DSF - A Distributed Security Framework for Heterogeneous Wireless Sensor Networks

Himali Saxena*, Chunyu Ai†, Marco Valero*, Yingshu Li*, and Raheem Beyah*

*Department of Computer Science, Georgia State University, Atlanta, Georgia 30303
†Computer Science, Troy University, Troy, Alabama 36082
Email: hsaxena1@cs.gsu.edu chunyuai@troy.edu {mvalero, yli, rbeyah}@cs.gsu.edu

***Abstract*—Wireless sensor networks (WSNs) have many applications that handle sensitive information such as surveillance, reconnaissance, and target tracking. Therefore, a WSN deployed in a hostile region should be resilient to attacks. The current approach to defending against malicious threats is to develop and deploy a specific defense mechanism for a specific attack. However, the problem with this traditional approach to defending sensor networks is that the solution for the jamming attack does not defend against other attacks (e.g., sybil, selective forwarding, and wormhole attacks). In reality, one cannot know a priori what type of attack an adversary will launch. Also, given the resource constraints of sensor nodes, the current defense mechanisms cannot be simply combined on the node to provide a complete solution. This work addresses the challenges with the traditional approach to securing sensor networks and presents a collaborative framework (the Distributed Security Framework - DSF) that can defend against all known attacks. The framework is extensible, therefore, as new attacks are discovered they can also be defended against. The DSF leverages existing defense mechanisms created by researchers. These defense mechanisms are distributed in such a way that they can, collectively, provide comprehensive defense to the network. The efficacy of the DSF is determined using simulations for scenarios consisting of multiple stationary and multiple mobile attackers. The simulation results show that though the DSF consumes more energy than single defense schemes, it can significantly enhance the network security even when the network is under multiple types of attacks.**

I. Introduction

Wireless sensor networks (WSNs) are used in areas such as health care, environmental, defense, etc. Given the importance of sensor networks in such applications, it is imperative that these networks are secured. However, the resource starved nature of sensor nodes in terms of energy, memory, and computational capability creates unique security vulnerabilities in sensor networks. Several attacks (e.g., sinkhole, wormhole, DoS, jamming, sybil, and hello flood) have been defined in the literature for WSNs. As a result of these attacks, many researchers have devised countering defense mechanisms. For example, [1] discusses the approach to defend against the hello flood attack. Additionally, the authors of [2] discuss their approach to detect node replication attacks in sensor networks. Also in [3], the authors discuss how to defend against the Sybil attack.

In general, for each of the aforementioned attacks there are specific solutions proposed by researchers. This gives the impression that, in theory, unattended WSNs should be able to be secured from attackers. Unfortunately, in reality, this is far from the truth. Although each of the aforementioned solutions has merit, they do not provide a solution for a realistic attack scenario. Each of the aforementioned techniques only defends against the specific attack(s) that it was designed to defend against. That is, to use a traditional approach to securing a WSN one must make an unrealistic assumption that the attacker will only employ the attack for which the network is prepared to defend. This approach is analogous to having an antivirus detection engine that contains one signature; operating with the assumption that the attack an adversary will launch, is the one for which the node is prepared to defend. In a real situation, one cannot know what type of attack an adversary will launch a priori. Moreover, there may be one or several attacks by an attacker or there may be several attackers at different places in the network targeting different nodes by using different attacks. Accordingly, the network must be prepared to defend against all known attacks at any given time. Additionally, the intuitive idea of combining the existing schemes will not suffice given the resource constraints of current inexpensive sensors. For example, if we consider the memory capacity of a sensor node (e.g., Mica2 mote with 4 KB RAM and 128 KB program memory), it is not possible to store mechanisms that detect and analyze many attacks, and therefore prevent the nodes/network from a security breach. For instance, [4] requires 2.2 KB RAM to defend against DoS attacks, [5] needs 1.5 KB RAM to defend against sinkhole attacks, and [1] requires approximately 1 KB RAM to defend against hello flood attacks. If we consider program memory, then 60KB is required for the operating system (e.g., TinyOS [6]), 45.26 KB to store a code dissemination tool such as [4], and 7.2KB (approx.) to provide link layer security [7], which consumes 88% of the available program memory while still leaving the node vulnerable to many attacks.

Currently, there is not a solution that can defend against all known attacks in realistic situations. As discussed in [8], although the security mechanisms are well established for each individual layer or individual attack, combining all of the mechanisms and making them work in collaboration is a hard research problem. Therefore, we propose to develop a framework that can provide this capability. Since providing defenses for all known attacks at different layers is not possible with memory constraints of low-end sensor nodes and using only high-end sensor nodes (gateway nodes) presents cost constraints, we propose to use a heterogeneous sensor network architecture where there is a combination of high-end sensor nodes along with low-end sensor nodes to define a general framework for security in sensor networks. As mentioned in [9], the use of a heterogeneous architecture has worked well to help improve routing.

In this framework, we assume that the network is divided into clusters. In each cluster, one gateway node is present and serves as the cluster head. The regular nodes collect data from the physical environment. They send this data to the gateway node in a multi-hop fashion. Gateway nodes gather the data, analyze, process, and transmit it to the base station. To provide security, gateway nodes have a large database of images (i.e., detection and defense schemes) to defend against various attacks. We assume that in the future more sophisticated attacks like buffer overflow attacks on sensors [10] are possible. Providing defense against all the known as well as future attacks requires us to have the defense mechanisms of all these attacks in memory. However, because of memory constraints, regular nodes cannot contain all defense mechanisms, so a subset of these mechanisms is stored in the local program memory of the regular nodes. Based on this subset, the regular nodes are capable of detecting and defending against some attacks. However, there may be some attacks whose defense mechanisms are not found in the regular nodes. In this situation, the gateway node detects and defends against these attacks by adding the image of the defense mechanism for the recent or most threatening attack to the previous subset. To add an image, it might be necessary to delete an image from the subset as the regular nodes have limited memory capacity. How to decide an image subset for regular nodes in the same cluster is an optimization problem. The image subset selection scheme is defined in Section IV and solved in Section V. After detecting an attack in its own cluster, a gateway node then propagates a warning to other gateway nodes. After receiving the warning information, the gateway node evaluates the likelihood of each attack and chooses a new image subset for regular nodes in its cluster.

In a cluster, an image subset of defense mechanisms is changed according to the network situations. Because of memory limitations, to defend against an attack whose mechanism is not in its program memory, a regular node needs to download the image of the defense mechanism from its cluster head (i.e., the gateway node). Thus, the gateway node propagates a program image including a countermeasure for a new attack to regular nodes. One way to do this is by using wireless network programming (remotely reprogramming sensor nodes through wireless links after they are deployed). For WSNs, wireless network programming includes three main components: code dissemination tools such as Deluge and Seluge, NetProg which provides functions for saving TinyOS state, and TOSBoot which boots the system and loads a program image [4]. Deluge [11] is a reliable code dissemination protocol for propagating large binary images to all sensor nodes in multihop wireless sensor networks. It can push about 90 bytes per second (11% of the maximum transmission speed of the radio supported by TinyOS). Moreover, each node can maintain multiple code images, and quickly switch between different programs. Code dissemination is a critical moment for wireless programming since both external attackers and potentially compromised nodes are threats in this phase. For instance, adversaries may introduce malicious code into sensor networks through replacing or modifying the code image. Seluge [4] is a secure extension of Deluge. It inherits efficiency, robustness, and reliability from Deluge. It also provides protection for code dissemination. Seluge not only promises the integrity of code images but also resistance to various DoS attacks. In this paper, we use Seluge as our code disseminator.

The main contributions of our work are summarized as follows:

1) We propose a Distributed Security Framework (DSF) which can detect and defend against all known attacks efficiently.
2) Our warning mechanism can inform other clusters to install defense mechanisms for potential attacks in advance, thus reducing the impact caused by attacks.
3) The security framework defined in this work is modular and scalable, thus defense mechanisms for new or future attacks can be easily added.

The rest of the paper is organized as follows. Section II discusses the related work in the field of sensor network security. Section III describes the network model which presents the network topology and threat model. We give the formal definition of the problem in Section IV. Section V describes the details of the proposed security framework architecture. In Section VI, we show the workings of the distributed security framework. The simulation results and analysis are presented in Section VII. We conclude the paper and discuss the future work in Section VIII.

II. Related Work

In sensor networks, sensors are left unattended for long periods of time after their deployment. Since these networks are used in many applications that handle sensitive information, security is an important issue. Security is a significant and open problem in wireless sensor networks which has been discussed in [12] and [13]. Various security requirements that should be met to protect sensor networks from adversaries have been discussed in [14].

Various attacks on sensor networks have been studied systematically in [15] and [16]. Some possible countermeasures against these attacks without implementation have been discussed in [17]. Furthermore, a variety of attacks (e.g., sybil, DoS, and wormhole), their detection mechanisms, and the possible countermeasures for these attacks have been discussed in [3], [18], and [19]. All these previous works provide solutions for only one attack in the network while we provide a solution for all known attacks at a given time in the network.

A set of secure protocols for sensor networks, SPINS, is given in [13]. It describes two important secure building blocks: SNEP and μTESLA, where SNEP gives the baseline security primitive (i.e., data confidentiality, two-party data authentication, and data freshness) whereas μTESLA is a protocol which provides authenticated broadcast in sensor networks. Although these security protocols detect and correct some classes of abnormal node behavior, they do not consider all scenarios of malicious activity that a node is susceptible to. For example, an adversary may start jamming the wireless channel and thus launch a DoS attack.

Previous research on sensor networks mainly considered homogeneous sensor networks where all sensor nodes have the same capabilities. It has been shown that homogeneous sensor networks have poor fundamental performance limits and scalability [9]. An introduction to heterogeneous sensor networks was given in [20] and the protocol requirements for heterogeneous sensor networks are discussed in [21]. To achieve better performance in terms of routing, a heterogeneous sensor network model has been proposed in [9]. Furthermore, [22] discusses how heterogeneity can be used in sensor networks to provide security. Several recent proposed WSN security techniques leverage this architecture. For example, [6] describes LIGER, a hybrid key management scheme for heterogeneous sensor networks. Furthermore, [22] presents an end-middle-end security framework where every node has a public/private key pair. In this framework, the resource-rich gateway nodes use public key cryptography to compute digital signatures and vouch for regular nodes that use symmetric cryptography. A scalable key management scheme based on random key pre-distribution for heterogeneous sensor network is discussed in [23]. Although these works illustrate the efficacy of using the heterogeneous paradigm to provide security, none provides a comprehensive approach to securing WSNs.

Our proposed technique uses the heterogeneous model to define a general distributed framework for securing WSNs. By leveraging and combining existing defense mechanisms (e.g., [3], [18], and [19]), the proposed security framework (DSF) can provide defense against all known attacks.

III. Network Model

In this section, we introduce our network topology and threat model.

A. Clustered Heterogeneous Networks

We consider heterogeneous WSNs in this work, where there are two kinds of nodes, gateway nodes and regular nodes. Regular nodes have limited energy, poor computation ability, short sensing, and small transmission ranges. Accordingly, we can deploy a large number of regular nodes in a vast area since they are inexpensive. Compared to the inexpensive regular nodes, gateway nodes have plentiful resources including more energy, a larger memory size, stronger communication ability, and more powerful computation ability. Nevertheless, a gateway node is more expensive than a regular node. As a result, fewer gateway nodes are deployed in a WSN. Usually, gateway nodes are responsible for complex computations and long distance packet delivery to increase processing capability and prolong network lifetime. An example of heterogeneous network is shown in Fig. 1.

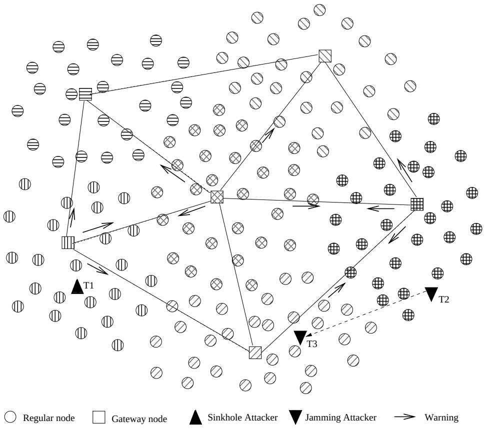


Fig. 1: DFS with various attackers.

The network is divided into clusters. The number of clusters is the number of gateway nodes, and every gateway node is the cluster head of its cluster. A regular sensor joins the nearest cluster which is located by the cluster head. In a cluster, in order to communicate within a cluster, the gateway node sends broadcast messages using a single hop to reach the regular nodes while regular nodes communicate with the gateway node over multiple hops. A gateway node communicates with other gateway nodes directly or through other gateway nodes. The gateway nodes have a database component in memory which maintains the records of various details regarding the previous threats/attacks as well as regarding the possible oncoming threats/attacks and their respective detection and defense schemes.

B. Threat Model and Assumptions

We assume that the base station and gateway nodes use tamper proof hardware, thus are trustworthy, and cannot be compromised by an adversary. On the other hand, regular nodes do not use tamper proof hardware and are always prone to attack. We assume that the attackers possess hardware capabilities either similar to or higher than that of legitimate regular nodes. We assume that if an adversary compromises a node, she can extract all key material, data, and code stored on that node. The malicious node is structurally the same as the normal nodes. It is captured by the attacker after the deployment. We assume that if the attacker is able to compromise a single node, she can launch several attacks using the information of a single node. As the whole network region is divided into clusters with the gateway node being the cluster head, the following scenarios for launching the attacks are possible:

1) There may be one attack or several attacks in a single cluster launched by one or several attackers.
2) There may be one attack or several attacks in multiple clusters launched by one or several attackers.

Moreover, the attacker launching an attack in a cluster may change his position to target other clusters. We also assume that when the attacker compromises a node in the network, all the one hop neighboring nodes which come within the transmission range of the compromised node are at higher risk of being attacked.

IV. Problem Definition

The goal of our scheme is to significantly reduce the effect of attacking/malicious nodes in the network. A formal definition of the security problem that we are addressing is given in this section.

Problem Definition: There is a set of attacks, A, where $A = \{A_1, A_2, \cdots, A_n\}$ and n is the number of attacks. For any attack A_i, there exists a defense scheme D_i. D is the set of defense schemes, and $D = \{D_1, D_2, \cdots, D_n\}$. For each defense scheme D_i, the program size is P_i. Since gateway nodes have enough program memory, all defense schemes, that is D, are stored in gateway nodes. However, for a regular node, available program memory P_R is less than $\sum_{i=1}^{n} P_i$. In other words, a regular node can only store a subset S of D. For each gateway node, how to determine S for regular nodes in its cluster is critical and significantly affects the security of the network. A *weight* W_{ji} of A_i is assigned for a gateway node G_j according to the possibility of the occurrence of the attack A_i in G_j. The larger the weight, the higher the possibility of attack A_i. Obviously, we prefer to include as many defense schemes as possible in S. Also, the defense scheme with higher weight should have priority to be chosen. Consequently, Equations 1 and 2 are a formalized description of our optimization problem. This problem is actually a 0-1 knapsack problem.

$$Maximize \sum_{D_i \in S} W_{ji} \tag{1}$$

$$Subject\ to \sum_{D_i \in S} P_i \leq P_R,\ where\ S \subset D \tag{2}$$

Since memory capacity is not a concern with gateway nodes, we assume that the gateway nodes have the detection and defense schemes available in their databases for all the known attacks. Whether the detection and defense scheme for an attack is available in the subset S of the regular sensor nodes or not, the gateway node is able to detect the attack. We assume that there are no false positives when the gateway node detects the attack. The gateway also spreads the information regarding an attack to other gateways in the network.

Since an attacker can repeat the same attack or launch a new attack at any time in the network, the likelihood of occurrence of each attack at a given time is measured at each gateway. The details of estimating this likelihood is described in the next section. Based on this likelihood, the gateway recalculates the subset S for sensor nodes in their respective clusters. As a result, an attacker can launch an attack and compromise a node or several nodes because S does not contain the detection and defense mechanism for that attack. However, the attacker cannot continue to compromise more nodes once the gateway detects it and propagates defense schemes to the regular nodes.

TABLE I: Program size of defense schemes

Attack	Program size
Sinkhole	4KB[[5]]
Hello Flood	1KB[[1]]
Jamming	5KB[approx]
Wormhole	5KB[approx]
DoS	5KB[approx]
Selective Forwarding	1KB[approx]
Sybil	4KB[approx]

V. The DSF Architecture

In this section, we propose our security framework which can defend against all known attacks efficiently.

A. Routing Protocol

Since a gateway node knows the locations of sensors in its cluster, it can calculate routes for each pair of nodes. These routes are the shortest paths based on the number of hops. The gateway node communicates with the regular nodes in its cluster using a broadcast message. The gateway node sends a routing table to each node in its cluster. Furthermore, a gateway node keeps all routing information in its cluster. Each regular node sends its current state to the gateway periodically. If the gateway does not receive the report on time from a specific node, it assumes that it has been compromised, is dead, or a link failure has occurred. Thus, the gateway node excludes this node from each routing table. New routing tables are sent to the regular nodes using route update messages. These updated routing tables contain the information regarding the destination node as well as all forwarding nodes from source to destination. In other words, these tables contain the information of the routing path that a regular node must take in order to communicate with the gateway node or other regular nodes in a cluster. For gateway level networks, Destination Sequenced Distance Vector (DSDV) [24] is used as the routing protocol.

B. Choosing the Defense Mechanism Subset

According to the problem definition in Section IV, to solve this optimization problem, we have to first determine the program size P_i for each attack A_i and the weight W_{ji} which indicates the likelihood of an A_i occurrence for a gateway node G_j. The program size determines the program memory space required to store the defense scheme. The program size of the defense mechanisms we use to defend against attacks are listed in Table I. Since we cannot find the program size for some defense mechanisms, an estimated value is provided in Table I according to the likely complexity of their algorithms.

Initially, we divide attacks in different categories based on our opinion of their security level as shown in Table II. For each attack, an *initial weight* is assigned to indicate how dangerous it is to the network. The weight for category 1 attacks is 1, for category 2 the weight is 2, and for category

TABLE II: Weight based on attack categories

Category 1	Category 2	Category 3
Selective forwarding	Sybil	Wormhole
Hello flood	Sinkhole	Jamming
-	-	DoS

3 it is 3. It is possible that for some applications category 1 attacks are more dangerous than category 2 or 3 attacks. Therefore, based on different applications the category of an attack can be changed according to requirements.

When a gateway detects an attack A_i (ID of this attack is k) in its own cluster, it sends a warning W_k to all gateway nodes as well as updates its own record regarding that attack. The warning information includes the type of attack A_i, the ID of the sender gateway G_s, and the weight of this warning WW_k (which indicates how dangerous this attack is). WW_k is equals to the product of the *initial weight* of A_i and the number of compromised nodes. In this work, the number of compromised nodes determine how powerful or dangerous the attacker is for the network. As it is more likely that the more nodes that are compromised, the more power the attacker has.

On a gateway node G_j, we maintain a received warning list L_j. Every warning W_k in L_j has the format as W_k = $\{A_i$,G_s,WW_k,$T_k\}$, where A_i represents the type of the attack, G_s is the sender of this warning, WW_k is the weight of this warning which is assigned by the sender, and T_k is the received timestamp of this warning. Once a gateway G_j receives a warning W_k = $\{A_i$,G_s,WW_k,$T_k\}$, it searches L_j first, if there is no entry for (A_i,G_S), this warning is inserted into L_j, otherwise it is replaced with this new warning. In other words, for warnings from the same gateway with the same type of attack, only one entry is maintained in L_j.

After updating the warning list, the likelihood of each attack occurring is calculated as follows. For each attack A_i in L_j, we use Formula (3) to calculate the weight W_{ji}. In (3), D is the distance function between the sender and the receiver gateway nodes. If the warning is from itself, D is substituted by the distance between the attacked or targeted node and the gateway. The weight of an attack is inversely proportional to the distance between the sending and receiving node. T_c is the current timestamp. Two characteristics of attacker behaviors are considered to calculate weights. Once an attacker intrudes at a location, it might try to extend its influence as far as possible. As a result, clusters close to that location have a high likelihood of being attacked. If an attack occured recently in one location, it might later appear in adjacent locations since several attackers might intrude in different locations in succession. Consequently, an attack detected recently has a high likelihood of being executed by other clusters. Thus, we consider the distance and time difference in our formula.

$$W_{ji} = \sum_{\forall W_k \in L_j \ and \ attack \ is \ A_i} \frac{WW_k}{D(G_s, G_j) * (T_c - T_k)} \quad (3)$$

After calculating weights, a gateway can solve the 0-1 knapsack problem described in Section IV to obtain S, the subset of defense mechanisms. Then if the new subset is not the same as the previous one which the regular nodes had already installed, the gateway sends the defense mechanism images, which are in the new subset but not in the previous subset, to all regular nodes in its cluster using Seluge [1]. After applying the corresponding defense mechanism, the attacker will no longer be able to compromise the target nodes. Our security scheme can provide a needed defense mechanism to regular nodes as soon as possible when clusters are under attack. Also, our warning system can enable other clusters to install defense mechanisms in advance to defend against potential upcoming attacks.

In the next section, we discuss the workings of the proposed security framework.

VI. Workings of the Security Framework

Now we discuss the working of the distributed security framework in detail. Fig. 1 illustrates that an attacker may be present at any location in the network. Suppose at time $T1$, $Attacker_1$ at position $(x1, y1)$ chooses to launch the sinkhole attack. We assume that all regular nodes which are within the range of the attacker are at high risk of being targeted. When a cluster is under a sinkhole attack, there are two possible cases:

1) The detection and defense scheme for the sinkhole attack is in the subset of the regular nodes (described in Section V-B).
2) The defense scheme is not present in the subset.

In case 1), the targeted nodes detect and defend against the attack and the attacker is not able to compromise nodes using the sinkhole attack. Whereas in case 2), target nodes will not be able to defend against the attack and therefore will be compromised. In both cases, the gateway detects the attack and checks the subset stored in the regular nodes. If the subset does not include the defense scheme, the gateway calculates the attack's weight and is given the highest priority as it occurs in its own cluster. It then changes the subset and propagates corresponding defense mechanism to all regular nodes except the compromised node in the cluster.

Whenever a gateway detects an attack, a warning message is sent to all gateways. It is possible that other gateways receive multiple warnings for multiple attacks from different clusters. So, each gateway calculates the occurrence possibility for each attack in its cluster (described in Section V-B). According to the possibilities and defense schemes' program sizes of all attacks, the gateway node chooses an optimal subset of defense schemes to propagate to the regular nodes in its cluster.

An example of a mobile attacker is shown in Fig. 1. At time T_2 $Attacker_2$ is present at position $(x2, y2)$ and launches a jamming attack and at time T_3 the $Attacker_2$ changes its position to $(x3, y3)$ and launches the same attack. Now suppose at time T_2, the regular nodes which are within the range of $(x2, y2)$ do not have the defense mechanism available for the jamming attack, these nodes would be susceptible to attack. However, its cluster head, the gateway node, detects the jamming attack and installs defense schemes for other

TABLE III: System Parameters and Setting.

Parameter	Setting
No. of regular nodes	2000
No. of gateway nodes	10
Network size	1000 m *1000 m
Transmission range of regular node	50 m
Transmission range of gateway node	500 m
Initial energy of regular node	1 J
Energy cost for sending a message by regular node	10 uJ
Energy cost for receiving a message by regular node	1 uJ

regular nodes in this cluster. Furthermore, a warning regarding this jamming attack is sent to other gateways. At time T_3, when $Attacker2$ changes its position to $(x3, y3)$ and tries the jamming attack again, the regular nodes already have the defense scheme available for this jamming attack. Thus, the adversary will be unsuccessful since the gateway receives the warning and installs the jamming defense scheme in advance. Also, once a gateway node detects an attack, it removes compromised nodes from every route.

As mentioned earlier, the database component of a gateway stores the detection/defense schemes for all known attacks. Moreover, if a gateway detects abnormal behavior, possibly indicating an unknown attack, it will report all information to the base station. When the base station has a defense mechanism for this new attack, it will be sent back to that gateway. After the gateway verifies the defense scheme, it is disseminated to all other gateways and is stored in the gateway's database. Therefore, our distributed security framework is extensible and can accommodate defense mechanisms for new attacks.

VII. Performance Analysis

We randomly deploy 2000 regular nodes and 10 gateways in an 1000m*1000m network. Every regular node has the same initial energy. Since a gateway has a rich supply of energy, we do not consider the energy consumption for gateway nodes. On the other hand, regular nodes lose energy when they defend against an attack, communicate with other regular nodes or communicate with the gateway node. Both static attackers and mobile attackers are considered in our simulation. Different system parameters and their settings used in the simulation are given in Table III.

Two metrics are used to evaluate the performance of our distributed security framework (DSF):

1) Success Rate: This is defined as the percentage of nodes alive after the attacks. This parameter is used to evaluate the efficiency of the security schemes.
2) Energy Consumption: This is defined as the average percentage of residual energy (compared to the initial energy) for all currently alive regular nodes.

We compare our scheme (DSF) with One Security Scheme (OSS) and Multiple Security Schemes (MSS), where OSS schemes provide defense against one fixed attack and MSS provides defense against three fixed attacks. OSS-SF, OSS-WH, and OSS-JAM provide defense for selective forwarding, wormhole, and jamming attacks respectively as presented in [26], [27], and [28]. We assume that no node will be compromised when respective defense schemes for the attacks are available. MSS provides defense for all three of these attacks simultaneously. Although, MSS does not exist in the literature, we thought it would be interesting to see how such a scheme would perform.

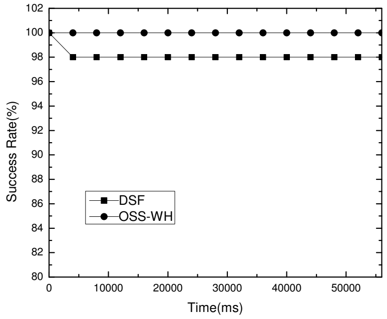
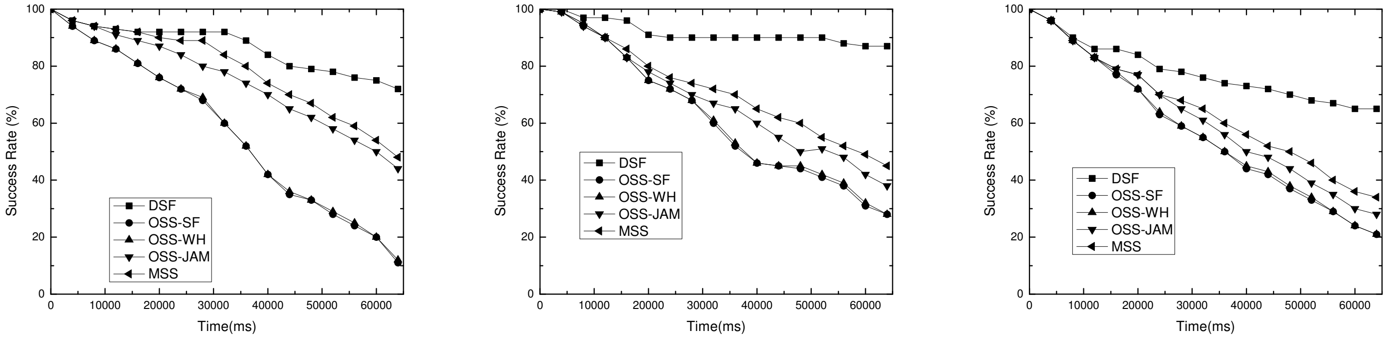


Fig. 2: Comparison between DSF and OSS-WH in case of only one attack.

Fig. 2 shows the drawback of the security framework compared to an example of a single attack and defense scheme (OSS-WH). In this scenario, 200 wormhole attacks are injected, and initially the DSF does not install the wormhole defense scheme. As shown in Fig. 2, OSS-WH outperforms DSF as no node (representing OSS-WH) is compromised since the defense scheme for the wormhole attack is already available to all the nodes. The DSF loses 2% of the nodes because there is no wormhole defense scheme initially installed. However, after a short period there is no further loss of nodes since once the gateway detects the wormhole attack, the wormhole defense scheme is immediately installed by all regular nodes. Thereafter, the upcoming wormhole attacks cannot attack the network successfully.

The success rate of three attacking scenarios are shown in Figs. 3a, 3b, and 3c respectively. Fig. 3a shows the scenario where there are only stationary attackers launching 200 attacks randomly which alternate between the 7 attacks given in Table I. Here the DSF performs 6.5 times better than OSS-SF, 6 times better than OSS-WH, 1.6 times better than OSS-JAM, and 1.5 times better than MSS. In Fig. 3b, 10 attackers are mobile in the network with the speed of 10 m/s injecting various attacks randomly in the network. In this scenario, the DSF performs 3 times better than OSS-SF and OSS-WH, 2.1 times better than OSS-JAM and 2 times better than the MSS scheme. In Fig. 3c, both static and mobile attackers are loosely scattered in the network. Here the DSF performs 3 times better than OSS-SF, and OSS-WH. In the same scenario, the DSF performs 2.3 times better than OSS-JAM and 2 times better than the MSS. As seen in the results, in all three scenarios, the DSF significantly outperforms OSS and MSS since the DSF can adjust and apply corresponding defense schemes according to occurring attacks. Moreover, MSS is better than

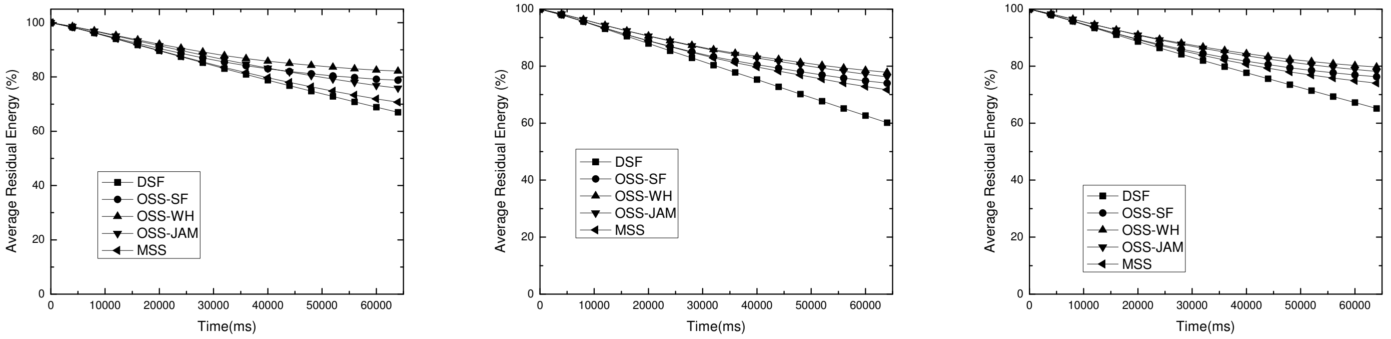


(a) Success Rate: Static Attackers.

(b) Success Rate: Mobile Attackers.

(c) Success Rate: Static Attackers and Mobile Attackers.

Fig. 3: Success Rate of Different Scenarios.



(a) Energy Consumption: Static Attackers.

(b) Energy Consumption: Mobile Attackers.

(c) Energy Consumption: Static Attackers and Mobile Attackers.

Fig. 4: Energy Consumption of Different Scenarios.

OSS because MSS has two more defense schemes installed than OSS.

The average residual energy percentages of the three scenarios are shown in Figs. 4a, 4b, and 4c respectively. The statistics show that in all the three scenarios OSS-SF, OSS-WH, and OSS-JAM schemes have approximately 20% more average residual energy than the DSF, while MSS scheme has 10% more average residual energy than the DSF. As seen from the results, the DSF consumes more energy than OSS and MSS. The main reason for this is that the regular nodes consume energy when receiving the defense scheme's program images. However, it is worth losing some energy to keep more nodes alive.

Fig. 5 shows the effect of speed of mobile attackers on the success rate. The figure represents a single network with mobile attackers having different speeds. The attacker with maximum speed is able to comprise the most nodes. The reason is that if an attacker moves fast, it has a higher likelihood to arrive at another cluster and launch attacks before that gateway receives a warning regarding this attacker and installs the corresponding defense schemes. Therefore, the speed of the attacker is inversely proportional to the success rate in the security framework.

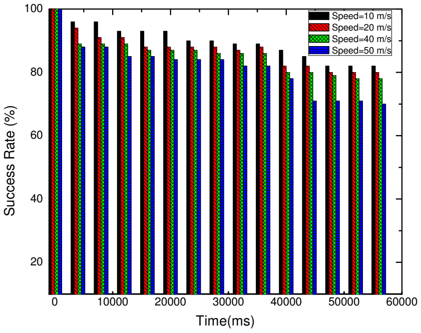


Fig. 5: Effect of Speed of Mobile Attackers.

In summary, the DSF can achieve a higher success rate than other schemes. Although the DSF consumes more energy, it might be considered is acceptable since it can keep more nodes alive to prolong the entire network lifetime.

VIII. Conclusions and Future Work

In this work, we presented a distributed security framework (DSF) for heterogeneous wireless sensor networks. In this framework, we dynamically use the available memory space of regular nodes to store a subset of defense schemes to provide security against multiple attacks. The gateway is responsible for updating this subset according to the current likelihood of the occurrence of an attack in its cluster. Our warning scheme can enable the regular nodes to install the defense schemes in advance of potential forthcoming attacks. Simulation results have confirmed that the DSF performs well in the presence of static as well as mobile attackers, each with multiple types of attacks.

In our future work, we plan to improve the success rate by determining the optimal subset of installed defense mechanisms for individual sensor node instead of every cluster. To improve our scheme, we will consider how to deal with an exceptionally powerful attacker who can compromise a gateway node too. We plan to address the case where a gateway node generates false positives and false negatives when detecting an attack. Moreover, for evaluating the efficiency and effectiveness of the DSF in real world setting, we plan to implement the framework on real sensor motes and verify its attack resistance in the presence of various attacks. Finally, we plan to consider thrashing attacks of the DFS where the attacker(s) deliberately alternate attacks to drain the energy of the system.

References

[1] M. A. Hamid, M. Mamun-Or-Rashid, and C. S. Hong, "Routing security in sensor network: Hello flood attack and defense," in *Proceedings of 1st International Conference on Next-Generation Wireless Systems*, January 2006, pp. 52–56.

[2] B. Parno, A. Perrig, and V. Gligor, "Distributed detection of node replication attacks in sensor networks," in *Proceedings of the 2005 IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, May 2005, pp. 49–63.

[3] J. Newsome, E. Shi, D. Song, and A. Perrig, "The sybil attack in sensor networks: analysis and defenses," in *Proceeding of Information Processing in Sensor Networks*, April 2004, pp. 259–268.

[4] H. Sangwon, N. Peng, L. An, and D. Wenliang, "Seluge: Secure and dos-resistant code dissemination in wireless sensor networks," in *Proceedings of the 7th International conference on information Processing in Sensor Networks*. Washington, DC, USA: IEEE Computer Society, April 2008, pp. 44–456.

[5] I. Krontiris, T. Dimitriou, T. Giannetsos, and M. Mpasoukos, "Intrusion detection of sinkhole attacks in wireless sensor networks," in *Proceedings of International Workshop on Algorithmic Aspects of Wireless Sensor Networks*. Springer Berlin / Heidelberg, July 2007, pp. 150–161.

[6] P. Traynor, R. Kumar, H. B. Saad, G. Cao, and T. L. Porta, "Liger: implementing efficient hybrid security mechanisms for heterogeneous sensor networks," in *Proceedings of the 4th international conference on Mobile systems, applications and services*. New York, NY, USA: ACM, 2006, pp. 15–27.

[7] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: Link layer security architecture for wireless sensor networks," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, 2004, pp. 162–175.

[8] A. Pathan, H.-W. Lee, and C. S. Hong, "Security in wireless sensor networks: issues and challenges," in *Proceedings of the 8th International Conference Advanced Communication Technology*, vol. 2, February 2006, pp. 1043–1048.

[9] X. Du, M. Guizani, Y. Xiao, and H.-H. Chen, "Two tier secure routing protocol for heterogeneous sensor networks," *IEEE Transactions on Wireless Communications*, vol. 6, no. 9, pp. 3395–3401, September 2007.

[10] A. Francillon and C. Castelluccia, "Code injection attacks on harvard-architecture devices," in *Proceedings of the 15th ACM conference on Computer and Communications Security*. New York, NY, USA: ACM, 2008, pp. 15–26.

[11] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2004, pp. 81–94.

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

[13] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "Spins: security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[14] S. Avancha, J. Undercoffer, A. Joshi, and J. Pinkston, "Security for wireless sensor networks," *Wireless sensor networks*, pp. 253–275, 2004.

[15] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," in *Proceedings of the First IEEE Sensor Network Protocols and Applications*, May 2003, pp. 113–127.

[16] A. D. Wood, L. Fang, J. A. Stankovic, and T. He, "Sigf: a family of configurable, secure routing protocols for wireless sensor networks," in *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*. New York, NY, USA: ACM Press, 2006, pp. 35–48.

[17] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for ad-hoc wireless networks." Springer-Verlag, 1999, pp. 172–194.

[18] J. Deng, R. Han, and S. Mishra, "Defending against path-based dos attacks in wireless sensor networks," in *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*. New York, NY, USA: ACM, 2005, pp. 89–96.

[19] W. Wang and B. Bhargava, "Visualization of wormholes in sensor networks," in *Proceedings of the 3rd ACM workshop on Wireless security*. New York, NY, USA: ACM, 2004, pp. 51–60.

[20] L. Yu, N. Wang, W. Zhang, and C. Zheng, "Deploying a heterogeneous wireless sensor network," in *Proceeding of Wireless Communications, Networking and Mobile Computing*, September 2007, pp. 2588–2591.

[21] X. Du and F. Lin, "Designing efficient routing protocol for heterogeneous sensor networks," in *Proceeding of IEEE International Performance, Computing, and Communications Conference*, April 2005, pp. 51–58.

[22] J. Mache, C.-Y. Wan, and M. Yarvis, "Exploiting heterogeneity for sensor network security," in *Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, June 2008, pp. 591–593.

[23] F. Kausar, S. Hussain, L. T. Yang, and A. Masood, "Scalable and efficient key management for heterogeneous sensor networks," *The Journal of Supercomputing*, vol. 45, no. 1, pp. 44–65, 2008.

[24] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications*. New York, NY, USA: ACM, 1994, pp. 234–244.

[25] E. Ngai, J. Liu, and M. Lyu, "On the intruder detection for sinkhole attack in wireless sensor networks," in *Proceedings of the IEEE International Conference on Communications*, vol. 8, June 2006, pp. 3383–3389.

[26] H.-M. Sun, C.-M. Chen, and Y.-C. Hsiao, "An efficient countermeasure to the selective forwarding attack in wireless sensor networks," in *Proceedings of the 2007 IEEE Region 10 Conference*, November 2007, pp. 1–4.

[27] J.-H. L. Ji-Hoon Yun, Il-Hwan Kim and S.-W. Seo, "Wodem: Wormhole attack defense mechanism in wireless sensor networks," *Ubiquitous Convergence Technology*, vol. 4412/2007, pp. 200–209, 2007.

[28] M. Rajani and O. L. Ann, "Jamming attack detection and countermeasures in wireless sensor network using ant system," in *Proceedings of SPIE, the International Society for Optical Engineering*, vol. 6248, 2004, pp. 62 480G.1–62 480G.12.