Towards Securing MintRoute in Wireless Sensor Networks

Islam Hegazy Reihaneh Safavi-Naini Carey Williamson
Department of Computer Science, University of Calgary, Calgary, AB, Canada
{islam.hegazy,rei,carey}@ucalgary.ca

***Abstract*—In a Wireless Sensor Network (WSN), the sensor nodes rely upon a multi-hop routing protocol to relay their data to the base station. However, most WSN routing protocols are vulnerable to attacks in which a malicious node can disrupt the routes, drop, modify, or divert data away from the base station. In this paper, we use the *ns-2* network simulator to demonstrate the vulnerability of the MintRoute protocol to link quality attacks by a malicious node. We then propose a novel "sequence number gap trick" as a lightweight means to test for and detect the presence of a malicious attacker. The simulation results show that judicious use of the sequence number gap trick provides robust detection of malicious nodes, preserving the data delivery capabilities of the WSN.**

I. Introduction

Wireless sensor networks (WSNs) are widely used in agricultural, environmental, industrial, and military monitoring applications. WSNs consist of many tiny sensor nodes, which collect data from their surrounding environment and send it to a base station. The sensor nodes themselves are rather limited in resources (e.g., CPU, memory, battery power), but communicate with each other and with the base station using short-range wireless communication. The base station controls the operation of the network, and is also responsible for storing and processing the collected data.

Because of the limited wireless transmission range for WSN nodes, multi-hop routing protocols are often used to transfer the collected data from the sensor nodes to the base station. Routing protocols differ in their operation. Some depend on hop count to determine routes, while others depend on available power or link quality. There are different metrics for evaluating link quality, such as signal strength, packet loss, and end-to-end delay [1]. MintRoute [2] is one example of a WSN link quality routing protocol; it uses packet loss estimate as the primary metric for determining network routes.

WSN routing protocols are susceptible to various types of attacks. An attacker may implant its own nodes in an unattended WSN. Malicious nodes may divert traffic from the base station or they may modify the collected data to cause the base station to make mis-informed decisions. Protecting the integrity of the data and the routes is vital to the successful operation of the WSN.

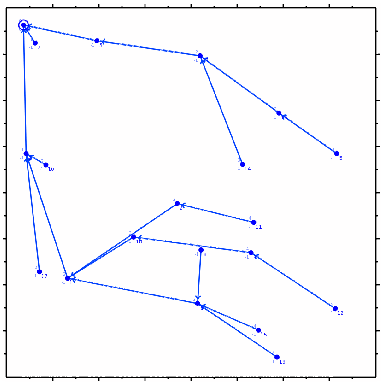
Figure 1 illustrates the basic problem that we address in this paper. Figure 1(a) shows a randomly-generated WSN topology with 20 nodes, and the (blue) routing tree established by MintRoute to reach the base station in the upper-left corner when there is no attacker in the network. Figure 1(b) shows the same WSN disrupted by a single malicious node. The attacker is located in the center of the network (indicated by the red dot), and its wireless transmission range is indicated by the red circle. The attacker advertises the highest possible link quality value in MintRoute (255), regardless of the actual packet loss rate observed. These false advertisements cause many neighbouring nodes to select the attacker as their parent node in the routing tree. As a result, many data delivery paths to the base station are disrupted. The effectiveness of the attack depends on the WSN topology, and the location of the attacker. Our simulation for 10 randomly-generated WSN topologies shows that the data delivery ratio when there is no attacker is effectively 99%. Losses are due to collisions and wireless channel errors. The data delivery ratio after the attack drops to between 20-60% depending on the WSN topology. In general, a single attacker, using this simple attack strategy, can disrupt data delivery for more than half of the WSN nodes.

Traditional security mechanisms for wired networks and ad hoc networks are rarely applicable in WSNs. As mentioned previously, sensor nodes are limited in resources, and do not have the capacity to perform complex computations. For example, using cryptographic techniques may not be viable in WSNs. Furthermore, cryptography cannot defend against all types of attacks that may occur in WSNs [3]. For instance, sensor nodes are also vulnerable to physical capture, allowing the attacker to steal cryptographic keys and compromise the network.

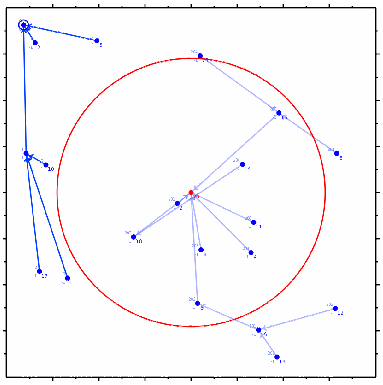
An Intrusion Detection System (IDS) can provide a complementary means of defence for WSNs. An IDS can actively or passively detect suspicious behaviours that indicate the presence of an attacker. IDSs for WSNs should be lightweight because of the limited storage and processing capabilities of the sensor nodes [4]. Traditional IDSs that use signature databases or log files are not applicable in WSNs [5]. IDSs that rely on cooperation between the nodes are also undesirable, because of the extra communication overhead and energy consumption involved.

In this paper, we introduce a lightweight IDS to detect link quality attacks on MintRoute in WSNs. Our IDS does not require cooperation between the nodes and does not add any communication overhead. In addition, it enhances the

978-1-4244-7265-9/10/$26.00



(a) Before attack



(b) After attack

Figure 1. Effect of Attack

Table I
SUMMARY OF SIMULATION RESULTS FOR WSN ATTACK SCENARIOS

Scenario	Malicious Node	Detection Mechanism	Data Delivery
0	None	None	$\approx$ 99%
1A	Simple	None	$\approx$ 20-60%
1B	Simple	Basic	$\approx$ 98%
2A	Adaptive	Basic	$\approx$ 15-97%
2B	Adaptive	Enhanced	$>$ 90%

capability of the sensor nodes to detect malicious behaviour without requiring specialized hardware.

Our paper makes three main contributions. First, we demonstrate the vulnerability of MintRoute to link quality attacks. Second, we propose a novel solution called the "sequence number gap trick". Third, we present simulation results showing that judicious use of the sequence number gap trick provides robust detection of misbehaving nodes. However, imprudent use of the approach is ineffective, since an attacker can easily detect the trick and adapt its strategy. Pedagogically, we present our results using several WSN scenarios, ranging from simple to adaptive attackers, and from basic to enhanced versions of our IDS. Table I summarizes our main simulation results.

The rest of this paper is organized as follows. Section II provides a brief survey of related work. Section III discusses the vulnerabilities of link quality routing protocols. Section IV introduces our solution, while Section V presents our simulation results. Section VI concludes the paper.

II. RELATED WORK

Many IDSs for WSNs have been developed to detect different types of attacks. However, to the best of our knowledge, most of these IDSs focus on detecting the activities of malicious nodes, rather than on the frailties of the routing protocols themselves.

Krontiris *et al.* [6] present one of the few works to detect an attack that depends on manipulating link qualities. They propose an IDS to detect sinkhole attacks that manipulate link qualities in MintRoute. In their scenario, the malicious node tries to persuade its neighbours to choose it as their parent. The malicious node impersonates the parents of its neighbours and forges weak link qualities on their behalf. Then the malicious node advertises high link quality to lure the surrounding nodes. To detect the malicious node, the neighbours broadcast their neighbour lists in response to suspicious activity. The neighbours then perform an intersection of the lists and if a node remains, they will flag it as the malicious node. This IDS fails if the intersection of the neighbour lists has more than one node. Moreover, they assume that the malicious node waits until the routing tree is constructed then begins the attack. Thus, their IDS will fail to detect malicious nodes that exist before constructing the routing tree. In addition, this IDS is limited to detecting sinkholes only.

Our IDS does not require exchanging connectivity information or location information. Moreover, we believe that our IDS is the first one to detect the manipulation in the routing protocol rather than detecting the effect of specific attacks. Thus, our IDS can prevent multiple attacks at once. Moreover, it detects attacks that exist from the beginning of the network.

III. PROBLEM STATEMENT

The attack that we consider in this paper applies in general to multi-hop WSN routing protocols that rely on link quality advertisements from the WSN nodes. MintRoute is one example of such a protocol. We use MintRoute as our illustrative example throughout the paper.

Routing protocols that depend on link quality have multiple metrics from which to choose, including signal strength, bandwidth, expected transmission count (ETX), or packet loss. However, these protocols depend on cooperation between the sensor nodes to calculate the best routes to the base station. In this section, we show how a malicious node can exploit this property.

A. Threat Model

In link quality routing protocols, sensor nodes exchange their link quality advertisements to determine good routes

to the destination. For example, in the Collection Tree Protocol (CTP) [7], each node exchanges its ETX value with its neighbours. Then each node calculates global ETX and chooses the minimum ETX to the base station. In MintRoute, each node exchanges its expected packet loss rate with its neighbours, then they calculate a total estimate for the packet loss to the base station. Each node chooses the route to the base station with the lowest packet loss.

Because of the scarcity of the resources of sensor nodes, WSN routing protocols rarely incorporate trust or security mechanisms. For link routing protocols, an untrusted protocol is itself a security threat to the WSN. Sensor nodes share information without any guarantee of the validity of this information.

The aim of a malicious node is to attract as much traffic as possible from its surrounding neighbours. It may then launch a sinkhole attack, a blackhole or selective forwarding attack, or a wormhole attack [8], [9]. An attacker may implant a new node to accomplish the attack, or compromise an existing node and reprogram it.

The strength of this attack stems from its transparency. The malicious node neither performs extra communication nor requires additional hardware and it behaves as stated in the protocol. Cryptography cannot prevent this attack because the malicious node may be an existing node that has been compromised. In this case, the malicious node has legitimate keys to communicate with the other nodes.

B. MintRoute Vulnerability

We elaborate on the operation of malicious nodes in the context of MintRoute. MintRoute is one of the main routing protocols in TinyOS [10], a popular operating system for WSNs.

In MintRoute, link quality is based on how many packets are sent and overheard successfully on the link. Thus, a node computes the number of overheard packets from a neighbour, but it relies on the neighbour to compute the number of successful transmitted packets. For example, suppose that node A and B are neighbours. Since A and B are neighbours, A can overhear all transmissions that B sends to any node. So, whenever A detects a packet transmission from B, it increments a counter of overheard packets from B. Then A computes an estimate for the incoming link quality from B using the number of overheard packets from B. Node B performs the same process regarding node A. To compute the total link quality, A needs to know its outgoing link quality to B. Thus, A depends on B to compute the outgoing link quality, which is the incoming link quality of B. Node A repeats the same process with all of its neighbours, as does node B. The problem arises if a node lies (i.e., exaggerates) about its link qualities with its neighbours to try to influence their calculations of the global link qualities.

A malicious node can lie (i.e., provide false information) about its incoming link quality. Suppose that node A is a malicious node and it sends the highest possible value for its link quality to B. Node B cannot refute or check the validity of this value. Basically, B knows how many packets it sent out but it does not know how many packets the malicious node overheard. A high link quality means that A overheard all the packets sent out by B, and did not miss any of them.

IV. Proposed Detection Mechanism

As we have seen in Section III, sensor nodes in a WSN that deploy MintRoute depend on each other in the calculation of link qualities. MintRoute does not apply any trust mechanisms. Thus, the sensor nodes cannot discover if a neighbour is lying about its link quality or not. In this section, we introduce a detection mechanism that will enhance the capabilities of sensor nodes to detect "lying" nodes.

A. System Model

We make the following assumptions about the WSN and the malicious node:

- WSN nodes are deployed uniformly at random in a planar square region. All nodes have the same wireless communication range, following the unit disk model.
- All nodes run the MintRoute routing algorithm, and have loosely synchronized clocks.
- Every node sends one data packet at a random time during a specified send interval S. The payload of each packet indicates the originator of the data. No encryption mechanism is deployed within the network.
- A single malicious node is present when the WSN is first deployed. The malicious node may be a compromised node or an implanted node. It has the same basic capabilities as legitimate sensor nodes.
- The malicious node participates in the MintRoute routing protocol, but may provide false information in its link quality advertisements. The malicious node may also drop, modify, or divert the traffic that traverses it.

B. Detection Mechanism

When the malicious node announces a high link quality to one of its neighbours, this neighbour assumes that the malicious node overheard all of its outgoing packets. There is no way for this neighbour to know if the malicious node missed any packet or not. In MintRoute, sensor nodes use sequence numbers to keep track of overheard and missed packets. The proposed detection scheme adds the ability to the sensor nodes to use their sequence numbers to detect neighbours that misrepresent their link quality.

A node can play a "trick" by introducing an artificial gap in its sequence number space. This gap implies a lower bound on the number of missed packets that its neighbours perceive. Now, the tricking node has a minimum number

of missed packets, the number of packets it sent, and the previous link quality of each neighbour. The tricking node estimates an upper bound for the next link quality report from each neighbour. When the tricking node receives a new link quality, it compares the received link quality with the estimated link quality of the corresponding neighbour. If the received link quality is larger than the estimated link quality, the corresponding neighbour will be suspected as malicious. The neighbours of the tricking node cannot tell if a packet has actually been missed or a sequence number gap trick is being played. If a malicious node is lying about its link quality, it will ignore the indicated miss and announce a high link quality to the tricking node. Once a malicious node is detected, its neighbour blacklists it. If any of these neighbours is a child of the malicious node, the neighbour will change its parent to avoid the malicious node.

V. Simulation Results

We simulated our IDS using *ns*-2 [11]. We used different thresholds for the attacker to determine whether to lie or not. To measure the success of our detection scheme, we configured *m* to divert traffic to a fake node. In other words, *m* behaved like a legitimate node, accepting traffic from its neighbours and forwarding it. With this approach, we could measure the delivery ratio of data at the base station with and without our detection scheme.

A. Simulation Setup

We simulated a WSN of 20 sensor nodes in an area of 100 $\times$ 100 units. The base station was placed in the northwest corner of the simulation area. The nodes had a communication range of 40 units. The malicious node was placed in the middle of the network and had the same communication range as the legitimate nodes. We simulated 10 different topologies such that *m* had randomly varied neighbourhood sizes. The nodes were configured to send a route update every 5 time units. The new estimate interval was E = 50 time units and the data send interval was S = 20 time units. Each simulation ran for 5000 time units, during which each node estimated link qualities 100 times. Our results represent the average values calculated from 25 independent runs of each configuration.

The primary performance metric is the data delivery ratio (i.e., the proportion of the WSN data that is successfully delivered to the base station). The simulator is instrumented to record the number of sequence number gap tricks that are played by each node, the number of lies by the attacker, and the volume of data delivered. As secondary metrics, we calculate the number of true positives, false positives, true negatives, and false negatives for the detection mechanism, which we use in the validation of our simulator. For space reasons, we do not report these values here.

Next, we incrementally present several WSN scenarios in which a malicious node attacks MintRoute, showing whether the malicious node is detected, and how the malicious node can try to escape detection. Table I summarizes the considered scenarios.

1) Scenario 0: As a baseline, we consider a WSN using MintRoute when there is no attacker. The sensor nodes build a routing tree rooted at the base station. The best route in MintRoute is calculated as the route with the least packet loss. To calculate the packet loss on a link, two neighbouring nodes calculate an estimate for the incoming link quality using the number of packets they overheard and missed on that link. The two neighbouring nodes then exchange their incoming link qualities to be the outgoing link quality in the corresponding neighbour. They also exchange the total cost of their respective routes to the base station. The nodes then calculate a route cost (quality) using the incoming and outgoing link qualities and the exchanged total route cost.

MintRoute offers a stable and robust routing tree that changes only when the quality of a link deteriorates. Our simulations show over 99% data delivery to the base station in WSNs using MintRoute. See the brown bars in Figure 2(a).

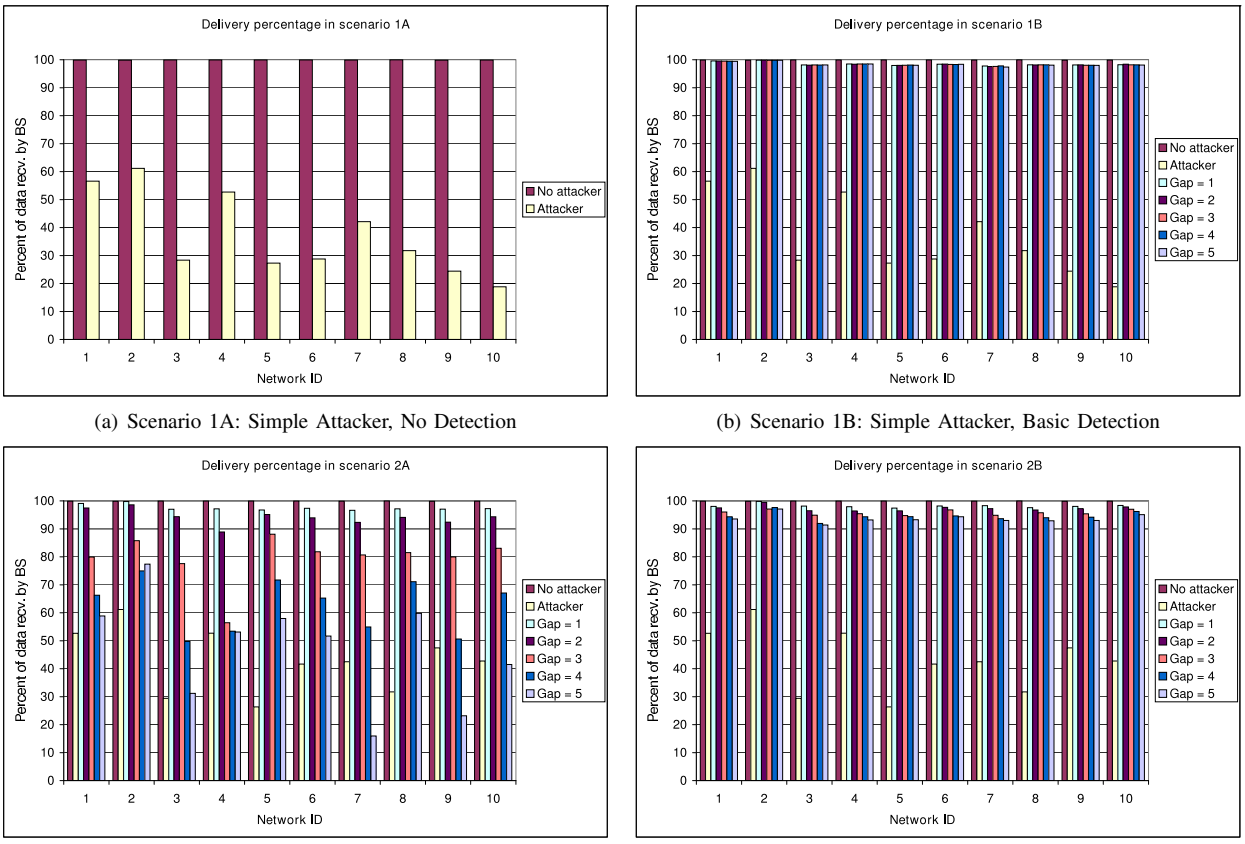
2) Scenario 1A: A malicious node, *m*, is introduced to the network with the aim of diverting as much traffic as possible. To achieve this goal, *m* lies about its incoming link qualities, which are the outgoing link qualities of its neighbours. The malicious node uses a very simple attack strategy, in which it always announces the highest possible link quality (255).

Our simulations show that the delivery ratio drops to 20% - 60% of the WSN traffic, depending on the size of the neighbourhood of *m* and its location. See the yellow bars in Figure 2(a).

3) Scenario 1B: In this scenario, *m* behaves as in Scenario 1A. However, the sensor nodes use the sequence number gap trick to try to detect it. The sensor nodes are configured to play the sequence number gap trick once (at a random time) during each interval E. This ensures that only one sequence number gap trick is used in the (per-neighbour) calculations of *m* and, at the same time, it does not severely affect the link qualities. Since *m* is passive, it ignores any missed packets and announces the highest possible link quality, which makes the malicious node easily detectable.

Figure 2(b) shows that data delivery ratio is restored to 98% or better in each of the 10 WSN topologies tested. In other words, the sequence number gap trick is a simple and effective means to detect the malicious node.

4) Scenario 2A: While *m* in Scenario 1B is easily detected, it is reasonable to consider a more sophisticated attacker. In particular, *m* in Scenario 2A is adaptive. It tries to detect when the sequence number gap trick is being played, and tell the truth about its link quality in this case. It is then able to avoid detection by the sequence number gap trick some of the time. See Figure 3(a).



(a) Scenario 1A: Simple Attacker, No Detection

(b) Scenario 1B: Simple Attacker, Basic Detection

(c) Scenario 2A: Adaptive Attacker, Basic Detection

(d) Scenario 2B: Adaptive Attacker, Enhanced Detection

Figure 2. Data Delivery Ratio Results for Different WSN Attack Scenarios

To elaborate, *m* calculates a threshold τ to represent the number of packets it expects to hear from each neighbour. Then it decides whether to lie or not depending on the observed packet traffic relative to the threshold. In essence, if unusually many packets are "missing", *m* suspects that a trick is being played, and tells the truth in order to avoid detection. Mathematically, *m* may use a threshold such as $E[X] + \sigma$, where $E[X]$ is the expected number of packets, and σ is the standard deviation.

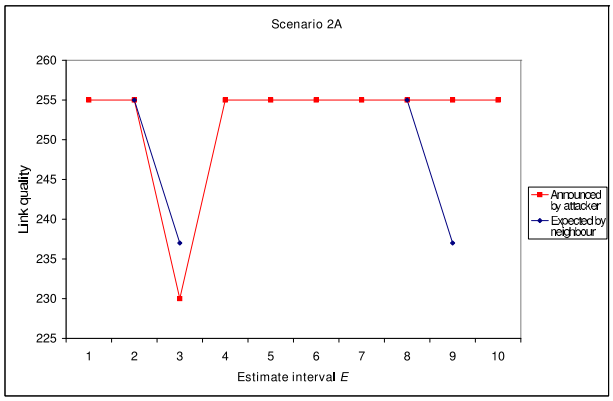
With a sequence number gap of size 1, the simulation results show that the data delivery ratio is about 97% across the 10 WSN topologies considered. See Figure 2(c). However, the data delivery ratio drops sharply as the sequence number gap size increases. The malicious node can detect the large sequence number gaps and thus it lies less often. For the largest sequence number gap size considered (5), the data delivery ratio approaches that of Scenario 1A. In other words, the sequence number gap trick is most effective with a small sequence number gap of size 1.

5) Scenario 2B: One weakness of the sequence number gap trick used in Scenario 2A is that the legitimate nodes play the trick only once per interval *E*. Furthermore, they only monitor the attacker's link quality advertisement immediately following a sequence number gap trick, rather than continuously.

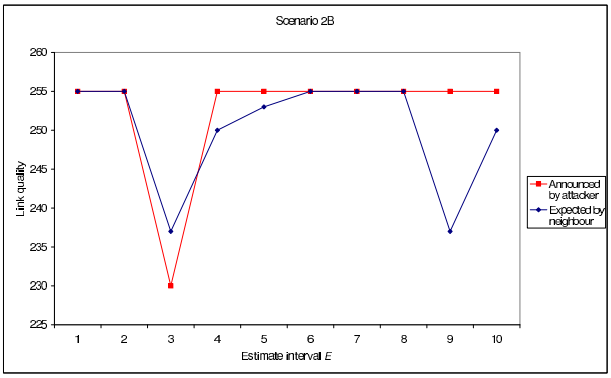
The obvious solution is to make sensor nodes more vigilant. In particular, because the link quality calculation in MintRoute uses an exponentially weighted moving average, it is impossible for a link quality estimate to jump dramatically from one interval to the next. An enhanced detection mechanism can monitor link quality more closely and detect an attacker, as illustrated in Figure 3(b).

The simulation results for this scenario show that the enhanced detection mechanism works well. For a sequence number gap of size 1, Figure 2(d) shows that the data delivery ratio is about 98% across the 10 WSN topologies considered. Furthermore, the approach works even for larger sequence number gap sizes, though there is still a slight decline in effectiveness as the gap size increases.

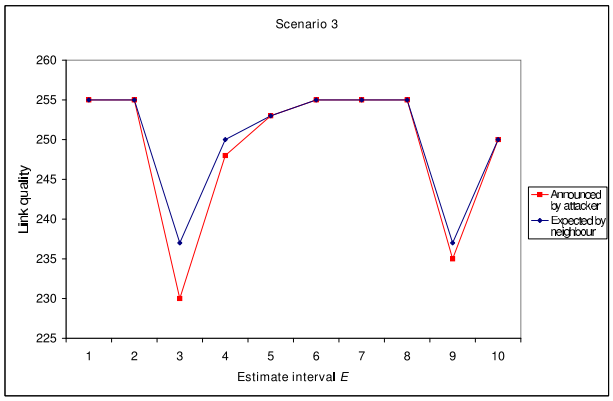
The enhancement to the IDS mechanism in Scenario 2B can also be applied to *m* itself. That is, following a (detected) sequence number gap trick, *m* can cautiously increase its link quality back to the maximum so as to avoid suspicion. See Figure 3(c). This strategy has effectively converted *m* into a conformant node.



(a) Scenario 2A: Adaptive Attacker, Basic Detection



(b) Scenario 2B: Adaptive Attacker, Enhanced Detection



(c) Conformant Node

Figure 3. Examples of Several WSN Attack Scenarios

VI. Conclusion

In this paper, we showed a vulnerability in link quality routing protocols that can be used to launch sinkhole attacks in WSNs. We described the threat that this vulnerability poses for WSNs and described our proposed IDS to discover malicious nodes that make use of this vulnerability. We used the *ns-2* network simulator to simulate our IDS in a WSN using the MintRoute routing protocol. Our results showed that our IDS was effective in detecting the malicious node.

Acknowledgment

This work is supported by Informatics Circle of Research Excellence (iCORE), Alberta, Canada. Islam Hegazy is on leave from FCIS, Ain Shams University, Cairo, Egypt for his PhD.

References

[1] J. N. Al-Karaki and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2004.

[2] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys)*. ACM press, 2003, pp. 14–27.

[3] C. H. Tseng, S.-H. Wang, C. Ko, and K. Levitt, "DEMEM: Distributed Evidence-Driven Message Exchange Intrusion Detection Model for MANET," in *Proceedings of the 9th International Symposium on Recent Advances in Intrusion Detection (RAID)*, ser. Lecture Notes in Computer Science, vol. 4219. Springer Berlin/Heidelberg, September 2006, pp. 249–271.

[4] R. A. Shaikh, S. Lee, Y. J. Song, and Y. Zhung, "Securing Distributed Wireless Sensor Networks: Issues and Guidelines," in *Proceedings of IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC)*, 2006, pp. 226–231.

[5] A. Perrig, J. Stankovic, and D. Wagner, "Security in Wireless Sensor Networks," *Communications of the ACM*, vol. 47, no. 6, pp. 53–57, June 2004.

[6] I. Krontiris, T. Dimitriou, T. Giannetsos, and M. Mpasoukos, "Intrusion Detection of Sinkhole Attacks in Wireless Sensor Networks," in *Proceedings of the 3rd International Workshop on Algorithmic Aspects of Wireless Sensor Networks (AlgoSensors)*, ser. LNCS. Springer, July 2007.

[7] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*. Berkeley, California: ACM, 2009, pp. 1–14.

[8] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," in *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications*, May 2003, pp. 113–127.

[9] E. Shi and A. Perrig, "Designing Secure Sensor Networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 38–43, December 2004.

[10] TinyOS, "http://www.tinyos.net/."

[11] ns-2, "http://www.isi.edu/nsnam/ns/."