RC24229 (W0704-017) April 4, 2007
Computer Science

IBM Research Report

A Secure and Efficient Traceback Mechanism in Sensor Networks

Fan Ye, Hao Yang, Zhen Liu
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

A Secure and Efficient Traceback Mechanism in Sensor Networks

Fan Ye, Hao Yang, Zhen Liu
{fanye,haoyang,zhenl}@us.ibm.com
IBM T.J. Watson Research Center
19 Skyline Drive, Hawthorne, NY 10532

Abstract

False data injection is a severe attack that compromised sensor nodes ("moles"[1]) can launch. These moles inject large amount of bogus traffic that can lead to application failures and exhausted network resources. Existing sensor network security proposals only passively mitigate the damage by filtering injected packets; they do not provide active means for fight back. This paper studies how to locate such moles within the framework of packet marking, when forwarding moles collude with source moles to manipulate the marks. Existing Internet traceback mechanisms do not assume compromised forwarding nodes and are easily defeated by manipulated marks. We propose a Probabilistic Nested Marking (PNM) scheme that is secure against such colluding attacks. No matter how colluding moles manipulate the marks, PNM can always locate them one by one. We prove that nested marking is both sufficient and necessary to resist colluding attacks. PNM also has fast-traceback: within about 50 packets, it can track down a mole up to 20 hops away from the sink. This virtually prevents any effective data injection attack: moles will be caught before they have injected any meaningful amount of bogus traffic.

Keywords: Security, Traceback, Sensor Networks, Colluding Attacks, Packet Marking, Probabilistic Marking

1 Introduction

Many wireless sensor networks are expected to work in a possibly adverse or even hostile environment. Due to their unattended operations, it is easy for an adversary to physically pick up and compromise sensor nodes, obtaining their stored data including secret keys. These compromised "moles" can launch various types of attacks, an important one of which is *false data injection* [18, 21]. One single mole can inject large amounts of bogus traffic to flood the sink, leading to application failures and wasting energy and bandwidth resources along the forwarding path. Recent research [18, 21, 17, 20, 19] has proposed a number of schemes to detect and drop such bogus messages en-route. However, they are all *passive* in that they only mitigate the damage of attacks. They do not provide active means for fight-back.

In this paper we study a crucial problem toward such active fight-back, that is, how to locate moles in sensor networks. Knowing their locations, we can isolate or remove them from the network, thus eradicating the root cause of the attack. Locating moles presents great research challenge. First, different from the Internet where routers are better protected and relatively trusted than end hosts, all sensor nodes are equally accessible by the adversary and uniformly un-protected. Any forwarding node may be compromised; there is no relatively trusted routing infrastructure that we can leverage. Second, the moles can collude. They can not only share their secret keys, but also manipulate packets in a coordinated manner to cover up their traces. Such manipulation attacks are far more sophisticated than simply increasing the amount of bogus traffic. Existing IP traceback schemes for the Internet [14, 16, 3, 6, 13, 5] do not consider such compromised forwarding nodes and become ineffective under such colluding attacks.

We propose a Probabilistic Nested Marking (PNM) scheme to locate colluding moles in false data injection attacks. We use packet marking [13] to deduce the true origin of packets: A node marks its identity in the packets it forwards. By collecting such marks, the sink can infer the route, thus the origin locations of the traffic. Although packet marking has been well explored in the Internet [6, 13, 5, 15, 10, 7], its applicability against colluding sensor moles, however, has never

[1] "Moles" are spies who operate from within an organization, especially agents operating against their own governments. We use it to refer to compromised sensor nodes.

been studied. Existing marking schemes for IP traceback can be easily defeated by an intermediate forwarding mole, which tampers the marks to hide the true locations of the source and itself, or even lead the sink to track to innocent nodes.

PNM achieves secure and efficient traceback against colluding moles using two techniques, namely *nested marking* and *probabilistic marking*. Nested marking supports single-packet traceback. Each forwarding node marks packets in a nested fashion such that its mark protects the marks from all previous forwarding nodes. This ensures that no matter how a colluding mole manipulates the marks, it either reveals the source' location, or that of its own. Probabilistic marking reduces the per-packet marking overhead to suit the resource-constrained sensors. Each node leaves a mark with certain probability, thus a packet carries only a few marks. Different from Internet marking schemes where a new mark may replace an existing one, in PNM new marks are simply appended to the packet.

We demonstrate the effectiveness and efficiency of PNM through both analytical and empirical evaluations. PNM has fast-traceback: within about 50 packets, the sink can locate a mole up to 20 hops away. It virtually prevents moles from launching effective data injection attacks: they will be caught before they can inject a meaningful amount of attack traffic. To the best of our knowledge, ours is the first work that thoroughly investigate the applicability of marking in sensor networks, and the first that can defeat the cover-up of colluding moles.

We make several contributions in this paper. First, we point out the need for *proactive* security against moles in sensor networks. We examine, within the framework of packet marking, various colluding attacks that moles can launch. Second, we thoroughly investigate the security of packet marking designs. We find that nested marking is both sufficient and necessary: if any portions of the previous nodes' marks are not protected (as in many seemingly natural designs), there exist attacks where a colluding mole can either hide the locations of the source and itself, or trick the sink trace to innocent nodes. Third, We show that a straightforward probabilistic extension to nested marking is subject to one colluding attack of selective dropping. To defeat this attack, we propose an effective probabilistic nested marking scheme where the IDs of marking nodes are anonymized.

The rest of the paper is organized as follows. Section 2 presents the network and threat models, including various colluding attacks that the moles can launch against packet marking. Section 3 demonstrates, through an existing IP traceback scheme, that straightforward employment of packet marking fails under colluding attacks. Section 4 presents nested marking and probabilistic marking, the two pieces of PNM. Section 5 analyzes how PNM withstands various colluding attacks and why nested marking is both sufficient and necessary. Section 6 evaluates the effectiveness and efficiency of PNM through analysis and simulations. Section 7 discusses a number of practical issues, and Section 8 compares PNM with the related work. Finally, Section 9 concludes the paper and discusses future work.

2 Models and Assumptions

In this section, we describe the system and threat models for our design, and present a taxonomy of malicious attacks within the packet marking framework.

2.1 System Model

We consider a static sensor network where sensor nodes do not move once deployed. These nodes sense the nearby environment and produce reports about interested events, which contain the time, location and description (e.g., sensor readings) of the events. The reports are forwarded to a sink by intermediate nodes through multi-hop wireless channels. The sink is a powerful machine with sufficient computing and energy resources.

The sensor nodes are resource-constrained and have limited computational power, storage capacity and energy supply. For example, the Mica2 motes [1] are battery powered and equipped with only a 4MHz processor and 256K memory. While public-key cryptography can be implemented in such low-end devices, it is too expensive in energy consumption. Thus we only consider efficient symmetric cryptography (e.g., secure hash functions) in our design.

We assume the routing is relatively stable. Routes do not change frequently in short time periods. When routes are stable, each node has only one next hop neighbor in its forwarding path and forwards all packets to the sink through this neighbor. This is consistent in any tree-based routing protocol [11] or geographical forwarding [8].

We also assume that each sensor node has a unique ID and shares a unique secret key with the sink. The ID and key can be pre-loaded into a node before it is deployed. The sink can maintain a lookup table for all node IDs and keys. While nodes may establish other keys for purposes such as neighbor authentication, PNM does not require such keys to work.

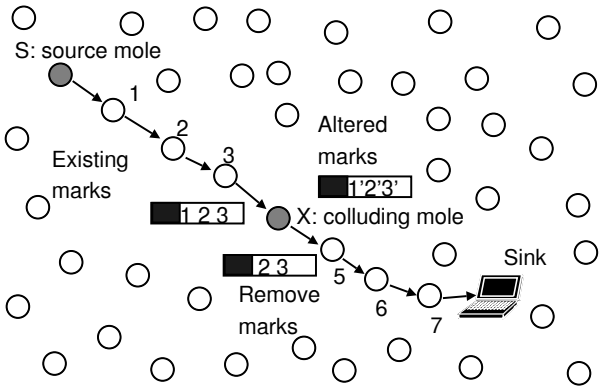

Figure 1. Moles S and X work together to cover their traces for injecting attack traffic. S injects bogus reports. X receives a packet with nodes $1, 2, 3$'s marks. X may manipulate the marks in various ways, such as altering these marks to $1', 2', 3'$, or remove the mark of node 1. The moles' goal is to hide their locations, or lead the sink track to innocent nodes.

2.2 Threat Model and Attack Taxonomy

The adversary may compromise sensor nodes through physical capture or software bugs, thus gaining full control of them. He has access to all the stored information, including secret keys, and can re-program them to behave in a malicious manner. We call such compromised nodes "moles". Moles may coordinate to maximize the damage. The sink is usually well protected. Although possible, we do not considered compromised sinks in this paper.

The context for traceback is the threat of *false data injection*. As illustrated in Figure 1, one mole S acts as a source and injects large amount of bogus sensing reports into the network. Such reports not only disrupt the user application but also waste network resources (e.g., energy, bandwidth) spent in forwarding them [18]. Traceback is the first step toward active defense. It allows the sink to identify the true origins of reports. The sink can then dispatch task forces to such locations remove moles physically, or notify their neighbors not to forward traffic from them. We leave the exact mechanism as future work and focus on traceback in this paper.

The challenge for an effective marking scheme is, a colluding mole X along the forwarding path may tamper the marks arbitrarily (see Figure 1). It can hide both its location and the source mole's location, or even trick the sink trace to innocent nodes. Hiding the locations allows continuous injection without being punished. This is needed for the injection to cause significant damage. Leaking any of their locations will lead to punishment such as network isolation or physical removal. Tricking the sink trace to innocent nodes is extra bonus: the sink may punish innocent ones, thus cutting un-contaminated resources and effectively punishing itself.

We present a taxonomy of colluding attacks against marking-based traceback by two colluding moles, S that injects bogus reports, and X on the forwarding path.

- *No-Mark Attacks*: A mole may not mark the report at all.
- *Mark Insertion Attacks*: Both the source mole and the forwarding mole may insert one or many faked marks into the reports.
- *Mark Removal Attacks*: A forwarding mole may remove existing marks left by upstream nodes in the reports.
- *Mark Re-ordering Attacks*: A forwarding mole may re-order existing marks in the reports.
- *Mark Altering Attacks*: A forwarding mole may alter existing marks in the reports and make them invalid.
- *Selective Dropping Attacks*: A forwarding mole may selectively drop those packets that, if received by the sink, would lead the traceback to them. [2]

[2] We do not consider the case where a forwarding mole drops all bogus traffic. In that case the sink cannot receive any such reports or marks, thus marking schemes are not applicable.

S	The source mole
M	Message generated by the source mole S. It contains event, location, timestamp. $M = E\|L\|T$
X	The colluding mole V_x at hop x
V_i	The forwarding nodes, at hop 1 to n
k_i	The secret key shared between V_i and the sink
M_i	The message forwarded by node V_i to its next hop neighbor V_{i+1}
m_i	The mark added by node i to M_{i-1}. It may include V_i's ID and MAC

Table 1. Notations

- *Identity Swapping Attacks*: S and X may know each other's key and impersonate each other.

For example, Figure 1 shows a chain of 7 forwarding nodes between a source mole S and the sink. Node X is the colluding mole. It receives V_3's message, which contains 3 valid marks $1, 2, 3$, left by nodes V_1, V_2, V_3. It may alter them to $1', 2', 3'$, making them invalid, thus the sink rejects these marks. It may remove mark 1 and leave only 2, 3, thus the traceback stops at innocent nodes.

2.3 Notations

To aid the presentation, we use the following notations (see Table 1) in the paper. A source mole S injects bogus reports that conform to the legitimate format. Each report M contains an event E, location L and timestamp T (i.e., $M = E|L|T$, where "|" denotes concatenation). Reports cannot all contain exactly the same content, otherwise they are considered redundant and by dropped by legitimate forwarding nodes. M is forwarded over a chain of n intermediate nodes $\{V_i\}(i = 1, ..., n)$ to the sink.

Each node V_i has a unique ID i and shares a unique key k_i with the sink. It can use its key to generate a Message Authentication Code (MAC) for the packets it generates or forwards, using an efficient and secure keyed hash function $H_k(\cdot)$, where k is the key. Specifically, V_i adds a mark m_i to the message it receives from previous hop V_{i-1} to construct its own message M_i. m_i may include V_i's ID i and MAC MAC_i. V_i then sends M_i to the next hop V_{i+1}.

Forwarding node V_x $(1 \leq x \leq n)$ is a colluding mole and we denote it X. X can manipulate the messages it receives from V_{x-1} in arbitrary manner, then pass it to V_{x+1}. It can use any one or a combination of the attacks in Section 2.2 to hide the locations of S and itself, or lead the traceback to innocent nodes.

3 Why Internet Marking Schemes Are Not Applicable

A number of marking schemes [6, 13, 5, 15, 2, 10, 7] have been proposed for IP traceback. They assume that the attacker compromises many end hosts, but usually not Internet routers. Routers simply mark the packets with their IP addresses in plain text without any security protection. Clearly, they cannot be directly applied in sensor networks where a forwarding mole can arbitrarily forge such marks. Nevertheless, the Authenticated Marking Scheme (AMS) [5] has considered compromised routers and cryptographically protects the marks. We will show that even AMS cannot withstand colluding attacks from only two moles. Our purpose is not to criticize, but rather, illustrate its strength and weakness and demonstrate why we need something different in sensor network context.

AMS protects the marks using a secure hash function. Each node shares a unique secret key with the sink. Upon receiving a packet, a forwarding node V_i probabilistically marks it with $H_{k_i}(IP_s|IP_d|i)$, where IP_s, IP_d are source and destination IP addresses. [3] In the original AMS, a packet carries at most one mark (due to the limits of available bits in the IP header). We extend it such that a packet can carry multiple marks, one from each forwarding node as $H_{k_i}(S|i)$ (Destination ID is removed as the sink is well-known in sensor network context). Consider the illustrative example shown in Figure 1, where S and X are two colluding moles. The security strength of this extended AMS is as follows.

1) It can handle no-mark attacks, because if S and X do not mark any packets, the sink will trace back to node 1 and know one of node 1's neighbors is a mole.

2) It can handle mark insertion attacks, because the marks are authenticated and the forged marks are discarded by the sink.

[3] This is one marking method as suggested by the authors in [5].

3) It cannot handle mark removal attacks. For example, if mole X removes all marks from S and node 1, the sink will trace back to innocent node 2.

4) It cannot handle mark re-order attacks. Mole X can exchange the marks of previous nodes in any order and they will still be valid.

5) It cannot handle mark altering attacks. Mole X can alter the marks of 1 and make it invalid, then the sink will trace back to innocent node 2.

6) It cannot handle selective dropping attacks. Node 4 can drop all packets containing marks left by node 1, the sink will again trace back to innocent node 2.

7) It can handle identity swapping attacks. The sink will trace back to either S or X.

Extended AMS fails because the mark added by a node does not protect its relation to marks left by previous nodes. Each mark can be individually manipulated without affecting the validity of other marks. In the following, our basic nested marking establishes a binding between each mark and all previous marks. We will also show that a probabilistic marking requires an additional feature, anonymity of IDs, to defeat selective dropping attacks.

4 PNM Design

PNM can locate colluding moles in false data injection attacks, within the precision of a *single suspected neighborhood*. This includes one node and its one-hop neighbors. A mole, either source or forwarding, is among them. PNM consists of two techniques, namely *nested marking* and *probabilistic nested marking*. The nested marking is the basic mechanism. It ensures that the sink can trace back to a mole using only one packet. However, it has a drawback of large message overhead since each forwarding node needs to place a mark on the packet. In large sensor networks this might not be efficient.

Subsequently, we use probabilistic marking to spread the message overhead over multiple packets. Each forwarding node places a mark with certain probability. Thus a packet carries only a few marks and per-packet overhead is greatly reduce. This trades off detection power for less message overhead. The sink may need multiple packets to identify the moles, which is reasonable as long as the moles are identified before they cause significant damage.

4.1 Basic Nested Marking

Packet Marking: Each forwarding node V_i appends to the packet its ID i and a secure MAC using the secret key k_i it shares with the sink. The MAC protects the *entire* message it receives from V_{i-1}. That is, $MAC_i = H_{k_i}(M_{i-1}|i)$. As an example (see Figure 1), the messages sent by neighboring nodes are:

$$
\begin{aligned}
&S -> V_1 : M \\
&V_1 -> V_2 : M_1 = M|1|H_{k_1}(M|1) \\
&V_2 -> V_3 : M_2 = M_1|2|H_{k_2}(M_1|2) \\
&\ldots \\
&V_i -> V_{i+1} : M_i = M_{i-1}|i|H_{k_i}(M_{i-1}|i)
\end{aligned}
$$

At each hop, the ID i indicates node i's presence on the route, the MAC $H_{k_i}(M_{i-1}|i)$ proves to the sink it is indeed node i that sends message M_i, and what the node receives was M_{i-1}. We can see that the MAC added by V_i protects not only its own ID but the entire message from the previous hop. This is where the name of *nested marking* comes from.

The MAC protects the IDs and MACs of all previous nodes, and their relative order. Any tampering with the previous IDs, or MACs, or their order, will make the MAC invalid. In Section 5, we will use formal security analysis to show that nested marking is *sufficient and necessary* for secure traceback. That is, it can withstand all colluding attacks, but any simpler design cannot. In extended AMS only the original message M and V_i's ID are protected, but not the mark's binding to previous marks inM_{i-1}. That is why it fails when marks are individually manipulated.

Traceback: After receiving packet M_n, the sink verifies the nested marks backwards. It first retrieves the ID of the last hop n and uses the corresponding key k_n to verify the last MAC MAC_n. If MAC_n is correct, it retrieves the ID of the previous hop $n-1$ and verifies MAC_{n-1}. The sink continues this process until either it has verified all MACs as correct, or it finds

an incorrect MAC_x. A mole (either source or forwarding) is located within the one-hop neighborhood of the node with the last verified MAC (including this node itself).

For example (Figure 1), if mole X alters the mark of node 1, marks from nodes 1, 2 and 3 will all become invalid. When X does not leave a mark or leaves an invalid mark, the traceback stops at node 5 and a mole (X) is among the one-hop neighbors of this stopping node; when X leaves a valid mark, the traceback stops at node X and the mole is this stopping node.

4.2 Probabilistic Nested Marking

The basic idea of Probabilistic Nested Marking is to let each forwarding node mark the packet with a small probability p. Thus on a forwarding path of n nodes, on average a message carries np marks. The probability p can be tuned such that the overhead of np marks is acceptable.

An Incorrect Extension: Extending to a probabilistic marking may look straightforward at first glance. However, it turns out to be non-trivial. Simply letting each node mark with probability p (see the following) is vulnerable to selective dropping attacks that can lead the traceback to innocent nodes.

$$
\begin{aligned}
&S->V_1 : M \\
&V_1->V_2(with\ p) : M_1 = M|1|H_{k_1}(M|1) \\
&V_1->V_2(with\ 1-p) : M_1 = M \\
&... \\
&V_i->V_{i+1}(with\ p) : M_i = M_{i-1}|i|H_{k_i}(M_{i-1}|i) \\
&V_i->V_{i+1}(with\ 1-p) : M_i = M_{i-1}
\end{aligned} \tag{1}
$$

Consider the example in Figure 1. Since the ID list is in plain text, the colluding mole X can see which of V_1, V_2, V_3 have marked the packet. It can drop all packets containing marks of V_1, and forward just those bearing marks from V_2, V_3. When the sink traces back, it will stop at V_2, whose one-hop neighborhood does not contain any mole. Actually, X can lead the traceback to any innocent node between itself and the source mole.

This attack works because in probabilistic marking, each packet carries only partial "samples" of nodes on the forwarding path. Due to the plain text ID, the mole can selectively pass certain "samples" so that the sink sees only a partial path ending at one of X's upstream nodes. It does not work in the basic nested marking, because every packet carries marks constituting the complete path. There exists no partial "samples" for selective dropping.

We face a dilemma here. We do not want any node be able to tell who have marked the packet. This way the colluding mole cannot know which packets to drop. However, the sink still needs to find out who have left marks to verify them. In the following, we exploit the asymmetry of the sink, extra knowledge about all secret keys and sufficient computing resources, to solve the problem.

Probabilistic Nested Marking: Instead of using its real ID i, a legitimate node V_i uses an anonymous ID i' in the packet. The mapping from real ID i to anonymous ID i' depends on the secret k_i, known by only V_i and the sink. The colluding mole does not possess the knowledge of k_i from uncompromised V_i, thus it cannot deduce the real ID from the anonymous one.

$$
\begin{aligned}
&S->V_1 : M \\
&V_1->V_2(with\ p) : M_1 = M|1'|H_{k_1}(M|1'), where\ 1' = H'_{k_1}(M|1) \\
&V_1->V_2(with\ 1-p) : M_1 = M \\
&... \\
&V_i->V_{i+1}(with\ p) : M_i = M_{i-1}|i'|H_{k_i}(M_{i-1}|i'), where\ i' = H'_{k_i}(M|i) \\
&V_i->V_{i+1}(with\ 1-p) : M_i = M_{i-1}
\end{aligned} \tag{2}
$$

In the above, $H'()$ is another secure one-way function that computes the anonymous ID. The anonymous ID i' is bound to M such that it changes for each distinct message V_i forwards. [4] This avoids a static mapping that can be accumulated over time by the attacker. Compared to the extended AMS, it has both nested marking and anonymous ID.

[4] Remember that to avoid being considered as redundant copies and dropped, reports forged by the source mole have different content.

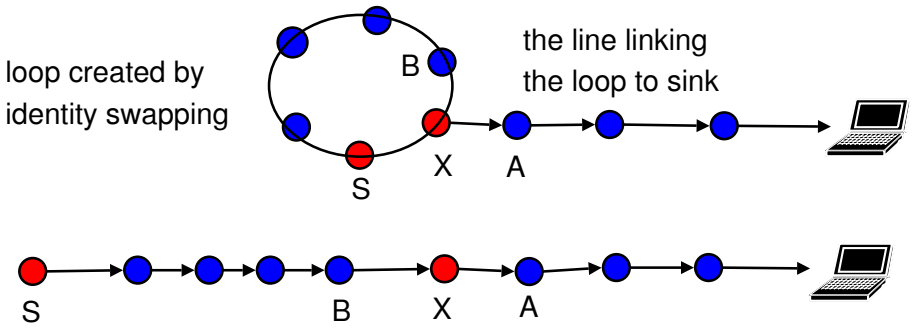

Figure 2. S and X can use each other's key to leave valid marks for some packets, thus causing a loop containing all nodes between S and X (including them) when the sink reconstruct the upstream relation among nodes. The sink can still traceback to where the loop breaks to the link and identify a mole within that neighborhood.

Mark Verification With the anonymous ID, the verification at the sink becomes different. It first needs to know the real ID, then it can use the corresponding secret key to verify the MAC. We exploit the abundant computing power at the sink: use exhaustive search to find the real ID.

After receiving M_n from node V_n, the sink first computes all the anonymous IDs for every node in the network. Knowing M, it can build a table to map all IDs i to i'. By looking up i', it knows the real ID i. Then it can use the corresponding key k_i to verify the MAC. This way, it can verify all MACs one by one.

The exhaustive search is feasible given the sink's computing power and the low data rate in sensor networks. For each distinct message M, it needs to compute a different table to do the lookup. Given that hash computation can be done at microsecond level (e.g., an Athlon 1.6G CPU can do 2.5 million hash per second[5]), building such a table for even a reasonably large network (a few thousand nodes) should take on the order of a few milliseconds. Thus the sink can verify several hundred or more packets per second. Because the sink receives from one sensor at a time, the incoming data rate is limited by the radio rate of sensors. Several hundred packets is already much higher than the current actual data rate on typical sensor hardware (e.g., 12kbps for Mica2 motes, under 100 packets per second[6]).

Traceback Locating moles becomes a two-step process. First the sink needs to reconstruct the route by collecting marks from a sufficient number of packets (the exact number will be analyzed in Section 6). Then it identifies which nodes have moles in their one-hop neighborhood. Algorithm 1 (pseudo code in Appendix 10.3) describes how the sink can locate moles. We briefly explain the main idea here.

The route can be reconstructed by finding the relative order of nodes (which is upstream to which) in the forwarding path. We use a matrix M to maintain the relative orders. The matrix is initially empty. When a correct MAC for a new node V_i is verified, one more row and one more column corresponding to V_i is added to the matrix. Whenever two consecutive MACs MAC_i, MAC_j within one packet are verified as correct, V_i should be upstream to V_j, and $M[i, j]$ records this relation (e.g., be set to 1) in the matrix. As more packets are received, the sink keeps updating this matrix. Given sufficient packets, the sink will be able to find out the upstream relation among all forwarding nodes, thus the complete route.

The sink will be able to reconstruct two types of routes: those that do not have loops, or those have loops. The first type happens when moles use attacks other than identity swapping, the latter when moles swap their identities to leave marks. In the first case, locating moles is equivalent to finding the most upstream node. Because a source mole produces packets by itself, it does not receive packets from others and it can be the most upstream node. A forwarding mole may "appear" to be the most upstream, if it removes marks left by its upstream nodes. In either case, a source or forwarding mole, is within the one-hop neighborhood of the most upstream node.

The moles may use identity swapping to create loops (see Figure 2), thus there does not exist a "most" upstream node. A source mole S and a forwarding mole X may leave valid marks the key of each other for some packets, and use their own keys for some other packets. The sink will find that S appears before X for some packets, and after X for other packets.

[5]These numbers are based on the measurement shown in http://www.azillionmonkeys.com/qed/hash.html
[6]http://mail.millennium.berkeley.edu/pipermail/tinyos-help/2003-June/001496.html

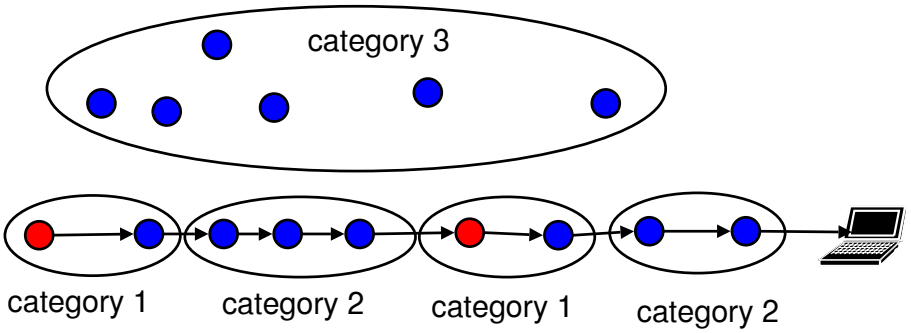

Figure 3. There are two categories of nodes on the forwarding path. Category 1: moles and their immediate next hop; Category 2: legitimate nodes that have immediate previous hop legitimate neighbor. Due to consecutive traceability, the traceback cannot stop in category 2. For category 3 nodes (legitimate nodes not on the forwarding path) they do not leave marks for message they do not forward. Thus the traceback can stop only in category 1 nodes.

It will also find that all nodes between S and X (including them) form a loop. For any two nodes U, V in such a loop, U appears both upstream and downstream to V.

However, this anomaly can be easily identified: the sink can find the rest of the nodes form a line from the loop to itself. A mole is located within the one-hop neighborhood of the most upstream node in this line (i.e., where the loop intersects with the line). We will present detailed analysis in Section 5.3.

5 Security Analysis

In this section, we analyze the security strength of PNM and compare it to alternative marking schemes. Our analysis shows that nested marking is both precise and necessary: It can track down moles to within one-hop neighborhood area despite colluding attacks, but any simpler design fail under certain attacks. The probabilistic nested marking can track down moles within one-hop neighborhood area *asymptotically* as the sink receives sufficient number of packets over time.

5.1 Security of Nested Marking

We first define two properties for marking schemes, namely *one-hop precision* and *consecutive traceability*, and then prove that they are equivalent. Then we prove that our basic nested marking scheme is one-hop precise by showing its consecutive traceability.

Definition 5.1 (One-hop precision): *A marking scheme has one-hop precision in traceback if it can always trace to either the source node's or a colluding mole's one-hop neighborhood.*

Definition 5.2 (Consecutive Traceability): *Consider two consecutive legitimate nodes U and V on a forwarding path (i.e., V receives messages from U and then forwards them). With a consecutive traceable marking scheme, if the sink has traced to V, it can always further trace to U.*

Theorem 1 *A marking scheme is one-hop precise if and only if it is consecutive traceable.*

Proof: We first prove the sufficiency. Suppose that the traceback stops at a node V, which is the last node (in the reverse order of forwarding) that has a valid MAC. V cannot be a legitimate node that is not on the forwarding path, because such nodes will not generate MACs for messages they do not forward, and the attacker does not know their secret keys. Thus, V is a either a mole, or a legitimate node on the forwarding path. If V is a mole, the sufficiency holds. Next we consider the case where V is a legitimate forwarding node.

Let U be the previous hop of V, i.e., V receives messages from U. There are only two possibilities: either U is a mole (source or colluding) or U is a legitimate node. In the first case, the sufficiency holds because V is in the neighborhood of a

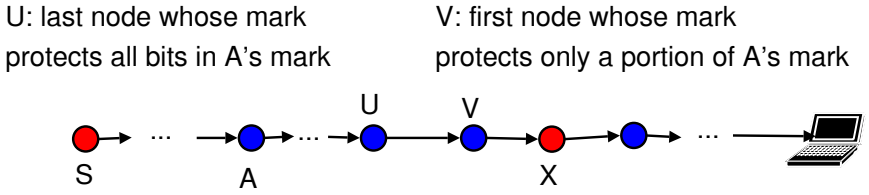

Figure 4. X alters the bits in A's mark not protected by V. Thus the V's mark is still correct, but not U's. Then the sink traces back to V, but cannot further trace to U.

mole U. On the other hand, by definition of consecutive traceability, the traceback will proceed to U and will not stop at V. Thus the second case cannot happen. This concludes the proof of sufficiency.

Next we prove the necessity by contradiction. Suppose a marking scheme is not consecutive traceable. That is, there exists cases when the sink has traced to a legitimate node V, but it cannot proceed to the previous legitimate node U. Thus the traceback stops at V, not necessarily the neighborhood of the source or a colluding mole. The the scheme is not one-hop precise by definition. □.

The intuition is (see Figure 3 for illustration), there are only two categories of nodes on a forwarding path: moles and their immediate next hop, and legitimate forwarding nodes that have legitimate previous hop neighbor. One-hop precision means the traceback stops within the first category; consecutive traceability means the traceback cannot stop within the second category – that is, it has to stop within the first one.

Theorem 2 *The nested marking scheme is consecutive traceable.*

Proof: Consider two consecutive legitimate forwarding nodes U and V. Let M_u be the message that U sends to V, and V sends $M_u|V|H_{k_v}(M_u|V)$ to the next hop.

Suppose the sink has traced to V. This means that it should have verified MAC'_v in a message $M'_u|V|MAC'_v$, and found that the recomputed MAC $(H_{k_v}(M'_u|V))$, is the same as the included MAC'_v. Because the attacker does not know k_v, MAC'_v must be the MAC_v generated by V. Thus M'_u and M_u must be the same; otherwise, the recomputed MAC would not match that produced by V.

Because M_u is sent by a legitimate node U, the last mark in M_u must carry a valid MAC from U. Therefore, by verifying this MAC, the sink can further trace to U. □

Corollary 5.1 *The nested marking scheme is one-hop precise.*

We enumerate the detailed case-by-case analysis of how nested marking resists different attacks in Appendix 10.1.

5.2 Necessity of Nested Marking

Theorem 3 *Any marking scheme that protects less fields than the nested marking is not consecutive traceable.*

Proof: In the nested marking, a node's MAC protects both its own ID and the entire message it receives from the previous hop. Now consider an alternative marking scheme Γ, in which the MAC protects less fields than the nested MAC. There must exist a node A, whose ID or MAC is not completely protected by all nodes after it; otherwise, Γ would become the nested marking scheme.

Let U be the last node that protects A's ID and MAC completely, and V be the next hop of U (See Figure 4). That is, there are some bits in A's mark not protected by V's MAC. Let us consider one mole downstream after V. The mole properly marks the report, and it alters the bits in A's mark not protected by V's MAC. In this case, the MACs of all nodes after V (including V) are correct, thus the sink can trace to V. However, because A's mark is tampered by the mole, U's MAC would appear invalid, thus the sink cannot further trace to U. In other words, Γ is not consecutive traceable. □

Corollary 5.2 *Any marking scheme that protects less fields than the nested marking is not one-hop precise.*

In Section 3, we have shown the failure of the extended AMS where a node's MAC only protects its own ID. In Appendix 10.1, we briefly describe two alternative schemes that protect only the IDs or the MACs and show how they fail under certain colluding attacks.

5.3 Security of Probabilistic Nested Marking

Theorem 4 *The probabilistic nested marking is asymptotically one-hop precise if the routes are stable.*

Proof: There are two possible cases when the sink reconstructs the upstream relation among nodes: either there is no loop, or there are loops.

When there is no loop, the proof is similar to that of Theorem 2. Let us consider two consecutive legitimate nodes U, V on the forwarding path. Due to the probabilistic marking, these two nodes may not always both leave marks in the packet. However, with enough number of packets, the probability that they do not both leave marks in the same packet, $(1-p)^{2n}$, becomes smaller and smaller as the number of packets n increases. Asymptotically, there will be packets where both U, V leave marks. A forwarding mole might be able to drop such packets. However, because of the use of anonymous IDs, it cannot always correctly guess and drop all such packets. Asymptotically, the sink will receive packets with marks from both U and V. Following similar reasoning in Theorem 2, once the sink has traced to V, it can further trace to U. Thus the traceback will not stop at V. Therefore, the traceback must stop at the most upstream node, which has moles within its one-hop neighborhood (including this node itself).

When there are loops, we prove that the joining point (node X) of the loop and the line (see Figure 2) must have moles within its one-hop neighborhood (including this node itself) by contradiction. Suppose all the 4 nodes within this one-hop neighborhood (X, S, A, B) are legitimate nodes. Because packets arrive at the sink from X to A, A must be the next hop neighbor on X's forwarding path. Because the loop also represent upstream relation among nodes, X must also forward packets to one of its neighbors on the loop (S or B). Thus X has two next hop neighbors on its forwarding path. However, any legitimate node should have only one next hop neighbor on its forwarding path when routes are stable. Thus these 4 nodes cannot all be legitimate nodes and one of them must be a mole. □

The intuition behind this proof is that there must be some abnormal behavior around where the loop connects to the line. For legitimate nodes, they do not form loops when routes are stable. Thus such abnormal behavior can only be explained by the presence of moles. Note that moles may also launch identity swapping attack in the basic nested marking scheme. However, with basic nested marking, since all nodes leaves marks in each packet, the sink does not need to traceback through the upstream relationship and it can always trace back directly to moles.

6 Performance Evaluation

6.1 Analysis

We first analyze N, the number of packets needed for the sink to collect at least one mark from each of the forwarding nodes $V_1, ..., V_n$. We can compute (details in Appendix 10.2) the probability that this is done within L packets is:

$$P(N \leq L) = (1 - (1-p)^L)^n \tag{3}$$

Figure 5 illustrates the probability that at least one mark from n nodes is collected within x packets. The average number of marks np a packet carries is fixed at 3. For a path containing 10 nodes, after receiving 13 packets, the sink has about 90% probability of having collected all marks. It takes 33 and 54 packets to achieve the 90% confidence for paths of 20, 30 hops. The results show after a relatively small number of packets, which have not wasted significant energy and bandwidth resources, the sink will have collected marks from all nodes.

6.2 Simulation Results

We run simulations to verify the analysis and further evaluate metrics that are difficult to analyze. The probability p is tuned for different path lengths n such at a packet carries 3 marks on average.

We first verify the analysis results. We set the number of nodes n to $10, 20, 30$. 200 packets are generated from the source in each run and we average the results over 5000 runs. Figure 6 shows the probability that marks from all nodes are collected after the sink receives x packets. We can see the result matches that of the analysis (Figure 5) very well. Figure 7 shows the portion of nodes whose marks are collected by the sink after x packets. When there are 10 nodes, on average 9 nodes' marks

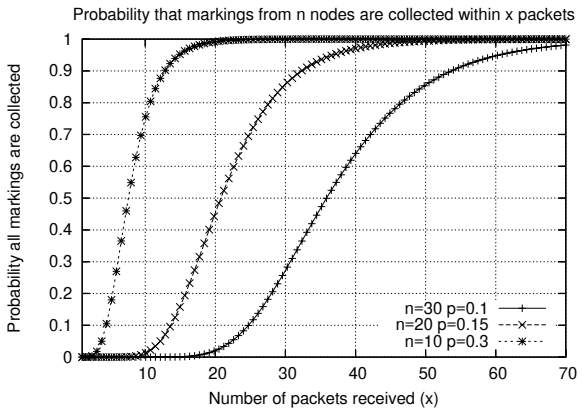

Figure 5. Analytical result of the probability that marks from all the n forwarding nodes are collected after the sink receives x packets. Within several dozen packets, the sink will have collected marks from all nodes.

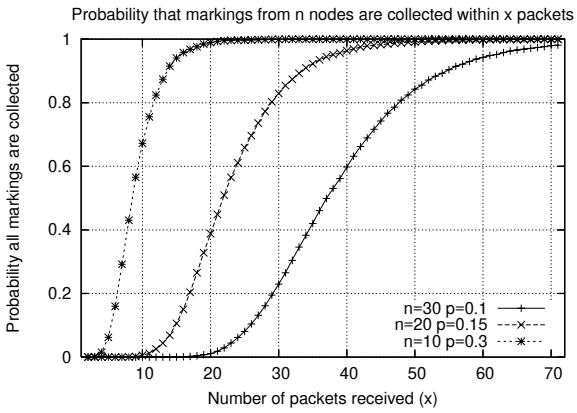

Figure 6. Simulation results to verify the analysis. It matches the results in Figure 5.

can be received with 7 packets. For paths of $20, 30$ nodes, it takes about $14, 22$ packets to collect marks from 90% of the nodes. Within a few dozen packets, the sink knows which are the forwarding nodes.

We also evaluate the performance of Algorithm 1. Figure 8 shows how the candidate source set changes as more and more packets are received for a run of 40-node path, node 0 being the source mole. At first node 8 is in the candidate list. As more marks are received, new nodes $8, 11, 9, 18$ at the beginning of partial paths are added. As their upstream nodes are discovered later, they are removed. The real source 0 is added on the $21th$ packet. After the 80th packet, no more candidate node other than 0 remain in the set. When the candidate set remains unchanged for a long time, the sink can *unequivocally* identify a mole.

Without sufficient numbers of packets, the sink may not be able to unequivocally reduce the candidate source set to only real moles. To test how many packets are needed, we change the number of packets the sink receives as 200, 400, 600 and 800. For each traffic amount, we run the simulation 100 times over each of 10 different path lengths from 5 to 50. We obtain for how many times the sink does not unequivocally identify the source. Figure 9 illustrates the number of failed runs as a function of total path length, for the 4 different traffic amount.

We can see that 200 packets are sufficient for path lengths up to 20. The algorithm almost always unequivocally identify the source in each run. 400 packets are enough for paths up to 30. Only for very long paths (e.g., 50 nodes), a large number of packets (e.g., 800) are needed to reduce the failure frequency to less than 5%.

We choose 800 as the traffic amount to measure the average number of packets the sink should receive to unequivocally identify the source. Figure 10 shows the results as a function of total path length, over all runs that successfully identify the source. For path lengths less than 20, on average it takes about 55 packets to unequivocally identify the source. This roughly match the result in Figure 6, where with 55 packets, the sink has over 99% probability of having collected marks from all the 20 nodes. Even for long paths such as 40 nodes, after about 220 packets the sink can unequivocally identify the source. The results demonstrate that PNM almost prevents moles from launching effective false data injection attacks: they will be located before they have caused sufficient damage to the network.

7 Discussions

In this section, we discuss a number of issues in our design.

Traceback Precision PNM can traceback moles up to one-hop neighborhood, which includes the node where traceback stops and its one-hop neighbors. One of them must be a mole, either a source, or a forwarding one. The precision is not single node because a source mole can claim different identities when injecting reports. Its next hop neighbor cannot tell which identity is true. PNM does not require pairwise keys between neighbors to work. However, the existence of pairwise keys may help improve the traceback precision.

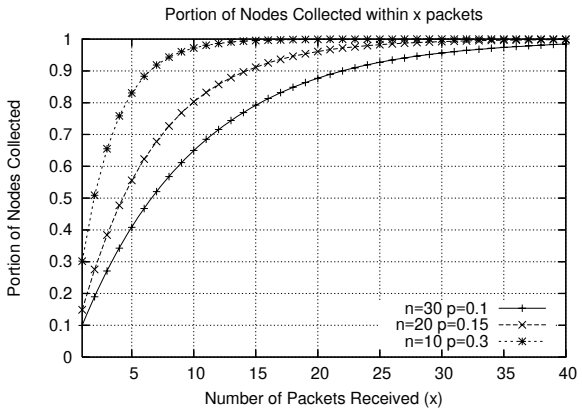

Figure 7. The average number of nodes whose marks are collected by the sink in the first x packets.

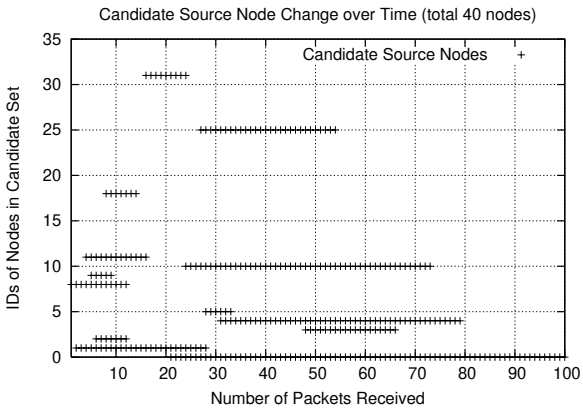

Figure 8. The candidate source set changes over time for a 40 nodes path. The Y axis is the ID of nodes in the set, the X axis is the number of packets received. The set grows first, but eventually is reduced to only the real source 0.

PNM tracebacks one mole at a time. We do not attempt to track down all moles within one round. The expectation is that some mole isolation mechanism will work together with PNM. It isolates the mole identified in each round. Thus over time, these moles are isolated from the network one by one. We will further investigate the mechanisms to nail down the exact mole, and the mechanisms we can use for mole isolation.

Background Traffic PNM provides means for the sink to traceback. However, it does not tell the sink which packets are produced by moles. This can be important because legitimate traffic may co-exist. The sink needs to know which packets are suspicious to decide which locations are possibly the moles. This might be done through other mechanisms, such as verifying whether reported events do exist, or certain characteristic of traffic such as volume and traversed route (e.g., attack traffic is usually large amount and follow certain routes). A thorough investigation on this issue is beyond the scope of this paper and we leave it as future work.

Impact of Routing Dynamics The mole locating algorithm works well when the route is relatively stable. Since moles usually inject large amount of traffic in short time to maximize the damage, collecting sufficient packets does not need too much time. For example, if moles inject at maximum radio rate, within ten seconds the sink can collect about 300 packets, enough to locate moles 40 hops away. The route is very likely to remain stable during this short period of time.

When dynamic routes change very fast, it may affect the locating algorithm. Because different sets of forwarding nodes may be used, the upstream relationship among nodes may not be consistent for different packets. The sink might have difficulty to reconstruct the original routes. However, as long as the route change is not too frequent, or the upstream relation among nodes remain the same when routes change, the sink is still able to locate the moles one at a time.

Replay Reports The source mole may replay reports with the same content multiple times, thus the mapping between real and anonymous IDs will stay fixed for each forwarding node. The colluding mole may even accumulate such mappings over time. However, such attacks can be easily thwarted by local suppression of redundant messages. A forwarding node can simply drop reports of the same content. This can be done by maintaining the signatures (e.g., a hash result) of the content (i.e., the event, location and timestamp) of recently seen messages and comparing those received message against them.

The source mole may also replay legitimate reports from a real reporting source node. The report content still presents correct information. To detect and drop such messages, the same local suppression can be used. There also exist other techniques. One of them is to require each node maintain an increasing sequence number for each message it sends or forwards; it includes the sequence number as part of the mark and protects it using the MAC. The sink can detect that replayed packets have the same sequence numbers. We do not elaborate the details here due to space limit.

Miscellaneous Issues There are some other issues that affect PNM and we spell them out for a complete picture. We do not consider mobility in PNM. An attacker may carry a mole and move around to inject traffic. PNM does not address such attacks if the mole moves too fast: the sink may traces back to its old location while it has moved to a new place. launch attacks on legitimate packets such as dropping or altering them. These attacks are different from the mark manipulation attacks for colluding moles and we do not consider them in this paper.

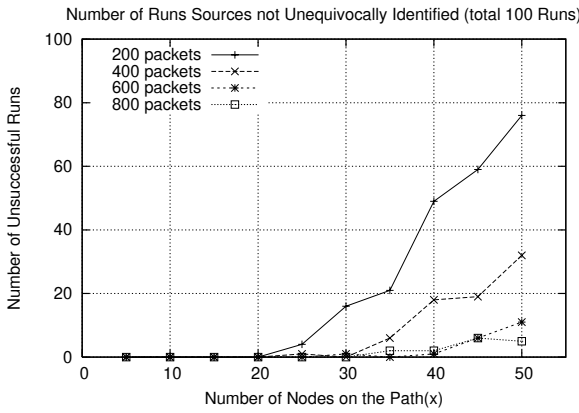

Figure 9. In 100 runs, how many runs fail to unequivocally identify the source, as a function of total path length. The four curves corresponds to 200, 400, 600 and 800 packets in each run.

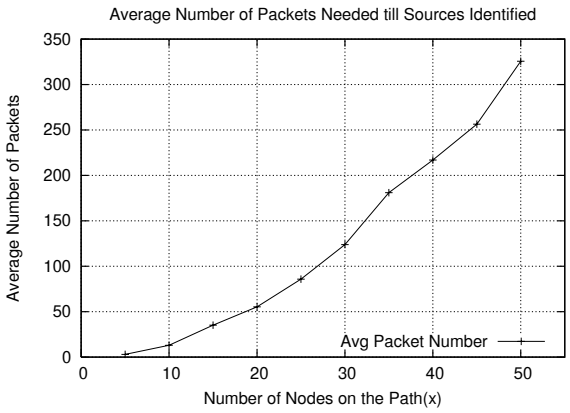

Figure 10. The average number of packets needed to unequivocally identify the source, as a function of total path length. 800 packets are received at the sink in each run.

Because nodes use different keys, the anonymous IDs of two different nodes may collide (e.g., $H'_{k_i}(M|i) = H'_{k_j}(M|j)$). A cryptographically sound hash function can minimize such collisions; when they do happen, the sink can simple exclude such packets from verification. In PNM, moles can also inflict higher per-packet overhead by inserting more bogus marks or using a higher probability than instructed. However, this causes only some reduction in the efficiency; the sink can still traceback to the moles.

There may exist multiple source moles which all send attack traffic in the same time. The basic nested marking can still identify them because each packet carries marks from all forwarding nodes. When the forwarding paths of these multiple source moles are disjoint, the PNM can construct different paths individually and locate the moles. These moles may also swap their identities to create loops, but the straight lines that link the sink to these loops can still be used to identify these moles one at a time.

8 Related Work

The false data injection attack is an important security problem in sensor networks. Several en-route filtering schemes [18, 21, 17, 20, 19] have been proposed to drop the false data en-route before they reach the sink. However, these schemes only mitigate the damage. First, none of them can achieve perfect filtering. Second, filtering does not prevent moles to continue injecting bogus reports. Even these reports are dropped after a few hops, they still waste the energy resource of legitimate nodes. Our traceback scheme complements filtering by locating the attacking moles. This makes it possible to physically remove or isolate such moles from the network, thus eradicating the root cause of the attacks.

A number of packet marking schemes [6, 13, 5, 15, 2, 10, 7] have been proposed for IP traceback in the Internet. They usually do not assume compromised forwarding nodes (i.e., routers) and are not designed to handle colluding moles on forwarding paths. Such moles can tamper the marks and trick the sink to trace to wrong nodes. Even the authenticated IP marking scheme [15] that considers compromised routers cannot withstand all colluding attacks. Our work is specifically designed to handle such colluding attacks from compromised forwarding nodes.

Besides packet marking, there are two other approaches for traceback, namely logging and notification. In logging schemes [14, 16], forwarding nodes store information about the packets they have forwarded. The sink can construct the path, thus locating the source of a packet, by querying which nodes have forwarded it. The notification schemes [3, 4] takes a different push model, where a forwarding node probabilistically notify the sink packets they are forwarding (e.g., using ICMP messages). PNM differs from them in two aspects. First, it requires no control messages such as query/reply or notification. Securing these signaling mechanisms and preventing moles from injecting bogus control messages is a challenging task. Second, it does not require a node to store previously forwarded packets. Such space can be better utilized for legitimate data on low-end sensors with limited storage capacity.

Techniques similar in spirit to nested marking have been used in [9, 12] for anonymous routing of messages. They address a reverse problem: how to prevent attackers from tracing back to the sender or receivers of messages. They also use public key cryptography to construct the routing "onions." We study the problem of tracing back to senders of messages, and we do not require any public key cryptography.

9 Conclusions and Future Work

False data injection attack has been studied by a number of work [18, 21, 17, 20, 19] since it was proposed. All existing work are passive in that they only mitigate the damage of attacks. Probabilistic Nested Marking is the first work that can locate moles despite colluding attacks. Combined with physical removal or network isolation, it can be used to actively fight-back moles. We have proved its security against colluding moles and demonstrated its efficiency with analysis and simulation. It can track down a mole up to 20 hops away from the sink using about 50 packets. This essentially prevents effective data injection attacks: moles are caught before they have caused any meaningful damage to the network.

We plan to further study the problem along several directions. First is to evaluate PNM's efficiency with actual implementation. This can help us understand how practical issues such as limited packet length and packet losses affect the performance of PNM, and how we can choose parameters such as marking probability appropriately. Second is how to further improve the traceback precision from one-hop neighborhood to single node. We conjecture that marking alone might not be sufficient. Because a source mole may not leave any information relating to its identity in injected messages, mechanisms such as collaborative local monitoring might be needed. The sink can notify nodes within the suspected neighborhood to collectively monitor and detect the source mole. Then it can instruct them not to communicate with the mole. Exactly how to notify nodes to monitor, isolate, how they perform local monitoring, and how to secure such mechanisms, should be studied for a complete fight-back solution. Finally, marking provides piece-by-piece information. The sink has to put together a jigsaw puzzle to see the complete picture. It would be interesting to see how information about network topology might be used to locate moles more efficiently.

References

[1] Xbow sensor networks. http://www.xbow.com/.

[2] M. Adler. Tradeoffs in Probabilistic Packet Marking for IP Traceback. In *ACM Symposium on Theory of Computing*, 2002.

[3] S. Bellovin. ICMP Traceback Messages. In *Internet Draft: draft-bellovin-itrace-00.txt*, 2000.

[4] H. Burch and B. Cheswick. Tracing Anonymous Packets to Their Approximate Source. In *USENIX Large Installation System Administration Conference*, 2000.

[5] D. Dean, M. Franklin, and A. Stubblefield. An Algebraic Approach to IP Traceback. In *Network and Distributed System Security Symposium*, 2001.

[6] T. Doeppner, P. Klein, and A. Koyfman. Using Router Stamping to Identify the Source of IP Packets. In *ACM Conference on Computer and Communications Security*, 2000.

[7] Q. Dong, M. Adler, S. Banerjee, and K. Hirata. Efficient Probabilistic Packet Marking. In *IEEE International Conference on Network Protocols*, 2005.

[8] B. Karp and H. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *ACM MOBICOM*, 2000.

[9] J. Kong and X. Hong. Anodr: Anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In *ACM MobiHoc*, 2005.

[10] J. Li, M. Sung, J. Xu, and L. Li. Large-Scale IP Traceback in High-Speed Internet: Practical Techniques and Theoretical Foundation. In *IEEE Symposium on Security and Privacy*, 2004.

[11] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. TinyDB: An Acquisitional Query Processing System for Sensor Networks. *TODS*, 2005.

[12] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16(4), 1998.

[13] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical network support for IP traceback. In *ACM SIGCOMM*, 2000.

[14] A. Snoeren, C. Partridge, L. Sanchez, C. Jones, F. Tchakountio, S. Kent, and T. Strayer. Hash-Based IP Traceback. In *ACM SIGCOMM*, 2001.

[15] D. Song and A. Perrig. Advanced and Authenticated Marking Schemes for IP Traceback. In *IEEE Infocom*, 2001.

[16] R. Stone. CenterTrack: An IP Overlay Network for Tracking DoS Floods. In *USENIX Security*, 2000.

[17] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh. Toward Resilient Security in Wireless Sensor Networks. In *ACM Mobihoc*, 2005.

[18] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical En-Route Filtering of Injected False Data in Sensor Networks. In *IEEE Infocom*, 2004.

[19] Z. Yu and Y. Guan. A Dynamic En-route Scheme for Filtering False Data. In *IEEE Infocom*, 2006.

[20] W. Zhang and G. Cao. Group Rekeying for Filtering False Data in Sensor Networks: A Predistribution and Local Collaboration-Based Approach. In *IEEE Infocom*, 2005.

[21] 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 *IEEE Symposium on Security and Privacy*, 2004.

10 Appendix

10.1 Examles for Two Corollaries

In this section, we give examples to illustrate the following two corollaries in Section 5.

Corollary 5.1 *The nested marking scheme is one-hop precise.*

As examples, we use the topology shown in Figure 1 to illustrate how the nested marking scheme can withstand the colluding attacks, as we have outlined in Section 2. In such attacks, node S and node X are two colluding moles.

1) *No-Mark Attacks*: If S does not mark, the sink will trace to node 1, which is the one-hop neighborhood of S. If mole X does not mark, the sink will still trace to 1, though it sees an incorrect route that misses mole X.

2) *Mark Insertion Attacks*: If mole X inserts a forged mark and claim it is from a legitimate node A, the sink will trace to node X, because the MACs of nodes X to 7 are all correct but the MAC of node A is wrong. Note that mole X may choose to not mark itself, in which case the traceback stops at node 5, which is in X's one-hop neighborhood. mole X can also insert a correct mark on behalf of another mole, say B. In this case, the traceback will proceed to S, even though the sink sees an incorrect route with node B in place.

In general, mole X can insert multiple marks to forge a route segment $V_1 \rightarrow ... \rightarrow V_k$ between nodes 3 and X. If the MAC of V_k is wrong, the traceback will stop at mole X (or node 5 if mole X chooses not to mark itself). Otherwise, V_k must be a colluding mole whose secret key is known to mole X. In this case, if the MAC of V_{k-1} is wrong, the traceback will stop at V_k; otherwise, V_{k-1} is also a mole and similar process continues. At the end, either all nodes in this segment are moles and the sink traces to S, or there is one legitimate node in the segment and the traceback stops at the previous mole. The same conclusion holds for the case where node S inserts forged marks. Note that X cannot insert marks before 3, because then 3's MAC will appear invalid and the traceback will stop at X.

3) *Mark Removal Attacks*: Due to the nature of nested marking, if mole X removes a previous mark, all marks between that removed node and mole X become invalid. Thus, if the mark of node 3 is still present, the sink will trace to mole X because the MACs of nodes X to 7 are all correct but the MAC of node 3 is wrong. Note that mole X may remove the mark of node 3, in which case the MACs are all correct and thus the sink can trace to node S.

In general, mole X can remove multiple marks from previous nodes. However, the traceback will stop at node X because node 3's MAC becomes invalid, unless mole X removes a consecutive segment immediately before it (e.g., both nodes 2 and 3). In the latter case, the sink will trace to node S.

4) *Mark Re-ordering Attacks*: If mole X changes the order of previous marks in any manner, the MAC of nodes 3 becomes invalid, but the MACs of nodes 5 to 7 are all correct. Thus, the sink will trace to either mole X or node 5, depending on whether mole X appends a correct mark.

5) *Mark Altering Attacks*: Similarly, if nodes alters the previous marks in any manner, the MAC of nodes 3 becomes invalid, and the sink will trace to either mole X or node 5.

Corollary 5.2 *Any marking scheme that protects less fields than the nested marking is not one-hop precise.*

In Section 3, we have shown the failure of the extended AMS where a node's MAC only protects its own ID. In what follows, we briefly describe other alternative schemes and why they are not complete against all colluding attacks. For simplicity, we use again the topology shown in Figure 1.

Scheme 1 (ID-in-Order): In this scheme, a node's MAC protects not only its own ID but also all previous IDs in the right order. That is, the MAC of node V_i is $MAC_i = H_{k_i}(M|1|2|...|i)$.

While this scheme can defend against mark insertion, removal, and re-ordering attacks, it fails under the mark altering attacks launched by a forwarding mole. For example, mole X can modify the MAC of node 2. As a result, the sink will see all received MACs are correct except that of node 2, thus the traceback incorrectly stops at node 3. In general, a forwarding mole can modify any previous MACs without affecting the correctness of the other MACs. Thus the sink can be fooled to trace to any legitimate forwarding node before the mole.

Scheme 2 (MAC-in-order): In this scheme, a node's MAC protects not only its own ID but also all previous MACs in the right order. That is, the MAC of node V_i is $MAC_i = H_{k_i}(M|i|MAC_1|MAC_2|...|MAC_{i-1})$.

Similar to the previous scheme, Scheme 2 can withstand mark insertion, removal, and re-ordering attacks, but cannot handle mark altering attacks launched by a forwarding mole. For example, mole X can modify the ID of node 2, thus the MAC added by node 2 becomes invalid, and the traceback incorrectly stops at node 3. In general, a forwarding mole can fool

the sink to trace to any previous legitimate node by modifying the ID of that node, because the MACs of other nodes are all valid.

10.2 Number of Packets needed to Collect All Marks

Now we give the detailed analysis for how many packets the sink need to receive before it can collect all marks.

Let N_i denote the number of packets the sink must receive to collect a mark from node V_i, N to collect one mark from each of nodes $V_1, ..., V_n$. The probability that exactly k packets are needed is:

$$P(N = k) = P(MAX_i\ N_i = k) = \sum_{j=1}^{n} P(exactly\ j\ N_i's = k\ and\ the\ other\ n-j\ N_i's < k)$$

$$= \sum_{j=1}^{n} C_n^j (P(N_i = k))^j (P(N_i < k))^{n-j} = (P(N_i < k+1))^n - (P(N_i < k))^n \quad (4)$$

We can compute the probability that it takes exactly k, and less than k packets, to collect a mark from V_i, as

$$P(N_i = k) = (1-p)^{k-1} p \quad (5)$$

$$P(N_i < k) = \sum_{j=1}^{k-1} P(N_i = j) = \sum_{j=1}^{k-1} p(1-p)^{j-1} = 1 - (1-p)^{k-1} \quad (6)$$

Substitute them in Equation 4, we have

$$P(N = k) = (1 - (1-p)^k)^n - (1 - (1-p)^{k-1})^n \quad (7)$$

From Equation 7, we can compute the CDF of N, thus the probability that marks from all nodes are collected within L packets,

$$P(N \leq L) = \sum_{j=1}^{L} P(N = j) = (1 - (1-p)^L)^n \quad (8)$$

10.3 Pseudocode for the Algorithm to Locate Moles at the Sink

In Section 4.2 we have described the algorithm that the sink uses to locate the moles, based on the received marks, in the probabilistic nested marking scheme. The pseudocode for this algorithm is given below.

Algorithm 1 Algorithm for Locating Moles

```
1: // Initialization
   candidate mole set S = φ, order relation Matrix M is 0 × 0, node set Ω = φ, loop node set Ψ=φ
2: // subroutine for adding an upstream relation i→j into and update the relation matrix
   //1 stands for V_i is upstream to V_j, -1 vice versa, 0 is undecided, 2 is ambiguous
   AddLink(i, j) {
      if (M[i][j] == −1)                    // if V_i has been showed downstream to V_j
         M[i][j] = M[j][i] = 2              // sets order between V_i, V_j to be ambiguous
      else if (M[i][j] == 0 && M[j][i] == 0)   // if the order is not decided
         M[i][j] = 1, M[j][i] = -1;         // sets V_i to be upstream to V_j
      for each k ∈ Ω                        //recursively update the orders of V_i, V_j to other nodes
         if (M[k][i] == 1 || M[k][i] == 2)  //If V_k is upstream to V_i, V_k is upstream to V_j
            AddLink(k,j);
         if (M[j][k] == 1 || M[j][k] == 2)  //If V_j is upstream to V_k, V_i is upstream to V_k
            AddLink(i,k);
      endfor
   }
3: // Given an order relation matrix M, identify nodes in a loop by identity swapping attacks
   FindLoop() {
      for each V_i ∈ Ω
         for each V_j ∈ Ω
            if(M[i][j] == 2)
               Ψ = Ψ ∪ {i, j}
   }
4: // Upon verification of k correct nested marks from nodes {V_{i_1} → V_{i_2} → ... → V_{i_k}}. Incorrect marks are discarded.
   ProcessPacket() {
      if(i_1 ∉ Ω)
         S = S ∪ {i_1}
      for each i_j ∉ Ω
         Ω = Ω ∪ {i_j}
         add a new row and a new column for i_j in M
         set all new elements in M to 0
      endfor
      for each link V_{i_j} → V_{i_{j+1}} s.t. M_{i_j,i_{j+1}} ≠ 1   //update M to record V_{i_j} is upstream to V_{i_{j+1}}
         AddLink(i_j, i_{j+1});
      endfor
      for each i ∈ S, find if there ∃ V_j ∈ Ω s.t. M_{j,i} = 1
         if so, S = S − {i}                 //remove those nodes having upstream nodes
      endfor
   }
   //continue on next page
```

Algorithm 2 Algorithm for Locating Moles (continued from previous page)

```
   // main procedure for locating moles
   Main() {
      //this ensures enough marks are received to reconstruct the route
      run ProcessPacket() for a sufficient number of received packets
      if(S ≠ φ)                  //If there exist nodes without upstream nodes
         S contains nodes whose one-hop neighborhood has moles
      else {
         run FindLoop() to remove nodes in the loop
         remove from M rows and columns corresponding to nodes in Ψ
         //Among nodes not in the loop, the most upstream one has moles in one-hop neighborhood
         find the most upstream nodes in M
      }
   }
```