left(P_{key} + \sum_{i=3}^{\infty} i * (i-2) * (1 - P_{key})^{i-2} * P_{key}^2 \right)$$
$$= E(T) \times \left(P_{key} + \frac{2 - 5 * P_{key} + 3P_{key}^2 + P_{key}^3 - P_{key}^4}{(1 - P_{key})^2 * P_{key}} \right)$$
$$E_{chain}(J) = E(J) * \left(P_{key} + \frac{2 - 5 * P_{key} + 3P_{key}^2 + P_{key}^3 - P_{key}^4}{(1 - P_{key})^2 * P_{key}} \right)$$

Because the algorithm is based on a single message being forwarded through numerous midpoints, both the expected delivery time and energy cost are based directly on the hop count. For purposes of comparison, when an individual node can carry 30 % of the public keys in the network, the average hop count is $5\frac{2}{3}$.

4.3 Fragmented analysis

Fragmented encryption is the first algorithm discussed in which the delivery time and the energy consumed are not

[1] The costs of encrypting the message are negligible compared to the transmission costs. During experiments with Mica2 nodes, for example, encrypting a 1 kB message required 12.96 μJ. Transmitting the message required 1.5 mJ.

Table 2 Chaining algorithm events

Event	Expected delivery time	Probability
Src node has Key_{dest}	$E(T)$	P_{key}
Src does not have Key_{dest}, both chosen links have Key_{dest}	$3E(T)$	$\overline{P_{key}}P_{key}^2$
Src does not have Key_{dest}	$4E(T)$	$\overline{P_{key}} * \overline{P_{key}} * P_{key} * P_{key} +$
One link must redirect once		$\overline{P_{key}} * P_{key}\overline{P_{key}} * P_{key}$
		$= 2\overline{P_{key}}^2 * P_{key}^2$
Links must redirect j times	$(j+3)E(T)$	$\overline{P_{dest}}^{j+1}P_{dest}^2$

directly proportional. As in the Chaining algorithm, the message routes through a set number of midpoints and continues routing until all fragments are received. For this reason, the energy cost is nearly identical to Chaining; only the number of fragments is different. The following equations assume that three fragments are sent to the destination, two of which are needed to decrypt the message. In order to send only the message through a single hop, the source node must have the destination key, so $P(HC = 1) = P_{key}$; otherwise, there will be at least six transmissions—the source will send the message to each of the three midpoints, and each of those three will send it to the destination if all three have the key—$P(HC = 6) = (1 - P_{key}) * P_{key}^3$. If a single midpoint must redirect, the hop count is 7, and the probability of all three midpoints redirecting is $P(HC = 7) = 3(1 - P_{key})^2 P_{key}^3$.

$$
\begin{aligned}
P(HC = 1) &= P_{key} \\
P(HC = 6) &= (1 - P_{key}) * P_{key}^3 \\
P(HC = 7) &= 3(1 - P_{key})^2 * P_{key}^3 \\
P(HC = i) &= \frac{(i-4)(i-5)}{2}(1 - P_{key})^{i-5} * P_{key}^3
\end{aligned}
$$

This allows us to track the total number of transmissions, which in turn is used to calculate the total energy consumed per message.

$$
\begin{aligned}
E_{frag}(J) &= E(J) \\
&\quad \times \left(P_{key} + \sum_{i=6}^{\infty} i * \frac{(i-4)(i-5)}{2}(1 - P_{key})^{i-5} * P_{key}^3 \right) \\
&= E(J)\left(\frac{3}{P_{key}} - 2 * P_{key} \right)
\end{aligned}
$$

Using the previous 30 % example, this method will require each message to be transmitted an expected 9.1 times before all fragments are delivered.

Because the Fragmenting scheme sends each message independently of the others, the total delivery time is actually much shorter. Considering the example in which three fragments are sent, two of which are needed to decrypt the message, the delay will be the time the second fragment takes to reach the destination. Each fragment has a minimum of two hops—one to reach the midpoint, and another to be forwarded to the destination. If the fragment must be redirected to find the destination key, another hop is added. This means that the probability of two fragments reaching the destination in two hops is the probability of all three midpoints immediately having the key, or two of the midpoints having the key and the third midpoint being greater.

$P(HC = 2) = P_{key} * P_{key} * P_{key} + P_{key} * P_{key} * (1 - P_{key})$. This can be expanded to show the fragment's hop-count probability.

$$
\begin{aligned}
P(HC = i) &= P_{key} \times (1 - P_{key})^{i-2} \\
P(HC > i) &= (1 - P_{key})^{i-1} \\
P(HC < i) &= 1 - P(HC = i) - P(HC > i) \\
&= 1 - P_{key} * (1 - P_{key})^{i-2} - (1 - P_{key})^{i-1}
\end{aligned}
$$

Calculating the hop count of the message is feasible when the individual fragment's hop count is known. A message will be delivered in i hops if one fragment is delivered in exactly i hops, one fragment is delivered in i or less hops, and the third is delivered in i or more hops (independent of order). There are four discrete possibilities: (1) all three fragments can be delivered in exactly i hops, (2) any one of the fragments can be delivered in less than i hops (because it does not matter which fragment is delivered, three combinations exist), (3) any one can be delivered in more than i hops, and (4) one can be delivered in less than i hops, while another is delivered in greater than i hops (likewise, this distribution can occur in six different ways). These possible delivery hop counts, shown in Eq. 1, can be used to derive the expected delivery time of the Fragmenting method, shown in Eq. 2.

$$
\begin{aligned}
P(HC_{msg} = i) &= P(HC = i)^3 \\
&\quad + 3 * P(HC < i) * P(HC = i)^2 \\
&\quad + 3 * P(HC = i)^2 * P(HC > i) \\
&\quad + 6 * P(HC < i) * P(HC = i) * P(HC > i)
\end{aligned} \tag{1}
$$

$$
E_{frag}(T) = E(T)\frac{P_{key}^4 - 2P_{key}^3 - 5P_{key}^2 + P_{key} - 5}{P_{key}^4 - 5P_{key}^3 + 9P_{key}^2 - 6P_{key}} \tag{2}
$$

To follow our original example, the Fragmenting security scheme, with each node carrying 30 % of the total number of keys in the system, will deliver a message in roughly 3.84 hops. This analysis confirms our earlier assertion that this scheme will deliver a message in much less time than the Chaining method, but with more energy consumption.

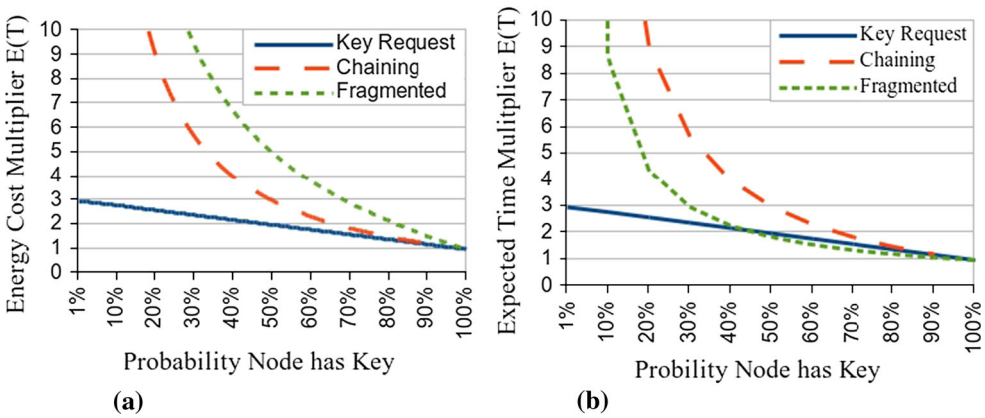


Fig. 1 Expected performance comparison. **a** Increased energy cost, **b** increase of time to deliver

Comparisons showing how the energy costs and delivery times vary with the P_{key} can be found in Fig. 1.

5 Security analysis

The purpose of MPE techniques is to provide security in unreliable networks. In environments with either unreliable, easily-compromised communications or nodes that have been compromised by an adversary, both Chaining and Fragmenting provide some measure of security, but at the cost of reduced performance and increased energy consumption. The conditions under which MPE fails must be determined to identify whether or not these techniques are beneficial despite their drawbacks.

Certain assumptions were made in evaluating the system. For example, the encryption method itself was considered secure. The network uses a node identification method that functions while the nodes are in direct contact. A certain percentage of the nodes, however, were assumed to have been compromised by an adversary. Another assumption was that messages were compromised if they were sent without encryption, even if they did not pass through a compromised node. However, in the simulations in Sect. 6 a link-layer encryption scheme was implemented, securing messages unless they were routed through compromised nodes.

5.1 Null encryption

Despite the title, the base security infrastructure will encrypt all of the messages it can. If a node does not have the public key, it will send the message unencrypted. Due to an adversary's ability to eavesdrop, a message can be read any time the source key does not have the destination key. Thus, a linear relationship exists between the number of keys a single node possesses and the number of compromised messages (Fig. 2).

5.2 Key-request scheme

A key-request scheme generally is broken by a man-in-the-middle attack, which is more difficult to implement in a MANET. Because nodes are mobile, messages tend not to follow the same message route continuously. The required position between the source and the destination is therefore more difficult to maintain. In practice, this means that a compromised node must lie somewhere on the path of the original key request (so the adversary can alter the public key), on the path of the reply containing the symmetric key

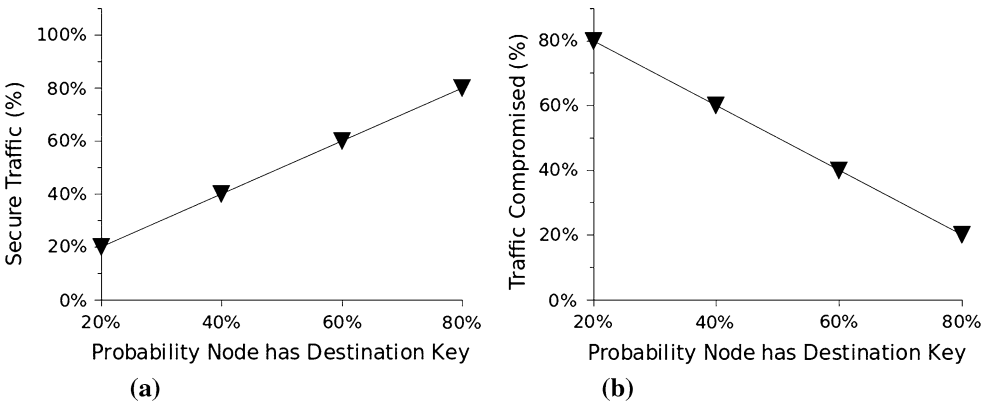


Fig. 2 Null/key request security analysis. **a** Message delivery rate, **b** compromised message rate

(to read the encrypted key), and on the path of the encrypted message. Because the compromised nodes work together, message security can be violated if one node on each path is compromised. However, if it has a dedicated communication channel, an eavesdropping adversary can reply to the source node with false key information immediately upon the message being sent. This attack renders the key-request scheme no more secure than employing no encryption scheme at all, although the attack is more difficult to implement (Fig. 2).

5.3 Chaining analysis

The major advantage of the Chaining method is that all messages are encrypted in one form or another. Eavesdropping attacks are thus rendered useless, so an adversary must rely on compromised nodes to intercept any traffic. To determine the security of this scheme, a Markov State process was used to simulate the current message status (Fig. 3). A message is sent securely from its origin to the destination only if the origin node already possessed the key. Otherwise, the node uses the Chaining algorithm, sending the message to the first link on the chain. The first link will either be compromised by the adversary, have the destination's public key, or redirect the message to another link (Fig. 4).

Three possibilities exist according to how many of the two links are compromised. If neither is compromised, the message is sent successfully to the destination. If only one is compromised, neither the destination nor the adversary can read the result. If both are compromised, the adversary can read the message. The probabilities of these possible scenarios are demonstrated in Fig. 5. If all midpoints are compromised, the message is compromised. If all are still secure, then the message is successfully delivered. If some are compromised and some not, however, the message is considered to have failed, but not been compromised—neither the adversary nor the final destination can read the message. This is considered acceptable in many scenarios, since the origin can resend the message.

5.4 Fragmenting analysis

As with Chaining, the adversary's inability to eavesdrop on any traffic means that the focus must be on compromised nodes. When using a threshold encryption scheme, an adversary can sometimes read a message with only a portion of the traffic read. This algorithm has no middle ground when using a 2 of 3 threshold encryption scheme. Eventually, either the adversary will read the message by intercepting two of the three fragments, or the message will be delivered successfully. The probabilities of these events are plotted in Fig. 6, based on the probability of any given node being compromised or possessing any particular encryption key. With a 2-of-3 threshold system, there is no possibility of message failure—either enough keys are compromised to allow the adversary to read the message, or the final destination will receive the message. These analyses demonstrate a tradeoff between performance and security, which we explore more in Sect. 6.

The results of this analysis illustrate that both Chaining and Fragmenting considerably reduce the percentage of messages compromised by the adversary. A small fraction of the message stream can be compromised even when only small portions of environments are compromised. Regardless, both of these algorithms reduce the expected percentage of messages compromised, doubling the number of messages securely transmitted in the base case in which a given node has 30 % of the keys available and 20 % of the nodes have been compromised, though at the cost of multiplying the total energy consumed by ten and tripling the transmission time (refer to Fig. 1). This increase in security is necessary in compromised environments, such as in vehicle to vehicle or UAV ad hoc network applications in military environment where energy limitation is not as restricted when compared with security requirements.

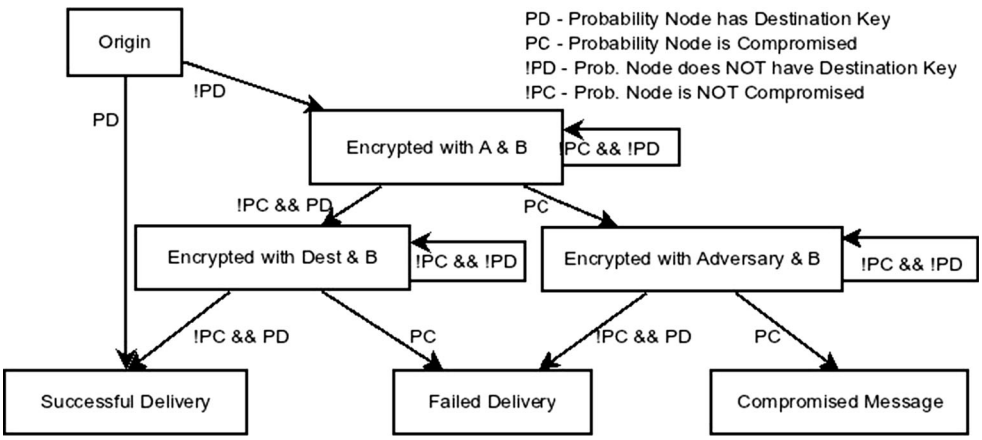
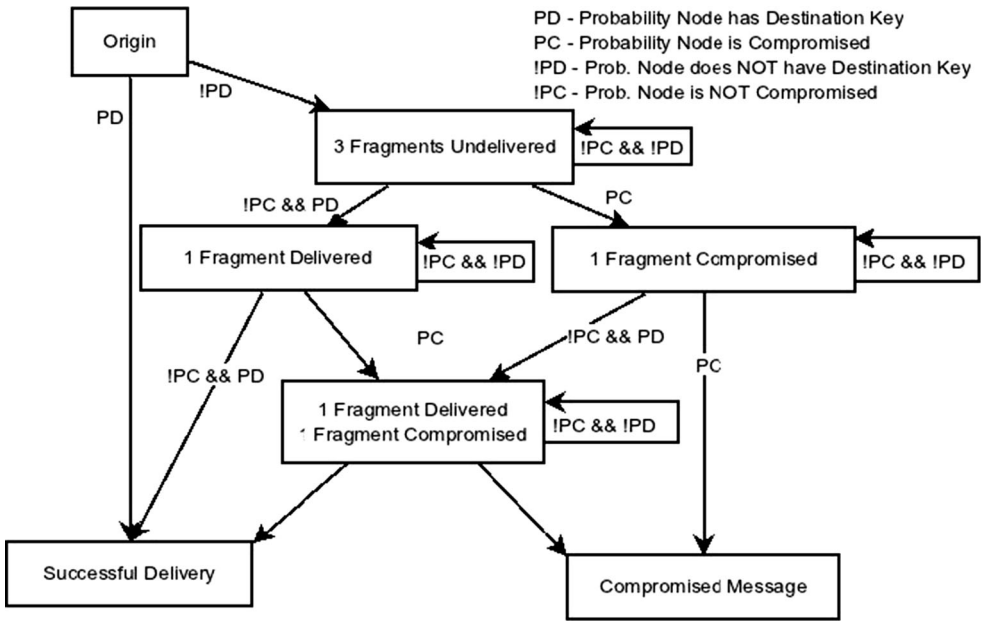


Fig. 3 2-Link chain process

Fig. 4 2 of 3 fragment process



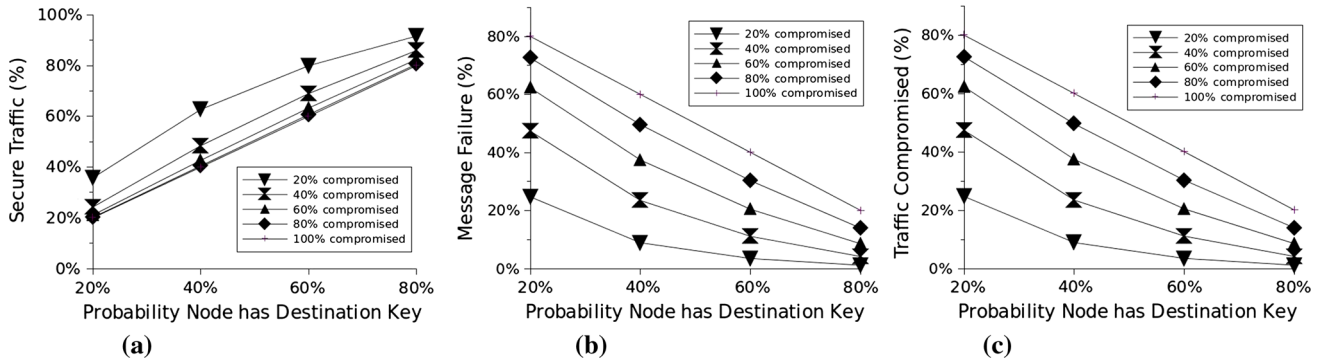


Fig. 5 Chaining security analysis. **a** Message delivery rate, **b** message failure rate, **c** compromised message rate

5.5 Other attacks

A Byzantine attack [27] is when nodes are compromised and then work in collusion to compromise security. While considered an advanced attack, other types of attacks also are available to the adversary.

Two attacks to consider in tandem are Black Hole [1, 28] and Wormhole attacks [10, 18]. The Black Hole attack is based on nodes identifying themselves falsely as being of high utility in order to direct all traffic through that node. Similarly, Wormhole attacks identify themselves as having high utility. In this case, however, the utility is at least partially correct because messages are routed with both high speed and reliability through a dedicated channel. When applied to a routing scheme, either attack can impact the number of messages delivered successfully. The fact that all message traffic is encrypted means that even directing all traffic through a particular node will not allow the adversary to read it. Incidentally, these attacks are the reason that the midpoint nodes are selected randomly. While delivery speed and reliability may be increased by assigning a utility value to a node (thus indicating its function as a midpoint), an adversary can use this function to route message traffic through compromised nodes. Thus, the current MPE model is based on midpoints being selected randomly.

Man-in-the-Middle (MitM) attacks are based on a midpoint intercepting a key exchange message and then altering that key to one that the adversary controls. When Alice sends her public key to Bob, midpoint Eve can replace Alice's key with her own. When Bob uses the key, Eve can easily read all of the messages that Bob sends to Alice. Most techniques for preventing the MitM attack are based on a trusted third party verifying the key, which does not work in a MANET, although other techniques are designed to function in such an environment [5, 12, 29].

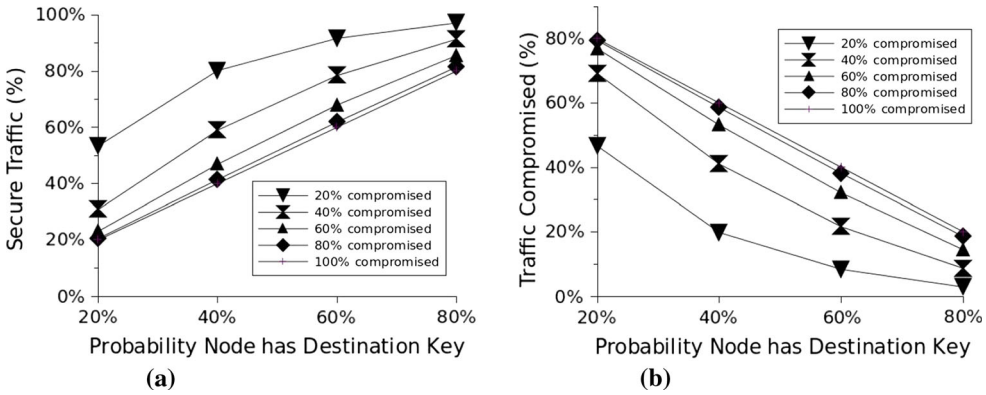


Fig. 6 Fragmented security analysis. **a** Message delivery rate, **b** compromised message rate

Both the Chaining and Fragmenting algorithms are designed to avoid this problem by only accepting keys in direct contact, thus preventing an intermediate node from replacing the keys used.

A final attack to consider is the Sybil attack [20], which is based on an adversary injecting simulated nodes into the network. Because the algorithm selects keys from existing nodes, a large number of false nodes increases the probability that a compromised node will be selected as a midpoint. A review of solutions to this attack is provided by Levine et al. [15].

In conclusion, both MPE algorithms render the majority of network attacks useless for the purpose of reading encrypted messages being sent across the network. While the Sybil attack remains a threat, solutions are available that can identify simulated nodes with high degrees of accuracy.

6 Performance evaluation

While all security systems have trade-offs regarding performance, the amount of performance delay should be compared to the increase in performance before implementation. For this reason, both the Chaining and Fragmenting algorithms were simulated using the Omnet++ Network Simulator. Performance and security metrics were gathered and compared to the null encryption scheme and the key-request scheme. When evaluating mobile networks, the mobility patterns followed by the devices should follow realistic patterns. The Small World in Motion mobility model [14] is a synthetic trace generator used to match real world datasets. Nodes visit locations, with the probability of each location determined by distance and popularity of the area. The performance evaluation was simulated using control parameters set to match the Cambridge'05 experiment. The increased message size of the encrypted packets was disregarded as it is both small enough to not impact large-scale communications systems, and there are methods to reduce such impact, such as stacking multiple cryptosystems.

6.1 Experimental setup

The algorithms were simulated in a 1,000m × 1,000m area, in which 100 mobile nodes were generated. Two separate simulations were performed. The first simulation set, designed to test the concept and optimize control variables, was implemented using simple nodes and the random waypoint mobility pattern. Each iteration varied the algorithm, the key space, and the buffer space available on each node. A message generation schedule was likewise generated in advance following a Gaussian distribution, with messages being sent every 30 s. For message routing, a PRoPHET routing algorithm was used [17] to route messages across the network. This routing algorithm was selected because it functions well in disconnected networks, especially where the nodes have a limited range. However, the random patterns followed by nodes reduced the efficiency of PRoPHET, which typically relies on long-term historical patterns to determine optimal paths.

The second simulation set was performed using more realistic parameters. The SWiM trace generator was implemented to simulate human-carried devices in a large-scale area. By measuring the social patterns, the routing algorithm PRoPHET was capable of identifying reliable message routes. This resulted in greater message efficiency, allowing more accurate measurements of the impact of the MPE security schemata (Table 3).

6.2 Comparisons

6.2.1 Simulation attack model

The potential attacker's capabilities are a primary factor when evaluating a security system. MANET uses radio communications, which allow attackers to eavesdrop on all unencrypted traffic. The simulation assumes that nodes use

Table 3 Random waypoint simulation parameters

System parameters	Settings
Length × width	1,000 m × 1,000 m
Number of nodes	100
TX power (tx)	0.25 mW
Signal-to-noise threshold (snr)	3.98×10^{-9}
Carrier frequency (cf)	2.4×10^{9} Hz
Transmission range between nodes	53 m
Message generation rate	30 s mean
Mobility pattern	Random waypoint
Movement speed	5 m/s
PRoPHET α	.1
PRoPHET β	.05
PRoPHET γ	.95

Table 4 SWiM simulation parameters

PRoPHET settings		
α	Direct contact impact setting	.007
β	Indirect contact impact setting	.008
γ	Probability decay rate	.9992
SWiM control settings		
Wait time exponent	Exponent of the power-law of the waiting time distribution	1.35
Wait time upper bound	Upper bound of the waiting time distribution	12 h
α	Distribution of home nodes	.75
Buckets per side	Bucket number per network area side (used for performance improvements)	14

link-layer security, meaning that all traffic is encrypted. Breaking message encryption is not considered feasible. Adversaries can, however, inject themselves into the network, disguising themselves as normal nodes. The simulation is based on the adversary compromising a certain percentage of the nodes. These nodes can read and modify any unencrypted messages going through them but are unable to break any encryption scheme used (Table 4).

6.2.2 *Random waypoint simulation results*

Based on the analysis described above, the Chaining method was expected to be the most secure, but at the cost of reduced performance, both in terms of the ratio of messages successfully delivered and the delivery time (Fig. 7).

For general comparison, all algorithms were submitted multiple times using a finite buffer from 10 to 1000 messages. The keychain also varied, holding from 10 to 100 % of the available keys in the network. The Chaining algorithm varied the number of required links from 2 to 5. Likewise, the Fragmenting algorithm varied both the number of fragments sent and the number needed for decryption from 2 to 5. Due to limited space, full results are shown for runs with buffers capable of carrying 400 messages, and 20 % of the node keys are displayed. The Chaining algorithm results for two links are shown, as are the Fragmenting results for sending three fragments, two of which are required to decrypt the message. The results (shown in Fig. 8) match the expected results; MPE algorithms offer much better security but at a greatly increased energy cost suitable for military applications among others. Note that MPE scheme is only to be initiated and used when keys are not accessible and thus, increased energy cost is not a deterrent factor (Fig. 9).

With so many ways of measuring the utility of a security system, it can be difficult to identify which it superior. To address this, the energy value for each secure message has been calculated. This value, show in Fig. 10, measure the energy consumed to securely send a message, assuming the system will resubmit the message if it fails to reach the destination or is compromised by the adversary. Based on the results shown, the Fragmenting system has the best cost-benefit tradeoff. It has a very high security, and it's higher delivery ratio, which serve to offset the higher cost of the fragments being sent redundantly. In contract, although Chaining sends messages more securely, the number of messages which are dropped and the significantly higher path length results in much higher energy consumption.

6.2.3 *Small world in motion simulation results*

One valuable trait of the MPE methods is that they are scalable. Security can be increased to suit compromised networks, though at a higher cost. The simulation (Fig. 11) shows that increasing the number of midpoints increases security at a faster rate than the performance degrades. Scaling the system up has drawbacks, including higher transmission costs and longer delivery times. In theory, however, the security can be improved indefinitely. Similar experiments were run for the Fragmenting method with various numbers of fragments. The results are complicated by the extra variable. The overview in Fig. 7 shows that performance varies in the same manner as in Chaining, but much faster. The performance starts off slightly worse than in Chaining and then drops quickly. The security, however, improves just as quickly. For comparison, when Chaining into 5 links, the delivery ratio is 16 %, with 1 % of the messages being compromised. A comparable delivery ratio can be found when Fragmenting to 3 messages, in which the compromised ratio is 2.6 %. In the same environments, by Fragmenting into 5 messages (all of which are needed),

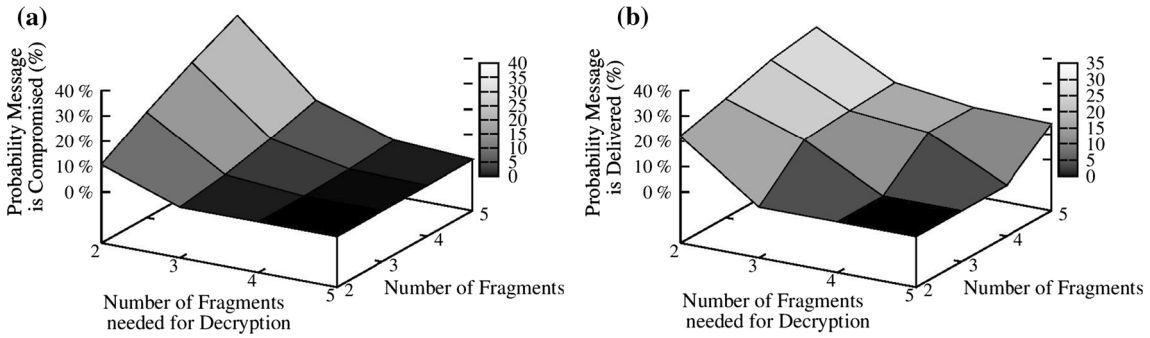


Fig. 7 Results of varying fragment count. **a** Comparison of message compromised rate, **b** comparison of message delivery ratio

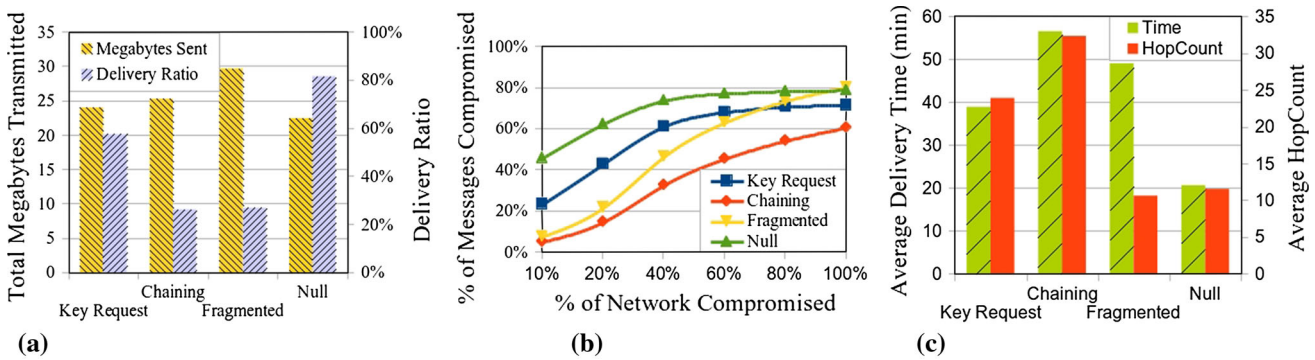


Fig. 8 Random waypoint simulation results, 400 message buffer, 20 % of keys. **a** Energy costs and delivery ratio, **b** security analysis, **c** message delivery time analysis

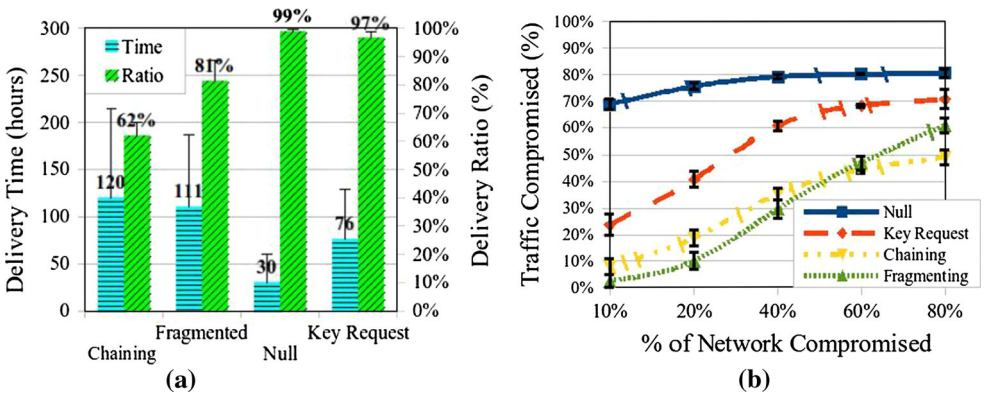


Fig. 9 SWiM experiment results, 400 message buffer, 20 % of keys. **a** Performance, **b** security

the delivery ratio is 14 %, but 0.1 % of the messages are compromised. Based on these results, we conclude that Fragmenting algorithm is more secure and better performing except in very unsecure networks.

The second set of simulations allowed more accurate measurements of the performance considerations in a realistic environment. Twenty sets of mobility patterns were generated, then submitted to each communication algorithm to determine the effect on performance, along with a confidence interval. The results, show in Fig. 9, indicate that changing the mobility pattern allowed much more reliability and improved security in the environments. Despite using the same routing algorithm, the more stable communication patterns improved the routing efficiency in the base case from 80 % to nearly 99 %. This improvement allows the MPE algorithms to function effectively, bringing their delivery ratio from 26 to 62 % for Chaining, and from 27 to 81 % for Fragmenting. This also increased the average delivery time, since messages identified slower more reliable routes to the destination. Before, nodes deliver messages only when the encountered midpoint can identify an immediate path, instead of incrementally

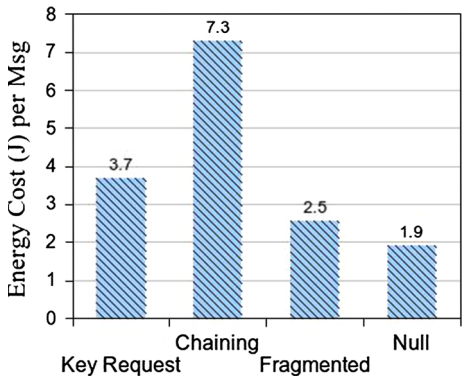


Fig. 10 RWP—energy cost per secure message transmission

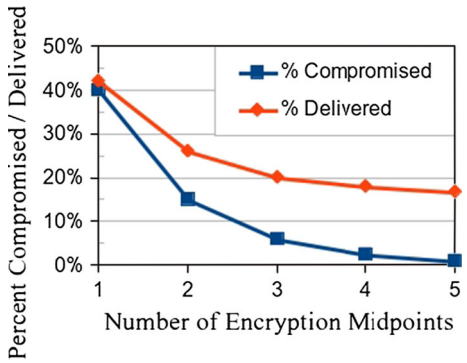


Fig. 11 Results of varying link count

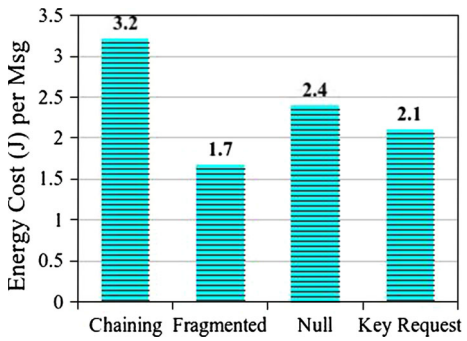


Fig. 12 SWIM—energy cost per secure message transmission

approaching the destination. The security of the MPE algorithm remains high, with the Fragmenting algorithm in particular performing well. Although Chaining is still more secure in a more hostile environment, until 50 % of the network is compromised Fragmenting is faster, more reliable, and more secure. Similar to the RWP experiments, the message/energy tradeoff shown in Fig. 12 indicates Fragmenting is worth the extra energy cost. More stable encounter patterns allow most schemes to perform better, since fewer messages are dropped, which helps Chaining and and the key-request scheme. Even so, Fragmented security outperforms the other schemes with a reliable message routing and improves security. Based on these results, the Fragmenting algorithm is more secure and better performing except in highly compromised and unreliable networks.

7 Conclusion

Both the experimental results and the analytical models indicate that MPE algorithms can improve the security in an otherwise unsecured network at the cost of increased network traffic and slower performance. While exact numbers vary based on both the network environment and the degree of security needed, results suggest that MPE algorithms serve as secure methods for routing without public keys. Thus, MPE algorithms are suitable in military environments where secure exchange of messages is a strict necessity and energy limitations are not as restricted. Additionally, both MPE schemes can be implemented without prior knowledge or trust schemes. They can be fine tuned to the degree of performance and security required by increasing the number of links or fragments. While the Chaining system boasts higher security and lower system costs, Fragmenting has a faster delivery time and can be modified more easily to suit a wider range of environments.

An ongoing issue to be addressed is the integration of MPE methods with a proper key management system. Because key management systems can use a variety of trust models to indicate the security of the individual key, a feasible approach is to merge the two, using key management when trust exceeds a certain threshold, and using an MPE method otherwise. In theory, this would achieve the best of both algorithms, making this a promising area for future development.

References

1. Al-Shurman, M., Yoo, S.-M., & Park, S. (2004). Black hole attack in mobile ad hoc networks. In *ACM Southeast regional conference* (pp. 96–97).
2. Bhutta, N., Ansa, G., Johnson, E., Ahmad, N., Alsiyabi, M., & Cruickshank, H. (2009). Security analysis for delay/disruption tolerant satellite and sensor networks. In *International workshop on satellite and space communications, 2009. IWSSC 2009* (pp. 385–389).
3. Cabaniss, R., Kumar, V., & Madria, S. (2012). Three point encryption (3PE): Secure communications in delay tolerant networks. In *SRDS*. IEEE (pp. 479–480).
4. Camtepe, S. A., & Yener, B. (2005). *Key distribution mechanisms for wireless sensor networks: A survey*. Technical report.
5. Capkun, S., Buttyn, L., & Hubaux, J.-P. (2002). Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, *2*, 52–64.

6. Dimitriou, T., & Michalas, A. (2014). Multi-party trust computation in decentralized environments in the presence of malicious adversaries. *Ad Hoc Networks*, *15*, 53–66 (Smart solutions for mobility supported distributed and embedded systems).
7. Dolev, S., Gilboa, N., & Kopeetsky, M. (2014). Efficient private multi-party computations of trust in the presence of curious and malicious users. *Journal of Trust Management*, *1*, 8. doi:10.1186/2196-064X-1-8.
8. El Defrawy, K., Solis, J., & Tsudik, G. (2009). Leveraging social contacts for message confidentiality in delay tolerant networks. In *2009 33rd annual IEEE international computer software and applications conference*. IEEE (pp. 271–279).
9. Golle, P., Jakobsson, M., Juels, A., & Syverson, P. (2002). Universal re-encryption for mixnets. In *RSA conference, cryptographer's track* (pp. 163–178). Springer.
10. Jain, M., & Kandwal, H. (2009). A survey on complex wormhole attack in wireless ad hoc networks. In *International conference on advances in computing, control, and telecommunication technologies (Washington, DC, USA, 2009), ACT'09* (pp. 555–558). IEEE Computer Society.
11. Katz, J., & Yung, M. (2001). *Threshold cryptosystems based on factoring*. Cryptology ePrint Archive, Report 2001/093.
12. Kong, J., Zerfos, P., Luo, H., Lu, S., & Zhang, L. (2001). Providing robust and ubiquitous security support for manet. In *Proceedings of IEEE international conference on network protocols (ICNP)*.
13. Kong, Y., Deng, J., & Tate, S. R. (2010). A distributed public key caching scheme in large wireless networks. In *Proceedings of IEEE global telecommunications conference—communication and information system security (GLOBECOM'10)*. Miami, FL, USA, December 6–10 2010.
14. Kosta, S., Mei, A., & Stefa, J. (2010). Small world in motion (SWIM): Modeling communities in ad-hoc mobile networking. In *Proceedings of the seventh annual IEEE communications society conference on sensor, mesh and ad hoc communications and networks, SECON 2010* (pp. 10–18). June 21–25, 2010, Boston, Massachusetts, USA.
15. Levine, B. N., Shields, C., & Margolin, N. B. (2006). *A survey of solutions to the sybil attack*. Technical report 2006–052, University of Massachusetts Amherst, Amherst, MA, October 2006.
16. Lewand, R. E. (2000). *Cryptological mathematics (classroom resource materials)*. Washington, DC: The Mathematical Association of America.
17. Lindgren, A., Doria, A., & Schelén, O. (2003). Probabilistic routing in intermittently connected networks. *SIGMOBILE Mobile Computing and Communications Review*, *7*, 19–20.
18. Madria, S. K., & Yin, J. (2009). Serwa: A secure routing protocol against wormhole attacks in sensor networks. *Ad Hoc Networks*, *7*(6), 1051–1063.
19. Menezes, A., & Ustaoglu, B. (2006). On the importance of public-key validation in the mqv and hmqv key agreement protocols. In *Proceedings of the 7th international conference on cryptology in India (Berlin, Heidelberg, 2006), INDOCRYPT'06* (pp. 133–147). Springer.
20. Newsome, J., Shi, E., Song, D., & Perrig, A. (2004). The sybil attack in sensor networks: Analysis defenses. In *Third international symposium on information processing in sensor networks, 2004. IPSN 2004* (pp. 259–268).
21. Patra, R., Surana, S., & Nedevschi, S. (2008). Hierarchical identity based cryptography for end-to-end security in dtns. In *4th international conference on intelligent computer communication and processing, 2008. ICCP 2008* (pp. 223–230).
22. Shamir, A. (1979). How to share a secret. *Communications of the ACM*, *22*(11), 612–613.
23. Stajano, F., & Anderson, R. (1999). The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proceedings of 7th International Workshop on Security Protocols*, Cambridge. Picture Notes in Computer Science (Vol. 1796, pp. 172–194). Berlin: Springer.
24. Stinson, D. R. (2005). *Cryptography: theory and practice, third edition (discrete mathematics and its applications)*. Boca Raton: Chapman & Hall/CRC.
25. Syverson, P. F., Reed, M. G., & Goldschlag, D. M. (1997). Private web browsing. *Journal of Computer Security*, *5*(3), 237–248.
26. Vakde, G., Bibikar, R., Le, Z., & Wright, M. (2011). Enpassant: Anonymous routing for disruption-tolerant networks with applications in assistive environments. *Security and Communication Networks*, *4*(11), 1243–1256.
27. Wu, B., Chen, J., Wu, J., & Cardei, M. (2007). A survey of attacks and countermeasures in mobile ad hoc networks. In Y. Xiao , X. S. Shen & D.-Z. Du (Eds.), Wireless network security, signals and communication technology (pp. 103–135). US: Springer.
28. Yin, J., & Madria, S. K. (2006). A hierarchical secure routing protocol against black hole attacks in sensor networks. In *SUTC (1)* (pp. 376–383).
29. Zhou, L., & Haas, Z. J. (1999). Securing ad hoc networks. *IEEE Network Magazine*, *13*, 24–30.

Roy Cabaniss earned his Doctorate of Philosophy in December 2012 from the Missouri University of Science and Technology. He works for LGS-Innovations. His areas of interests are in routing and security in Delay Tolerant Networks and Mobile Ad-Hoc Networks. He also served as a teaching assistant for the Introduction to C++ classes, both the lab and the lecture, and mentored summer workshops teaching wireless sensor systems to undergraduate students.

Vimal Kumar received a Ph.D. in computer science from the Missouri University of Science and Technology and a B.S. in computer science and engineering from Indraprastha University, India. He is currently an Assistant Professor at the University of West Florida. His research interests include the broad areas of sensor clouds, wireless sensor networks, and cloud computing, particularly security in these areas. He's a member of IEEE.

Sanjay Madria received his Ph.D. in Computer Science from Indian Institute of Technology, Delhi, India in 1995. He is a Professor, and Associate Chair for Research in the Department of Computer Science at the Missouri University of Science and Technology (formerly, University of Missouri-Rolla), USA and Site Director, NSF Industry/University Center on Net-centric System Software. Earlier he was Visiting Assistant Professor in the Department of Computer Science, Purdue University, West Lafayette, USA. He authored more than 200 journal and conference papers in the areas of mobile data management, security in sensor networks, and cloud computing. He served as a general co-chair of IEEE Mobile Data Management conference in 2010, and of IEEE Symposium on Reliability in Distributed Systems in 2012. He received UMR faculty excellence awards in 2007, 2009, 2011 and 2013, Japanese Society for Promotion of Science invitational fellowship in 2006, and Air Force Research Lab's visiting faculty fellowship from 2008 to 2012. He is IEEE Senior Member, IEEE Golden Core, served/serving as a speaker under IEEE and ACM Distinguished Visitor Program.