Securing Ad Hoc Wireless Networks Against Data Injection Attacks Using Firewalls

Jun Cheol Park and Sneha Kumar Kasera
School of Computing, University of Utah
Email: {jcpark, kasera}@cs.utah.edu

Abstract— **We propose to secure ad hoc networks against data injection attacks by placing firewall functionality at strategic locations in the ad hoc network. We first show that, given the locations of attackers and victims, the problem of placement of firewall functionality at a fixed number of ad hoc nodes while minimizing the impact of the data injection attack is identical to the k-Coverage problem [1]. This problem is known to be NP-hard. Then, we develop a near-optimal approximate algorithm for placing firewall functions. We also incorporate the loss behavior of wireless links in our algorithm.**

Next, we develop an architecture to determine the location of the attackers. Our architecture uses a separate control network (a cellular network in this paper) in conjunction with ad hoc networks to provide a provable attack detection mechanism.

We evaluate our firewall placement algorithm for various topologies obtained from ns-2 [2] simulations. Our results show that our algorithm can find near-optimal solutions. Based on a simple analysis and measurement results, we also find that the overhead of our provable attack detection mechanism is low.

I. INTRODUCTION

Ad hoc networks are expected to play a critical role in many applications including military applications, disaster management, and community wireless networks (e.g., a rooftop wireless network [3]). They are also expected to play a greater role in social networking. One of the key impediments in the use and deployment of ad hoc networks is the concern for their security. Due to lack of an infrastructure and a well defined perimeter, ad hoc networks are susceptible to a variety of attacks. In recent years, various attacks, specifically those targeted towards ad hoc wireless networks, have been identified, and protection mechanisms to thwart these attacks and ensure security resilience [4]–[8] have been proposed.

In particular, there has been a lot of research on designing and developing secure ad hoc routing algorithms (Ariadne [9], ARAN [10], SAODV [11], SEAD [12], SRP [13], etc) that are focused mainly on securing the process of ad hoc routing (route request/reply packets). However, none of the existing work has comprehensively addressed the threat of *a data injection attack*. In a data injection attack, malicious nodes establish unrequested or unwanted ad hoc paths to some victim nodes, send undesired traffic, and thereby waste both the wireless bandwidth on links towards the victim nodes, and the local resources of the victim nodes as well. The secure ad hoc routing protocols [9]–[13] that use either secret or public key infrastructure could mitigate the impact of such data injection attacks to some extent by preventing any illegitimate nodes from establishing unwanted ad hoc paths. However, malicious nodes that are able to successfully establish secure ad hoc paths *can still launch data injection attacks*. The lack of protection against data injection attacks is the main motivation for our work.

One approach to deal with the data injection attack is to deploy local firewall functions directly at potential victim nodes. These firewalls drop unwanted packets at the victim nodes. However, this solution results in a wastage of bandwidth all along the paths from the attack nodes to the victim nodes, and also wastes local resources such as precious battery resources in battery powered ad hoc nodes. It is most beneficial if attack packets could be dropped as close to the attack nodes as possible. In this paper, *we propose to secure an ad hoc network against data injection attacks by placing firewall functions at strategic node locations*. Our solution has the following two main components which operate in concert to prevent data injection attacks.

First, assuming that the locations of attacker nodes and victim nodes are known, we provide an algorithm to determine where to place the firewall functions. To minimize the overheads of firewall functions at ad hoc nodes, firewall functions must be placed at minimal number of nodes while maximizing their efficiency. We find that the problem of placing firewall functions at a fixed number of nodes while minimizing the impact of the data injection attack is identical to the k-Coverage problem [1]. The k-Coverage problem is known to be NP-hard. Therefore, we develop a near-optimal heuristic algorithm for placing firewall functions. In our solution, we also incorporate the loss behavior of wireless links, by assigning link weights that are a function of the expected transmission time (ETT) [3].

Second, we develop an architecture to detect attacks at victim nodes and determine the locations of attackers. In our approach, victim nodes can themselves determine that they are being attacked when they receive unwanted packets. However, it is not easy for them to prove to a third party that they are being attacked by certain attack nodes. This issue is critical in preventing blackmailing attacks where ad hoc nodes construct fake attacks to make legitimate nodes appear malicious. Our architecture uses a separate control network (a cellular network in this paper) in conjunction with an ad hoc network, to provide a provable attack detection mechanism. In the past, the use of ad hoc networks in conjunction with cellular networks has been proposed to enhance performance of cellular users. However, we believe that cellular networks could also be used to control ad hoc networks, especially in aiding their security.

A base station[1] in our separate cellular control network is considered as a dependable entity (third party) which all the ad hoc nodes can trust.

Our provable attack detection mechanism requires nodes along ad hoc paths to probabilistically mark packets. A packet marking includes a keyed hash of an ad hoc node's address as well as the time at which the packet is received at that node. Each ad hoc node shares a secret key with the base station. This key is used for computing the hash. On being attacked, a victim node makes the entire snapshot of the attack packets, that are probabilistically marked by nodes along the path from the attacker(s) to the victim, to the base station. This snapshot provides a proof of the attack on an ad hoc node. The base station then uses this information for identifying attack nodes and consequently for placing firewall functions.

Using a variety of topologies obtained from ns-2 [2] simulations, we show that our firewall placement algorithm finds near-optimal solutions. Based on a simple analysis and measurement results, we also find that the overhead of our provable attack detection mechanism is low.

The remainder of this paper is organized as follows. Section II describes our problem setting. In Section III, we develop a near-optimal heuristic algorithm for placing firewall functions and evaluate its performance. We propose an architecture which uses a separate cellular network in conjunction with ad hoc networks in Section IV. Here, we present a provable attack detection mechanism. In Section V, we survey one research effort that is closest to our work on a provable detection mechanism. We conclude our work and indicate directions for future work in Section VI.

II. Problem Setting

Network Assumptions: We consider ad hoc networks that use the IEEE 802.11 [14] medium access control protocol. We assume that Dynamic Source Routing (DSR) [15] is used as the underlying ad hoc routing algorithm and that ad hoc nodes are all peers with no special nodes. It is also assumed that the integrity of data packets is preserved using the existing work [6], [16] and that their source addresses are authenticated so that attack nodes cannot spoof addresses or modify data packets without being detected.

Threat Model: We assume that malicious nodes can establish unrequested or unwanted ad hoc paths to some victim nodes, send undesired traffic, and thereby waste both the wireless bandwidth on links towards the victim nodes, and the local resources of the victim nodes as well. We do not consider the situation where two nodes, a source and a destination node, collude to simply waste wireless bandwidth along the ad hoc path between them. This attack has been studied in [17].

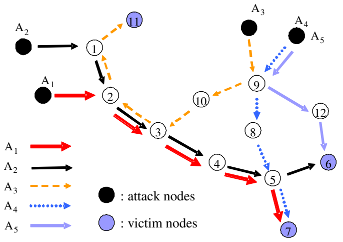

Fig. 1. Multiple attacks in ad hoc networks

Detection: In our approach, all destination nodes of ad hoc paths examine the packets they receive to detect malicious packets based on their own pre-defined attack detection rules. For detecting data injection attacks, the destination nodes use the existing threshold-based detection mechanism such as Juniper Networks' router filtering rules [18]. Although our work is not restricted to these mechanisms, developing new data injection attack detection mechanisms is beyond the scope of this paper. In the threshold-based attack detection mechanism [18], when the number of a certain type of incoming packets including ICMP, UDP, SYN, or SYN/ACK within a fixed amount of time τ, at a destination node, exceeds a threshold value N (e.g., $N/\tau = 1000$ packets/sec), the destination nodes is considered to be under a data injection attack. The attack packets could arrive from one or mode source nodes. When a data injection attack is detected at a victim node, the victim node can identify the source nodes of attack flows using the entire ad hoc path inscribed in the DSR data packets.

III. Optimal Placement Algorithm

We address the problem of placing firewall functions at a fixed number of nodes while minimizing the impact of data injection attacks. This optimal placement problem translates to the k-Coverage problem. In this section, we first describe the k-Coverage problem in general terms and later show how it applies to our ad hoc network setting.

The definition of k-Coverage problem is as follows [1].

- Instance: $F = \{S_1, S_2, ..., S_n\}$ where S_i is a subset of elements in a universal set U, $k < n$ is an integer, an associated weight function $w(u)$ for each element $u \in U$, and a set weight function $W(T)$ (= $\sum_{u \in T} w(u)$) for $T \subset U$.
- Problem: Which k subsets of F maximize the weight of the union of selected subsets?

To understand how our optimal placement problem translates to k-Coverage problem, in Fig. 1, consider the data injection scenario in the ad hoc network in Fig. 1. There are four multiple attack nodes that are injecting data packets at three victim nodes.

Let A_i denote an attack flow with flow id i, $A_i(j)$ denote the j^{th} link from the attack node in the attack flow A_i, and S_i denote a set of throttled links when node i is used as a firewall. Table I shows all the elements of S_i, for all i, and its size. In our ad hoc network context, S_i depends upon the ad hoc network topology and the location of attack flows[2], and the universal set U is a union of all S_i for $i = 1, ..., n$. Note that we exclude the immediate first links from attack nodes in S_i because they cannot be throttled at all as long as

[1] Our architecture could use any other entity in the cellular network, e.g. a radio network controller, gateway nodes, etc. This paper does not use any specific properties of cellular base stations.

[2] S_i's are not disjoint, they may have common elements.

TABLE I

TRANSLATION TO THE k COVERAGE PROBLEM

S_i	a set of throttled attack links	$\|S_i\|$
S_1	$\{A_2(2), ..., A_2(6), A_3(6)\}$	6
S_2	$\{A_1(2), ..., A_1(5), A_2(3), ..., A_2(6), A_3(5), A_3(6)\}$	10
S_3	$\{A_1(3), ..., A_1(5), A_2(4), ..., A_2(6), A_3(4), ..., A_3(6)\}$	9
S_4	$\{A_1(4), A_1(5), A_2(5), A_2(6)\}$	4
S_5	$\{A_1(5), A_2(6), A_4(4)\}$	3
S_6	{ } (victim node)	0
S_7	{ } (victim node)	0
S_8	$\{A_4(3), A_4(4)\}$	2
S_9	$\{A_3(2), ..., A_3(6), A_4(2), ..., A_4(4), A_5(2), A_5(3)\}$	10
S_{10}	$\{A_3(3), ..., A_3(6)\}$	4
S_{11}	{ } (victim node)	0
S_{12}	$\{A_5(3)\}$	1

```
General $k$-Coverage Approximate Algorithm: $GenKCover$
Input: $F = \{S_1, S_2, ..., S_n\}$ where $S_i$ is a subset of elements in a universal set $U$, $k < n$ is an integer, an associated weight function $w(u)$ for each element $u \in U$, and a set weight function $W(T)$ (= $\sum_{u \in T} w(u)$) for $T \subset U$.
$F' = F$; $C = \phi$; $I = \phi$
for (iter=0; iter< $k$; iter++) {
    select $S_i$ among $F'$ to maximize $W(S_i \cup C)$
    $C = C \cup S_i$; $I = I \cup i$
    $F' = F' - \{S_i\}$ }
Output: $C$ is a near-optimal solution and $I$ is an associated index set.
```

Fig. 2. $GenKCover$

the attack nodes continue to generate malicious packets. As a result, $A_i(1)$ for all attack flow id i in Fig. 1, is not in U. Now the problem of finding the k nodes at which firewall functions should be placed such that the weighted sum of the protected links is maximized is equivalent to the k-Coverage problem of finding which k subsets of F can maximize the weight of the union of selected subsets.

A. Approximate Algorithm, GreedyUnion

The k-Coverage problem has been shown to be NP-hard in [1]. A simple way to find an optimal solution is to use a brute-force search of all the cases of $C(n, k)$ for subset selection. However, since the brute-force search algorithm runs in an exponential order of time, it is infeasible for large n.

Therefore, we need to develop a heuristic firewall placement algorithm that runs in polynomial time while minimizing the impact of the attacks flows or maximizing the weighted sum of protected links. Fig. 2 shows a general approximate algorithm [1], $GenKCover$, for the k-Coverage problem, which will be our base algorithm to solve the optimal firewall placement problem. When an approximate algorithm guarantees to find a solution that is at least ρ times the optimal solution, it is called a ρ-approximation algorithm. The $GenKCover$ turns out to be a $(1 - 1/e)$-approximation algorithm [1]. It runs in polynomial time of $O(mnk)$ where m is the size of S_i, n is the size of F. When $k << n$, it asymptotically runs in $O(mn)$.

```
$GreedyUnion$
Input: $F = \{S_1, S_2, ..., S_n\}$ where $S_i$ is a subset of all attack links denoted by $A_i(j)$, $w(u)$ for each attack link $u$, $W(T)$ (= $\sum_{u \in T} w(u)$), and a threshold $\sigma$.
for (iter=1; iter< $maxk$; iter++) {
    $C$ = $GenKCover(iter)$ // $iter$ for $k$ in $GenKCover$
    if $W(C)/W(U) > \sigma$, then stop; else continue; }
Output: $C$ is a near-optimal solution and $I$ is an associated index set.
```

Fig. 3. $GreedyUnion$

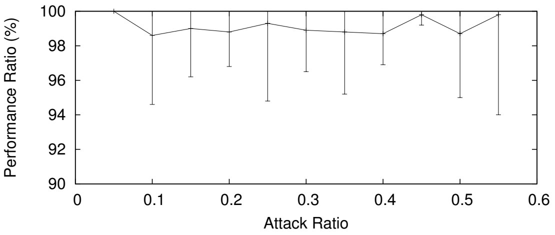

Fig. 4. Performance of $GreedyUnion$, compared to optimal algorithm

Now we refine the original k-Coverage problem so that at least σ% of the links that are under attack can be protected.

- Instance: $F = \{S_1, S_2, ..., S_n\}$ where S_i is a subset of all attack links, an associated weight function $w(u)$ for each attack link u denoted by $A_i(j)$ for all i, j, a set weight function $W(T)$, and a threshold σ.
- Problem: Given a threshold σ, which nodes are the optimal locations for firewall placement?

Fig. 3 describes the $GreedyUnion$ algorithm that solves our refined problem. The $GreedyUnion$ is still a $(1 - 1/e)$-approximation algorithm because each call to $GenKCover$ guarantees $(1-1/e)$-approximation. It runs in polynomial time in $O(mnk^2)$, where k satisfies σ. As before, when $k << n$, it asymptotically runs in $O(mn)$.

The weight function $w(u)$ of a link u can be considered as the achieved benefit when the link is protected using firewalls. We can use this weight function to incorporate different characteristics of ad hoc network link including loss, bandwidth, and delay etc.

B. Incorporation Of Loss Rates

In this work, we use the metric ETT [3] as a weight function of a link, that is, $w(u) = ETT(u)$ where $ETT(u)$ is ETT value of the link u. Since ETT corresponds to the average medium occupancy time while delivering a data packet on a wireless link, it could be considered a measure of the wasted medium occupancy time, when the link transmits a malicious data packet. With ETT as the weight function, the goal of our near-optimal heuristic algorithm is to place firewalls such that the total sum of ETT values on saved links is maximized. Refer to our technical report [19] for alternative weight functions.

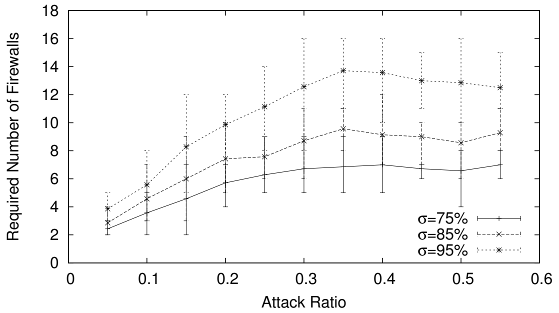

Fig. 5. Required number of firewall nodes as attack ratio increases

C. Performance of Greedy-Union

In this section, we evaluate the performance of the $GreedyUnion$ on topologies generated using the ns-2 [2] simulator. We use a total of 90 ad hoc nodes, each with a transmission range of 250 m, in a square region of 1500 m $\times$ 1500 m. Let "attack ratio" denote the ratio of the number of attack nodes to the total number of ad hoc nodes in the ad hoc network. The attack ratio ranges from [0.05, 0.55] with the interval of 0.05. In order to obtain a variety of topologies, we generate seven random node distributions for each attack ratio. In each distribution of an attack ratio, attack nodes are randomly selected among 90 nodes. We then randomly choose victim nodes and establish ad hoc paths using the DSR routing algorithm. Each generated distribution of attack flows translates to the k-Coverage problem, and is supplied as an input to the $GreedyUnion$ algorithm.

In order to compare our approximate algorithm to an optimal solution, we also implement a brute-force algorithm to find the optimal solution. We use ETT of each link as the weight function $w(u)$ in $GreedyUnion$. The ETT value of each wireless link is randomly assigned from the interval [κ, 2κ] (corresponding to the loss rates 0% to 50%, where $\kappa = S/B$, S is packet size, and B is the achievable data rate of a link). In Fig. 4, the x-axis represents the attack ratio. Using min-max-average values of seven distributions of each attack ratio, the y-axis represents the performance ratio of the solutions found by the $GreedyUnion$ to optimal solutions. The results show that the approximate algorithm, $GreedyUnion$, is able to find solutions that are in the worst case more than 94% of the optimal solutions over all cases when σ in the $GreedyUnion$ is 75%. For larger values of σ, the performance of the $GreedyUnion$ is essentially identical (not shown in this paper).

Fig. 5 shows the required number of firewalls using min-max-average values of seven distributions for each attack ratio for various σ values (75%, 85%, and 95%). Interestingly, as the attack ratio increases, the required number of firewalls does not increase after a specific attack ratio. Even for large attack ratios (> 0.3) in Fig. 5, only about 14 firewall nodes (around 15% out of 90 nodes) on average are required to throttle 95% of the total ETT values of attack links. This result shows that our firewalls placement algorithm is able to prevent multiple data injection attacks in ad hoc networks with a moderate number of firewalls only, regardless of the attack ratios.

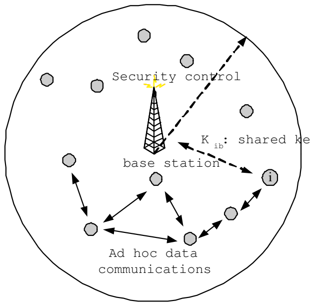

Fig. 6. Architecture using a separate security control network

IV. Architecture For Provable Attack Detection

To effectively place firewall functions at strategic nodes, there are two main issues that we must address: how to locate attack nodes, and how to prove to a third party that a victim node has been attacked. In order to resolve these issues, we propose an architecture that uses a separate control network (a cellular network) in conjunction with an ad hoc network as shown in Fig. 6. This architecture could be viewed as a hybrid wireless network in the sense that two separate networks operate simultaneously. The key idea behind this approach is to leverage a hybrid wireless network for efficient security resilience, that might not be guaranteed in completely distributed ad hoc networks.

In such an architecture, an ad hoc node is equipped with two interfaces, a wireless LAN (IEEE 802.11) interface and a wide-area cellular network interface. In the past, hybrid wireless networks have been investigated mainly for achieving higher performance and better scalability (e.g, UCAN [20]). However, in our architecture, the cellular network is used to aid security in terms of key management and in providing a dependable control entity that all ad hoc nodes can trust.

Scalable Key Management: Each ad hoc node i has a shared key K_{ib} with the base station b. As a result, the total number of shared keys in the system is in $O(n)$ where n is the number of ad hoc nodes. This is more scalable than an all pairwise key management system where each node has a distinct shared key with all the other nodes and consequently, the total number of keys required is in $O(n^2)$.

The shared keys, K_{ib}, can be either pre-deployed or dynamically re-established using an authenticated Diffie-Hellman exchange [21] as in 3G CDMA2000 1xEV-DO [22]. Since a mobile node in a cellular network already has a secret key that it shares with its network service provider, this key could be used as the pre-shared secret in the authenticated Diffie-Hellman exchange.

Dependable Entity: In our approach, the base station is considered as a dependable entity which ad hoc nodes can trust. This implies that a base station cannot be compromised, so the shared key K_{ib} can be used for the base station to authenticate data packets originated from ad hoc nodes. The base station is responsible for collecting the entire snapshot of attack flows from ad hoc nodes which declare that they are being attacked. It is also responsible for finding a set of near-optimal nodes of firewall functions using $GreedyUnion$

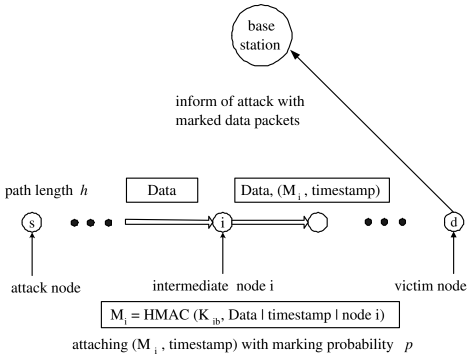
base station
inform of attack with marked data packets
path length h
Data
Data, (M$_i$, timestamp)
s
i
d
attack node
intermediate node i
victim node
M$_i$ = HMAC (K$_{ib}$, Data | timestamp | node i)
attaching (M$_i$, timestamp) with marking probability p

Fig. 7. Random marking algorithm with probability p for path length h

(Section III-A), and then securely setting up firewall functionality using the shared keys with the selected nodes.

A. Provable Attack Detection

A fundamental question in our architecture is how a victim ad hoc node can prove to the base station that it is being attacked by certain attacker nodes. The base station must make sure that malicious packets have really been delivered to a victim node from a certain source node.

We develop a provable attack detection mechanism using *a random marking scheme* as depicted in Fig. 7. The key idea of our scheme is to attach a timestamp to a data packet. A message authentication code (MAC) of the data packet, the timestamp, and the node id is computed and also attached to the packet. Nodes do not attach MACs to all packets. Instead, they do so with a probability p. This probabilistic marking scheme allows us to tradeoff the overhead of marking scheme and the number of packets marked. Incorporating timestamps is crucial not only in preventing replay attacks, but also in proving in what period of time those attack packets are sent to the victim nodes. More formally, let $mark_i(t)$ denote a 3-tuple of $(t, i, M_i(t))$ where t is a timestamp, i is an ad hoc node id, and $M_i(t)$ is the MAC associated with the mark, $mark_i(t)$. $M_i(t)$ is computed as follows:

$$M_i(t) = MAC(K_{ib}, Data|timestamp\ t|node\ id\ i)$$

where $|$ is the concatenation operator. More specifically, we use an HMAC (keyed-hash MAC) [23] function to protect the integrity of the mark[3].

When the node decides to mark a packet (with probability p), it *appends*, unlike existing probabilistic packet marking schemes [24]–[27], its own mark to the data packet which might already have marks from previous nodes. The marking node does not overwrite an existing mark. This helps in reducing the overall number of marked packets required to prove attacks. In the case of ad hoc networks, we do not have to restrict ourselves to fitting the marks in the IP header. As opposed to the universally deployed Internet, additional functionality can be relatively easily added in ad hoc networks. Already, in the DSR protocol, a data packet includes the entire ad hoc path. We also show later in this section that the overhead of appending marks is very low.

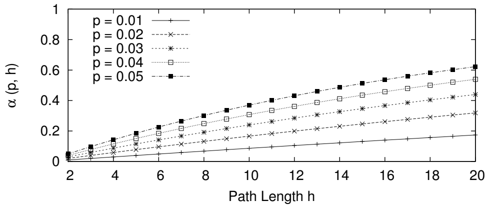

Fig. 8. Expected fraction of data packets being marked

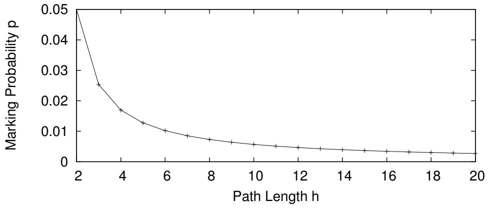

Fig. 9. Marking probability p v.s. path length h when α=0.05

A victim node stores only those data packets that are marked. However, it counts all the unwanted packets, including those that do not have any marks, towards its threshold-based attack detection. When it detects an attack flow, the victim node forwards the stored marked data packets to the base station as a provable evidence together with a request to block the attacker node. On receiving this request, and the attack evidence, the base station verifies whether the timestamps are within a legitimate time-bound, and also checks the integrity of each mark by using the secret keys, K_{ib}, that it shares with each ad hoc node in the attack path.

B. Selection Of Marking Probability

We describe a simple methodology for an ad hoc node to select the probability p with which it randomly mark packets before forwarding them. An important goal of our random marking scheme is to minimize the marking overhead while being robust enough to prove an attack detection.

Let the random marking probability be p for every node on an ad hoc path and the path length be h. Let $\alpha(p, h)$ be the expected fraction of data packets being marked. $\alpha(p, h)$ can be expressed as follows.

$$\alpha(p, h) = 1 - (1 - p)^{h-1} \tag{1}$$

This is because $\alpha(p, h)$ is 1 - the probability that a data packet does not have a single mark from any of the intermediate nodes excluding the attack node and the victim node. Fig. 8 shows the value of α as h increases for various marking probabilities from 0.01 to 0.05.

In other words, p represents the minimal random marking probability at each node when a victim node requires α fraction of data packets marked to be able to detect an attack, and prove the attack to the base station.

$$p = 1 - e^{\frac{ln(1-\alpha)}{(h-1)}} \tag{2}$$

Therefore, once we fix α to a desired value, the marking probability p of each node can be determined by Eq. (2). The

[3] The size of a mark would be 24, 28, or 40 bytes in total: 4 bytes for timestamp, 4 bytes for node id [3], and 16, 20, or 32 bytes for HMAC-MD5, HMAC-SHA1, or HMAC-SHA256 [23].

minimal values of p for each h to ensure that at least $\alpha = 0.05$ fraction of data packets are marked are shown in Fig. 9.

The next important question is how to find a desirable value of α. In order to be convinced that a certain node belongs to the attack path, the base station requires a certain minimum number of marked packets per second from that node. Let M be the average number of marked packets per second per node required by the base station. When N is the number of attack packets required to detect an attack in a given time τ, the expected total number of marked packets, α, is determined by the following equation.

$$\alpha = \frac{(h-1)M}{N/\tau}$$

Since N is likely to be large, the value of α is likely to be small. For instance, for $h = 6$, $N/\tau = 1000$, and $M = 10$, the value of α must be greater than or equal to 0.05. Once we decide the value of α, we can determine a suitable marking probability p based on α and h. In this example, $p \simeq 0.01$. Note that, since the DSR is used as the underlying ad hoc routing algorithm, every node is able to figure out the path length h of each flow, and then determine a suitable marking probability p based on the desirable value of α.

C. Overhead of Provable Attack Detection

Based on a simple analysis and measurement results, we find that the overhead of our provable attack detection mechanism is low (Refer to our technical report [19]).

V. Related Work

In this section, we mainly focus on one research effort: Authenticated Marking Scheme (AMS) [28] that is closest to our work on a provable attack detection mechanism. AMS uses authenticated packet markings for path reconstruction. However, it differs from our provable attack detection mechanism in the following significant ways. First, due to space limitations in the IP header, a router in the AMS attaches authenticated marks without including its own identity. The victim node must try all the keys associated with all the routers to identify the router that marked the packet. Our scheme includes the identity of marking node thereby simplifying the identity detection. This comes at the cost of increasing the packet size slightly but saves large amount of computation at the victim. Second, AMS uses implicit timestamps (marks are generated based on keys that are associated with specific time intervals). The time intervals are thus dependent on the key generation. We use explicit timestamps that are independent of the key infrastructure. Third, AMS requires a complete knowledge of the entire network topology at the victim node. We do not require this information in our scheme.

VI. Conclusions And Future Work

In this paper, we proposed to secure ad hoc networks against data injection attacks using firewalls. We first developed a near-optimal heuristic algorithm to place firewall functions at strategic locations in the ad hoc network. Next, we developed an architecture to determine the location of the attackers and provided a provable attack detection mechanism with minimal overhead.

We would like to study the impact of node mobility in placing firewall functions at optimal strategic positions. In this paper, we used a threshold-based detection mechanism [18], but more sophisticated detection algorithms can be further investigated for various types of attacks in ad hoc networks.

References

[1] D. S. H. et al., “Analysis of the greedy approach in problems of maximum k-coverage,” in *Naval Research Logistics*, vol. 45, 1998.

[2] “ns-2 simulator,” http://www.isi.edu/nsnam/ns.

[3] J. B. et al., “Architecture and evaluation of an unplanned 802.11b mesh network,” in *Proceedings of ACM MOBICOM*, 2005.

[4] Y.-C. H. et al., “Packet leashes: A defense against wormhole attacks in wireless networks,” in *IEEE INFOCOM*, 2003.

[5] ——, “Rushing attacks and defense in wireless ad hoc network routing protocols,” in *Proceedings of ACM WiSe*, Sept. 2003.

[6] J. K. et al., “A secure adhoc routing approach using localized selfhealing communities,” in *Proceedings of ACM MOBIHOC*, May 2005.

[7] I. A. et al., “Denial of service resilience in ad hoc networks,” in *Proceedings of ACM MOBICOM*, Sept. 2004.

[8] H. Y. et al., “Self-organized network-layer security in mobile ad hoc networks,” in *Proceedings of ACM WiSe*, Sept. 2002.

[9] Y.-C. H. et al., “Ariadne: A secure on-demand routing protocol for ad hoc networks,” in *Proceedings of ACM MOBICOM*, 2002.

[10] K. S. et al., “A secure routing protocol for ad hoc networks,” in *Journal on Selected Areas in Communication, special issue on Wireless Ad hoc Networks*, vol. 23, no. 3, Mar. 2005.

[11] M. G. Zapata, “Secure ad hoc on-demand distance vector (saodv) routing,” Aug. 2001, draft-guerrero-manet-saodv-00.txt.

[12] Y.-C. H. et al., “Sead: Secure efficient distance vector routing for mobile wireless ad hoc networks,” in *Journal of Ad Hoc Networks*, vol. I, 2003, pp. 175–192.

[13] P. P. et al., “Secure routing for mobile ad hoc networks,” in *Mobile Computing and Communications Review*, vol. 7, no. 1, Jan. 2003.

[14] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, ANSI/IEEE Std., 1999.

[15] D. B. J. et al., “Dynamic source routing in ad hoc wireless networks,” in *Computer Communications Review - Proceedings of SIGCOMM*, 1996.

[16] S. M. et al., “Mitigating routing misbehavior in mobile ad hoc networks,” in *Proceedings of ACM MOBICOM*, 2000, pp. 255–265.

[17] W. Y. et al., “Secure cooperative mobile ad hoc networks against injecting traffic attacks,” in *Proceedings of IEEE SECON*, 2005.

[18] “Denial of service and attack protection,” 2005, http://www.juniper.net/products/integrated/dos.pdf, Juniper Networks Co.

[19] J. C. P. et al., “Securing ad hoc wireless networks against data injection attacks using firewalls,” University of Utah, Tech. Rep., Oct. 2006, http://www.cs.utah.edu/˜jcpark/publications/UUCS-06-011.pdf.

[20] H. L. et al., “Ucan: A unified cellular and ad-hoc network architecture,” in *Proceedings of ACM MOBICOM*, Sept. 2003, pp. 353–367.

[21] C. K. et al., *Network Security: Private Communication in a Public World, 2nd Edition*. Prentice-Hall, 2002.

[22] Qualcomm, “1xev-do security,” 2003, white paper. [Online]. Available: http://www.wheresmyload.com/cdma/1xEV/media/web_papers/wp_security.pdf

[23] H. K. et al., “Hmac: Keyed-hashing for message authentication,” RFC 2104, Feb 1997.

[24] S. S. et al, “Practical network support for ip traceback,” in *Proceedings of ACM SIGCOMM*, Aug. 2000.

[25] D. D. et al., “An algebraic approach to IP traceback,” *Information and System Security*, vol. 5, no. 2, pp. 119–137, 2002.

[26] Q. D. et al., “Efficient probabilistic packet marking,” in *IEEE ICNP*, Nov. 2005.

[27] J. L. et al., “Large-scale ip traceback in high-speed internet: Practical techniques and theoretical foundation,” in *Proceedings of the IEEE Symposium on Security and Privacy*, 2004.

[28] D. X. S. et al., “Advanced and authenticated marking schemes for IP traceback,” in *Proceedings IEEE INFOCOM*, 2001.