Launching a Sinkhole Attack in Wireless Sensor Networks; the Intruder Side

Ioannis Krontiris, Thanassis Giannetsos, Tassos Dimitriou
Athens Information Technology, 19002 Peania, Athens, Greece
Email: {ikro,agia,tdim}@ait.edu.gr

***Abstract*—One of the reasons that the research of intrusion detection in wireless sensor networks has not advanced significantly is that the concept of "intrusion" is not clear in these networks. In this paper we investigate in depth one of the most severe attacks against sensor networks, namely the sinkhole attack, and we emphasize on strategies that an attacker can follow to successfully launch such an attack. Then we propose specific detection rules that can make legitimate nodes become aware of the threat, while the attack is still taking place. Finally, we demonstrate the attack and present some implementation details that emphasize the little effort that an attacker would need to put in order to break into a realistic sensor network.**

I. Introduction

The pervasive interconnection of wireless sensor devices has given birth to a broad class of exciting new applications in several areas of our lives. However, as every network, sensor networks are exposed to security threats which, if not properly addressed, can exclude them from being deployed in the envisaged scenarios. Their wireless and distributed nature and the serious constraints in node battery power prevent previously known security approaches to be deployed and has created a large number of vulnerabilities that attackers can exploit in order to gain access in the network and the information transferred within.

Securing sensor networks against these threats is a challenging research area, necessary for commercially attractive deployments. Encryption and authentication mechanisms provide reasonable defense for mote-class outsider attacks. However, cryptography is inefficient in protecting against laptop-class and insider attacks. It remains an open problem for additional research and development since the presence of insiders significantly lessens the effectiveness of link layer security mechanisms. This is because an insider is allowed to participate in the network and have complete access to any messages routed through the network and is free to modify, suppress, or eavesdrop on the contents.

What makes it even easier for attackers is the fact that most protocols for sensor networks are not designed having security threats in mind. As a consequence, deployments of sensor networks rarely include security protection and little or no effort is usually required from the side of the attacker to perform the attack. So, it is very important to study realistic attacker models and evaluate the practicality and efficiency of certain attacks.

This paper investigates one of the most severe routing attacks in sensor networks, namely the sinkhole attack [1], from the attacker's point of view. Our goal is to describe the most effective ways to launch this attack and demonstrate them in practice. We reveal the weaknesses of the routing protocols that are most widely used by the research community, hoping that this will lead to a better awareness of the threats and the study of more efficient security protocols. Then we propose some countermeasures against these threats in the direction of intrusion detection. Some first intrusion detection systems have started to appear for sensor networks, but rarely do they include specific detection rules [2]. Rules against specific attacks, like the one we present here, if properly generalized could lead to better and more realistic IDS designs.

This paper is structured as follows: In Section II, we state the routing protocols considered here and why they were chosen. Then we state our assumption about the attacker in Section III and previous work in Section IV. Section V describes the sinkhole attack in detail and how it can be launched successfully against the considered routing protocols. Then, in Section VI, we discuss specific detection rules that could make legitimate nodes aware that such an attack takes place in their neighborhood. Finally, Section VII demonstrates our implementation of the attack and Section VIII concludes the paper.

II. The Routing Layer Model

There appears to be a great diversity in deployed routing protocols for sensor networks. In this paper we concentrate on the two most popular ones, the MintRoute protocol, and the MultiHopLQI. MintRoute is used in most real sensor networks deployments today, as for example in [3]–[5] and has also served as the basis for TinyOS 2.x Collection Tree Protocol.

Many current sensor node platforms, like the micaZ, Telos and Intel Mote2, use the same radio chip, the CC2420. This radio chip provides a hardware indicator, called Link Quality Indicator (LQI), which is believed to be a better indicator of link quality than RSSI. This has led several routing protocols to adopt LQI, indicated by the last packet as the criterion for parent selection. In this paper we investigate one of these protocols, the MultiHopLQI [6], which is widely used in MoteIV Tmote Sky. It has been also used in several sensor network deployments [7]–[9]. The Drain collection protocol is a derivative of MultiHopLQI and served as the basis for the TinyOS 2.x dissemination service. Several other custom routing protocols were designed based on MintRoute and MultiHopLQI.

III. Threat Model

In this paper we assume the presence of an attacker that can access (and eventually change) the internal state of a sensor node. This type of attack is referred to as *node capture* in the literature [10]. Most existing routing schemes for sensor networks can be substantially influenced, even if the attacker captures one node or a minute portion of the network [1]. So, for the attacks described here we will assume that the attacker has captured just one node (although the same setting can be generalized to more nodes), which was previously a legitimate member of the network. To avoid detection, we assume that the attacker does not reprogram the memory of the node, but she rather connects the node to a laptop in order to monitor the packets received. Then she can change the contents of the packets and resend them using the attached node. Therefore, the attacker has access only to her immediate vicinity and does not use a stronger transmitter or an outbound communication channel.

IV. Related Work

A first approach on the detection of sinkhole attacks has been presented by Ngai et. al. [11]. This approach involves the base station in the detection process, resulting in a high communication cost for the protocol. The base station floods the network with a request message containing the IDs of the affected nodes. The affected nodes reply to the base station with a message containing their IDs, ID of the next hop and the associated cost. The received information is then used from the base station to construct a network flow graph for identifying the sinkhole.

Other existing protocols build detecting mechanisms for sinkhole attacks in sensor networks that are based on routing protocols usually deployed in Ad-Hoc networks, like the Ad-hoc On-demand Distance Vector Protocol (AODV) [12] and the Dynamic Source Routing (DSR) Protocol [13].

In our experience, routing protocols specifically designed for sensor networks, like MintRoute and MultiHopLQI, require much less resources and are usually preferred for such networks. A first effort to detect Sinkhole attacks on MintRoute was made in one of our earlier publications [14]. We include this discussion here also in a more formal way and complement it with new ways to launch this attack and detect it.

V. The Sinkhole Attack

The *sinkhole* attack is a particularly severe attack that prevents the base station from obtaining complete and correct sensing data, thus forming a serious threat to higher-layer applications. In a Sinkhole attack [1], a compromised node tries to draw all or as much traffic as possible from a particular area, by making itself look attractive to the surrounding nodes with respect to the routing metric. As a result, the adversary manages to attract all traffic that is destined to the base station. By taking part in the routing process, she can then launch more severe attacks, like selective forwarding, modifying or even dropping the packets coming through.

A. Sinkhole Attack on MintRoute

MintRoute uses link quality estimates as the routing cost metric to build the routing tree toward the base station. For the calculation of these link estimates, MintRoute uses the packet *error rate*. The nodes periodically transmit a packet, called "*route update*" and each node estimates the link quality of its neighbors based on the *packet loss* of the packets received from each corresponding neighbor. The list of these estimates for each neighbor is broadcasted by the node periodically in its route update packets.

Every node maintains a Neighbor Table and updates it when it receives a route update packet. This table stores a list with the IDs of all neighboring nodes and their corresponding link costs. The node chooses its "parent node" to be the one with the best link quality in the Neighbor Table. Note that the hop distance of each neighbor to the base station is not taken under consideration in choosing the parent, unless two nodes have the same link quality.

The parent changing mechanism is triggered every time the link quality of one or more nodes becomes 75% better than the link quality of the current parent, or the link quality of the current parent drops below 25 in absolute value (with 255 being the maximum value). In such case, the node with the highest quality becomes the new parent. However, if two of such candidate nodes happen to have the same link quality, the new parent will be the one with the smaller hop count to the base station.

In the case of a routing protocol, like MintRoute that uses link estimates as the routing metric, the compromised node launching the sinkhole attack will try to persuade its neighbors to change their current parents and choose the sinkhole node as their new one. There are two ways to do that:

1) Advertise an attractive link quality for itself,
2) Make other nodes look like they have worse link quality than itself.

Note that the attacker cannot launch a sinkhole attack by advertising that it has a lower hop count to the base station, as this metric is not the primary criterion in this routing protocol. So the attacker needs to come up with more sophisticated ways.

Moreover, just advertising a high link quality to the other nodes may not be enough, since most of these routing protocols try to be robust, meaning that they don't allow the nodes to change parents frequently and for no good reason. For example, when a node changes its parent, this could create a routing cycle in the network, which is followed by an extra cost to resolve it. Therefore, aside from advertising a high link quality for itself, another way for the attacking node to launch the sinkhole attack is to make the current parents look like they have a very poor link quality, which will trigger the parent changing mechanism in their children. Then the new parent to be chosen will be the sinkhole node.

The way to do that is to change the link quality estimates sent by the parent nodes, within their route update packets. The attacker listens to the route update messages from its

neighbors, alters them and replays them impersonating the original sender. Even if there is an underlying key mechanism that nodes can use to communicate securely with each other, most probably the attacker will be using a broadcast key shared with the nodes to be able to overhear, change and send these packets.

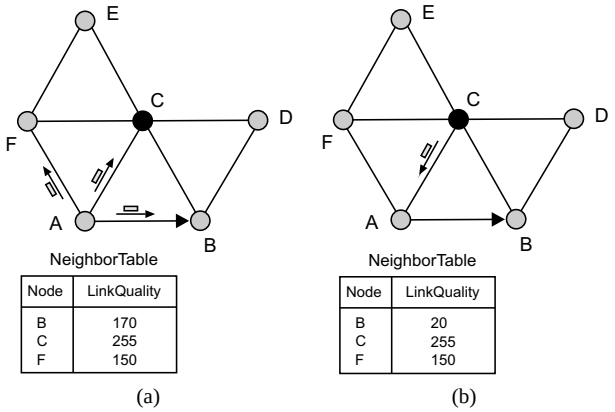


Fig. 1: The two phases of sinkhole attack on MintRoute. (a) Node C (attacker) receives the route update packet of node A. (b) Node C sends a forged packet to A, impersonating B. In both cases the Neighbor Table of node A is indicated.

Let's take for example the case shown in Figure 1, where node C is the attacker and node B is the current parent of node A. Node C has sent its own route update packet advertising a fake link quality (at the maximum value of 255), but this is not enough to make node A change its parent. Therefore, when it receives the route update packet of node A, it changes the link quality of node B to a low value and sends it back to A as a unicast packet, impersonating B. Upon receiving this packet, node A thinks it is a route update packet from B, it extracts the link quality estimation and updates the corresponding entry in the Neighbor Table. This will trigger the parent changing mechanism and since the link quality of node B is below 25, that node will be ignored in the selection algorithm and node C will be chosen.

After performing the above attack for all of its neighbors, the Sinkhole node will eventually attract the traffic passing through these nodes.

B. Sinkhole Attack on MultiHopLQI

From what we described in the previous section, the weakness of MintRoute is that each node is based on the advertised link quality from other nodes to decide on its parent. In MultiHopLQI, the nodes calculate the link quality based on their own hardware. Each node periodically broadcasts a *beacon* message and the receivers extract the LQI given by their radio chip. This number is given to a function that calculates the *cost* of the corresponding link. The cost is inversely proportional to the LQI. The most attractive link is the one with the lowest cost. In what follows, we will use the notation $Cost_{AB}$ to indicate the cost estimation of node A for the link between itself and B.

The payload of the beacon message includes the sender's current parent and a cost for the whole path to the base station (i.e., the *path cost*). This cost is calculated as the sum of all the costs of the links that make the path. For a node B that has a parent D, its path cost is calculated as

$$Cost_B = Cost_{BD} + Cost_D \quad (1)$$

The value of $Cost_B$ is included in the beacon of node B. Node A that receives the beacon, reads and stores the value in a table. It also calculates $Cost_{AB}$ as we described above and calculates its own path cost, $Cost_A$, using Equation (1). Node A chooses as its parent the node that minimizes $Cost_A$.

According to this algorithm, we identify three ways for an attacker C to launch the sinkhole attack:

1) Advertise a low path cost with its parent,
2) Make other nodes look like they have worse path costs than itself,
3) Change its parent to the neighbor with the minimum path cost.

Let us describe each of the above strategies using the example shown in Figure 2. Let's suppose that initially the nodes have chosen their parents as depicted in Figure 2(a). The path costs for each node are also indicated. Node C is compromised by an attacker and her goal according to the sinkhole attack is to attract as much traffic as possible from the neighboring nodes, convincing them to choose C as their parents.

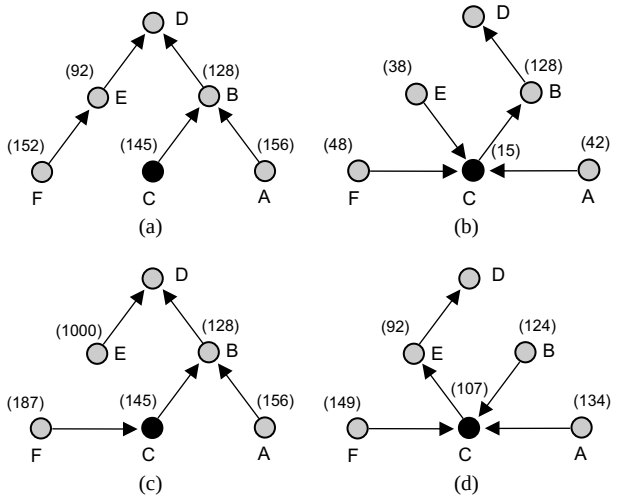


Fig. 2: Three sinkhole attacks on MultiHopLQI. Case (a) shows the original settings of the network before the attack, while cases (b), (c) and (d) show the result of each of the three strategies.

The first and easiest way is to advertise the minimum path cost to the base station. This is shown in Figure 2(b). According to the function built in MultiHopLQI, the path cost that corresponds to the maximum LQI is 15. The result of this attack is that nodes A, E and F change their parents to C, as this reduces their corresponding path costs. This will also trigger the parent changing mechanism at the parent of

the attacker, node B. However, choosing any of the children of C will result in the formation of a routing cycle since B is the attacker's parent, and eventually will be forced to go back to its old parent D. In the experiments, we noticed that this behavior of B kept repeating, however according to the routing protocol, it is legitimate, so we consider that the goal of the attack has been reached.

The second way to launch a sinkhole attack is for node C to impersonate a node and advertise a very high path cost on its behalf. For example, in Figure 2(c), the attacker broadcasts beacons impersonating node E and advertises a path cost equal to, let's say, 1000. Its child F updates its own path cost to $1000 + \mathit{Cost}_{EF}$ and realizes that choosing node C as its parent will reduce it substantially. Since node E will keep broadcasting its legitimate beacons periodically (with path cost 92), the attacker needs to do the same with its spoofed messages, immediately after the messages of E. This will keep Cost_E in the memory of node F at the attacker's desirable value. If node C follows the same strategy for each node in its vicinity, it will manage to attract all the traffic.

The third strategy for the attacker is to look for the node with the minimum path cost in the neighborhood and advertise the best possible, but also legitimate, path cost for itself. For example, in Figure 2(d), node E has the best path cost. In this network, it is the case that

$$\mathit{Cost}_E + \mathit{Cost}_{EC} > \mathit{Cost}_B + \mathit{Cost}_{BC},$$

so node C had chosen B as its parent. For the attack, however, node C chooses E as its parent and advertises a very attractive path cost, i.e., $\mathit{Cost}_E + 15$. This is much less than the path cost it was advertising before. The neighbors will update this value in their tables and hopefully their corresponding path costs will drop by choosing C as their parents, as it is the case with Figure 2(d). In Section VI, we will see that this form of attack is the hardest to detect, since the attacker does nothing that is not legitimate.

VI. Detecting the Sinkhole Attack

Based on the vulnerabilities of the routing protocols that we exposed in the previous section, we now move a step further and propose specific rules that can be used to detect the attack. Since all communication in a WSN is conducted over the air, nodes can listen on the network and capture and examine individual packets passing from their immediate neighborhood in real time. So, the question that we try to answer in this section is whether nodes can autonomously realize that a sinkhole attack takes place in their neighborhood, *without* the help of the base station or clusterheads.

A. Detection on MintRoute

In order to detect the sinkhole attack on MintRoute, we add a rule that will trigger an alert whenever a malicious node tries to impersonate another node, according to the attack we described in Section V-A. The intuition is that route update packets should originate only from their legitimate sender and the nodes should defend against impersonation attacks.

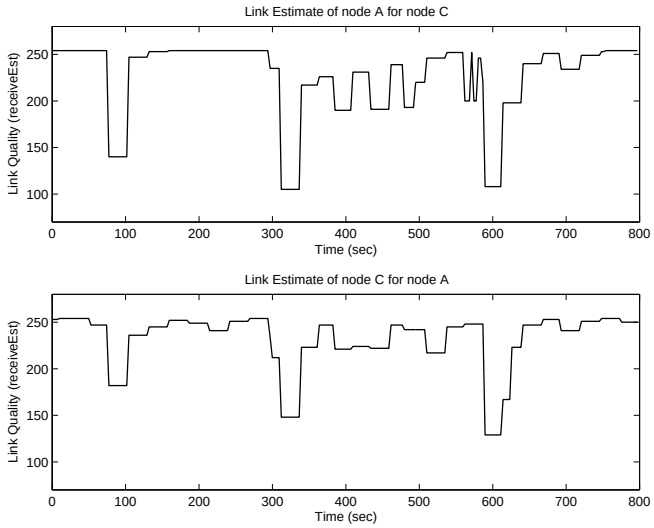


Fig. 3: The estimates of node A and node C for the quality of the link between them, based on the packet loss rate.

Rule 1: For each overheard route update packet, check the sender field, which must belong to one of your neighbors.

For the example shown in Figure 1, the rule will be triggered in node B, since it will overhear the packet sent by node C impersonating B. It will be also triggered in nodes E and F, which will overhear a packet from node B without being neighbors of B. As the attacking node tries to acquire more nodes using this method, the rule will be triggered in more of its neighbors.

There is also another rule that can be used by legitimate nodes, which is based on anomaly detection. In particular, we make the following observation: according to MintRoute, each node independently measures the link quality estimate of each neighbor and receives their estimates through the route update packets. As one expects, these values cannot have a big deviation from each other. For example, let's take the link between the two nodes, A and C of the network in Figure 1. Figure 3 shows the estimate of node A for the quality of that link and the estimate of node C for the same link and for the same period of time. As it is expected, the estimates of the two nodes for the same link are almost the same, with some small deviation. In particular, the maximum difference that we found between the two link estimates was 49, which corresponds to 19.2%.

As we said in Section V-A, an attacking node may try to advertise a very high link quality for itself, hoping that this value will be more than 75% better than the link quality estimate of a node for its current parent. This advertisement is overheard by its neighbors. However, this value will not correspond to the link quality estimate that the node has for the link with the attacking node. This observation can help us define a second rule, as follows.

Rule 2: For each $[parent, child]$ pair of your neighbors, compare the link quality estimate they advertise for the link between them. Their difference cannot exceed 50.

Let us note that for a node that detects an anomaly according to the above rules, it is only an *indication* that a sinkhole attack is in progress. For example, using Rule 1, there is no way to know which node is trying to launch the attack, since the sender field is altered. The only conclusion that can be drawn so far is that the attacker is one of the neighboring nodes, since the route update packets are only broadcasted locally. Similarly, for Rule 2, a monitoring node cannot know which of the two nodes advertises fake link quality. However, we believe that this is a promising research direction and if these rules are used properly in an intrusion detection system that induces *collaboration* with other nodes in the area, successful detection can occur [14], [15].

B. Detection on MultiHopLQI

Since some of the attacker's strategies are common between the two routing protocols, the corresponding rules can also be applied to detect the sinkhole attack in MultiHopLQI. In particular, in the case that the attacker tries to impersonate another node, and advertise a high path cost (ref. Figure 2(c)), Rule 1 from the previous section can be applied here as well.

For the strategy described in Figure 2(b), where the attacker advertises the minimum path cost, there is an inconsistency in the protocol itself that we can take advantage and define a new rule. We notice that the path costs should be increasing as we move more hops away from the base station. In other words, each node should be advertising a bigger path cost than its parent, as it is derived by Equation (1). In this attack, it's not hard to see that this condition is violated. According to the description of the attack, the attacker advertises a path cost which is smaller that its parent. The nodes that are neighbors of both the attacker and its parent have their path costs stored in their memory, according to the protocol. So they could apply the following rule and detect the attacker:

Rule 3: For each beacon, check that the advertised path cost of the node is bigger than the path cost of its father.

If this rule is violated, one of the two nodes lies about its path cost and it has to be the one that advertises the smaller cost. In a different case, in which the attacker for whatever reason advertises a bigger path cost than its legitimate child, that child would immediately update its path cost according to Equation (1) and may trigger the parent changing mechanism, depending on the result. In any case however, the rule would not be violated. So this rule can lead to immediate detection.

Detecting the third attack that we described in Section V-B for MultiHopLQI is more difficult, because as we said, the attacker advertises a path cost that is within the limits and is higher than the cost of its parent, as it is supposed to be. However, the advertised cost is still fake and does not correspond to the real link quality, so, like we did in the case of MintRoute, we turn to anomaly detection. We made the same experiment and compared the LQI of two nodes, E and C, for the link between them. As shown in Figure 4, they are the same, except for a small deviation. The maximum observed difference was 7. So, for MultiHopLQI, we can define an equivalent rule with Rule 2, as follows.

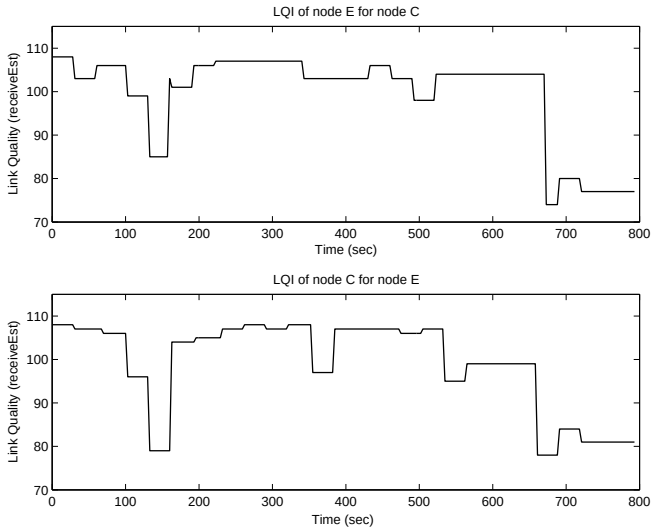


Fig. 4: The estimates of node E and node C for the quality of the link between them, based on the LQI.

Rule 4: For each $[parent, child]$ pair of your neighbors, compare the LQI they advertise for the link between them. Their difference cannot exceed 10.

The only problem about applying this rule in practice is that nodes in MultiHopLQI do not advertise the LQI that they calculate for their links. We strongly suggest this modification for future designs of similar routing protocols. Alternatively, such a mechanism could be embedded in an IDS system.

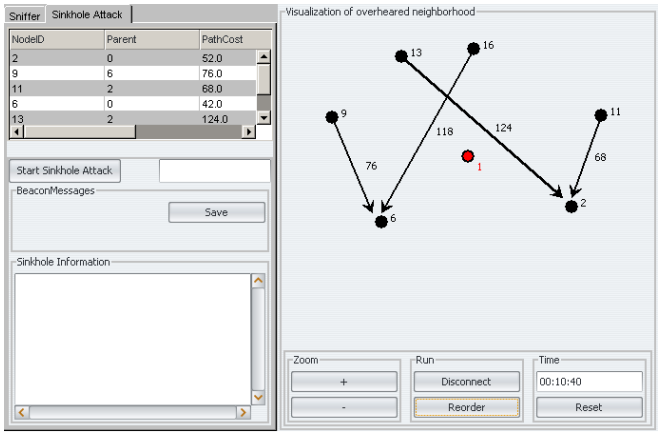
VII. Implementation of the Attack

In this section we demonstrate the sinkhole attack on a real sensor network deployment, describing the details of our setting and the tools that we developed. In particular, we have decided to implement the third strategy described in Section V-B, as it is the most interesting and difficult to detect. In this way, we make it even clearer that launching a sinkhole attack against an unprotected network is a trivial task for an attacker and stress the need for security measures.

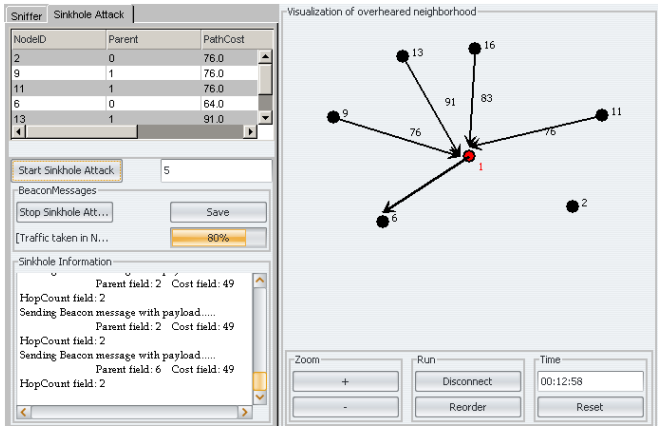
In our implementation we used Tmote Sky nodes from MoteIV. We took the side of the attacker and we placed our node inside a sensor network that was set up using MultiHop-LQI as the routing protocol and Delta at the application layer. We assumed no previous knowledge for the network, so we first built a packet sniffer, in order to acquire the necessary information for the attack. As we said, MultihopLQI periodically transmits a beacon message containing information of the node's parent path cost. These packets can be used to reconstruct the network connectivity from the sniffer traces.

The sniffer is implemented as a modified version of the TinyOS TOSBase (standard application distributed with TinyOS), only that the code was modified to pass every packet received over the radio to the serial port, regardless of destination address or AM group ID. In this way we had all overheard packets forwarded to our laptop. A graphical user interface of our Java application revealing the IDs of the nodes

in our vicinity, their parents and the path costs is shown in Figure 5(a). The base station does not appear in the topology, since it is 2-hops from our "malicious" node.



(a)



(b)

Fig. 5: Launching the sinkhole attack on a real deployed network, using MultiHopLQI. (a) The network topology before the attack. (b) The network topology after the attack.

After acquiring the necessary information, the first step for the attack consists of identifying the node with the smallest path cost. This is done automatically, after listening to broadcast packets. In the example of Figure 5, it is node 6, with $Cost_6 = 42$. The second step is to broadcast beacons through our node, advertising that our parent is node 6, and our path cost to it is $Cost_1 = Cost_6 + 15 = 57$. The result is shown in Figure 5(b). All nodes, except node 2, minimized their path costs by choosing our node as their new parent. We intentionally chose to include this example, to show that the attack does not necessarily succeed to attract all nodes. It solely depends on the path cost that the nodes had with their parents. Let us note however that by trying different topologies, we rarely missed attracting 100% of the nodes.

VIII. Conclusions

In this paper we identified several vulnerabilities of two popular routing protocols of sensor networks and showed how they can be exploited by an attacker to launch a sinkhole attack. It turns out that the effort the attacker has to put is minimal, and the attack can go undetected, unless certain detection rules are applied. We identified such rules and highlighted any modifications that are necessary for the routing protocols. The results of this work serve a two-fold purpose: they motivate a better design of routing protocols that can make them more resilient to attacks and they also open the way for defining more general and formal rules in intrusion detection systems.

Acknowledgements

The authors would like to thank the reviewers for their helpful comments.

References

[1] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," *AdHoc Networks Journal*, vol. 1, no. 2–3, pp. 293–315, September 2003.

[2] R. Roman, J. Zhou, and J. Lopez, "Applying intrusion detection systems to wireless sensor networks," in *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC '06)*, Las Vegas, USA, January 2006, pp. 640–644.

[3] G. Werner-Allen, K. Lorincz, M. Welsh, O. Marcillo, J. Johnson, M. Ruiz, and J. Lees, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing*, vol. 10, no. 2, pp. 18–25, 2006.

[4] T. Schmid, H. Dubois-Ferrière, and M. Vetterli, "SensorScope: Experiences with a Wireless Building Monitoring Sensor Network," in *Proceeding of the Workshop on Real-World Wireless Sensor Networks (REALWSN '05)*, Stockholm, Sweden, June 2005.

[5] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Wireless sensor networks for structural health monitoring," in *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, 2006, pp. 427–428.

[6] TinyOS, http://www.tinyos.net/tinyos-1.x/tos/lib/MultiHopLQI, 2004.

[7] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *OSDI '06: Proceedings of the 7th symposium on Operating systems design and implementation*. Berkeley, CA, USA: USENIX Association, 2006.

[8] G. Giorgetti, S. Mastroianni, J. Lewis, G. Manes, and S. Gupta, "The personal sensor network: A user-centric monitoring solution," in *BodyNets '07: Proceedings of the 2nd International Conference on Body Area Networks*, 2007.

[9] J. Paek and R. Govindan, "RCRT: rate-controlled reliable transport for wireless sensor networks," in *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2007, pp. 305–319.

[10] A. Perrig, J. Stankovic, and D. Wagner, "Security in wireless sensor networks," *Communications of the ACM*, vol. 47, no. 6, pp. 53–57, 2004.

[11] E. C. H. Ngai, J. Liu, and M. R. Lyu, "On the intruder detection for sinkhole attack in wireless sensor networks," in *Proceedings of the IEEE International Conference on Communications (ICC '06)*, Istanbul, Turkey, June 2006.

[12] D. Dallas, C. Leckie, and K. Ramamohanarao, "Hop-count monitoring: Detecting sinkhole attacks in wireless sensor networks," in *ICON '07: Proceedings of the 15th IEEE International Conference on Networks*, Adelaide, SA, 2007, pp. 176–181.

[13] A. A. Pirzada and C. McDonald, "Circumventing sinkholes and wormholes in wireless sensor networks," in *IWWAN '05: Proceedings of International Workshop on Wireless Ad-hoc Networks*, 2005.

[14] 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 '07)*, Wroclaw, Poland, July 2007.

[15] I. Krontiris, T. Giannetsos, and T. Dimitriou, "LIDeA: A distributed lightweight intrusion detection architecture for sensor networks," in *Proceedings of the 4th International Conference on Security and Privacy for Communication Networks (SecureComm '08)*, Istanbul, Turkey, September 2008.