Secure Data Transmission in Mobile Ad Hoc Networks

Panagiotis Papadimitratos and Zygmunt J. Haas
School of Electrical and Computer Engineering
Wireless Networks Laboratory
Cornell University
Ithaca, NY 14853, U.S.A
{papadp, haas}@ece.cornell.edu
http://wnl.ece.cornell.edu

ABSTRACT

The vision of nomadic computing with its ubiquitous access has stimulated much interest in the Mobile Ad Hoc Networking (MANET) technology. However, its proliferation strongly depends on the availability of security provisions, among other factors. In the open, collaborative MANET environment practically any node can maliciously or selfishly disrupt and deny communication of other nodes. In this paper, we present and evaluate the Secure Message Transmission (SMT) protocol, which safeguards the data transmission against arbitrary malicious behavior of other nodes. SMT is a lightweight, yet very effective, protocol that can operate solely in an end-to-end manner. It exploits the redundancy of multi-path routing and adapts its operation to remain efficient and effective even in highly adverse environments. SMT is capable of delivering up to 250% more data messages than a protocol that does not secure the data transmission. Moreover, SMT outperforms an alternative single-path protocol, a secure data forwarding protocol we term Secure Single Path (SSP) protocol. SMT imposes up to 68% less routing overhead than SSP, delivers up to 22% more data packets and achieves end-to-end delays that are up to 94% lower than those of SSP. Thus, SMT is better suited to support QoS for real-time communications in the ad hoc networking environment. The security of data transmission is achieved without restrictive assumptions on the network nodes' trust and network membership, without the use of intrusion detection schemes, and at the expense of moderate multi-path transmission overhead only.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Network protocols, Security and protection; C.4 [**Performance of Systems**]: Fault tolerance

General Terms

Security, Reliability, Performance, Algorithms

Keywords

MANET security, Secure Message Transmission, Multi-path Routing, Secure Routing, Secure Routing Protocol

1. INTRODUCTION

The communication in mobile ad hoc networks comprises two phases, the route discovery and the data transmission. In an adverse environment, both phases are vulnerable to a variety of attacks. First, adversaries can disrupt the route discovery by impersonating the destination, by responding with stale or corrupted routing information, or by disseminating forged control traffic. This way, attackers can obstruct the propagation of legitimate route control traffic and adversely influence the topological knowledge of benign nodes. However, adversaries can also disrupt the data transmission phase and, thus, incur significant data loss by tampering with, fraudulently redirecting, or even dropping data traffic or injecting forged data packets.

To provide comprehensive security, both phases of MANET communication must be safeguarded. It is noteworthy that secure routing protocols, which ensure the correctness of the discovered topology information, cannot by themselves ensure the secure and undisrupted delivery of transmitted data. This is so, since adversaries could abide with the route discovery and be placed on utilized routes. But then, they could tamper with the in-transit data in an arbitrary manner and degrade the network operation.

Upper layer mechanisms, such as reliable transport protocols, or mechanisms currently assumed by the MANET routing protocols, such as reliable data link or acknowledged routing, cannot cope with malicious disruptions of the data transmission. In fact, the communicating nodes may be easily deceived for relatively long periods of time, thinking that the data flow is undisrupted, while no actual communication takes place.

One way to counter security attacks would be to cryptographically protect and authenticate all control and data traffic. But to accomplish this, nodes would have to have the means to establish the necessary trust relationships with each and every peer they are transiently associated with, including nodes that just forward their data. Even if this were feasible, such cryptographic protection cannot be effective against denial of service attacks, with adversaries simply discarding data packets.

Portions of this article have been reprinted from "Ad Hoc Networks" journal, vol. 1, no. 1, Papadimitratos and Haas, "Secure Message Transmission in Mobile Ad Hoc Networks," pp. 193-209, Copyright 2003, with permission from Elsevier.

This work was supported in part by the National Science Foundation under grant numbers ANI-0081357 and by the DoD Multidisciplinary University Research Initiative (MURI) program administered by the Office of Naval Research under contract number N00014-00-1-0564.

WiSe'03, September 19, 2003, San Diego, California, USA.

To secure the data transmission phase, we propose and evaluate the *Secure Message Transmission (SMT)* protocol, an end-to-end secure data forwarding protocol tailored to the MANET communication requirements. The SMT protocol safeguards pair-wise communication across an unknown frequently changing network, possibly in the presence of adversaries that may exhibit arbitrary behavior. It combines four elements: end-to-end secure and robust feedback mechanism, dispersion of the transmitted data, simultaneous usage of multiple paths, and adaptation to the network changing conditions. SMT detects and tolerates compromised transmissions, while adapting its operation to provide secure data forwarding with low delays.

We underline that the goal of SMT is not to securely discover routes in the network – the security of this phase should be achieved by one of the protocols proposed in the literature [1,2,5,23-25].[1] The goal of SMT is to ensure secure data forwarding, after the discovery of routes between the source and the destination has been already performed. In other words, SMT assumes that there is a protocol that discovers routes in the ad hoc network, although such discovered routes may not be free of malicious nodes.[2] Then, the goal of SMT is to ensure routing over such routes, despite of the presence of such adversaries.

In addition to SMT, we present and evaluate here the *Secure Single Path (SSP)* protocol, an end-to-end secure data forwarding protocol that utilizes a single route. Unlike SMT, SSP does not incur multi-path transmission overhead. Thus, it does not require that the underlying routing protocol discover multiple routes either. As a result, SSP imposes less routing overhead per discovery than SMT. Overall, we examine SSP and compare it to SMT as an alternative, lower cost, more flexible protocol to secure the data-forwarding phase.

Our results show that SMT outperforms SSP consistently over a wide range of experiments. The advantages of SMT over SSP become more pronounced in highly adverse environments: SMT delivers up to 22% more data packets than SSP, and achieves up to 94% lower delays than SSP. It is also very interesting that SMT imposes up to 68% less routing overhead than SSP, although overhead was expected to be lower for SSP. In contrast, SSP provides only up to 48% lower transmission overhead than SMT. We especially emphasize the low-delay characteristic of SMT, as we believe that one of the main applications of SMT is in support of QoS for real-time traffic.[3]

In the rest of the paper, we first provide an overview of the SMT protocol and present its operation. Then, in Section 4, we outline the operation of SSP and evaluate the performance of the two protocols. Related work is discussed next, followed by a discussion and description of future work in Section 6, before our conclusion.

[1] Nevertheless, care should be taken in such a selection, as some protocols can support single-path forwarding and others multiple route discovery.

[2] Clearly, an adversary could hide its malicious behavior for a long period of time and strike at the least expected time – it would be impossible to discover such an adversary prior to its attack.

[3] SMT, due to its operation over multiple paths, allows elimination of retransmissions of packets that were lost due to adversarial nodes.

2. OVERVIEW OF SMT

SMT requires a *security association (SA)* only between the two end communicating nodes – the source and the destination. Since a pair of nodes chooses to employ a secure communication scheme, their ability to authenticate each other is indispensable. The trust relationship can be instantiated, for example, by the knowledge of the public key of the other communicating end.[4] However, none of the end nodes needs to be securely associated with any of the remaining network nodes. As a result, SMT does not require cryptographic operations at these intermediate nodes.

With SMT, at any particular time, the two communicating end nodes make use of a set of diverse, preferably node-disjoint paths that are deemed valid at that time. We refer to such a set of paths as the *Active Path Set (APS)*. The source first invokes the underlying route discovery protocol, updates its network topology view, and then determines the initial APS for communication with the specific destination.

With a set of routes at hand, the source disperses each outgoing message into a number of pieces. At the source, the dispersion, based on the algorithm in [3], introduces redundancy and encodes the outgoing messages, as described in Section 3.2. At the destination, a dispersed message is successfully reconstructed, provided that sufficiently many pieces are received. In other words, the message dispersion ensures successful reception even if a fraction of the message pieces is lost or corrupted, either due to the existence of malicious nodes, or due to the unavailability of routes (e.g., breakage of a route as a result of nodes' mobility).

Each dispersed piece is transmitted across a different route and carries a *Message Authentication Code (MAC)* [4], so that the destination can verify its integrity and the authenticity of its origin. The destination validates the incoming pieces and acknowledges the successfully received ones through a feedback back to the source.

The feedback mechanism is also secure and fault tolerant: it is cryptographically protected and dispersed as well. This way, the source receives authentic feedback that explicitly specifies the pieces that were received by the destination. A successfully received piece implies that the corresponding route is operational,[5] while a failure is a strong indication that the route is either broken or compromised.

While transmitting across the APS, the source updates the rating of the APS paths. For each successful or failed piece, the rating of the corresponding path is increased or decreased, respectively, as we explain in Section 3.3. A path is discarded once it is deemed failed and a precaution is taken not to use the same path, if it is discovered again within some time after it has been discarded. While continuously assessing the quality of the utilized paths, the protocol adapts its operation according to the feedback it receives from the trusted destination. Based on its interaction with the network, the protocol adjusts its configuration to remain effective in highly adverse environments and efficient in relatively benign conditions.

[4] The two nodes can negotiate a shared secret key, e.g., via the Elliptic Curve Diffie-Hellman algorithm [16,18] and then, using the SA, verify that the principal that participated in the exchange was indeed the trusted node. For the rest of the discussion, we assume the existence of a shared secret key $K_{S,T}$.

[5] Although this does not ensure that the path is free of malicious nodes.

If a sufficient number of pieces are received at the destination, the destination proceeds to reconstruct the message. Otherwise, if a dispersed message cannot be reconstructed at the destination, it awaits the missing packets that are retransmitted by the source. The number of re-transmissions is limited to $Retry_{max}$ per serviced message.

An illustrative example of a single message transmission is shown in Fig. 1. The sender disperses the encoded message into four packets, so that any three out of the four packets are sufficient for successful reconstruction of the original message. The four packets are routed over four disjoint paths and two of them arrive intact at the receiver. The remaining two packets are compromised by malicious nodes lying on the corresponding paths; for example, one packet is dropped, and one (dashed arrow) is modified.

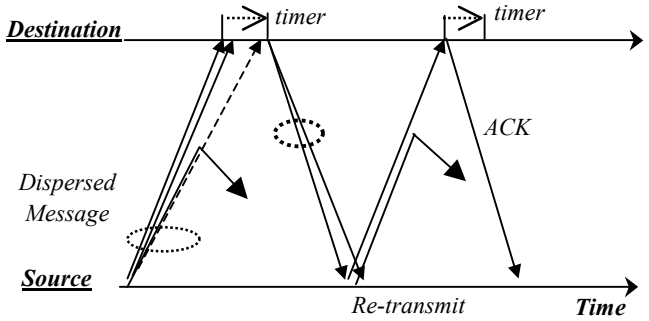


Figure 1. Simple example of the SMT protocol.

The receiver extracts the information from the first incoming validated packet and waits for subsequent packets, while setting a reception timer. When the fourth packet arrives, the cryptographic integrity check reveals the data tampering and the packet is rejected. At the expiration of the timer, the receiver generates an acknowledgement reporting the two successfully received packets and feedbacks the acknowledgment across the two operational paths.

It is sufficient for the sender to receive and cryptographically validate only one acknowledgement, ignoring duplicates. The two failing paths are discarded and the two missing pieces are then retransmitted over other paths; one of the two packets is now lost, for example, because of intermittent malicious behavior, or a benign path breakage. The receiver acknowledges the successful reception immediately, before the timer expiration, since an adequate number of packets (3 out of 4) have been received. Note that after transmission of the first packet, the sender sets a retransmission timer, so that total loss of all the message pieces or of all the acknowledgments can be detected.

3. DETAILS OF SMT OPERATION

3.1 Determination of the APS

SMT can operate with any underlying routing protocol,[6] although the use of a secure protocol is essential to reap the benefits of SMT. Otherwise, adversaries could disable communication by continuously providing false routing information. SMT is independent of the route discovery process – for example, it can operate in conjunction with a reactive or a proactive protocol. However, the knowledge of the actual nodal connectivity and the use of source routing result in two advantages. First, it is possible for the sender to implement an arbitrary path selection algorithm in order to increase the reliability of the data transmission. For example, the path selection algorithm could incorporate subjective criteria, such as nodes to be explicitly included or excluded from the APS. Second, no discretion on route decisions is left to intermediate nodes, in order to enhance the robustness of the protocol. This way, the communicating end nodes can explicitly correlate the failed or successful transmissions with the corresponding routes. As a result, non-operational and possibly compromised routes are unambiguously detected at the source node, so that newly determined routes can be entirely different from previously utilized and discarded routes. For the rest of the paper, we assume that a secure routing protocol provides a number of routes to *SMT*, every time the route discovery protocol is executed. The source constructs an APS of k node-disjoint paths, depending on the actual node connectivity of its topology view.

3.2 Message Dispersion and Transmission

The information dispersal scheme is based on Rabin's algorithm [3], which acts in essence as an erasure code: it adds limited redundancy to the data to allow recovery from a number of faults. The message and the redundancy are divided into a number of pieces, so that even a partial reception can lead to the successful re-construction of the message at the receiver. In principle, the encoding (and dispersion) allows the reconstruction of the original message with successful reception of any M out of N transmitted pieces. The ratio $r = N/M$ is termed the *redundancy factor*.

Messages, i.e., raw data, can be viewed as a stream of integers, or m-bit characters, so that each integer is in the $[0...2^m-1]$ range. It suffices to select a prime number $p > 2^m-1$, so that all encoding and decoding operations are performed in a finite field *mod* p.[7] Initially, N random M-vectors, organized as rows $\{a_i\}$ of matrix A, are selected, with any M of them linearly independent. These a_i vectors can be constructed by selecting N different elements u_i of the finite field and set $a_i=[1,u_i,...,u_i^{M-1}]$, $1 \le i \le N$, and $N<p$. The vectors of matrix A should be selected from a pre-computed set used by both ends, which we assume are agreed upon as part of the *SA* establishment process.

The encoding of a message first segments the original message of length FS into L sequences of characters, each of length M, with padding if necessary. The segments of the original message are denoted by $s_1, s_2, ..., s_L$ and they are arranged as columns of an M-by-L array B. Then, each piece w_i of the dispersed message is created as a character sequence of length L: to do so, the original message segments are multiplied by the corresponding random vector a_i, and the resultant piece is $w_i=[a_is_1, a_is_2, ..., a_is_L]$.

Upon reception of any M pieces, the original message can be reconstructed. Let $v_1, v_2, ..., v_M$ denote the M pieces used for reconstruction, which are in fact a subset of the N transmitted

[6] As long as the routing protocol is capable of discovering multiple routes.

[7] The operations can be performed in finite fields of the form $GF(2^m)$, to avoid the use of excessive bits per represented character. For example, if *8-bit* characters are used, the use of $p=257$ imposes an excess of one bit per character, while $GF(2^8)$ suffices, without the excess [3,17].

pieces, w_i.[8] Each one of the v_i pieces corresponds to one of the a_i vectors, which are, by definition, linearly independent. The matrix $[A']_{M\times M}$ comprising these vectors is thus invertible. To reconstruct the original message, it suffices to multiply each of the v_i pieces by the inverse of A'. If v_i are the rows of a *M-by-L* array, W, the original message reconstruction can be written as $B = [A']^{-1} \times W'_{M\times L}$.

(i). $A = \begin{pmatrix} 1\ 45\ dc \\ 1\ 7d\ f5 \\ 1\ b0\ 7a \\ 1\ 5b\ 95 \end{pmatrix}$

(ii). $B =$

```
( c  e2 9f 5  c0 6  4c 62 c7 80 6f 8e 87 8c 7a 2c 31 4e 5a ee a5 5
70 c6 eb 99 b2 21 30 23 15 86 5d 85 72 2e 7f 42 b  3f cc 9b 89 c5
8d 46 ba 7f f9 61 19 49 a9 93 d1 f1 75 3  74 62 bf 77 f4 f4 4e 8  )
```

(iii). $W =$

```
(e7 27 5d 11 81 4c 13 8e 6d 34 33 cb a0 2c df 8  9d 2  53 95 8a ca
54 24 b1 f9 60 d5 31 7b 1e 1d 1f 23 7e 49 2d 57 58 2d 75 1a f  b0
e8 a2 d5 ca 62 5f 7e 4a ff 87 b1 a0 c1 ef e5 ae 5a 44 b9 ee d7 15
5f 17 a2 af d2 9e d6 c7 6b 87 e2 aa a3 a4 3  17 9d d6 7f cd b1 b6)
```

(iv). $W' =$

```
(e7 27 5d 11 81 4c 13 8e 6d 34 33 cb a0 2c df 8  9d 2  53 95 8a ca
54 24 b1 f9 60 d5 31 7b 1e 1d 1f 23 7e 49 2d 57 58 2d 75 1a f  b0
e8 a2 d5 ca 62 5f 7e 4a ff 87 b1 a0 c1 ef e5 ae 5a 44 b9 ee d7 15)
```

(v). $A' = \begin{pmatrix} 1\ 45\ dc \\ 1\ 7d\ f5 \\ 1\ b0\ 7a \end{pmatrix}$

(vi). $B = [A']^{-1} \times W' =$

```
( c  e2 9f 5  c0 6  4c 62 c7 80 6f 8e 87 8c 7a 2c 31 4e 5a ee a5 5
70 c6 eb 99 b2 21 30 23 15 86 5d 85 72 2e 7f 42 b  3f cc 9b 89 c5
8d 46 ba 7f f9 61 19 49 a9 93 d1 f1 75 3  74 62 bf 77 f4 f4 4e 8  )
```

Figure 2. Example of the IDA operation with *r=N/M=4/3*. (i) Matrix *A* holds *N=4* random vectors, (ii) a message of *FS = 64* bytes is segmented (after padding) into *L = 22* segments, which are the columns of matrix *B*, (iii) the dispersed message, with its pieces as rows of matrix *W*, (iv) the partially received message (*3* out of *4* pieces), along the rows of *W'*, (v) the random vectors that correspond to the *M* received pieces, and (vi) the reconstructed message, identical to the original *B*. (All data values are *8-bit* integers, shown in their hexadecimal representation.)

Fig. 2 provides an illustrative example of the IDA operation, continuing the example in Fig. 1. *N=4* pieces are sent and *M=3* pieces are received and used in the message reconstruction at the receiver, i.e., *r = 4/3*. Raw data are treated as bytes and take values between *0* and *255*. The encoding and decoding operations are performed in the $GF(2^8)$ finite field. Matrix *A* is created based on the (randomly) selected $u_i = \{69, 125, 176, 91\}$, and it is shown in Fig. 2(i). The message has size *FS=64* bytes and it is padded with $PD = M \cdot \lceil FS/M \rceil - FS$ bytes. The message is segmented into the *L = (FS+PD)/M* columns of *B* (Fig. 2(ii)). The encoded message *W* is shown in Fig. 2(iii), with each row of the array being one piece to be dispersed through the network. Now, for instance, let the w_4 piece be the one that is never received by the destination. The message pieces available to the receiver are the rows of matrix *W'* shown on Fig. 2(iv). Matrix *A'* holds the $\{a_i\}$ vectors that correspond to the received pieces, and the reconstructed message, shown on Fig. 2(vi), is identical to the transmitted one.

3.3 APS Adaptation

As the source transmits the dispersed messages across the *APS*, it updates the ratings of the utilized paths based on the feedback (or its absence) provided by the destination. Each path is associated with two ratings: a short-term and a long-term rating. The short-term rating, r_s, is decreased by a constant α each time a failed transmission is reported, and it is increased by a constant β for each successful reception. The long-term rating, r_l, is a fraction of successfully received (and in fact, acknowledged) pieces over the total number of pieces transmitted across the route. If either r_s or r_l or both drop below a threshold value, rsthr and r_l^{thr} respectively, the corresponding path is discarded. Both thresholds and constants are protocol selectable parameters.

The r_s rating takes values in the interval $I = [r_s^{thr}, r_s^{max}]$, with $r_s^{thr} \geq 0$, r_s^{max} the maximum value for the path rating, and $r_s(0)$ its initial rating, assigned when a path is first added to the APS.[9] The constants α and β take values in the $(0, r_s^{max}]$ interval. After the *i-th* transmission across a path that is not deemed failed yet, its rating is updated:

$$r_s(i) = \begin{cases} \max\{r_s(i-1) - \alpha, r_s^{thr}\}, & \text{if a piece is lost} \\ \min\{r_s(i-1) + \beta, r_s^{\max}\}, & \text{if a piece is received} \end{cases} \quad (1)$$

If *i* transmissions across a path include *s* successfully received (thus acknowledged) pieces and *l* lost ones, then *i = s + l*, with *s*, *l* integers. If $r_s(i)$ has already reached the maximum value, then, additional successive acknowledged pieces do not increase the rating any further. If s_0 denotes the number of such successful receptions, and s_1 denotes the number of successful receptions while the path rating is below rsmax, then $s=s_0+s_1$. Thus, the rating of the path can be written as $r_s(i) = r_s(0) + \beta s_1 - \alpha l$. For any route that is not deemed failed yet, $r_s(i) \geq r_s^{thr}$. Then, $s_1\beta - l\alpha \geq r_s^{thr} - r_s(0)$, (for s_1, l integers not simultaneously zero). If we set $d = r_s(0) - r_s^{thr} \geq 0$, we can re-write the previous inequality as:

$$\beta s_1 - \alpha l + d \geq 0 \quad (2)$$

The rating mechanism should guarantee that a non-operational route is promptly discarded, independently of its prior history. In other words, the detection of route failures should be fast even for routes that were fully operational for a long period of time and their rating reached its maximum allowed value, r_s^{max}. In that case, the failed route would be discarded after at most $f = \lceil (r_s^{max} - r_s^{thr})/\alpha \rceil$ successive failed transmissions. The value of *f* can be regulated by selecting, for example, an appropriate value for the constant α. If *f* is low (e.g., 1), a transient failure will result in discarding an operational path, while a high *f* may allow repeated transmissions over a broken path and thus overhead before determining the path breakage.

Nevertheless, an adversary lying on a path may select an arbitrary attack pattern to disrupt the transmissions without letting $r_s(i)$ to drop below r_s^{thr}. This way, the attacker can retain its ability to degrade the network operation, trying to maximize the number of dropped data packets, while the route is still considered

[8] In case more than *M* pieces are received, the first *M* could be used for the reconstruction of the message, for efficiency reasons. Another option would be to use the *M* most credible pieces, if soft-detection decoding is used.

[9] The initial value is set to $r_s(0) = \delta \cdot (r_s^{\max} - r_s^{thr})$, with $0<\delta<1$.

operational. Intuitively, the attacker would be most effective if it never allows the reception of data pieces when the path rating is equal to r_s^{max} (i.e., $s_0=0$).

In order to determine precisely the effectiveness of the path rating mechanism, we define the *bandwidth loss over a path*, *BWL*, as the fraction of packets that an adversary can discard or corrupt without the route determined to be non-operational (i.e., Eq. (2) holds for the route). Based on the previous discussion, the *BWL* for i transmissions (s successful and l failed ones) across a single path is

$$BWL = \frac{l}{i} = \frac{l}{s+l} \tag{3}$$

For any number of successfully received packets, $s \leq i$, that the attacker allowed to reach the destination, the attacker can select any l packets to drop without being detected. Clearly, $l \leq i - s$ and from Eq. (2'), (with $\alpha \neq 0, \beta \neq 0$) l will be

$$l \leq \frac{\beta}{\alpha}\left(s + \frac{d}{\beta}\right) \tag{4}$$

Thus, the maximum number of dropped packets is

$$l^* = \frac{\beta}{\alpha}\left(s + \frac{d}{\beta}\right) \tag{4'}$$

The *BWL* would be maximized when l is maximized ($l = l^*$). As the number of transmissions increases and, thus, s increases, we get from Eq. (4') and Eq. (3):

$$BWL \leq BWL^* = \lim_{s \to +\infty} \frac{l^*}{s + l^*} = \frac{\beta}{\alpha + \beta} \tag{5}$$

The bound for data loss provided in Eq. (5) is independent of the attack pattern. Thus, a judicious selection of α and β can reduce the impact of an intelligent adversary that stays undetected. Clearly, it is necessary for α not to be zero ($\alpha > 0$); otherwise, the attacker would have full control over a path ($BWL^* = 1$). Furthermore, it must hold that $\alpha > \beta$, in order to keep $BWL^* < 0.5$; in fact, the smaller β is compared to α, the lower BWL^* will be.[10]

Depending on the selection of values for α and β, the loss of data could be significant, especially if the utilized route that contains the intelligent attacker is a long-lived one. An additional line of defense is provided by r_l, whose threshold can be set to detect a possible abuse of the r_s rating. If the running average of delivered over transmitted pieces drops below an acceptable threshold, then the path is discarded independently of the r_s rating. For example, if $\beta/\alpha = 1/10$, an adversary could discard approximately *9%* of the transmitted packets; then, r_l^{thr} could be set equal to *95%* for instance to ensure lower loss of data.

The mechanisms for updating both the r_s and r_l are necessary, because we cannot make any assumption on the attack pattern. An adversary could be latent for a long period, exhibiting fully benign behavior, and be activated exactly when it can cause the greatest harm. Or it could behave maliciously in an intermittent and apparently pseudo-random manner. SMT can mitigate such malicious behavior since it does not rely on "test packets" or a "testing period" to assess the path security. Such an approach would fail, since the communicating nodes can be easily misled to deem all paths as "safe." For instance, if the adversary can distinguish the test packets, it could forward them and later tamper with the actual data. If test packets are indistinguishable, then, the adversary needs to forward a number of packets until the end of the testing period, and then launch its attack.[11] And the more extensive the testing period, the higher the imposed transmission overhead and delay, without any guarantee that the "security" of the paths could be determined and malicious nodes could be isolated.

In contrast, while SMT transmits data, it provides effective probing at a low-cost due to the simultaneous routing across multiple routes. In other words, the actual routing across APS allows determination of the paths' condition. The transmission of a piece across a low-rated path, although it may appear as a costly operation, can be, indeed, beneficial. Due to the message dispersion, the source can easily tolerate loss of a piece, if indeed the path is not operational. At the same time, if the reduction of the rating was due to transient faults (either malicious or benign), the successfully received piece will still contribute to the re-construction of the message and, possibly, to the re-instatement of the path rating.

4. PERFORMANCE EVALUATION

Our experiments verify that the proposed protocol can, indeed, successfully cope with a high number of adversaries, while operating only in an end-to-end manner. SMT can deliver successfully more than twice the number of packets delivered by a protocol that secures only the route discovery phase but not the data-forwarding phase. Moreover, we find that SMT is successful in delivering data with low end-to-end delay, low routing overhead, and limited transmission overhead, when compared to SSP.

The *Secure Single Path (SSP)* protocol is the limiting case of SMT without the dispersion of outgoing messages and the use of a single path for each message transmission. SSP is equipped with the same end-to-end feedback and the fault detection mechanisms as SMT. SSP also re-transmits each failed message $Retry_{max}$ times, provides data integrity, authenticity, and replay protection as SMT does, and selects the shortest path in hops. SSP determines, utilizes, and maintains a single path only. Once the utilized path is deemed failed, a new route discovery may be needed in order to determine a new route.

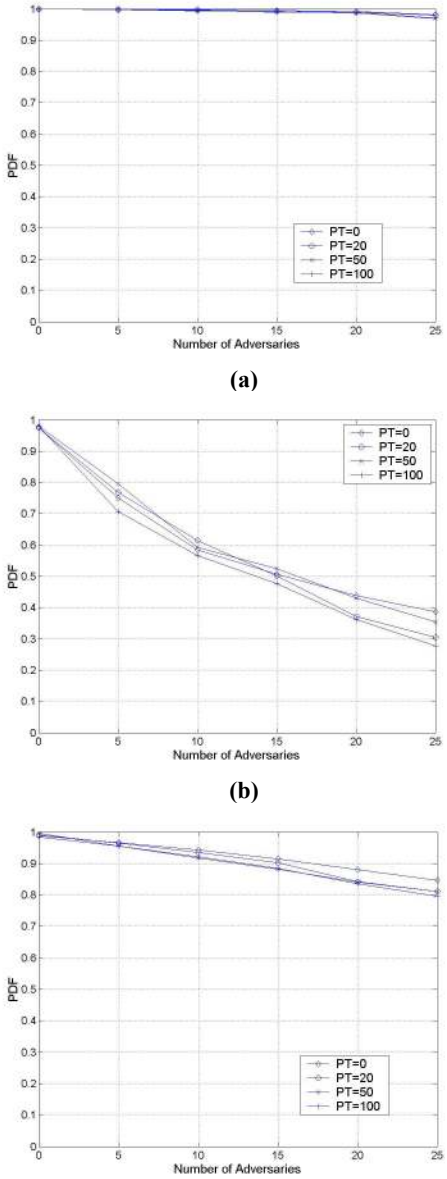
We evaluate here three protocols: (i) a single-path data forwarding protocol that does not employ any security mechanism to protect data transmissions, which we term the *Non-Secure Single Path (NSP)* protocol, (ii) the SSP protocol, and (iii) the SMT protocol. In all cases, we assume that the route discovery is secured, that is, the correctness of the discovered connectivity information is guaranteed.[12] Here, the secure discovery of one or more routes is performed by the *Secure Routing Protocol (SRP)*

[10] Care should be taken in the selection of β, since very small β values will cause very slow reinstatement of paths after experiencing short and transient losses.

[11] If the content of the packets can be analyzed, the attack could be selective, targeting packets of high importance. The selection of the packets to corrupt could depend on the knowledge of the employed protocols and the supported applications or could be purely subjective. For example, the loss of the last message of a multi-round interactive protocol can have a severe impact.

[12] But, again, this does not imply paths free of malicious nodes.

[1,2]. Multiple routes are discovered for SMT at the expense of increased overhead per route discovery, while a single route is discovered for SSP and NSP. We do not make any additional trust assumptions beyond the end-to-end security associations. Each source is securely associated with one destination, and sources transmit data to the same destination throughout the simulated period. OPNET™ simulation models were implemented.



(a)

(b)

(c)

Figure 3. Fraction of delivered messages. (a) SMT, (b) NSP, (c) SSP.

The network coverage area is a 1000m by 1000m square with 50 mobile nodes, with any two nodes able to communicate if they are within the reception distance, which is set to 300m. The resultant network topologies are bi-connected with high probability, i.e., for any two nodes it is highly likely that two node disjoint paths exist [22]. The nodes are initially uniformly distributed throughout the network area and their movement is determined by the random waypoint mobility model [7]. The node speed is uniformly distributed between 1 m/sec and 20 m/sec, and the pause times (PT) are 0, 20, 50, and 100 sec, with the simulated time equal to 300 seconds. The supported data rate is 2Mbps, and the medium access control protocol models transmission, queuing, and propagation delays and provides reliable communication at the data link level. Ten constant-bit-rate sources generate 4 messages/second with message/packet payload of 64 bytes. We note that the size of the buffer was not a limiting factor, i.e., no packets were lost due to buffer overflow at the source node. Each point on the presented graphs corresponds to the average over 15 randomly seeded runs and the number of adversarial nodes varies: 0, 5, 10, 15, 20, and 25 attackers.

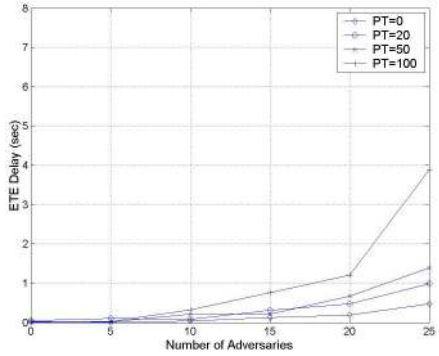
Our model is equivalent to the model that the attackers comply with the route discovery phase, relaying all the route requests, replies, or route and link state updates, in order to be placed on one or more utilized routes. Once they become part of a utilized route, attackers discard all data packets forwarded across the route(s) they belong to. Adversaries have the same features as the benign nodes (mobility, reception range) and are not assigned as sources or destinations. The protocol parameters used for these experiments include $Retry_{max}=3$, $r_s^{thr}=0.0$, $r_s^{max}=1.0$, $\alpha=0.5$, $\beta=0.05$.

The benefit from the presence of SMT is clearly shown in Fig. 3. In Fig. 3(a), SMT delivers more than *99%* of the transmitted messages within the range of *5* to *15* adversaries, and more than *96%* of the packets even when *50%* of the nodes are malicious. In contrast, the fast degradation of the NSP protocol comes as no surprise, as shown in Fig. 3(b). The average SMT improvement ranges from *32%* to *250%* as the number of adversaries increases. Without a mechanism that can detect malicious faults, an NSP source can detect a compromised route only if a link breakage is reported. This is true for any reactive secure routing protocol that does not secure the data transmission phase. In a malicious setting, such feedback could reach the source if it originated from a node at an upstream position relative to the first attacker lying on the route. As a result, even a small fraction of adversaries can inflict substantial packet loss – for example, with NSP and *5* adversaries present (*10%* of the network nodes), the average fraction of data packets dropped at the adversaries over the total number of transmitted packets ranges from *20*% to *28%*, depending on the node mobility. We re-emphasize that NSP does not re-transmit data.

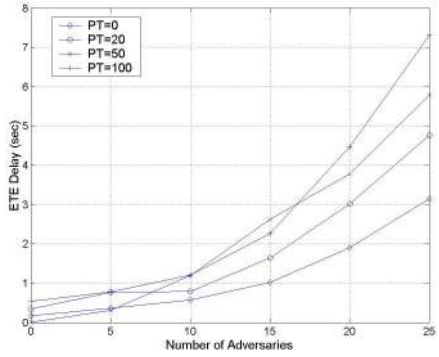
Although SSP can cope with adversaries much better than NSP, as Fig. 3(c) suggests, it becomes less effective as the number of adversaries increases, delivering, for example, only *84%* of the data with 20 adversaries present. More importantly, SMT delivers *3%* to *22%* more messages than SSP. SMT is more effective than SSP due the use of multiple paths and the message dispersion, which allow the delivery of data mainly without retransmissions. In contrast, SSP lacks the SMT's ability to simultaneously probe a set of routes; moreover, SSP can deliver a dropped message only by re-transmitting it. As a result, SSP (re-) transmissions may be successively attempted across (newly discovered) routes that are compromised, with messages lost after $Retry_{max}$ attempts.[13]

[13] Clearly, the fraction of delivered data could approach *100%* if the number of allowed retransmissions increased. However, such an improvement would come at the expense of significantly higher delays.

The most important advantage of SMT over SSP is revealed by the comparison of Fig. 4(a) and Fig. 4(b): SMT achieves dramatically lower end-to-end delays for all conducted experiments. The end-to-end delay is calculated from the time the message is created at the application layer until it is successfully delivered and after experiencing queuing and transmission delays to the destination, including possible re-transmissions. The average is taken over all received messages. The difference between SMT and SSP is evident even for completely or relatively benign environments, while SSP's delay increases at a much higher rate, as the number of adversaries increase. SMT achieves on the average a *94%* (for *5* adversaries) to *84%* (*25* adversaries) decrease in delay over SSP, when the pause time (PT) ranges from *0* sec to *50* sec, with SMT's improvement ranging from *94%* to *73%* for the case of more static networks (*PT = 100*). However, the most important observation is not the percentage of the decrease in delay, but rather its absolute values. SMT's delays are from *3* to *48* times lower than those for SSP. The only exception is the case for *25* adversaries and *PT = 100*, where the delay of SMT is approximately half of the delay of SSP.



(a)



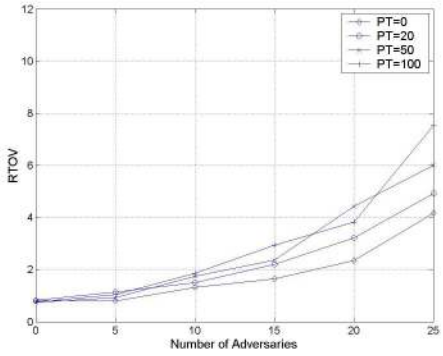
(b)

Figure 4. End-to-end message delay. (a) SMT, (b) SSP.

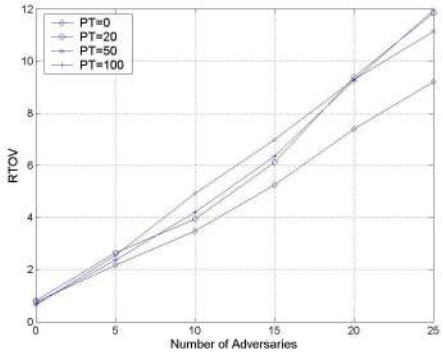
As the number of adversaries increases, it is more likely that the discovered route(s) will include adversaries. Thus, it becomes more probable that transmitted data will be lost, and that messages will be received after one or more retransmissions. As a result, the end-to-end delay increases as the number of adversaries increase. This is especially true for SSP, which relies on retransmissions and thus suffers higher end-to-end delays.

We also observe that lower mobility is detrimental to the protocol operation, or, inversely, higher mobility is conducive to the SMT's (and SSP's) goal, which is the successful and fast delivery of data at their destination. In our experiments, the higher the pause time, the lower the mobility. The more static the network is, the more probable it is for successive route discoveries to include the same adversaries and hold data buffered at their sources until "safe" route(s) are discovered. Fig. 4 shows that delays are in general higher when the mobility is lower, with the trend becoming clearer for high numbers of adversaries.

The successive route discoveries, which are necessary when compromised routes are repeatedly discovered, are responsible for the increase of the routing overhead, shown in Fig. 5(a) and Fig. 5(b), as the number of adversaries increase. The routing overhead is calculated as the ratio of all the transmitted routing query packets over the number of successfully received messages. The impact of decreasing mobility is apparent on the routing overhead curves as well.



(a)

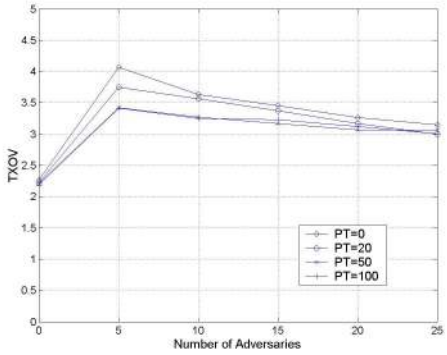


(b)

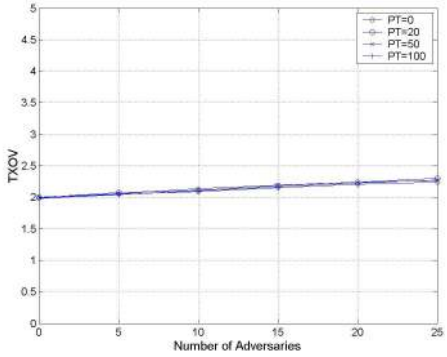
Figure 5. Routing overhead (RTOV). (a) SMT, (b) SSP.

It is important to note that SMT imposes significantly lower routing overhead than SSP, as the comparison between Fig. 5(a) and Fig. 5(b) shows. On the average, SMT achieves up to *68%* decrease in routing overhead (up to *63%* decrease when the *PT=100* scenarios are accounted for). The reason for this improvement is that SMT can mask route failures much more effectively than SSP does, and thus requires much less frequent route discoveries than SSP. To probe further, we take a look at the components of the routing overhead per discovery: the broadcasted route query packets, and the route reply packets. Both SMT and SSP discoveries incur the same cost due to route query packets. But SMT incurs approximately 6 times higher cost due to route replies. Nevertheless, SMT requires *57%* to *76%* fewer route

discoveries than SSP in an adversarial environment (with *19%* to *28%* improvement when there are no adversaries).



(a)



(b)

Figure 6. Transmission overhead (TXOV). (a) SMT, (b) SSP.

5. RELATED WORK

The protection of the traffic exchanged between two communicating nodes has been a fertile area of research outside the MANET community, with the Internet security architecture (*IPsec*) being the most prominent effort [8-11]. Goals such as the end-to-end authentication, integrity, and replay protection apply equally to the MANET context as well. However, the IPsec protocols assume the existence of a fixed routing and security infrastructure and need to be adapted to the MANET environment, if possible. Moreover, IPsec does not provide the means to determine the 'quality' of the routes and tolerate data loss, issues of paramount importance in networks with frequently changing connectivity and a significant fraction of adversaries.

Two transport layer protocols have features that bear some resemblance to those of our scheme, although there are fundamental differences. It has been proposed to use the *IDA* algorithm [3] to introduce redundancy, so that dropped Asynchronous Transfer Mode (*ATM*) cells would not cause a *TCP* segment to be dropped [13]. However, in that work, no security services are provided, there is no notion of multiple paths, and the types of failures are radically different than those we study here. The second related protocol is the *Stream Control Transport Protocol (SCTP)* [12]; it relies on the security services of IPsec and identifies multi-homed end-points using more than one transport address. However, SCTP cannot be applied in our malicious MANET context, as it does not determine the actual routes. In fact, SCTP data transmitted to different addresses might follow different routes. Such an operation can be harmful, since switching to a different "path" (transport address) does not provide any assurance that the actual multi-hop route will be different. Moreover, SCTP can be vulnerable to intermittent attacks, with adversaries forwarding "heartbeats," but dropping the actual packets

The use of multiple paths has been widely studied for the provision of quality of service (*QoS*) guarantees and load balancing in wired networks. In MANET, multiple paths have been utilized as a means to tolerate path breakages due to mobility. One such scheme proposes the use of diversity coding and provides an approximation for the probability of successful data transmission [6]. Another more recent scheme proposes the collection of link quality metrics, and the determination of a highly reliable set of link-disjoint paths (as opposed to node disjoint paths that we use here). The fast determination of the path set yields long-lived path sets that support communication with infrequent interruptions [26]. None of the two above-mentioned schemes provides security features or mechanisms to assess the quality of utilized routes in an end-to-end manner.

As for security solutions targeting MANET data transmission, the use of multiple routes existing in multi-hop topologies has been proposed in the early work of [27] and then in [1]. From a different perspective, it has been proposed to detect misbehaving MANET nodes and report such events to the rest of the network. All the network nodes maintain a set of metrics reflecting the past behavior of other nodes and then select routes through relatively well-behaved nodes [14]. A more recent work [19], makes the additional provision that all nodes have a secure association with all other network nodes. Thus, they can authenticate the misbehavior reports they exchange with their peers, seeking to detect and isolate malicious nodes that do not forward data packets. Another method to detect an attacker lying on the utilized route has been proposed in [20]. Once the communication across the route experiences a loss rate beyond a tolerable threshold, the source node initiates a search along the route to determine where the failure occurred. To do so, an encrypted and authenticated dialogue is initiated with each node along the route, with all network nodes assumed being securely associated with all their peers. Finally, a different approach [15] provides incentive to nodes, so that they comply with protocol rules and properly relay user data. The assumed greedy nodes forward packets in exchange for fictitious currency.

In this work, we have assumed the underlying routing protocol to be the Secure Routing Protocol (SRP) [1,2]. In SRP, only the end nodes have to be securely associated, with no need for cryptographic operations at the intermediate nodes, two factors that render SRP efficient and scalable. SRP provides one or more route replies, whose correctness is verified by the route "geometry" itself, while compromised and invalid routing information is discarded. A novel way of query identification protects the query propagation and the end nodes from DoS attacks, and query packets are handled locally by a priority scheme that enhances the robustness and the responsiveness of the protocol. Additionally, SRP, assisted by the Neighbor Lookup Protocol [5], ensures that adversaries cannot hide themselves from a route, and they cannot present themselves as multiple nodes, thus providing link-level correct connectivity information.

6. DISCUSSION AND FUTURE WORK

In this work, we showed how the data-forwarding phase can be secured by a protocol that operates solely in an end-to-end manner, without any further assumptions on the network trust and behavior of the adversaries. In fact, SMT can counter any attacker pattern, either persistent or intermittent, by promptly detecting non-operational or compromised routes. Moreover, SMT bounds the loss of data incurred by an intelligent adversary that avoids detection through manipulation of the path rating scheme. At the same time, SMT provides robustness to benign network faults as well, whether transient or not. The resilience to transient faults is very important, as it avoids discarding routes that are operational,[14] thus avoiding unnecessary overhead. Furthermore, resilience to benign faults, along with malicious ones, is important, since in MANET they may be frequent and in practice indistinguishable from forms of denial-of-service attacks.

Fault tolerance is dependent on the ability of the protocol to determine and utilize alternative, new routes when it detects non-operational ones. The multiplicity of routes that are, in general, expected to be available in MANET multi-hop topologies can be clearly beneficial. The availability or timely determination of such redundant routes may be the single most important factor for successful transmission across an adverse network. A rich APS, or many alternative routes, can be available only at the expense of routing overhead. This is generally true for any underlying routing protocol, even though the exact amount and type of routing overhead depends on the employed routing protocol. Increasing the size of the APS will most probably increase the routing overhead, which, in the case of reactive routing protocols, may result from more frequent route requests and additional replies, or, in the case of proactive protocols, more frequent link state updates. However, by trading off higher routing overhead, increased reliability (that is, higher fraction of delivered messages) and lower delays can be achieved.

In fact, the number of available diverse routes appears to control the trade-off between the delay, the routing and the transmission overhead, and the fraction of delivered messages. For example, the larger the size of the utilized APS, the more probable the successful reconstruction of the dispersed message will be and, consequently, the fewer the data re-transmissions and, thus, the lower the message delay.

The protocol adapts to either reduce the overhead or increase its fault tolerance, by selecting for each message the number of paths, among those available, and the redundancy factor. It starts with selecting an APS of K shortest (in terms of hops) paths [21]. Without having the opportunity to "probe" the paths and assuming that initially all nodes are equally probable to be malicious, selecting the shortest paths is equivalent to the selection of the most secure paths. The source maintains an estimate, p_i, of the probability that each APS path is operational. For each combination of the number of paths, m, and the feasible values of r, the probability that a transmission is successful is calculated with the estimated values for p_i–s in hand. The source selects m and r that yield a probability of successful delivery equal or as close as possible to the required probability of successful message delivery, P_{GOAL}, (determined, for example, by the application layer). The reader is referred to [28] for additional discussion and implementation details.

An open issue of interest is how to obtain estimates or predictions of the probability that a route will be operational. The complexity of such a task is increased, because of the numerous factors that affect the condition of the utilized routes. Mobility, congestion, transmission impairments, and an arbitrary, possibly intermittent and changing over time attack pattern, have to be taken into consideration. Through its interaction with the network and the feedback it obtains from the trusted destination, each node can gradually 'construct' such estimates. Clearly, the network conditions and characteristics can change over time. More simply, parameters such as the network connectivity, density, or the number of attackers present can differ according to the nodes' neighborhood. In any case, a feasible estimation method would be able only to continuously track[15] such changes and to provide rough estimates.

A plausible approach to obtain the probabilities of operational routes would be to collect statistics on the lifetimes of all the utilized routes.[16] It would be helpful to categorize routes according to attributes such as the length or whether the route includes any additional trusted nodes, other than the destination. Moreover, it would be more meaningful to update such measurements by assigning a lower weight to earlier observations in order to account for the network dynamics. For example, a node could quantize path lifetimes and retain measurements and estimates for a set of intervals. Then, if a newly determined path of length i has been operational for a period t in the $[t_x,t_{x+1}]$ interval, the node utilizes the estimate of the probability that such a path will survive for a period $t' > t$, with t' in the $[t_{x+1},t_{x+2}]$ interval. The investigation and evaluation of such mechanisms are left as future work.

Finally, we note that, despite the use of re-transmissions, SMT does not assume the role of a transport layer protocol - it operates at the network layer to secure the data forwarding and improve significantly the reliability of message delivery. However, SMT provides security and protects from frequent disruptions at the expense of increased traffic at the network, especially when data loss is detected. If there is not enough capacity in the network (at the link and at the network layers) to accommodate both the data flows and the SMT's overhead, the upper layer data rate could be decreased, for example, by the congestion control mechanism of the transport layer protocol.

7. CONCLUSIONS

In this paper, we have presented the *SMT* protocol to secure the data forwarding operation for *MANET* routing protocols. Our protocol takes advantage of topological and transmission redundancies and utilizes feedback, exchanged only between the two communicating end-nodes. This way, SMT remains effective even under highly adverse conditions. Moreover, features such as low-cost encoding and validation mechanisms, and partial retransmissions render the scheme efficient. By relying solely on the end-to-end security associations, *SMT* can secure effectively the data transmission without prior knowledge of the network

[14] For example, a transient loss can be caused due to network impairments or due to an adversary that employs a selective, intermittent attack pattern to avoid detection. Nevertheless, the route links may remain intact after such transient failures.

[15] Rather than determine from 'cold'.

[16] The lifetime defined as the period from the determination of a route till the route is deemed failed.

trust model or the degree of trustworthiness of the intermediate nodes.

Our performance evaluation confirms that *SMT* can naturally complement any protocol that secures the route discovery and can shield the network operation by delivering up to *250%* more packets despite the presence of substantial fraction of nodes as attackers. We also confirmed that SMT outperforms SSP, a single-path secure data transmission protocol equipped with the SMT's mechanisms. The end-to-end delays achieved by SMT are up to *94%* lower than the delays of SSP. Yet, SMT delivers up to *22%* more messages. And it does so with *68%* lower routing overhead and only with up to *48%* data and feedback transmission overhead. In conclusion, SMT's low overhead and its efficient and effective operation render SMT applicable to a wide range of MANET instances. The highly successful delivery of messages, in spite of the presence of adversaries and, most importantly, the low end-to-end delay clue on the ability of the protocol to support QoS for real-time traffic.

8. REFERENCES

[1] 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.

[2] 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.

[3] M.O. Rabin, "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance," *Journal of ACM*, Vol. 36, No. 2, pp. 335-348, Apr. 1989.

[4] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," *RFC 2104*, Feb. 1997.

[5] 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.

[6] A. Tsirigos and Z.J. Haas, "Multipath Routing in the Presence of Frequent Topological Changes," *IEEE Comm. Magazine*, pp. 132-138, Nov. 2001.

[7] J. Broch, D.A. Maltz, D.B. Johnson, Y-C. Hu, J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," in proceedings of the *4th International Conference on Mobile Computing (Mobicom'98),* 1998.

[8] S. Kent and R. Atkinson., "Security Architecture for the Internet Protocol," *IETF RFC 2401*, Nov. 1998.

[9] S. Kent and R. Atkinson, "IP Authentication Header," *IETF FC 2402*, Nov. 1998.

[10] S. Kent and R. Atkinson, "IP Encapsulating Security Payload," *IETF FC 2406*, Nov. 1998.

[11] D. Maughan, M. Schertler, M. Schneider, and J. Turner, "Internet Security Association and Key Management Protocol," *IETF RFC 2408*, Nov. 1998.

[12] R. Stewart et al, "Stream Control Transmission Protocol," *IETF RFC 2960*, Oct. 2000.

[13] A. Bestavros and G. Kim, "TCP-Boston: A Fragmentation-Tolerant TCP Protocol for ATM networks," in proceedings of the *IEEE Infocom '97*, Kobe, Japan, Apr. 1997.

[14] S. Marti, T.J. Giuli, K. Lai, M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," in proceedings of the *6th MobiCom*, BA Massachusetts, Aug. 2000.

[15] L. Buttyan and J.P. Hubaux, "Enforcing Service Availability in Mobile Ad Hoc WANs," in proceedings of the *1st MobiHoc*, BA Massachusetts, Aug. 2000.

[16] W. Diffie and M.E. Hellman, "New directions in cryptography," *IEEE Transactions in Information Theory 22*, 1976

[17] M. O. Rabin, "Probabilistic algorithms in finite fields," *SIAM Jornal of Comput. 9*, 1980.

[18] R. Zucceratto and C. Adams, "Using Elliptic Curve Diffie-Hellman in the SPKM GSS-API," *IETF Internet Draft,* Aug. 1999.

[19] S. Buchegger and J.Y. LeBoudec, "Performance Evaluation of the CONFIDANT protocol," in proceedings of the *Third ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2002)*, Lausanne, Switzerland, Jun. 2002.

[20] B. Awerbuch, D. Holmer, C. Nita-Rotaru and H. Rubens, "An On-Demand Secure Routing Protocol Resilent to Byzantine Failures," in proceedings of the *ACM WiSe 2002*, Atlanta GA, Sept. 2002.

[21] R. K. Ahuja, T. L. Magnati, and J.B. Olin, "Network Flows," *Prentice Hall*, Upper Saddle River, NJ, 1993.

[22] C. Bettstetter, "On the Minimum Node Degree and Connectivity of a Wireless Multihop Network," in proceedings of the *Third ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2002)*, Lausanne, Switzerland, Jun. 2002.

[23] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: A secure on-demand routing protocol for ad hoc networks," in proceedings of the *8th ACM International Conference on Mobile Computing and Networking (MobiCom)*, Sept. 2002.

[24] B. Dahill, B.N. Levine, E. Royer, C. Shields. "A Secure Routing Protocol for Ad Hoc Networks," *Technical Report UM-CS-2001-037, EE&CS, Univ. of Michigan*, Aug. 2001.

[25] M. G. Zapata and N. Asokan, "Securing Ad hoc Routing Protocols," ," in proceedings of the *ACM WiSe 2002*, Atlanta GA, Sept. 2002.

[26] P. Papadimitratos, Z.J.Haas, and E.G.Sirer, "Path Set Selection in Mobile Ad Hoc Networks," in proceedings of the *Third ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2002)*, Lausanne, Switzerland, Jun. 2002.

[27] L. Zhou and Z.J. Haas, "Securing Ad Hoc Networks," *IEEE Network Magazine*, vol. 13, no.6, Nov/ Dec. 1999.

[28] P. Papadimitratos and Z.J. Haas, "Secure Message Transmission in Mobile Ad Hoc Networks," *Elsevier Ad Hoc Networks Journal*, vol. 1, no. 1, Jan/Feb/March 2003.