Ad Hoc Networks

Volume 1, Issue 1, July 2003

ELSEVIER

Amsterdam – London – New York – Oxford – Paris – Shannon – Tokyo

Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

ELSEVIER

Ad Hoc Networks 1 (2003) 193–209

Ad Hoc Networks

www.elsevier.com/locate/adhoc

Secure message transmission in mobile ad hoc networks

Panagiotis Papadimitratos *, Zygmunt J. Haas

Electrical and Computer Engineering Department, Cornell University, 395 and 323 Rhodes Hall, Ithaca, NY 14850, USA

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 propose the secure message transmission (SMT) protocol to safeguard the data transmission against arbitrary malicious behavior of network 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 83% more data messages than a protocol that does not secure the data transmission. Moreover, SMT achieves up to 65% lower end-to-end delays and up to 80% lower delay variability, compared with an alternative single-path protocol—a secure data forwarding protocol, which we term secure single path (SSP) protocol. Thus, SMT is better suited to support quality of service 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.

Keywords: MANET security; Secure routing; Secure routing protocol; Secure message transmission; Multipath routing

1. Introduction

Secure communication, an important aspect of any networking environment, is an especially significant challenge in ad hoc networks. The MANET paradigm seeks to enable communication across networks whose topology and membership can change frequently. Its distinctive feature is that network nodes need to collaborate with their peers in supporting the network functionality. In such an environment, malicious or selfish nodes can disrupt or even deny the communications of potentially any node within the ad hoc networking domain. This is so, exactly because every node in the network is not only entitled, but is in fact required, to assist in the network establishment, the network maintenance, and the network operation.

The challenge in addressing these security vulnerabilities is due to the above particular MANET characteristics and due to the fact that traditional security mechanisms may be inapplicable here. First, the practically invisible or non-existent administrative boundaries encumber the a priori classification of a subset of nodes as trusted.

* Corresponding author.
E-mail addresses: papadp@ece.cornell.edu (P. Papadimitratos), haas@ece.cornell.edu (Z.J. Haas).
URL: http://wnl.ece.cornell.edu

1570-8705/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S1570-8705(03)00018-0

Moreover, in such a volatile communication environment, the determination of the nodes that can be trusted based on monitoring of the node's interactions with the rest of the network can be very difficult, and the overhead and especially the delay to make such inferences can be prohibitively large.

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 place themselves 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.

To secure the data transmission phase, we present here the secure message transmission (SMT) protocol, a secure end-to-end data forwarding protocol tailored to the MANET communication requirements. SMT safeguards the communication across an unknown, frequently changing network in the presence of adversaries that exhibit arbitrary malicious behavior. We emphasize that the goal of SMT is not to securely discover routes in the network—the security of this phase should be achieved by protocols such as the secure routing protocol (SRP) [1,2]. 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. [1] Then, the goal of SMT is to ensure routing over such routes, in spite of the presence of such adversaries. In this sense, SMT is a protocol that allows tolerating rather than detecting and isolating malicious nodes.

In the rest of paper, we first provide an overview of SMT and then describe its operation in Section 3. The performance evaluation of SMT through simulation experiments that compare SMT to alternative protocols follows. Next, we briefly present related work. In Section 6, we provide a discussion and describe future work, before we conclude.

2. Overview of SMT

The SMT protocol safeguards pair-wise communication across an unknown frequently chang-

[1] 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.

ing network, possibly in the presence of adversaries. 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. Its goal is to promptly detect and tolerate compromised transmissions, while adapting its operation to provide secure data forwarding with low delays. We especially emphasize the low-delay characteristic of SMT, as we believe that one of the main applications of SMT is in support of quality of service (QoS) for real-time traffic. [2]

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. [3] 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, [4] 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 based on 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.

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,

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

[3] The two nodes can negotiate a shared secret key, e.g., via the elliptic curve Diffie–Hellman algorithm [17,19] 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 $K_{S,T}$.

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

it awaits the missing packets that are retransmitted by the source. The number of retransmissions is limited to $\text{Retry}_{\max}$ 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.

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 (three out of four) 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.

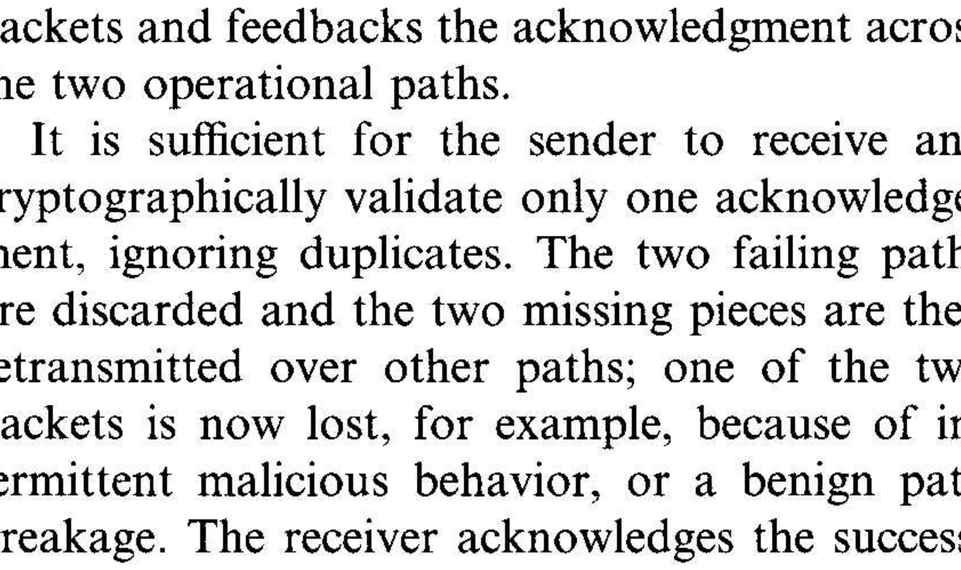


Fig. 1. Simple example of the SMT protocol.

3. Details of SMT operation

3.1. Determination of the APS

SMT can operate with any underlying routing protocol, [5] 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, such as SRP [1,2] or SLSP [5], 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.

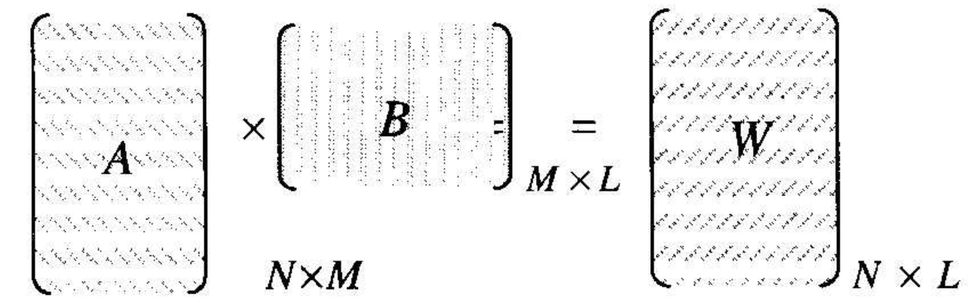
[5] As long as the routing protocol is capable of discovering multiple routes.

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 reconstruction 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 $[0 \ldots 2^m - 1]$ 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*. [6] 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 $a_i = [1, u_i, \ldots, u_i^{M-1}]$, $1 \leqslant i \leqslant N$, and $N < p$. The vectors of matrix A should be selected from a precomputed set used by both ends, which we assume is 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 $s_1, s_2, \ldots, s_L$ and they are arranged as columns of the array B, as illustrated on Fig. 2(a). 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 $w_i = [a_i s_1, a_i s_2, \ldots, a_i s_L]$.



(a)

(i). A =

1	45	dc
1	7d	f5
1	b0	7a
1	5b	95

(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' =

1	45	dc
1	7d	f5
1	b0	7a

(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

(b)

Fig. 2. (a) Example of an encoding of a message: a message of F bytes is segmented into pieces, which are the columns of matrix B, with $L = FS/M$. Matrix A holds N random vectors, and W is the resultant dispersed message, with its pieces as rows of matrix W. (Note that bytes/characters are treated as integers.) (b) Example of the IDA operation: all data values are 8-bit integers, shown in their hexadecimal representation.

Upon reception of any M pieces, the original message can be reconstructed. [7] Let $v_1, v_2, \ldots, v_M$ denote the M pieces used for reconstruction, which are in fact a subset of the N transmitted pieces, w_i. Each one of the v_i pieces corresponds to one of the a_i vectors, which are, by definition, linearly

[6] 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 [3,18].

[7] 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.

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 an $M \times L$ array, W', the original message reconstruction can be written as $B = [A']^{-1} \times W'_{M\times L}$.

Fig. 2(b) 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(b: i). The message has size $FS = 64$ bytes and it is padded with $PD = M \cdot \lceil FS/M \rceil - FS$ bytes. It is segmented into $L = (FS + PD)/M$ segments, arranged as the columns of the array B shown in Fig. 2(b: ii). The encoded message W is shown in Fig. 2(b: 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(b: iv). Matrix A' (Fig. 2(b: v)) holds the $\{a_i\}$ vectors that correspond to the received pieces, and the reconstructed message, shown on Fig. 2(b: 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, r_s^{thr} and r_l^{thr} respectively, the corresponding path is discarded. Both thresholds are protocol selectable parameters.

The r_s rating takes values in the interval $I = [r_s^{thr}, r_s^{max}]$ interval, with $r_s^{thr} \geqslant 0$, 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. [8] The constants α and β take values in the $(0, r_s^{max}]$ interval. After the ith transmission across a path that is not deemed failed yet, its rating is updated:

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

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 (successful) 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 r_s^{max}, then $s = s_0 + s_1$. Thus, the rating of the path can be written as

$$r_s(i) = r_s(0) + \beta s_1 - \alpha l. \tag{2}$$

For any route that is not deemed failed yet, $r_s(i) \geqslant r_s^{thr}$. Then, from Eq. (2) we get that $s_1\beta - l\alpha \geqslant r_s^{thr} - r_s(0)$. If we set $d = r_s(0) - r_s^{thr} \geqslant 0$, we can rewrite the previous inequality as

$$\beta s_1 - \alpha l + d \geqslant 0 \tag{2'}$$

where s_1 and l take integer values that are not simultaneously zero.

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

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

$f = \lceil (r_s^{max} - r_s^{thr})/\alpha \rceil$ 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 will still be considered operational. For example, 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 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

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

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

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

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 Eqs. (4′) and (3):

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

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 that α is not zero ($\alpha > 0$); otherwise, the attacker would have full control over a path ($\mathrm{BWL}^* = 1$). Furthermore, it must hold that $\alpha > \beta$ in order to keep $\mathrm{BWL}^* < 0.5$; in fact, the smaller β is compared to α, the lower BWL^* will be. [9]

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 up to 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 pseudorandom 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 only needs to forward a number of packets until

[9] 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.

the end of the testing period, and then launch its attack. [10] 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 reconstruction of the message and, possibly, to the reinstatement of the path rating.

3.4. Protocol autoconfiguration

The primary goal of the protocol adaptation is to maximize its effectiveness in highly adverse environments. An obvious solution would be for the source to discover and maintain a sufficiently high number of paths in order for the dispersed message to be successfully received. However, the APS selection is coupled to the rest of the protocol parameters, the network environment, and even to the requirements of the supported application.

In particular, the protocol adaptation can be viewed as the result of the interplay among the following parameters: (i) K, the number of utilized APS paths, (ii) k, the (S,T)-connectivity, i.e., the maximum number of $S \rightarrow T$ node disjoint paths from the source (S) to the destination (T), (iii) r, the redundancy factor of the information dispersal, and (iv) x, the (maximum) number of malicious nodes.

If M out of N transmitted packets are required for successful reception then $r = N/M$. For an allocation of one piece per path, K should be at least N, and the required number of packets is $M = K/r$. Equivalently, the higher x is, the larger K should be for a fixed r, in order to tolerate a higher number of faults. The condition for successful reception, showing the relationship among the parameter values, is

$$x \leqslant \lceil K \times (1 - r^{-1}) \rceil. \tag{6}$$

If the adversarial nodes constitute a cut of cardinality C_X, the result could be either a partitioned network (if $C_X \geqslant k$) as seen by S and T, or a mere failure to reconstruct the message at the receiver (if $k > C_X \geqslant k - M$).

The misbehavior pattern of the adversaries is an additional factor that affects the operation of the protocol; if, ideally, the behavior of the adversary could be predicted, the protocol could be optimally reconfigured. However, the behavior of the attacker can be arbitrary and time dependent. Moreover, the two communicating nodes will have, in general, no a priori knowledge about the security of the network or the trustworthiness of the rest of the nodes. [11] Since the source has no initial knowledge of the security of the individual paths or nodes, any node on a determined path can be malicious and disrupt the protocol operation. Alternatively, we can consider initially any single node as equally probable to be an adversary.

The protocol starts with selecting an APS of K shortest (in terms of hops) paths [22]. Without having the opportunity to "probe" the paths and assuming that all nodes are equally probable to be malicious, selecting the shortest paths is equivalent to the selection of the most secure paths. The shorter a path, the fewer the intermediate nodes, and thus the lower the probability that the path will be compromised.

[10] 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 has a severe impact.

[11] This is clearly true for an open, civilian network with disparate nodes collaborating only for the provision of basic networking services. But it could be true even for a battlefield network, if a number of initially trusted nodes are hijacked.

In addition to the utilization of a high number of paths, the chances of successful message reconstruction increase as the redundancy factor increases. Thus, the use of a higher redundancy factor could be particularly useful during the initial transmissions across a newly determined APS, when the uncertainty on the quality of the routes is higher, or, when the protocol operates in a highly adverse environment with a limited number of paths. After a number of such costly transmissions, the source can switch to a smaller number of paths and a lower redundancy factor. The use of all available paths, especially if it were combined with high redundancy, may not be desirable at all times. It can introduce unnecessary network overhead, especially when the nodes operate in a low-risk environment.

For efficient operation of the protocol without compromising its effectiveness, we propose the following method. The source determines an APS of K node disjoint paths and calculates an estimate p_i, the probability that each path is operational, for each of the APS paths. The calculation of such estimates is beyond the scope of this paper and can be done in a number of alternative ways, based on the interaction of the nodes with the network. [12] We discuss alternative ways in Section 6. For each message transmission, one piece is assigned to each of the paths, which implies that all possible values of r are easy to determine. For an APS of K paths, the source can select to utilize $i = 1, 2, 3, \ldots, k$ paths. If i paths are utilized, then at most i pieces can be sent; i.e., N must be equal to i. Consequently, r can take one of the following values: $i/1, i/2, \ldots, i/M, \ldots, i/i = 1$. In total, there are exactly $k^2/2$ feasible values for r and they can be pre-calculated.

For a particular combination of r and i and the estimated values for p_i, it is a straightforward exercise to calculate the probability that a transmission is successful. The calculation is based on the assumption of disjointness of routes, which allows us to assume that the route failures are independent. The event of a successful message reconstruction is complementary to the event that any $i - M + 1 = i - i/r + 1$ paths or more fail, and the sought probability can be calculated numerically, or with the help of an approximation [6].

The source can then select the required redundancy and the number of paths in the following way. If P_{GOAL} is the required probability of successful packet delivery (for example, as determined by the application layer), then the source can select i and r that yield a value equal to P_{GOAL}, or the closest possible if this cannot be achieved.

Note that similar probabilities of success can be achieved with different combinations of i and r. One way to select this pair of values is to determine first the least number of paths $K_{\min}$ that are necessary to achieve P_{GOAL}, and, then, select the minimum r among the feasible values, given $K_{\min}$. Essentially, this is equivalent to searching first along the diagonal of the (i, r) matrix of the calculated probabilities of success and then searching along the selected row.

Finally, we note that a high r can compensate for low K to some extent, while a low r may yield a low probability of success even if K is very high. On one hand, it is not possible to do anything more than maximally dispersing the message(s) when a small-size APS limits the operation of the protocol and, on the other hand, it is preferable to utilize fewer paths and higher redundancy, when paths are long or similarly when their probabilities of operation are low.

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 and not the data-forwarding phase. Moreover, we find that SMT is successful in delivering data with low end-to-end delay, low delay jitter,

[12] One possibility is for SMT to approximate p_i based on r_1, the fraction of delivered over transmitted pieces. Clearly, the higher the number of pieces transmitted across a given path, the more meaningful this approximation is. However, we must emphasize that an increasing number of samples cannot provide any additional guarantee that the path will remain operational in the future.

and limited overhead, when compared to a protocol that uses no message dispersion.

The secure single path (SSP) transmission 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 retransmits each failed message $\text{Retry}_{\max}$ times, provides data integrity, authenticity, and replay protection as SMT does, and selects the shortest path in hops. SSP switches to a new route, once the selected route is deemed failed, and the links of the failed route are removed from the topology view.

Both SMT and SSP are provided with the same topology view. For all the experiments presented here, an *update_topology_view*() process provides the full picture of the network connectivity to the source nodes, when one or more routes are needed. This idealized route discovery incurs no delay and no control overhead in acquiring the connectivity information, and ensures that no stale routing information is utilized. [13] Various routing protocol incur different delays, routing and processing overhead, and impose differing constraints and limitations on the SMT and SSP operation. To isolate the performance of our protocols from the underlying routing protocol and avoid such dependencies on the underlying route discovery phase, we made the decision to use the above idealized route discovery mechanism. In the actual implementation of SMT, instead of our topology update primitive, different secure routing protocols among the ones presented in the literature [1,2,5,24–26] could be employed. [14]

For comparison purposes, 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 the cases, we assume that the route discovery is secured, that is, the correctness of the discovered connectivity information is guaranteed. [15] 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. For the simulation, we implemented OPNET™ models of the above protocols.

The network coverage area is a 1000 m × 1000 m square with 50 mobile nodes, with any two nodes able to communicate if they are within the reception distance, which is set to 300 m. 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 [23]. 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 and 20 m/s, and the pause times (PT) are 0, 20, 50, and 100 s, with the simulated time equal to 300 s. The supported data rate is 2 Mbps and 10 constant-bit-rate sources generate 4 messages/s with 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. The medium access control protocol models transmission, queuing, and propagation delays and provides reliable communication at the data link level. Each point on the presented graphs corresponds to the average over 15 randomly seeded runs and the number of adversarial nodes is 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 as-

[13] Unutilized links are flushed when their age exceeds a *Max_Link_Age* threshold; here *Max_Link_Age* = 5.0 s.

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

[15] But, again, this does not imply paths that are free of malicious nodes.

signed as sources or destinations. The protocol parameters used for these experiments include $P_{GOAL} = 0.99$, $\text{Retry}_{max} = 3$, $r_s^{thr} = 0.0$, $r_s^{max} = 1.0$, $\alpha = 0.33$, $\beta = 0.033$.

The benefit from the presence of securing the data transmission is clearly shown in Fig. 3. In Fig. 3(a), SMT delivers more than 99% of the transmitted messages within the range of 5–15 adversaries and more than 95% of the packets even when 50% of the nodes are malicious. Similar performance (approximately 1–2% degradation compared to SMT) is achieved by SSP, as shown in Fig. 3(c). In contrast, the fast degradation of the NSP protocol comes as no surprise, as shown in Fig. 3(b). The improvement of SMT over NSP ranges from 14% to 83% as the number of adversaries increases.

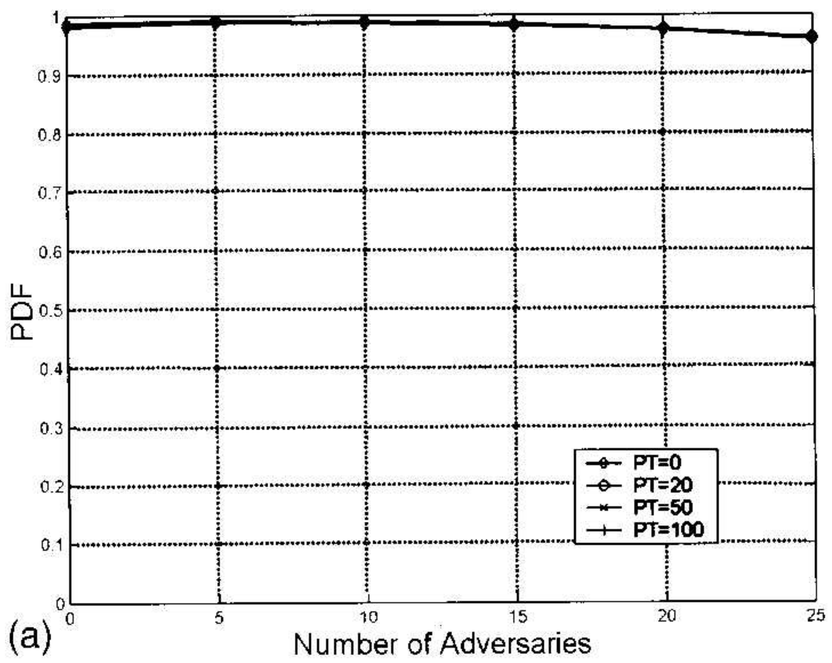
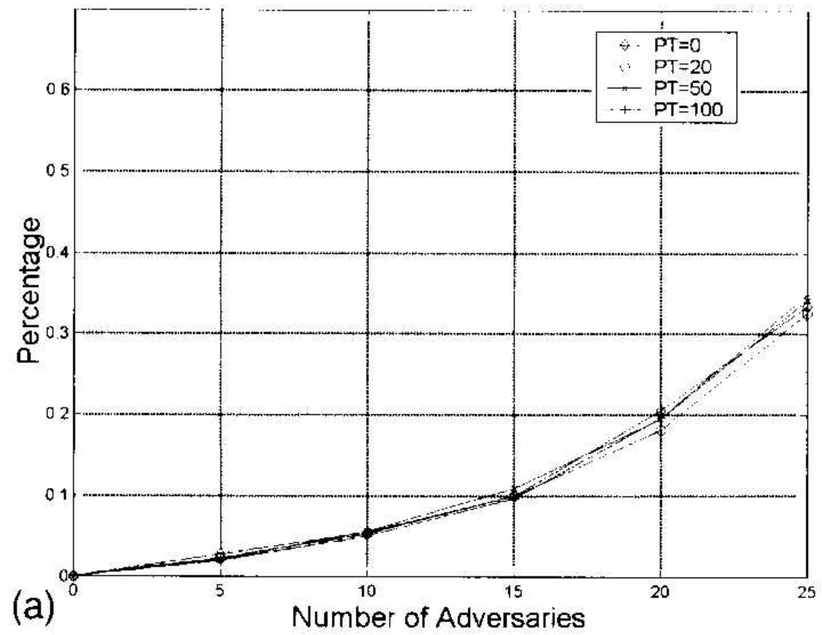
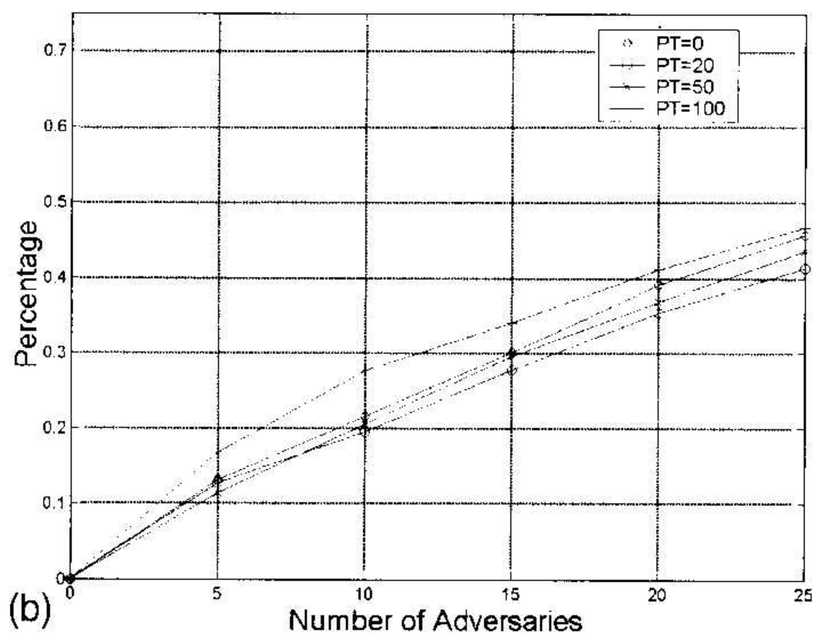


Fig. 3. Fraction of delivered messages: (a) SMT, (b) NSP, (c) SSP.

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. Fig. 4(b) verifies the limitation of such a mechanism, showing the average fraction of data packets dropped at the adversaries over the total number of transmitted packets from all the sources. Even a small fraction of adversaries can inflict substantial packet loss: for example, with five adversaries present (10% of the network nodes), NSP experiences a loss of up to 17% of the transmitted data. We reemphasize that NSP does not retransmit data.

In contrast, the percentage of packets lost at the adversaries when SMT or SSP is employed is significantly lower and increases at a much lower rate. For SMT, it is 10% or less with 30% of adversaries present and 20% or less even when 40% of the network nodes disrupt the communication (Fig. 4(a)). Comparatively, SSP allows lower loss of data at the adversaries—6% of packets are dropped with 30% of the nodes being adversaries, and less than 11% when 40% of the nodes discard data. This is mainly due to the fact that SMT increases the dispersion factor and the number of utilized routes, as the number of adversaries increase. The higher the number of paths, the more likely it is that compromised routes will be utilized and thus adversaries will have the opportunity to discard data. Nevertheless, due to the dispersion, overall, such a loss is not harmful for SMT, as we have seen in Fig. 3.





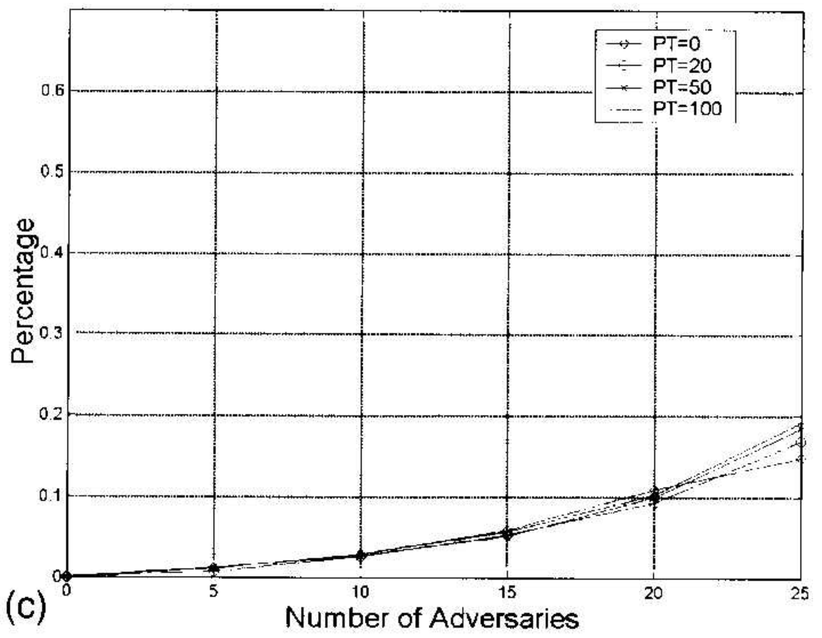
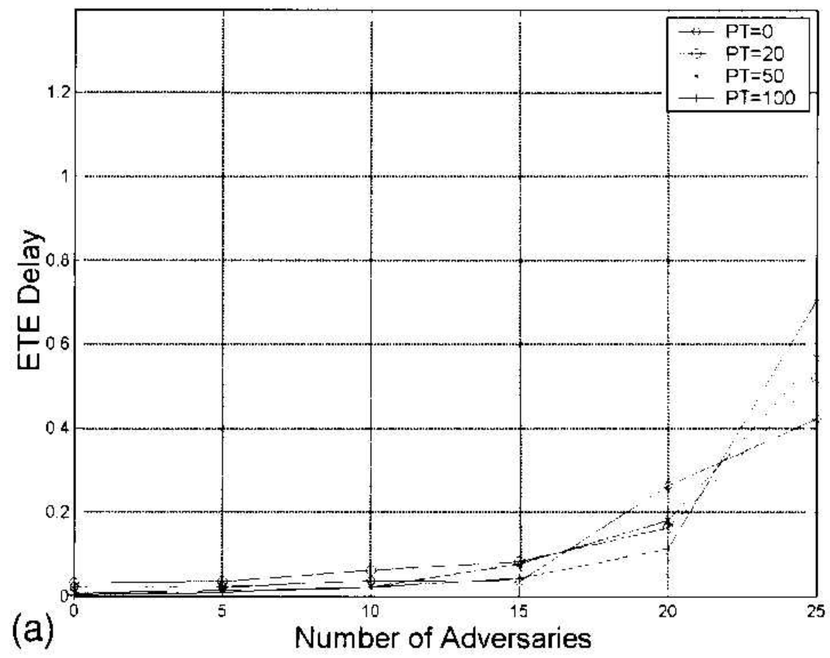


Fig. 4. Percentage of data packets dropped by attackers: (a) SMT, (b) NSP, (c) SSP.

The most important advantage of SMT over SSP is revealed by Fig. 5(a) and (b). Due to the simultaneous usage of multiple routes, SMT achieves lower end-to-end delays. The improvement becomes more evident as the number of adversaries increase. SMT yields up to 65% lower end-to-end delays, as compared with SSP, when the delays are calculated as the period from the generation of a message at the transport/applica-



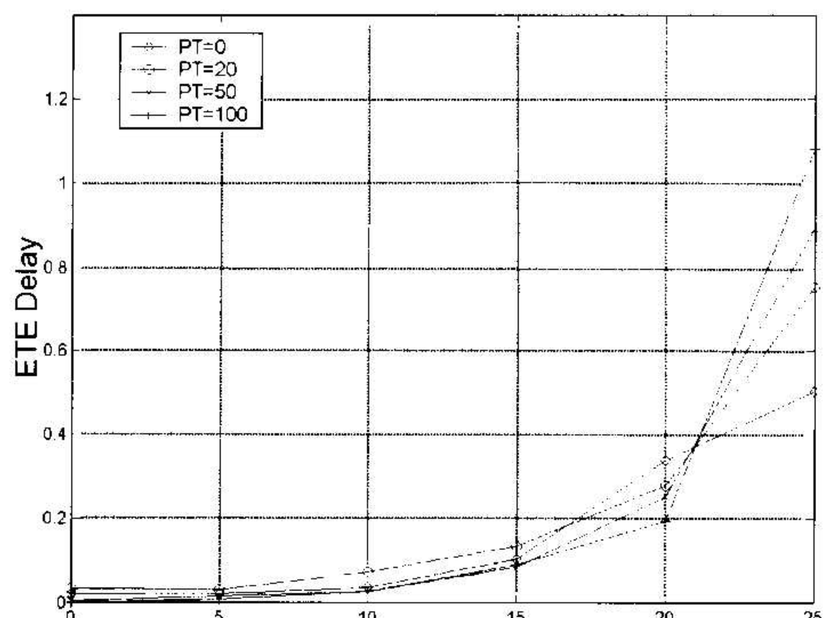


Fig. 5. End-to-end message delay: (a) SMT, (b) SSP.

tion layer until its successful delivery at the destination. Additionally, SMT provides significantly lower variability of the end-to-end delays; the variance of the delay for SMT ranges from 5% to 80% lower than the end-to-end delay variance for SSP. These two observations suggest that SMT is more capable of supporting real-time traffic.

Finally, both SMT and SSP introduce transmission overhead, due to the limited data retransmissions, the transmission of feedback, and for the case of SMT, the message dispersion. Fig. 6 shows the average overhead calculated as the sum of the transmitted message and the feedback pieces over the number of successfully delivered messages. We observe that both SMT and SSP have a trend of increasing transmission overhead as the number of adversaries increase. SMT, due to the simultaneous usage of multiple paths, incurs additional overhead (Fig. 6(a)), ranging from 6% up to 52% higher than the SSP overhead (Fig. 6(b)). Such an increase is relatively small (i.e., not pro-

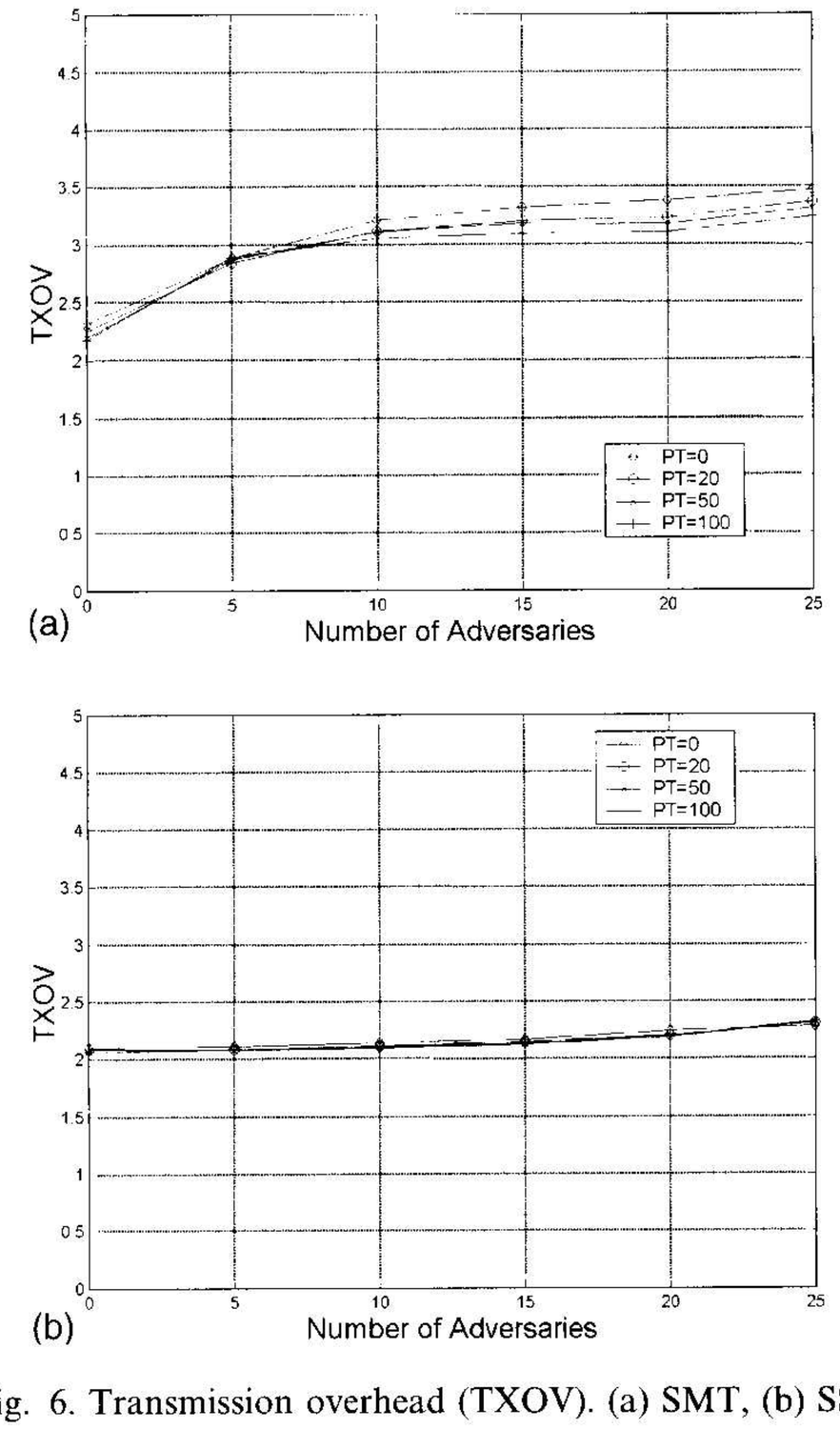


Fig. 6. Transmission overhead (TXOV). (a) SMT, (b) SSP.

portional to the number of available routes), and SMT efficiently trades bandwidth for delay.

SMT is efficient due to the combination of message dispersion and end-to-end feedback. It does not simply replicate messages or transmit redundant data. In our experiments, the average size of the APS is five paths, with a standard deviation of 3.4 paths. The spread distribution of the APS sizes implies that SMT operates with a low number of paths for a significant fraction of message transmissions, without this undermining its effectiveness. However, the less adverse the environment, the lower the fraction of available paths that is utilized. Moreover, the higher the number of adversaries, the more likely it is that one or more of the APS paths will be discarded. As long as message transmissions are successful, SMT operates with the remaining available and smaller-size APS.

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, because they assume the existence of a fixed routing and security infrastructure, are in general not applicable to MANET. [16]

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 [14]. 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) [13]; 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.

[16] The "router implementation" of IPsec would not make sense within a MANET domain. Similarly, the "tunnel mode" will not be applicable, unless a master/slave association exists (e.g., Bluetooth [12]), even though the dependent devices would be practically invisible at the routing layer. Furthermore, the fixed infrastructure that provides the routing functionality and the facilities for the distribution of IPsec policies is absent in MANET.

The use of multiple paths has been widely studied for the provision of 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 [27]. 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 [28] 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 [15]. A more recent work [20], 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 [21]. 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 [16] 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.

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. [17] This reduces the 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. For example, both SMT and SSP achieve highly reliable data delivery, despite their different methods of utilizing routes. Both of the protocols have access to the same, in most cases, connectivity information and thus an equally rich set of routes. As a result, both protocols 'converge' to the non-compromised route(s) among many discovered, if such route(s) exist at a particular instance.

A *rich* APS, or many alternative routes, can be available only at the expense of routing overhead.

[17] 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.

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. To increase the size of the APS, one has to impose increased routing overhead, which can be, in the case of reactive routing protocols, 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 higher the size of the utilized APS, the more probable the successful reconstruction of the dispersed message will be and, consequently, the fewer the data retransmissions and, thus, the lower the message delay.

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 [18] such changes and to provide rough estimates.

A plausible approach would be to collect statistics on the lifetimes of all the utilized routes, with the lifetime defined as the period from the determination of a route till the route is deemed failed. 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. In addition, maintain one set of such measurements for each category of paths of length i. 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.

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 trust model or the degree of trustworthiness of the intermediate nodes. In addition, secure transmissions can be achieved with low overhead in terms of the transmitted data, while SMT can operate effectively even when it utilizes a low number of APS paths. Consequently, SMT can be beneficial in low-connectivity topologies as well.

Our performance evaluation confirms that SMT can naturally complement protocols that secures the route discovery and can shield the network operation from adversarial behavior by delivering

[18] Rather than determine from "cold".

up to 83% more data packets, as compared with protocols that lack the secure data transmission feature. Furthermore, we find that SMT can reduce the end-to-end delay and the variation of delays up to 65% and 80%, respectively, compared with a single-path secure data forwarding protocol. These findings suggest that SMT is more capable to support real-time traffic in adverse MANET environments. 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 clues on the ability of the protocol to support QoS for real-time traffic in MANET.

Acknowledgements

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.

The authors would like to thank Prof. F.B. Schneider for his valuable suggestions and comments in the early stages of this work.

References

[1] P. Papadimitratos, 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, January 27–31, 2002.

[2] P. Papadimitratos, Z.J. Haas, P. Samar, The secure routing protocol (SRP) for ad hoc networks, Internet Draft, draft-papadimitratos-secure-routing-protocol-00.txt, December 2002.

[3] M.O. Rabin, Efficient dispersal of information for security, load balancing, and fault tolerance, J. ACM 36 (2) (1989) 335–348.

[4] H. Krawczyk, M. Bellare, R. Canetti, HMAC: Keyed-hashing for message authentication, RFC 2104, February 1997.

[5] P. Papadimitratos, 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, January 2003.

[6] A. Tsirigos, Z.J. Haas, Multipath routing in the presence of frequent topological changes, IEEE Comm. Mag. (November) (2001) 132–138.

[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, R. Atkinson, Security architecture for the Internet protocol, IETF RFC 2401, November 1998.

[9] S. Kent, R. Atkinson, IP authentication header, IETF FC 2402, November 1998.

[10] S. Kent, R. Atkinson, IP Encapsulating security payload, IETF FC 2406, November 1998.

[11] D. Maughan, M. Schertler, M. Schneider, J. Turner, Internet security association and key management protocol, IETF RFC 2408, November 1998.

[12] Bluetooth Special Interest Group, Specifications of the bluetooth system, http://www.blutooth.com.

[13] R. Stewart et al., Stream control transmission protocol, IETF RFC 2960, October 2000.

[14] A. Bestavros, G. Kim, TCP-Boston: A fragmentation-tolerant TCP protocol for ATM networks, in: Proceedings of the IEEE Infocom'97, Kobe, Japan, April 1997.

[15] S. Marti, T.J. Giuli, K. Lai, M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in: Proceedings of the 6th MobiCom, Boston, MA, August 2000.

[16] L. Buttyan, J.P. Hubaux, Enforcing service availability in mobile ad hoc WANs, in: Proceedings of the 1st MobiHoc, Boston, MA, August 2000.

[17] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Trans. Inf. Theory 22 (1976).

[18] M.O. Rabin, Probabilistic algorithms in finite fields, SIAM J. Comput. 9 (1980).

[19] R. Zucceratto, C. Adams, Using elliptic curve Diffie–Hellman in the SPKM GSS-API, IETF Internet Draft, August 1999.

[20] S. Buchegger, 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, June 2002.

[21] B. Awerbuch, D. Holmer, C. Nita-Rotaru, H. Rubens, An on-demand secure routing protocol resilent to byzantine failures, in: Proceedings of the ACM WiSe 2002, Atlanta, GA, September 2002.

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

[23] 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, June 2002.

[24] Y.-C. Hu, A. Perrig, 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), September 2002.
[25] 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, University of Michigan, August 2001.
[26] M.G. Zapata, N. Asokan, Securing ad hoc routing protocols, in: Proceedings of the ACM WiSe 2002, Atlanta, GA, September 2002.
[27] P. Papadimitratos, Z.J. Haas, 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, June 2002.
[28] L. Zhou, Z.J. Haas, Securing ad hoc networks, IEEE Network Mag. 13 (6) (1999).

Panagiotis Papadimitratos (papadp@ece.cornell.edu) is a Ph.D. candidate in the School of Electrical and Computer Engineering, at Cornell University, Ithaca, New York, affiliated with the Wireless Networks Laboratory. His research is concerned with security of computer and communication networks, design of network protocols, and wireless mobile networks. His work focuses on the security and fault tolerance of routing protocols, with an emphasis on solutions to support mobile ad hoc networking. His personal URL is: http://www.people.cornell.edu/pages/pp59/.

Zygmunt J. Haas received his B.Sc. in EE in 1979 and M.Sc. in EE in 1985. In 1988, he earned his Ph.D. from Stanford University and subsequently joined AT&T Bell Laboratories in the Network Research Department. There he pursued research on wireless communications, mobility management, fast protocols, optical networks, and optical switching. From September 1994 till July 1995, he worked for the AT&T Wireless Center of Excellence, where he investigated various aspects of wireless and mobile networking, concentrating on TCP/IP networks. As of August 1995, he joined the faculty of the School of Electrical and Computer Engineering at Cornell University. He is an author of numerous technical papers and holds fifteen patents in the fields of high-speed networking, wireless networks, and optical switching.

He has organized several workshops, delivered numerous tutorials at major IEEE and ACM conferences, and serves as editor of several journals and magazines, including the IEEE Transactions on Networking, the IEEE Transactions on Wireless Communications, the IEEE Communications Magazine, and the ACM/Kluwer Wireless Networks journal. He has been a guest editor of IEEE JSAC issues on "Gigabit Networks," "Mobile Computing Networks," and "Ad-Hoc Networks". He is a Senior Member of IEEE, a voting member of ACM, and the Chair of the IEEE Technical Committee on Personal Communications. His interests include: mobile and wireless communication and networks, personal communication service, and high-speed communication and protocols. His e-mail is: haas@ece.cornell.edu and his URL is: http://wnl.ece.cornell.edu.