An Anomaly-based Algorithm to Detect Compromised Nodes in Wireless Sensor Networks

Mary Mathews [a], Min Song[b], Sachin Shetty[c], Frederic D. McKenzie[d]

Abstract

Wireless sensor networks are typically deployed in unattended and hostile environments which are vulnerable to compromises. An adversary can reprogram the sensors with malicious code to launch an intrusion attack. Once a sensor node has been compromised, the security of the network degrades quickly if there are not measures taken to deal with this event. In this paper, we develop an anomaly-based intrusion detection algorithm to detect compromised nodes in wireless sensor networks. The idea is to use packet arrival time as the primary parameter to differentiate the legitimate nodes and the suspicious nodes. Once the suspected node has been deemed compromised by the base station, this information is then propagated to the rest of the network. A simulator is developed to verify the algorithm design. Analysis and simulation results show that our algorithm is able to achieve high detection rate with lower false positives.

Key words: sensor network, intrusion detection, algorithm

1. Introduction

Wireless sensor networks are comprised of sensor nodes that are designed to gather information regarding environmental data such as light, temperature, sound, and pressure. Each individual node is comprised of one or more sensing devices, a processor, a communication unit, and a power supply [3, 4]. The actual sensors gather the data that represents the physical conditions being monitored once the network has been deployed. The sensor readings that are gathered periodically are sent to the processing unit that houses the data and program memory. This processing unit usually converts the data values into a human readable format if this is desired or needed. The operating system and other programs stored in the program memory dictate all operations of the sensor nodes while they are in use. While the power source of these devices normally comes in the form of a battery, there has been research conducted into other sources such as solar cells. The wireless aspect of the communication in these sensor networks is usually achieved through a radio antenna, although some sensors have instituted infrared or laser communication schemes. In many deployments, each sensor node is given a unique identification number when the sensors are programmed. This means that node identification is part of the code in the program memory and is included in all outgoing packets sent to the rest of the network.

1.1 Resource Constraints in Sensor Nodes

Sensor nodes are designed with the goals of being small in order to be utilized in different environments and relatively cheap so that many nodes can be deployed in the testing environments. This has lead to the sensor nodes having constraints in terms of low computation capability, limited memory, and short-lifetime power. Therefore, any security method added to these networks will inevitably consume some of these resources [5]. A good portion of the memory is already needed for the code that runs on the sensors that dictates to them what conditions to sense, when to sense the conditions, etc.. Additional code that is required to implement any algorithm needs to fall within the range of the provided memory minus the existing code memory. Increasing the resources on the sensor nodes is not a viable solution if the goals mentioned above or the general operating efficiency of the network are sacrificed. Since many wireless sensor networks are utilized in applications where the data gathered is confidential, security has become a critical issue. Implementing any form of security measures onto the sensor nodes requires the use of resources that are already constrained in these networks [1, 2].

1.2 General Security Issues

The objective of any security method being utilized is to maintain authentication, secrecy, and data integrity within the network [6, 7]. Authentication involves the receiver of a packet being able to validate that the alleged sender is in fact the real sender and that it is a valid node of the network. Secrecy (a.k.a. confidentiality) ensures that the data sent is not received by unintended parties. Data integrity ensures that the data received is the same as the data that was sent.

Different types of attacks on wireless sensor networks focus on exploiting the resource constraints to cripple one of the three parameters mentioned above [8]. An attacker can passively eavesdrop on the communication occurring within the network. By doing so, any of the sensitive information that is being sensed by the nodes will be available to the listening party. For a more active assault, a malicious party could inject false packets into the network that would be perceived as valid information by the other nodes [9]. This also ties up network resources that could have been used for legitimate packets. An attacker might also alter the contents of a valid packet, which undermines the authentication and data integrity of the network.

[a]Electrical & Computer Engineering Department (ECE)
Norfolk, Virginia 23529, USA
mmath014@odu.edu

[b]ECE
Norfolk, Virginia 23529, USA
msong@odu.edu

[c]ECE
Norfolk, Virginia 23529, US
sshetty@odu.edu

[d]ECE
Norfolk, Virginia 23529, USA
rdmckenz@odu.edu

Another assault that plagues wireless sensor networks is the selective forwarding attack [8]. In this attack, the adversary selectively forwards packets sent by other nodes in the network which results in lost information. For this to work, the malicious party needs to somehow include itself into the actual path of the packets being sent. If a compromised node has incorporated itself into the network and is undetected, it could easily perform this attack since the other nodes that consider it a neighbor would continue to send it packets. Information such as network updates vital to sensor network operation and packets containing sensor values would be prevented from propagating through the network correctly. This causes damage to the traffic flow of the network as well.

Most security algorithms employ some form of cryptography where data is encoded and then decoded by the base stations and sensor nodes of the wireless sensor network [10]. Cryptography allows for authentication, secrecy, and data integrity to be maintained within the network. However, the security of many of the algorithms degrades when one or more nodes have been compromised [11, 12]. This is because the adversary now has the cryptography keys that were used by a legitimate node. The rest of the network would not be able to identify the malicious node from a valid node if there are not additional security measures included in the network.

1.3 Intrusion Detection Systems

Intrusion detection systems are used to detect malicious behaviors that can compromise the security and trust of a given networking system [13]. They are generally classified into two main types: misuse intrusion detection (MID) and anomaly-based intrusion detection (AID) [14]. Both types strive for the same characteristics of a 100% attack detection rate and a 0% false positive rate. A false positive occurs when a legitimate node is identified as an intruder. For obvious reasons, this is detrimental to the integrity of the system.

The MID systems work under the concept that the attacks that plague a network exhibit certain unique characteristics that can form a signature for said attack. The individual attacks are introduced onto the network and studied in order to look for patterns with which to identify the attack. While the network is deployed, it is constantly being monitored for activity that matches any of the signatures. The problem with the MID systems is that unknown attacks can pass through undetected.

In AID systems, an assumption is made that the intruder's behavior deviates from the normal network behavior. In this type of intrusion detection systems, each sensor node will be monitoring its neighbors to keep track of the normal behavior for a given set of parameters. Any node that strays from its standard actions will trigger an alarm in its neighbors. The disadvantage of the AID systems is that there is a high false positive rate.

1.4 Challenges

In this paper we present an anomaly-based intrusion detection system that deals with the threat imposed by the selective forwarding attack on wireless sensor networks. When dealing with this type of environment, the resource constraints that are characteristic of these networks was taken into consideration. The additional code needed to implement the proposed algorithm needed to work on top of the existing code that manages all sensor operational activities. It is crucial that the additional memory space required for each node to store profiles containing acceptable behavior information regarding its neighbors be kept to a minimum. Any computations involved with determining whether a node deviates from its normal behavior was carefully weighed for usefulness and necessity. These computations are to be performed for every single incoming packet that a node receives from its neighbors. Apart from the resource limitations, there is also the wireless communication aspect to consider. The fact that the all traffic within the network uses the same communication channel implies a higher number of packets lost and a higher number of packets dropped. This affects the normal behavior profiles stored for the neighboring nodes. There are also the questions of how to efficiently determine a node as compromised and how to propagate the identification of a compromised node to the rest of the nodes in the network.

The rest of the paper is organized as follows. Section 2 presents the related work. The new algorithm to detect compromised sensor nodes is introduced in Section 3. Simulation results and analysis are provided in Section 4. Finally, Section 5 concludes the paper.

2. Related Work

2.1 Localization-based Anomaly Detection

Many wireless sensor networks utilize a GPS system to gather data regarding the location of the sensor nodes. In large sensor networks, providing each node with GPS capability might be too expensive; instead, many times beacon nodes, which have a GPS receiver, are implemented. These beacon nodes will know their own location, and the other sensor nodes use these nodes to figure out their own location in the network.

Liu *et al.* propose a technique to detect malicious beacon nodes [15]. They reason that it would be difficult for a compromised beacon node to send undetected beacon signals with the wrong location information. This is because the location and beacon signal sent by the malicious beacon node will both have to be falsified.

Beacon nodes in the network are given a set of node identification numbers and keys that allow them to communicate with the other beacon nodes of the network while appearing to be a non-beacon node. Compromised nodes are detected when a valid beacon node gets a beacon signal from a malicious beacon node whose estimated location based off the beacon signal is different from the location given by the beacon signal. Attacks using locally replayed beacon signals are discovered since it is difficult for the compromised beacon node to achieve the expected round trip time for communication between neighbors.

2.2 Neighbor Stability Based Anomaly Detection

Onat and Miri developed an intrusion detection system by exploiting certain characteristics of the sensor nodes, like their stable neighborhood information [16]. The network topology is many-to-one wherein the sensor nodes send their information to a single or fixed destination along paths that are more or less stable. Therefore, the HELLO flood packets that nodes use to identify their neighbors would not be needed throughout the lifetime of the network. It was assumed that new nodes did not appear in the network after initial deployment and that the nodes were not mobile. Also, there were thought to be no changes in transmission power levels. Every node in the network had the ability to distinctively identify its neighboring nodes. Each node used the same hardware and same algorithm stack running on it. The sensor node clocks were not synchronized with each other.

Given the stability of the network that was assumed, the sensors should know what to expect from their neighbors. To further exploit this concept, two parameters were chosen from the sensor network on which to store information regarding their neighbors. A buffer containing a predetermined number of packets is maintained in this algorithm. These packets are used to calculate the range of acceptable values of packet arrival rate and receive power for subsequent packets. If the received values for these parameters do not fall in range that is being constantly updated, an intrusion is detected. The intrusion detection system relied on the nodes to inform neighboring nodes of its findings regarding a possible intruder in the network. If a node hears these intruder messages from a set number of its neighbors, it flags the suspected node as compromised.

2.3 Software-based Attestation Detection

A different approach to detecting compromised nodes involves using code attestation to validate the actual program code running on the sensor nodes. Hardware-based methods of attestation exist where a secure coprocessor is utilized to check the memory contents of the embedded device in question. Seshadri *et al.* developed a SoftWare-based ATTestation technique (SWATT) [17].

SWATT was designed with the intention of creating a method to externally verify the code running on embedded devices. A trusted verifier is the key component in achieving the goal. The malicious node will contain at least one line of code that is different from the expected code running on legitimate sensors. The verifier has a copy of the memory contents residing in legitimate nodes. The verifier sends a "challenge" to the node, which it uses as the input to a pseudo-random generator to create random memory addresses. A checksum is performed in the device on each of these memory addresses. The verifier runs the same verification procedure locally to compute the expected value of the checksum. This expected value is compared to the value returned by the node in question.

A compromised node that has altered the memory contents would have to discern whether each memory location created by the pseudo-random generator has been changed. For SWATT to perform well enough in wireless sensor networks, the additional time needed to perform this check and run the verification procedure should be noticeable to the verifier. The experiments were conducted on a simulator contained in AVR studio version 4.0. The results showed that the difference in time to compute the checksum becomes more prominent as the number of memory locations accessed increases.

3. The Anomaly-based Algorithm

3.1 Network Model and Notations

The network model can be described as follows:

- Nodes are stationary and no new nodes are added once the network is deployed;
- Sensor node clocks are not synchronized;
- All sensor nodes (except the base station) have the same hardware and software;
- Sensors communicate with the base station in a multi-hop manner;
- The base station is a fully trusted party and is identifiable by the sensor nodes of the network

The following table lists the notation for base station.

$node\_id_a$	Identifier of suspected compromised node
$node\_id_b$	Identifier of node that sent ALERT
alert_buffer[Y]	buffer that stores the $node\_id_b$ of last *Y* ALERT messages received

The following table lists the notation for sensor nodes.

$node\_id_p$	Node identifier of

	received packet
arrival_time	Arrival time of packet based on system clock
arrival_time_buffer[*N*]	Buffer that stores information regarding packets received for *N* nodes
max_buffer_packets	Maximum number of packets to use for *arrival_time_buffer*
num_buffer_packets	Number of packets already used for *arrival_time_buffer*[*N*]
high_value	Highest acceptable packet arrival time
low_value	Lowest acceptable packet arrival time
transmission_time_buffer[*X*]	Buffer that stores last *X* transmission times
$node\_id_s$	Identifier of sensor node
$node\_id_{compr}$	Identifier of compromised nodes identified by base station
$node\_id_{valid}$	Identifier of valid node identified by base station
compromised[]	buffer that stores node ids of compromised nodes identified by base station

3.2 The Algorithm

The main idea of the new algorithm is that a compromised node has to do something different compared to the legitimate nodes of the network. The base station is alerted to the presence of a potential compromised node and uses code attestation to verify the possible threat. The following pseudocode provides a basic idea of how the algorithm works.

Pseudocode for Sensor Nodes

```
On {arrival of} packet
   If (node_id_p != 0) AND (node_id_p NOT in
   compromised[])
      If node_id_p already in arrival_time_buffer
         If num_buffer_packets < max_buffer_packets
            If arrival_time > high_value
               high_value = arrival_time
            Else If arrival_time < low_value
               low_value = arrival_time
            End If
         Else
            If (arrival_time > high_value) OR
            (arrival_time < low_value)
               Send ALERT(node_id_p, node_id_s) to
               base station
            End If
            End If
         Else
            If max N not reached
               Add node_id_p to arrival_time_buffer
            End If
         End If
   End If

On {arrival of} REQUEST_TRANSMISSION_TIMES()
   Send
     TRANSMISSION_TIMES(transmission_time_buffer)
     to base station

On {arrival of}
   COMPROMISED_NODE_FOUND(node_id_compr)
   Clear out any values in arrival_time_buffer for
      node_id_compr
   Add node_id_compr to compromised[]

On {arrival of} VALID_NODE_FOUND(node_id_valid)
   Update transmission_time_buffer values for node_id_valid
```

Pseudocode for Base Station

```
On {arrival of} ALERT(node_id_a, node_id_b)
   Send REQUESTION_TRANSMISSION_TIMES()
   to node_id_a

On {arrival of}
TRANSMISSION_TIMES(transmission_time_buffer)
   For i < size of packet_ buffer
      If transmission_time _buffer[i] !=
               transmission_time  _buffer[i + 1]
         Compromised node identified
         Send broadcast COMPROMISED_NODE
         _FOUND(node_id_a)
      End If
   End For
   If compromised node not identified
      Send VALID_NODE_FOUND(node_id_valid) to
      node_id_b
   End If
```

The algorithm designed consists of essentially two parts. The first part is the initialization phase during which the sensor nodes start communicating with their neighbors. Each time a packet comes in, the $node\_id_p$ of the packet is checked against the current entries in the *arrival_time_buffer*[*N*]. If the $node\_id_p$ is not found, an entry is created for it if the maximum number of allowed entries (*max_buffer_packets*) into the buffer has not been reached. The *arrival_time_buffer*[*N*] stores the largest *arrival_time* (*high_value*) and the smallest *arrival_time* (*low_value*) calculated for *max_buffer_packets*. Enough time was given to the nodes to get these values for *N* neighbors. Each node of the network also keeps a buffer

that contains the last *X* transmission times of the packets it sent.

During the second part, the compromised nodes are introduced into the network. A compromised node is one that performs some function that is different from those seen on legitimate nodes. In this paper, these nodes were set to perform the selective forwarding attack. The detection scheme works as follows. If the *arrival_time* of the packet node *B* sends to node *A* does not fall within the high_value and low_value that *A* has stored for *B*, node A sends an ALERT message to the base station. The base station then asks node *B* for its *transmission_time_buffer*[*X*]. If these values are not consistent, the base station labels node *B* as compromised. The base station will then send a broadcast message informing the nodes of the network to the presence of the intruder. Once a sensor node receives this message, two things happen. First, it clears out any entry it has in its packet arrival time buffer for the intruder. Second, it adds the node id of the intruder to *compromised*[], which allows the node to cease all communication with the malicious party.

However, if the base station sees that the transmission times of node *B* are consistent, it informs node *A* that node *B* is not compromised. Node *A* will then update its high or low value in the *arrival_time_buffer[N]* for node *B* accordingly. By making the sensors update these values, the number of ALERT message sent to the base station is decreased, which decreases the number of packets injected into the network by the algorithm.

It should be noticed that the base station plays an important role in the algorithm designed. It is a trusted entity by all nodes of the network, which allows the base station to verify whether a node is compromised or not and for all the sensors to automatically accept this decision. Therefore, the base station is assumed to be an uncompromised node throughout the network lifetime. In this algorithm, it would be ideal if the base station has a higher processing speed than the other nodes since it either sends or receives all the packets involved with the detection part of the algorithm.

3.3 Attack Thwarted by the Algorithm

In the selective forwarding attack that plagues wireless sensor networks, packets that should be sent by a compromised node if it was still a valid node are selectively dropped. The compromised nodes purposely eliciting malicious behavior will only perform this type of attack. They will be valid nodes of the network that are set to be compromised after a certain amount of time. The other nodes of the network that consider this node a neighbor should realize if there is a noticeable time difference between incoming packets. When this happens, they will send the ALERT message to the base station which will in turn detect the differences in the transmission time buffer of the compromised node. The selective forwarding attack is defeated when the other nodes of the network received the broadcast message sent by the base station informing them to cease communication with the malicious party.

4. Simulations and Analysis

The TinyOS simulator TOSSIM [18] was chosen to simulate the designed algorithm for detecting compromised nodes in a wireless sensor network. To begin the simulation process, there were several parameters altered for different runs to determine what kind of effect they had. The number of neighbors each node has was determined by using the following equation

$$(Z \% 10) + 1$$

where Z is the number of nodes in the network. The number of packets used for the packet arrival time buffer was experimented on, and the results showed that this particular value did not have much of an effect unless a high value was chosen. A larger number meant that the initialization phase would last much longer than intended. Therefore, 10 packets was the chosen value for each simulation regardless of the network size.

The number of values to be stored in the transmission time buffer was kept at a constant value of 5 for all simulation runs. This is because all legitimate nodes send packets every 3 seconds. The transmission time buffer for compromised nodes would not have consistent values, so a higher number was not needed. The number of compromised nodes to be introduced onto the network equaled to the network size modulo 10. All compromised were injected around the same time for each network size. The larger networks needed more time in the initialization phase which meant the larger the network, the more time before compromised nodes were introduced.

The sensor nodes are turned on at different times, meaning the time elapsed since the simulator started can be different from another node. The main reason for this is that if all the nodes are turned on at the same time, there would be constant collisions in the network. The packets would all be jammed and communication in the network would drop significantly. Also, in real applications of a wireless sensor network, the sensor nodes would need to be turned on manually and thus would not all have the same time. The simulations were run until either all compromised nodes were found or a problem was discovered.

Fig. 1 shows the average time it takes for all intruders to be found after they have been introduced into the

network. The smaller networks take less time than the larger networks for two main reasons. The first reason is that there are more compromised nodes in the larger networks, so more time is needed to detect all of them. The second reason is that there is more traffic generated in the larger networks, so it takes longer for the base station to process all the requests.

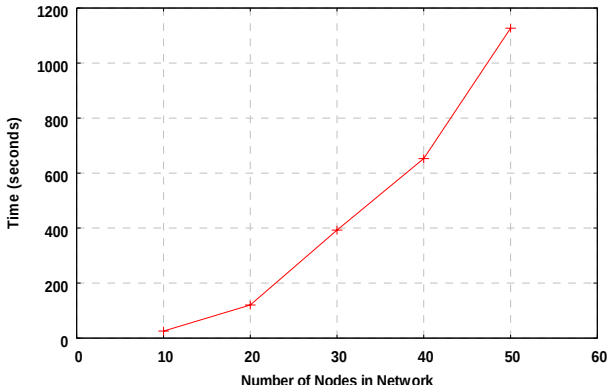

Figure 1. Average time to detect all compromised nodes

What needs to be considered when looking at all of these graphs is that 10% of the number of nodes in the network was compromised, i.e., a network of 20 nodes will have two compromised nodes. Since the base station needs to service all ALERT messages, the time needed to identify the additional nodes is higher in the larger sized networks. This attributed to the larger number of packets propagating through the network, which consequently leads to a higher number of dropped packets. The alert messages buffer kept by the base station aided in reducing the number of ALERT messages ignored. A base station with better computational resources would have helped to decrease the amount of time necessary to detect the compromised nodes. Also to keep in mind is that there is only one base station for every network size. It is expected that it would take longer to detect the compromised nodes in a larger wireless sensor network

Figure 2 shows the average number of packets sent by a compromised node once it has been injected into the network. The algorithm works efficiently since only a few packets at most are sent compared to the hundreds of valid packets. To get an idea of the ratio, the total number of packets sent in the 20-node network was 1825.67 packets, and the average number of compromised packets sent was 1.5. This means the number of packets sent by the compromised nodes made up on average 0.0822% of all traffic. Once the malicious party is found, any further communication with the compromised node is discontinued and any packets received from said party are ignored.

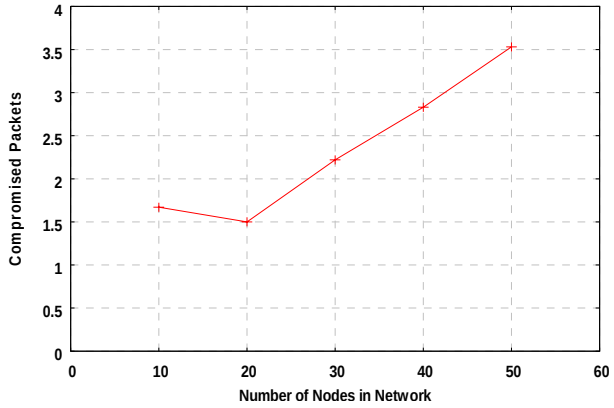

Figure 2. Average number of compromised packets sent

As mentioned earlier, false positives are often a problem when dealing with intrusion detection systems. False positives occur when a valid node is labeled as a compromised node in the network. When the number of false positives increases, the efficiency of the algorithm decreases significantly. In our algorithm, the trusted base station takes care of this problem since there were no false positives. In other words, a valid node was never identified as a compromised node by the base station. Figure 3 shows the number of false positives that were prevented by our algorithm.

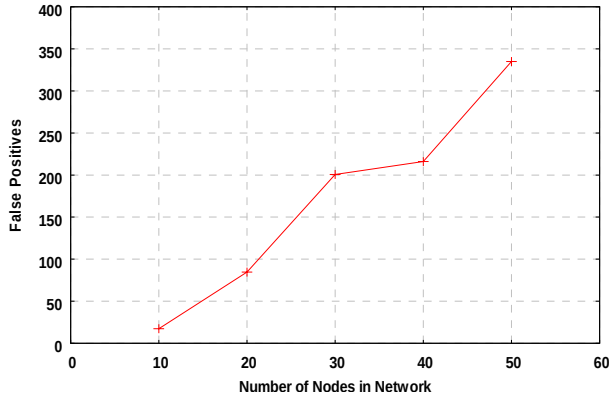

Figure 3. Average number of false positives prevented

In the next set of simulations, the network size is kept at a constant value of 20 while the percentage of compromised nodes is increased. The nodes do not detect the anomalies in the packet arrival times until a packet is actually sent. Therefore, it takes more time to detect them when there are more compromised nodes performing the same attack which is seen in Figure 4. The importance of these figures is that as the number of compromised nodes increases, the nodes and the base station are still able to work together to discover the intruders.

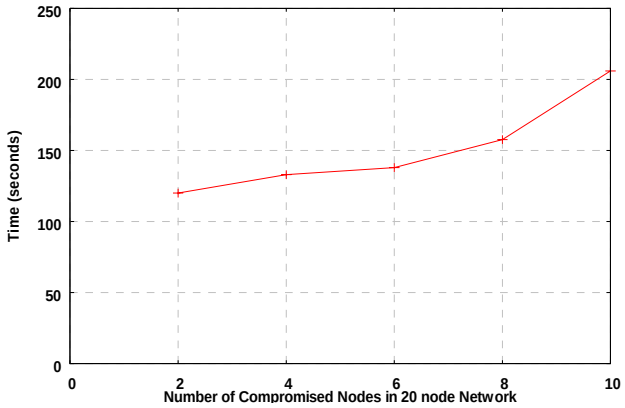

Figure 4. Average time to detect all compromised nodes

Figure 5 shows the effect of the compromised nodes in a 20-node network regarding the number of packets that are sent while the node is compromised. Examining the results shown in Figures 4 and 5 together indicates that the malicious nodes are stopped quickly before they have much time to wreak havoc in the network. Figure 5 basically implies that the average number of compromised packets being sent is the same for all nodes since this number fluctuates slightly around two packets per compromised nodes, regardless of the number of compromised nodes present in the network.

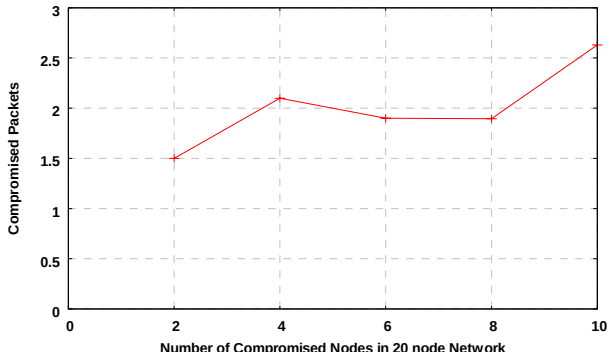

Figure 5. Average number of compromised packets sent per compromised node

Figure 6 presents the number of false positives that are prevented from occurring within the network. What is interesting to note here is that the number of false positives decreases as the network size increases. This figure can be explained by considering the fact the network size is staying the same while the number of compromised nodes performing the selective forwarding the attack increases. Since more nodes are not sending packets, this means the number of messages sent throughout the network is decreased.

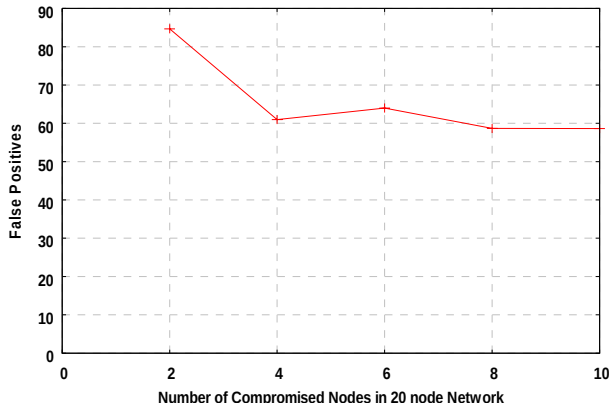

Figure 6. Average number of false positives prevented

Table 1 illustrates the overhead associated with implementing the designed algorithm in a 20-node network with two compromised nodes. The 2nd row represents the packets sent by the sensors nodes during normal operation. The 3rd row represents the communication involved with identifying compromised nodes. This includes the ALERT messages sent to the base station, the base station requesting packet transmission time buffers, the response the suspected node sends back, and the broadcast messages sent by the base station identifying compromised nodes. The packets sent by the base station indicating valid nodes are represented in the 4th row. There is a decrease in the number of packets associated with normal operation since the sensor nodes are event-driven, and it takes time to construct, process, and receive packets.

Table 1. Overhead of implementing the algorithm

Packets Sent	*Algorithm Average*
Normal sensor nodes	1382.33
Compromised nodes	358.67
Base station	84.67

The simulations we have performed mainly focus on the performance in small networks. To conduct the analytical study of our algorithm, larger network sizes are required. Next, we analyze the detection rate of our algorithm in a large sensor network.

The detection rate, which is the probability of a compromised node being detected by any of the non-compromised nodes, is an important metric used to measure the performance of the algorithm. Let us assume that a compromised node sends messages that are received by a fraction of receiving nodes p_h, which will compute the arrival time to be greater than the highest acceptable packet arrival time, and p_l, which will compute the arrival time to be less than the lowest acceptable packet arrival time. The probability of a compromised node being detected by one

node can be estimated by $p_h \times p_l$. So, if there are d nodes capable of detecting compromised nodes, the probability P_d of a compromised node being detected by a normal node can be estimated by

$$P_d = 1 - (1 - P)^d$$

where $P = p_h \times p_l$ represents the probability that a normal node receives a message from a compromised node. Figure 7 shows the relationship between the detection rate and the probability of receiving a message from a compromised node. In our algorithm, a sensor node, which alerts a base station of a compromised node when the arrival time is greater than the highest acceptable packet arrival time or lesser than the lowest acceptable packet arrival time, is a detector node.

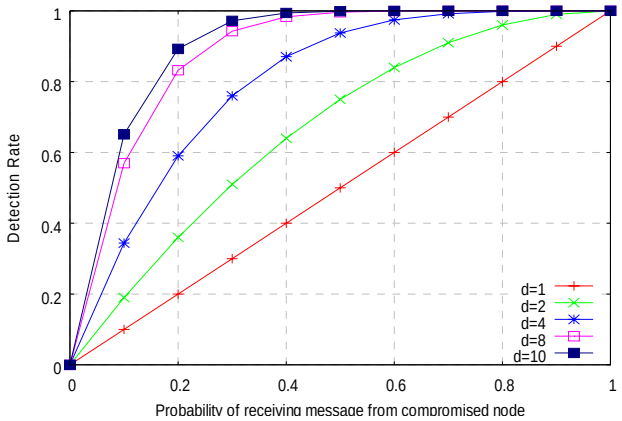

Figure 7 Detection rate vs. the probability of receiving messages from a compromised node

In Figure 7, we have plotted the values for P_d with increasing values of P and the number of detecting nodes (d). It can be seen that the detection rate of the algorithm significantly improves as an increasing number of messages from a compromised node are received by detecting nodes. This leads to more computations of arrival time by the detecting nodes and increases the number of alerts received at the base station. With more alerts generated in the network, the detection rate of the algorithm increases. We also compare the detection rate of the algorithm with the increasing number of detecting nodes. We observe that the detection rate increases at a faster rate as the number of detecting nodes increases. This indicates a compromised node cannot go undetected if the messages are received by a large number of detecting nodes.

5. Conclusions

We have presented an algorithm to detect the compromised nodes in wireless sensor networks. The algorithm is based on the anomaly-based intrusion detection technique. It uses the event-driven characteristics of sensor networks to verify whether a node is sending packets in fixed time intervals. The base station is alerted to claims of abnormal behavior and verifies them by checking the difference in packet transmission times of the suspected node. Efficiency and accuracy are two primary metrics in designing the algorithm. Simulations are conducted to demonstrate both performance metrics.

Consequently, the base station itself was where the most improvement could be made. We believe that a base station that has more computation and power resources would further decrease the detection time. If there was a method provided to distinguish between the different types of nodes, mainly the base station and sensor nodes, this would have given a more realistic idea of the capability of this algorithm where the base station is concerned. The algorithm was still able to efficiently identify compromised nodes and eliminate the occurrence of false positives. The base station and sensor nodes cooperate to deal with intruders performing the selective forwarding and propagate this information, even in the face of network problems including congestion.

References

[1] Yee Wei Law and Paul J.M. Havinga, "How to Secure a Wireless Sensor Network," *Proceedings of the 2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference*, Dec 2005, pp. 89 – 95.

[2] Adrian Perrig, John Stankovic, and David Wagner, "Security in Wireless Sensor Networks," *Communications of the ACM*, Vol. 47, No. 6, Jun 2004, pp. 53 – 57.

[3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, Mar 15, 2002, p 393-422.

[4] Marcos August0 M. Vieira, Claudionor N. Coelho Jr., Di6genes Cecilio da Silva Junior, and Jose M. da Mata, "Survey on Wireless Sensor Network Devices," *Proceedings of the ETFA '03 IEEE Conference on Emerging Technologies and Factory Automation*, vol. 1, Sep 2003, pp. 537 – 544.

[5] Sasha Slijepcevic, Miodrag Potkonjak, Vlasios Tsiatsis, Scott Zimbeck, and Mani B. Srivastava, "On Communication Security in Wireless Ad-Hoc Sensor Networks," *Proceedings on the Eleventh IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Jun 10 – 12, 2002, pp. 139 – 144.

[6] Elaine Shi and Adrian Perrig, "Designing Secure Sensor Networks," *IEEE Wireless Communications*, Vol. 11, No. 6, Dec 2004, pp. 38 – 43.

[7] Chris Karlof, Naveen Sastry, and David Wagner, "TinySec: A link layer security architecture for wireless sensor networks," *Proceedings of the Second*

International Conference on Embedded Networked Sensor Systems, 2004, pp. 162-175.

[8] Chris Karlof and David Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," *Proceedings of the First IEEE 2003 IEEE International Workshop on Sensor Network Protocols and Applications*, May 11, 2003, pp. 113 – 127.

[9] Paul Brutch and Calvin Ko, "Challenges in Intrusion Detection for Wireless Ad-hoc Networks," *Proceedings on the 2003 Symposium on Applications and the Internet Workshops*, Jan 27 – 31, 2003, pp. 363 – 373.

[10] Germano Guimaraes, Eduardo Souto, Djamel Sadok, and Judith Kelner, "Evaluation of Security Mechanisms in Wireless Sensor Networks," *Proceedings of the 2005 Systems Communications*, 2005, pp. 428 – 433.

[11] Harald Vogt, Matthias Ringwald, and Mario Strasser, "Intrusion Detection and Failure Recovery in Sensor Nodes," in Tagungsband INFORMATIK 2005, Workshop Proceedings, LNCS, Heidelberg, Germany, Sep 2005. Springer-Verlag.

[12] Carl Hartung, James Balasalle, and Richard Han, "Node Compromise in Sensor Networks: The Need for Secure Systems," Technical Report CU-CS-990-05, Department of Computer Science, University of Colorado at Boulder.

[13] Amitabh Mishra, Ketan Nadkarni, and Animesh Patcha, "Intrusion Detection in Wireless Ad Hoc Networks," *IEEE Wireless Communications*, Vol. 11, No. 1, Feb 2004, pp. 48 – 60.

[14] Yoshinori Okazaki, Izuru Sato, and Shigeki Goto, "A New Intrusion Detection Method based on Process Profiling," *Proceedings of the 2002 Symposium on Applications and the Internet 2002*, Jan 28 - Feb 1, 2002, pp. 82 – 90.

[15] Donggang Liu, Peng Ning, and Wenliang Du, "Detecting Malicious Beacon Nodes for Secure Location Discovery in Wireless Sensor Networks," Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05), pp. 609 – 619, 2005.

[16] Ilker Onat and Iand Ali Miri, "An Intrusion Detection System for Wireless Sensor Networks," *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob'2005*, Vol. 3, 2005, pp. 253 – 259.

[17] Arvind Seshadri, Adrian Perrig, Leendert van Doorn, and Pradeep Khosla, "SWATT: softWare-based attestation for embedded devices," *Proceedings of the 2004 IEEE Symposium on Security and* Privacy, May 9-12, 2004, pp. 272-282.

[18] Philip Levis, Nelson Lee, Matt Welsh, and David Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications," *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems* (*SenSys*), Nov 2003.

Mary Mathews is a PhD student in the Department of Electrical and Computer Engineering Department at Old Dominion University. She has been conducting research in the security issues of wireless sensor networks. She has studied different security applications and their efficiency and feasibility with respect to wireless sensor networks. She is the recipient of DOE GAANN Fellowship.

Min Song received his PhD in Computer Science from the University of Toledo in 2001. Dr. Song is presently an Associate Professor in the Department of Electrical and Computer Engineering at Old Dominion University. His research interests include protocols design and performance analysis of mobile ad hoc networks and wireless sensor networks, computer networks security, wireless communications, and distributed data mining. Since joining Old Dominion University in summer 2002, Dr. Song has published more than 60 international journal articles, book chapters, and conference papers, received $1.4 million federal research funding, severed as a TPC Chair, Session Chair, TPC member, and reviewer in more than 30 international conferences, and an Editor/Guest Editor of three international journals. Dr. Song is the recipient of NSF CAREER Award. He received early Tenure and Promotion in June 2007. Dr. Song is an IEEE Senior Member.

Sachin Shetty received Bachelor of Science in Computer Engineering from Mumbai University, India in 1998. He received his Master of Science in Computer Science from University of Toledo in 2002. He then earned his PhD degree from Old Dominion University in 2007. He has authored and co-authored 15 international refereed

conference publications and book chapters. His main research areas are wireless network security, sensor networks and distributed data mining.

Frederic (Rick) D. McKenzie is an Associate Professor of Electrical and Computer Engineering at Old Dominion University. Dr. McKenzie received his Ph.D. in Computer Engineering from the University of Central Florida in 1994. Prior to joining ODU, he held a senior scientist position at Science Applications International Corporation (SAIC), serving as Principal Investigator for several distributed simulation projects. At SAIC he was a Team Lead on a large distributed simulation system. Before joining SAIC, Dr. McKenzie worked as a student researcher on research projects involving both NASA Kennedy Space Center and NASA Marshall Space Flight Center. He has several years of research and development experience in the software and artificial intelligence fields, including object-oriented design and knowledge-based systems. Dr. McKenzie is an IEEE Senior Member.