Secure Reporting of Traffic Forwarding Activity in Mobile Ad Hoc Networks

Heesook Choi, William Enck, Jaesheung Shin, Patrick McDaniel, Thomas F. La Porta
Department of Computer Science and Engineering
Pennsylvania State University, University Park, PA 16802
E-Mail:{hchoi,enck,jsshin,mcdaniel,tlp}@cse.psu.edu

Abstract

Nodes forward data on behalf of each other in mobile ad hoc networks. In a civilian application, nodes are assumed to be selfish and rational, i.e., they pursue their own self-interest. Hence, the ability to accurately measure traffic forwarding is critical to ensure proper network operation. These measurements are often used to credit nodes based on their level of participation, or to detect loss. Past solutions employ neighbor monitoring and reporting on node forwarding traffic. These methods are not applicable in civilian networks where neighbor nodes lack the desire or ability to perform the monitoring function. Such environments occur frequently in which neighbor hosts are resource constrained, or in networks where directional antennas are used and reliable monitoring is difficult or impossible.

In this paper, we propose a protocol that uses nodes on the data path to securely produce packet forwarding reports. Reporting nodes are chosen randomly and secretly so that malicious nodes cannot modify their behavior based upon the monitoring point. The integrity and authenticity of reports are preserved through the use of secure link layer acknowledgments and monitoring reports. The robustness of the reporting mechanism is strengthened by forwarding the report to multiple destinations (source and destination). We explore the security, cost, and accuracy of our protocol.

1. Introduction

The establishment of a wireless infrastructure is non-trivial, especially in volatile environments where node mobility dominates. Occasionally, erecting fixed infrastructures is not feasible due to location or temporal validity. For example, it is not possible to build a wireless tower in the middle of a hostile battlefield. Furthermore, the tower cannot be moved as an attack progresses. In other mission-oriented scenarios such as search and rescue, terrestrial obstacles, e.g. avalanche prone mountains, inhibit the creation of fixed access points.

In the absence of a fixed infrastructure, mobile ad hoc networks (MANETs) can be used. By not requiring a fixed infrastructure or centralized control for communication, MANETs are well suited for the aforementioned scenarios. Within the network, multi-hop paths are created between nodes that formerly could not communicate. Ideally, each node selflessly forwards each packet to the next node in the path. As nodes move, they leave and join various communication links, thus promoting many ephemeral paths.

Reliable operation in a MANET requires explicit cooperation between nodes. While this is feasible to assume for mission-oriented scenarios, careful consideration needs to take place when applying MANETs to civilian applications. In a civilian mobile ad hoc network, communicating nodes will use any relay points present. It is conceivable that selfish or malicious nodes exist in these networks. Additionally, reliability can be severely impacted by network congestion and mobility. Ergo, there is a need to detect selfish or malevolent behavior, promote cooperation between nodes, and route around network congestion.

One method for detecting malicious behavior is to generate reports on traffic flow between nodes. This information can be used to not only detect misbehavior, but also to indicate good network citizens. By identifying nodes that play fairly, a payment scheme can be implemented in order to further promote cooperation. Traffic reports can also be used to detect bottlenecks.

Previously proposed solutions rely on neighboring nodes to eavesdrop on data transmissions in order to generate reports. While this may work well in networks with trusted nodes, i.e. military settings, it is not feasible for civilian ad hoc networks. Furthermore, such techniques may also fail in military settings if directional antennas are used, since nodes cannot reliably monitor data transmissions.

In this paper, we propose a secure random reporting protocol for a civilian ad hoc network, in which the source and destination collect reports from intermediate nodes on the routing path. Every data packet delivered initiates a report from one intermediate node that is randomly chosen by a source node. The chosen node then integrates its self-report into the packet before forwarding the transmission. The symmetric-key construction efficiently prevents disclosure of the selected node's identity from all adversaries except those that can mount large scale traffic analysis attacks. Note that reports may become lost due to mobility and congestion. In order to provide robustness in the face of loss, the report is sent to the source, the destination, or both.

While the secure random reporting protocol provides secret node selection, as well as integrity and authenticity of reports, it does not guarantee that the self-report is accurate. Although nodes cannot manipulate others' reports, they may not be trusted to generate accurate reports. To rectify this inadequacy, we propose a forgery detection scheme that provides proofs of delivery implemented by secure network layer acknowledgments.

We have simulated these schemes using ns-2 [5]. Our results show that we accurately monitor packet forwarding activity even in lossy networks. We further simulate malicious packet dropping to look at the effectiveness of our secure random reporting protocol.

The rest of this paper is organized as follows. Section 2 reviews previous research in malicious node detection and cooperation in ad hoc networks. Section 3 describes possible threats in civilian ad hoc networks. Section 4 presents an overview of the proposed random reporting protocol. This scheme is then strengthened in Section 5 as we extend it to provide report integrity, node selection confidentiality, and prevention of falsified reports. Next, Section 6 provides simulation results and computational overhead of the secure random reporting protocol. Finally, Section 7 concludes.

2. Related Work

Detection of malicious behavior and collection of cooperation history for crediting are two motivating factors for monitoring nodes. This section discusses previous research in these areas.

2.1. Detection of Malicious Behavior

The Watchdog/Pathrater [10] scheme proposes the use of a watchdog for detecting misbehaving nodes, and a pathrater to help the routing protocol avoid detected misbehaving nodes. The design utilizes intermediate nodes along the routing path, wherein a node sends a packet to an intermediate downstream node and verifies the node that forwards it. If the node does not send the packet within a predefined period, it is declared as misbehaving, and the monitoring node notifies the source. Pioneering the area of intrusion detection in ad hoc networks, Zhang and Lee [16, 17] propose a general architecture, in which all nodes participate in the monitoring of data transmission. Each node is responsible for monitoring a transmission range and cooperating with neighboring nodes in order to detect intrusions. Zhang and Lee later proposed a second scheme to reduce the number of nodes involved in monitoring [1]. In this cluster-based scheme, a cluster head (CH) is elected for monitoring data traffic within the transmission range. The elected CH is responsible for monitoring all neighboring nodes and checking statistics. AODVSTAT [14] implements an intrusion detection system (IDS) within the AODV [13] routing protocol. The system monitors for routing message drops, data-packet drops, MAC/IP spoofing, and resource depletion attacks. In AODVSTAT, an IDS monitors all observable transmissions from neighbors. Note that all of the above schemes require some level of communication eavesdropping. These solutions are not feasible in our target environments because reliable eavesdropping is not possible.

Awerbuch et al. [2] propose an alternate scheme that uses intermediate nodes on the data path. If a source does not receive an ACK from a destination, the source begins probing all intermediate nodes. This causes each node along the path to send an ACK back to the source. Unfortunately, due to the dynamic characteristics of MANETs, data paths can change frequently, possibly before the failed link is found.

2.2. Cooperation

Many times, cooperation between nodes cannot be expected without incentives. Several algorithms have been proposed that use payment schemes. A node may be paid via a credit for behaving cooperatively or excluded/penalized for misbehaving.

Sprite [18] proposes an incentive system where selfish nodes are encouraged to cooperate. In Sprite, each node is motivated to honestly report its actions, even in the presence of selfish node collusion. Intermediate nodes retain receipts of received messages. The receipt is then sent to the CCS (Credit Clearance Service) as proof of forwarding, and the CCS then charges/credits based on the received reports.

CORE [12], another cooperation algorithm, uses a collaborative reputation mechanism to encourage nodes to cooperate. The reputation is calculated via both direct and indirect observation by a node and its neighboring nodes, respectively, within the transmission range. In similar scheme, CONFIDANT [3], each node monitors nodes existing one hop away. If a node detects and concludes malice, it generates an ALARM message to either a source or a friend. This, in turn, causes misbehaving nodes to be excluded from the community.

All of the aforementioned detection and cooperation schemes require the observation of neighboring nodes. Additionally, these schemes deal only with detection or cooperation. Our reporting protocol targets more general applications, including both detection of malicious behavior and crediting for cooperation. The information provided by the reporting scheme is also vital for detecting data bottlenecks.

3. Threat Model

When considering civilian ad hoc networks, there is a potential for anomalous behavior. Inconsistencies arise from self-interest, maliciousness, network congestion, and mobility. This section discusses these threats and how they pertain to packet forwarding activity and report collection. The discussion illuminates the set of threats to which we aim to be resilient.

It is important to note that the high loss and delay prevalent in wireless and mobile networks exacerbates the problem of detecting selfish/malicious nodes. If a node drops packets and moves, it is difficult to detect whether the packet loss is from mobility or selfishness/maliciousness. Likewise, in a congested network, packets are dropped because of packet buffer overflows. Distinguishing between selfish/malicious drops and congestion is difficult. Regardless of the reasons for packet loss, a source node may wish to avoid particular nodes due to the mere occurrence of lost packets, whether it be the result of selfish/malicious behavior or simply network congestion.

Most forms of non-cooperation result in *denial of service* (DoS). In the extreme case, an ill-performing node would simply refrain in participating in routing, and hence would never be placed on a path. Possibly more damaging, a similar attack would allow the node to accept a position on the path, but it would not forward data packets. Our protocol does nothing to prevent or detect attacks on the routing protocol, but rather focuses on accurate reporting of packet forwarding.

Nodes may also drop packets selectively. For example, a selfish node may choose not to forward packets for a specific source or destination, or conversely, simply favor a source or destination by dropping traffic for others when they are in competition. Similarly, the node can choose particular applications to drop or show preferential treatment. Finally, a node may randomly drop packets in order to simply save energy.

Note that only a few well-selected drops are necessary to vastly reduce the throughput between a source and destination: each drop causes the congestion control algorithm to aggressively throttle traffic [6]. Connection recovery is slow, and the attacker gains advantage with little effort [15].

More subtle attacks exist. In credit based systems, a node benefits from forwarding more packets than its neighbors. To gain an advantage, a malicious node injects fake packets. This expends the energy of all forwarding nodes, thereby rendering them incapable of forwarding future legitimate packets. The known defense for this attack is to use interleaved hop-by-hop authentication schemes [19, 20], where fake packets are filtered mid-transmission. This paper does not address this type of attack.

Existing proposed cooperation schemes for civilian ad hoc networks use rewards or penalties to encourage cooperation. Rewards and penalties are dictated by reports of mobile node behavior. The credit for relaying other traffic might be money, more bandwidth, or higher priority service. The policy motivates mobile nodes to cheat, manipulate, or drop the reports so that they get more credit and avoid being penalized. Defending against potential forwarding and replay attacks on the reporting data is a challenging issue for monitoring the packet forwarding activity.

4. Overview of Random Reporting Protocol

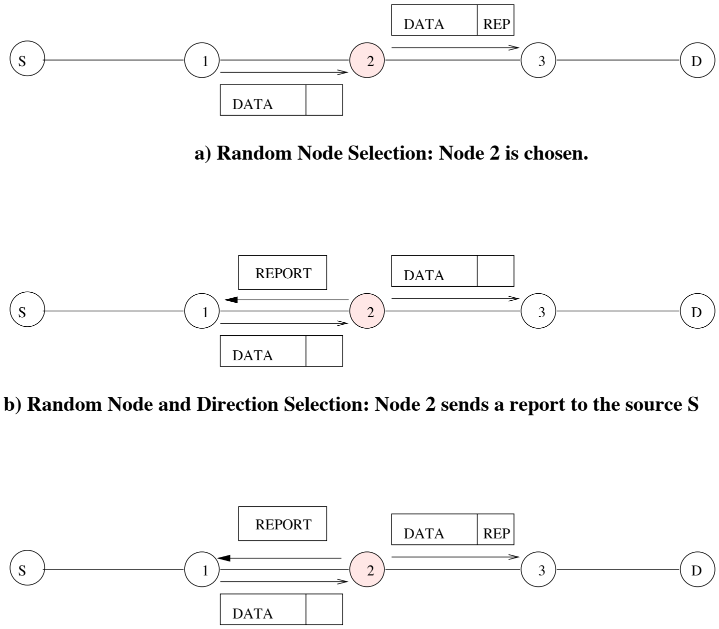
For the purposes of this paper, it is assumed that dynamic source routing (DSR) [8] is used. The DSR routing protocol provides a full path between the source and destination. This is advantageous when choosing a random intermediate node. It is reasonable to assume that the source and destination nodes are trusted, as they are the entities responsible for the data traffic. The protocol focuses on the secure reporting of forwarding activities for the data transmission.

Each intermediate node only needs to keep track of its own contribution, instead of observing the actions of other nodes. Using intermediate nodes in this manner is rational when dealing with a civilian ad hoc network. The rest of this section provides an overview of the Random Reporting Protocol. While alone the Random Reporting Protocol is not secure, Section 5 introduces the Secure Random Reporting Protocol.

4.1. Basic Periodic Reporting

Basic Periodic Reporting is a simplistic method in which intermediate nodes periodically send reports to the destination. These reports are collected by the destination and used to analyze network paths. The compiled report is then used for future path engineering, crediting, and determination of anomalous points.

This simple periodic reporting scheme functions well for static networks, but it does not work well for dynamic networks, or networks with malicious nodes. First, the scheme's quality is highly dependent on report transmission frequency. Additionally, rapid changes due to mobility or congestion quickly degrade its effectiveness, because reports may be lost or paths may change before reports are gathered. Since reliable transmission is not guaranteed, the disappearance of a node's report may cause it to be viewed as an anomalous or congested point, even if it has correctly forwarded all data packets. The report data is transmitted via the same path as normal data. This allows a selfish or



a) Random Node Selection: Node 2 is chosen.

b) Random Node and Direction Selection: Node 2 sends a report to the source S

c) Random Bidirectional Reporting: At node 2, report is transmitted to both S and D

Figure 1. Random Reporting Protocol: source S and destination D

malicious node to know the source of any report. The node can then drop or change particular reports for their own self interest.

4.2. Random Reporting Node Selection (RRNS)

In order to address the aforementioned problems with packet manipulation and dynamic networks, we propose a Random Reporting Node Selection (RRNS) method. For every packet, the source randomly chooses one intermediate node to send a report to the destination. This is accomplished by coupling each data packet with a report, so that when the report is received by the destination, the relaying activity of intermediate nodes can be dynamically observed.

In RRNS, if the path consists of n intermediate nodes, any node can be chosen with probability $1/n$. Figure 1-(a) illustrates RRNS where node 2 has been randomly chosen. In general:

1. For all packets p, source S randomly chooses (uniform distribution) intermediate node n_i to send a report. S attaches a report request RR to p, identifying n_i as the chosen node.
2. For a packet p with RR for n_i, n_i attaches report R to p before forwarding to destination D.
3. Destination D receives p, including R from all intermediate nodes, and periodically analyzes the reports, looking for traffic deviations.

The idea of choosing a random node is motivated by micro-payment [7, 11], in which a randomly chosen transaction is used for a merchant to deposit some amount of money. Applied to RRNS, the randomly chosen intermediate node should add to the forwarding packet the number of packets it has forwarded since joining the path.

Since the intermediate node is selected randomly, other nodes are unable to predict the selection schedule. This prevents nodes from timing their attacks to maximize their duration. While the randomness provides better reports, the described scheme is vulnerable to attack. Without taking precautions, reports may be manipulated by downstream nodes with selfish intentions. Section 5 addresses this by introducing secret node selection. In summary, RRNS is advantageous, because it gathers reports from nodes in real time and has very low communication overhead, due to the coupling of reports with every data packet. We quantify this overhead in Section 6.

4.3. Random Reporting Node and Direction Selection (RRNDS)

In RRNS, if packets are lost due to congestion or mobility, the destination will only receive the reports sent before the anomaly occurred. Thus, the destination may misinterpret the location of the problem.

Random Reporting Node and Direction Selection (RRNDS) is proposed to make RRNS more robust. RRNDS extends Step 2 of RRNS by randomly deciding the direction to send the report at the chosen node. If the report is sent towards the destination, it is attached to the data packets, just as in RRNS. On the other hand, if a source-bound direction is chosen, a separate report message is transmitted. Figure 1-(b) shows this scheme.

4.4. Random Bidirectional Reporting (RBR)

The report in RRNDS is transmitted to either the source or the destination. Unfortunately, the amount of report information received by the destination or source is reduced in RRNDS. Due to this shortage of reports, the source or destination may not precisely analyze the relaying activity of intermediate nodes.

We address this problem by modifying Step 2 of RRNS to transmit the report to both the source and destination. This technique, shown in Figure 1-(c), is referred to as Random Bidirectional Reporting (RBR). In the figure, node 2 sends a report to the destination and source node. Simulation results reported in Section 6 show that bidirectional reporting improves effectiveness in the face of mobility.

Additionally, for both RRNDS and RBR, if the communication between source and destination is bidirectional, source-bound reports are attached to data packets destined for the source. This reduces communication overhead.

5. Secure Reporting Protocol

The random reporting protocols discussed in Section 4 are based upon random node selection. If intermediate nodes (selfish or malicious) discover a packet including a report and the selected node, the information may be manipulated or dropped.

This section proposes an efficient construction that conceals the node selection from other intermediate nodes. In civilian mobile ad hoc networks, the intermediate nodes cannot be assumed to be honest; lying may provide more credit. Thus, in order to assure the validity of node reports, a chain of HMACs on the link layer acknowledgments is proposed. This addition provides forgery detection.

The following notations are used in the secure reporting protocol and forged report detection schemes.

- ID_i: Identifier of node n_i.
- K_{ij}: a pair-wise key between node n_i and n_j.
- $hash(x)$: Cryptographic hash function computation for x
- σ: $HMAC(K_{SD}, DATA|ID_i)$ computation result for the data and ID_i.
- $DATA$: Data transmitted between the source and destination.

Mobile devices are less powerful in computation and have a battery of limited lifetime. Therefore, symmetric cryptography is often more appropriate for mobile devices in ad hoc networks. The secure random reporting protocol requires three pairs of symmetric keys: source and destination, source and intermediate nodes, and destination and intermediate nodes. Key management schemes for distributing symmetric keys in ad hoc networks have been proposed [9, 4], and thus will not be discussed. Any efficient symmetric key management scheme can be used for distributing symmetric keys to mobile nodes; its choice does not impact our results.

5.1. Secure and Random Reporting Protocol

DSR allows the source node to know the full routing path. The source node chooses one intermediate node n_i uniformly at random, and computes $Token$, which is added to the data packet. The $Token$ contains the node selection information which is not disclosed. The use of an $HMAC$ in the computation of the $Token$ provides randomness and secrecy in the node selection.

```
Sender:
  - Choose one intermediate node $n_i$
  - Compute $\sigma = HMAC(K_{SD}, DATA|ID_i)$,
  - Compute $H_i = hash(K_{Si}|\sigma)$
  - Generate $Token = \sigma \oplus H_i$
              → first intermediate node: $[DATA, \sigma, Token]$
```

When a node receives a packet, it needs to determine if it is the randomly selected node. At the same time, no other nodes can be allowed to know which node has been chosen. Upon receiving a data packet$(DATA, \sigma, Token)$, an intermediate node n_j computes $H_j = hash(K_{jS}|\sigma)$ and XORs it with the received $Token$. If the result of the XOR operation is equal to the received σ, the node knows it was chosen. This is only satisfied at node n_i since the source used a pairwise key K_{Si}. Since the above test in other intermediate nodes is not satisfied, they do not generate reports.

The chosen intermediate node n_i sends its report by attaching it to the data packet. The report R includes the number of packets the node forwarded for the source and destination. For integrity purposes, the chosen intermediate node computes $hash$ with its report R and its pair-wise symmetric key shared with the destination.

$$\begin{aligned} H_D &= hash(K_{iD}|R) \\ Report &= [R, H_D] \end{aligned}$$

Table 1. Random and Secure Reporting

Sender: - Choose one intermediate node n_i - Compute $\sigma = HMAC(K_{SD}, DATA \vert ID_i)$ - Compute $H_i = hash(K_{Si} \vert \sigma)$ - Compute $Token = \sigma \oplus H_i$ - Send the packet $(DATA, \sigma, Token)$
Intermediate Node n_i: - Compute $H_i = hash(K_{iS} \vert \sigma)$ - XOR H_i with $Token \rightarrow H_i \oplus Token = H_i \oplus \sigma \oplus H'_i$, where H'_i is received - Check if $XOR(Token, H_i) == \sigma$ - If n_i is chosen, **o** Compute $H_D = hash(K_{iD} \vert R)$ **o** Generate $Report = [R, H_D]$ and attach it to the data packet
Destination: - Evaluate if $hash(K_{Di} \vert R) = H_D$ to find out the chosen node. - Save the report and check whether there exists a misbehavior. - If the report is not valid, ignore the report.

This scheme is resilient to another node n_k replacing $Token = \sigma \oplus H_i$ with $\sigma \oplus H_k$, because the resulting $HMAC(K_{SD}, DATA|ID_k)$ is not equal to the received $\sigma = HMAC(K_{SD}, DATA|ID_i)$. Without knowing K_{SD}, the node n_k cannot change σ to masquerade as a selected node. When receiving a data packet, the destination checks that σ and the received report R are valid, i.e. if $hash(K_{Di}|R) = H_D$ and $\sigma = HMAC(K_{DS}, DATA|ID_i)$ are satisfied.

In RRNDS and RBR, reports are transmitted to the source node. The only difference is that the chosen intermediate node uses the symmetric key K_{iS} to generate the source-bound report. The report generation is the same as described above. The key idea of node selection is to conceal the node selection from other nodes. Table 1 summarizes the secure random reporting protocol.

While the secret random reporting protocol protects the identity of the selected node from in-path adversaries, it is susceptible to traffic analysis. If an adversary can overhear traffic going in and out of node n_i, determining whether or not n_i was selected is trivial. If the incoming and outgoing data does not match, n_i has attached a report. Therefore, if this adversary is downstream from n_i, it can selfishly strip out the report. However, we expect this to be limited to the direct neighbors of the selected nodes in most cases.

5.2. Report Forgery Detection Scheme

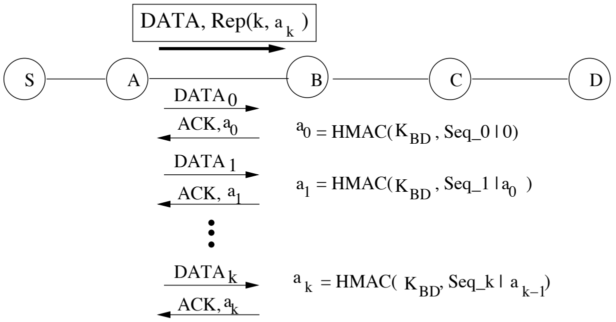
Even if the report transmission is secure, we still need a way to validate the information provided by the selected node. To address this, a report forgery detection scheme is proposed.

In wireless networks, the link layer provides an acknowledgment (ACK) by which a receiving node confirms to a sending node that it received the packet successfully. The sending node waits for an ACK from the receiver. If it does not receive an ACK after retransmitting a packet several times, it considers the link to the receiving node broken and sends a ROUTE ERROR message back to the source node. If this occurs, the source node will change the data path.

This property of the link layer protocol is used to provide forgery detection of reports. Each data packet is sent in a link layer frame. For each data packet i, successfully received in frame j_f, the receiver sends an ACK(j_f, seq_i) and α_i, which is an HMAC of α_{i-1} and seq_i. Intermediate nodes generate self-reports for each flow defined by the source and destination addresses. Intermediate nodes may relay data packets for multiple flows. They generate ACKs with HMAC chains for the packet sequence number pertaining to the flow.

For the HMAC's symmetric key, the receiver R uses the pair-wise K_{RD} shared with the destination. The forgery detection scheme does not use a key shared between two neighboring nodes in order to prevent these nodes from colluding and allowing one node to manipulate the HMAC.

Figure 2 provides an example communication flow of the report forgery detection scheme. Data transfer begins with initial packet DATA0. For this **first packet**, node B computes $\alpha_0 = HMAC(K_{BD}, seq_0|0)$. Node B then sends to A both a link layer ACK and the computed α_0. After the initial packet, node B uses the previous HMAC, α_{i-1}, in


DATA, Rep(k, a k)
S
A
B
C
D
DATA 0
ACK, a0
a0 = HMAC(KBD , Seq_0 | 0)
DATA 1
ACK, a1
a1 = HMAC(KBD , Seq_1 | a0)
DATA k
a k = HMAC(KBD, Seq_k | a k–1)
ACK, ak

Flow (S,D) Table

Node ID	Reported Chained HMACs
A	$a_k = HMAC(K_{BD}, Seq\_k \mid a_{k-1})$
B	$b_k = HMAC(K_{CD}, Seq\_k \mid b_{k-1})$
C	$c_k = HMAC(K_{D}, Seq\_k \mid c_{k-1})$

Figure 2. Chained HMACs to Detect Report Forgery

the computation of α_i.

$$\alpha_i = HMAC(K_{BD}, seq_i|\alpha_{i-1})$$

When node A is randomly chosen to send a report for data packet k, both k and α_k are transmitted. In the case of Figure 2, since the report direction is towards the destination, the report is attached to a DATA transmission.

Mobility is fundamental to MANETs. When a path changes, the destination may not have the initial sequence number for a particular path. Therefore, for the **first report** in a path, node A must include α_0, the initial sequence number, α_i, the sequence number map, and the number of packets it has forwarded. Since the destination knows the symmetric key, K_{BD}, it can verify α_0 was created correctly. The most recent HMAC, α_i, can then be verified by computing the HMAC chain from α_0.

The chained HMACs are intended to detect if nodes forge their reports. The forgery detection scheme protects against two types of misbehavior. In Figure 2, node A may try to cheat by sending a report saying it forwarded 100 packets when it really only forwarded 50 packets. In order for A to cheat under the described scheme, it must provide α_{100}. Since generation of α_{100} requires knowledge of K_{BD}, only known by node B and D, A cannot fake the report. The only way for A to learn α_{100} is for node B to tell it. This only occurs after A forwards B 100 packets.

The forgery detection scheme also protects against a malicious node B. In one case, node B may purposefully choose to not send an ACK to node A. If this occurs, node A will send a ROUTE ERROR to the source node, and a new data path will be chosen. If, on the other hand, node B sends A an ACK, but purposefully generates the wrong HMAC, for the initial α_0 or the intermediate α_i, node A will report the wrong α_0 or α_i to the destination.

Fortunately, the destination can determine if α_0 or α_i is incorrect. Since the destination can compute $HMAC(K_{BD}, seq_0|0) = \alpha_0$, if the resulting α_0 does not match the received α'_0, it can detect that node B is misbehaving. Similarly, the destination can compute α_k and compare it to the received α'_k. As shown in Figure 2, the destination retains knowledge of the state by keeping a table consisting of the node identifier and the most recently received HMAC, α_i. The destination determines the expected α_k by calculating the HMAC chain.

$$\alpha_k = HMAC(K_{BD}, seq_{k-1}| \ldots HMAC(K_{BD}, seq_{i+1}|\alpha_i))$$

If the received α'_k does not match the computed α_k, the destination can determine if B is behaving maliciously.

The chained HMACs scheme prevents nodes from lying. Additionally, it prevents nodes in collusion from dropping packets. Nearby colluding nodes may exchange information to determine the random node selection. If a packet selects a downstream cooperative node that is not in collusion to generate a report, the colluding nodes drop the packet and generate false report that they all transmit. However, they need to have proofs that the cooperative node successfully received these packets. In this case, malicious behavior can be detected since the colluders do not have the chained HMACs generated by the cooperative node. Section 6 examines the resulting computational overhead.

6. Reliability and Computational Overhead

In order to analyze the reliability of the proposed reporting schemes, we simulated our protocols using ns-2. These simulations illustrate the robustness of the RBR and other related schemes. Additionally, we explore the cost of the underlying cryptographic constructions based on empirical data.

6.1. Simulation Environments

The experimental testbed is as follows. Table 2 shows the used simulation parameters of the ns-2 simulations. Mobile nodes use IEEE 802.11 MAC with a transmission range of 250m. Additionally, the CMU scenario generation tool was used to create a network consisting of 50 mobile nodes in an 1100X1100 range. The model used random waypoint mobility with speeds of 15 m/sec. For simulating network congestion, 10 burst flows exponentially distributed were used between other nodes. Finally, a CBR traffic flow was used between the specific source and destination.

Table 2. Simulation Parameters

Simulation Time	1000 seconds
Number of nodes	50
Packet Size	512 bytes
Mobility	15 meter/second Random waypoint mobility model
Routing Protocol	Dynamic Source Routing (DSR)
Transport Protocol	UDP

The observation period is a fixed time that the destination and source observe the reports to analyze the network properly. For cases of high CBR traffic rates (28.57 packets per second), a basic observation period of five seconds was used. For low traffic rates, the period was extended inversely proportional to the traffic rate in order to gather enough information for analysis. For example, if one packet is generated per second, a five second observation time is not long enough to collect reports from intermediate nodes. Therefore, the observation time was extended.

Traffic flows are defined by the source and destination addresses. In order to adapt to the dynamic characteristic of path changes, the reports were collected based on the flow and path. This is required, because a flow may change its path due to a link failure caused by mobility or congestion. As the path changes, the source and destination keep track of the flow state, the current path state, and the active path list that consists of paths transmitting the flow traffic during the observation period.

When a node encounters a link failure and sends a ROUTE ERROR message to the source, the appropriate sequence number is included in the notification. Until the source receives the ROUTE ERROR, it continues to use the current path to transmit data. Therefore, all packets containing sequence numbers higher than the notified packet on the path causing the ROUTE ERROR are lost.

6.2. Simulation Results and Discussions

Packet loss occurs due to mobility, congestion, and malicious dropping. Identifying the source of packet loss is difficult due to the random nature of its occurrence. In order to determine the effectiveness of the secure reporting protocol, we measured the average packet loss rate over the simulation time, excluding malicious nodes. This results in 12% average packet loss (caused by congestion and mobility) from this baseline test.

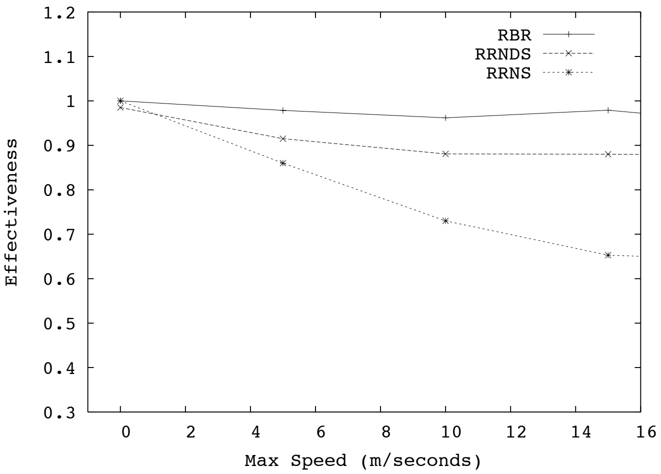


Figure 3. Effectiveness vs. Mobility

The following experiments assess the accuracy of monitored reports. Using the measured average packet loss, 12%, as a guideline, a second battery of experiments simulated an adversary that dropped 17% of the received packets. We designated any node that the reports showed to have a loss rate greater than 12% as anomalous. The efficiency of the protocol (shown in the following figures) is the percentage of experiments that correctly identify the malicious nodes as anomalous. In this conservative model, the adversary behaves only slightly different than well behaved nodes: nodes that more aggressively drop packets will be detected more easily, and the protocols will be more effective.

In the secure random reporting protocol, a report is embedded in the data packet towards the source and destination. Packet loss occurs either before or after a report is attached. In both cases, the report does not reach to the destination or source. As mobility increases or more traffic is injected, packet loss increases, which results in lost reports. Figure 3 shows the impact of mobility on the effectiveness of the three protocols. This simulation used the high bit rate CBR traffic between the source and destination, a fifteen-second attack period, and no other traffic flows. In the static case, all three protocols showed an almost 100% detection rate. However, as mobility is introduced, the effectiveness of RRNS decreases sharply since the reports embedded in datat packets are lost. By contrast, the RBR protocol remains highly effective in all experiments. In RBR, the report is transmitted to both the source and destination. The redundant transmission improves the robustness of reports in the presence of mobility.

We further simulated adversaries mounting attacks of variable length. As shown in Figure 4, RRNS was the least effective. Most of the undetected drops in RRNS occurred

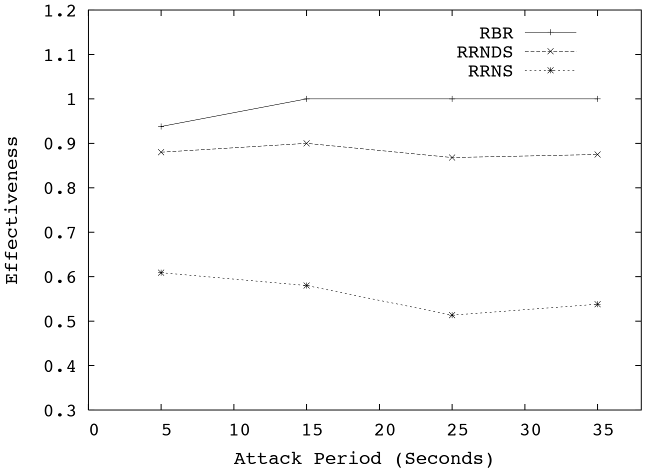


Figure 4. Effectiveness in Random Reporting Protocols

where the source changes to a new path, one intermediate node on the new path drops packets, and then the data path changes due to the following link failure. This does not allow any reports to reach the destination, and the forwarding activity cannot be discovered. The results also show that RRNDS improves RRNS's effectiveness by 1.5. The RBR protocol achieves near perfect effectiveness under attacks of all lengths.

RBR was also tested under different traffic rates. Again, as the traffic rate decreases, the observation time needed to be extended in order to collect enough reports and probe the relaying by intermediate nodes. Not shown in these graphs is the performance of RBR under the different traffic rates, which was essentially constant. From these experiments, we can conclude that the secure reporting protocol is feasible for collecting the relaying activities of intermediate nodes in civilian ad hoc networks.

The secure random reporting protocol requires three new fields: $Token$, *R*eport, and HMAC. Both the $Token$ and HMAC are the same size as the cryprographic hash output (16 bytes for MD5). The *R*eport is composed of the number of forwarded packets (4 bytes) and its cryptographic hash output (16 bytes). The total overhead incurred by the secure random reporting protocol is 52 bytes which is only 3% of maximum data packet size. On the other hand, the existing eavesdropping schemes require a separate packet transmission to informing other nodes of the reports. This separate packet transmission incurs additional transmission delay and energy consumption. Our reporting protocol removes all these extra costs by embedding a report in data packet with small overhead in packet size.

6.3. Computational Overhead

One possible concern of the Secure Random Reporting Protocol is the computational overhead of the HMAC calculation. Depending on the chosen cryptographic digest function, the overhead will vary. To test the HMAC performance, we wrote a module for the Linux 2.6 kernel, using the available cryptographic API. Tests were performed on a Pentium 3 800MHz, 512MB RAM system using a stock Linux 2.6.10 kernel compiled by GCC 3.3. The tests covered MD5, SHA1, and SHA256 digest functions with varying key sizes (64bit, 128bit, 160bit, and 256bit). All obtained results were the average raw cycle count over 1000 test runs. The high number of tests runs was chosen to ensure more accurate results.

Table 3 shows the computation time for individual calculations, which emulated the actual data used by the protocol. As described earlier, the input data for each stage of the HMAC chain consists of a sequence number and the output of a previous HMAC. Thus, the total input data size for the HMAC chain is four bytes for the sequence number and the number of bytes outputted by each cryptographic digest function (MD5 = 16 bytes, SHA1 = 20 bytes, SHA256 = 32 bytes). The HMAC data column in the table represents the overhead for performing the HMAC on the Maximum Transmission Unit (MTU), 1500 bytes. As expected, there was no performance difference for varied key sizes, therefore, the average raw cycle count for the four key sizes was used. Finally, using the `cpu_khz` value of 797556, actual time latencies were calculated.

Table 3. HMAC Computational Overhead (797,556K cycles/sec)

Algorithm	HMAC Chain	HMAC Data	Total
MD5	4.985 μs	22.832 μs	195.214 μs
SHA1	7.537 μs	36.411 μs	298.932 μs
SHA256	14.140 μs	76.430 μs	577.06 μs

In our simulation, the longest path has ten intermediate nodes. For this longest path, we estimate the computational overhead of the secure reporting protocol. The total computation consists of three factors: two HMACs of data packets at the source and destination, chained HMACs at each intermediate node, and a hash computation at each of the intermediate nodes and the destination. According to the results of HMAC computational overhead, the secure random reporting protocol and forgery detection scheme have less than 600 μseconds overhead as Table 3 shows. Under optimal circumstances, it takes 65 milliseconds to transmit data from the source to the destination (10 intermediate nodes). The total HMAC and hash computation takes only 0.8% of the total time ($577.06\mu s/65.577ms = 0.57706/65.577$).

7. Conclusions

A mobile ad hoc network does not require a fixed infrastructure and a centralized control to enable communication between mobile nodes. Most applications target mission oriented scenarios such as battle fields and emergency rescue. In these scenarios, mobile nodes actively cooperate with each other to achieve a goal. This is different than civilian mobile ad hoc networks, where nodes are not necessarily cooperative.

In this paper, we propose a secure random reporting protocol for a civilian ad hoc network, in which the source and destination collect reports from intermediate nodes on the routing path. Every data packet initiates a report from one intermediate node that is randomly chosen by a source node. Through a symmetric cryptographic construction, we ensure that the node selection is not disclosed to other intermediate nodes.

Although the report is securely transmitted to the destination, we cannot assure that it is accurate, since nodes may cheat in order to get credit. We devise a chained HMAC scheme on the link layer acknowledgments to verify the validity of the received report.

From both security and performance perspectives, the secure random reporting protocol is advantageous for gathering the forwarding activities of mobile nodes in civilian ad hoc networks. The protocol has a small communication overhead due to the increase in packet size caused by including the real time reports with data transmission. Our simulation results demonstrate the promising possibility of the reporting protocol.

Many applications require a secure and accurate report of traffic forwarding. The report can be used for determining whether congestion exists in network, engineering the traffic, crediting nodes with how many packet they relayed, and detecting that nodes maliciously drop packets.

References

[1] Y. an Huang and W. Lee. A Cooperative Intrusion Detection System for Ad Hoc Networks. *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks(SASN)*, 2003.

[2] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An On-Demand Secure Routing Protocol Resilient to Byzantine Failures . *ACM WiSe*, 2002.

[3] S. Buchegger and J.-Y. L. Boudec. Performance Analysis of the CONFIDANT Protocol(Cooperation Of Nodes: Fairness in Dynamic Ad-hoc NeTworks). *MOBIHOC*, 2002.

[4] L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. *ACM Conference on Computer and Communication Security*, 2002.

[5] http://www.isi.edu. The Network Simulator - ns-2, 2000.

[6] P. J. Transmission Control Protocol - DARPA Internet Protocol Program Specification. *IETF*, Sep. 1981. RFC 793.

[7] M. Jakobsson, J.-P. Hubaux, and L. Buttyan. A Micro-Payment Scheme Encouraging Collaboration in Multi-Hop Cellular Networks. *In Proceedings of Financial Cryptography*, 2003.

[8] D. B. Johnson, D. A. Maltz, Y.-C. Hu, and J. G. Jetcheva. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR). *http://www.ietf.org/internet-drafts/draft-ietf-manet-drIETF draft*, 2004.

[9] D. Liu and P. Ning. Establishing Pairwise Keys in Distributed Sensor Networks. *ACM Conference on Computer and Communication Security*, 2003.

[10] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. *Proc. of ACM Mobicom*, 2000.

[11] S. Micali and R. Rivest. Micropayments Revisited. *CT-RSA*, 2002.

[12] P. Michiardi and R. Molva. CORE: A Collaborative Reputation Mechanism to enforce node cooperation in Mobile Ad Hoc Networks. *In Proceedings of The 6th IFIP*, 2002.

[13] C. E. Perkins and E. Belding-Royer. Ad hoc On-Demand Distance Vector (AODV) Routing. *IETF RFC3561*, 2003.

[14] G. Vigna, S. Gwalani, K. Srinivasan, E. Belding-Royer, and R. Kemmerer. An Intrusion Detection Tool for AODV-based Ad hoc Wireless Networks. *20th Annual Computer Security Applications Conference*, 2004.

[15] X. Zhang, S. Wu, Z. Fu, and T.-L. Wu. Malicious Packet Dropping: How It Might Impact the TCP Performance and How We Can Detect It. In *Proceedings of ICNP 2000*, Nov. 2000.

[16] Y. Zhang and W. Lee. Intrusion Detection in Wireless Ad Hoc Networks. *6th International Conference Mobile Computing and Networks*, 2000.

[17] Y. Zhang and W. Lee. Intrusion Detection in Wireless Ad Hoc Networks. *Proc. of ACM Mobicom*, 2000.

[18] S. Zhone, J. Chen, and Y. R. Yang. Sprite: A Simpe, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks. *Proc. of IEEE INFOCOM*, 2003.

[19] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An Interleaved Hop-by-Hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks. *In Proc. of IEEE Symposium on Security and Privacy*, 2004.

[20] S. Zhu, S. Xu, S. Setia, and S. Jajodia. LHAP: A Lightweight Hop-by-Hop Authentication Protocol For Ad-Hoc Networks. *In Proc. of the 23rd International Conference on Distributed Computing Systems Workshops*, 2003.