HADOF: Defense Against Routing Disruptions in Mobile Ad Hoc Networks

Wei Yu*, Yan Sun† and K. J. Ray Liu*
*Department of Electrical and Computer Engineering,
University of Maryland, College Park, MD 20742
Email: weiyu, kjrliu@umd.edu
†Department of Electrical and Computer Engineering,
University of Rhode Island, 4 East Alumni Ave, Kingston, RI 02881
Email: yansun@ele.uri.edu

***Abstract*— HADOF is a set of mechanisms to protect mobile ad hoc networks against routing disruption attacks launched by inside attackers. First, each node launches a *route traffic observer* to monitor the behavior of each valid route in its route cache, and to collect the packet forwarding statistics submitted by the nodes on this route. Since malicious nodes may submit false reports, each node also keeps *cheating records* for other nodes. If a node is detected as dishonest, this node will be excluded from future routes, and the other nodes will stop forwarding packets for it. Third, each node will try to build *friendship* with other nodes to speed up malicious node detection. *Route diversity* will be explored by each to discover multiple routes to the destination, which can increase the chance of defeating malicious nodes who aim to prevent good routes from being discovered. In addition, *adaptive route rediscovery* will be applied to determine when new routes should be discovered. HADOF can handle various attacks and introduces little overhead to the existing protocols. Both analysis and simulation studies have confirmed the effectiveness of HADOF.**

I. Introduction and Background

A *mobile ad hoc network* is a group of mobile nodes without requiring centralized administration or fixed network infrastructure, in which nodes can communicate with other nodes out of their direct transmission ranges through cooperatively forwarding packets for each other. One underlying assumption is that they communicate through wireless connections. Since ad hoc networks can be easily and inexpensively set up as needed, they have a wide range of applications, such as military exercises, disaster rescue, and mine site operations.

Before mobile ad hoc networks can be successfully deployed, security concerns must be addressed first [1]–[6]. However, due to mobility and the ad hoc nature, protecting mobile ad hoc networks is particularly hard: the wireless links are usually fragile with high link broken ratio; nodes lack of enough physical protection can be easily captured, compromised and hijacked; the sporadic nature of connectivity and the dynamically changing topology may cause frequent routes update; the lack of centralized monitoring or management points further deteriorates the situations. Attackers can easily launch a variety of attacks ranging from passive eavesdropping to active interfering.

During the last decade, extensive studies have been conducted on routing in mobile ad hoc networks, and have resulted in several mature routing protocols [7]–[10]. However, in order to work properly, these protocols need trusted working environments which are not always available. In many situations, the environments may be adversarial. For example, some nodes may be selfish, malicious, or compromised by attackers. In the literature, many schemes have been proposed to secure ad hoc network routing protocols. However, most schemes focus on preventing attackers from entering the network through secure key distribution/authentication and secure neighbor discovery, such as [5], [6], [11]–[15]. These schemes are not effective in the situations where the malicious nodes have entered the network, or some nodes in the network have been compromised.

In this paper, we consider the scenario that all nodes in the network belong to the same authority and pursue common goals, and propose a set of integrated mechanisms to defend against routing disruption attacks launched by inside attackers. Under this scenario, we can categorize the nodes into two classes: *good* and *malicious*. Good nodes will try their best to forward packets for others, that is, they are fully cooperative, while malicious nodes may manipulate routing messages, (selectively) drop data packets, and frame up other good nodes, with the objective of degrading the network performance and consuming valuable network resources.

We use "HADOF" (the acronym of Honesty, Adaptivity, Diversity, Observer, and Friendship) to refer to the set of proposed mechanisms to defend against routing disruption attacks, which in brief works in the following way. Each node launches a *route traffic observer* to monitor the behavior of each valid route in its route cache, and to collect the packet forwarding statistics submitted by the nodes on those routes. Since the proposed reports submission mechanism does not rely on monitoring neighbors' forwarding activities, it is much more energy efficient than the watchdog mechanism. Since malicious nodes may submit false reports, each node also keeps a *cheating record* database that indicates whether some nodes are dishonest or have been suspected as dishonest. If a node is detected as cheating, then this node will be excluded from future routes. Furthermore, other nodes will stop forwarding packets originated from this cheating node as punishment. In many situations, if malicious nodes are smart, it is hard to find concrete evidence to prove that they are cheating. To address this issue and speed up the malicious node detection, each node can also build *friendship* with other nodes that it trusts.

The next two mechanisms are to explore the *route diversity* and the dynamic nature of mobile ad hoc networks. Since there may exist more than one route from a source to a destination, the source can try to find multiple routes to the destination, and adaptively determine which route should be used. By exploring *route diversity*, we expect that the frequency of route discovery can be reduced, and the case that malicious nodes try to prevent good routes from being discovered can be better handled. Due to node mobility and dynamically changing traffic pattern, a route which was good before may not be necessarily good currently. Instead of waiting for all the routes in the route cache becoming invalid, *adaptive route rediscovery* tries to trade the route discovery overhead with the route quality through dynamically determining when new route discoveries should be initiated.

The rest of this paper is organized as follows. Section II presents the related work. Section III outlines our assumptions. Section IV describes HADOF in detail. Section V analyzes the security of HADOF. Simulation methodology and performance metrics are described in Section VI. Section VII presents the simulation results and performance evaluation. Finally Section VIII concludes this paper.

II. Related Work

To secure the ad hoc network, the first step is to prevent attackers from entering the network through secure key distribution/authentication and secure neighbor discovery, such as [5], [6], [11]–[15]. However, these schemes cannot work well when attackers have entered the network. To defend against inside attackers, schemes based on monitoring packet forwarding activities have been shown to be promising solutions [2], [3], [16]–[19].

Papadimitratos and Haas [11] proposed a secure routing protocol for mobile ad hoc networks that guarantees the discovery of correct connectivity information over an unknown network in the presence of malicious nodes. However, it is still vulnerable to several attacks, such as rushing attacks and wormhole. Sanzgiri et al [12] considered a scenario that nodes authenticate routing information coming from their neighbors while not all the nodes on the route will be authenticated by the sender and the receiver. However, this scheme cannot handle compromised nodes. Hu, Perrig and Johnson [5] proposed Ariadne, a secure on-demand ad hoc network routing protocol, which can prevent attackers or compromised nodes from tampering with uncompromised routes that (only) consist of uncompromised nodes. In [6], [14], they described how to defend against rushing attacks through secure neighbor discovery and how to apply packet leashes to defend against wormhole attacks. Later, Capkun and Hubaux investigated secure routing in ad hoc networks in which security associations exist only between a subset of all pairs of nodes [20]. However, none of the above schemes can handle inside attackers well.

To defend against attackers that have entered the network and can be on discovered route, reputation system based on monitoring traffics in the network can be used. Initial work using these mechanisms is proposed by Marti et al [3]. In their paper, they considered the case that nodes agree to forward packets but fail to do so, and proposed two tools that can be applied upon source routing protocols: *watchdog* and *pathrater*. However, their scheme suffers many problems. First, watchdog requires the *promiscuous* mode of the wireless interface which is not always available. Second, since nodes using watchdog have to keep receiving packets from their neighbors, the network capacity may be reduced and a lot of energy will be wasted. Third, the watchdog cannot distinguish malicious behavior from misbehavior caused by temporary network malfunction, such as collision or network congestion. Therefore, watchdog suffers a lot of false alarms. Fourth, the pathrater defines the route quality as the average reputation of the nodes on the route, which in general is not the best metric. Another major problem, which has also been reckoned in their paper [3], is that their schemes are not collusion resistant, and also vulnerable to the attacks that aim to frame up innocent nodes. In [16], [21], the authors extended the ideas in [3], and allowed the reputation to propagate throughout the network. However, since they still rely on watchdog, schemes in [16], [21] also suffer the same types of problems as [3]. Furthermore, once reputation can propagate, selfish or malicious nodes can collude to frame other nodes.

In [17]–[19], the authors consider the scenario that nodes are selfish, and may be unwilling to forward packet on the benefits of other nodes. They proposed schemes to stimulate cooperation among nodes based on credit system or game theory. However, those schemes cannot handle malicious nodes in the network whose objective is to maximize the damage they cause to the network, instead of maximizing their own benefits obtained from the network.

III. Assumptions

A. *Physical and MAC Layers Assumptions*

We assume that nodes can move freely inside a certain area, and communicate with each other through wireless connections. We assume that the links are bidirectional, but not necessarily be symmetric. That is, if node A is capable of transmitting data to node B directly, then node B is also capable of transmitting data to A directly, though the two directions may have different bandwidths. This assumption holds in most wireless communication systems. In this paper, *neighbor* refers to that two nodes are in each other's transmission range, and can directly communicate with each other. We assume that the MAC layer protocol supports *acknowledgement* (ACK) mechanism. That is, if node A has sent a packet to node B, and B has successfully received it, then node B needs to notify A of the reception immediately.

B. Dynamic Source Routing

We adopt DSR [22] as the underlying routing protocol, which is an *on-demand source routing* protocol for mobile ad hoc networks. *On-demand routing* means that routes are discovered at the time when a source wishes to send a packet to a destination and no existing routes are known by the source. *Source routing* means that when sending a packet, the source lists in the packet header the complete sequence of nodes through which the packet is to traverse. There are two basic operations in DSR: *route discovery* and *route maintenance*.

In DSR, when a source S wishes to send packets to a destination D but does not know any routes to D, S will initiate a route discovery by broadcasting a ROUTE REQUEST packet, specifying the destination D and a unique ID. When a node receives a ROUTE REQUEST not targeting on it, it first checks whether this request has been seen before. If yes, it will discard this packet, otherwise, it will append its own address to this REQUEST and rebroadcasts it. When the REQUEST arrives at D, D then sends a ROUTE REPLY packet back to S, including the list of accumulated addresses (nodes). A source may receive multiple ROUTE REPLYs from the destination, and can cache these routes in its Route Cache.

Route Maintenance handles link breakages. If a node detects the link to the next hop is broken when it tries to send a packet, it will send a ROUTE ERROR packet back to the source to notify this link breakage. The source then removes the route having this broken link from its Route Cache. For subsequent packets to the destination, the source will choose another route in its Route Cache, or will initiate a new Route Discovery when no route exists.

C. Attacks and Node Behavior Assumptions

Since we consider the scenario that all nodes belong to the same authority and pursue common goals, without loss of generality, we assume that nodes are either good or malicious. Malicious nodes can launch a variety of attacks in almost all layers of mobile ad hoc networks. For example, an attacker can use a jammer to interfere the transmission in the physical layer. It can also attack MAC layer by exploring the vulnerability of existing protocols [5], [23]. Defense against attacks launched in physical and MAC layers is out of this paper's scope, we will focus on security issues in network layer.

Two types of attacks have been widely used to attack the network layer in ad hoc networks: *resource consumption* and *routing disruption* [5]. Resource consumption attacks refer to that the attackers inject extra packets into the network in attempt to consume valuable network resources. Routing disruption attacks, which are the focus of this paper, refer to that attackers attempt to cause legitimate data packets to be routed in dysfunctional ways, and consequently cause packets to be dropped or extra network resources to be consumed.

Some examples of routing disruption attacks are: *black hole, gray hole, wormhole, rushing attack,* and *frame-up* [5], [6], [14], [15]. The attackers can create a wormhole through collusion in the network to short circuit the normal flow of routing packets [14], or can apply rushing attack to disseminate ROUTE REQUEST quickly through the network [6]. By creating a wormhole or applying rushing attacks, the attackers can prevent good routes from being discovered, and increase their chance of being on discovered routes. Once an attacker is on certain route, it can create a black hole by dropping all the packets passing through it, or create a gray hole by selectively dropping some packets passing through it. If the protocols have the mechanism to track malicious behavior, an attacker can also frame up good nodes. In addition, an attacker can modify the packets passing through it, which has similar effects as dropping packets, but a little bit more severe because more network resources will be wasted when the following nodes on this route continue forwarding this corrupted packet.

D. Security and Key Setup Assumptions

We assume that each node has a public/private key pair, and there is a tight coupling between a node's public key and its address, such as deriving the IP address of the node from its public key using the methods described in [24], [25]. We also assume that a node can know or authenticate other nodes' public keys, but no node will disclose its private key to others unless it has been compromised. We do not assume that nodes trust each other, since some nodes may be malicious or be compromised. But if there exists some trust relationship, we will take advantage of it.

We assume that all the nodes in the network are legitimate, that is, they have been authorized to enter the network, and have certified public keys. Attackers without certified public keys can be excluded from the routes through necessary key authentication. We assume that if two nodes set up communication between them, they must have built a trust relationship, and they trust the information reported by each other. This trustiness can be built outside of the context of the network (e.g. friends), or through certain authentication mechanisms after the network has been set up.

To keep the confidentiality and integrity of the transmitted content, the sources encrypt and sign each packet sent by them. Since the source and the destination trust each other, they can create a temporarily shared secret key to encrypt the communication and use an efficient hash chain to authenticate the communication [26]. For each intermediate node on the route, authentication is activated only when the destination has detected abnormal corruptions in data packets, which means that some malicious nodes are on the route.

IV. Descriptions of HADOF

Before describing the detail of HADOF, we first introduce some notations listed in Table I. In this paper, we use S to denote the source and use D to denote the destination. Also, *traffic pair* refers to a pair of nodes (S, D) communicating with each other directly or indirectly. Based on our assumption, S and D trust each other.

A. Route Traffic Observer

Each node launches a route traffic observer (RTO) to periodically collect the traffic statistics of each valid route

TABLE I

NOTATIONS FOR EACH TRAFFIC PAIR

S	The source
D	The destination
R_i	The i^{th} available route from S to D in S's Route Cache.
L_i	Number of intermediate nodes on the route R_i.
$FN_{cur}(A,S,R_i)$	The number of packets originated from S and forwarded by A via route R_i in this interval.
$RN_{cur}(A,S,R_i)$	The number of packets originated from S and received by A via route R_i in this interval.
$FN_{tot}(A,S)$	The total number of packets originated from S and forwarded by A.
$RN_{tot}(A,S)$	The total number of packets originated from S and received by A.
$P_{cur}(A,S,R_i)$	$\frac{FN_{cur}(A,S,R_i)}{RN_{cur}(A,S,R_i)}$, the packet delivery ratio of A for S via route R_i in this interval.
$P_{avg}(A,S)$	$\frac{FN_{tot}(A,S)}{RN_{tot}(A,S)}$, the overall packet delivery ratio of A for S.
H(A, S)	A's honesty score in S's point of view.

in its route cache. A *valid route* refers to a route without receiving any link breakage report. At the end of each predetermined interval, the RTO examines each traffic pair (S, D) and each route R_i to D in S's route cache that has been used in this interval. In particular, the RTO collects $RN_{cur}(A,S,R_i)$ and $FN_{cur}(A,S,R_i)$ reported by each node A on this route. This can be done by letting D periodically send back an agent packet to collect such information, or letting each node periodically report its own statistics to S. For each node A known by S, S's RTO also keeps a record of $RN_{tot}(A,S)$ and $FN_{tot}(A,S)$. To reduce overhead, the RTO of S will request reports from the intermediate nodes of a route only when S realizes that some packets have been dropped on this route in this interval based on the reports submitted by D.

After the RTO has finished collecting packet forwarding statistics, it recalculates the expected quality of those routes that have been used in this interval. In general, route quality is affected by many factors, such as the forwarding history of each node on this route, the hop number, the current traffic load and traffic distributions, etc. Before defining the expected route quality metric, we first define the expected packet delivery ratio of A for S, $P(A,S)$, as follows:

$$P(A,S) = (1-\beta)P_{avg}(A,S) + \beta P_{cur}(A,S,R_i). \quad (1)$$

That is, $P(A,S)$ is a weighted average of $P_{cur}(A,S,R_i)$ and $P_{avg}(A,S)$, and β is used to adjust the weight between them. The intuition behind this is that when predicting a node's future performance, we consider not only this node's current performance, but also its past history. It is easy to see that the range of $P(A,S)$ is between 0 and 1. In HADOF, the expected route quality $Q(R_i)$ for route R_i is calculated as follows:

$$Q(R_i) = \prod_{A\in R_i} P(A,S) * H(A,S) - \lambda * L_i, \quad (2)$$

where $H(A,S)$ is A's honesty score in S's view indicating the suspicious degree of A. $H(A,S)$ ranges from 0 to 1, with 1 indicating being honest and 0 indicating being malicious. The criteria of calculating $H(A,S)$ is presented in Section IV-B. In (2), a small positive value λ is introduced to account for the effects of hop number. As a result, if two routes have the same value for the product in the right hand of (2), the route with less hops is favored. The intuition behind this is that we expect a route with less hops having less influence on the network. In HADOF, the values of $P(S,S)$, $P(D,S)$, $H(S,S)$ will always be 1, since a source trusts itself and the corresponding destination.

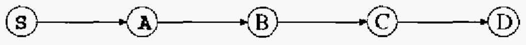

Fig. 1. Detection of Cheating Behavior

B. Cheating Record and Honesty Score

When S's RTO collects packet forwarding statistics, malicious nodes may submit false reports. For example, it may report a smaller RN value and a larger FN value to cheat the source and frame up its neighbors. To address this, each source keeps a *Cheating Record* (CR) database to track whether some nodes have ever submitted or been suspected to submit false reports to it. S will mark a node as malicious if S has enough evidence to believe that the node has submitted false reports.

Initially, S assumes that all nodes are honest, and sets the honesty score $H(A,S)$ for each node A to be 1. After each report collection which is performed periodically, S will try to detect whether some nodes on a route are cheating through checking the *consistence* of the received reports. For example, in Fig. 1, both A and B are on the route R with A being ahead of B. A cheating behavior is detected if S finds that $FN_{cur}(A,S,R) \neq RN_{cur}(B,S,R)$. If one of them (A or B) is trusted by S (e.g., that node is S itself or D), then the other node can be marked as cheating by S, and the honesty score of the cheating node will be set to be 0. Otherwise, S can only suspect that at least one of them is cheating. In this case, the honesty scores of both nodes are updated as

$$H(A,S) = \alpha H(A,S) \quad (3)$$

$$H(B,S) = \alpha H(B,S) \quad (4)$$

where $0 < \alpha < 1$ is used to indicate the punishment degree. In addition, if $FN_{cur}(A,S,R) > RN_{cur}(B,S,R)$, S will reset the value of $FN_{cur}(A,S,R)$ using $RN_{cur}(B,S,R)$, reset the value of $RN_{cur}(B,S,R)$ using $FN_{cur}(B,S,R)$, and recalculate $FN_{tot}(A,S)$ and $RN_{tot}(B,S)$ using the updated values. Since $FN_{cur}(A,S,R) < RN_{cur}(B,S,R)$ does not make sense, we will not consider this situation.

Once a node has been detected as cheating, punishment should be applied on it. In HADOF, when S detects a node B being malicious, S will put B in its blacklist (equivalent to set $H(B,S)$ to be 0), stop forwarding any packets originated from B, and refuse to be on the same route as B in the future.

Next we introduce a mechanism to recover the honesty scores of nodes that have been framed up by malicious nodes. We still use the example in Fig. 1 to illustrate this mechanism. When S finds the reports submitted by A and B conflicting with each other, that is, $FN_{cur}(A,S,R) > RN_{cur}(B,S,R)$, besides decreasing A's honesty score, S will also increase the number of possible frame-up attacks launched by B to A, and records the difference between $FN_{cur}(A, S, R)$ and $RN_{cur}(B, S, R)$. Similarly, S does the same thing to B. If later S detects that B is a cheating node, S will check how many nodes have ever been framed up by B and for each node how many times. Assume A has been framed up by B m times, S will recover A's honesty score as follows:

$$H(A, S) = \frac{H(A, S)}{\alpha^m}, \tag{5}$$

which is always bounded by 1. Meanwhile, S also needs to increase $FN_{tot}(A, S)$ or decrease $RN_{tot}(A, S)$ to recover the inaccuracy caused by frame-up attacks launched by B.

C. Friendship

Since a malicious node knows the source and destination of each route that it is on, to avoid being detected, it will only frame up its neighbors who are neither the source nor the destination. Therefore, even when the CR database has been activated, the malicious nodes can only be suspected, but cannot be proved as cheating by the source. This can be mitigated by taking advantage of the existing trustiness relationship. Each node maintains a private list of trust nodes that it considers to be honest. Now if B submits false reports to S to frames up A, while S trusts A, B can be detected by S immediately, and $H(B, S)$ will be set to be 0.

D. Route Diversity

Since there may exist more than one route from a source to a destination, it is usually beneficial to discover multiple routes. In [27], [28], the authors have shown that using multiple routes can reduce the route discovery frequency. In this paper, we investigate how route diversity can be used to defend against routing disruption attacks. In DSR, discovering multiple routes from a source to a destination is straight-forward. Let $MaxRouteNum$ be the maximum number of ROUTE REPLYs that the destination can send back for the route requests with the same request ID. By varying $MaxRouteNum$, we can discover different number of routes. By exploring route diversity, we have better chance to defeat attackers who aim to prevent good routes from being found. Meanwhile, since there may exist multiple routes, the source can always use the route with the best quality according to certain criteria.

When a new route R is discovered, for each node A on this route, $FN_{cur}(A, S, R)$ and $RN_{cur}(A, S, R)$ should be initialized to be 0. Since this route has never been used before, its expected quality can be calculated as

$$Q(R) = \prod_{A \in R} P_{avg}(A, S) * H(A, S) - \lambda * L. \tag{6}$$

The difference between (6) and (2) lies in that only nodes' past history on the route are used in (6).

Since there may exist multiple routes to D in S's Route Cache, S needs to decide which route should be used. One possible way is to always use the one with the best expected quality. However, this may not be the best choice. For example,

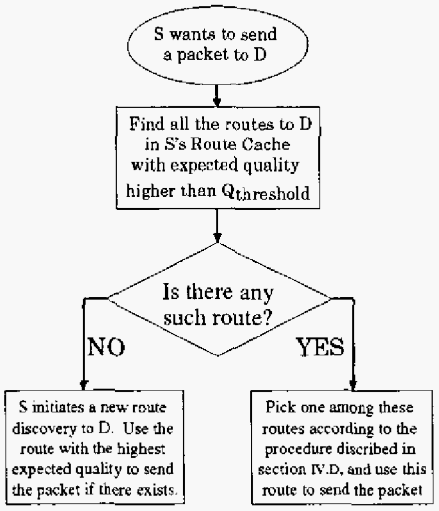
S wants to send a packet to D
Find all the routes to D in S's Route Cache with expected quality higher than Qthreshold
Is there any such route?
NO
YES
S initiates a new route discovery to D. Use the route with the highest expected quality to send the packet if there exists.
Pick one among these routes according to the procedure discribed in section IV.D, and use this route to send the packet

Fig. 2. Packet Sending Procedure

the quality of a route may degrade dramatically after being injected into a lot of traffics. In this paper, the following procedure is used to distribute traffics among multiple routes, and adaptively determine which route should be used. Let $Q_{threshold}$ be a pre-determined quality threshold, and let $R_1, \ldots, R_K$ be the K routes with the expected quality higher than $Q_{threshold}$. Once S wants to send a packet to D, S randomly picks a route among them. The probability that route R_i $(1 \le i \le K)$ will be picked is determined as

$$Prob(R_i) = \frac{Q(R_i)}{Q(R_1) + \cdots + Q(R_K)} \tag{7}$$

If no route has expected quality higher than $Q_{threshold}$, the route with the highest expected quality will be selected.

E. Adaptive Route Rediscovery

Due to mobility and the dynamically changing traffic patterns, some routes may become invalid after a while, or their quality may change. Usually, a new route discovery should be initiated by S when there exist no available routes from S to D. In this paper, we use an *adaptive route rediscovery* mechanism to determine when a new route discovery should be initiated: if S wants to send packets to D, and there exist no routes to D with quality higher than $Q_{threshold}$ in S's route cache, S then initiates a new route discovery.

F. Implementation of HADOF

We have implemented HADOF upon DSR, which includes two major procedures: packet sending procedure and traffic statistics and cheating records updating procedure. The packet sending procedure is described in Fig. 2. When S wants to send a packet to D, S first checks its route cache to find whether there exist valid routes to D. If there exist no valid routes, S initiates a new route discovery with the destination being D. If there exist some valid routes, but none has expected quality higher than $Q_{threshold}$, S picks the route with the best expected quality, and initiates a new route discovery. Otherwise, S randomly picks one route according to the procedure described in Section IV-D.

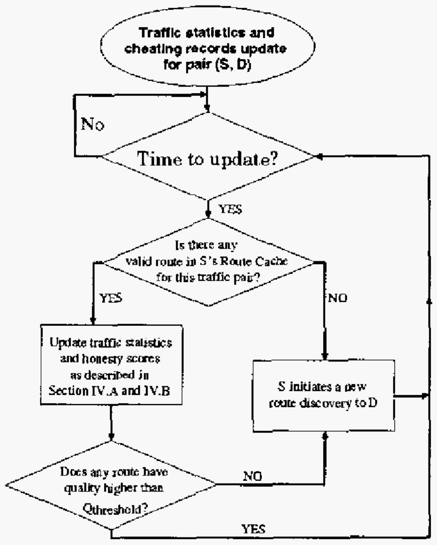
Traffic statistics and cheating records update for pair (S, D)
No
Time to update?
YES
Is there any valid route in S's Route Cache for this traffic pair?
YES
NO
Update traffic statistics and honesty scores as described in Section IV.A and IV.B
S initiates a new route discovery to D
Does any route have quality higher than Qthreshold?
NO
YES

Fig. 3. Updating/Maintaining Traffic Statistics and Cheating Records

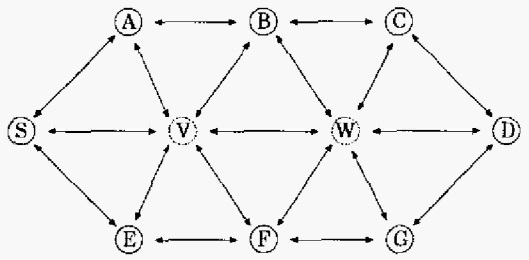

Fig. 4. A simple example

The procedure for updating/maintaining traffic statistics and cheating records is described in Fig. 3. The source S periodically calls this procedure to collect traffic statistics for each route that has been used in this interval. Based on the mechanisms described in Section IV-A and Section IV-B, S updates the expected route quality and cheating records. If necessary, a new route discovery should be initiated when certain conditions are satisfied, as described in Section IV-E.

V. SECURITY OF HADOF

This section analyzes the security aspects of HADOF in terms of defending against various routing disruption attacks. Throughout this section, we will use Fig. 4 as a simple example to illustrate different situations.

Black Hole and Gray Hole Attacks: In HADOF, the source can quickly detect a gray hole or black hole based on the reports it has collected and past records of each node. Without loss of generality, assume B has created a gray hole on route "SABCD" in Fig. 4. Based on the reports submitted by A, B, C, and D, S can know that some of them have dropped packets. Node B can be detected as creating a black/gray hole by S if $P_{avg}(B,S)$ and $P_{cur}(B,S,\text{"}SABCD\text{"})$ are low, and $RN(B,S)$ value is larger than a pre-defined threshold, where a relatively large $RN(A,S)$ is used to make sure that this is not transient phenomenon.

Frame-up Attacks without Collusion: Besides dropping packets, a malicious node can also submit false reports to cheat the source and frame up its neighbors. For example, on the route "SABCD", if B is malicious, B can submit a smaller RN value to frame up A and a larger FN number to frame up C. In HADOF, a source can detect frame-up attacks through checking the consistence of the reports it has collected. We still use the route "SABCD" as an example, and assume that the malicious nodes work alone. If B has reported a larger $FN_{cur}(B,S,R)$ to frame up C, S can detect this by finding $FN_{cur}(B,S,R) > RN_{cur}(C,S,R)$ where R denotes the route "SABCD". Now we analyze the possible consequence of this frame-up. First, B cannot increase its $P_{cur}(B,S,R)$ and $P(B,S)$ since S will use $RN_{cur}(C,S,R)$ to replace $FN_{cur}(B,S,R)$. Second, B can only make S suspect C, but cannot make S believe that C is malicious. Third, if C is trusted by S, then B can be detected immediately, and will be excluded from any route originated from S in the future. Fourth, B's own honesty score will be decreased. Therefore, B can cause only limited damage by framing up others, but has to take the risk of being detected as malicious, especially when friendship has been introduced.

Frame-up Attacks with Collusion: Next we show that collusion in frame-up attacks cannot further deteriorate the situation. We still use the route "SABCD" as an example. In the first case, the malicious nodes are neighbors of each other. For example, B and C. Without loss of generality, we can view them as one node B′, with $RN_{cur}(B',S,R) = RN_{cur}(B,S,R)$ and $FN_{cur}(B',S,R) = FN_{cur}(C,S,R)$. That is, B and C together have the same effects as B′ working alone, and the only difference is that they can release one node by sacrificing of the other one, that is, by letting it take all the responsibilities. In the second case, the malicious nodes are not neighbors of each other. For example, A and C are malicious and work together to frame up B. It can be seen that the effect of A and C jointly framing up B is the same as that of A and C framing up B independently. Thus we conclude that in HADOF collusion cannot further improve the capability of frame-up attacks.

Rushing Attacks: In rushing attacks, an attacker can increase its chance of being on the route by disseminating ROUTE REQUESTs quickly and suppressing any later legitimate ROUTE REQUESTs [6]. For example, in Fig. 4, if V can broadcast the ROUTE REQUESTs originated from S more quickly than A and E, then all the ROUTE REQUESTs broadcasted by A and E will be ignored. The direct consequence is that V appears on all the routes returned by D. Later V can drop packets and frame up its neighbors. Now we show how rushing attacks can be handled using HADOF. If S detects that no routes to D in its route cache work well, it will check whether these routes share a critical node where all packets from S to D pass through it. In this example, the critical node is V. If V has low $P_{avg}(V,S)$ value and low $H(V,S)$, S has reasons to suspect that V has launched rushing attacks. S then initiates a new route discovery and explicitly exclude V from being on discovered routes.

Wormhole Attacks: A pair of attackers can create a wormhole in the network via a private network connection to disrupt routing by short circuiting the normal flow of routing packets [14]. For example, in Fig. 4, if W and V are attackers and have created a wormhole between them, V can quickly forward any ROUTE REQUESTs it receives to W, and let W broadcast them. There are two variations based on whether V and W append their addresses to the REQUESTS. If they append their addresses, they are similar as rushing attackers, and the method discussed above can be used to handle them. The situation becomes more severe if they do not append their addresses. For example, W and V can make S believe that D is its neighbor, and later V can create a black hole to drop all the packets originated from S and targeting D. In HADOF, if S finds no routes returned by D are valid, or S has not received any acknowledgement from D, S has reason to suspect that there exists a wormhole between S and D. S then activates an neighbor discovery techniques such as in [6], [14] to prevent attackers from creating wormholes.

In summary, HADOF can handle various routing disruption attacks very well, such as gray hole, black hole, frame-up, and rushing attacks, and wormhole attacks, and is collusion-resistant.

VI. Simulation Methodology

A. Simulator and Simulation Parameters

In our simulations, we use an event-driven simulator to simulate mobile ad hoc networks. The physical layer assumes a fixed transmission range model, where two nodes can directly communicate with each other successfully only if they are in each other's transmission range. The MAC layer protocol simulates the IEEE 802.11 Distributed Coordination Function (DCF) [29]. DSR is used as the underlying routing protocol.

The simulation parameters are listed in Table II. We use a rectangular space of size 1000m × 1000m. The total number of nodes is 100, and the maximum transmission range is 250m. There are 20 traffic pairs randomly generated for each simulation. For each traffic pair, the packet arrival is modelled as a Poisson process, and the average packet inter-arrival time is uniformly chosen between 0.04 and 0.2 second, such that each traffic pair injects different traffic load to the network, which we expect could better simulate the reality than using the same inter-arrival time for all the traffic pairs. The size of each data packet after encryption is 512 bytes, and the link bandwidth is 1 Mbps. Among the 100 nodes, we vary the total number of malicious nodes from 5 to 20. In our implementation, the malicious nodes will submit false reports only when it has dropped packets and this false reports cannot be detected easily. For example, a malicious node will not submit false reports to frame up the sources or the destinations.

In the simulations, each node moves randomly according to a *random waypoint* model [22]: a node starts at a random position, waits for a duration called the *pause time* that is modeled as a random variable with exponential distribution, then randomly chooses a new location and moves towards the new location with a velocity uniformly chosen between 0 and v_{max}. When it arrives at the new location, it waits for another random pause time and repeats the process. In the simulations,

TABLE II
Simulation Parameters

Number of nodes	100
Maximum Velocity (v_{max})	20 m/s
Dimensions of Space	1000m × 1000m
Maximum Transmission Range	250 m
Number of Traffic Pairs	20
Average Packet Inter-Arrival Time	0.04-0.2 second
Data Packet Payload Size	512 bytes
Link Bandwidth	1 Mbps
MaxRouteNum	5
MaxHopNum	10
α	0.9
β	0.6
λ	0.02
$Q_{threshold}$	0.8
Update Interval	1 second

two sets of average pause time are used: 0 second and 50 seconds. The average pause time of 0 second represents a high mobility case where nodes keep moving, while the average pause time of 50 seconds represents a moderate mobility case.

B. Baseline and Watchdog

In our simulations, the baseline system is implemented as follows: the basic DSR described in Section III-B is used, and for each route discovery, only one route is returned. No adaptive route rediscovery is used, and no malicious node detection mechanisms are applied. It is expected that the baseline system will perform badly in most situations.

For comparison, the mechanism proposed in [3] has also been implemented, which includes two major components: watchdog and pathrater. To make watchdog work properly, we have modified the MAC layer protocol to ensure the following property: after node B receives a packet from node A and needs to forward this packet to node C, B can start the forwarding only if both A and C are idle and ready to receive packets. When using watchdog, a node will report to the source when another node refuses to forward more than certain number of packets for it. In our implementation, we set the threshold to be 5. In addition, each route discovery initiated by source S will return at most 5 routes, and the route with the best quality (calculated using pathrater) will be used. When the route in use becomes invalid due to link breaks, instead of using the routes in S's Route Cache, S will initiate a new route discovery. The reason is that with a very high probability those routes may also not work or may work badly due to mobility and traffic dynamics. The SSR (Send extra Route Request) extension has also been implemented.

C. Performance Metric

The following metrics will be used to evaluate the performance of HADOF.

- Packet drop ratio: The percentage of data packets that have been sent by not been received by the destinations, which equals to 1 minus end-to-end throughput.
- Overhead: In this paper, we consider *routing overhead*, *energy consumption overhead*, *encryption overhead*, and

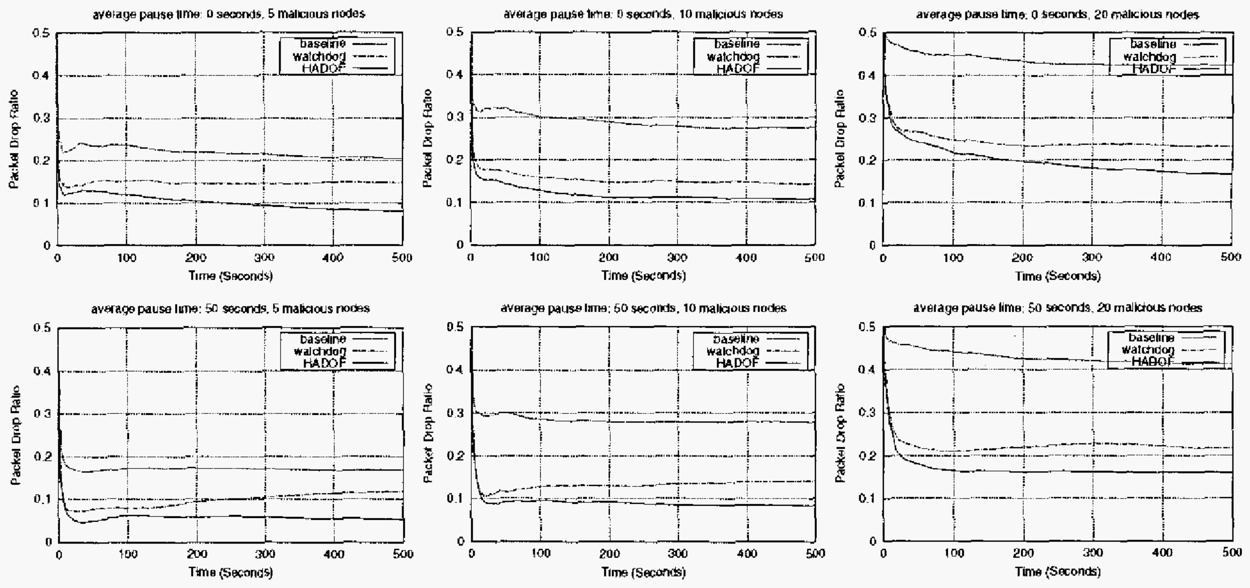

Fig. 5. Packet drop ratio comparisons under gray hole attacks

complexity overhead. Given a traffic pattern, routing overhead indicates how many route discoveries have been initiated by the sources. Energy consumption overhead denotes how much extra energy need to be consumed. To keep the confidentiality and integrity of the transmitted content, extra cryptographic operations are needed, and the encryption overhead describes how many extra cryptographic operations are needed by these mechanisms. Complexity overhead accounts for the extra storage and computations needed by applying these mechanisms.

VII. Performance Evaluation

We use "baseline" to denote the baseline system, "watchdog" to denote the system based on watchdog and pathrater, and "HADOF" to denote the system based on HADOF. We use different node movement patterns for each simulation by changing the average pause time and the seed of random number generator. By varying the number of malicious nodes and the average pause time, we get different configurations. For each configuration, the results are averaged over 25 rounds of simulations, where at each round we change the random seed to get different movement and traffic patterns. To make fair comparison, for each configuration and each round of simulation, the same movement and traffic patterns were used by all the three systems.

A. Packet Drop Ratio Comparisons

We compare the packet drop ratios of the three systems under different scenarios. First, we compare the packet drop ratios under only gray hole attacks. That is, no nodes will submit false reports. Second, we compare the packet drop ratios under both gray hole and frame-up attacks, where some malicious nodes will drop packets and frame up their neighbors when possible. Third, we show how friendship mechanism can mitigate the effects of frame-up attacks.

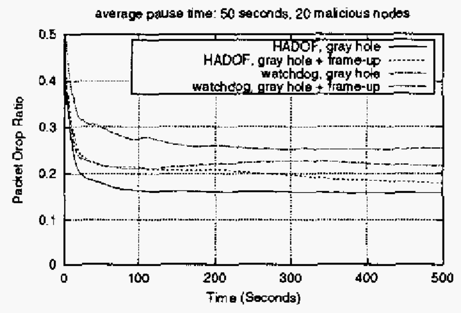

Fig. 6. Effects of frame-up attacks

1) Gray hole: In our simulations, we vary the number of malicious nodes from 5 to 20. The gray hole is implemented in such a way that each malicious node drops half of the packet passing through it. The simulation results under different configurations are plotted in Fig. 5. From these results we can see that HADOF outperforms watchdog in all situations. For example, under the configuration of pause time 50 seconds, 20 malicious nodes, the packet drop ratio of baseline is more than 40%, watchdog can reduce the packet drop ratio to 22%, while for HADOF, the packet drop ratio is only 16%, that is, more than 33% improvement is obtained over watchdog under this configuration. Under the configuration of pause time 50 seconds, 5 malicious nodes, more than 55% improvement is obtained over watchdog by HADOF.

2) Gray hole plus frame-up attacks: We investigate the packet drop ratio under both gray hole and frame-up attacks. In HADOF, the only way for a malicious node to frame up a good node is to let the source *suspect* that the good node is cheating. To achieve this, a malicious node can report a smaller RN number than the actual value to frame up the node ahead of it on the route, and/or report a larger FN number than the actual value to frame up the node just following it on the route. However, the malicious node can never make the source *believe* that a good node is cheating, since malicious

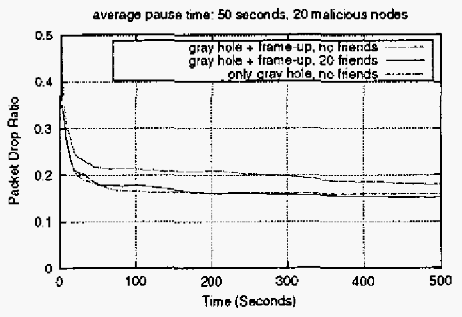

Fig. 7. Effects of friendship mechanism

node cannot create solid evidence.

In watchdog, there exist a variety of ways for a malicious node to frame up good ones. For example, if node A has sent a packet to B and asks B to forward it to C, B has many ways to make A believe that it has sent the packet to C while B did not send packets or intentionally caused transmission failure. As reckoned in [3], many reasons can cause a misbehaving node not being detected, such as ambiguous collisions, receiver collisions, limited transmission power, false misbehavior, collusion, and partial dropping. In our simulations, we only implement the frame-up attacks through receiver collisions. That is, B will forward packet to C only when it knows that C cannot correctly receive it (e.g., C is transmitting data to another node, or receiving data from another node). Since A can only tell whether B has sent the packet to C, but cannot tell whether C has received it successfully, B can easily frame up its neighbors.

Fig. 6 shows the simulation results with the configurations of 20 malicious nodes, half of them applying frame-up attacks. First we can see that the degradation of HADOF caused by frame-up attacks is limited. Second, we see that frame-up degrades the performance of both, and affects watchdog more than HADOF. Meanwhile, it is important to point out that we have shown the best-case results for watchdog because we have made many assumptions which favor watchdog, such as no collusion attacks, only receiver collisions, perfect MAC protocol. For HADOF, no extra assumptions are needed except for those listed in Section III.

3) Effectiveness of Friendship: In the previous simulations, friendship has not been introduced and the source only trusts the destination. Next we show the results after introducing friendship mechanism to combat frame-up attacks. We conduct simulations under the situations that each source has 20 friends which are randomly chosen among all good nodes in the network. Fig. 7 shows the simulation results using HADOF with the configuration of average pause time 50 seconds, 20 malicious nodes, half of them launching both gray hole and frame-up attacks, and half of them only launching gray hole attacks. From these results we can see that the effects of frame-up attacks can be overcome when trustiness has been established among certain number of users. For example, with 20 friends, the packet drop ratio, which is 15%, is even lower than the situation that no frame-up attacks are launched, which is 16%.

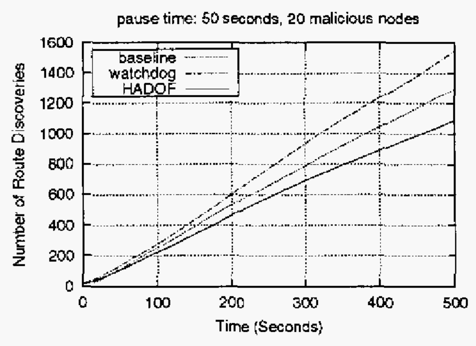

Fig. 8. Route discovery overhead comparison

B. Overhead Comparisons

Routing Discovery Overhead: For each simulation, we have counted the total number of route discoveries that have been initiated by all the sources. Fig. 8 shows the results under the configuration of average pause time 50 seconds, 20 malicious nodes, and only gray hole attacks. From these results we can see that though HADOF needs to initiate route discoveries preventively, it still has the lowest routing discovery overhead. In the baseline system, only one route is returned for each route discovery, which explains why baseline needs to initiate more route discoveries. This also verifies the effectiveness of path diversity. Surprisingly, watchdog has the highest route discovery overhead, which comes from its high false alarm ratio, since a new route discovery will be initiated once no route has average reputation larger than 0.

Energy Consumption Overhead: One major drawback of watchdog is that it consumes much more energy than HADOF, because each node has to keep monitoring its neighbors' transmission activities. We use Fig. 1 to illustrate why watchdog needs to consume extra energy. For example, after B sends a packet to C and asks C to forward the packet to D, B has to keep listening C's transmission. If C is a malicious node, C can launch resource consumption attacks to consume B's energy by putting off forwarding packets for B. Even if C is a good node, B still needs to consume extra energy to receive, decode, and compare the packets transmitted by C with the packets stored in B's buffer. This consumes a lot of extra energy. By requiring nodes to keep monitoring their neighbors, watchdog not only reduces network capacity, but also consumes extra energy. On the other hand, HADOF has no such drawbacks.

Encryption Overhead: As we discussed in Section III, all packets should be encrypted and signed to ensure data confidentiality and integrity. Otherwise, outside attackers can easily intercept those messages through eavesdropping. Compared with the baseline system, HADOF introduces some encryption overhead which comes from encrypting the reports. In most situations only the destination needs to submit reports, and the source and the destination already share a secrete key for data encryption. Thus, the reports from the destination can just be encrypted by this secrete key, which introduces little overhead. In addition, if the amount of data for reporting packet forwarding statistics is much less than the amount of data, which is generally true, the overhead of encrypting reports of intermediate nodes on the route will become negligible compared with data encryption overhead.

Complexity Overhead: In HADOF, each source needs to launch a route traffic observer to maintain and update traffic statistics, and maintain records to keep track of cheating behavior. However, both can be implemented using simple data structures, and consume little memory. The computation overhead comes from updating traffic statistics, route quality, and cheating records. These operations will not introduce much computation burden. In watchdog, each node also needs to keep a reputation database and need to calculate route quality. Moreover, each node in watchdog needs to keep an extra buffer to store the packets that it has requested its neighbors to forward but have not been confirmed, which consumes a lot of extra memory, and may introduce extra computation overhead to compare the packets.

VIII. Conclusion

Mobile ad hoc networks have attracted a lot of attentions from military, industry, and academy. However, before mobile ad hoc networks can be successfully deployed, the security issues have to be resolved first. In this paper we proposed HADOF to defend against routing disruption attacks launched by inside attackers, which can be implemented upon the existing source routing protocols. HADOF is capable of adaptively adjusting routing strategies according to the network dynamics and nodes' past records and current performance. It can handel various attacks launched by malicious nodes, such as black hole, gray hole, frame-up, rushing attacks, and wormhole attacks. Moveover, HADOF introduces little overhead to the existing routing protocols, and is fully distributed.

Extensive simulation studies have also confirmed the effectiveness of HADOF. For example, in the presence of 20 malicious nodes with each launching a gray hole attack by selectively dropping half of the packets passing through it, and with half of them also launching frame-up attacks, the system without protection schemes has 40% packet drop ratio, the system using watchdog and pathrater can reduce the packet drop ratio to at most 26%, while the system using HADOF can reduce the packet drop ratio to only 15%. The simulation results have also shown that HADOF introduces little routing discovery, encryption and complexity overhead.

Acknowledgments

This work was supported in part by the Army Research Office under URI Award No. DAAD19-01-1-0494.

References

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

[2] Y. Zhang and W. Lee, "Intrusion Detection in Wireless Ad-Hoc Networks," in *MobiCom*, Boston, MA, USA, August 2000.

[3] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," in *MobiCom*, Boston, MA, USA, August 2000.

[4] J. P. Hubaux, L. Buttyan, and S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks," in *MobiHOC*, May 2001.

[5] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," in *MobiCom*, Atlanta, GA, USA, Sep. 2002.

[6] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols," in *WiSe*, San Diego, CA, USA, Sep. 2003.

[7] T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyumn, and L. Viennot, "Optimized Link State Routing Protocol," Internet-Draft, draft-ietf-manet-olsr-06.txt, Sep. 2001.

[8] C. E. Perkins, E. M. Belding-Royer, and S. R. Das, "Ad Hoc On Demand Distance Vector (AODV) Routing," Internet-Draft, draft-ietf-manet-olsr-10.txt, Jan. 2002.

[9] D. B. Johnson, D. A. Maltz, Y. C. Hu, and J. G. Jetcheva, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)," Internet-Draft, draft-ietf-manet-dsr-07.txt, Feb. 2002.

[10] R. G. Ogier, F. L. Templin, B. Bellur, and M. G. Lewis, "Topology Broadcast Based on Reverse-Path Forwarding (TBRPF)," Internet-Draft, draft-ietf-manet-tbrpf-05.txt, Mar. 2002. Work in progress.

[11] P. Papadimitratos and Z. Haas, "Secure Routing for Mobile Ad hoc Networks," in *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, Jan 2002.

[12] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "A Secure Routing Protocol for Ad Hoc Networks," in *Proceedings of ICNP*, Nov. 2002.

[13] M. G. Zapata and N. Asokan, "Securing Ad Hoc Routing Protocols," in *WiSe*, Sep. 2002.

[14] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet Leashes: A Defense against Wormhole Attacks in Wireless Networks," in *IEEE INFOCOM*, 2003.

[15] Y.-C. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," *Ad Hoc Networks Journal*, vol. 1, pp. 175–192, 2003.

[16] S. Buchegger and J.-Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol," in *Mobihoc*, 2002, pp. 226–236.

[17] L. B. and J.-P. Hubaux, "Stimulating Cooperation in Self-organizing Mobile Ad Hoc Networks," *Mobile Networks and Applications*, vol. 8, no. 5, pp. 579 – 592, Oct. 2003.

[18] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks," in *IEEE INFOCOM*, 2003.

[19] V. Srinivasan, P. Nuggehalli, C. F. Chiasserini, and R. R. Rao, "Cooperation in Wireless Ad Hoc Networks," in *IEEE INFOCOM*, 2003.

[20] S. Capkun and J.-P. Hubaux, "BISS: Building Secure Routing out of an Incomplete Set of Security Associations," in *WiSe*, San Diego, CA, USA, Sep. 2003.

[21] P. Michiardi and R. Molva, "Core: a COllaborative REputation Mechanism to Enforce Node Cooperation in Mobile Ad Hoc Networks," in *IFIP - Communications and Multimedia Security Conference*, 2002.

[22] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile Computing," In *Moible Computing*, edited by Tomasz Imielinski and Hank Korth, chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

[23] P. Kyasanur and N. Vaidya, "Selfish MAC Layer Misbehavior in Wireless Networks," *IEEE Transactions on Mobile Computing*, April 2004.

[24] G. O'Shea and M. Roe, "Child-Proof Authentication for MIPv6 (CAM)," *ACM Computer Communications Review*, April 2001.

[25] G. Montenegro and C. Castelluccis, "Statiscially Unique and Cryptographically Verifiable (SUCV) Identifiers and Addresses," in *Proceedings of NDSS*, 2002.

[26] D. Coppersmith and M. Jakobsson, "Almost Optimal Hash Sequence Traversal," in *Proceedings of the Fourth Conference on Financial Cryptography*, 2002.

[27] R. Castaneda A. Nasipuri and S. R. Das, "Performance of Multipath Routing for On-Demand Protocols in Ad Hoc Networks," *ACM/Kluwer Mobile Networks and Applications (MONET) Journal*, vol. 6, no. 4, pp. 339–349, 2001.

[28] P. Papadimitratos, Z. J. Haas, and E. G. Sirer, "Path Set Selection in Mobile Ad Hoc Networks," in *MobiHOC*, EPFL Lausanne, Switzerland, June 2002.

[29] IEEE Computer Society LAN MAN Standards Committee, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1007," The Institue of Electrical and Electrics Engineers.