ABSTRACT

Title of Document: NODE REPLICATION DETECTION IN SENSOR NETWORKS

Hulya Resul, Master of Science, 2005

Directed By: Professor V. Gligor, Department of Electrical and Computer Engineering

Sensor nodes are small devices with limited capabilities. They are used for a variety of applications and can be deployed in public environments, which makes them vulnerable to physical attacks. An adversary can capture a node, read its cryptographic information, replicate these data onto multiple sensor nodes and insert them in the network.

In this thesis, we analyze two methods to detect node replicas in a sensor network, namely the randomized multicast and line-selected multicast. Randomized multicast distributes location information to a randomly chosen set of witness nodes, while line-selected multicast uses the routing topology of the network to detect collisions. We derived an analytical solution and conducted simulations to verify our

predictions. The results were at par with our expectations. Both methods detect node replication with high probability, but the line-selected multicast algorithm is more efficient in terms of communication costs.

NODE REPLICATION DETECTION IN SENSOR NETWORKS

By

Hulya Resul

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2005

Advisory Committee:

Professor Virgil Gligor, Chair

Professor Yavuz Oruc

Professor Charles B. Silio

Acknowledgements

I would like to thank my advisor, Professor Virgil Gligor for giving me a great opportunity to work on an interesting and challenging project, as well as for his guidance and advice.

I thank Professor Charles B. Silio and Professor Yavuz Oruc for agreeing to serve on my thesis committee.

I would also like to thank my parents for their many sacrifices, love and support all along. Sincere thanks to my office colleagues for their advice and valuable comments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Sensors are inexpensive, low-power devices, which have limited resources. They are small and have wireless communication capabilities for short distances. They can be deployed on the ground, in the air, under water, on bodies, in vehicles and inside buildings.

The capabilities of the sensor nodes for sensor networks range from those of Smart Dust sensors that have only 8 Kb of program, 512 bytes for data memory, processors with 32 8-bit general registers that run at 4 MHz and 3 V, to sensors that are over an order of magnitude more capable in processing speed and memory capacity.

A sensor node consists of the following basic components: a sensing unit, a processing unit, a transceiver unit and a power unit. Additionally they may also

have the following application-dependent components, such as: location finding system, power generator and mobilizer. The sensing unit is composed of the sensors and analog to digital converters, which convert the analog signals produced by the sensors and feed them to the processing unit. The transceiver connects the node to the network. The location finding system is used by the routing protocols and the sensing tasks to get high accuracy information about the location of the nodes. A mobilizer is used for moving the sensor node.

A sensor network consists of a large number of sensor nodes that are deployed without prior knowledge of their exact location. This allows random deployment in inaccessible environments and, at the same time, it implies that the network protocols should have self-organizing capabilities. Another feature of sensor networks is the cooperation between the nodes. They use their processing unit to perform simple computations and transmit only the required and partially processed data, instead of the raw data. These features enable the sensor networks to perform different tasks. They can be used for a variety of applications:

- *Traffic monitoring* – the sensors can gather travel speeds, lane occupancy, and vehicle counts.
- *Structure monitoring* – sensor nodes can be installed along bridges, highways and buildings for monitoring vibration and displacement in these structures.
- *Environment monitoring* – sensor nodes can be used for flood detection, fire detection, level of pollution, etc.

- *Military monitoring* – sensor nodes can be used for monitoring military perimeters

Most of the hardware for sensor networks is based on RF circuit design. Another possible mode of communication between nodes is by using infrared communication. The infrared transceivers are cheaper and much easier to build, and the infrared communication is license free. Optical communication has been implemented in Smart Dust motes, a self powered sensing and computing communication system. The disadvantage with the infrared and optical communication modes is that they require a line of sight between the sender and the receiver.

Wireless sensor networks architectures are organized in hierarchical and distributed structures. In the hierarchical model, there is a hierarchy among the nodes based on their attributes and capabilities: base stations, cluster heads, and sensor nodes. A base station collects sensor readings, performs costly operations, and manages the network. In addition, a base station can be viewed as a gateway to another network or as an access point for human interface. Cluster heads (usually nodes with better resources) are used to collect and merge local traffic and send it to the base stations. Transmission power of the base station is usually enough to reach all nodes but sensor nodes depend on the intermediate nodes to reach the base station.

In a distributed sensor network there is no fixed infrastructure and the network topology is not know prior to deployment. Sensor nodes are deployed

randomly in the targeted area. After deployment, each node uses its communication range coverage to find its neighbors and each sensor sends the data directly to the sink. Distributed sensor networks are dynamic in the sense that they allow insertion and deletion of sensor nodes after deployment to grow the network or replace non-functional and unreliable nodes.

Communication in wireless sensor networks is done in an ad-hoc manner, very similar to wireless ad-hoc networks. Likewise, wireless sensor networks have dynamic topologies because the communication range and the network connectivity changes by time. Some sensor nodes die and others join the network. However, sensor networks are more constrained, and much denser than the wireless ad-hoc networks.

Because the sensor nodes are often deployed in accessible environments they are exposed to physical attacks, so it is necessary to have security build in the design. Sensitive information is delivered in the network and must be protected from disclosure to unauthorized third parties. Resource limitations of the sensor nodes, lack of fixed infrastructure, unknown network topology prior to deployment, high risk of physical attacks, make security in wireless sensor networks challenging.

Confidentiality, integrity and authentication services are critical in preventing an adversary from compromising the security of a distributed sensor network. Deployed sensor nodes must accept only legitimate queries, commands and software updates. Since sensor networks are based on wireless

communications, attackers can listen on the radio transmissions, inject bits in the channel or replay captured packages.

An additional issue that sensor networks are facing is resistance against node capture attacks. Because the sensors are deployed in accessible areas, it is possible that an adversary captures sensor nodes, extracts their cryptographic information, modifies their programming and replaces them with malicious nodes that are under his control. Moreover, due to cost considerations it is impractical to use shielding for the sensor nodes, which would detect pressure, temperature and voltage changes that an adversary could use to access a sensor's internal state. This makes it easy for an attacker to capture, replicate and insert duplicated nodes in the networks. If the adversary captures at least one node then he can replicate it as many time he wants, thus making the network vulnerable. After replicating a node, the adversary can send false information and suppress legitimate data. Moreover, if the replicas are placed in judiciously chosen locations they can revoke the legitimate nodes and disconnect the network.

A Distributed Node-Replica Detection Scheme

Parno, Perrig and Gligor [21] propose two emergent algorithms to detect node replication in distributed sensor networks. A node replication attack is an attempt by the adversary to insert one or more nodes in the network that have the same ID as another node in the network. Their approach avoids centralized monitoring and

revokes the replicated nodes, so that non-faulty nodes in the network do not communicate with the faulty ones. The first algorithm, Randomized Multicast, sends location claims to a randomly selected set of witnesses in the network. The second algorithm, Line Selected Multicast, selects the witnesses by exploiting the routing topology. The algorithms try to minimize the power consumption by limiting communication, but still operate within the limited memory capacity that is characteristic for sensor nodes.

Previous approaches for detecting node replication use centralized monitoring, which requires that all nodes in the network transfer a list of their neighbors' claimed location to a central base station that can examine the lists for conflicts. The disadvantage of these schemes is that the base station represents a single point of failure. If an attacker compromises the base station or interferes with its communication then it will fail.

Parno *et al.* present several techniques based on loose time synchronizations that reduce the storage requirements of the protocols, point out some issues of the public key cryptography, and discuss symmetric alternatives, which would require less computational overhead at the price of additional communication.

The emergent algorithms of Parno *et al.*, which are explained in detail in Chapter 3, represent a new approach to the security of sensor networks and can be applied to other types of networks as well, i.e. peer-to-peer networks, ad-hoc

networks, in which nodes can be compromised, replicated and re-inserted by an attacker.

Contribution of the Thesis

In this thesis, we derived an analytical solution to determine the probability of detecting a node replica in the Line-Selected Multicast algorithm and conducted simulations to verify our predictions. The simulations consisted of tests on different network topologies. For each topology, we chose a legitimate node, inserted multiple replicas in the network, and computed the probability of detecting a node-replica. The results confirmed our analytical solution in the following areas:

1. Variations in the number of witnesses affects the probability of detecting a node-replica; e.g., using a small number of witnesses the chances to detect a node-replica are low. As the number of witnesses grows, we observe that the probability of a replica detection increases exponentially.
2. The density of the network does not affect the probability of detecting a node-replica, but the size of the network and the routing protocol have a significant contribution.
3. Compared to the randomized multicast case, the line-selected multicast achieves lower communication costs. In the worst case, the line-selected

multicast may have the same performance as the randomized multicast algorithm.

Thesis outline

Chapter 2 presents related work and analyzes the shortcomings of prior methods. In Chapter 3 we derive analytical solutions for determining the probability of node-replica detection in the randomized multicast and line-selected multicast algorithms. Chapter 4 discusses the results of the experiments that were carried out to verify the analytical solution. Chapter 5 concludes the thesis and presents future work that needs to be done in this arena.

Chapter 2

Related Work

Previous protocols for detecting node replication in distributed sensor networks have relied on a trusted base station to find any conflicting location claims or on neighborhood voting systems, which fail to detect distributed replications. Centralized schemes require that all nodes in the network send a list with their neighbors' location claims to the base station, which will examine the lists for conflicts.

Eschenauer and Gligor [11] propose a centralized node revocation method in sensor networks by using a simple key pre-distribution scheme. Their key management scheme relies on probabilistic key sharing among the nodes of a random graph and uses simple shared-key discovery for key distribution, revocation and node re-keying. A random set of keys is selected from the key

space. Each sensor node receives prior to deployment a random subset of keys from the key pool and any two nodes able to find one common key within their subsets can use that key as their shared secret to initiate communication.

When a sensor node is compromised, it is required to revoke the entire key ring of that node. A base station broadcasts a revocation message containing a signed list of *k* key identifiers for the compromised key ring to all the nodes in the network. After receiving this message, the sensor nodes in the network locate the identifiers in its key ring and remove the corresponding keys.

Eschenauer and Gligor also discuss the resiliency to sensor node capture. Usually tamper detection technologies are used to shield sensor nodes in the sense that any physical manipulation of the node will cause the erasure of its key ring and the disabling of its operation. Although they assume node shielding, the authors also prove that their key distribution scheme is more robust than the previous schemes (single mission key scheme and pair wise private key sharing scheme) in the case of physical attacks against captured unshielded sensor nodes.

In the single mission key scheme, if a node is captured then all communication links are compromised and in the pair-wise private key sharing all *n-1* links of the captured node are compromised. In Eschenauer and Gligor's scheme only $k<<n$ keys of a single ring are obtained, which leaves the adversary with a probability of approximately *k/P* (where P is the number of keys in the pool) to attack successfully any link in the network. Their simulations show that

the compromise of one key leads to the compromise of another link with probability of 0.3 and of two other links with probability 0.1.

Eschenauer and Gligor's approach is scalable and flexible, given the sensor-node computation and communication limitations. Trades-offs can be made between sensor-memory cost and connectivity, and design parameters can be fit in the operational requirements of a particular environment.

Chan, Perrig and Song [5] try to eliminate some of the disadvantages associated with the base station and propose the *random-pairwise keys scheme* in which any node can revoke the identity of the neighboring nodes. This method enables node-to-node authentication between neighbors and node revocation based on voting without involving a base station. Thus, the neighboring nodes can broadcast public votes against a detected misbehaving node. For example, if any node *X* observes more than a number of *t* public votes against some node *Y*, then *X* cuts all communication with *Y*. The base station, which listens to the network, can send these votes to a secure location where undeployed nodes are stored. There any yet undeployed node erases the pairwise keys associated with *Y* from the undeployed nodes' key rings, thus permanently removing node *Y* from the network.

The public voting implies that each public vote should be propagated across the network to the voting members, i.e. the set of nodes that can vote against the detected misbehaving node, *Y.* Having each node re-broadcast all votes heard on the network can lead to a denial of service attack. To avoid this

type of attack, only the voting members will re-broadcast any received public votes to each other, while all the other nodes will ignore the broadcast.

The Sybil attack

The Sybil attack is related to the node replication attack. In the Sybil attack, a malicious node behaves as if it were a large number of nodes, for example by impersonating other nodes or simply by claiming false identities. An adversary may also generate a number of additional node identities using only one physical device.

The Sybil attack was first described by Doucer in peer-to-peer networks [10]. Doucer proposed resource testing as a countermeasure against the Sybil attack. The resources used for this purpose are computation, storage and communication. The first two resources are not applicable to sensor networks, because it is possible that the adversary will use a much more powerful physical device than the sensor node. Communication testing implies that a request for identities is broadcasted in the network and only the identities of those nodes that reply within a given time interval are accepted. The disadvantage of this testing method is that it causes congestion in the part of network where the verifier resides because all the replies will converge to it. Karlof and Wagner [15] noted also that the Sybil attack can be a threat to the routing mechanism in sensor networks.

Newsome et al. [10] analyze the threat posed by the Sybil attack in wireless sensor networks and classify different types of attacks, and the countermeasures against each of them. The authors explain how an adversary can make use of these different types of attacks to compromise several sensor network protocols. Their solutions include radio resource testing, key validation for random key predistribution, position verification and registration.

Among these solutions, only the centralized node registration technique can detect a node replication attack. Such an attack is referred to as an *identity replication attack*, when the same identity exists in multiple places in the network. Identities are registered at a central trusted location, which manages the network, knows the deployment of the nodes and is able to securely disseminate the information to the network. In order to detect a possible Sybil attack, a node can poll the network and see if the actual deployment is similar to the one known. To validate a node as legitimate, a node can check a list of known good identities, thus being able to prevent a Sybil attack.

The registration method proves to be a good solution in many scenarios, but it has some disadvantages as well. The list of good identities must be protected from unauthorized modification. If an adversary successfully compromises the list, he can insert identities to the list, thus adding Sybil nodes to the network. Moreover, the initial deployment information, with which the current deployment is compared to, must be protected and secured by the central location

that manages the network. In conclusion, the registration method is weak in the face of node compromise and has a high overhead.

An algorithm for counting the number of nodes in a peer-to-peer network is presented by Bawa *et al.* [2]. They use random samples to compute a statistic based on which it is possible to estimate the size of the network. Using the birthday paradox, after $O(\sqrt{n})$ samples are drawn and a duplicate node is sampled, the estimate of the network size can be computed.

The birthday paradox, which is used in the randomized multicast protocol states that for $\sqrt{n}$ independent samples from a set of size *n*, the probability that a pair of samples will have the same value is 0.5.

Centralized algorithms are conceptually simple but they have several disadvantages. First, the base station represents a single point of failure. Compromising the central station or the communication channels around it will lead to the failure of the network. Moreover, the nodes located close to the base station will receive a large number of packets, thus becoming a target for the adversary. Another drawback of centralized detection is the delay in the protocol because the base station must wait until it receives all reports, then analyze them for conflicts and finally flood the revocations in the network. Using a distributed algorithm will make the revocation of replicated nodes much faster.

From a security point of view, if all messages reach the base station successfully, the centralized protocol will detect all replicated nodes with

probability one. From a performance point of view, assuming that the average path length from a node to the base station is $O(\sqrt{n})$ and the average number of neighbors for each node is $d (d << n)$, then the protocol will require $O(n\sqrt{n})$ communication for the location claims from each node to reach the base station.

Chapter 3

Algorithms for node replication detection

3.1 Background

Existing protocols for detecting node replication have relied either on a trusted base station to find any replicated nodes inserted by a malicious party or on localized voting mechanisms. Neither solution is able to detect distributed node replication. In this chapter, we analyze in some detail two detection algorithms, namely randomized multicast and line selected multicast, which overcome the drawbacks of the previously discussed methods, and allow revocation of the subverted node and its replicas. We evaluate each protocol's security by examining the probability of detecting a replicated node given that an

adversary has inserted a number of replicas in the network. In addition, we will evaluate the efficiency of each algorithm from communication and storage points of view. Communication bandwidth is extremely important since each bit transmitted consumes about as much power as executing 800-1000 instructions [15] thus, any message expansion caused by security mechanisms raises the cost significantly. Minimizing the storage is also important since sensor nodes have limited amount of memory, often in the order of a few kilobytes.

In analyzing the security of the sensor network, we make the following assumptions. We consider the sensor nodes being unshielded, which allows an adversary to read their cryptographic information from memory. Afterwards he can create replicas by loading this information on different sensor nodes. Since the sensor networks are deployed randomly, the clones can easily be inserted into different locations in the network.

The two algorithms proposed by Parno *et al*. are based on a simple broadcast scheme in which each sensor node uses an authenticated message to flood the network with its geographical coordinates. If it is assumed that all messages reach the destination safely, (this can be assured by using acknowledgement messages) then the scheme guarantees 100% detection of replicated nodes. However, it has a serious drawback and that is the communication cost. Since there are *n* nodes in the network and each of them has to send its location claim to the *n-1* remaining nodes, the order of communication cost will be $O(n^2)$. This value is not a problem when the size of the network is

relatively small, but as *n* grows, the communication becomes too costly. If the schema is modified such that node's location claim is not send to all *n-1* nodes in the network and just to a limited set of nodes, called *witnesses*, then the communication cost is reduced.

The witnesses are chosen as a function of the node's ID. If a replica exists somewhere in the network, then the witnesses will receive two conflicting location claims for the same node. If the size of the set of witnesses is *g* and the degree of each node *d,* then if each of the neighbors randomly picks $\frac{g \ln g}{d}$ the coupon collector's problem [8] assures that each of the witnesses will receive at least one location claim. This scheme although it reduces the cost of communication has a drawback. Since the function for computing the witnesses is deterministic, an adversary can predict them as well. If he is able to capture or jam all of the messages for the witnesses then he can create as many replicas as he wants [21].

3.2 Randomized Multicast

The randomized multicast [21] improves the simple broadcast example from the previous subsection by choosing the witnesses in a random fashion as opposed to using a deterministic function to compute them. This prevents an adversary from anticipating their identities. Whenever a node broadcasts its location claim to his neighbors, with some probability *p* these will forward it to a

randomly chosen set of *g* witnesses. Parno *et al.* show that the probability of selecting the same node more than once is negligible.

The security scheme used in the sensor networks presented in this thesis is an identity based public key system. Each sensor node γ is deployed with a private key K_γ^{-1} and other nodes can verify its signature by computing γ's public key. This can be done by using its ID, e.g. $K_\gamma = f(\gamma)$. The traditional public key infrastructure where γ's public key would be signed with the public key of a trusted authority and distributed to the network is inefficient because it has a high communication overhead.

The location claim of a node γ, loc_γ is accompanied by its ID and its signature in a message that has the following format:

$$Message = \left\langle ID_\gamma, loc_\gamma, \left\{ Hash(ID_\gamma, loc_\gamma) \right\}_{K_\gamma^{-1}} \right\rangle.$$

When a witness node receives such a message it will compute γ's public key and then use this key to retrieve the hash of the ID and the location.

$$\left\{ \left\{ Hash(ID_\gamma, loc_\gamma) \right\}_{K_\gamma^{-1}} \right\}_{K_\gamma} = Hash(ID_\gamma, loc_\gamma) = H$$

Next, the witness will compute its own hash $Hash(ID_\gamma, loc_\gamma) = H'$ and compare it with the value H. If they are equal this means that the message was send by γ and that it was not corrupted. The next step is to check whether a different location claim was received from the same node ID. If it is the case then

the respective node was replicated so a revocation message is sent through the network to revoke the node. The figure below illustrates a conflict situation at a witness node.

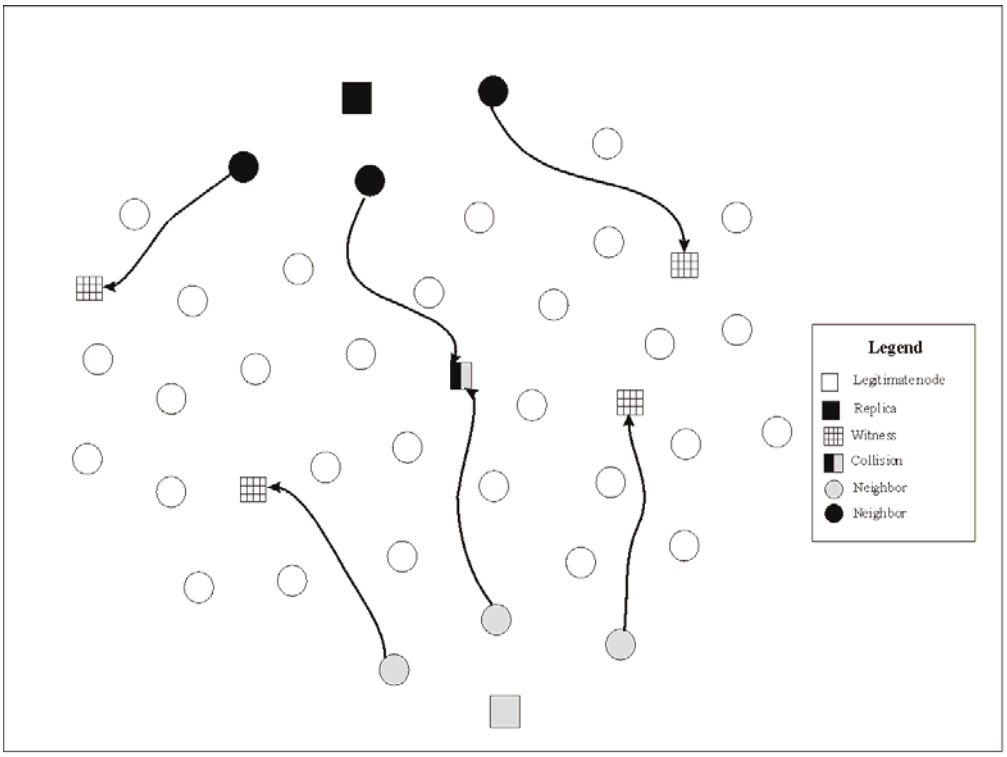

Figure 3.1 Randomized Multicast: A witness node detects a node-replica

Following the same reasoning as in the standard deviation of the birthday paradox, it is possible to determine the probability that two different location claims coming from the same ID will be detected by a witness node [21].

We assume that there are L replicas of a node α, inserted by a malicious party at locations $l_1, l_2, ..., l_L$ in the network. Let l_α be the location of the legitimate

node α, the probability that there will be no collisions is equal to the probability that the intersection of the sets of witnesses, which receive the locations, $l_1, l_2, ..., l_L, l_\alpha$ is zero.

Each set of witnesses $S_1, S_2, ..., S_L, S_\alpha$ has $p \cdot d \cdot g$ nodes:

$$\begin{aligned} S_1 &= \{w_1^1, w_2^1, ..., w_{pdg}^1\}, \\ S_2 &= \{w_1^2, w_2^2, ..., w_{pdg}^2\}, \\ &\vdots \\ S_L &= \{w_1^L, w_2^L, ..., w_{pdg}^L\}, \\ S_\alpha &= \{w_1^\alpha, w_2^\alpha, ..., w_{pdg}^\alpha\}. \end{aligned}$$

We want to find the probability that any two sets do not have any common nodes. The probability that none of the recipients of location claim l_1 receive any of the $p \cdot d \cdot g$ copies of claim l_2 is

$$\begin{aligned} P_{nc1} = P(S_2 \cap S_1 = 0) &= P(w_1^2 \cap S_1 = 0) \cdot P(w_2^2 \cap S_1 = 0) \cdot ... \cdot P(w_{pdg}^2 \cap S_1 = 0) \\ &= \left(1 - \frac{p \cdot d \cdot g}{n}\right) \cdot \left(1 - \frac{p \cdot d \cdot g}{n}\right) \cdot ... \cdot \left(1 - \frac{p \cdot d \cdot g}{n}\right) \\ &= \left(1 - \frac{p \cdot d \cdot g}{n}\right)^{p \cdot d \cdot g} \end{aligned} \qquad (3.1)$$

In a similar manner, the probability that the recipients of claims l_1 and l_2, i.e. the nodes in set S_1 and set S_2, do not receive any of the copies of claim l_3 is:

$$P_{nc2} = \left(1 - \frac{2 \cdot p \cdot d \cdot g}{n}\right)^{p \cdot d \cdot g} \qquad (3.2)$$

Thus, the overall probability of having no collisions is:

$$P_{nc} = \prod_{i=1}^{L} \left(1 - \frac{i \cdot p \cdot d \cdot g}{n}\right)^{p \cdot d \cdot g} \quad . \tag{3.3}$$

Using the approximation $(1+x) \le e^{x}$, we get the following inequality:

$$P_{nc} \le \prod_{i=1}^{L} e^{-\frac{i \cdot p^2 \cdot d^2 \cdot g^2}{n}} = e^{-\frac{i \cdot p^2 \cdot d^2 \cdot g^2}{n} \cdot \frac{L(L+1)}{2}} \tag{3.4}$$

The probability of detecting a collision will simply be:

$$\begin{aligned} P_{c_{RM}} &= 1 - P_{nc} \quad \Rightarrow \\ P_{c_{RM}} &\ge 1 - e^{-\frac{(p \cdot d \cdot g)^2}{n} \cdot \frac{L \cdot (L+1)}{2}} \end{aligned} \tag{3.5}$$

Thus, if n=3,000, g=55, d=20, and p=0.05 the protocol will detect a single replication with probability greater than 0.63 and if there are two replicas with probability greater than 0.95.

In Chapter 4 we will show how this probability changes for different values of *p, d,* and *g* and for different network sizes and what will be the optimal value for these parameters such that the probability of detecting a replicated node to be high.

In the random multicast algorithm, the sensor network detects and defeats the node replication in a distributed fashion and by selecting the witness nodes randomly, prevents the adversary from anticipating their identities.

3.3 Line Selected Multicast

Line-Selected Multicast exploits the routing topology of the network to select witnesses for a node's location. The sensor nodes are sensing units as well as routers, so when a location claim is send out it passes through several intermediate nodes. If the intermediate nodes check for replicas then we achieve higher efficiency than in the previous method. When an intermediate node receives a location claim it verifies the signature on the claim, then it compares it with the location claims already in its buffer and if there is no conflict it stores a copy of the location claim and then passes it to the next node on the route. If the intermediate node finds that it had already received conflicting location information from the same ID, it will flood the network with a revocation message, discrediting that node. Figure 3.2 illustrates the line-selected protocol:

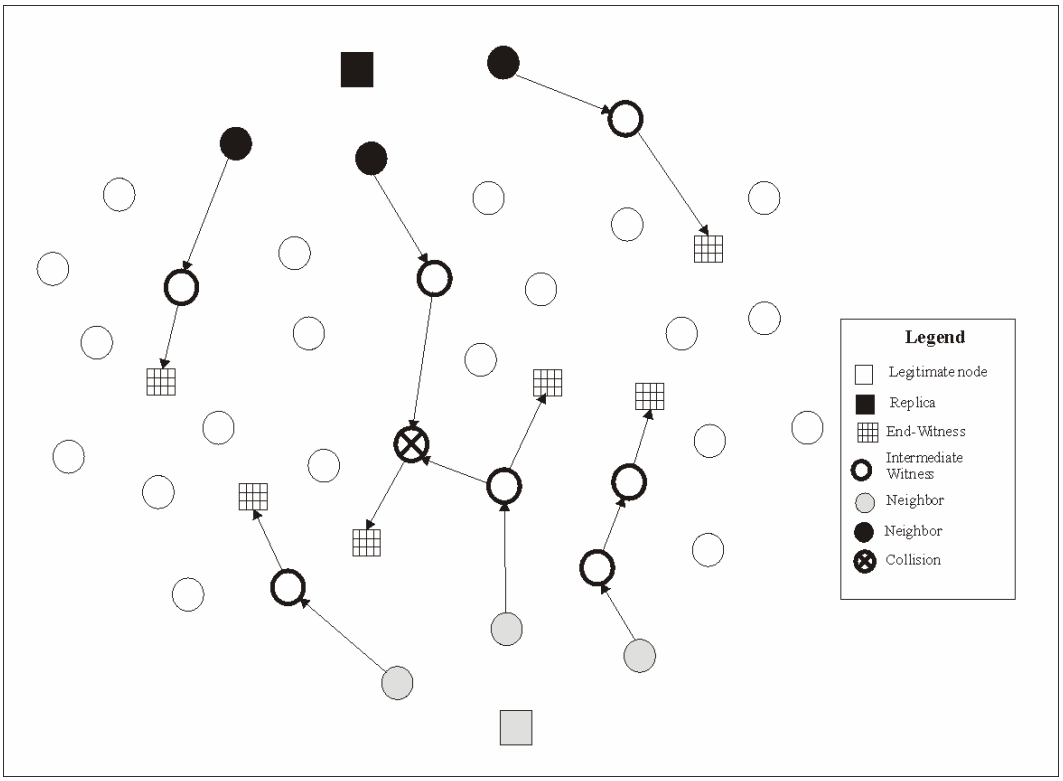

Figure 3.2 Line Selected Multicast: An intermediate witness detects a node-replica

The protocol actually draws line-segments through the network. Each node chooses $p \cdot d$ neighbors, which forward its location to g witness nodes, through some intermediate nodes, which we argued that also have the role of witnesses. From now on, we will refer to the intermediate nodes, as intermediate-witnesses and to the initial randomly chosen g witnesses as end-witnesses. The paths on which the information is send are the line-segments.

To find the probability that two line-segments intersect we use the same reasoning as in randomized multicast. First, let us consider that we have one replica. The legitimate node and its replica send out their location claims, l_α and

l_1, to $p \cdot d \cdot g$ witnesses. We assume that $p \cdot d \cdot g$ is a constant number r. Suppose that the actual claim is forwarded through the routes $Route_1^{\alpha}, Route_2^{\alpha}, ..., Route_r^{\alpha}$ and the replica's location via the routes: $Route_1^1, Route_2^1, ..., Route_r^1$. To have no collisions implies that the routes do not intersect with each other.

Let

$A_1 = \left[Route_1^1 \cup Route_2^1 \cup ... \cup Route_r^1\right]$ and $A_2 = \left[Route_1^{\alpha}, Route_2^{\alpha}, ..., Route_r^{\alpha}\right]$,

where A_1 is the set of nodes that receive location claim l_1 and A_2 the set of nodes that receive location claim l_{α}. Let L_1 and L_2 be the sizes of A_1 and A_2.

The probability that the nodes in A_1 do not receive any copies of the claim l_2 is the probability that any node in A_1 is not in A_2:

$$P_{nc1} \approx \left(1 - \frac{L_1}{n}\right)^{L_2} \leq e^{-\frac{L_1 \cdot L_2}{n}} \tag{3.6}$$

Let us consider having two replicas of a node, then the probability that any of the recipients of claims l_1 and l_{α} do not receive any copies of claim l_3 is:

$$P_{nc2} \approx \left(1 - \frac{L_1 + L_2}{n}\right)^{L_3} \leq e^{-\frac{(L_1 + L_2) \cdot L_3}{n}} \tag{3.7}$$

The probability of having no collisions is the product of P_{nc1} and P_{nc2} :

$$P_{nc} \leq e^{-\frac{L_1 L_2 + L_1 L_3 + L_2 L_3}{n}} \tag{3.8}$$

which leads to a lower bound for the probability of detecting a replica

$$P_{c_{LSM}} \geq 1 - e^{-\frac{L_1L_2 + L_1L_3 + L_2L_3}{n}} \quad (3.9)$$

We can proceed in a similar matter for cases with more than two replicas. If we assume that there are L replicas of a node, then the probability becomes:

$$P_{c_{LSM}} \geq 1 - e^{-\frac{1}{n} \cdot \sum_{i=1}^{L+1} \sum_{j=i+1}^{L+1} L_i L_j} \quad (3.10)$$

Equation 3.10 represents the expression for the minimum probability of detecting a replica, for a network of size n. To compute it ones needs to know the routing topology of the network and the number of nodes that receive a certain location claim.

In Chapter 4, we will compare the two methods and we will evaluate their performance. Simulations will be conducted on different network sizes, with different node densities, and random topologies. The results will be compared with the theoretical values obtained in this chapter and will evaluate the probability of error.

3.4 Network connectivity

Since sensor networks operate in a decentralized and self-organizing manner and do not rely on fixed infrastructure, it is important to assure that the there is a path between any two pair of nodes in the network. This propriety of a network is referred to as *network connectivity.*

In our simulation, the sensor nodes are deployed at random on a two-dimensional plane and are uniformly distributed. Two nodes are considered neighbors if they are within each other's communication range. Line-Selected Multicast algorithm exploits the routing topology of a network, so if there is no path between a pair of nodes, then a location claim cannot reach its destination. Therefore, to assure the accuracy of our simulations we must have a connected network.

To guarantee the connectivity of a graph we need to find the minimum communication range such that we'll have a sufficient average number of neighbors so that there will be no islands in the network. The connectivity depends on the number of nodes per unit area (node density) and their transmission range. Each single node contributes to the connectivity of the entire network. The correct adjustment of the nodes' radio transmission power is therefore an important feature. Coverage, communication cost and resource management depend on the positioning of the sensor nodes.

A graph G consists of a set of n vertices and a set of m pair of edges. The nodes of the network are the vertices of the graph and the connection between the nodes are the edges of the graph.

A random graph is a graph in which the edges between two vertices are established with a probability *p*:

$$p = P(\textit{two nodes } x_1 \textit{ and } x_2 \textit{ are neighbors}) = \begin{cases} 1, \ \textit{if } d(x_1, x_2) \le r_0 \\ 0, \ \textit{if } d(x_1, x_2) > r_0 \end{cases}, \qquad (3.11)$$

where, $d(x_1, x_2)$ is the Euclidian distance between nodes x_1, x_2 and r_0 is the communication range.

The links between the nodes are established based on their communication range, i.e. all the nodes which are within the communication range of a particular nodes will be considered its neighbors. If we increase the transmission power of a node, it will achieve higher transmission range and thus reach more other nodes via a direct link. If the range is very small then it is possible that some nodes remain isolated from the rest of the network.

Given an ad hoc network with *n* nodes and a homogeneous node density ρ in nodes per unit area, we want to be sure, with a probability of at least *p*, that no node in this ad hoc network is isolated, i.e. each node has at least one neighbor. Therefore, we find that the minimum communication range is [3]:

$$r_0 \ge \sqrt{\frac{\ln(1 - p^{1/n})}{\rho\pi}} \qquad (3.12)$$

The node density ρ is computed as the number of nodes in the network divided by the area *A* of the surface on which they are deployed:

$$\rho = \frac{n}{A} \tag{3.13}$$

In our simulations, we considered that the nodes are deployed on an area A=500x500 m^2 and that the number of nodes n varies between 1,000 and 10, 000. In equation (3.12) if we consider different values for p ranging from 0.9 to 0.99999 we obtain the graph in figure 3.3 which illustrates how the communication range varies for different network sizes.

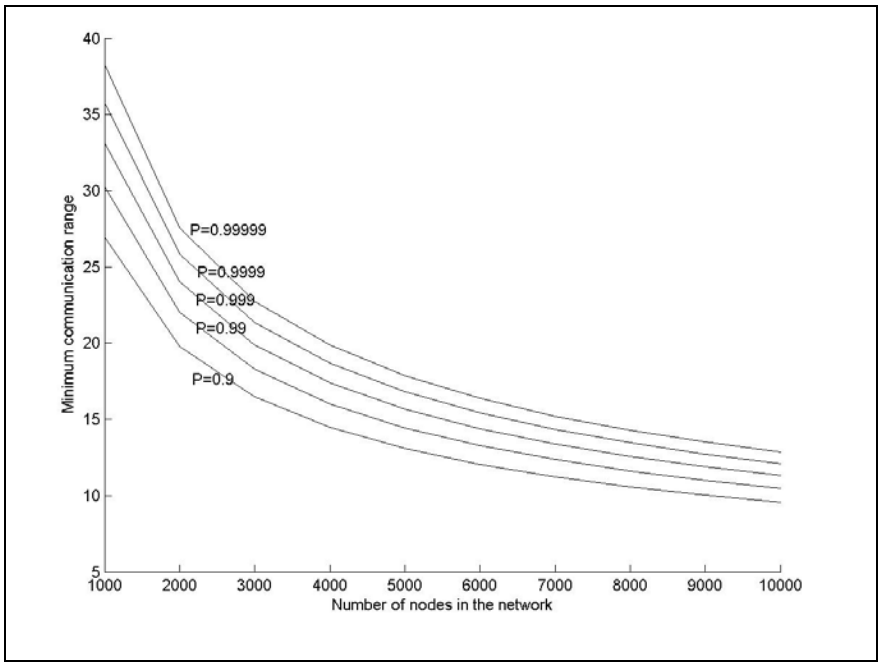

Figure 3.3 Minimum communication range for different network sizes

In order to achieve a higher probability of connectivity the communication range should be larger which implies a higher average number of neighbors also.

As the size of the network grows the expected number of neighbors decreases so a smaller communication range is necessary.

Two network topologies are presented below. First figure is an example of a disconnected network. We consider the size of network *n=500* and using equation (3.12) we obtain that the minimum communication range needed to have a connected graph is $r_0 = 41.48m$. Any communication range under this value would lead to isolated groups of nodes in the network. In the second example, we choose a communication range higher than r_0 and obtain a connected network.

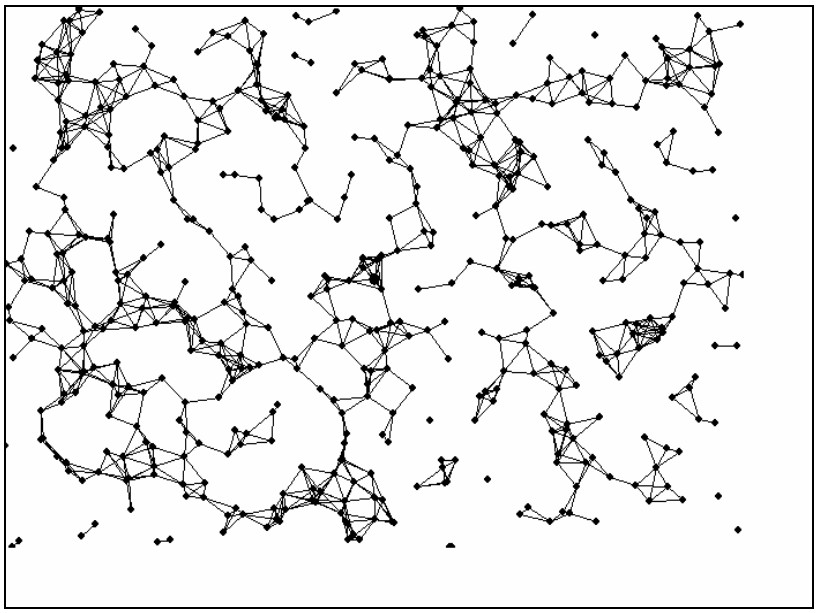

Figure 3.4 Example of a disconnected network with $n = 500$ and $r_0 = 30$

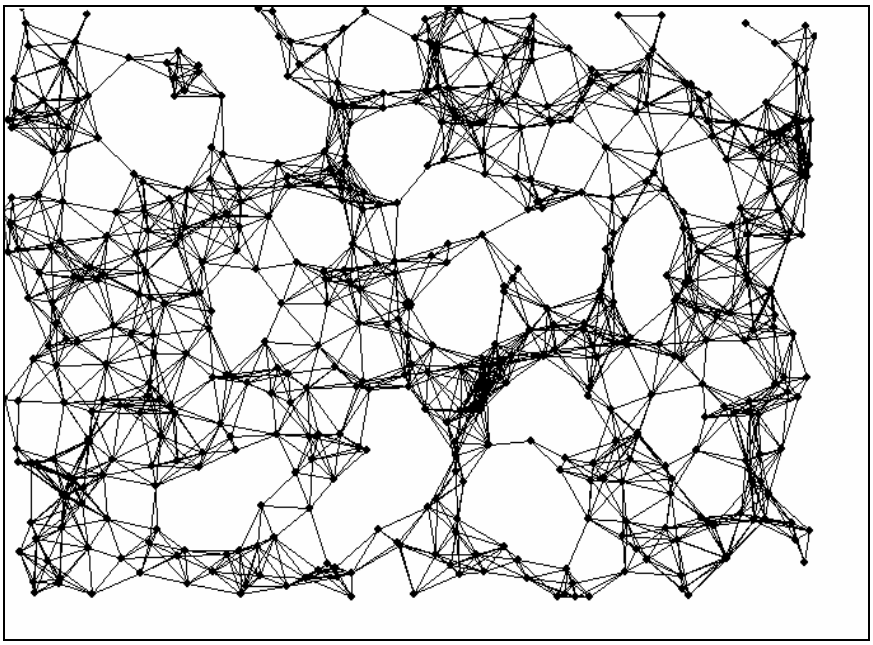

Figure 3.5 Example of a connected network with $n = 500$ and $r_0 = 45$

Analyzing the two figures above, we see that if the communication range is too small, the network will be disconnected. Although connectivity and message routing are improved by increasing the communication range, this implies more energy, thus higher battery lifetime, which raises the costs. Therefore, we need to look for an optimal communication range for a particular distributed sensor network, and this can be achieved by computer simulations.

To determine the average number of neighbors within a node's communication range *r*, we need to find first the probability that a node has *i* neighbors and then compute the expected number of neighbors.

If we have n nodes uniformly distributed on an area A with probability per unit area p, then the probability of existence of a node on area A is one so, we can determine the value of p:

$$\int_A pdA = 1 \Rightarrow p = \frac{1}{A} \tag{3.14}$$

The probability that a node is within the communication area πr^2 will be $p\pi r^2$ and the probability that a node has i neighbors out of $n-1$ possibilities is:

$$P(i\ neighbors) = \binom{n-1}{i} (p\pi r^2)^i \left(p\pi r^2\right)^{n-i-1} \tag{3.15}$$

Then the expected numbers of neighbors $E(d)$ is simply:

$$E(d) = \sum_{i=0}^{n-1} i \cdot P(i\ neighbors) = \sum_{i=0}^{n-1} i \cdot \binom{n-1}{i} (p\pi r^2)^i \left(p\pi r^2\right)^{n-i-1} = \\ = (n-1)p\pi r^2 \tag{3.16}$$

In Chapter 3 we presented our analytical solutions to the problem of determining a node-replica in sensor networks. We will also discusse the connectivity of the network, which is an important issue in the line-selected multicast protocol, because the probability of detecting a replica depends on the routing topology of the network.

Chapter 4

Evaluation

In this chapter, we present the results of our simulations. We have performed multiple tests for different network topologies and sizes and compared the results with the analytical solutions described in chapter three. The simulation results follow very closely the predictions made in the previous section.

4.1 Simulation framework

Simulations were carried out to analyze the performance of the two algorithms in detecting a node replica. The simulator was written in Matlab and run for different values of the parameters *p, d* and g. We uniformly deployed *n*

nodes on a square surface of area 500x500m^2 and using the derivations for network connectivity presented in chapter 3 determined an optimal communication range. Table 4.1 gives the value of the minimum communication range and the minimum number of neighbors for different network sizes. Since our maximum network size does not exceed 7,000, an average of 40 neighbors will be sufficient.

Number of nodes	Minimum comm. range	Minimum degree/node	Number of nodes	Minimum comm. range	Minimum degree/node
1000	38.28	18	6000	16.37	27
2000	27.57	19	7000	15.21	30
3000	22.75	20	8000	14.28	31
4000	19.85	22	9000	13.50	34
5000	17.85	25	10000	12.84	36

Table 4.1 Minimum communication range and minimum average number of neighbors

The nodes within the communication range of a specific node are considered its neighbors. We assume that a node has at least a legitimate neighbor. Without this assumption, the protocols may fail to detect node replication. For example, if the adversary captures all of node α's neighbors, then he can create a replica without being detected, since the neighbors will not be able to send out location claims for the legitimate node α. To create a second replica he must capture all the nodes surrounding the first replica. In order for the

protocol not to detect the replication, for each replica created he must compromise *d* additional nodes. Parno *et al.* describe a method to thwart this attack using pseudo-neighbors [21].

In our simulations, we considered that the adversary captures a single node and creates several clones of it. We made this simplifying assumption because we want to measure the probability of detecting one replication. Once we have this information we can easily determine the probability of detecting replicas of more than one captured node.

We fixed the position of the legitimate node, and we randomly picked the locations of the replicas. In each trial we kept the location of the original node unchanged, but modified the replicas' positions. The number of replicas was chosen between one and six.

As a routing protocol between the nodes of the sensor network, we considered the shortest path routing protocol [8]. The nodes will have a single path on which they will send the packets to the destination node. The aim of the shortest path algorithm is to find a route from the source node to the destination node that has the smallest number of hops possible. Based on the destination address of the information packet, each node keeps a table called *NextHop*, which indicates to which node to forward the packet. For example, a node will store a table similar to the one below:

If destination is node 2 then next hop is node 45
If destination is node 13 then next hop is node 59
…

Table 4.2 Next Hop Table

Some networks use flooding method for routing information. This implies that each node sends the packet to all of its neighbors; the neighbors will send it further to their neighbors and so on. This approach is very inefficient compared to the shortest path algorithm, and it consumes many network resources. However, the shortest path method has some disadvantages also. Since each node maintains the next hop value for each destination, the approach uses a lot of memory. Moreover, the design assumes that the network has a static structure. It would not be effective for mobile topologies, because the protocol for determining the routes will need to be run each time the topology changes.

However, in our simulations we are not concerned with the dynamic topologies as our aim is to measure the probability of detecting a replication. For this purpose, having just a snapshot of the topology at a certain moment of time is sufficient.

Due to the randomness of our simulation, for each plot point we took 100 samples. A 95% confidence interval was built at each point on the plot. Assuming that the sample mean of *m* observations is $\overline{Y}$, then $\overline{Y}$ is an approximation to μ, the mean of the population. However, $\overline{Y}$ has some variation from μ, so we must

estimate some interval around $\overline{Y}$ such that we can predict with some confidence that μ falls within the interval. Thus, the random variable $\overline{Y}$ is normalized by the transformation:

$$Z = \frac{(\overline{Y} - \mu) \cdot \sqrt{n}}{\sigma} \tag{4.1}$$

where σ is the population variance computed for different plot values. Z has the standard normal distribution and by letting $z_{\alpha/2}$ denote the upper $\alpha/2 \times 100$ percentile of the standard normal distribution, we have:

$$P\left[\overline{Y} - z_{\alpha/2} \cdot \frac{\sigma}{\sqrt{n}} \leq \mu \leq \overline{Y} + z_{\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}\right] = 1 - \alpha \tag{4.2}$$

The random interval $\overline{Y} \pm z_{\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}$ is the confidence interval and $1 - \alpha$ is the confidence level. We considered the value of the confidence interval to be 0.95, which yields: $z_{\alpha/2} = z_{0.025} = 1.96$ [18]

4.2 Results and observations

4.2.1 Randomized Multicast

In Chapter 3, we showed that the lower bound probability for detecting a collision at a witness node has the following expression:

$$P_{cRM} \geq 1 - e^{-\frac{(p \cdot d \cdot g)^2}{n} \cdot \frac{L \cdot (L+1)}{2}} = P_{\min} \quad (4.3)$$

We are interested in analyzing the behavior of $P_{\min}$ for different values of the number of witnesses and for different replicas. In addition, we want to verify the accuracy of our predictions by running simulations and comparing the experimental results to the theoretical ones.

In Figures 4.1 and 4.2, we plotted $P_{\min}$, for different values of the number of witnesses $p \cdot d \cdot g$ and for network sizes $n = 5000$ respectively $n = 7000$. The rate at which the probability increases is proportional to the square of the number of witnesses. To find the minimum value for $p \cdot d \cdot g$ such that the probability of detecting a node replication is greater than 50% we took $P_{\min} = 0.5$ in equation (4.3):

$$p \cdot d \cdot g \geq \sqrt{\frac{1.38 \cdot n}{L(L+1)}} \quad (4.4)$$

Equation 4.4 shows that to detect a collision with a probability greater than 50% $p \cdot d \cdot g$ needs to be of the order $O(\sqrt{n})$. This is also obvious from the definition of the birthday paradox, which states that for $\sqrt{n}$ independent samples from a population of size *n,* the probability that a pair of samples will have the same value is at least 0.5. This implies a relatively high storage cost, because on average each node will need to store $p \cdot d \cdot g$ location claims. Similarly, the communication requirements are high as well. If we consider that the average distance in a network randomly deployed on the unit square to be approximately

$\sqrt{n}/2$, then the total communication cost will be about $O(n \cdot \sqrt{n} \cdot p \cdot d \cdot g)$. Since, $p \cdot d \cdot g \approx O(\sqrt{n})$ we get that the communication cost is approximately $O(n^2)$, equivalent to the ones of the simple broadcast scheme discussed in Chapter 3 [21].

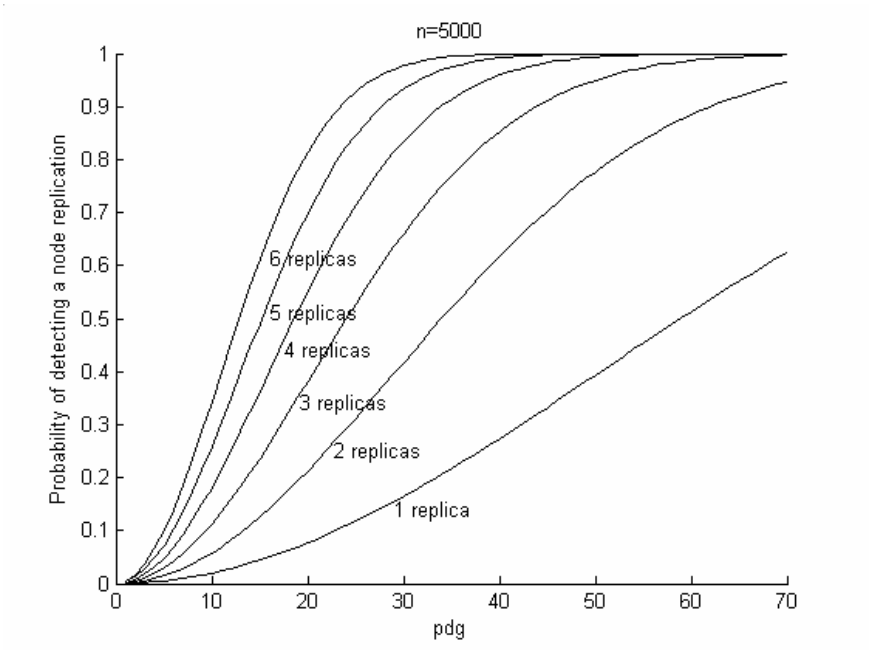

Figure 4.1 Randomized Multicast –analytical solution n=5000

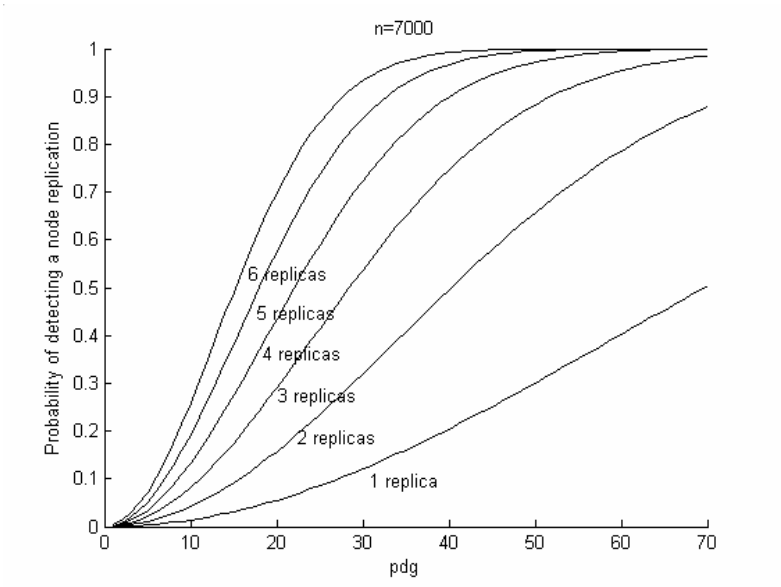

Figure 4.2 Randomized Multicast – analytical solution n=7000

We have run multiple simulations to verify the theoretical derivations. For each value of $p \cdot d \cdot g$ and L , we took 100 samples, ran the protocol and averaged the results. The reason we took 100 samples was that we wanted to have a good estimate for the probability of detection, meaning a smaller confidence interval. Our results show that for small values of $p \cdot d \cdot g$ the confidence intervals do not overlap but as $p \cdot d \cdot g$ increases and the probability of detecting a replica is higher than 90% there are some overlaps (for different values of L). Overall, the confidence intervals turned out to be at a small range of less than 6% deviation from the mean values.

For the legitimate node as well as for the replicas we choose $p \cdot d \cdot g$ nodes, which will receive their location claim, and we measured the probability of detecting a replica by counting the number of successes out of the total number of times we ran the protocol:

$$P_{\text{detection}} = \frac{\text{number of successes}}{\text{number of repetitions}} \tag{4.5}$$

The results are presented in Figures 4.3 and 4.4. Comparing the output of the simulation with the analytical results, we see a very high similarity. In the simulated results the probability of detecting a replicated node has an exponential form and is higher than the theoretical result (minimum value), which indicates that the analytical formulas in Chapter 3 are valid.

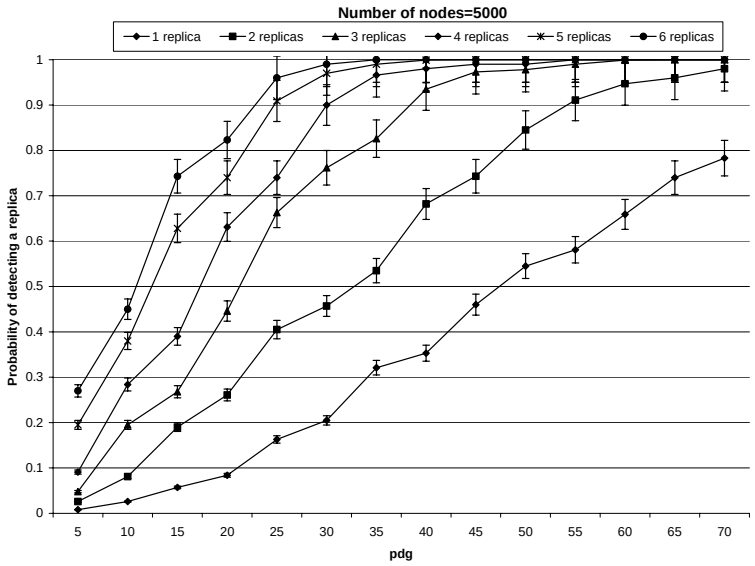

Figure 4.3 Simulation results for Randomized Multicast n=5000

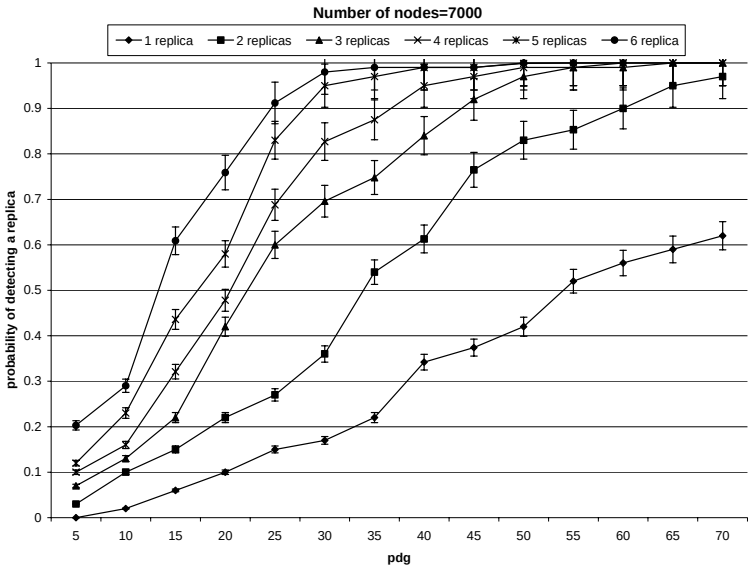

Figure 4.4 Simulation results for Randomized Multicast n=7000

We also analyzed cases when one of the parameters *p, d* or *g* varies and the others are constant. The results show that it does not matter how *p, d*, or *g* are changing, as long as the number of witnesses, $p \cdot d \cdot g$, increases, the chance of a collision increases.

4.2.2 Line-Selected Multicast

In Line-Selected Multicast, the probability of detecting a replica depends on the topology of the network and on the routing protocol. For each location, $p \cdot d$ neighbors choose *g* witness nodes to which they send the information. The location claim is routed through intermediate nodes, which check for collisions as

well. They are also counted as witnesses. In Chapter 3 we found a lower bound for the probability of a detecting a replica using line-selected multicast algorithm.

$$P_{c_{LSM}} \geq 1 - e^{-\frac{1}{n}\cdot\sum_{i=1}^{L+1}\sum_{j=i+1}^{L+1} L_i L_j} \tag{4.6}$$

In equation (4.6), $L_1, L_2, ..., L_L$ represent the number of nodes that receive location claims $l_1, l_2, ..., l_L$ sent by the replicas, and L_α is the number of nodes that receive location claim l_α sent the legitimate node α. These values are determined as the average of a series of tests. In Figure 4.5, we considered just a single node and based on the number of line segments radiating from it, we estimated the average size of the set of nodes that receive its location.

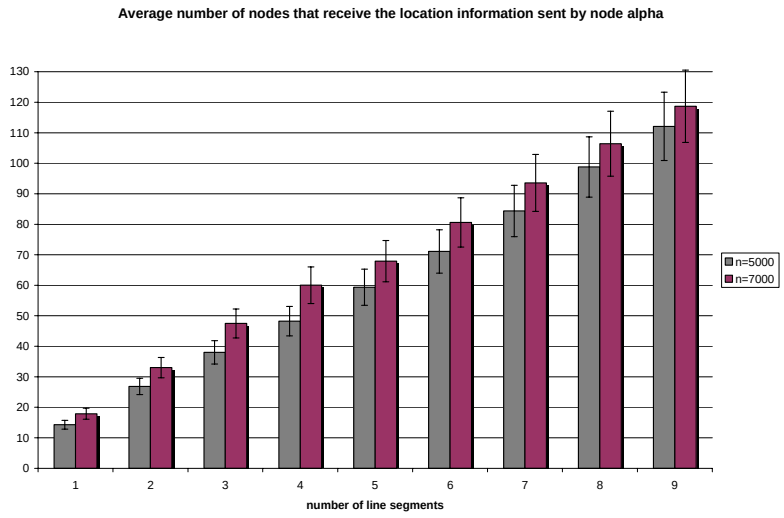

Figure 4.5 Average number of nodes that will receive a location claim

As expected, for larger network sizes since the length of the routes are longer, the size of L_i is larger. In line-selected multicast protocol, the sets of nodes

that receive location claims $l_1, l_2, ..., l_L, l_\alpha$ do not have a fixed size as in randomized multicast where the size of these sets was constant $p \cdot d \cdot g$. Here, $p \cdot d \cdot g$ represents the number of line segments radiating from a node and each node on these segments is counted as witness.

The minimum probability of detecting a node replica is severely impacted by the number of line segments, $p \cdot d \cdot g$. For small values, the chances of a collision are low, but as additional segments are added it grows exponentially, as it can be seen from Figures 4.6 and 4.7.

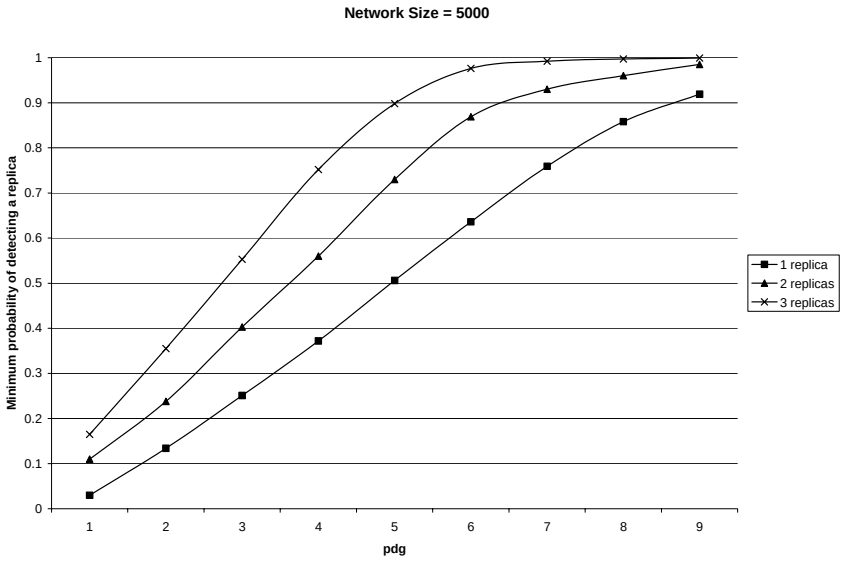

Figure 4.6 Line-Selected Multicast – analytical solution n=5000

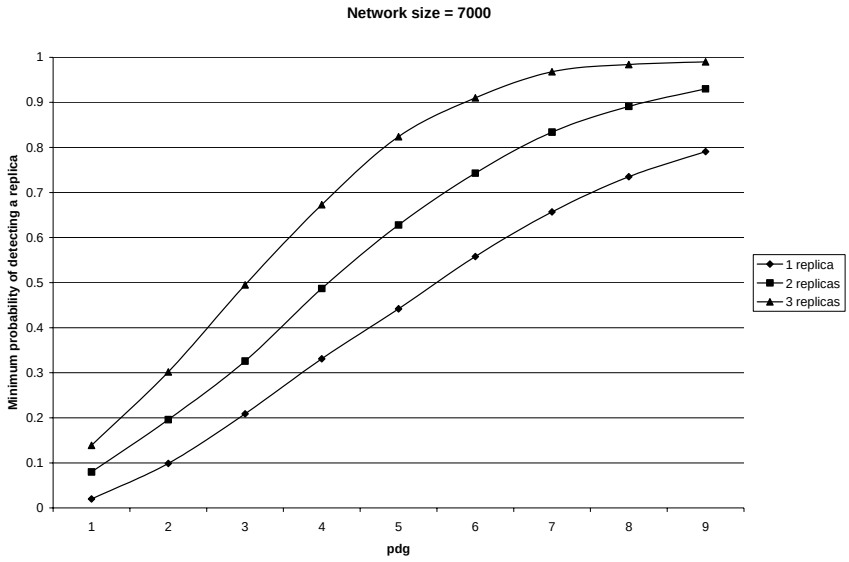

Figure 4.7 Line-Selected Multicast – analytical solution n=7000

We observe that just by increasing $p \cdot d \cdot g$ with one unit, the probability of detecting a replica increases at a higher rate than in the randomized multicast. The reason is the different interpretation of $p \cdot d \cdot g$ in the two protocols.

To check the validity of the analytical formulas, we ran a series of simulations. In order to detect a collision we need to have at least two line-segments, transmitting conflicting location information, intersect. In our simulation, we generated the line-segments by using the shortest distance routing algorithm and then checked to see if any two routes have common nodes. If they did, then we marked the trial as successful, otherwise as failure. We repeated the protocol several times and took the average of the results.

We choose 10 random network topologies and for each of them ran 10 trials. The confidence interval for the various plot points was taken at 95% and the results showed 10% deviation from the resulting mean.

In each trial, we ran the protocol until we obtained a collision. We measured the probability of detecting a node replica as:

$$P_{\text{detection}} = \frac{1}{\text{number of repetitions}} \tag{4.7}$$

and then averaged the probabilities obtained for each network configuration. The results can be seen in Figures 4.7 and 4.8.

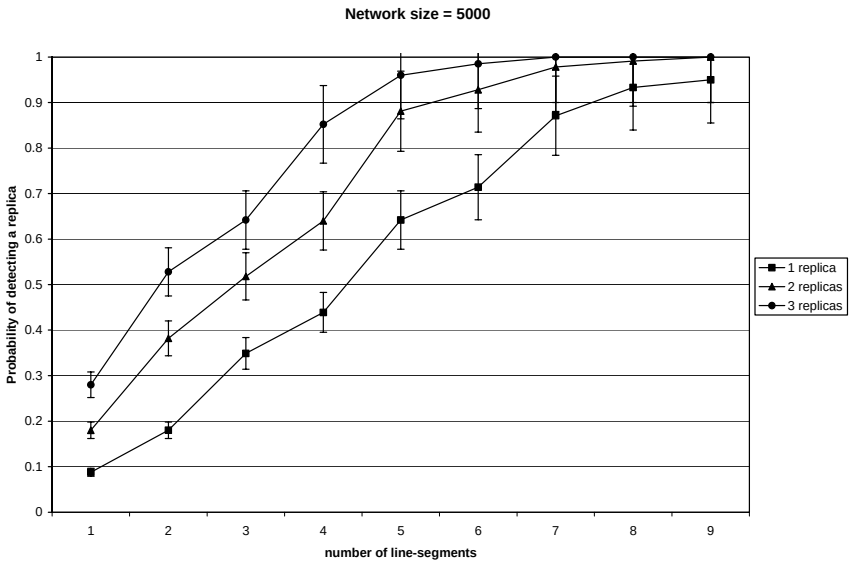

Figure 4.8 Simulation results for Line-Selected Multicast n=5000

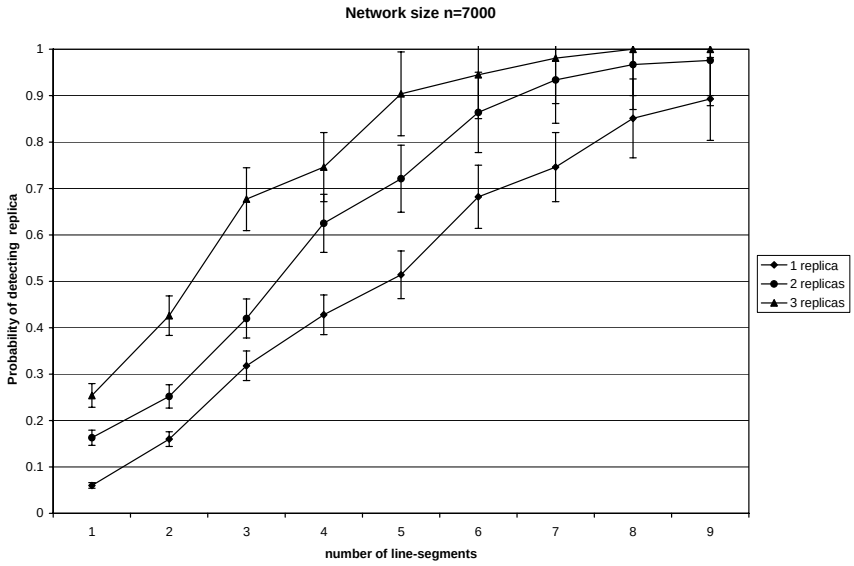

Figure 4.9 Simulation results for Line-Selected Multicast n=7000

By comparing the simulations with the analytical formulas, we see a high similarity. The probability has an exponential growth and it reaches higher values than the theoretical results, as expected. Thus, it is possible to detect a collision with probability 60% just by using six line-segments for a network of size 7,000 and five line-segments for a network size of 5,000.

Looking closely at the two protocols, we see that the numbers of nodes that actually check for collision are of the same order. If we take the average path between two random nodes in a network randomly deployed on the unit square to be approximately $\sqrt{n}/2$, we obtain that on average each node will need to store $p \cdot d \cdot g \cdot \sqrt{n}$ locations, thus the memory cost will be of the order $O(p \cdot d \cdot g \cdot \sqrt{n})$.

Since $p \cdot d \cdot g$ is much smaller than $\sqrt{n}$, the memory cost can be approximated with $O(\sqrt{n})$, which is equal to the memory cost of the randomized multicast scheme. The communication cost, will be of the order $O(p \cdot d \cdot g \cdot n\sqrt{n})$, and because $p \cdot d \cdot g$ is a small constant, the order simplifies to $O(n\sqrt{n})$. Compared to the randomized multicast scheme it is smaller by a factor of $\sqrt{n}$. Thus, Line-Selected Multicast proves to be more efficient in terms of communication cost, which is an important issue in sensor networks. The communication for Randomized Multicast scales linearly with the number of nodes, while in Line-Selected Multicast it grows with $O(\sqrt{n})$.

In Line-Selected Multicast protocol as well as in Randomized Multicast we observe that if we keep the number of witnesses $p \cdot d \cdot g$ constant the probability of detecting a replica will decrease as the size of the network increases. The reason is simple: since the network becomes more populated, the chance of two or more location claims to collide at a node becomes lower.

We also analyzed cases in which we keep constant the number of neighbors $p \cdot d$, that send out the location claim of a node, and vary the number of witnesses *g*, and cases in which *g* was kept unchanged and $p \cdot d$ varied. Our simulation results showed that it does not matter that much whether $p \cdot d$ changes or *g* changes. The only thing that influences the probability is the product $p \cdot d \cdot g$. If it is larger, then there will be more witness nodes, so the chances of detecting a

collision are higher. In the case of Line-Selected Multicast, this requires a large number of line segments, so more nodes will be engaged in the replicated node detection protocol. To improve the probability of detection we can repeat the protocol or add a few additional line segments per node.

Chapter 5

Conclusions and Future Work

Due to the low-cost nature of sensor nodes, it is impractical to use shielding methods that would protect the nodes against access to its memory, processing, sensing and communication components. Thus, if the sensor nodes are deployed in hostile environments, an adversary can easily capture legitimate nodes, replicate and insert replica nodes in the network. This thesis evaluated the performance of two distributed algorithms for detection of node replication attacks in sensor networks, randomized multicast and line-selected multicast. First, we derived an analytical solution for the detection problem, and then we ran simulations to verify our predictions. In particular, we studied the probability of detecting a node replica in the network for variations in the input parameters: the number of witnesses or line-segments and the number of replicas.

The results obtained from the simulations were at par with our expectations. In randomized multicast, increasing the number of witnesses, *g,* or

the number of neighbors $p \cdot d$ that send out the location claim of a node α, led to increase in the probability of detecting a replica. We observed that increasing $p \cdot d$ while keeping *g* constant, or increasing *g* and keeping $p \cdot d$ constant led to the same results, so in our simulations we varied $p \cdot d \cdot g$, the total number of nodes that receive the location information of node α.

In line-selected multicast scheme (LSM), $p \cdot d \cdot g$ represents the number of line-segments that radiate from a node α, carrying out its location information. Each intermediate node on the route from a neighbor to a randomly chosen witness node, checks for replicas as well. Therefore, they are witness nodes as well. Our results show that with just a handful of line-segments radiating from a node there is a high chance to detect a node replica.

For each number of line-segments we also gave an estimate on the average number of nodes that will receive the location information of a node α. If we compare this numbers with the number of witnesses from the randomized multicast protocol, we see that in order to obtain the same probability of a collision we need about the same number of nodes. The difference is that line-selected multicast is more efficient in terms of communication overhead. The reason is that in line-selected protocol, the intermediate nodes are not idle, and by including them in the detection scheme, we do not need to choose additional witnesses.

Considering that distributed sensor networks are developing fast, the node-replica detection problem, presented in this thesis has to be studied further. One

might use different routing protocols and compare the performance of the two algorithms. The results obtained will allow us to study the impact these schemes have on different topologies of sensor networks, as well as on choosing the optimal input parameters.

Bibliography

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci. “A survey on Sensor Networks”, *IEEE Communication Magazine*, August 2002

[2] M.Bawa, H. Garcia-Molina, A. Gionis and R.Motwani. “Estimating aggregates on a peer-to-peer network”. Technical Report, Stanford University, 2003.

[3] C. Bettstetter, “On the minimum node degree and connectivity of a Wireless Multihop Network” In Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing, June 2002.

[4] D.Braginsky and D. Estrin. “Rumor routing algorithm for sensor networks”. In *Proceedings of ACM Workshop on Wireless Sensor Networks,* 2002.

[5] H. Chan, A. Perrig, and D.Song. “Random key predistribution schemes for sensor networks”. In *Proceedings of IEEE Symposium on Security and Privacy,* May 2003.

[6] M. Chen, W. Cui, V. Wen and A. Woo. “Security and deployment issues in a sensor network. Ninja Project: A scalable Internet Services Architecture”, Berkeley

[7] C-Y Chong and S. P. Kumar. “Sensor Networks: Evolution, Opportunities, and Challenges”. In *Proceedings of the IEEE, Vol. 91, No. 8,* August 2003.

[8] T. Cormen, C. Leiserson, R.Rivest and C. Stein. “Introduction to Algorithms”. MIT Press, 2001

[9] J. Deng, R. Han, and S. Mishra. “A performance evaluation of intrusion-tolerant routing in wireless sensor networks.” In *2nd International Workshop on Information Processing in Sensor Networks (ISPN ’03)*

[10] J.R. Douceur. “The Sybil attack”. In *Proceedings of Workshop on Peer-to-Peer Systems (IPTPS),* Mar. 2002.

[11] L. Eschenauer and V.Gligor. “A key-management scheme for distributed sensor networks”. In *Proceedings of the ACM Conference on Computer and Communication Security (CCS),* Nov. 2002

[12] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar, “Next century challenges: Scalable coordination in sensor networks” In *Mobile Computing and Networking*, 1999

[13] D. Hwang, B. Lai, and I. Verbauwhede “Energy-memory-security tradeoffs in distributed sensor networks” In *3rd International Conference on Ad-Hoc Networks and Wireless (ADHOC-NOW 2004)*

[14] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, “System architecture directions for networked sensors,” in *Proceedings of ACM ASPLOS IX*, November 2000.

[15] C. Karlof and D. Wagner. “Secure routing in wireless sensor networks: Attacks and coutermeasures”. In *First IEEE International Workshop on Sensor Network Protocols and Applications,* pages 113-127, May 2003

[16] B. Karp and H. T Kung. “GPSR: greedy perimeter stateless routing for wireless networks”. In *International Conference on Mobile Computing and Networking,* pages 243-254, 2000.

[17] C. Kaufman, R. Perlman and M. Speciner, “Network Security: Private Communication in a Public World”, Prentice Hall, 2002

[18] H. Kobayashi, “Modeling and Analysis: An introduction to System Performance Evaluation Methodology”, Prentice Hall, 2002

[19] R. Motwani and P. Raghavan. “Randomized Algorithms” *Cambridge University Press.* June 1995.

[20] J. Newsome, E. Shi, D. Song and A. Perrig. “The Sybil attack in sensor networks: Analysis and defenses”. In *ACM Conference on Embedded Networked Sensor Systems (IPSN),* Nov. 2003.

[21] B. Parno, A. Perrig, V. Gligor, “ Distributed detection of node replication attacks in sensor networks” In *IEEE Security and Privacy Symposium,* May 2005

[22] A. Perrig, R. Szewczyk, V. Wen, D. Culler and J.D. Tygar. “SPINS: Security protocols for sensor networks”. In *ACM Conference on Mobile Computing and Networks (MobiCom),* July 2001.

[23] N. Sastry, U. Shankar, and D. Wagner. “Secure verification of location claims”. In *Proceedings of ACM Workshop on Wireless Security (WiSe 2003),* September 2003.

[24] J. Spencer. “The Strange Logic of Random Graphs Algorithms and Combinatorics” Springer-Verlag 2000, ISBN 3-540-41654-4.

[25] H. Solomon. *Geometric Probability.* Society for Industrial and Applied Mathematics (SIAM), 1978

[26] S. Weingart. “Physical security devices for computer sub-systems: A survey of attacks and defenses”. In *Cryptographic Hardware and Embedded Systems (CHES),* Aug. 2000.