SPREAD: Enhancing Data Confidentiality in Mobile Ad Hoc Networks

Wenjing Lou
Department of Electrical and Computer Engineering
Worcester Polytechnic Institute
Worcester, MA 01609
wjlou@ece.wpi.edu

Wei Liu Yuguang Fang
Wireless Networks Laboratory (WINET)
Department of Electrical and Computer Engineering
University of Florida, Gainesville, FL 32601
{liuw@, fang@ece.}ufl.edu

Abstract – **Security is a critical issue in a mobile ad hoc network (MANET). In this paper, we propose and investigate a novel scheme, Security Protocol for REliable dAta Delivery (SPREAD), to enhance the data confidentiality service in a mobile ad hoc network. The proposed SPREAD scheme aims to provide further protection to secret messages from being compromised (or eavesdropped) when they are delivered across the insecure network. The basic idea is to transform a secret message into multiple shares by secret sharing schemes and then deliver the shares via multiple independent paths to the destination so that even if a small number of nodes that are used to relay the message shares are compromised, the secret message as a whole is not compromised. We present the overall system architecture and investigate the major design issues. We first describe how to obtain message shares using the secret sharing schemes. Then we study the appropriate choice of the secret sharing schemes and the optimal allocation of the message shares onto each path in order to maximize the security. The results show that the SPREAD is more secure and also provides a certain degree of reliability without sacrificing the security. Thirdly, the multipath routing techniques are discussed and the path set optimization algorithm is developed to find the multiple paths with the desired property, i.e., the overall path set providing maximum security. Finally, we present the simulation results to justify the feasibility and evaluate the effectiveness of SPREAD.**

Keywords *— ad hoc networks, multipath rouitng, network security*

I. INTRODUCTION

A mobile ad hoc network (MANET) is a self-configurable, self-organizing, infrastructureless multi-hop mobile wireless network. Each node in a MANET is capable of moving independently, thus the network topology can change continuously and dramatically. Each node also functions as a router that discovers and maintains routes to other nodes and forwards packets for other nodes. A MANET can be promptly deployed without any wired base stations or infrastructure support. Few administrative actions need to be performed to set up such a network. The rapidly deployable and self-organizing features make a MANET very attractive in tactical and military applications, where fixed infrastructures are not available or reliable, but fast network establishment and self-reconfiguration are required. Primary applications of a MANET include the tactical communications in a battlefield, the disaster rescue after an earthquake, and so on, where the environment is hostile and the operation is security-sensitive.

Needless to say, security is a critical issue in a MANET. As compared with a fixed network or a wired network, the characteristics of a MANET pose many new challenges in security. First of all, the wireless channels are more susceptible to attacks such as passive eavesdropping, active signal interference, and jamming. Secondly, most ad hoc routing protocols are cooperative in nature and rely on implicit trust relationship among participating nodes to route packets. The dependency of the cooperation makes it more vulnerable to data tampering, impersonation, and denial of service types of attacks. Thirdly, the lack of a fixed infrastructure and a central concentration point makes it difficult to apply many conventional security solutions which are based on centralized control mechanisms. For example, it is difficult for an intrusion detection system to collect audit data, and it also impedes the deployment of widespread asymmetric cryptography because of the lack of a PKI (Public Key Infrastructure), where a centralized certificate authority (CA) is needed. Fourthly, mobile devices tend to have limited processing power and power consumption is also a major concern, which limits the practical deployment of computationally intensive or more comprehensive security schemes in MANET environments. Finally, the continuous and unpredictable ad hoc mobility clouds the distinction between normalcy and anomaly, thus makes it difficult to detect the malicious behaviors [1].

Due to these new challenges, many security solutions that have been effective in a wired network become inapplicable in a MANET. Much effort has been made to develop applicable security solutions dedicated to a MANET environment. Among them, key management, probably the most critical and fundamental security issue in a MANET, has attracted intensive attention in the last few years [2,3,4]. A number of secure routing protocols have also been proposed to protect the correctness of different types of ad hoc routing protocols [5,6,7,8]. Some other issues that have been addressed particularly for a MANET in the current literature include handling node misbehavior [9,10,11], intrusion detection [12], and other issues [1].

This work was supported in part by the U.S. Office of Naval Research under Young Investigator Award N000140210464 and under grant N000140210554, and the U.S. National Science Foundation under Faculty Early Career Development Award ANI-0093241 and under grant ANI-0220287.

0-7803-8355-9/04/$20.00 ©2004 IEEE.

The scheme suggested in this paper addresses data confidentiality service in a MANET. Data confidentiality is the protection of transmitted data from passive attacks, such as eavesdropping. Sensitive information, such as tactical military information transmitted across a battlefield (a MANET), requires confidentiality. Leakage of such information to enemies could cause devastating consequences. The wireless channel in a hostile environment is vulnerable particularly to the eavesdropping. Messages transmitted over the air can be eavesdropped from anywhere without having the physical access to the network components. Conventionally, confidentiality is achieved by cryptography. However, the limited resources, such as the limited battery power and processing capability, restrict the use of computationally intensive encryption schemes in a MANET. The computationally efficient encryption schemes sometimes are not secure enough. For example, the WEP (Wired Equivalent Privacy) protocol defined in IEEE 802.11 uses RC4 algorithm, which is a stream cipher and computationally efficient. However, it has been discovered that it can be decrypted through traffic analysis and dictionary-building attack that, after analysis of about a day's worth of traffic, allows real-time automated decryption of all traffic [13]. A more severe problem in a MANET is that, mobile nodes usually reside in an open and hostile environment. Nodes themselves might be compromised. For example, in the battlefield scenario, nodes might be captured. In this case, all the credential stored in the nodes would be compromised, including the keys used to encrypt the message. Any encryption scheme, no matter how secure it is, would not help.

Based on these observations, we propose a novel scheme, called Secure Protocol for REliable dAta Delivery (SPREAD), to statistically enhance data confidentiality in a MANET. The fundamental idea of SPREAD is shown in Figure 1. Assume that we have a secret message, if we send it through a single path, the enemy can compromise it by compromising any one of the nodes along the path. However, if we divide it into multiple pieces, and send the multiple pieces via multiple independent paths, then the enemy has to compromise all the pieces from all the paths to compromise the message. Improved security can be expected by this means.

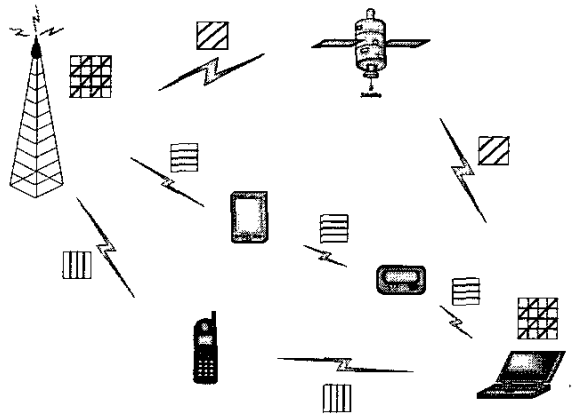

Figure 1 Fundamental idea of SPREAD

Here, to compromise the message, the enemy must accomplish at least two things. First, the enemy must physically intercept all pieces of the message. This can be done by either eavesdropping or compromising nodes. Either way, by spreading the message pieces over multiple paths, the enemy will have more difficulty to collect all the pieces. Secondly, we assume link encryption between neighboring nodes, and each link with different keys. Although, in general, key management is problematic in a MANET, the establishment of a shared session key between neighboring nodes is not that difficult [3]. Link encryption is also widely used to prevent the traffic analysis [24,25]. So even the enemy collected all the pieces, he/she has to decrypt all of them. The decryption can be done by either compromising the nodes or by brute-force type of attacks or traffic analysis, while the latter requires a large amount of encrypted data by the same key. The more data, the better chance the decryption. By spreading the traffic onto multiple paths, it is harder for the enemy to decrypt the message.

In this paper, we focus on how to exploit this SPREAD idea to develop a security enhancement protocol to strengthen the data confidentiality in mobile ad hoc networks. We address the improved security by dealing with the compromised nodes and eavesdropping problem. We evaluate the performance for both individual attacks and colluded attacks (multiple compromised nodes are working together to recover the message). We assume that the adversaries, after compromising the nodes, will attempt to remain in the network by launching only passive attacks in order to acquire more secure information. If the compromised nodes launch active attacks, such as stopping forwarding packets for other nodes or altering the information when forwarding, some intrusion detection mechanism [12] or the misbehavior detection schemes such as a watchdog proposed in [9] can be used to identify the compromised node quickly so that it will be excluded from the network.

The rest of the paper is organized as follows. In section II we provide a high-level conceptual system architecture and briefly discuss the major design issues involved in the implementation of the system. Then section III, IV, and V are dedicated to the three major design issues respectively, namely, secret sharing, share allocation, and path finding. The simulation results are presented in section VI and the paper is concluded in section VII.

II. System Architecture and Major Design Issues

A. *Threshold Secret Sharing*

The first issue is how to divide the message into multiple pieces? Simply chopping the message into multiple segments involves the least processing overhead. However, it does not provide extra security protection. Since each segment contains partial content of the message, which might be used to infer the content of the whole message. It is also difficult to protect the integrity of the message. In our SPREAD scheme, we use the threshold secret sharing algorithm to divide the secret message

into multiple pieces. Threshold secret sharing algorithms could divide a secret into N pieces, called *shares* or *shadows*. From any less than T shares one cannot learn anything about the system secret, while with an effective algorithm, one can reconstruct the system secret from any T out of N shares. This is called a (T,N) threshold secret sharing scheme [14-16]. Thus with a (T,N) secret sharing algorithm, the secret message can be divided into N message shares such that in order to compromise the message, the enemy has to compromise at least T shares. With less than the threshold, namely T, shares, the enemy could learn nothing about the message and has no better chance to recover the secret than an outsider who knows nothing at all about the message. This gives us the desired security properties. Another reason that we use secret sharing is that the generation of the message shares and the reconstruction of the message are all linear operations over a finite field (Shamir's Lagrange interpolating polynomial scheme [15]). In addition, the secret sharing scheme can be designed with cheating detection and cheater identification [17]. It is possible that after compromising a node, the adversary may attempt to cheat our system by sending us the faked or altered message shares. By embedding the cheater detection and identification, we can deterministically detect cheating and identify the cheater, no matter how many cheating shares are involved in the secret reconstruction. This is a very useful detection mechanism in an unreliable ad hoc network environment and helps to protect the integrity of the message transmitted.

B. *Share Allocation*

The second issue is how to select the paths, how to choose an appropriate value of (T,N), and how to allocate the shares onto each selected path such that the maximum security can be achieved. We consider the case that a message is compromised due to compromised nodes. We assume that if a node is compromised, all the credentials of that node will be compromised. So the message shares traveling through that node are all intercepted and recovered. Given the available independent paths and their corresponding security characteristics, the fundamental objective is to maximize the security by allocating the shares in such a way that the adversary has to compromise all the paths to recover the message. The simplest and most intuitive share allocation scheme is to choose N as the number of available paths, apply (N,N) secret sharing, and allocate one share onto each path. This will achieve the desired maximum security with least processing cost. However, in an ad hoc network, wireless links are instable and the topology changes frequently. Sometimes packets might be dropped due to the bad wireless channel condition, the collision at MAC layer transmission, or stale routing information. In the case that packet loss does occur, this type of non-redundant share allocation will disable the reconstruction of the message at the intended destination. To deal with this problem, it is usually necessary to introduce some redundancy (i.e. $T<N$) in the SPREAD scheme to improve the reliability, i.e. the destination would have better chance to receive enough shares for reconstructing the message. Generally speaking, security and reliability are two contradictive design goals - more redundancy implies better reliability but worse security. However, due to the salient feature of the threshold secret sharing, we develop the redundant SPREAD share allocation which could tolerate certain packet losses while at the same time maintain the maximum security, i.e. forcing the adversary to compromise all the paths to compromise the message. We formulate the share allocation into a constrained optimization problem, with the objective to minimize the message compromise probability. Our investigation to the optimal share allocation reveals that, by choosing an appropriate (T,N) value and allocating the shares onto each path carefully, we could improve the reliability by tolerating certain packet loss without sacrificing the security. The maximum redundancy we can add to the SPREAD scheme without sacrificing security is identified. The optimal share allocation is proposed. More details about share allocation are presented in section IV.

C. *Multipath Routing and Path Set Optimization*

The third issue is the multipath routing in ad hoc networks – how to find the desired multiple paths in a mobile ad hoc network and how to deliver the shares to the destination using these paths? Routing in a MANET presents great challenge because the nodes are capable of moving and the network topology can change continuously, dramatically and unpredictably. A great effort has been made in designing ad hoc routing protocols in response to the frequent topological changes. Multipath routing technique is a promising choice since the use of multiple paths in a MANET could diminish the effect of unreliable wireless links and the frequent topological changes. Several multipath routing schemes have been proposed to improve the reliability, fault-tolerance, end-to-end delay for bursty traffic, as well as to achieve load balancing. [18,19,20,21].

For our SPREAD scheme, we need independent paths, more specifically, node disjoint paths, because we are dealing with node compromising problem. Several multipath routing protocols have been proposed in MANETs with the design goal to find node-disjoint paths, such as the split multipath routing [19], the diversity injection technique [20], and the on-demand multipath routing [21]. The dynamic source routing (DSR) protocol itself is also capable of maintaining multiple paths from the source to a destination. Those proposed protocols are all on-demand, due to the network bandwidth limitation, and source routing type, as the source routing provides the source with the maximal capability of controlling the disjointness of the paths. Those on-demand protocols work by broadcasting the route inquiry messages throughout the network and then gathering the replies from the destination. Although those routing protocols are able to find multiple node-disjoint paths, the paths found directly by them might not be optimal for our SPREAD scheme as the path selection is usually based on the hop count or propagation delay, not necessary the security. For our SPREAD scheme, we take a similar on-demand and source routing type of approach. However, we make use of the "link cache" organization we proposed in [22] where each path returned to source is decomposed into individual links and represented in a unified graph data structure. Using such a link cache organization allows us to further optimize the path set used for SPREAD. Although we rely on an underlying routing protocol to provide us with a partial view of network topology, the optimization of

the path set can be done solely based on the discovered partial network topology, which is independent of the underlying routing protocols. For the optimization of the paths, we propose a security related link cost function such that the path can be found according to their security level (i.e. the probability that the path might be compromised). Then, we apply a maximal node disjoint path finding algorithm to discover as many paths as possible and at the same time as secure as possible. The multipath routing and path set optimization will be discussed in section V.

III. THRESHOLD SECRET SHARING SYSTEM

To better understand the scheme, we give a brief introduction to the threshold secret sharing system, which is used to generate the shares from a message (messages). Details can be found in [14]. Suppose that we have a system secret K and we divide it into N pieces, $S_1, S_2, ..., S_N$, called *shares* or *shadows*. Each of N participants of the system, $P_1, P_2, ..., P_N$, hold one share of the secret respectively. The generation of the secret shares guarantees that any less than T participants cannot learn anything about the system secret K, while with an effective algorithm, any T out of N participants can reconstruct the system secret K. This is called a *(T,N) threshold secret sharing scheme* [15,16]. A secret sharing scheme consists of two algorithms. The first is called the *dealer*, which generates and distributes the shares among the participants. The second is called the *combiner*, which collects shares from the participants and re-computes the secret, i.e., it produces the secret K from any T correct shares. A combiner fails to re-compute the secret if the number of the correct shares is less than T.

Several threshold secret sharing schemes have been developed. For illustration purpose, we take the Shamir's Lagrange interpolating polynomial scheme as an example. The dealer obtains the ith share by evaluating a polynomial of degree $(T-1)$

$$f(x) = (a_0 + a_1 x + ... + a_{T-1} x^{T-1}) \bmod p$$

at $x=i$:

$$S_i = f(i)$$

where p is a large prime number greater than any of the coefficients and is made available to both the dealer and the combiner, and the coefficient $a_0 (= K)$ is the secret while other coefficients $a_1, a_2, ..., a_{T-1}$ are all randomly chosen. Then, at a combiner, once T shares have been obtained, the combiner can reconstruct the original polynomial by solving a set of linear equations over a finite field $GF(p)$. For example, assume that the received T shares are $S_{i1}, S_{i2}, ..., S_{iT}$, the original polynomial $f(x)$ can be recovered by Lagrange interpolation.

$$f(x) = \sum_{j=1}^{T} S_{i_j} \cdot l_{i_j}(x) \bmod p$$

where

$$l_{i_j}(x) = \prod_{k=1, k \neq j}^{T} \frac{x - i_k}{i_j - i_k}$$

Particually, the original secret K can be recovered by calculating $f(0)$.

Efficient ($O(T \log^2 T)$) algorithms for polynomial evaluation and interpolation have been discussed in [26]. Moreover, depending on the number of paths in a MANET, the (T,N) value in our SPREAD will not be large. Even the straightforward quadratic algorithms are fast enough for practical implementation.

In the SPREAD, the source is the natural dealer, and can choose the a_0 as the message and choose other coefficients randomly on the finite field $GF(p)$, where p is appropriately chosen. The destination is a natural combiner. Upon receiving T shares, it is able to recover the original secure message.

Figure 2 is the illustration of applying the secret sharing algorithm onto the secret message at the source node. If a message is too large, it can be chopped up as we normally do in the transport layer. During this process, some scrambling may be helpful. Limited by the size of the chosen prime number p, the secret sharing is applied on a block by block basis, which is similar to any block cipher used to encrypt a large message. In addition, depending on the number of paths used, the SPREAD seems to waste a lot of bandwidth. To save the network bandwidth, in SPREAD all the coefficients $a_0, a_1, a_2, ..., a_{T-1}$ can be assigned message blocks, as shown in the figure.

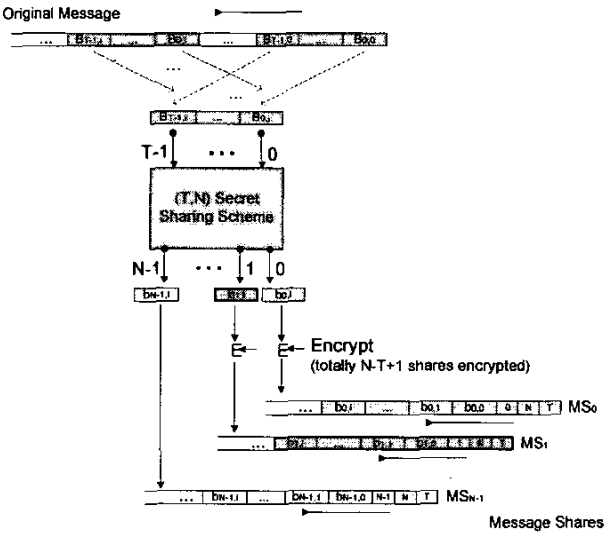

Figure 2 (T,N) Secret Sharing System

IV. OPTIMAL SHARE ALLOCATION

How to choose the appropriate values of (T,N) and how to allocate the N shares onto each selected path is another important issue in SPREAD design. Provided the available paths and their corresponding security characteristics, the first objective is to maximize the message security. In this section, we discuss the share allocation scheme that exploits the

redundant secret sharing scheme (where $T<N$) to achieve both data confidentially and reliability.

A. Problem Formulation and Notations

Assume that (T,N) secret sharing algorithm is applied to the message to be protected at source node. In the network layer, we assume that there are totally M node disjoint paths, path *1*, path *2*, ..., path M, available from the source to the destination. We use vector $\underline{p}=[p_1,p_2,\cdots,p_M]$ to denote the security characteristics of the paths, where $p_i\ (i=1,2,\cdots,M)$ is the probability that path i is compromised. Without loss of generality, we further assume $p_1 \le p_2 \le \cdots \le p_M$, which means that the paths are ordered from more secure one to less secure one. Note that the path security information, p, is available at source from the multipath routing protocols we will discuss in the next section. We assume that if one node were compromised, all the shares traveling through that node would be compromised. Therefore, we define that a path is compromised as when any one or more of the nodes along the path is compromised. For each path, we consider that if it were compromised, all the shares allocated to it would be compromised. Otherwise, if the path were not compromised, all shares on that path would be safe. As those paths are node disjoint, we further assume that the probability that one path is compromised is independent of others. As we pointed out in previous section, SPREAD scheme only enhance the data confidentiality statistically when the data are transmitted across the network. Thus the probability p_i does not include the probability that the source or the destination node is compromised, i.e., we assume source and destination are trustworthy. The protection of a node from being compromised is another issue and is out of the scope of this paper.

A share allocation scheme is used to allocate the N shares onto the M available paths. Denote the share allocation as $\underline{n}=[n_1,n_2,\cdots,n_M]$, where n_i is the number of shares allocated to path i, n_i is an integer, $n_i \ge 0$, $\sum_{i=1}^{M} n_i = N$. According to the secret sharing algorithm, the probability that the message is compromised equals to the probability that T or more shares are compromised. We denote the probability that the message is compromised in terms of the share allocation $\underline{n}$ as $P_{msg}(\underline{n})$. Then, the share allocation can be formulated to a constrained optimization problem

$$\textit{minimize}\ \ P_{msg}(\underline{n})$$

$$\textit{subject to}\ \ \sum_{i=1}^{M} n_i = N,\ n_i \text{ is an integer},\ n_i \ge 0$$

B. Maximum Security without Redundancy

Let define $r=1-T/N$ as the redundancy factor of the (T,N) secret sharing scheme. A non-redundant SPREAD scheme is one where r=0, e.g. $N=T$. It is easy to derive that given the number of available paths, M, and the corresponding path security characteristics $\underline{p}=[p_1,p_2,\cdots,p_M]$, the non-redundant (N,N) $(N>=M)$ secret sharing scheme would give the maximum security, i.e., minimum message compromise probability, when at least one share and at most *T-1* shares are allocated to each of the available paths, i.e.

$$\begin{cases} 1 \le n_i \le T-1, & i=1,\ldots,m \\ \sum_{i=1}^{m} n_i = N \end{cases}$$

This share allocation forces the adversary to compromise all the paths to compromise the message. This probability equals to the probability that all the paths are compromised.

$$P_{msg}(\underline{n}) = \prod_{i=1}^{M} p_i$$

It is noted that the maximum security provided only depends on the paths chosen. As p_i is a probability satisfying $0 \le p_i \le 1$, the more paths we use to distribute the shares, the less the probability is, and the more secure the message delivered. Thus, if given required security level (in terms of message compromise probability) γ_{P_n}, the SPREAD scheme should chose the first m paths, path *1*, path *2*, ..., path m, which satisfying $P_{msg}(\underline{n}) = \prod_{i=1}^{m} p_i \le \gamma_{P_n}$, to deliver the message.

C. Maximum Security with Redundancy

It is intuitive that non-redundant secret sharing scheme provides the maximum security to the message. However, it requires the successful reception of all the shares in order to reconstruct the original message. In an ad hoc network, wireless links are not stable. Packets might be dropped due to broken links, heavy MAC layer collision, or wireless signal fading, and so on. With our SPREAD scheme, as the shares are spread onto multiple paths, long paths might be used. The reliability of the message, in terms of transmission error, packet lost ratio, etc., may be further degraded. It is usually necessary to add some redundancy for reliability.

Redundancy is a common way to improve the reliability. It is based on the idea of sending more information than minimum requirement, so that the original message can be reconstructed in the event of loss in the network. It may be used independently of the multipath routing, by adding Forward Error Correction (FEC) code to each individual share. We denote this type of redundancy as serial redundancy. Serial redundancy is good for correcting noise-like random errors introduced to the bit stream, while helpless for the persistent errors or link failure. With a (T,N) secret sharing scheme, when $T<N$, we actually introduce redundancy from another dimension, the parallel redundancy. When this type of redundancy is used in combination with multipath routing, the system becomes more error tolerant, because a certain number $(N-T)$ of message shares can be corrupted or lost without affecting the reconstruction of the original message. The system also becomes more fault-tolerant because a certain fraction of the paths can be affected by failure without interrupting the flow of the information.

Using the same path selection criteria, i.e., choosing the first m most secure paths which satisfy the required security level, it is intuitive to show that, in order to achieve the

maximum security, the total number of shares allocated to any $m-1$ or fewer paths should be less than T. Again, this share allocation forces the adversary to compromise all the m paths to compromise the message. This is also a necessary and sufficient condition to achieve the maximum security. This condition can be simplified as

$$\begin{cases} N - n_i < T, & \forall i \in (1,2,\cdots,m) \\ n_1 + n_2 + \cdots + n_m = N \end{cases}$$

Recall $r = 1 - T/N$ is the redundancy factor of the secret sharing scheme. Then, we could derive a necessary condition for achieving the maximum security, i.e.

$$r < 1/m \quad (m \geq 2)$$

This is an important condition as it defines the maximum redundancy we can add to the SPREAD scheme without sacrificing the security. It indicates that to maintain the maximum security achievable from the chosen path set, the maximum redundancy we can add to the secret sharing algorithm is bounded by $r < 1/m$, where m is the number of chosen paths $(m \geq 2)$. In other words, we could claim that for a r-redundancy SPREAD scheme, the maximum security can be achieved only if the redundancy factor r satisfies $r < 1/m$ $(m \geq 2)$. Then by choosing an appropriate (T,N) value satisfying

$$T \geq N\frac{m-1}{m} + 1 \qquad (m \geq 2),$$

an optimal share allocation scheme can be designed so that the maximum security can be achieved while at the same time certain (r) redundancy can be provided. Any allocation that conforms to the constraints

$$\begin{cases} N - T + 1 \leq n_i \leq T - 1, & \text{i} = 1,\ldots,\text{m} \\ \sum_{i=1}^{M} n_i = N \end{cases}$$

is an optimal share allocation in terms of security. An optimal share allocation will force the adversary to compromise all the paths to compromise the message, while at the same time, it can tolerate a certain number $(N-T)$ of share lost during the transmission. The optimal share allocation is not unique. Other optimization objectives, such as the minimal delivery cost, balanced bandwidth usage, or maximum reliability, might be set to further optimize the share allocation for other purposes.

V. Multipath Routing and Path Set Optimization

A. Multipath Routing

As we mentioned earlier in the paper, routing in ad hoc networks presents great challenge and the multipath routing, as a promising technique to combat the link instability problem, has been studied in [18-21]. The results show that multipath routing are effective in improving the reliability, fault-tolerance, end-to-end delay for bursty traffic, as well as in achieving load balancing.

For our SPREAD scheme, we need independent paths, more specifically, node disjoint paths, because we are dealing with node compromising problem. Several multipath routing protocols have been proposed in the literature to find node disjoint paths in an ad hoc network [19,21]. Most of the proposed protocols are on-demand, due to the network bandwidth limitation, and use source routing technique to control the disjointness of the paths at source node. For an on-demand routing protocol, whenever it needs a path to a certain destination but does not know one, it starts a route discovery process by broadcasting the route inquiry messages throughout the network, the destination (or intermediate nodes that have a valid route to the destination) will reply by sending back the route. Some type of cache is necessary to store the routes previously found so that the node does not have to perform the costly route discovery for each individual packet. In DSR and the multipath extension of DSR, the route replies back to the source contain the complete node list from the source to the destination. By caching each of these paths separately, a "path cache" organization can be formed. This type of cache organization has been widely used. However, the paths found by this means might not serve our purpose because they may not be the most secure paths. In [22], we designed an alternative cache organization, called a "link cache", in which routes are decomposed into individual links and represented in a unified graph data structure, as illustrated in Figure 3. Given the same amount of route reply information, the routes existing in a path cache can always be found in a link cache. Thus, a link cache has the potential to use the route information more efficiently. We also developed an adaptive stale link removal scheme to work together with the link cache. By using such a link cache, we could separate the routing and the path set optimization. An underlying routing protocol will provide us with a partial view of network topology. However, the optimization of the path set used to deliver the message shares can be done independent of the routing protocols used, solely based on the discovered partial network topology.

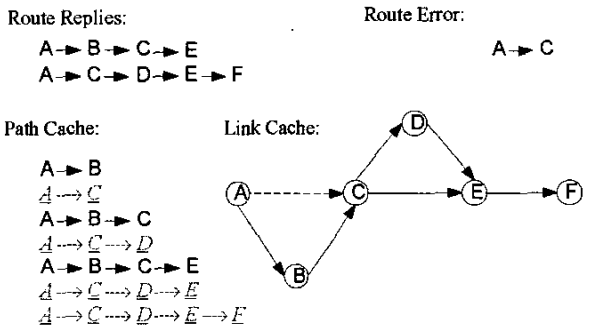

Figure 3 Path cache and link cache

B. Path Set Optimization

In this section, we present a maximal path finding algorithm to optimize the path set for SPREAD based on the partial view of the network topology, which could be discovered by any underlying routing protocol, such as [19-21].

Step 1. Find the first most secure path by modified Dijkstra algorithm, select the path

Step 2. Perform a graph transformation as follows
For each selected path:

a. Replace the links used in the path with directed arcs – for the arc that is directed towards the source, make its cost the negative of the original link cost; make the cost of the arc directed towards the destination infinite (i.e., remove it)

b. Split each node on the selected paths (except the source and destination) into two collocated subnodes; Connect the two subnodes by an arc of cost 0 and directed towards the source node.

c. Replace each external link that is connected to a node in the selected paths by its two component arcs of cost equal to the link cost – let one arc terminate on one subnode and the other one emanate from the other subnode such that along with the zero-cost arc, a cycle does not result.

Step 3. Run the modified Dijkstra algorithm, find the most secure path in the transformed graph; if no more path can be found, go to Step 6.

Step 4. Transform back to the original graph; erase any interlacing edges; group the remaining edges to form the new path set.

Step 5. Compute the overall security achieved by the new path set, and compare it with the path set obtain in the previous iteration. If the new path set does not provide better security than the previous path set, go to Step 6; otherwise record the path set found and go to Step 2.

Step 6. Exit with the recorded path set.

Figure 4 Maximal node disjoint paths finding algorithm

Assume that with probability q_i that node n_i is compromised. Then, the probability that a (s,t) path consisting of node s, n_1, n_2, ..., n_l, t is compromised equals to

$$p = 1-(1-q_1)(1-q_2)\cdots(1-q_l)$$

Since we consider the protection of messages when they are transmitted across the network, we assume that the source and the destination are safe with $q_s=q_d=0$. Note that the probability q_i indicates the security level of node i and could be estimated from the feedback of some security monitoring software and/or hardware such as firewalls and intrusion detection devices. It could also be assigned manually by administrators based on the level of physical protection of nodes, the positions of nodes, or the rankings of nodes, and so on. For example, the headquarters in the rear has the highest security level (lowest q_i value) while the individual scouts penetrating into enemy ground will have lower security level (higher q_i value).

Ideally, given a network, we wish to find an optimal path set, such that the probability P_{msg} is minimized. Recall that $P_{msg}(\underline{n}) = \prod_{i=1}^{M} p_i$. Intuitively, since p_i is a probability which is always less than 1. The more items of p_i, the less the probability, and the better the security. So the goal of our path finding algorithm is to find as many paths as possible while at the same time as secure as possible.

The maximal path finding algorithm used for our SPREAD scheme is modified from the node disjoint shortest pair algorithm [23]. A modified Dijkstra algorithm is used so that negative links are allowed (but no negative loop) in the graph [23]. The modified Dijkstra algorithm modifies the standard Dijkstra algorithm by allowing the permanent labeled node going back to a tentative label when a smaller cost to that node is found. We define the following link cost function to convert the security characteristics into an additive link cost function so that the shortest path algorithm can be readily used in our secure path finding algorithm.

We define the cost function of link between node n_i and n_j as

$$c_{ij} = -\log\sqrt{(1-q_i)(1-q_j)}$$

Then, the cost of the (s,t) path using shortest path algorithm is

$$\begin{aligned} cost(s,t) &= c_{s1} + c_{12} + \cdots + c_{i-1,l} + c_{lt} \\ &= -\log(1-q_1) - \log(1-q_2) - \cdots - \log(1-q_l) \\ &= -\log\{(1-q_1)(1-q_2)\cdots(1-q_l)\} \end{aligned}$$

With the shortest path algorithm, we have

$$\begin{aligned} &cost(s,t) \text{ is minimized iff} \\ &-\log\{(1-q_1)(1-q_2)\cdots(1-q_l)\} \text{ is minimized iff} \\ &(1-q_1)(1-q_2)\cdots(1-q_l) \text{ is maximized iff} \\ &p = 1-(1-q_1)(1-q_2)\cdots(1-q_l) \text{ is minimized.} \end{aligned}$$

So the path found by the shortest path algorithm would be the most secure path when the proposed cost function is used.

The maximal path finding algorithm is an iterative procedure. The most secure path is found first and added to the path set. In a following iteration, the number of paths in the set is augmented by one. Figure 4 summarizes the steps taken to find the maximal number of paths. Each time after a new path is added to the selected path set, a graph transformation is performed, which involves a vertex splitting of the nodes on the selected paths (except the source and destination node). Then, the modified Dijsktra algorithm is executed to find the most secure path in the transformed graph. Then, the split nodes are transformed back to the original one, any interlacing edges are erased, and the remaining edges are grouped to form the new path set.

Figure 5 shows an example of the path finding algorithm. After finding the first two node-disjoint paths, the third one temporarily makes use of the selected nodes but using the link in the reverse direction. After the interlacing removal and regrouping, a path set consisting of 3 paths is found instead of 2.

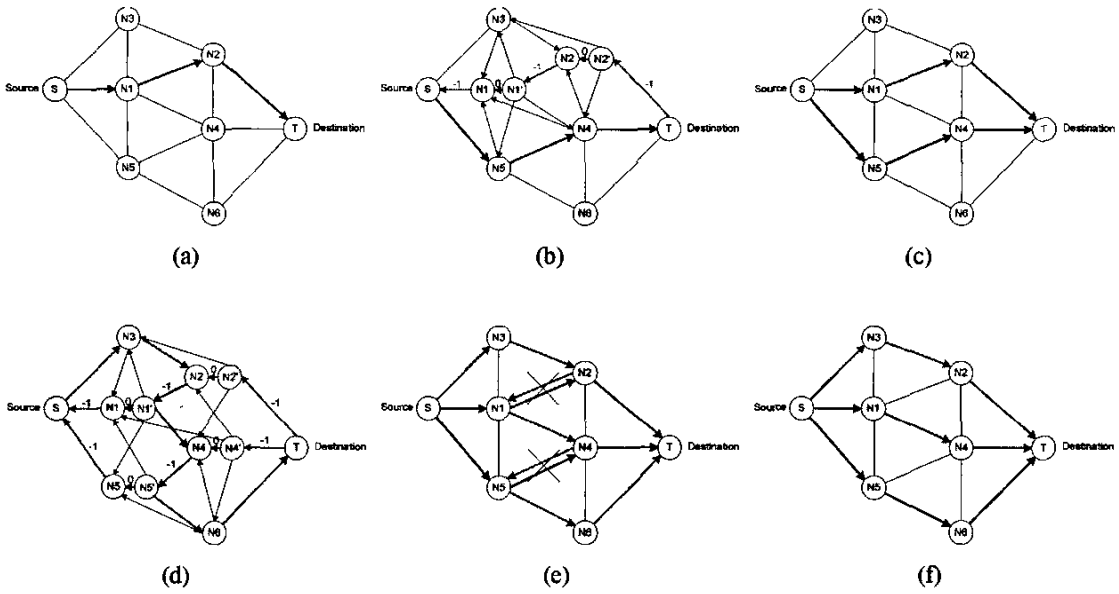

Figure 5 An Illustration of the maximal node disjoint paths algorithm

(a) Iteration 1 – modified Dijkstra algorithm; (b) Iteration 2 – graph transformation and modified Dijkstra algorithm;

(c) Iteration 2 – resulting 2 paths; (d) Iteration 3 – graph transformation and modified Dijkstra algorithm;

(e) Iteration 3 – edge regrouping; (f) Iteration 3 – resulting 3 paths

Because of the regrouping of edges, the augmentation of the path set in each iteration is not simply an addition of a path into the existing path set. All the links might be reshuffled to form the paths in the new path set. So we re-calculate *Pmsg* after each iteration. If *Pmsg* is not getting smaller in the iteration, the path set found in the previous iteration will be taken. The path finding algorithm terminates.

VI. PERFORMANCE EVALUATION

In this section, we present the simulation results to show the performance of SPREAD.

We simulate an ad hoc network with 100 nodes randomly deployed in a 1000m by 1000m area. The transmission range (TR) of each node is set equal in each simulation and varies in different simulations. The simulation results are averaged over 20 randomly deployed networks. To factor out the effect of routing protocols, in the simulation we assume the network topology is known. In each network, we find 1, 2, ..., till maximal node-disjoint paths for each source-destination pair which is at least three hops away. Two sets of simulations are executed. In the first set, each node is assumed equally likely to be compromised with probability q_i=0.152. In the second set of simulation, each node is assigned a probability randomly: 10% of nodes with probability q_i=0.50, 30% of nodes with q_i=0.20, 40% of nodes with q_i=0.10, and 20% of nodes with q_i=0.01. In the first set, all the links are of same cost. In the second set, we use the proposed link cost function to define the link cost based on the node security level (q_i).

Table 1 gives some basic parameters of the network topology of the simulated ad hoc networks. We see that ad hoc networks typically have dense connectivity that allows the exploitation of multipath routing techniques.

TR(m)	200	250
Node degree	10.3	15.4
Diameter	9	6.8

Table 1 Network parameters

Figure 6 shows the probability that multiple paths are found in the simulated network. It is observed that the probability that multiple node disjoint paths exist in an ad hoc network is pretty high. Since our SPREAD scheme depends on the availability of multiple paths, the existence of such multiple paths justifies the feasibility of our scheme.

In fact, if we run the maximal node disjoint paths finding algorithm purely for finding the maximum number of paths without considering the security property of the path set, the number of paths found in both sets would be identical. This implies that the maximum number of paths the algorithm is able to find is independent of the link costs; it solely depends on the network topology although the actual paths found might be different for different link costs. In our simulation, we stop augmenting the path set when the security property of the

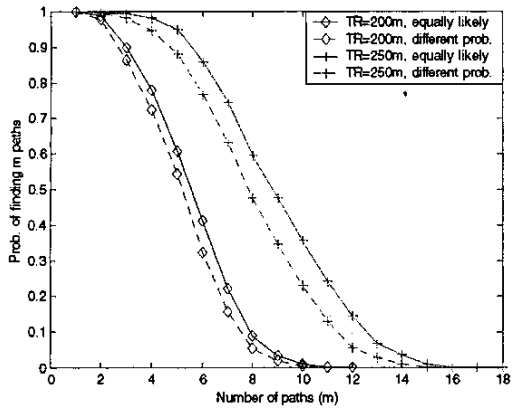

Figure 6 Capability of path finding

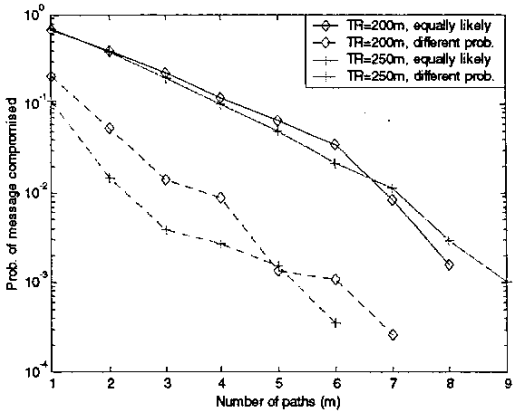

Figure 7 Message compromise probability

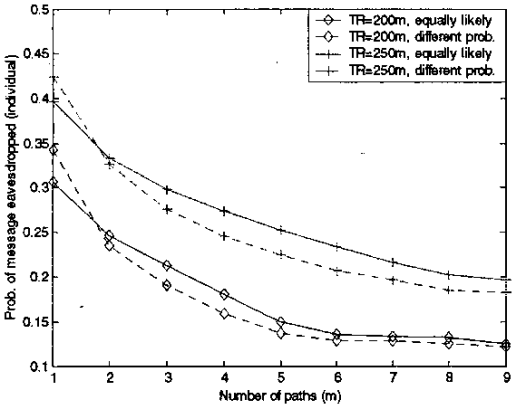

Figure 8 Message eavesdropping probability

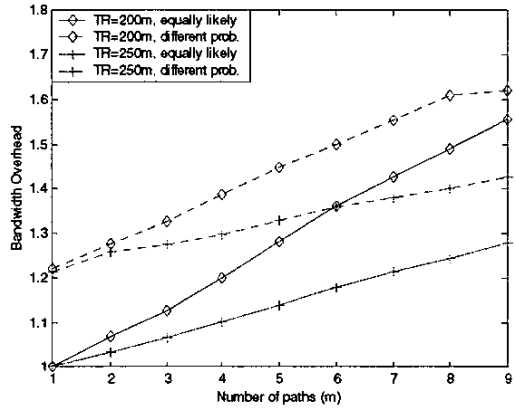

Figure 9 Bandwidth overhead

found paths does not improve. Table 2 gives the probability that the path finding algorithm stops for this reason before finding the maximal number of paths. It indicates that when the nodes are of equal security level, the number of paths plays the most significant role. Basically, the more the paths, the more secure. However, if nodes are of different security levels (probabilities), the security of each path will have more impact on the overall security of the path set. It is not necessarily true that more paths imply higher security. This also explains why in Figure 6 the number of paths selected in simulation set 2 is less than that in simulation set 1.

TR(m)	200	250
Simulation Set 1	0.45%	0.33%
Simulation Set 2	22.7%	38.8%

Table 2 The path finding algorithm stops before finding the maximum number of paths

Figure 7 shows the probability that the message is compromised when multiple paths are used. Here, we consider the case that the message is compromised due to the node compromising. This probability is the probability for colluded attacks. One message is considered compromised when at least one compromised node is located on each of the paths selected to deliver this message. This probability for individual attack is zero because no single node is able to relay all the necessary shares due to the spreading of the shares. Noticing the logarithmic scale of the probability, we observe that the probability drops quickly (actually exponentially fast) with the increase of the number of paths used. This result verifies the effectiveness of our SPREAD idea. We also noticed that when nodes are with different security level, our algorithm tends to select more secure paths that further decrease this probability significantly. The discontinuity of the figure indicates that no compromised message is found in our simulation (over 50,000 messages for TR=200m and over 40,000 messages for TR=250m).

Figure 8 shows the probability that message is eavesdropped when multiple paths are used. Since the wireless channel is a broadcast channel, anyone sits within the transmission range of a transmitting node is able to eavesdrop (overhear) the node's transmission. This figure actually presents the probability for individual attack. The probability

for colluded attack is pretty high (almost 1) because in our simulation, we have about 15 compromised nodes among the totally 100 nodes. It is observed that, with the increase of the number of paths, this probability decreases. However, the decrease becomes less significant when more paths are used. In fact, there is a lower bound of this probability because anyone sits within the transmission range of the source node would be able to overhear all the shares. Of course, this probability is the one that an adversary might overhear a message, it does not mean that the message can be compromised because the message shares are encrypted as well. Again, this verifies that the SPREAD idea makes it harder for an enemy to collect enough data to break the secret.

Figure 9 shows the bandwidth overhead calculated on a per-hop basis when multiple paths are used compared with the single minimum-hop path case. We can see that using multipath does consume more network bandwidth because longer paths are used. However, this is the tradeoff. For security critical applications, the network efficiency might not be a major concern.

VII. CONCLUSION AND DISCUSSION

The basic idea of SPREAD is to distribute the secrecy, first by secret sharing algorithm at the source node and then by multipath routing while shares are delivered across the network, so that in the event that a small number of shares are compromised, the secret as a whole will not be compromised. In this paper, we investigate the major design issues of SPREAD idea. The simulation results show that SPREAD can provide more secure data transmission when messages are transmitted across the insecure network. In addition, we show that a redundant SPREAD scheme can be designed in such a way that a certain degree of reliability can be provided without sacrificing the security. A few remarks are in order. First, the SPREAD scheme considers the security when messages are transmitted across the network, assuming the source and destination are trusted. Secondly, the SPREAD scheme cannot address the confidentiality alone. It only statistically enhances such service. For example, it is still possible for adversaries to compromise all the shares, e.g. by collusion. Finally, the SPREAD can be made adaptive in the sense that the source node could make final decision whether a message is delivered at certain time instant according to the security level and the availability of multiple paths. Moreover, the chosen set of multiple paths may be changed from time to time to avoid any potential capture of those multiple shares by adversaries.

REFERENCE

[1] W. Lou, Y. Fang, "A survey on wireless security in mobile ad hoc networks: challenges and available solutions," book chapter in *Ad Hoc Wireless Networking*, Kluwer, May 2003

[2] L. Zhou and Z. J. Haas, "Securing ad hoc networks," *IEEE Network Magazine*, vol. 13, no. 6, November/December 1999

[3] J. Kong, P. Zerfos, H. Luo, S. Lu and L. Zhang, "Providing robust and ubiquitous security support for manet," *ICNP* 2001

[4] J-P. Hubaux, L. Buttyan and S. Capkun, "The quest for security in mobile ad hoc networks," *MobiHOC*'01, 2001.

[5] Y.-C. Hu, D. B. Johnson and A. Perrig, "SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks," *WMCSA*'02

[6] Y.-C. Hu, A. Perrig and D. B. Johnson, "Ariadne : a secure on-demand routing protocol for ad hoc networks," *MobiCom* 2002, September 2002.

[7] P. Papadimitratos and Z. J. Haas, "Secure routing for mobile ad hoc networks," *CNDS* 2002, San Antonio, TX, January 2002

[8] H. Yang, X. Meng and S. Lu, "Self-organized network-layer security in mobile ad hoc networks," *ACM WiSe*'02, September 2002.

[9] S. Marti, T. Giuli, K. Lai and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," *MobiCom*'00, Boston, MA, USA, August 2000.

[10] L. Buttyan and J.-P. Hubaux, "Enforcing service availability in mobile ad hoc networks," *MobiHOC*'00, 2000

[11] S. Buchegger and J.-Y. Le Boudec, "Performance analysis of the CONFIDENT protocol," *MobiHOC*'02, June 2002.

[12] Y. Zhang, W. Lee and Y. Huang, "Intrusion detection techniques for mobile wireless networks," *ACM Wireless Networks Journal*, Vol. 9, No. 5, Sep 2003

[13] N. Borisov, I. Goldberg, D. Wagner, "Intercepting mobile communications: the insecurity of 802.11", *MobiCom'01*, Rome, Italy, July 2001,

[14] G.J.Simmons, "An Introduction to Shared Secret and/or Shared Control Schemes and The Application", in *Contemporary Cryptology: The Science of Information Integrity*, IEEE Press, 1992, pp.441-497

[15] A. Shamir, "How to Share a Secret", *Communications of the ACM*, 22(11):612-613, Nov 1979

[16] Bruce Schneier, *Applied Cryptography*, 2nd edition, Chapter 23, John Wiley & Sons, 1996

[17] T.-C. Wu, T.-S. Wu, "Cheating detection and cheater identification in secret sharing schemes", *IEE Proc. Comput. Digit. Tech*,. Vol. 142, No.5 September 1995

[18] A. Tsirigos, Z.J. Haas, "Multipath routing in the presence of frequent topological changes", *IEEE Communication Magazine*, Nov 2001

[19] S-J. Lee, M. Gerla, "Split multipath routing with maximally disjoint paths in ad hoc networks", *ICC*'01

[20] M.R. Pearlman, Z.J. Haas, P. Sholander, S. S. Tabrizi, "On the impact of alternate path routing for load balancing in mobile ad hoc networks", *MobiHOC*, 2000

[21] K. Wu, J. Harms, "Performance study of a multipath routing method for wireless mobile ad hoc networks", *9th international symposium on modeling, analysis and simulation of computer and telecommunication system*, 2001

[22] W. Lou, Y. Fang, "Predictive caching strategy for on-demand routing protocols in ad hoc networks", *Wireless Networks*, vol.8, issue 6, Nov 2002

[23] R. Bhandari, *Survivable Networks – Algorithms for diverse routing*, Kluwer Academic Publisher, 1999

[24] S. Jiang, N.H. Vaidya, W. Zhao, "A dynamic mix method for wireless ad hoc networks," *Milcom*'01, pp. 873-877, McLean, VA, Oct 2001

[25] S. Jiang, N. H. Vaidya, W. Zhao, "Preventing traffic analysis in packet radio networks," *DARPA Information Surviviability Conference & Exposition II (DISCEX'01)*, vol.2, 2001

[26] T. Cormen, C. Leiserson, R. Rivest, *Introduction to algorithms*, MIT Press, 1990