MISPAR: Mitigating Stealthy Packet Dropping in Locally-Monitored Multi-hop Wireless Ad Hoc Networks

Issa Khalil
College of Information Technology
United Arab Emirates University, UAE
Email: ikhalil@uaeu.ac.ae

Saurabh Bagchi
Dependable Computing Systems Lab (DCSL)
School of Electrical & Computer Engineering
Purdue University, USA
Email: sbagchi@purdue.edu

ABSTRACT

Local monitoring has been demonstrated as a powerful technique for mitigating security attacks in multi-hop ad-hoc networks. In local monitoring, nodes overhear partial neighborhood communication to detect misbehavior such as packet drop or delay. However, local monitoring as presented in the literature is vulnerable to a class of attacks that we introduce here called *stealthy packet dropping*. Stealthy packet dropping disrupts the packet from reaching the destination by malicious behavior at an intermediate node. However, the malicious node gives the impression to its neighbors that it performed the legitimate forwarding action. Moreover, a legitimate node comes under suspicion. We introduce four ways of achieving stealthy packet dropping, none of which is currently detectable. We provide a protocol called MISPAR based on local monitoring to remedy each attack. It presents two techniques – having the neighbors maintain additional information about the routing path, and adding some checking responsibility to each neighbor. We show through analysis and simulation that the basic local monitoring fails to mitigate any of the presented attacks while MISPAR successfully mitigates them.

Keywords: Packet dropping, multi-hop wireless networks, local monitoring, misrouting, transmission power control.

1 INTRODUCTION

The traffic in wireless ad-hoc networks can be broadly classified into *data* and *control* traffic. Control traffic contains information to set up the network for data traffic to flow. Typical examples of control traffic include routing, monitoring the aliveness of nodes, topology discovery, and system management. Examples of data traffic include sensor readings and alert messages in surveillance environments.

It has been shown in the literature that wireless ad-hoc networks are vulnerable to a wide range of security attacks. The open nature, the fast deployment practices, and the hostile environments where they may be deployed, make them more susceptible to various kinds of attacks against both control and data traffic. Moreover, many ad-hoc networks such as sensor networks are resource-constrained, not only on energy but on bandwidth and computation as well. This limitation presents an additional challenge that any security protocol must live under. Control traffic attacks include wormhole [5], rushing [4], and Sybil [10]. The most notable data traffic attacks are blackhole, selective forwarding, and delaying of packets, in which respectively a malicious node drops data (entirely or selectively) passing through it, or delays its forwarding, and misrouting attack in which the attacker relays packets to the wrong next-hop with the effect of indirectly dropping them. These attacks could result in a significant loss of data or degradation of network functionality, say through disrupting network connectivity.

Cryptographic mechanisms alone cannot prevent these attacks since many of them, such as the wormhole and the rushing attacks, can be launched without needing access to cryptographic keys or violating any cryptographic check. To mitigate such attacks, many researchers have used the concept of cooperative *Local Monitoring* (*basic LM*) within a node's neighborhood (e.g., [1],[2],[6]-[8], [24],[28]). In local monitoring, nodes oversee part of the traffic going in and out of their neighbors. Different types of checks are done locally on the observed traffic to make a determination of malicious behavior. For systems where arriving at a common view is important, the detecting node initiates a distributed protocol to disseminate the alarm. Many protocols have been built on top of local monitoring for intrusion detection (e.g., [3]), building trust and reputation among nodes (e.g. [1], [2]), protecting against control and data traffic attacks (e.g. [6]-[8]) and in building secure routing protocols (e.g., [8], [9]). These attacks are detected by a group of nodes, called *guard nodes* that perform local monitoring. The guard nodes are normal nodes in the network and perform their basic functionality in addition to monitoring. Under local monitoring, a guard node verifies for a fraction of the packets if it is being forwarded within the requisite delay bound, without modification and without fabrication.

In this paper, we introduce a new class of attacks against traffic in wireless multi-hop ad hoc networks called *stealthy packet dropping*. In stealthy packet dropping the attacker achieves the objective of disrupting the packet from reaching the destination by malicious behavior at an intermediate node. However, the malicious node gives the impression to its neighbors capable of overseeing the packet that it has performed the required action (e.g., relaying the packet to the correct next-hop *en route* to the destination). This class of attacks is applicable to packets that are not acknowledged end-to-end (note that link-by-link MAC acknowledgment is irrelevant to the issue). Due to the resource constraints of bandwidth and energy, much traffic in multi-hop ad hoc wireless networks (e.g., sensor networks) is *unacknowledged* or only selectively acknowledged [29],[30]. This is particularly

Conference name: SecureComm 2008, September 22 - 25, 2008, Istanbul, Turkey.

true for the more common data traffic or broadcast control traffic than for rare unicast control traffic.

In this paper, we introduce four modes of the stealthy packet dropping attack. We distinguish between an external malicious node, which does not possess the cryptographic keys in the network, and internal compromised nodes, which do and are created by compromising an erstwhile legitimate node. Consider a scenario in which a node called *S* is forwarding a packet to a compromised node called *M*. *M* is supposed to relay the packet to the next-hop node *D*. The first form of the attack is called *packet misrouting*. In this mode, *M* relays the packet to the wrong next-hop neighbor, i.e., a neighbor node other than *D*. The result is that the packet does not reach its intended next-hop (*D*) while *M* appears to the guards as doing its job correctly. The second mode is called the *power control* attack. In this mode, *M* controls its transmission power to relay the packet to a distance less than that of the *D*. Therefore, the packet does not reach the next-hop while the attacker avoids detection by many guards. The third form of the attack is called the *controlled-jamming* attack. In this mode, the attacker uses another colluding node (external or internal) in the range of *D* to transmit data at the same time when *M* starts relaying the packet to *D*. Therefore, a collision occurs at *D*, which prevents the packet from being correctly received by *D*, while *M* looks to be performing its functionality correctly. The final mode of stealthy packet dropping is called the *identity delegation* attack. In this mode, the attacker colludes with a node *E* placed close to the source node *S*. *E* is allowed to use *M*'s identity and transmit the packet. Since *E* is almost at the same place as *S*, *D* does not receive the packet while the guards of *M* are deceived that *M* relays the packet to the next-hop. In each of these attack types, the adversary not only can successfully perform the attack, but also it hides the presence of its malicious activities. Additionally, in each attack type, a legitimate node is accused of packet dropping. We acknowledge that the attack model calls for smart adversaries (e.g., they can collude and can spend significant energy in launching the attacks). However, we believe that if the network is critical enough, we *do* have to worry about such smart adversaries.

We provide a protocol called MISPAR (**MI**tigating **S**tealthy **PA**cket D**R**opping in Locally-Monitored Multi-hop Wireless Ad Hoc Networks) that is constructed on local monitoring and that can mitigate each attack type introduced above. The MISPAR mitigation technique takes two forms – having guard nodes maintain additional next-hop information gathered during route establishment, and adding some checking responsibility to each neighbor. The latter technique makes use of the fact that under three of the attacks, neighbors have differing views of a node in terms of amount of forwarding traffic generated by the node. Hence, a single one-hop broadcast cannot convince *all* the neighbors. On the other hand, we show that of the four modes of the stealthy packet dropping attack, basic LM [6]-[8][24] is unable to detect any instance of three attack types (all except drop through power control) while it is able to detect specific instances of the drop through power control attack, depending on how the adversary constrains the range of the forwarded packet. Moreover, the work by Buchegger *et al.* [2][23] also relies on overhearing packet forwarding of neighbors and building reputation scores based on it. The attack class introduced here would be damaging to such a solution since the malicious actions cannot be detected and the adversary nodes will achieve high reputation scores. To the best of our knowledge, we are the first to propose a protocol suited to resource constrained wireless networks that can detect these four attack types.

We provide a theoretical analysis for the probability of success of the stealthy packet drop attack in a locally monitored network. We present the work for local monitoring in static sensor network. However, the technique is also valid under mobile situations. The requirement would be a primitive for determining the neighbor relation securely. Several such protocols exist in the literature [5][25]-[27]. We also analyze the resource consumption cost of MISPAR. Our analysis shows that MISPAR maintains detection coverage above 90% for the transmission control packet drop attack type for the configuration in which basic LM has less than 50% coverage. Additionally, we build a simulation model for the misrouting attack type using ns-2 and perform a comparative evaluation of local monitoring with and without MISPAR. Our simulation results show that MISPAR can deliver 60% of packets to the destination under 20% nodes compromised, while basic LM fails to deliver almost any packet. The likelihood of framing of legitimate nodes is also three-folds under basic LM for the same network. The performance advantages come at the expense of a slightly higher false isolation (due to natural collisions on the channel) and end-to-end delay in MISPAR.

We summarize our contributions in this paper as follows:

1. We introduce the stealthy packet dropping class of attacks and detail four methods by which it can be launched in locally-monitored networks without the malicious node being detected.
2. We provide mitigation remedy for each attack type with minimal addition to the resource consumption and responsibility of a node over baseline local monitoring.
3. We provide a mathematical analysis of the probability of success in launching the stealthy packet dropping attack and probability of detection in both basic LM and MISPAR.
4. We show through simulations the security advantage of MISPAR over basic LM.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 provides the foundations and background knowledge. Section 4 describes the stealthy packet dropping attack. Section 5 describes MISPAR and presents its mitigation techniques. Sections 6 and 7 present the mathematical analysis and the simulation results respectively. Finally, Section 8 concludes the paper.

2 RELATED WORK

In the last few years, researchers have been actively exploring many mechanisms to ensure the security of control and data traffic in wireless networks. These mechanisms can be broadly categorized into the following classes–authentication and integrity services, protocols that rely on path diversity, protocols that use specialized hardware, protocols that require explicit acknowledgements or use statistical methods, protocols that overhear neighbor communication.

The path diversity techniques increase route robustness by first discovering multi-path routes [9], [13] and then using these paths to provide redundancy in the data transmission between a source and a destination. The data is encoded and divided into multiple shares sent to the destination via different routes. The method is effective in well-connected networks, but does not provide enough path diversity in sparse networks. Moreover, many of these schemes are expensive for resource-constrained wireless networks due to the data redundancy. Additionally, these protocols could be vulnerable to route discovery attacks, such as the Sybil attack, that prevent the discovery of non-adversarial paths.

Examples of protection mechanisms that require specialized hardware include [5] , and [11]. The authors in [5] introduce a scheme called *packet leashes* that uses either tight time-synchronization or location awareness through GPS hardware. The work in [11] relies on hardware threshold signature implementations to prevent one node from propagating errors or attacks in the whole network.

A technique proposed to detect malicious behavior involving selective dropping of data, relies on explicit acknowledgement for received data using the same channel [13], or an out-of-band channel [12]. This method would render stealthy packet dropping detectable at the end point. However, the method incurs high communication overhead and has to be augmented with other techniques for diagnosis and isolation of the malicious nodes. A natural extension would be to reduce the control message overhead by reducing the frequency of ack-ing to one in every N data messages (in the above papers N=1). However, this may delay the adversary detection which may result in significant damage. In contrast, in MISPAR, the node is detected and diagnosed locally by its neighbors. Statistical measures have been used by some researchers for detection, e.g., [14] to detect wormhole attacks.

A widely used technique for mitigating control and data attacks in multi-hop wireless networks is cooperative local monitoring by overhearing traffic in the vicinity. The idea of overhearing traffic in the vicinity (e.g. [1]-[3]) has been used to build trust relationships among nodes in networks (e.g. [1], [2]), detect and mitigate certain kinds of attacks (e.g. [3], [6]-[8]), or discover routes with certain desirable properties, such as being node disjoint (e.g. [13]). Work in [8] provides detection of a wide class of control attacks against static sensor networks. *However, local monitoring, as used by all researchers to date, fails to mitigate the stealthy packet dropping attack.*

3 FOUNDATIONS

3.1 Attack Model and System Assumptions

Attack model: An attacker can control an external node or an internal node, which, since it possesses the keys, can be authenticated by other nodes in the network. An insider node may be created, for example, by compromising a legitimate node. A malicious node can perform packet dropping by itself or by colluding with other nodes. The collusion may happen through out-of-band channels (e.g., a wireline channel). However, we do not consider the denial of service attacks through physical-layer jamming [22], or through identity spoofing and Sybil attacks [10]. There exist several approaches to mitigate these attacks – [22] for jamming and [10] for the Sybil attack. A malicious node can be more powerful than a legitimate node and can establish out-of-band fast channels (e.g., a wireline link) or have high-powered controllable transmission capability. The attacks do not affect only a specific routing protocol; rather, they apply to a wide class where the requirement is an intermediate node determines the next hop node toward the final destination. This includes routing protocols specific to WSNs such as the beacon routing protocol.

System assumptions*:* We assume that all the legitimate communication links are bi-directional. We assume that secure neighbor discovery has been performed and that every node knows both first and second hop information. This can be achieved through the protocol described in [21] as well as by approaches developed by other researchers [4]. Note that while this knowledge is enormously useful, this by itself cannot mitigate many attack types. For example, further work is needed to detect the wormhole attack. Intuitively this information subsets the nodes from which a given node will accept packets but does not eliminate the possibility of malicious nodes within that subset. Local monitoring assumes that the network has sufficient redundancy, such that each node has more than an application defined threshold number of legitimate nodes as guards. We assume a key management protocol, e.g., [15], exists such that any two nodes can communicate securely.

3.2 Background: Local Monitoring

Local monitoring is a collaborative detection strategy where a node monitors the control traffic going in and out of its neighbors. This strategy was introduced in [6] for *static sensor* networks and here we give the background needed to understand the concepts presented in this paper.

For a node, say α, to be able to watch a node, say N_2, α must be a neighbor of both N_2 and the previous hop from N_2, say N_1. Then we call α a *guard* node for N_2 over the link $N_1 \rightarrow N_2$. We use the notation $R(N)$ to denote the set of all nodes that are within the radio range of node N and $G(N_1, N_2)$ to denote the set of all guard nodes for N_2 over a link $N_1 \longrightarrow N_2$.

Formally, $G(N_1, N_2) = R(N_1) \cap R(N_2) - N_2$, where $N_2 \in R(N_1)$.

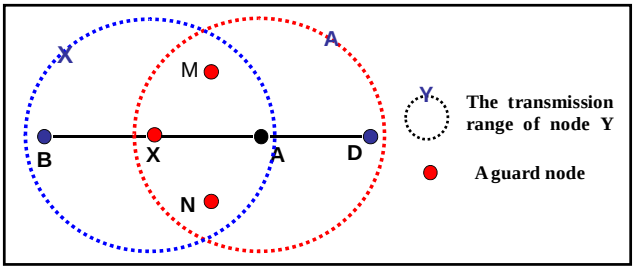

Figure 1: X, M, and N are guards of A over X→A

For example, in Figure 1, $G(X,A)=\{M,N,X\}$. Information from each packet sent from X to A is saved in a *watch buffer* at each guard. The guards expect that A will forward the packet toward the ultimate destination, unless A is itself the destination. Each entry in the watch buffer is time stamped with a time threshold, τ, by which A must forward the packet. Each packet forwarded by A with X as a previous hop is checked for the corresponding information in the watch buffer. The check can be to verify if the packet is fabricated or duplicated (no corresponding entry in the buffer), corrupted (no matching hash of the payload), dropped or delayed (entry is not matched within τ).

A malicious counter ($MalC(i,j)$) is maintained at each guard node, i, for a node, j, at the receiving end of each link that i is monitoring over a sliding window of length T_{win}. $MalC(i,j)$ is incremented for any malicious activity of j detected by i. The increment to $MalC$ depends on the nature of the malicious activity, being higher for more severe infractions. When the growth in the counter value maintained by a guard node i for node j ($MalC(i,j)$) crosses a threshold rate ($MalC_{th}$) over T_{win}, node i revokes j from its neighbor list (called *direct isolation* since it will henceforth not perform any communication with node j), and sends to each neighbor of j, an authenticated alert message indicating j is a suspected malicious node. When a neighbor N_i gets the alert, it verifies the authenticity of the alert message. When N_i gets enough alert messages about j, it marks the status of j as revoked (called *indirect isolation*). The notion of enough number of alerts is

quantified by the *detection confidence index* γ. Each node maintains a memory of nodes that it has revoked through a local blacklist so that a malicious node cannot come back to its neighborhood and claim to be blameless. This constitutes *local isolation* of a malicious node by its current neighbors.

4 STEALTHY DROPPING ATTACK DESCRIPTION

In all the modes of stealthy packet dropping, a malicious intermediate node achieves the same objective as if it were dropping a packet. However, none of the guard nodes using basic LM become any wiser due to the action. In addition, some legitimate node is accused of packet dropping. Next, we describe the four attack types for stealthy packet dropping.

4.1 Drop through Misrouting

In the misrouting attack, a malicious node relays the packet to the wrong next-hop, which results in a packet drop. Note that, in basic LM [6], a node that receives a packet to relay without being in the route to the destination either drops the packet or sends a one-hop broadcast that it has no route to the destination. The authors in [6] argue that that latter case would be more expensive and dangerous since it gives malicious nodes valid excuses to drop packets. Therefore, they go with the first choice, even though it may result in some false accusations.

Consider the example scenario in Figure 2. Node *A* sends a packet to the malicious node *M* to be relayed to node *B*. Node *M* simply relays the packet to node *E* which is not in the route to the final destination of the packet. Node *E* drops the packet. The result is twofold: (i) node *M* successfully drops the packet without being detected since all the guards of *M* over *A*→*M* (regions I & II) have been satisfied by the transmission of *M*→*E*, and (ii) legitimate node *E* will be wrongly accused by its guards over *M*→*E* (regions II & III) as maliciously dropping the packet.

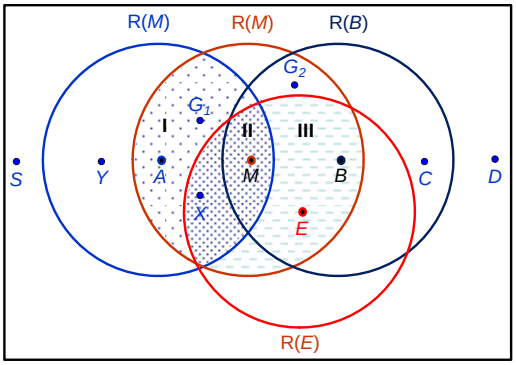

Figure 2: Misrouting scenario

4.2 Drop through Power Control

In this type of attack, a malicious node relays the packet by carefully reducing its transmission power, thereby reducing the range and excluding the legitimate next-hop node. This kind of transmission power control is available in today's commercial wireless nodes, such as the Crossbow Mica family of nodes.

Consider the scenario shown in Figure 3. A node *S* sends a packet to a malicious node *M* to be relayed to node *T*. Node *M* drops the packet by sending it over a range that does not reach *T* (the dotted circle centered at *M*). Figure 3(a) shows the guards of *M* that are satisfied by the controlled transmission of *M* (region II) and the set of guards that detect *M* (region I) as dropping the packet since they did not overhear *M*. Figure 3(b) shows all the guards of *M* over *S*→*M*. Figure 3(d) shows the set of guards of *T* over *M*→*T* that wrongly accuse *T* of dropping the packet. The farther *T* is from *M* the better it is for the attacker since more guards can be satisfied and therefore, the stealthier the attack. For this attack to succeed, the attacker must know the location of each neighbor and the detection confidence index γ. Typically security is not achieved through obfuscation and therefore protocol parameters such as γ are taken to be known to all and location determination is routinely run upon deployment of nodes. When the number of guards that are not satisfied by the controlled-power transmission is greater than γ-1, an intelligent attacker will refrain from lowering the transmission power since it will be detected and isolated by all its neighbors either directly or indirectly (Section 3.2). Here too, a successful attack, not only achieves the effect of dropping the packet, but also causes a subset of the guards of *T* over *M*→*T* to accuse node *T* of dropping the packet.

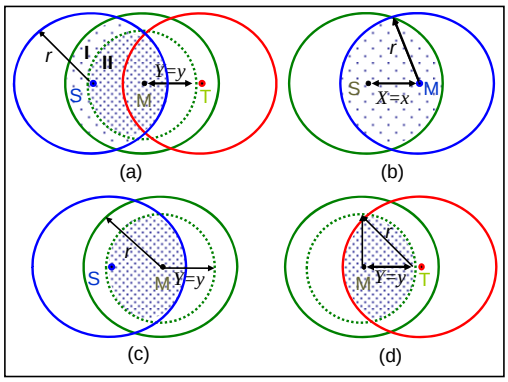

Figure 3: (a) The guards of M over S→M (I & II); (b) Separation between S and M = x; (c) the subset of guards of M over S→M that has been satisfied by the controlled power transmission of M; (d) the subset of guards of T over M→T that wrongly accuses T of dropping the packet

4.3 Drop through Colluding Collision

In many wireless sensor network deployment scenarios, the 802.11 MAC protocol RTS-CTS mechanism that reduces frame collisions due to the hidden terminal problem and the exposed terminal problem are disabled for the sake of energy saving. This is also explained by the fact that packets in some wireless networks such as sensor networks are often quite small and fall below the threshold for packet length for which RTS/CTS is turned on.

The attacker may exploit the absence of the RTS/CTS frames to launch a stealthy packet dropping attack through collision induced by a colluding node. The colluding node creates a collision in the vicinity of the expected next-hop node at an opportune time. Consider the scenario shown in Figure 4. The malicious node M_1 receives a packet from *S* to be relayed to *T*. Node M_1 coordinates its transmission with a transmission of some data generated by its colluding partner M_2 to *T*. It has the effect that *T* is unable to get the packet relayed by M_1. The damage caused by this attack is twofold: (i) M_1 successfully drops the packet due to a collision at *T* without being detected, and (ii) node *T* is accused of dropping the packet by some of its guards over the link M_1→*T* (the guards that are out of the range of M_2, region I). Note that for M_2 to be able to send data to *T*, it is has to be a legitimate neighbor (compromised by the attacker), otherwise, the attack would be considered a

physical layer jamming [22], which is assumed to be detectable through techniques complementary to that presented in the paper (e.g., [8][22]).

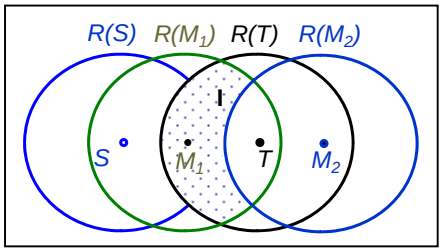

Figure 4: Colluding collision illustration scenario

4.4 Drop through Identity Delegation

In this form of the attack, the attacker uses two malicious nodes to drop the packet. One node is spatially close to the sender. The other node is the next-hop from the sender. The first malicious node could be external or an internally compromised node while the latter has to be an internally compromised node. Consider the scenario shown in Figure 5, node *S* sends a packet to a malicious next-hop node M_2 to be relayed to node *T*. The attacker delegates the identity and the credentials of the compromised node M_2 to a colluding node M_1 close to *S*. After *S* sends the packet to M_2, M_1 uses the delegated identity of M_2 and transmits the packet. The intended next-hop *T* does not hear the message since $T \notin R(M_1)$. The guards of M_2 over $S \rightarrow M_2$ are the nodes in the shaded areas I & II and they are all satisfied since they are in $R(M_1)$. Again, the consequences of this attack are twofold: (i) the packet has been successfully dropped without detection, and (ii) the set of nodes in the shaded area II overhear a packet transmission (purportedly) from M_2 to *T*. These nodes are included in $G(M_2,T)$ and will subsequently accuse *T* of dropping the packet.

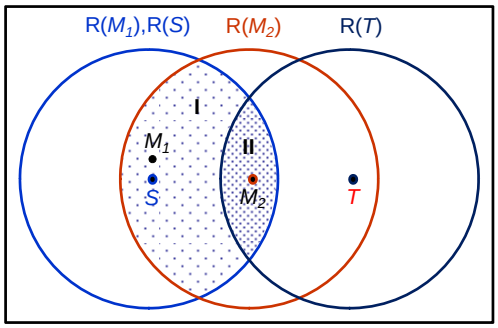

Figure 5: Identity delegation illustration scenario

5 STEALTHY DROPPING ATTACK MITIGATION

In this section we propose two mechanisms to augment traditional local monitoring to detect stealthy packet dropping. The first mechanism mitigates the misrouting stealthy packet drop while the second mitigates the rest of the attack types.

5.1 Mitigating Misrouting Packet Drop

To detect this attack, the local monitoring has to incorporate additional functionality and information. The basic idea is to extend the knowledge at each guard to include the identity of the next-hop of the packet being relayed.

This additional knowledge can be collected during route establishment. Many multi-hop wireless routing protocols provide this knowledge without any modification while some changes are necessary in others. The first class includes both reactive routing protocols such as Dynamic Source Routing (DSR) and its variants [16] and proactive routing protocols such as TinyOS beacon routing [18] and Destination Sequenced Distance Vector routing (DSDV [19]). In all source routing protocols, the packet header carries the identity of all the nodes in the route from the source to the destination. Therefore, no additional traffic is required to be generated for the guard nodes to be able to detect this kind of attack. Moreover, no additional information is required to be maintained at the guards since each packet carries the required information in its header. In TinyOS beacon routing, the base station periodically broadcasts a beacon to establish a breadth first search tree rooted at the base station. Each node within the transmission range of the base station overhears the beacon, sets its parent to be the base station, sets the hop count to the base station to be one, and rebroadcasts the beacon. Each beacon carries the identity of the broadcasting node, the identity of its parent, and the hop count to the base station. Each guard overhearing the beacon broadcasting saves parent node identity for each neighbor. Later, when a node, say *B*, is sent a packet to relay, the guard of *B* can detect any misrouting by *B* since it knows the correct next-hop *en route* to the base station.

The second class of routing protocols requires modification to the protocol to build the next-hop information at the guards. Examples of these protocols are the reactive routing protocols that use control packet flooding of route requests (*REQ*) and route replies (*REP*) to establish the route between the source and the destination (e.g., LSR [8] and AODV [17]). In these protocols, when a source node desires to send a message to some destination node and does not already have a valid route to that destination, it initiates a route discovery process to locate the other node. It broadcasts a route request (*REQ*) packet to its neighbors, which then forward the request to their neighbors, and so on, until either the destination or an intermediate node with a "fresh enough" route to the destination is located. Along with its own sequence number and the broadcast ID, the source node includes in the *REQ* the most recent sequence number it has for the destination. During the process of forwarding the *REQ*, intermediate nodes record in their route tables the address of the neighbor from which the first copy of the broadcast packet is received, thereby establishing a reverse path. Once the *REQ* reaches the destination, the destination node responds by unicasting a route reply (*REP*) packet back to the neighbor from which it first received the *REQ*. As the *REP* traverses along the reverse path, nodes along this path set up forward route entries in their route tables which point to the node from which the *REP* came.

Next, we show the required changes to the basic version of AODV to enable the guards to build the necessary knowledge for detecting the misrouting attack. The idea behind the solution is that during route establishment, when the relation about which node to forward a packet between a given source-destination pair is determined, this information is broadcast by a neighbor to the guards which will be responsible for monitoring the node. To collect the next-hop identity information, the forwarder of the *REQ* attaches the previous two hops to the *REQ* packet header. Let the previous hop of *M* be *A* for a route from source *S* to destination *D*, and the next hop from *M* be *B* (Figure 2). When *M* broadcasts the *REQ* received from *A*, it includes the identity of *A* and its own identity (*M*) in the *REQ* header <*S*, *D*, *REQ_id*, *A*, *M*>. When *B* and the other neighbors of *M* get the *REQ* from *M*, they keep in a

Verification Table (*VT*) <*S, D, RREQ_id, A, M, ->* (last field is currently blank). When *B* broadcasts the *REQ*, the common neighbors of *M* and *B* update their *VT* to include *B* <*S, D, RREQ_id, A, M, B*>. When *B* receives a *REP* to be relayed to *M*, it includes in that *REP* the identity of the node that *M* needs to relay the *REP* to, which is *A* in this example. Therefore, all the guards of *M* now know that *M* not only needs to forward the *REP* but also that it should forward it to *A* and not any other neighbor.

Therefore, two tasks have been added to the functionality of the guards in monitoring the *REP* packets. First, the guard *G* of a node *N* verifies that *N* forwards the *REP* to the correct next-hop. In the example above, G_2 verifies that *M* forwards the *REP* to *A*. Second, *G* verifies that *N* has updated the forwarded *REP* header correctly. In the example shown above, G_2 verifies that when the input packet to *M* from *B* is <*REP, S, D, REQ_id, C, B, M*>, then the output packet from *M* should be <*REP, S, D, REQ_id, B, M, A*>. Thus *M* and its guards over the link *B*→*M* know that the next-hop is *A* from the information built in the *VT* table during the *REQ* flooding.

Using the additional information mentioned above, MISPAR detects misrouting attacks as follows. In the example above, assume that *S* is sending a data packet to *D* through a route that includes <*Y,A,M,B,C*>. The malicious node *M* cannot misroute the data packet received from *A* to a node other than the next-hop, *B* since each guard of *M* over the link *A*→*M* has an entry in its *VT* which indicates *B* is the correct next-hop. This results in an additional checking activity for the guard node involved in local monitoring–verifying the data packet is forwarded to the correct next hop, as indicated by the entry in the guard node's *VT*. Moreover, *M* cannot frame another neighbor, say *X*, by misrouting the packet to *X*. The guards of *X* over *M*→*X* do not have an entry like < *S, D, REQ_id, Y, A, M, X*> and therefore, they would not increment the MalC of *X* when it drops the packet.

5.2 Mitigating other Stealthy Drop Attacks

The key observation behind the other types of the stealthy packet dropping attack is that the attack defeats local monitoring based detection by reducing the number of guards that overhear a packet to zero or to a number that is less than the confidence index γ. In the power control attack shown in Figure 3(a), the attacker narrows the guards that can detect the packet drop into the lightly shaded area (region I in Figure 3(a)) while the majority of the guards (region II in Figure 3(a)) are satisfied. In the colluding collision attack (Figure 4) and identity delegation attack (Figure 5), the attacker completely evades detection by satisfying all the guards (the nodes in region I of Figure 4 and of Figure 5).

The countermeasure we propose against these attacks is based on the observation that an adversary evades detection of dropping packets by allowing only a subset of guards to overhear the message being forwarded. Therefore, we expand the set of nodes that can guard a node from only the common neighbors of the node being monitored and its previous-hop node to include all the neighbors. Since all neighbors are included in verifying the node, by definition, some neighbor will see evidence of stealthy packet drop. The detection technique makes use of the fact that, under the stealthy packet dropping attacks, neighbors have differing views of a node in terms of the volume of traffic it has forwarded and all the neighbors cannot be convinced by a single broadcast. To achieve this goal we need to introduce additional tasks for the nodes in the network. (i) Each node keeps a count of the number of messages each of its neighbors had forwarded over a predetermined time interval and (ii) each node has to announce the number of packets it has forwarded over some period of time. The adversary evades detection of stealthy packet dropping by allowing only a subset of guards to overhear the packet being forwarded. Thus, the subset of guards that had overheard the packet forwarding would have a higher count than the nodes that did not overhear the forwarding. By forcing a node to announce the number of messages it has forwarded over some period of time, a malicious node would have the problem of satisfying two sets of neighbors that expect to hear different counts through a single broadcast.

A neighbor of a node, say *N*, that collects the number of forwarded packets by *N* and compares the result with the count announced by *N* is called a *comparator* of *N*, denoted by *C(N)*. For any node *N* all nodes in radio range *R(N)* act as comparators of *N*. Recall that a guard of a node *B* over the link *Y*→*B*, has been defined in the basic LM as any node that lies within the transmission range of both *Y* and *B*. Therefore, each guard of *N* over a certain link is a comparator of *N*, however, not every comparator of *N* is a guard of *N*. The function of a comparator is to count the total number of packets forwarded from the node within a time period. During some time periods node *N* may be required to announce the number of messages it has forwarded in that period. If a comparator's count is not within an acceptable range of the announced forward count, the comparator increments its malicious counter for the announcing node.

In order to reduce radio traffic, we do not require all nodes to announce their forward count for every time period. Instead a node must announce within the time period that it receives a broadcast message request to announce. Whenever a node, say *A*, overhears a packet from a node *N* that is not within the neighbor list of *A*, node *A* broadcasts a 3-hop request for *N* to announce its forward count. If node *N* and all of its neighbors are within 3 hops of the requestor then the neighbors of *N* will act as comparators of *N* and expect to hear the correct forward count announced. The basic idea is that a malicious node that has dropped a packet faces a dilemma; some of its neighbors have overheard the dropped packet and expect it to be included in the send count while other neighbors have not heard the packet so they expect a send count of one less message. However, note that a suspicion would not be raised by a discrepancy of one due to natural losses (channel conditions and collisions). Detection is triggered only when the discrepancy crosses a predetermined threshold.

For simplicity of exposition, for the following examples, we will consider that a discrepancy of a single packet is sufficient for detection. Consider the power drop attack scenario shown in Figure 3(a), the neighbors of *M* within the dotted circle would have one more count for the number of packets forwarded by *M* as compared to the counters in the rest of *M*'s comparators. In each of the last three attack modes, the attacker is faced by two sets of neighbors that have different views about him. The best the attacker can do is to satisfy the larger set, however, the nodes of the other set would detect the discrepancy and propagate the detection knowledge to the nodes of the other set. All the nodes of the smaller set would then directly isolate the malicious node. The nodes of the larger set indirectly isolate the malicious node if the number of nodes in the smaller set is greater than or equal to the detection confidence index γ.

6 ANALYSIS

The analysis gives the detection probability for a malicious node indulging in the drop through power control attack type. It provides the result for basic LM (basic LM) and MISPAR under different detection confidence index (γ) values.

Assumptions: We consider a homogeneous network of nodes where the nodes are uniformly distributed in the field with density *d*. For simplicity, we assume that the field is large enough that edge effects can be neglected in our analysis. Consider any two randomly selected neighbor nodes, *S* and *M*, as shown in Figure 3(a). Nodes *S* and *M* are separated by a distance *X*, and the communication range is *r*. *X* is a random variable that has the probability density function of $f_X(x) = 2x/r^2$ with range (0,*r*). This follows from the assumption of uniform distribution of the nodes.

Attacker model: The malicious node uses an omni-directional antenna. Its goal is to have the effect of dropping the packet from reaching the legitimate next-hop node. The detection probability is a lower bound since we assume that the adversary can control the transmission power level to be infinitesimally smaller than that required to reach node *T*. The reduced transmission range of *M* is represented as *y*.

Basic Local Monitoring (LM): The guards of *M* over the link from *S*→*M* lie on the shaded area shown in Figure 3 (b). The subset of guards that can be satisfied by the controlled power transmission of *M* lies on the shaded area shown in Figure 3 (c), we call these guards the happy guards N_h. Finally, the subset of guards of *T* over *M*→*T* that wrongly accuse *T* of dropping the packet are shown in the shaded area of Figure 3 (d), we call these guards the fooled guards, N_f. The shaded area in Figure 3 (c) is found to be

$$Area(c) = \begin{cases} \{r^2 \cos^{-1}(r/2y) + y^2(\pi - 2\cos^{-1}(r/2y)) \\ -(r/2)\left(\sqrt{4y^2 - r^2}\right)\} & \text{when } (y > r/2) \\ \pi y^2 & \text{when } (y \le r/2) \end{cases}$$

which is the same as the shaded area in Figure 3(d). Recall that *Y* is a random variable with probability density function of $f_Y(y) = 2y/r^2$ with range (0,*r*). Therefore, $N_h = N_f = Area(c) \times d$. Finally, the number of guards that can detect the power control attack is $N_d = N_g - N_h$. The condition for a successful attack is $N_d < \gamma$. The probability of detection of the attack in basic LM is plotted in Figure 6 as the detection confidence index is varied. For the plot, transmission range is 50 m, distance between *S* and *M* is the transmission range (which gives the smallest number of guards on *S*→*M* and hence the lower bound on the detection probability), and each node has on an average 40 neighbors. The analytical result shows that as γ increases, the probability of detection decreases sharply. For a reasonable γ value of 3, the detection probability is less than 0.5.

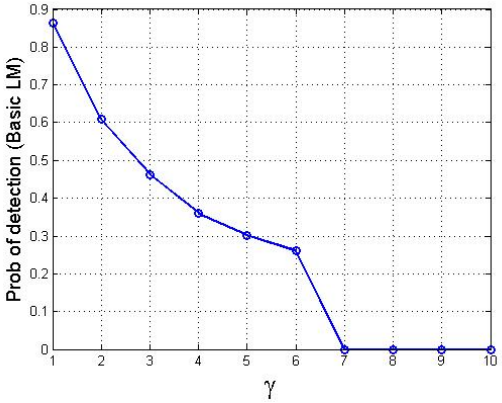

Figure 6: Probability of detection of attack type drop through power control for basic LM

MISPAR: The expected number of comparators of any node is $N_c = \pi r^2 d$. The subset of comparators that can overhear *M* are those that lie within the dotted circle of Figure 3(c), we call these comparators the *Plus Comparators* C_p. The subset of comparators that cannot overhear the transmission of *M* are those that lie within the legitimate transmission range of *M* but out of the dotted circle, we call these the *Minus Comparators* C_m.

$$C_p = \pi y^2 d \text{ and } C_m = N_c - C_p = \pi(r^2 - y^2)d$$

The condition for successful attack is $\min(C_p, C_m) < \gamma$, since the intelligent adversary broadcasts a message count that satisfies the larger of the two sets. The probability of successful detection is plotted in Figure 7 as the detection confidence index (γ) is varied for the same parameters as in basic LM. As γ increases, the detection probability expectedly goes down since it becomes more difficult for the nodes to agree to isolate the malicious node. However, MISPAR is considerably more effective in detecting the attack than basic LM. This is due to the design of having comparators and verifying the forwarded message counts.

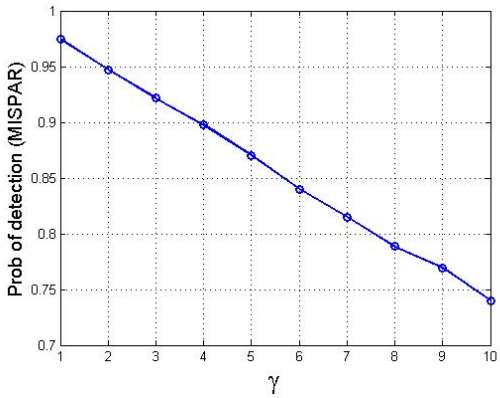

Figure 7: Probability of detection of attack type drop through power control for MISPAR

Finally, we analyze the additional resource requirements of MISPAR over basic LM. These are (i) state maintenance of the next-hop node, counter for the number of packets forwarded by each neighbor both of which are linear in terms of number of neighbors, (ii) broadcast of the counters in an on-demand basis, triggered by a relatively rare event, (iii) two node identifiers in each route request and reply packet. These are not onerous additions even for a resource constrained environment.

7 SIMULATION RESULTS

We use the *ns-2* simulation environment [20] to simulate a data exchange protocol, individually with basic LM and with MISPAR. We distribute the nodes randomly over a square field with a fixed average node density. Thus, the field size varies with the number of nodes (80×80 m to 204×204 m). We use a generic on-demand shortest path routing protocol that floods route requests and unicasts route replies in the reverse direction. A route, once established, is not used forever but is evicted from the cache after an idle period $TOut_{Route}$ if no other packet has been forwarded to the particular destination. We simulate the misrouting attack as a representative of stealthy packet dropping. When a malicious node gets a data packet, it relays the packet to a wrong next-hop with a probability of f_{dat}. A malicious node does not generate any data of its own. The simulation also accounts for losses due to natural collisions. The guards inform all the neighbors of the detected

malicious node through multiple unicasts. For each run, malicious nodes are chosen at random.

Input parameters: Each node acts as a data source and generates data using an exponential random variable with inter-arrival rate ϕ. The destination is chosen at random and used for a random time following an exponential distribution with rate ξ. We use N_M for the number of malicious nodes and N for the total number of nodes. The input parameters with the experimental values are given in Table 1, we use the same settings as in [6] so that the results are comparable.

Output parameters: The output parameters include (i) the fraction of data packets received (delivery ratio) calculated as the total number of packets successfully received by final destinations over the total number of packets sent, (ii) the framing ratio, which is defined as the fraction of good nodes that have been incorrectly isolated due to the attack over the total number of good nodes, (iii) the false isolation ratio, which is defined as the fraction of good nodes that have been isolated due to natural causes (collisions and losses on the wireless channel) over the total number of good nodes, (iv) the malicious node isolation ratio (*true isolation*), which is defined as the number of malicious nodes isolated to the total number of malicious nodes, (v) the average end-to-end delay of data packets, which is the time a packet takes after leaving the source until it reaches its final destination. Note that here we only consider framing as a result of the misrouting attack and we do not consider the kind of framing where enough number of malicious nodes in a neighborhood frame a legitimate neighbor. The latter kind of framing is identical to that in basic LM and has already been analyzed [6].

The output parameters are measured at the end of the simulation time (1500 seconds). The output parameters are obtained by averaging over 30 runs. The reasoning provided for some experimental results was arrived at by careful examination of the simulation logs. When a claim is made of difference between MISPAR and the baseline, the difference is significant at the 95% confidence level.

Table 1: Input parameters for MISPAR simulation

Param.	Value	Param.	Value
Tx Range (r)	30 m	ξ, ϕ	0.02 , 0.2
MalC increment	15	f_{dat}	0.6
$TOut_{Route}$, T_{win}	50 s, 200	N_M	0-20
C_t, γ	150, 3	τ	0.5 sec
# nodes (N)	100	BW	40 kbps

Figure 8 shows the variations in delivery ratio as the number of malicious nodes varies. The figure shows that the delivery ratio decreases as N_M increases. This is due to the packets dropped before the malicious nodes are detected and isolated. As N_M increases, this initial drop increases and thus the delivery ratio decreases. Moreover, as N_M increases, the true isolation decreases. Therefore, the malicious nodes that could not be detected continue to drop packets and this decreases the delivery ratio. The delivery ratio in basic LM is much less than in MISPAR and the difference increases as the number of malicious nodes increases. This is due to two main reasons. The first is that basic LM fails to detect any of the malicious nodes and thus they continue to drop packets constantly. The second is that some of the good nodes in basic LM get framed by the adversary and thus become isolated and reduce the overall throughput.

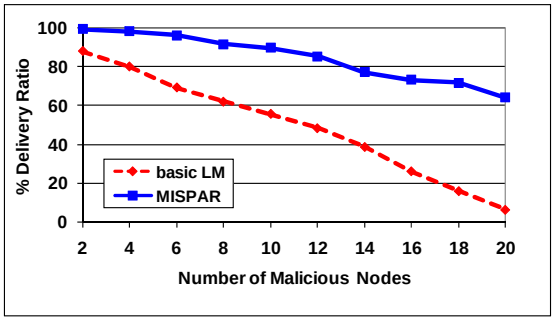

Figure 8: Effect of N_M on delivery ratio

Figure 9 shows the variations in true isolation as N_M varies. Most significantly, we observe that basic LM has a very poor performance while MISPAR achieves above 80% isolation of malicious nodes with up to 12% compromised nodes. The figure shows that the true isolation decreases as we increase N_M. This is because the number of available guards and comparators in the network decreases as more and more nodes get compromised. Furthermore, as N_M increases, local isolation becomes less effective since the number of legitimate neighbors decreases and if this goes below γ, then local isolation has to wait for direct isolation individually by each legitimate neighbor. Moreover, as N_M increases the data traffic in the network decreases (in the simulation malicious nodes do not send data) which results in a decrease in the number of packets that a single malicious node may drop. This in turn results in decreasing the likelihood that the malicious node is detected and isolated.

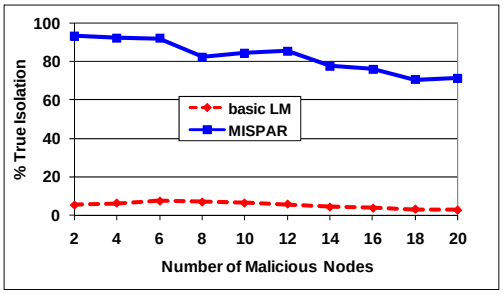

Figure 9: Effect of N_M on true isolation

Figure 10 shows the variations in false isolation as N_M varies. The figure shows that the false isolation initially increases as we increase N_M and starts to decrease beyond a point. This is because not all guard nodes come to the decision to isolate a malicious node at the same time. Therefore, a given guard node may suspect another guard node when the latter isolates a malicious node but the former still has not. The occurrence of this situation increases with N_M and hence the false isolation increases with N_M. For example, a guard node G_1 detects a malicious node M earlier than the other guard nodes for the link to M. G_1 subsequently drops all the traffic forwarded to M and is therefore suspected by other guard nodes of M. This problem can be solved by having an authenticated one-hop broadcast whenever a guard node performs a local detection. An opposing pull comes from the fact that the number of good nodes decreases as we increase N_M. This in turn results in a decrease of the indirect false isolation since a node may not have more than γ good nodes to agree on falsely isolating a neighbor. Moreover, as N_M increases, the data traffic decreases since malicious nodes are not generating data. This in turn decreases the

chance for collisions and consequently decreases false isolation. Beyond a point (N_M=6), the latter factors dominate the first factor and there is a decrease in false isolation ratio. The false isolation in MISPAR is slightly higher than in basic LM due to more aggressive detection with an increased level of monitoring.

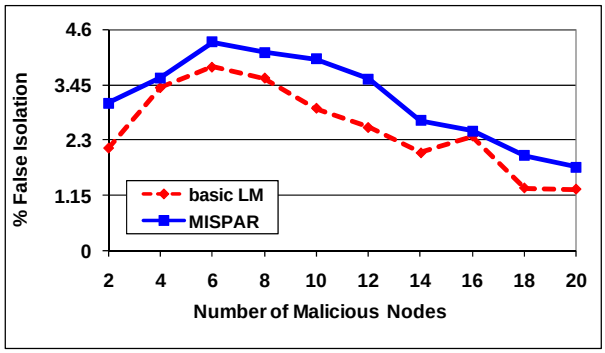

Figure 10: Effect of N_M on false isolation

Figure 11 shows the variations in end-to-end delay as N_M is varied. The figure shows that the end-to-end delay initially increases as N_M increases and then starts to decrease. As N_M increases, the route reestablishment frequency increases. This is due to the fact that a route remains active for a time $TOut_{Route}$ and this timer is reset with every packet forwarded using that route. Consequently cutting the flow of packets (by maliciously dropping the packet) causes the route entries to stale. Therefore, additional traffic is generated to reestablish the route which increases the end-to-end delay. The opposing pull comes from the fact that as N_M increases the traffic decreases. This reduces the contention in the network which in turn decreases the end-to-end delay. As N_M increases beyond a point, the latter factor dominates and the overall result is a decrease in the end-to-end delay. The end-to-end delay in MISPAR is slightly higher than in basic LM. This is due to the modification of the routing protocol in MISPAR which makes the route establishment time slightly higher.

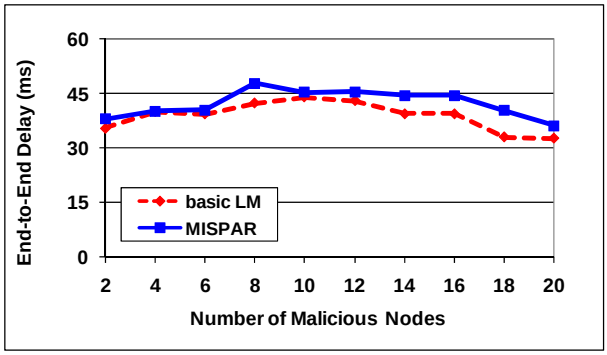

Figure 11: Effect of N_M on end-to-end delay

Figure 12 shows the variations in framing ratio as N_M varies. The figure shows that the framing ratio increases as N_M increases for both basic LM and MISPAR. However, the framing ratio in basic LM is much higher than in MISPAR. The framing in basic LM occurs as a consequence of successful and continuous misrouting attack (basic LM fails to detect and isolate the malicious nodes). As N_M increases, the framing ratio increases due to the increase in the number of attack occurrences. As N_M increases more and more the framing ratio starts to level off since the traffic in the network becomes low and no more good nodes can be framed. On the other hand, the little framing in MISPAR is due to imperfect true isolation of malicious nodes due to collisions, channel conditions, or insufficient number of guards (from Figure 9, we see that the coverage is not 100%). As N_M increases, the true isolation decreases and thus framing increases.

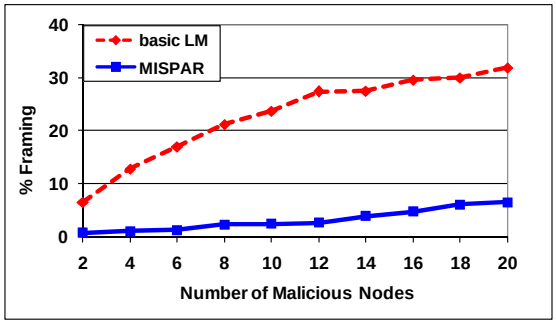

Figure 12: Effect of N_M on framing

8 DISCUSSION

Here we have described the design of MISPAR, which fundamentally relies on the ability of some guard nodes to overhear the behavior of neighboring nodes. This basic feature of wireless networks has been leveraged by many researchers, for almost a decade now starting from [28]. Any technique that relies on this has the shortcoming that it can be bypassed by a powerful adversary that can accurately place malicious nodes or precisely control transmission power of a malicious node. Intrinsically, the placement or the transmission power control can be used to hide the behavior from the requisite number of guard nodes, e.g., the next hop node does not get the packet but the guards see it. In that case, no detection will occur. MISPAR suffers from this shortcoming as does *all* the work that relies on the feature.

The memory cost of a technique like MISPAR may be of concern since overheard packets have to be maintained in memory. However, the common case behavior is that of nodes behaving legitimately. Therefore, the packets are forwarded quickly and do not have to be kept in memory for long. Our experiments on a real testbed have shown that a buffer size of 5 is adequate for a density where each node has 8 neighbors. The method to limit the overhearing energy cost has been shown in [24]. Our prior work has shown that the resource consumption of local monitoring is acceptable even to resource-constrained WSNs. By extension, since MISPAR does not contribute significant additional overhead, it will be a resource fit for WSNs.

9 CONCLUSION

We have introduced a new class of attacks called *stealthy packet dropping* which disrupts a packet from reaching the destination by malicious behavior at an intermediate node. This can be achieved through one of four attack types–misrouting, controlling transmission power, malicious jam at an opportune time and malicious identity sharing. However, the malicious behavior *cannot be detected by any behavior-based detection scheme presented to date*. Specifically, we showed that local monitoring based detection which relies on overseeing behavior of a neighboring node cannot detect these attacks. Additionally, it will cause a legitimate node to be accused. We then presented a protocol called MISPAR based on local monitoring to remedy each attack. The solution takes two forms – having nodes maintain additional routing path information, and adding some checking responsibility to each neighbor. We showed through analysis and simulation that basic LM fails to mitigate any of the presented attacks while MISPAR successfully mitigates them.

In future work, we are considering detection techniques for multi-channel wireless networks. The listening activity for detecting malicious behavior is more complicated due to the presence of multiple channels. We also plan to analyze the impact of the detection technique on the network throughput under different adversary models.

REFERENCES

[1] A. A. Pirzada and C. McDonald, "Establishing Trust In Pure Ad-hoc Networks," in Proceedings of the 27th Australasian Computer Science Conference (ACSC 04), 26(1), pp. 47-54.

[2] S. Buchegger, J.-Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes-Fairness in Distributed Ad-hoc NeTworks," in MOBIHOC'02, pp. 80-91.

[3] Y. Huang and W. Lee, "A Cooperative Intrusion Detection System for Ad Hoc Networks," in Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN), 2003, pp. 135-147.

[4] Y. C. Hu, A. Perrig, and D. Johnson, "Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols," ACM Workshop on Wireless Security (WiSe'03), pp. 30-40.

[5] Y. C. Hu, A. Perrig, and D.B. Johnson, "Packet leashes: a defense against wormhole attacks in wireless networks," in IEEE INFOCOM, 2003, pp. 1976-986.

[6] I. Khalil, S. Bagchi, and N. Shroff, "LITEWORP: A Lightweight Countermeasure for the Wormhole Attack in Multihop Wireless Networks," in the International Conference on Dependable Systems and Networks (DSN), pp. 612-621, 2005.

[7] I. Khalil, S. Bagchi, and N. B. Shroff, "MOBIWORP: Mitigation of the Wormhole Attack in Mobile Multihop Wireless Networks," Elsevier Ad hoc Networks Journal, Volume 6, Issue 3, pp. 344-362, May 2008.

[8] I. Khalil, S. Bagchi, and C. Nina-Rotaru, "DICAS: Detection, Diagnosis and Isolation of Control Attacks in Sensor Networks," in IEEE/CreateNet SecureComm, p.p. 89-100, September, 2005.

[9] S.J. Lee and M. Gerla, "Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks," in IEEE International Conference on Communications (ICC), pp. 3201-3205, 2001.

[10] Q. Zhang, P. Wang, D. Reeves, and P. Ning, "Defending against Sybil Attacks in Sensor Networks," SDCS 2005, pp. 185-191.

[11] C. Basile, Z. Kalbarczyk, and R. K. Iyer, Neutralization of Errors and Attacks in Wireless Ad Hoc Networks, DSN 2005, pp. 518-527.

[12] B. Carbunar, I. Ioannidis and C. Nita-Rotaru, JANUS: Towards Robust and Malicious Resilient Routing in Hybrid Wireless Networks, WiSe 2004, pp. 11-20.

[13] B. Awerbuch, R. Curtmola, D. Holmer, C. Nita-Rotaru and H. Rubens, ODSBR: An On-Demand Secure Byzantine Resilient Routing Protocol for Wireless Ad Hoc Networks, to appear in ACM TISSEC journal, 2007.

[14] R. Molva, G. Tsudik, and D. Westhoff (Eds.), Statistical Wormhole Detection in Sensor Networks, ESAS 2005, LNCS 3813, pp. 128–141, 2005.

[15] D. Liu and P. Ning, Establishing Pair-wise Keys in Distributed Sensor Networks, CCS 2003, pp. 52-61.

[16] D. Johnson, D. Maltz, and J. Broch, "The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks," Ad Hoc Networking, Addison-Wesley, 2001.

[17] C. E. Perkins and E. M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," in Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), 1999, pp. 90-100.

[18] D. Ganesan, B. Krishnamurthy, A.Woo, D. Culler, D. Estrin, and S. Wicker. An empirical study of epidemic algorithms in large scale multihop wireless networks. Technical Report IntelIRP-TR-02-003, Intel Research, March 2002.

[19] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," Computer Communication Rev.'94, pp. 234–44.

[20] "The Network Simulator- ns-2," www.isi.edu/nsnam/ns/

[21] S. Bagchi, S. Hariharan, N. B. Shroff, "Secure Neighbor Discovery in Wireless Sensor Networks," Purdue University TR ECE 07-19. At "http://docs.lib.purdue.edu/ecetr/360/".

[22] R. Muraleedharan and L. A. Osadciw, "Jamming attack detection and countermeasures in wireless sensor network using ant system" in Wireless Sensing and Processing, proceedings of the SPIE, volume 6248, pp.62480G, 2006.

[23] S. Buchegger and J. L. Boudec, "Robust reputation system for p2p and mobile ad-hoc networks," in Economics of Peer-to-Peer Systems, 2004.

[24] I. Khalil, S. Bagchi, and N. Shroff, "SLAM: Sleep-Wake Aware Local Monitoring in Sensor Networks," in the 37th IEEE Dependable Systems and Networks Conference (DSN'07), p.p. 565-574, June 2007.

[25] N. Sastry, U. Shankar, and D. Wagner, "Secure verification of location claims," in ACM Workshop on Wireless Security (WiSe), pp. 1-10, 2003.

[26] L. Hu and D. Evans, "Using Directional Antennas to Prevent Wormhole attacks," in Network and Distributed System Security Symposium (NDSS), pp. 131-141, 2004.

[27] S. Hariharan, N. Shroff, and S. Bagchi, "Secure Neighbor Discovery in Wireless Sensor Networks," Purdue Technical Report, TR ECE 07-19, 2007.

[28] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in Proceedings of the 6th Annual International Conference on Mobile Computing and Networks (MOBICOM), pp. 255-265, 2000.

[29] R. de Oliveira and T. Braun, "A Dynamic Adaptive Acknowledgment Strategy for TCP over Multihop Wireless Networks," Infocom '05, pp. 1863-1874.

[30] M. Vutukuru, K. Jamieson, and H. Balakrishnan, "Harnessing Exposed Terminals in Wireless Networks," NSDI '08, pp. 59-72.