ABSTRACT

Title of Dissertation: Security of Wireless Sensor Networks in the Presence of Captured Nodes

Seyed Farshad Bahari,
Doctor of Philosophy, 2008

Dissertation directed by: Professor Virgil D. Gligor
Department of Electrical & Computer Engineering

Wireless sensor networks (WSNs) pose unique security challenges due to the fact that their nodes operate in an unattended manner in potentially hostile environments. A particularly difficult problem not addressed to date is the handling of node capture by an adversary. A key goal for solving this problem is that of limiting the damage caused by captured nodes. This is important since node capture cannot be prevented: by definition, there is no practical physical mechanism that could keep an adversary from physically accessing a sensor node discovered in an unattended area. Hence, the presence of the adversary within a WSN must be detected, and of course, the earlier the better. Adversary detection is predicated on the fact that access to a captured node's internal state, which includes secrets such as cryptographic keys, incurs a nonzero time delay. This suggests that adversary detection be divided into two phases: (i) *in-capture* detection, namely detection before the

adversary completes the capture process and gets a chance to access a node's internal state and do any network damage, and (*ii*) *post-capture* detection, namely detection after the adversary already accessed and possibly used a node's internal state and secrets. Since the adversary is already active in the network in the latter case, it is important to determine the overall network resiliency; i.e., the ability of the network to operate in the presence of an active adversary. In this work we focus on the former case in which we try to identify the presence of the adversary prior to completion of a node capture.

To address the problem of *in-capture* adversary detection, we propose two probabilistic schemes called the *pairwise pinging* scheme and *quorum pinging* scheme, whereby the network continuously monitors itself in a distributed and self-organizing manner. We investigate the trade-offs between the network cost-performance and security of these schemes via a *Markov Chain* model, and present analytical solutions which allow us to choose appropriate performance parameters, such as the expected residual time-to-false-alarm, and security, such as the probability of a missed detection. We show that the quorum pinging is superior to pairwise pinging in terms of both cost-performance and security. Furthermore, we will show that both schemes are scalable with network size and their complexities are linearly proportional to the average node degree of the network.

We also analyze the optimum strategy for an adversary to deploy its agents over a sensor network; i.e., the strategy that enables the adversary to achieve the maximum capture ratio with fixed number of agents. The order of node capture, distribution, and location of agents are investigated and an

analytical model is provided that describes the optimum path for deploying of agents to target nodes. Numerical data are presented to compare different scenarios for deploying agents and the corresponding performance of each deployment strategy. The proposed optimum strategy validates the physical interpretation under practical scenarios and demonstrates the feasibility of our capture strategy in practice. Finally, the resiliency of the underlying quorum pinging scheme for detecting adversary agents is investigated despite collusion among agents via optimum capture strategy.

Security of Wireless Sensor Networks in the Presence of Captured Nodes

by

Seyed Farshad Bahari

Dissertation 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
Doctor of Philosophy
2008

Advisory Committee:

Professor Virgil D. Gligor, Chair/Advisor
Professor Charles Silio
Professor Shuvra Bhattacharyya
Professor John Baras
Associate Professor Mark Austin

Dedication

To my father, Hashem, and my mother, Aghdas

Acknowledgments

I would first like to thank my advisor, Professor Virgil Gligor. Virgil is a man of integrity and honor. From the beginning he inspired me and helped me build my confidence in this project. Through his technical expertise, he fostered my capabilities and mentored me with patience, respect and grace, steering me to the point where I would excel and be successful. I was honored to work with him for the duration of this project. He has truly been, and continues to be a role model for my professional life.

I acknowledge the financial support of this work by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0011 for the Collaborative Technology Alliance in Communications and Networking.

I would also like to give special thanks to my dear friend Dr. Mehdi Kalantari. I have known Mehdi as a friend and colleague for many years. I have been privileged to have him as a mentor since the beginning of my program. His input over the course of my work has truly been invaluable to me.

I also express my appreciation to Mr. Steve Amar, who tirelessly proofread this manuscript, compensating for my shortcomings in the English language

and making valuable suggestions to improve its readability.

Finally, I give my greatest thanks to my parents, Hashem and Aghdas. Throughout my life, they have always encouraged and supported me from as early as I can remember. They tirelessly provided for me and made it possible for me to excel in my life. Their continual emotional, spiritual support and involvement; even during our years of separation and over the distance, has truly been immeasurable. If it were not for their input, love and leadership, I would not have been able to progress to where I am now in my life. In many ways, this achievement belongs to them also. I love you both very much.

TABLE OF CONTENTS

List of Tables

List of Figures

Chapter 1

Introduction

Wireless Sensor Networks (WSNs) consist of arrays of *sensor nodes* that are battery powered, have limited computational capabilities and memory, and rely on intermittent wireless communication via radio frequency and, possibly, optical links. WSNs also include *base stations*, which have two primary functions: (*i*) collection and caching of data received from sensor nodes and making these data available for processing to application components of the network, and (*ii*) monitoring the status and broadcast of simple control commands to sensor nodes. Although in WSNs most nodes have limited, if any, mobility after deployment, some nodes can be highly mobile (e.g., base stations placed on humans, vehicles, aircraft). This implies that the array of sensor nodes may need to implement dynamic routing functions that channel the sensed data to the nearest base station.

WSNs have several characteristics that distinguish them from traditional wireless networks, namely: their scale is orders of magnitude larger than that of typical wireless networks (e.g., tens of thousands as opposed to

just tens of sensor nodes); they are extensible in the sense that they allow the addition and deletion of sensor nodes after network deployment without administrative intervention or physical contact by a human; and they may be deployed in hostile areas where communication is monitored and sensor nodes are subject to capture and manipulation by an adversary.

WSNs have received significant research attention due to their unique characteristics, such as limited resource constraints and unreliable wireless communication over an infrastructure-less network with ad-hoc connectivity. WSNs are intended to provide services such as data acquisition, data aggregation, and data routing. Their application spectrum covers various issues such as environmental monitoring, structural health monitoring, surveillance, seismic analysis, and tracking moving objects.

Goals. WSNs pose unique security challenges due to the fact that their nodes operate in an unattended manner in potentially hostile environments. A particularly difficult problem not addressed to date is the handling of node capture by an adversary. A captured node can disclose all keying material required for providing security services to the adversary. As a consequence all security services in the network may be compromised. To date, most security protocols fail to deal with this type of attack as they cannot distinguish between legitimate and illegitimate nodes having access to keying material [1, 2]. A key goal for handling such attacks is that of limiting the damage caused by captured nodes, since node capture cannot be prevented. By definition, there is no practical physical mechanism that could keep an adversary from physically accessing a sensor node discovered in an unattended area. Stajano's *big*

stick principle [3], which states that whoever has physical control of a device is allowed to take it over, suggests that such an adversary is more powerful than the Dolev-Yao [4] and traditional Byzantine adversaries [5], and hence difficult to counter. Hence, the presence of the adversary within a WSN must be detected, and of course, the earlier the better. However, the detection of an adversary that does *not* attempt to access a node's internal state and secrets but instead manipulates a sensor node's inputs is less relevant to this research. Redundant coverage of the sensed area with multiple nodes and statistical analysis of correlated data can be used to detect the existence of corrupt inputs [2, 6].

A desirable but unrealistic goal would be to prevent an adversary from accessing a node's internal state and secrets, and hence to prevent any damage to network and application operations. Traditional mechanisms for accomplishing this include use of physical tamper-proof nodes and node shielding. Current state-of-the-art technology used in protecting device secrets (e.g., cryptographic keys) via physical security mechanisms - which currently range from those employed by smart cards (very little tamper resistant), to IBM's 4758/64 crypto co-processors (highest FIPS 140-1 certified evaluation hardware modules [7]), and to Physically Unclonable Functions [8, 9, 10](PUFs, very good but not perfect physical security) - will continue to require network security measures. That is, network security measures will continue to be necessary for complementing physical device protection for the foreseeable future.

Previous research has already addressed the issue of the dependability of the device's physical protection by the network security measures. The re-

sult was that most physical security measures were either unsuitable because they were too weak (e.g., those employed by smart card technologies) or, if they were strong, their form factor and power requirements made those devices unsuitable for most MANET and sensor network applications (e.g., form factor and power requirements in the case of IBM 4758/64 crypto co-processors). The form factor of such devices (e.g., size is at least double that of a sensor node) can contribute to the discovery of a sensor placed in a hostile environment. Further, the use of such devices is prohibitively expensive (e.g., the cost of IBM's 4758 card used traditionally in banking applications is between three and four order of magnitude higher than the cost of a typical sensor node). Also, other technologies such as PUFs offer limited protection against cryptographic attacks. Once PUF security is broken (e.g., secret key discovered with a non-negligible probability) the PUF device becomes unusable, since the secret key cannot be changed. Furthermore, recent data obfuscation techniques [11, 12] provide resiliency against node capture to some extent by postponing an adversary's success in accessing the internal states of the target node.

Research Problems and Approaches. In our research, adversary detection is predicated on the fact that access to a captured node's internal state, which includes secrets such as cryptographic keys, incurs a nonzero time delay. After discovering a sensor node location, the adversary must disassemble that node in order to get access to its state and secrets. Hence, such access is a time consuming procedure. The amount of time it takes for the adversary to access a node's internal state without inadvertently erasing most of it is a non-deterministic variable which depends on the degree of physical node protection, data obfuscation level, environmental conditions of the sensors' in-

stallation site, and the skill and technology level of the adversary. Regardless of the nonzero amount of time it takes for an adversary to get to a captured node's internal state, that node would face some delay in its normal operation. This suggests that adversary detection be divided into two phases: (*i*) *in-capture detection*, namely detection before the adversary gets a chance to access a node's internal state and do any network damage, and (*ii*) *post-capture detection*, namely detection after the adversary has accessed and possibly used a node's internal state and secrets. Since the adversary is already active in the network in the latter case, it is important to determine the overall network resiliency; i.e., the ability of the network and its applications to operate in the presence of an adversary.

Our research focuses on the *in-capture* detection phase and its impacts on important applications in WSN such as message authentication and data aggregation because they are basic to a WSN's operation. The security of the data aggregation process needs to be provided in the face of adversary attacks such as false data injection and data corruption. To address the problem of *in-capture* adversary detection, we propose two probabilistic schemes called the *pairwise pinging scheme* and the *quorum pinging scheme*, whereby the network continuously monitors itself in a distributed manner. Intuitively, adversary detection can be achieved if the node absence from the network exceeds the node monitoring period during which the node has to respond to neighbors' queries regarding its status. We investigate the trade-offs between the network cost-performance and security of these schemes via a *Markov Chain* model, and present an analytical solution which allows us to choose appropriate performance parameters, such as the expected residual time-to-false-alarm

and probability of false alarm; and security, such as the probability of a missed detection. Performance evaluations of the proposed schemes over a wide range of parameters will support our analytical results. We then generalize our proposed model to the case in which the adversary can deploy multiple agents in the WSN over various targets in a collaborative manner, i.e., as colluders. The quorum pinging scheme is in charge of detecting the presence of adversary agents presiding over target nodes. On the other hand from adversary's perspective, the adversary's objective is to maximize the number of successfully captured nodes using its limited number of on hand agents. We investigate the optimum strategies for an adversary in terms of the proper distribution and order of deploying its agents in order to maximize its gain. Required countermeasures to combat against the proposed optimal capture strategies are presented as well.

Research Outline. In Section 1.1 of this work, we summarize the operational assumptions made and the definitions used. In Section 1.2, we present the communications and adversary models used. In Chapter 2, we present a brief overview of the proposed schemes addressing the captured node problem. Our goal is to point out the drawbacks of these approaches and identify the properties of a potential solution which can be capable of countering the captured node problem properly. It is worth mentioning that our proposed methodology presented in the following chapters, is not directly related to any of these previous approaches, rather we use these schemes in order to precisely identify the required features that we plan to achieve. In Chapter 3, we present the proposed pinging method to tackle the problem defined above, namely the *in-capture* adversary detection. Following that, we introduce the two proposed

schemes, namely the pairwise pinging scheme and the quorum pinging scheme in details. Chapter 4 focuses on achieving optimal capture strategies for an adversary with multiple agents engaged in a collusion. Finally, Chapter 5 concludes this research.

1.1 Operational Assumptions and Definitions

In WSN applications, we have to impose certain specifications into any proposed mechanism which tries to address the captured node problem. The nature of traffic is *event driven* which means that the observed data expressing the occurrence of the event of interest happens occasionally and are sensed as *redundant information* by multiple nodes, mostly regardless of their locality. The presence of an adversary requires *node-to-node authentication* guaranteeing secure communication against man-in-the-middle attacks across the network terrain. Finally, resource constraints of battery powered nodes restrict the usage of only *lightweight cryptographic* tools.

The architecture of WSN requires another set of properties as follows. The design should be *scalable* as participating nodes can be incrementally added or dropped from the network during the network lifetime. The ad-hoc formation of the network imposes an infrastructure-less connectivity with no central authority and no *a priori* knowledge of random neighbors. Any proposed mechanism should be implemented in a *distributed* and *self-organizing* manner which is capable of countering underlying *unreliable wireless communication* issues such as packet loss.

Assumption 1 The end-nodes and the aggregators can be captured by the

adversary.

Each node at any given time can be an information source (*end-node*), a relay with local function (*aggregator*), or a destination node (*base station*). The first two can encounter the captured node problem.

Assumption 2 The base station is assumed to be trusted, and to be capable of one hop downlink communications.

The base station is usually assumed to be a permanent reliable and trusted authority and impervious to compromise by adversaries. The base station is located in some safe location that adversaries can not have access to and is usually run in the presence of a human operator. Furthermore, there are enough resources for the base station to be able to broadcast its queries across the entire network by high power radio as it requires only one investment and is therefore affordable.

Assumption 3 The pairwise key establishment mechanism is considered to be available for authentic communication among neighbors by one of the schemes proposed in [2, 13, 14, 15, 16]. Also, we assume to have a distributed revocation mechanism to revoke a detected captured node [17].

Below, we define the new terms we use throughout this work which are not defined previously in the literature or are defined differently than what we mean here.

Definition 1 Neighbors of a node is a set of nodes which are in one hop distance of that node and have common shared keys with it.

Definition 2 Node degree is the number of neighbors for a particular node.

Definition 3 Effective node degree is the number of legitimate neighbors for a particular node.

Definition 4 Independent adversaries refer to those adversary agents which try to capture legitimate nodes without any knowledge from other adversary agents' activities.

Definition 5 Colluding adversaries refer to those adversary agents which have common knowledge on the information gathered by each of them.

1.2 Communications and Adversary Models

Capturing a node by an adversary can happen in three stages. (*i*) The adversary has to discover a network node in order to physically gain access to it. The adversary at this stage can only attack the network by modifying the inputs to its target nodes. Despite the fact that this environment-manipulation attack is a severe attack by an adversary, it can be very expensive for the adversary to launch this attack. Because of the redundant observation assumption of an event of interest, the adversary has to modify the inputs of enough nodes to impose a significant impact on the network. We assume that the occurrence of such an attack is not very likely. However, we should note that in terms of countering this threat, nothing can be done at this stage because of the typical deployment of the network in a hostile environment. (*ii*) After the discovery of a target by the adversary, the adversary starts its capture process on the target node by disassembling it in order to get access to the target's internal state,

particularly its secrets. This accessing time to the target's internal state is a time consuming procedure for the adversary in order to complete its capture process. The amount of time it takes for the adversary to succeed the capture process is non-deterministic and depends on the physical node protection, the level of obfuscation, the environmental conditions of the sensors' installation site, and the skill and technology level of the adversary. During this stage, the target node would face some delay in its normal function. This delay and interruption of the normal functionality of a node under capture is implied as the adversary must disassemble the hardware of its target node in order to get to its internal states. The adversary's goal at this stage is to complete the capture process as soon as possible in order to be able to corrupt the communication in the network through the captured node. Using the continuous monitoring phenomenon, we require each node to engage with others to insure its survival. Therefore, the adversary on a captured node needs to follow this monitory protocol too. The adversary's goal here is to finish the capture process fast enough to be able to participate in the monitoring protocol. The *in-capture* phase, introduced earlier, is in charge of dealing with this stage. (*iii*) After completion of the capture process, the adversary gets access to the key materials of the captured node and can launch various attacks against the network. At this stage, the adversary may or may not launch an attack but it certainly has that capability to do so at will. Potential attacks at this stage include colluding with other captured nodes, control of the captured node's activities such as malicious aggregation of incoming traffic, multiple replications of the captured node and distributing them across the network [17], replacement of captured nodes in different neighborhoods, and running a Sybil attack [18]. The introduced *post-capture* detection phase is supposed to counter this

stage of the adversary's behavior, which is out of the scope of this research.

Below, we summarize the common assumptions for communications and adversary models in the literature addressing the captured node problem in WSNs. We adopt some of the following assumptions from [17] and we leave protocol-specific assumptions for later as necessary.

Assumption 4 The adversary has a universal communication presence in any part of the network at any time.

This assumption implies that an adversary can simultaneously send and receive an arbitrary number of messages in any part of the network at any time through the captured nodes.

Assumption 5 The adversary can perform selective but incremental node capture.

Assumption 6 The adversary can not block or significantly delay communications. We assume that an adversary is unable to jam or delay communications between legitimate nodes because of the underlying radio broadcast property.

This assumption stresses the distinction between the coercive and stealthy attackers. The focus of this work is particularly on the latter case for which the objective is to detect the presence of stealthy attackers presiding over captured nodes. However, addressing coercive attackers, e.g., jammers, is beyond the scope of this research mainly because their presence is obviously detectable

by the network. The challenge in these cases is how to counter against coercive attackers despite knowledge of their existence, rather than in the stealthy cases the challenge is whether there are any attackers or not.

Assumption 7 Neither the adversary nor the random failure of nodes can partition the network. So, we assume that the network is connected even with the presence of adversaries.

Assumption 8 The adversary can not manipulate the environment. This assumption excludes those adversaries that change the surrounding environment of sensor nodes in order to make them report false data.

Assumption 9 Collisions or transmission errors in communication are possible with a low probability.

Chapter 2

Related Work

In this chapter, we will briefly go over the previous works that are related to the node capture problem in WSNs. In this context, we categorize the proposed schemes in four different classes: spatial diversity, temporal diversity, replication detection, and secure aggregation.

Spatial Diversity. There is a couple of works done in spatial diversity in which the basic idea is to protect the secrets by distributing them among multiple modules hoping that compromising some, but not all, of the modules by adversaries would result in only partial exposures. The most known works based on this idea include threshold cryptographic schemes, all-or-nothing protection method, proactive secret sharing methods, remotely-keyed scheme, key-insulated scheme, and intrusion-resilient scheme. We will not go into further detail of these classes of methodologies since there are fundamental weaknesses that all of them suffer from. Some common prohibitive features of these schemes are as follows. Distribution is costly and not affordable for ordinary nodes in WSN made of low cost commodities. Also, there is no guaranteed

connectivity among multiple machines containing partial share of the secrets. This problem becomes clearer while encountering nodes having less degree than the required threshold. No matter what happens, they are incapable of successful activities due to shortages of partial shares.

Recently, a new scheme based on spatial diversity has been proposed using a threshold based method known as interleaved hop-by-hop authentication [19], which is capable of countering a fixed number of captured nodes. If supposedly they want to counter with $N-1$ number of captured nodes, they decompose the scheme into modulus N, grouping each set of N nodes in parallel branches such that the nodes are ordered sequentially in a route towards destination modulo N. Every node corresponds to the one on the same order in the consecutive modulus batches. This is basically to parallelize the system into N branches and place them in order sequentially. Therefore, if there are at most $N-1$ captured nodes having malicious behavior, after one cycle of N hops at least one legitimate node would not follow their conspiracy. Instead, that one legitimate node behaves inconsistently with them, resulting in the detection of any existing malicious activity in that cycle. Then at most after N hops penetrating bogus traffic into the network, this mal-functionality will be detected and therefore the required revocation procedure would be in charge of dealing with suspicious compromised nodes. Also there are variations to these schemes in which they split N group member nodes more sparsely [20]. This will cause more robustness to the tolerable number of captured nodes while letting deeper penetration of illegitimate traffic into the network. Also revocation would exclude the whole suspicious group, sacrificing more innocent nodes than the original version.

The problem of these resiliency approaches lie in the practicality of the choice of the threshold N. From one aspect, although N can theoretically be any number of nodes, it should be as large as possible in order to be able to overcome more captured nodes. In contrast, N can not be too large as it imposes more overhead into traffic. Also the larger the value of N is, the more penetration of the adversary and consequently the stronger the impact of attack against network operation. At minimum, this can result in a battery of nodes being exhausted in the route. So, typically the proposed number for N in literature is around 5-10. Our claim is that there is no reason that an adversary who is capable of compromising this number of nodes can not capture more than that. This implies that an adversary knows the threshold of breakage in security of these systems in advance and therefore would invest in overcoming this threshold to launch a successful attack against network operation. This is a common problem in most majority voting schemes with some fixed property in the underlying infrastructure in which deterministic criterion known *a priori* to everyone including adversaries let them defeat the threshold and break the system. Moreover, if captured nodes are located less than N hops away from the base station, there is ambiguity in detection and therefore the scheme is vulnerable in the vicinity of the base station, particularly within a partial cycle. This issue refers to the region which usually carries the most traffic load of the network and often is considered as the hot region of the network.

Two other works have been introduced recently [21, 22] using attestation techniques with different approaches to tackle the compromised node problem.

The authors in [21] propose a scheme to detect the trustworthiness of cluster heads based on the verification of cluster nodes through a Trusted Platform Module placed on each cluster head. This verification procedure can be either performed as a broadcast or unicast between cluster head and cluster nodes and determines whether or not the system integrity of the cluster head has been tampered with. The main drawback of this scheme comes from the underlying assumption that the adversary's goal is to compromise cluster heads. In contrast, it is quite rational for an adversary to look for the most vulnerable points in the network to launch its attack, which in this case would be cluster nodes, relay nodes, and the scheme is incapable of countering against it. On the other hand, [22] provides a software-based attestation scheme to verify the integrity of code running each node in a distributed manner through the interaction among its neighbors. Their proposed schemes are based on a pseudo-random noise generation mechanism and a lightweight block-based pseudo-random memory traversal algorithm. The scheme is essentially a self-organized checksum-based approach to detect any inconsistency between the expected memory content of the suspicious node and the actual memory content of some of its neighbors. Despite the usage of obfuscation by imposing pseudo-random memory traversal to harden data extraction for attackers upon a node capture, severely restricted memory space in sensor nodes does not provide enough flexibility for such maneuvering. In addition, the proposed scheme relies on the assumption of an available safe interval upon deployment during which nodes can exchange their secret shares. This assumption is simply violated by the scalability requirements of sensor networks just because once nodes are added during the lifetime of the network, a safe interval can no longer be available.

In the context of resiliency against node capture, there are a few recent works that try to introduce relevant metrics to assess security vulnerabilities in the presence of node capture. Patrick Tague, *et al.* in [23] propose an analysis toolkit to find the vulnerability points of the network which takes into account the joint effects of routing and security protocols. This is based on the introduced vulnerability metric and can be employed either by the designer or by the attacker, except the attacker needs to put in more effort to estimate some unknown parameters of the scheme. Also, node capture attack is formalized by virtue of the defined metric. Similarly, Di Pietro, *et al.* in [24] provide another metric for characterization of node capture.

Temporal Diversity. Now we want to look at some recent works using temporal diversity. The main approaches in this category include key evolution methods, forward and backward security mechanism, and tamper evidence method. As opposed to the previous case, a legitimate node has more control than adversaries as he decides when to do what procedures in order to make the whole system secured, to the extent that he needs. In addition, adversaries usually do not have enough flexibility at the time of capturing and launching their attacks. It is desirable from the adversaries' point of view to capture nodes as soon as possible right after deployment of WSN, but doing so is not feasible right away and usually takes some time for them to physically capture particular nodes or a particular fraction of network. Along this idea Anderson, *et al.* [25] proposed a scheme for maintaining system security over time which assumed that right after deployment full security has existed among nodes and they can simply establish pair-wise key in plain text for later us-

age. They justify this assumption by investigating the influence of potential attacks provided the realistic economical barriers of adversaries are present all across the network region upon deployment for sudden capturing and exposing keys. However, this assumption neglects the fact that it is highly likely for the adversary to capture very few nodes and remain on them for long term, launching malicious activities over time which can be disseminated across the network and have significant impact on network traffic. Another way to look at this approach is to assume the network is fully trusted initially [26] and try to preserve this trust relation over time. Although this seems to be a good choice, it conflicts with the scalability assumption required for WSN applications as this does not allow the addition of any new nodes to the network later on. In our methodology we consider the case in which the network is started with the initial trust granted to all nodes from the same central authority and we do our best to prevent its decay over the lifetime of network's operation as a result of an increase of the amount of threats from adversaries.

The basic idea behind key evolution methods is to perform more cryptographic operations being implemented securely using a single key. This is usually done by a random evolution procedure on the signer's state which leads to limiting the effects of compromising the current key. The Monotone Signature scheme [27] is an instantiation of such a methodology where some, but not all, secrets are revealed to the adversary under duress and the adversary can get valid signatures according to the current verification scheme. Here there is no question regarding detection of the adversary's presence as everyone is supposedly well aware of any victim nodes under duress. However, the issue is how to counter such a situation. As a countermeasure, the verifi-

cation algorithm in the recovery phase after attack would be updated in such a way that all signer's signatures before and after the update remain valid but the adversary's signatures become invalid in the updated version of the verification method. A variant of this approach is to keep the updated verification secret, known as Funkspiel [28]. Pedersen and Pfitzmann proposed the Fail-Stop signatures [29] in which none of signer's secrets are exposed to the adversary. Instead, each public key corresponds to a large number of valid private keys. The idea here is that the adversaries are assumed to be computationally powerful while legitimate nodes are relatively slow. Therefore, the adversary is capable of computing all the secret keys associated with a specific public key. However, she can not find out which one is used by a legitimate signer and the signer can repudiate the forged signature.

Bellare and Yee proposed a backward security scheme [30] in which they try to preserve previous system security even with exposure of current key materials to adversaries and save the security of old traffic. Most of the proposed research in this field is implemented based on the PKI cryptosystem where a series of public keys and their corresponding secret keys go through an update procedure [31]. Then only a single root public key needs to be certified and the rest can be evaluated from the root key using the update scheme. In the update procedure, first the new key is calculated and then the previous one is discarded after usage. In symmetric key cryptosystems, backward security can be achieved following these steps. Any shared key between two principals is usually passed through a one way hash function at several agreed upon times in order to update the new keys. These schemes are stateful in the sense that in each invocation of the update procedure, outputs are produced as a function

of the current state resulting in an updated state and the deletion of the old state. The initial state (seed) is generated through a probabilistic scheme such as a pseudo-random bit generator, which stretches a short seed into a longer sequence. Then a deterministic state update procedure is employed for key evolution. The operation of these schemes is divided into consecutive stages and update and deletion take place at the end of each stage. Thus if an adversary breaks into the system, she can only obtain the current state at any single point in time. And by using a one way update procedure, recovery of previous states or previous outputs would be infeasible due to their computational indistinguishability resulting from their random independent sequences. Interestingly, pseudo-random functions deployed here are only required to be secure against a very small number of queries which can be instantiated by block ciphers or hash functions. These methods fall into a bigger class of generators using iterated one way permutation based paradigms. The typical applications of such schemes can be used in various domains such as message authentication or secure audit logs. In the authentication scenario, key exposure makes previous authentic messages untrustworthy. So parties involved in the authentication procedure have to use the current key for both creation and verification of authentication tags in order for messages to correspond to the current stage. As a result, a break-in adversary having the current key is unable to forge the MAC generated by previous keys. Ordinary MAC protocols such as CBC-MAC, HMAC, and UMAC can be used for the authentication process. On the other hand, in the secure audit log application of backward security, the attacker tries to modify past log entries and erase the record of previous attempts at break-ins to change and improve its reputation history. From the administrator's perspective it is desirable to prevent such modifi-

cations and to create a suitable scenario for deploying a backward security feature. Also in order to prevent the re-ordering and deletion of previous messages one can simply use appropriate sequence numbers.

Anderson in [32] proposed another security feature along the same line, known as forward security, in which the goal is to prevent the exposure of future traffic by compromising current keys. This can be achieved by various techniques such as regular re-keying, auto re-keying, and offline storage of all future keys. Particularly the auto re-keying methods are studied in symmetric cryptosystems in which shared keys between pairs are hashed at agreed times with all exchanged messages since the last key change. This leads to a security recovery in the event the attacker misses only a single message exchanged between the two parties. Although this seems like a promising approach, it can easily backfire since the synchronicity between parties are very fragile and it might unintentionally hurt the legitimate interaction between the parties for any slight asynchrony.

Another work recently proposed in this context is tamper evidence [33], whose goal is to provide security after the full exposure of key materials to the inconspicuous adversary. It promises to detect an adversary's presence in the system even after all secrets are exposed to the adversary. The way it works is based on a stateless scheme in which two signatures are given to it as inputs and it detects an inconsistency between these two signatures if they are not issued by the same identity, either a legitimate insider or an adversarial insider. The reason is that if the two signatures are derived from the same identity, based on the construction of private keys one of these two key strings should

contain the other, i.e., one of them is the subset of the other. If two different identities have issued them, they do not have such a relation. Therefore the verifier, which can be a suspicious node's neighbor, would take two instances of signatures from the suspicious node and check for a possible containment relation. If it does not hold, then the verifier detects the divergence between the two given signatures. The idea is to use a random evolution mechanism of two versions of signatures associated with each identity, i.e., legitimate signer and forger. The scheme also allows some cases of tampering to be missed and depending on how long the time differences between the two given signatures are, various modifications of the scheme give a different security level of tamper detection with their corresponding costs. In addition to the PKI based cryptosystem's design, the tamper evidence mechanism suffers from some other shortcomings as follows. As the adversary is considered to have full control over all inputs and outputs of the victim node, she can choose the time of the two signatures issued for the verifier. Therefore in a new attack, which was not previously considered, the adversary can simply pass two forged signatures with proper evolution which is consistent with the verification procedure and can pass it successfully. This attack scenario is likely for adversaries who stay on the victim node over a period of time and, unfortunately, the proposed method is incapable of countering it. As long as the adversary remains on the victim node, the verification procedure would fail until the legitimate signer has the opportunity to cast his signature to the verifier. A more suitable case seems to be when a stealthy adversary captures a node, exposes its keys and leaves that victim node, possibly replicating that identity somewhere else. Through this simultaneous challenge between legitimate signer and adversary, the verification could be successful provided that these two signature versions

are received by the same verifier, which is an unrealistic scenario.

Replication Detection. Recently there is some other research proposed to encounter captured node problems from different viewpoints [34, 35]. In [34] the authors propose a scheme for detection and recovery of node cloning attacks. Cloned nodes are multiple replications of a legitimate node captured by an adversary and distributed in the network. Their scheme is based on usage statistics of nodes' keys. The drawbacks of this scheme include the central detection approach, revocation of the keys and not necessarily revoking the cloned nodes, and its inappropriate assumption that the distribution of key usage is a direct consequence of the number of cloned nodes. The other research in this context is replication detection [35] in which the scheme finds the presence of replicated nodes in the network in a distributed fashion. This scheme has the ability to find collisions of multiple copies of the same identity somewhere in the network by some random observer nodes and propagate collision flags throughout the network. Although this scheme works properly for replication detection scenarios, it can not handle the detection of each captured node individually.

H. Fu, *et al.* [36] follow the research on replication attacks. The main focus of their work is on the resiliency analysis of the well-known random key pre-distribution schemes against replication attacks. In their work, they proposed three different metrics to capture the notion of resiliency for analyzing the security of wireless sensor networks in the presence of replicated nodes. The soundness of their proposed metrics are highly under question for the following reasons. (i) They impose the assumption that in their approach the adversary

somehow has the control over the structure of the hardware of a replicated node. This assumption is not true in general, and at most the attacker may copy the malicious code into the replicated node's memory, but cannot change the size of its memory. (*ii*) The impact of a replicated node on the security of the network across the network terrain is not uniform, which is considered in contrary here. (*iii*) The evaluation of the metrics of interest are quite costly.

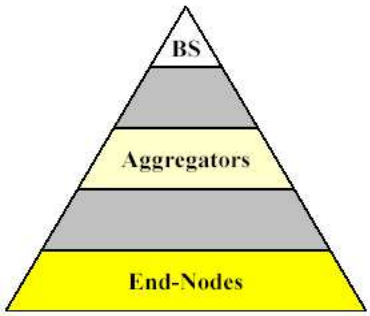


Figure 2.1: Sensor nodes can have various functionalities such as end-nodes, aggregators, or base station.

Secure aggregation. In numerous research, secure aggregation has been addressed ranging from protocol designs [37, 38] to application development [39] and mathematical foundations [40]. In the Secure Information Aggregation method (SIA) [37], there are three types of nodes considered in WSN; end-nodes, aggregators, and base station as is shown in Fig. 2.1. Aggregators are in charge of local processing within the network and the design requires sub-linear communications between the aggregator and base station in order to aggregate the measurements and pass the result and also guarantee that reported result is valid and not corrupted by the aggregator. This is done through a three phase process namely as aggregate-commitment-proof. In the

aggregation phase, the authenticated inputs are combined by the desired aggregation function with the result being forwarded to the base station. In the Commitment phase, inputs define a binary Merkle hash tree [41] in which the root is a commitment to all participating inputs and the authentic root is sent to the base station. Whenever the base station wants to verify the authenticity of any input, it may require the aggregator to forward that input with the corresponding inputs in the tree authenticated by end-nodes and then the base station confirms the root by making a dual Merkle tree to obtain a dual root. In the proof phase, the aggregator and base station are engaged in an efficient interactive proof process in order to check the validity of the result reported by the aggregator. Private shared keys are required between each end-node/aggregator pair and end-node/base station pair which lead to guaranteeing the authenticity of reading from end-nodes in the aggregators. Also using an efficient interactive proof between each aggregator and base station guarantees the legitimate activity of the aggregator deriving results and reporting to the base station. However, there is no way to check the validity of reading in the compromised end-nodes as adversaries have access to all keying information in end-nodes upon capturing them.

As mentioned above, aggregators are the main part of their design; however, there are no arguments about their placement and also their capabilities. It is assumed that they are the same as end-nodes which is true based on a flat network scenario, but there are a couple of issues with the design. (i) Putting particular function into specific nodes, i.e., fixed aggregators, causes them to be exhausted quickly and also increases their vulnerabilities as their additional roles make them more critical targets from the adversaries' point of

view. (*ii*) There are issues about connectivity to aggregators. From one end, there should be as many aggregators as possible so that end nodes can reach them with the least number of intermediate hops (relays) in order to reduce communication overheads and get a better aggregation factor. This implies that there should be more aggregators placed in lower levels of the routing tree, i.e. closer to end-nodes. However from the other end, it is desirable to have fewer aggregators in higher levels of the routing tree, i.e., towards the base station, where the traffic is more dense and fewer intermediate hops (relays) are needed to connect the aggregators with the base station. This is crucial as the authentication and interactive proof schemes between the aggregator and the base station assumes a trusted path between them. Otherwise, it is easy for adversaries to be in the middle of the two and to disrupt the security interaction between them in order to cause a failure of legitimate interactions, especially through colluding adversaries placed before and after the aggregators.

Furthermore, each end-node needs to have a shared key with the corresponding aggregators in the potential route towards the destination, i.e. base station. But because of the ad-hoc nature of WSN, there is no infrastructure in the network and the placement of nodes prior to deployment is random. Therefore, there is no guarantee which nodes should act as aggregators and also with which aggregators, nodes should keep their corresponding shared keys, which imposes inefficient overheads in trusted path key establishments. These arguments suggest considering that aggregation be done through the WSN randomly over any nodes rather than through fixed aggregators with battery exhaustion and critical target vulnerabilities. However doing so on

every hop basis may not gain a significant aggregation factor. Therefore by using some predefined counter, aggregation can be made once every couple of hops in each route. This parameter should optimally be evaluated in run time or prior to deployment for specific traffic and topology. It is more likely to get a better estimate of that in run time; however, we should notice that it needs to be evaluated in a distributed manner. As another choice of when to do aggregation, one may choose to aggregate on every hop basis, provided that there is something to aggregate. Otherwise wait until there is enough inputs required for aggregation (at least two inputs) and then evaluate and pass the aggregation result to the next hop.

Yi Yang, *et al.* in [42] follow the SIA protocol proposed in [37] to deal with secure aggregation. The proposed scheme consists of two steps, namely divide-and-conquer and commit-and-attest and overcomes the shortcomings of SIA from the following aspects. (*i*) Using a probabilistic approach for choosing aggregators prevents adversaries to reveal them in advance. This results in the elimination of targeting aggregators *a priori*. (*ii*) The scheme uses an in-network aggregation approach, which leads to a higher aggregation ratio and lower energy consumption compared to the ones for SIA. (*iii*) The scheme provides a weighted aggregation feature among all subtrees. However, the scheme has its own drawbacks including its reliance on the redundant observability assumption of the event of interest and lack of defense against bogus data injection through captured leaf nodes. It may be quite possible that the aggregation values corresponding to some rare events are mistakenly considered illegitimate merely because they are not supported globally and are only partially observed in some subtrees of the network. In addition, aggregation

queries are initiated through base stations rather than as an event driven process by network nodes.

On the other hand, secure aggregation in [38] proposes a method to aggregate gradually while going in the route towards the base station. Therefore there is no particular aggregator node which aggregates all local inputs. Instead, each node aggregates whatever it gets and passes it on to the next hop towards base station. The key idea in their design is to use the μTESLA key chain [1] and disclose keys with some delay so that after the disclosure of the keys, no one can use those keys for authentication but the verifier, i.e., the base station, which can check the messages authenticated by nodes using those keys previously. This disclosure delay causes transfer and storage of whole aggregated messages during this delay interval until the base station finishes the verification phase. Also the verification process is not an efficient scheme and it is initiated by the base station and goes downward until the end-nodes and comes back to it which is longer path than being originated by some neighborhood entities closer to end-nodes which are information sources. This suggests eliminating the corrupted data from network traffic as early as possible and preventing penetration of the bogus traffic based on local decisions made in the closest vicinity of its origin. The problem with their security mechanism is that all aggregation results go all the way up to the base station and remain there until the base station authenticates the messages downstream or else it discards them. So, if there are illegitimate messages, there is no reason for them to be carried all the way up to the base station and discarded after a while since it imposes significant communication overheads, which is in contrast to the notion of aggregation.

Besides, there is a spatial delay in aggregation nodes, meaning that inputs will not be aggregated in their immediate parent but in their immediate grandparent. This delay in aggregation placement causes more overhead in exchange for obtaining resiliency against a single node being captured by an adversary. Although this can be a valuable feature for the price of extra overhead to their scheme, there is no reason that the underlying sensor network can not face with additional captured nodes. As discussed earlier, for the same reason that an adversary can compromise one node, she would be able to capture as many nodes as necessary in order to disrupt network functionality or mislead it by corrupting inputs leading to unintentional behavior of the network. Unfortunately, this scheme is unable to counter such attack scenarios.

Chapter 3

Proposed Pinging Methods for *in-capture* Detection

Dealing with the proposed problems in Chapter 1, we use the following three facts which have a significant impact on our proposed methods to the well-known captured node problem in WSNs.

Fact 1 *Capturing a target node by an adversary in order to get the target's internal states is a physically time consuming process. This process requires a nonzero interval of time for an adversary to complete its capture process.*

Fact 2 *During the capture process of a target node, the target node will be unable to function properly due to its hardware disassembly by the adversary. This operational interruption upon hardware manipulation is governed by the current hardware technology used in sensor nodes.*

Fact 3 *Any solution with a fixed resiliency threshold is incapable of countering the captured node problem in WSNs.*

The schemes proposed previously regarding the captured node problem are typically capable of overcoming the problem up to a certain fixed number of captured nodes. This *resiliency threshold* can not be too high because of the prohibitive overhead introduced by the underlying schemes. Therefore, having a threshold of the order of five to at most ten captured nodes resiliency would not be attainable because of the following reasons. (*i*) For the same reason that an adversary is capable of capturing up to the threshold number of nodes, it can exceed that threshold and eventually break the resiliency of the scheme. (*ii*) The threshold is a pre-defined deterministic one and even though the adversary may not be aware of it in advance, it can gradually increase the number of captured nodes until the scheme is broken. This seems to suggest that it is better to prevent an adversary's success in capturing any target nodes as much as possible. Here we try to stop an adversary's progress in capturing nodes by detecting its presence on target nodes during the capture process and revoking those nodes under capture.

As mentioned earlier, the adversary's goal in capturing nodes is to influence the legitimate traffic of the network as much as possible so that it can deceive the base station or deplete the limited resources of existing nodes encountering bogus traffic. This can be done by modifying ongoing traffic or by injecting illegitimate data through captured nodes. Transferred messages should be authentic in the receiver or else they will be discarded. In order to authenticate messages, the sender needs to have valid key materials, thus the adversary can get them by means of capturing legitimate nodes already participating on the network. Therefore once the node is captured, the adversary gains access to the internal state of the node and keys are revealed.

After the capture phase is complete, any traffic from that node would not be distinguishable from the legitimate traffic from receiver's point of view. However, we can restrict the impact of the adversary's presence at the captured nodes during the capture phase. This approach would be feasible by using Fact 1, i.e., the capture phase which refers to the duration that a capture is in progress, which requires the physical touch and presence of an adversary on the participating nodes targeted for being captured. For an adversary, it takes some time in order to eventually get into the internal memory of any node and during this capture phase there is a discontinuity in the normal operational behavior of the target node. If we assume this time consuming procedure of capture completion is comparable with the execution period of our proposed mechanisms for captured node detection, we can detect when the node is under the capture phase before the adversary succeeds in capturing the node and starts injecting bogus data on to the network. We refer to these approaches as the *in-capture* detection mechanism. In the following sections we will discuss our two proposed approaches, the *pairwise pinging scheme* and the *quorum pinging scheme* in details in order to achieve this goal. The summary of the notation used throughout this work is given in Table 3.1.

3.1 Pairwise Pinging Scheme

In this scheme each node is watched by its neighbors. Time is divided into consecutive epochs. In each epoch, we define K different time intervals which are assigned for the operation of each node based on its identification (ID). A possible assignment policy can be as follows:

Parameter	Description
k_{ij}	Shared key between nodes i and j
$H(k_{ij};.)$	Keyed Hash function of a message using key k_{ij}
K	Number of communication intervals in each epoch
`Interval_no(i)`	Index of the interval corresponding to node i
T_p	Duration of an interval assigned for each node's communication
T_e	Duration of each epoch containing all slotted communication intervals
T	Duration of detection interval
n	Index of the corresponding epoch
M	Number of states or number of epochs in each detection interval
X	Random variable for the time an adversary can complete a capture
$f_X(x)$	The *pdf* of X
η_x	Average of X
d	Average node degree
d_{eff}	Effective node degree
p_r	Probability of pinging a node in the assigned interval
p_l	Probability of packet loss
p_e	Probability of successful reception of a ping response message
P_s	Probability of being at state s
N_s	Number of times a node is in state s
T_s	Random variable for the residual time-to-false-alarm given in state s
$T^*_{(i)}$	Arrival time of the i^{th} false alarm
q	Quorum size
P_m	Probability of a missed detection for pairwise scheme
$P_m^{(q)}$	Probability of a missed detection for *q-node* quorum scheme
L_f	Expected residual time-to-false-alarm for pairwise scheme
$L_f^{(q)}$	Expected residual time-to-false-alarm for *q-node* quorum scheme
$P_{FA}^{(q)}$	Probability of false alarm for *q-node* quorum scheme
N_A	Total number of deployed adversary agents
$n_A^{(j)}$	Number of collaborative agents associated with j^{th} adversary agent
A_S	Number of successful adversary agents in capturing their targets
d_{eff}^j	Effective node degree of a node targeted by j^{th} adversary agent

Table 3.1: List of used notations

$$\texttt{Interval\_no(i)} = [i \ mod(K-1)] + 1 \qquad 1 \leq \texttt{Interval\_no(i)} \leq K-1 \tag{3.1}$$

Each node with its particular ID, (e.g., i) can send out its ping messages only during its corresponding time interval, however, the response messages can be heard at any time interval. These messages include a nonce and are authenticated by the pairwise key k_{ij} shared between neighbors i and j using

a keyed hash function $H(k_{ij};.)$. Note that $k_{ij} = k_{ji}$ in our symmetric setting. The nonce is a random number chosen by the sender. Once a neighbor sends its ping message at its corresponding time interval to the watched node, the watched node has to respond in the next time interval. In order to check the successful reception and validity of this exchange, the watched node is required to increment the nonce in the ping message and authenticate it again in the response message with their common shared key, then respond back to its neighbor. The structure of the ping and response messages are as follows:

$$\begin{aligned} &\text{ping message:} \quad i \rightarrow j \qquad i, j, (nonce), H(k_{ij}; nonce) \\ &\text{response message:} \quad j \rightarrow i \qquad j, i, (nonce + 1), H(k_{ji}; nonce + 1) \end{aligned} \tag{3.2}$$

If the watched node misses this acknowledgement, it will be flagged as a *suspicious node* by its neighbor who initiated the exchange. The acknowledgement failures include cases in which a response is not received during the next time interval, the response is received with inappropriate message content, or the messages fail to be delivered due to unreliable wireless communication. These events would lead to a raised a flag against the watched node, and depending on the deployed revocation policy, the required procedure for making a decision regarding that node would be executed.

Obviously, this exchange among each pair of neighboring nodes imposes a certain overhead to the network performance, both in communications and computations. We modify the scheme in such a way that the neighbors initiate the pinging exchange in a random fashion with a specific probability. The

intuition behind this modification is as follows. (i) By imposing randomness to the initiation of the exchange, the following property holds: the invoking of the exchange procedure by the adversary at an inappropriate time can penalize the adversary and can be used as a sign for the presence of the adversary on a node. (ii) By not pinging the watched node in every epoch, the communication and computation costs of the proposed scheme is reduced.

Now, if an adversary starts capturing a node, the node can not respond to a potential ping message initiated by at least one of its neighbors. As we discussed earlier, the engaging of the node by the adversary into a capture phase causes an interruption in the normal operation of that node. Given that there is a neighbor of the node that initiated the ping message, missing the opportunity for the node to respond to its neighbor leads to a flag raised by the neighbor against the node.

Though, to account for cases in which either the ping message or its response may be lost due to communication errors, the neighbor should not immediately declare the watched node as captured upon not receiving the response message. Instead, we set a detection interval during which if the response is missed for a fixed number of times, then the neighbor can raise a flag against the watched node. Following this procedure, in order to have a successful capture, the adversary needs to complete the capture phase of the target node within the detection interval.

The above argument suggests that the pinging rate be increased in order to increase the rate of the detection of the presence of an adversary on

a target node by at least one of the target node's neighbors. However, this results in higher communication and computation costs to the network which may become a prohibitive factor for using the pinging scheme.

Besides, changing the length of the detection interval is an optimization problem of security versus performance. By increasing the detection interval, more opportunity is given to the adversary to complete its capture process of a target node, while reducing the detection interval leads to the increase of the probability of raising a flag against an innocent watched node by mistake, resulting in a false alarm. Hence, we present a mathematical model in the following section as a tool to set proper values for the aforementioned parameters. The parameters of the model can be selected in such a way that the pinging scheme provides the desired security and performance measures such as the probability of a missed detection and the expected lifetime of a node for a given adversary. Our objective of invoking the pinging scheme is to achieve a certain performance of the proposed mechanism with acceptable costs, namely minimizing the probability of a missed detection and maximizing the expected node and network lifetime.

3.1.1 Modeling of the Pairwise Pinging Scheme

In this section, we present a model based on discrete event formulation [43, 44] for the proposed scheme in which we are interested in evaluating the probability of detecting a target node under the capture phase by one of its neighbors. Each node, potentially a node targeted for being captured by the adversary, is considered to have d neighbors and each of the neighbors with certain prob-

ability p_r decides to initiate the ping message for the target node. A sensor node may fail to respond to a ping triggered by one of its neighbors for different reasons, such as; being under capture phase, its battery being depleted, or even being physically damaged. The ping message or its response can be lost due to collision or congestion during communication. We assume a ping message in the assigned interval or its response in the next interval can be lost with probability p_l.

To avoid being flagged as captured, a target node has to respond to the ping message upon receiving it from any of the neighbors. By appropriate choices of the parameters in normal mode of operation, each node would be able to respond the acknowledgement message within the next time interval. However, in case of capture as discussed in Section 1.2, the target node would face some delay in its functionality due to physical access time to the internal states, memory content, and particularly its keys by the adversary. The access time procedure is not deterministic. The time it takes depends on the physical node protection, level of obfuscation, environmental conditions of the sensors' installation site, and the skill and technology level of the adversary. Therefore, we use a random variable X with probability distribution function (pdf) $f_X(x)$ to model the necessary time for the adversary to complete the capture of a sensor node. This includes the search time for sensors' components, disassembling them, and the accessing time to their internal states. We assume η_x denotes the expected value of random variable X. In practical situations, η_x varies from several minutes to several tens of minutes. We can achieve $f_X(x)$ *a priori* by invoking an offline statistical procedure for fitting a theoretical distribution. The proposed theoretical distribution should be matched with

the histogram of the actual data of the corresponding variable gathered by the offline experiments.

The other parameter of interest in mathematical formulation is the timing of receiving a ping message by the target node. Assume the time is slotted into epochs of length T_e, and each epoch of length T_e is slotted into K intervals, each with length T_p, i.e., $T_e = KT_p$. One of the intervals of length T_p is assigned to each of the nodes, during which time it decides whether or not to ping its neighbor. Fig. 3.1 shows the placement of intervals in each epoch and their assignments to each node based on their index. A sensor node pings its neighbor with probability p_r during each assigned interval. Randomization of ping messages is necessary in order to make them unpredictable for the adversary. After pinging, the node that initiated the pinging waits one interval T_p for a response from its watched neighbor. As long as a watched node responds to its ping messages in the next assigned intervals, its neighbors assess it as a legitimate node. A small deviation can be accepted to account for possible collisions or transmission errors in the physical communication layer, however, a long idle interval causes the neighbor of the watched node to consider the watched node as a captured node.

Assumption 10 The pinging scheme for each pair of neighbors is performed in a synchronized manner, i.e., *pairwise synchronicity*.

In order for neighbors to follow the scheme and communicate with each other through ping messages and responses, we assume that neighbors are synchronous. Here, we can relax this to a loose synchronization assumption by picking an appropriate value for T_p. T_p has to account for round trip delay

and processing time of ping message and its response. By over provisioning the value of T_p appropriately, the scheme can tolerate some asynchrony. Note that the synchronicity assumption is only for the execution of the pinging scheme and it does not apply to communication pattern of network traffic.

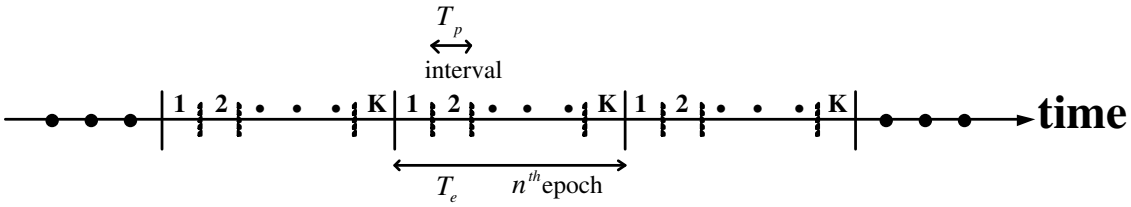


Figure 3.1: Time intervals in each epoch.

We assume node i pings node j with probability p_r in one of the intervals of length T_p indexed by `Interval_no(i)`. After sending the ping message, node i waits for one time interval T_p for the response from node j. Node j is marked *captured* by node i if node i does not receive any response from node j within a long detection interval T. Obviously, $T_e < T$ and hence $T_p \ll T$. In order to find different properties of this detector, we define the following measures:

Definition 6 False Alarm. A false alarm [45] is an event in which a node that is not captured is classified as captured.

A false alarm can happen for two reasons. First, the random sending of ping messages with probability p_r can happen in such a way that no ping message is sent from node i to node j in a detection interval of length T. The second cause of a false alarm is when the ping message or its response is lost with probability p_l due to the natural transmission errors in the physical layer during a detection interval. We assume L_f denotes the expected value

of the lifetime of a node before a false alarm is generated or in other words, the expected residual time-to-false-alarm. The goal is to maximize L_f.

Definition 7 Missed Detection. A missed detection is an event in which the adversary successfully captures a node without being detected.

A missed detection can happen if the adversary performs the capture procedure so fast that the adversary completes capture of the target node before the detection interval T elapses. We assume P_m denotes the probability of a missed detection or simply the probability of miss. The goal is to minimize P_m.

Given the above information, it is desirable to find the expected residual time-to-false-alarm L_f, and the probability of a missed detection P_m for the introduced *in-capture* detection mechanism using pairwise pinging scheme.

Note that in order to detect a node capture, node i should not receive any response from node j for at most $M = T/T_e$ consecutive epochs. Therefore, in order to model the system, we consider a *Markov Chain* with a discrete state S_n to represent the state of suspicion of a neighbor i about a target node j at epoch n, and thus have $0 \leq S_n \leq M$. The state variable is set to M and is decremented in each epoch of length T_e at the corresponding node's interval. In the n^{th} epoch if node i receives a ping response from node j, the state is reset to M. If it does not receive such a response, the state is decremented. Once the state reaches 0, the neighbor declares node j as a captured node. The smaller the state is, the less time remains for an adversary to complete its capture process. It would be to the adversary's benefit to be able to respond

to an initiated ping message as soon as possible, provided that the adversary could complete its capture on a target node and would be able to generate a valid response message using the compromised keys of the target node. If the adversary has not successfully captured the node within the next time interval of an initiated ping message, the corresponding state of the target node, which is kept in the neighbor, is decremented once in every epoch due to the fact that a node under the capture phase is unable to respond to a ping message immediately.

We use the above *Markov Chain* model to analyze the system given that the node is not captured (in a nonzero state) and the system is in a steady state. Because of the probabilistic pinging rate p_r and transmission errors with probability p_l, the state of the system at any time instant can be a value between 1 and M. The value of the state depends both on the number of successive epochs that node i decides not to ping node j and on the number of successive epochs that node i has not received any responses after pinging node j or any mixture of these two events. The state value defines the remaining time for the adversary to complete its capture process before the detection interval elapses. Consequently, we can investigate the behavior of the system once the adversary starts the capture procedure at any time instant.

Given the above model for the state of the system, we need to define transition probabilities of states under the assumption that the adversary has not started a node capture. Let $p(S_n, S_{n+1})$ represent the transition probability from state S_n at epoch n to state S_{n+1} at epoch $n + 1$.

Suppose p_e represents the probability that node i receives a response from node j during the n^{th} epoch given node j is not attacked. In order to find the transition probabilities, we need to find p_e. Node i receives a ping response from node j if two events happen: (i) node i sends a ping message to node j at the interval indexed by `Interval_no(i)` of the n^{th} epoch, and (ii) neither the ping message nor its response is lost due to communication error. Since the two events are independent, the probability of receiving a ping response at the n^{th} epoch can be written as the product of the probabilities of each of the events:

$$p_e = p_r(1 - p_l) \tag{3.3}$$

in which the first term is the probability of sending a ping message and the second term is the probability of successful communications between nodes i and j in both directions. Note that given the capture procedure is not in progress, whenever a node with state S_n, $1 \leq S_n \leq M$ receives a ping response, the state can only have either a transition to state $S_{n+1} = M$ with probability $p(S_n, M) = p_e$, or a transition to state $S_{n+1} = S_n - 1$ with probability $p(S_n, S_n - 1) = 1 - p_e$. All other transitions happen with probability 0. Therefore, the transition probability of the states can be written as:

$$\begin{cases} p(0,0) = 1 \\ p(0,S) = 0 \qquad 1 \leq S \leq M \\ p(S,M) = p_e \qquad 1 \leq S \leq M \\ p(S,S-1) = 1 - p_e \qquad 1 \leq S \leq M \\ p(S,S') = 0 \qquad 1 \leq S \leq M, \quad S' \neq S-1, \quad S' \neq M \end{cases} \tag{3.4}$$

The discussed *Markov Chain* and its transition probabilities are shown in Fig. 3.2.

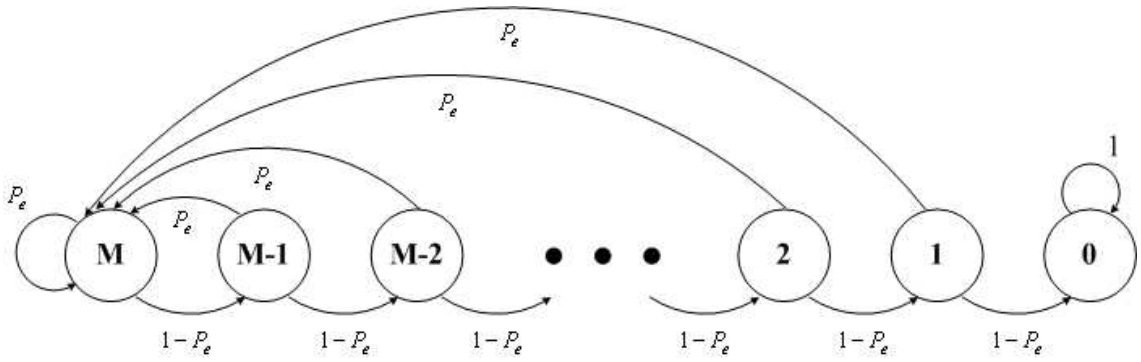


Figure 3.2: *Markov Chain.*

In the next step, we find the steady state probabilities of presence at each of the states in the *Markov Chain*. Assume node j is not being captured or identified as a captured node. We define P_s as the steady state probability of being at state s given $s \neq 0$. From a practical point of view, P_s means the probability that $S_n = s$ if we observe the state of *Markov Chain* at a randomly chosen time epoch n. In order to find these probabilities, we consider a significantly large time duration T_L, such that $T_L = NT_e$, and $M \ll N$. During the time duration T_L, the *Markov* takes N states. Assume N_s represents the number of times out of N that the *Markov Chain* is in state s. For a very large N by using SLLN [1], we can write the following equations:

[1]SLLN refers to the Strong Law of Large Number.

$$
\begin{aligned}
&E[N_s] = (1-p_e)E[N_{s+1}] \qquad 1 \leq s \leq M-1 \\
&E[N_M] = \textstyle\sum_{s=2}^{M} p_e E[N_s] + E[N_1] \\
&\textstyle\sum_{s=1}^{M} E[N_s] = N
\end{aligned}
\tag{3.5}
$$

The first equation above is due to the fact that every time that we are in the state $s+1$, with probability $(1-p_e)$, we have a transition to the state s, and since there is no other way for a transition to state s, we have $E[N_s] = (1-p_e)E[N_{s+1}]$. The second equation uses a similar notion for state M by considering the fact that since it is given that the chain has not transitioned to state $s = 0$, a transition from state 1 to state M happens with probability 1. The third equation is due to the fact that the sum of all the values of N_s for all $s \neq 0$ has to be N. If we divide both sides of the above three equations by N, we find:

$$
\begin{aligned}
&\tfrac{E[N_s]}{N} = (1-p_e)\tfrac{E[N_{s+1}]}{N} \qquad 1 \leq s \leq M-1 \\
&\tfrac{E[N_M]}{N} = \textstyle\sum_{s=2}^{M} p_e \tfrac{E[N_s]}{N} + \tfrac{E[N_1]}{N} \\
&\textstyle\sum_{s=1}^{M} \tfrac{E[N_s]}{N} = 1
\end{aligned}
\tag{3.6}
$$

Now, we use the fact that $\frac{E[N_s]}{N}$ is the steady state probability of being at state s in a random time epoch n. Therefore:

$$
\begin{aligned}
&P_s = (1-p_e)P_{s+1} \qquad 1 \leq s \leq M-1 \\
&P_M = \textstyle\sum_{s=2}^{M} p_e P_s + P_1 \\
&\textstyle\sum_{s=1}^{M} P_s = 1
\end{aligned}
\tag{3.7}
$$

The above is a linear system of equations that can be easily solved to

find the value of P_s for every state s. The solution can be found by noting:

$$P_s = P_M(1 - p_e)^{M-s} \tag{3.8}$$

The above can be found by iteratively using the first equation of Eqs. 3.7. By substituting this value in the second and the third equations of Eqs. 3.7, we find:

$$P_s = \frac{p_e(1 - p_e)^{M-s}}{1 - (1 - p_e)^M} \qquad 1 \leq s \leq M \tag{3.9}$$

A. Probability of a Missed Detection. Now, we use the steady state probabilities in order to find the probability of a missed detection. Assume the adversary starts the capture process at time epoch n. The *Markov Chain* can be in any of the states except state 0 when the adversary starts capturing the node. These state values may be caused by possible state decrements in preceding epochs either by not pinging or by messages being lost during pinging conversations. The probability of success for the adversary depends on the value of the state at time epoch n. If the *Markov Chain* is at state s and at time 0, then the adversary needs to finish capturing in sT_e. In other words, at state s and at time 0, node i has not received any response from node j for $(M - s)T_e$, either because it has not pinged node j or has not received any responses for that many epochs due to packet loss. Therefore, the adversary has only sT_e time left before the capture of node j is detected by the neighbor i. We can summarize this by the following lemma.

Lemma 1 *Given in state s, the necessary capture time for an adversary, X should be less than or equal to the remaining time in the detection interval,*

sT_e for a successful capture by an adversary.

Proof: In the *Markov* modeling of the scheme, reception of a ping response corresponds to regenerative points in which the system goes back to state M. Obviously $X \leq T$ in order to prevent the elapsing of the detection interval. If after the last regenerative point, the adversary waits for X^* and then starts its capture process, it takes the adversary X to complete its capture. So, in order to get a successful capture, $(X^* + X) \leq T$. This implies that given in state s after X^*, the adversary has at most sT_e to complete its task, i.e., given that in state s, $X \leq sT_e$.

Hence, the probability of miss given $S_n = s$ is:

$$p(miss|S_n = s) = P(X < sT_e) \tag{3.10}$$

Recall that X is the random variable that describes the necessary capture time for the adversary. The right hand side of the above equation can be written in terms of the cumulative density function of X:

$$p(miss|S_n = s) = F_X(sT_e) \tag{3.11}$$

in which:

$$F_X(x) = \int_0^x f_X(\tau)d\tau \tag{3.12}$$

The probability of miss can be written as a conditional sum that takes into account the probability of being at all states:

$$P_m = \sum_{s=1}^{M} p(miss|S_n = s)p(S_n = s) = \sum_{s=1}^{M} p(miss|S_n = s)P_s \tag{3.13}$$

By substituting values of P_s from Eq. 3.9 and $p(miss|S_n = s)$ from Eq. 3.11 in Eq. 3.13 we find:

$$P_m = \sum_{s=1}^{M} \frac{p_e(1-p_e)^{M-s}}{1-(1-p_e)^M} F_X(sT_e) \tag{3.14}$$

Note that the probability of a detection P_d which represents the probability of detecting a captured node is simply $P_d = 1 - P_m$.

B. Expected Residual Time-to-false-alarm. In the next step of our analysis, we find the expected value of the lifetime of a node given no attack happens. This measure gives the quality of our proposed mechanism as a level of its robustness to false alarms. Assume T_s represents the expected value of the time it takes for the transition to state 0 given we are at state s at time 0. We have $T_0 = 0$ and:

$$T_1 = T_e(1-p_e) + p_e(T_e + T_M) = T_e + p_e T_M \tag{3.15}$$

We arrive at the above equation because if we are at state $s = 1$, then it takes one time epoch T_e with probability $1 - p_e$ to transition to state 0, and with probability p_e to transition to state M, in which case we need to wait an average of T_M until we have a transition to state 0. If we write the similar equations for all the states, we find:

$$
\begin{aligned}
T_1 &= T_e(1-p_e) + p_e(T_e + T_M) = T_e + p_e T_M \\
T_s &= (T_e + T_{s-1})(1-p_e) + p_e(T_e + T_M) = T_e + (1-p_e)T_{s-1} + p_e T_M \qquad 2 \le s \le M-1 \\
T_M &= T_e + (1-p_e)T_{M-1} + p_e T_M
\end{aligned}
\tag{3.16}
$$

Which gives us a linear set of M equations and M unknown parameters.

$$
\begin{bmatrix}
1 & 0 & 0 & \dots & 0 & -p_e \\
-(1-p_e) & 1 & 0 & \dots & 0 & -p_e \\
0 & -(1-p_e) & 1 & \dots & 0 & -p_e \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & 1 & -p_e \\
0 & 0 & 0 & \dots & -(1-p_e) & 1-p_e
\end{bmatrix}
\begin{bmatrix}
T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_{M-1} \\ T_M
\end{bmatrix}
=
\begin{bmatrix}
T_e \\ T_e \\ T_e \\ \vdots \\ T_e \\ T_e
\end{bmatrix}
\tag{3.17}
$$

Assume L_f denotes the expected value of the time before a false alarm is generated. We call L_f the expected residual time-to-false-alarm and it can be written as:

$$
L_f = \sum_{s=1}^{M} T_s P_s \tag{3.18}
$$

By substituting values of P_s from Eq. 3.9 and T_s from Eq. 3.17 in Eq. 3.18, we find:

$$
L_f = \sum_{s=1}^{M} \frac{p_e(1-p_e)^{M-s}}{1-(1-p_e)^M} T_s \tag{3.19}
$$

Remark 1 The reason for defining residual time-to-false-alarm as a measure of describing false alarm is because of the special structure of the *Markov Chain*. As state 0 is an absorbing state with nonzero transition probability, in the long-run each state has nonzero probability to go to state 0 and remain there. Eventually no matter where the initial state is, the final state would be 0, which can cause a false alarm due to various reasons such as malfunctioning, battery depletion, etc. Therefore, the long-run value of the probability of false alarm would be 1. Instead, we use the residual time-to-false-alarm corresponding to each state T_s in order to measure how long it takes to reach state 0 and remain there, conditioned on being in each state. Averaging these quantities over all possible states would result in the metric L_f which describes the expected value of time before reaching state 0, or equivalently the inverse rate of reaching a false alarm.

Equations 3.14 and 3.19 give us the basis for the design and analysis of the performance of the proposed pairwise pinging scheme. We will discuss this issue in the following section.

3.1.2 Performance Evaluation

To investigate how the scheme works, we present a numerical example and the analytical solutions for the desired performance measures. We pick a *lognormal* distribution for X as it can describe a non-negative continuous random variable with nonzero mode and mean. Fig. 3.3 shows the *pdf* of a *lognormal* distribution with a mean of $300sec$ and parameter $\sigma = 0.5$. Equation 3.20 shows the *pdf* of a *lognormal* distribution with its mean and variance.

$$
\begin{aligned}
&f_X(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}} \qquad x \geq 0 \\
&\eta_x = e^{\mu+\sigma^2/2} \\
&\sigma_x^2 = (e^{\sigma^2}-1)e^{2\mu+\sigma^2}
\end{aligned}
\tag{3.20}
$$

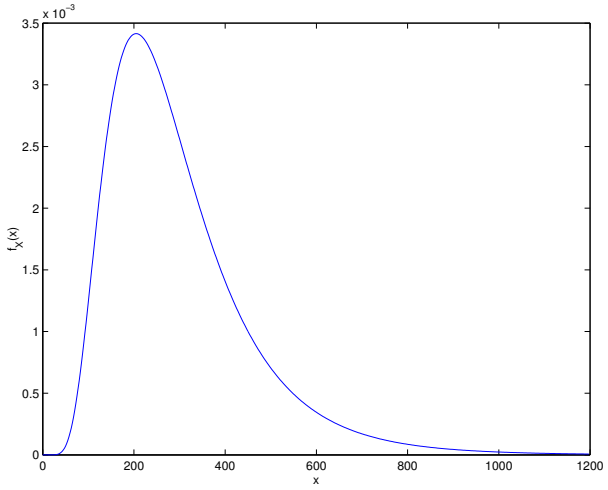


Figure 3.3: *pdf* of a *Lognormal* distribution with $\eta_x = 300$ *sec* and parameter $\sigma = 0.5$.

Now, as an example we consider the following values for the corresponding parameters:

$$
\begin{aligned}
&M = 20 \\
&p_l = 10^{-3} \\
&T_e = 5 \ sec
\end{aligned}
\tag{3.21}
$$

Fig. 3.4 depicts the probability of being at each state s for different values of p_r by solving Eq. 3.9 numerically. For example, the pinging rate of $p_r = 0.3$ results in concentrating most of the mass function from states 10 to 20, while in the case of $p_r = 0.1$, the mass function is spread over the entire state span. This observation can be explained as follows. By increasing the

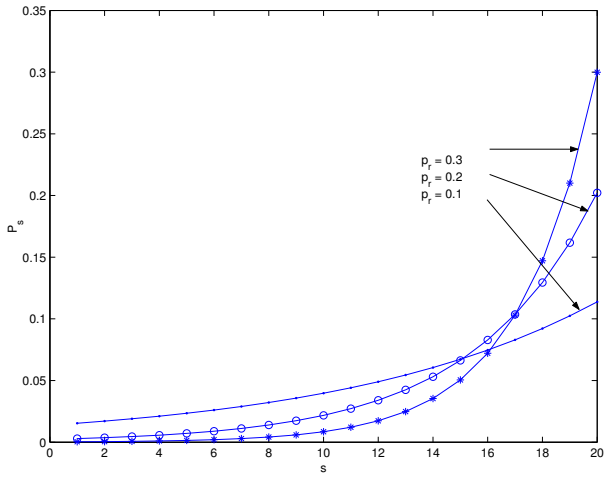


Figure 3.4: Probability of being at each state P_s v.s. state s for different pinging rate p_r.

value of p_r, the system tends to be set to its regenerative points more often and remain in higher states. Consequently, there is more time for adversaries to complete their capture phase either before being detected or before the elapsed detection interval. This would lead to an increase in the probability of a missed detection P_m.

Now, we try to evaluate the sensitivity of the probability of a missed detection P_m with respect to the detection interval T. By solving Eq. 3.14 we can see in Fig. 3.5 that for any particular value of p_r, increasing the detection interval or equivalently, increasing the number of states M would result in a higher probability of a missed detection P_m. For example, for the pinging rate of $p_r = 0.2$, by increasing the detection interval from 120 *sec* to 200 *sec* (which correspond to $M = 24$ and $M = 40$ number of states, respectively), the probability of a missed detection is increased from 0.01 to 0.1, respectively. This observation can be justified as follows. The higher the

detection interval we have, the longer vulnerable interval can be used by the adversary to complete the capture phase and escape from our proposed scheme which in turn leads to a higher probability of a missed detection P_m. Also it is clear that by increasing the pinging rate p_r, the graphs tend to grow slower and eventually reach some saturation levels, as higher values of p_r force the nodes to remain in higher states. This implies that in those saturated regions no matter what the values of p_r are, P_s has most of its mass in its highest states. Therefore, the adversary has roughly the maximum time of about one detection interval to succeed in its capture, thus causing a relatively constant P_m regardless of p_r.

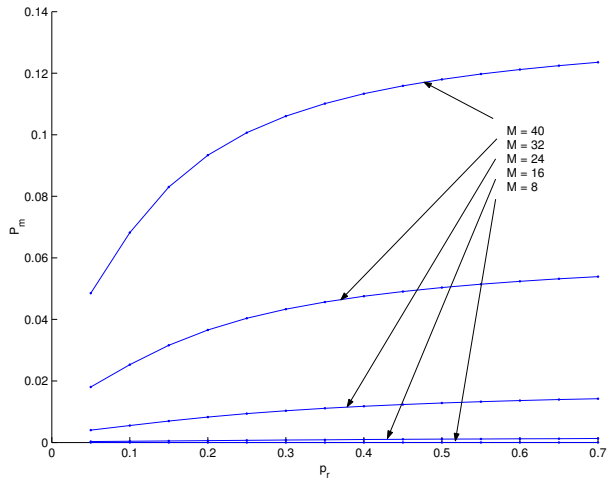


Figure 3.5: Probability of a missed detection P_m w.r.t. p_r and M. The distance between graphs shows exponential dependency of P_m v.s. M.

Using Eq. 3.19, we can get the expected residual time-to-false-alarm of each node before any occurrence of false alarm. This measure tries to take into account the average lifetime of any node in the network. Fig. 3.6 illustrates the dependency of expected residual time-to-false-alarm L_f versus p_r for different values of the detection interval T, or a proportional number of states M. It

shows that increasing p_r improves the lifetime of each node. For example, for $M = 24$ which corresponds to detection interval of 120 *sec*, by changing the pinging rate from $p_r = 0.2$ to $p_r = 0.4$, we can increase the average residual time-to-false-alarm of nodes from 2.8 *hours* to 278 *hours*. The results from Eq. 3.9 and Fig. 3.4 imply that having a higher pinging rate maintains nodes in higher states which in turn causes a longer lifetime before reaching state 0 and having a false alarm. Additionally, for any particular value of p_r, increasing the detection interval improves the residual time-to-false-alarm. Longer detection interval provides more opportunity to receive a ping message and sets the state back to M before reaching state 0. This in turn implies a longer time-to-false-alarm or equivalently a smaller rate of reaching a false alarm.

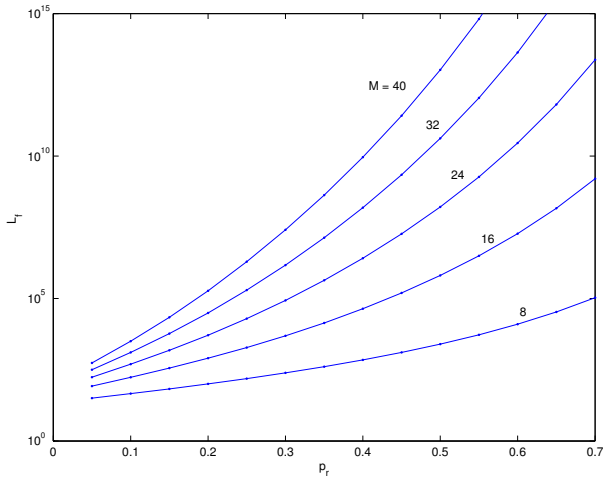
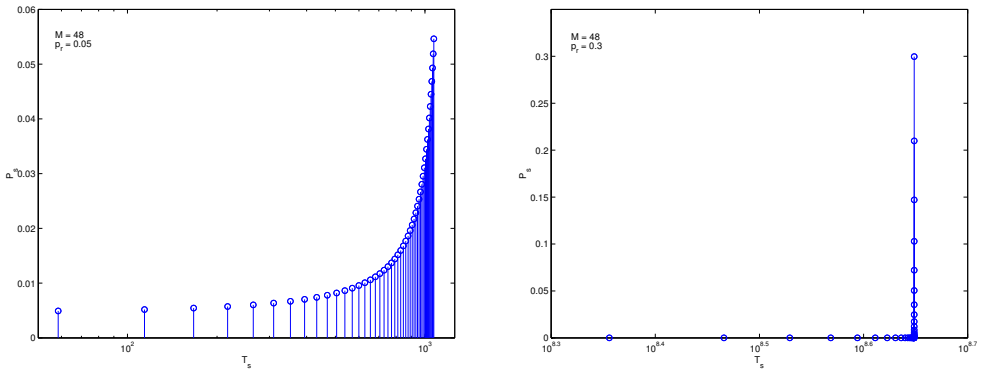


Figure 3.6: Expected residual time-to-false-alarm L_f w.r.t. p_r and M. Logarithmic vertical axis shows the exponential dependency of L_f v.s. p_r.

Fig. 3.7 shows the sensitivity of the residual time-to-false-alarm in terms of p_r. The left graph in this figure corresponds to a $p_r = 0.05$ and the right graph corresponds to a $p_r = 0.3$ pinging rate. Recall that the pmf of Eq. 3.9

gives the probability of being at state s, and hence it gives the probability of having the residual time-to-false-alarm T_s given being in state s. In the first case, the residual time-to-false-alarm takes nonzero probabilities over a long range of states with a tendency towards higher states, each having their corresponding residual time-to-false-alarms. While in the second case, the majority of mass is concentrated around state M which prevents approaching the lower states and increases the residual time-to-false-alarm of each state. Note that in the second case, the order of the residual time-to-false-alarm is in the range of $10^{8.6}$ sec or equivalently 12 $years$.



(a) Case 1: $p_r = 0.05$ (b) Case 2: $p_r = 0.3$

Figure 3.7: Sensitivity of L_f and T_s w.r.t. p_r

Remark 2 The communication cost per node per epoch is $p_r d$ message receptions and transmissions. The computation cost per node per epoch is also $p_r d$ number of MAC [2] verifications, increments, and MAC generations. Also, the scheme requires each node to keep a state counter for each neighbor resulting in d registers per node.

[2]MAC refers to a Message Authentication Code procedure.

Remark 3 If we have more than one adversary agent present at a time in a specific neighborhood, we need to look at their cross effects on our scheme. Considering one of the target nodes under the capture phase, the rest of the neighbors are either legitimate, under the capture process, or already captured. If they are legitimate, they will follow the scheme properly. If they are under the capture phase, then they can not influence the adversary's success on the target node other than reducing the effective node degree d_{eff} of the target by not participating in the scheme. Hence, we can consider each of the adversaries independently. However, if they have already succeeded in capturing a node, they may have a stronger impact on the scheme, a scenario which needs to be investigated in more details. In this case, the scheme can not count on the cooperation of the adversaries in any way. However, since the scheme is designed based on pairwise interaction between the target node and each of its neighbors, the influence of adversaries is only limited to reducing the effective node degree. Further discussion regarding the impact of cooperations among multiple adversary agents on the proposed scheme will be presented in details in Chapter 4.

Remark 4 All neighbors of each node have to be able to initiate a ping message. Given node degree d, we need to pick K such that $K \geq d$. However, in order to reduce the probability of collision among one hop neighbors, we need to select $K \gg d$. Also another way to reduce collision is to reduce p_r. We should notice that reducing p_r may not be desirable as it also affects the performance of our pinging scheme.

Remark 5 The main performance measures in our scheme are P_m and L_f. Looking at Fig. 3.5 shows strong sensitivity of P_m to M or T, while Fig. 3.6

illustrates the exponential dependency of L_f to p_r. Note that p_r has a direct relationship with the computation and communication costs of the scheme. So having an efficient scheme requires us to keep p_r as small as possible as far as cost is concerned. These observations give us a trade-off in selecting appropriate values for the parameters in order to meet the desired performance objectives.

Remark 6 Given the required performance measures P_m and L_f, we can get a design procedure for setting the appropriate parameters of the scheme. Note that in WSN applications, the importance of P_m is higher than L_f because of the fact that by not detecting a captured node we may be vulnerable to severe malicious influence on the network traffic by the adversary. However, by raising a false alarm for an innocent node we waste the available resources of a network. Table 3.2 summarizes the steps to obtain the appropriate parameters for the scheme. In this design scheme we simply solve for the best values of p_r and M iteratively. This leads to a sub-optimal solution of the design parameters for meeting the desired performance.

Desired P_m and L_f are given.
d is defined by the deployed network topology.
T_p is set based on the round trip delay and the computational speed of nodes.
$f_X(x)$ or η_x are known *a priori*.
K is set with respect to d such that $K \gg d$.
Using Fig. 3.6 for the desired L_f, for each M we find $min\ p_r$.
Using Fig. 3.5 for the desired P_m, for each M we find $max\ p_r$.
Intersection of the above two regions for p_r gives corresponding values for M and p_r.
If there is no intersection, the design is not achievable by the desired L_f and P_m.
Set M to the lowest possible value.
For the selected value of M, pick $min\ p_r$ as the final value of p_r.

Table 3.2: Summary of design procedure

3.2 Quorum Pinging Scheme

As mentioned earlier, the pairwise pinging scheme is conducted based on a pairwise interaction between any two neighbors. When a neighbor invokes this scheme for a potential target node, it can raise a flag against the target node. Then, this flag may be used as a basis for judging the status of the target node from that neighbor's viewpoint. Since any of the neighbors can be captured by the adversary itself, or they might have a bad judgement about the target node for any other reason, we can not rely solely on a single raised flag as a basis for our revocation decision. Rather, we have to reach a certain number of flags against the target node in order to conclude that the target node has been captured and needs to be revoked. Otherwise, the captured neighbor can maliciously run this scheme against its target node and revoke that innocent neighbor. This leads to an increased false alarm rate caused by the adversary, causing the detection process to backfire. In the distributed revocation policy proposed in [17], q is the decision threshold for the revocation of any target node. To exploit this observation in our scheme, we need to generalize the pairwise pinging scheme to a *q-node quorum pinging scheme* in which all the neighbors of the target node are engaged in a decision making procedure in order to reach a consensus about the status of the target node. Once the consensus regarding the captured status of the target node is achieved, the result is spread across the network by those neighbors of the target node which participated in the decision process and the network can reliably revoke that captured node.

3.2.1 Modeling of the Quorum Pinging Scheme

In order to maintain the self-organizing and distributed requirements discussed earlier, we have to be very careful about the potential coordination among the neighbors of the target node in the consensus based decision process. In the proposed quorum pinging scheme, each node keeps a counter corresponding to each of its one-hop neighbors. While running the pairwise pinging scheme between each pair of neighbors independently, the issuer of a potential flag raised against a target node broadcasts its flag in its own neighborhood. Then, some of the target's neighbors would receive that flag and increment the counter associated with that particular target node. Once the number of collected flags, or equivalently the level of the target node's counter kept by its neighbors, reaches the required threshold q, the neighbors take that as a consensus among at least q neighbors of the target node in their common neighborhood that the target node has been captured. Then, all those participating neighbors in the quorum issue a revocation flag against the target node and spread the revocation flag across the network. After that, it is up to the deployed revocation mechanism as to how to revoke that captured node's ID across the network.

Note that the required coordination among at least q neighbors of target nodes are aligned with the distributed and self-organizing properties of our interest. The execution of the pairwise pinging scheme and the generation of each raised flag is independent of the other neighbors of their common target node. Also, the counter update of the target node is done individually in each neighbor and it is quite possible that some neighbors have different

values in their own counter corresponding to the particular target node. This can happen if some neighbors do not hear the flag broadcasted by another neighbor of the target node for various reasons such as packet loss, collision, out of communication range between two neighbors of the target node, etc.

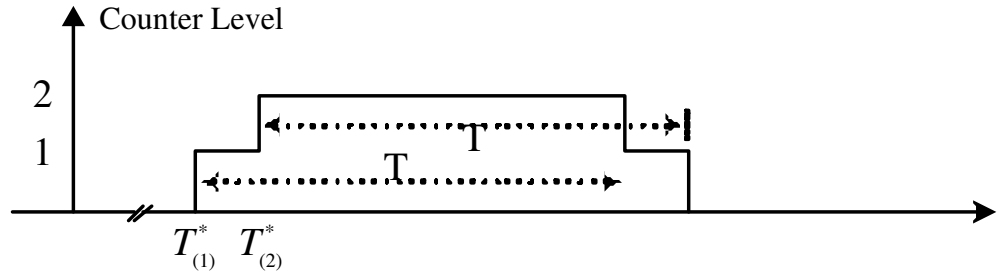
The coordination for aggregate flags has to be performed in a timely manner such that all the required q flags against the target node to be revoked need to occur before the adversary can complete its capture process of the target node. This is done by assigning a lifetime period for each flag raised by a neighbor. The lifetime of each flag is equal to the detection interval T which was defined earlier in the pairwise pinging scheme section. The reason for such a choice for the lifetime duration of each flag lies on the following premise. In order to reach a consensus among all q participating neighbors regarding the captured status of a target node in their common neighborhood, all the flags have to be raised within a time duration of length T. This is because of the fact that each neighbor runs their own *Markov Chain* corresponding to the target node and each can be in one of the states between 1 to M. Once one neighbor raises a flag against the target node, it means that its state is 0 at the current time. For other neighbors in order to support this flag or not by their own judgement about the target node, it takes at most MT_e or equivalently T unit of time to reach the same assessment about the target node. In other words, if some other neighbor is in state s at current time, it takes sT_e unit of time to reach state 0 which corresponds to its own raised flag. Note that because the objective of the pinging scheme is to detect the presence of an adversary on a target node, once the adversary is physically present on the target node then the effect of its presence would be observable by all the neighbors. This would

result in the occurrence of dependent events on all the neighbors of the target node. As a result, the flags from the rest of the neighbors would be raised with at most a lag of T units of time. The beauty of this setting is that even though the interactions of each neighbor with the target node are independent from each other, the desirable event of interest, namely the captured status of the target node, would be dependent upon the observations among its neighbors. This property allows us to expect the same reaction from all the neighbors within a certain time duration of length T.

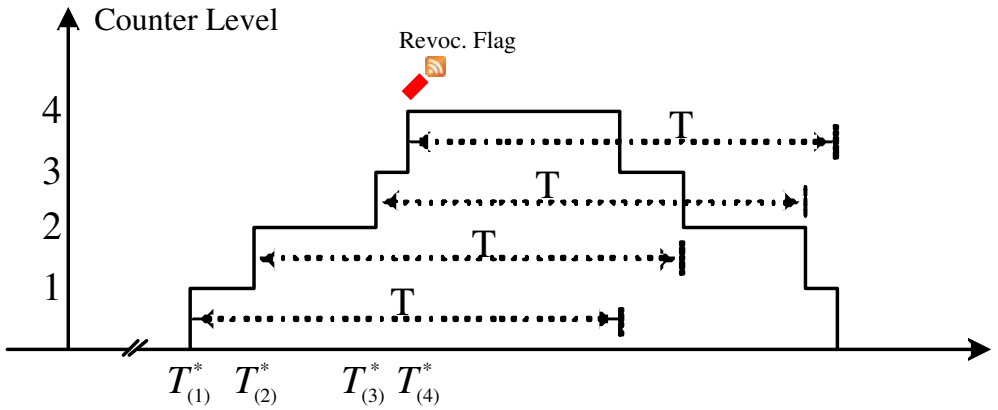
Thus, each node has a counter corresponding to each of its neighbors and once a flag against a target node is raised by itself or another neighbor of the target node which is broadcasted by the initial neighbor, that counter will be incremented. Now if at any point in time the value of the counter reaches the threshold q, all the neighbors which have issued those q flags commit to a revocation flag and broadcast it across the network. Then, depending on the deployed revocation policy the network initiates the revocation procedure of the target node which is now identified by those q neighbors as captured by an adversary. The other possibility is that the counter does not reach the threshold q after T units of time from the time that the first flag was issued. This means that the judgement about the target node is not supported by enough of its neighbors watching it. At that time the first issued flag would be expired and therefore, the counter would be decremented once. This issue could be repeated for the other flags if they are not supported by at least $q - 1$ other successive flags. Note that upon the expiration of any flag, the corresponding *Markov Chain* would reset back to its original state M and the execution of the pairwise pinging scheme between the neighbor which issued

the expired flag and the target node starts over.

Clearly, if we were to observe the counter level over time, it would look like a queue which goes up and down one step at a time upon the arrival or expiration of each flag until the time that the counter either hits the zero level or reaches the level q. If it returns to zero, then the scenario repeats itself and the scheme waits for the next raised flag. On the other hand, once the counter reaches the level q, a revocation flag will be issued and broadcasted across the network. Consequently, the network will revoke the target node and all the neighbors of that captured node would cease to monitor it. So, it should be clear that upon reaching the level q, the queue associated with the captured node no longer exists. Fig. 3.8 depicts two possible sample paths of such queue. Note that it is quite possible that the queue fills to some level less than q and empties out until it reaches level q or may never reach level q, however reaching level q can occur at most once. This implies that over time the queue may go through sample paths similar to the left graph in Fig. 3.8 several times while it can go through the right graph only once. Note that $T^*_{(i)}$ denotes the ordered statistics of the arrival time of the i^{th} flag raised by one of the neighbors of the target node.



(a) Case 1: The counter does not reach its threshold level within the lifetime of the raised flags



(b) Case 2: The counter reaches the threshold level and a revocation flag is raised

Figure 3.8: Queue sample paths for the quorum size $q = 4$

This queue can be modeled as $G/G/1/q$ based on Kendall's notation.As there is not a closed form analytical solution for this queue model, we try to find an approximation model in order to evaluate the desired metrics of this queue.

Similar to the pairwise pinging scheme, we are interested in finding the expected residual time-to-false-alarm $L_f^{(q)}$, and the probability of a missed detection $P_m^{(q)}$ for the *q-node* quorum pinging scheme.

Contrary to the pairwise pinging scheme, a single raised flag against the target node would not be interpreted as an assessment for the captured status of the target node. Instead, we have to have at least q supporting flags within a time duration of length T raised by the target node's neighbors in order to reach a consensus that the target node is really captured and needs to be revoked. This would define the underlying emergent property [46] of the quorum pinging scheme in which a significant gain in performance of captured node detection is viable in exchange of reasonable cost only through the collective actions of multiple nodes, namely the neighbors of any target node.

A. Probability of a Missed Detection. Increasing the quorum size q means involving more neighbors in the decision making process regarding the status of the target node, although this leads to a slower decision procedure. We will consider this effect on missed detection and false alarm events and compare those with their counterparts in the pairwise pinging scheme in which q is basically set to 1.

A missed detection event in this case occurs when the target node is being captured by the adversary and there are at most $q-1$ neighbors which witness this event and raise flags against that target node. Therefore, in such circumstances consensus cannot be achieved by the neighbors and the raised flags would expire one after the other. Mistakenly, the overall judgement of the neighbors is that the target node is still legitimate and not captured. Intuitively, increasing the quorum size q should lead to a higher probability of a missed detection. This is because given the target node is under the capture process, the scheme requires more neighbors to raise flags against the target node in order to detect the presence of an adversary on it. By increasing q, the occurrence of this consensus becomes more unlikely and there is a greater chance of missing a captured node, which in turn leads to a higher probability of a missed detection. We will investigate the validity of our intuition through our performance evaluations using different values of the quorum size q in order to compare its effects on the probability of a missed detection in the quorum pinging scheme.

In order to determine the probability of a missed detection for the quorum pinging scheme $P_m^{(q)}$, given that the target node is captured, a missed detection event can happen only if at least $d-q+1$ neighbors miss the common target node and consider it as a legitimate one even though the target node is actually under capture by an adversary. This is equivalent to saying that at most $q-1$ neighbors detect the target node as a captured node if the node is really under capture. Thus, the probability of a missed detection in this case can be written as follows:

$$P_m^{(q)} = \binom{d}{d-q+1} P_m^{d-q+1}(1-P_m)^{q-1} + \binom{d}{d-q+2} P_m^{d-q+2}(1-P_m)^{q-2} + ...$$
$$... + \binom{d}{d-q+i} P_m^{d-q+i}(1-P_m)^{q-i} ... + \binom{d}{d} P_m^d \tag{3.22}$$

in which $\binom{d}{d-q+i} P_m^{d-q+i}(1-P_m)^{q-i}$ corresponds number of possibilities that $q-i$ out of d neighbors of the target node detect the target node as under capture and the remaining neighbors miss that event.

B. Expected Residual Time-to-false-alarm. In this step, we try to analyze the behavior of the quorum pinging scheme with respect to false alarms. Here we consider the event in which, given the target node is not captured, the overall decision of the scheme is that the target node is marked as captured. This false alarm event can happen only if at least q neighbors identify the target node as captured in error and raise flags against it. To proceed with this consensus successfully, all these flags have to be raised within the time duration of length T from each other otherwise some of the flags may expire. If this happens, then a revocation flag will be mistakenly issued by the neighbors to revoke the target node permanently.

Similar to the pairwise pinging scheme, we are interested in finding the average time it takes for at least q neighbors of a legitimate target node to reach a consensus about it and identify it as a captured node, which is called the expected residual time-to-false-alarm $L_f^{(q)}$. Again our intuition tells us that increasing quorum size q in a false alarm event should lead to a lesser likelihood of false alarms or, equivalently, a longer time-to-false-alarm. This

is because by including more neighbors in the decision process, it would be harder for them to reach a consensus about the captured status of the legitimate target node. Note that not reaching a consensus in any situation means that the participating neighbors cannot declare the target node as captured. This in turn means that the target node will be treated normally as a legitimate one regardless of its actual status. Therefore, the only possible decision resulting from our detection mechanisms, either the pairwise or quorum pinging schemes, is to declare the target node under investigation as captured. Otherwise, the implication of not reaching a final decision is that the target node is legitimate and should be treated normally as other legitimate nodes of the network.

As pointed out earlier, in order to model the time-to-false-alarm metric, we need to solve the $G/G/1/q$ queue analytically. To get around this step, we propose an approximation approach in order to evaluate the time-to-false-alarm and to get a sense of performance evaluation in the case of false alarms. To do so, we recall the way the quorum scheme is executed. As each neighbor of a target node performs a separate pairwise pinging scheme with the target node independently, the time it takes for each of them to reach a false alarm flag against the target node is on average L_f. In the case of generating a false alarm in the quorum scheme, at least q flags need to be raised by the neighbors of a legitimate target node within a time duration of length T. So, if one of the flags takes about L_f *sec*, the rest of the flags need to be raised within time T of the first one to lead to a raised revocation flag. So the bounds can be achieved by:

$$L_f \leq L_f^{(q)} \leq L_f + T \tag{3.23}$$

Considering the nominal values of the quantities in Eq. 3.23, we can impose the *Sandwich theorem* in order to approximate the value of the expected residual time-to-false-alarm in the quorum pinging scheme $L_f^{(q)}$. For all practical applications, the value of T is much less than L_f, i.e., $L_f \gg T$ which results in a tight bound in Eq. 3.23 and can be derived as:

$$liminf \quad L_f^{(q)} = L_f = E[T_s] \tag{3.24}$$

Hence, the expected residual time-to-false-alarm in the quorum pinging scheme $L_f^{(q)}$ can be well approximated for our practical purposes by:

$$L_f^{(q)} \approx L_f \qquad \text{given} \qquad T^*_{(i+q-1)} - T^*_{(i)} \leq T \tag{3.25}$$

Recall that $T^*_{(i)}$ is the ordered statistics of the arrival time of the i^{th} flag raised by one of the neighbors of the target node. In this case, the i^{th} flag is the first flag of the first batch of q flags within time T. Here these flags are sorted based on the order they are raised in time regardless of which neighbors have issued them.

Considering Eq. 3.25 seems to suggest that as far as false alarm is concerned, we do not experience any gain from the quorum scheme compared to its pairwise counterpart. However, the result in Eq. 3.25 is subject to the condition that the consensus among q neighbors is achievable. In order to

achieve the above conditional expectation, we have to find the probability of occurrence of false alarm among at least q neighbors of a target node. This probability can be evaluated based on the state values of the target node kept in its neighbors. Reaching a consensus about the captured status of a legitimate node leads to raise a false alarm against the target node. To do so, all the q flags have to be raised within time T of each other. Note that in a false alarm case, the target node is legitimate and as opposed to a missed detection case, the observable event of the target node for its neighbors are independent from each other. This results in independent behavior among all the neighbors. Some may react correctly and judge the target node as legitimate and some may mistakenly reach to a conclusion that the target node is under capture. In the latter case, those neighbors have reached state 0 of the *Markov Chain* corresponding to the target node. For those neighbors who reside in lower values of state for the target node and raise a false alarm flag against the target node, it takes them less time to do so. These times are already evaluated based on the pairwise interactions of each neighbor with the target node in Eq. 3.16 and denoted as T_s. As each neighbor runs a similar *Markov Chain* with the same parameters for a common target node, the solutions of T_s would be the same among those neighbors. However, each neighbor can possesses different state values corresponding to the same target node at any given point in time. Now, in order to evaluate the probability of a false-alarm in the quorum pinging scheme, we have to add all possible *q-tuples* masses in which their corresponding T_s are within the time duration of length T. Therefore, we need to construct a sliding window of length T and move it along the values of T_s. Then for those states with proper T_s, find their collective mass and sum these collective masses for all possible options. This

quantity would result in the probability of a false-alarm in the quorum pinging scheme which is denoted as $P_{F.A.}^{(q)}$. In the following section, we will evaluate this quantity and along with the expected residual time-to-false-alarm in the quorum pinging scheme $L_f^{(q)}$ derived in Eq. 3.25, we will be able to conclude the gain that we can achieve using the quorum pinging scheme compared to its pairwise counterpart. Note that in the pairwise case, there is not any consensus requirements and the judgement about the target node is solely raised by each neighbor of the target node. Thus, the occurrences of false alarms are only dependent upon each neighbor and the expected residual time-to-false-alarm L_f will almost surely be achieved.

Assumption 11 Quorum size q is smaller than the number of legitimate neighbors of any target node, known as the effective node degree d_{eff} (i.e., $q \leq d_{eff}$).

Assumption 12 We have a low probability of collision among ping messages and responses due to the short duration of messages with respect to bit rate.

3.2.2 Performance Evaluation

To compare the performance of the pairwise and quorum pinging schemes, we present several numerical examples in this section. In each of these examples, we try to see the effects that different parameters have on the performance of the two proposed schemes. We also will investigate the validity of our intuitions about the behavior of the schemes with respect to various parameters.

The examples are presented in four different scenarios. The first scenario corresponds to the nominal range of parameters and all three other scenarios

are compared against the first one in order to show the effect of various parameters on the performance of our proposed schemes.

Scenario 1. In this example we consider the following values for the corresponding parameters:

$$\begin{aligned} d &= 20 \\ p_l &= 10^{-3} \\ T_e &= 5 \ sec \\ \eta_x &= 400 \ sec \end{aligned} \tag{3.26}$$

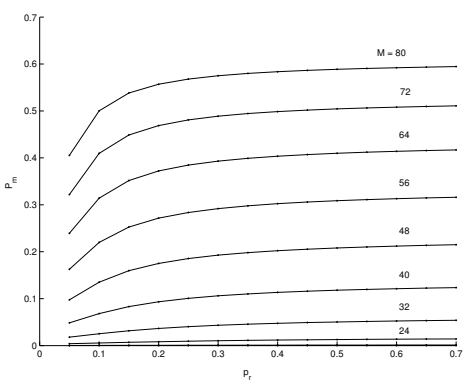
The typical design objectives for time-to-false-alarm and the probability of a missed detection in a WSN application is considered in this scenario. Here we need to meet around 116 days of time-to-false-alarm and 0.20 missed detection rate as our objective performance measures.

$$\begin{aligned} P_m &\leq 0.2 \\ L_f &\geq 10^7 \,\text{sec} \approx 116 \ days \end{aligned} \tag{3.27}$$

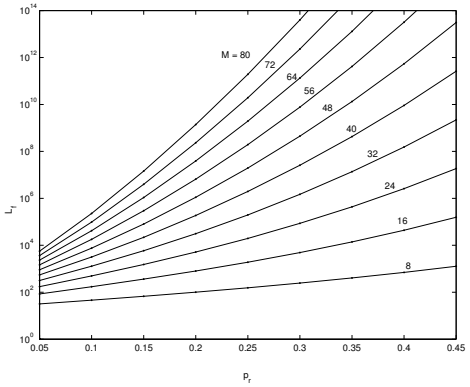
Following the results depicted in Fig. 3.9 leads us to have 48 states and a 0.23 pinging rate for the proposed pairwise pinging scheme which is shown in Eq. 3.28.

$$\begin{aligned} M &= 48 \\ p_r &= 0.23 \end{aligned} \tag{3.28}$$

Selecting the parameters in Eq. 3.28 satisfies the inequalities in Eq. 3.27 and leads to a 0.12 probability of a missed detection precisely, i.e., $P_m = 0.12$.



(a) Probability of a missed detection P_m w.r.t. p_r and M.

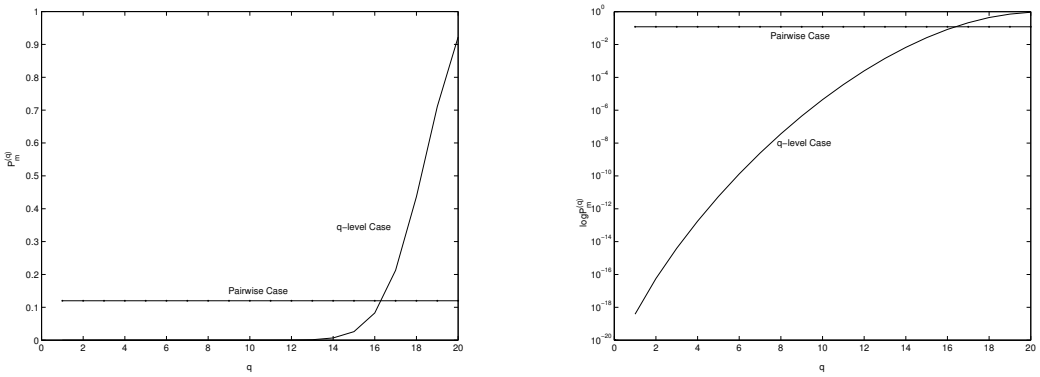


(b) Expected residual time-to-false-alarm L_f w.r.t. p_r and M.

Figure 3.9: Relevant graphs to design pairwise pinging scheme parameters for Scenario 1.

Now, we use the *q-node* quorum pinging scheme to show the performance gain that we can achieve. By imposing the quorum scheme with the corresponding parameters derived in Eq. 3.28 from the pairwise scheme, we use Eq. 3.22 to evaluate the effective probability of a missed detection for this scheme $P_m^{(q)}$. Fig. 3.10 shows the value of $P_m^{(q)}$ for different values of q. The left graph is the actual values of $P_m^{(q)}$ for corresponding q and the right graph is the logarithmic values of $P_m^{(q)}$ to show a better picture for smaller values of q.

Observe that for the pairwise case the value of $P_m^{(q)}$ is constant regardless of q and it is equal to the original P_m. This is because in the pairwise case q is effectively equal to 1 and the evaluated P_m in that case is the probability of a missed detection for any potential neighbor of the target node. To explain why the value of $P_m^{(q)}$ corresponding to $q = 1$ in quorum pinging is not equal to the case of pairwise pinging, we have to consider the following. As opposed



(a) Actual value of $P_m^{(q)}$ versus q.

(b) Logarithm of $P_m^{(q)}$ versus q.

Figure 3.10: Probability of a missed detection for the *q-node* quorum pinging scheme w.r.t. the one in the pairwise pinging scheme for various quorum size q.

to the pairwise case, in the quorum scheme there is a race among at least q out of d neighbors of each target node to reach a consensus based judgement (in order to constitute a quorum) regarding its captured status. For $q = 1$ and given that the target node is really under the capture process, it can be detected as captured if at least one of its neighbors detects the adversary's presence on that node. So, the only way a missed detection event can happen is that no neighbor detects the captured status of the target node. Clearly, the occurrence of such events are very unlikely compared to the case of the pairwise scheme in which the probability of a missed detection is evaluated for any potential neighbor of the target node and it has an average sense of a missed detection event. Thus the probability of a missed detection for the latter case will be higher.

Now, to justify the monotonically increasing regime of $P_m^{(q)}$ with respect

to q, we can consider the following case. Following the argument given above, consider the case in which $q = 5$ and $d = 20$. Given that the target node is under capture, a consensus for detecting its captured status can be achieved if at least 5 of its neighbors detect that event. This means that if there are 4 or less neighbors which detect that event, then the consensus cannot occur and the scheme will miss it. This implies that we can have more possibilities which lead to a missed detection event compared to the case of $q = 1$. This argument shows the reason for increasing the trend of $P_m^{(q)}$ with respect to q. As a result, increasing the quorum size q would not be helpful as far as the missed detection rate is concerned and we prefer to keep q as low as possible. Note that as Fig. 3.10 shows, we have some gain in the probability of a missed detection for the cases in which $q \leq 16$ and in particular for $q \leq 13$, this measure is reasonably negligible.

To evaluate the projected gain for the false alarm rate in the quorum scheme, we refer to Eq. 3.25 which shows a similar expected residual time-to-false-alarm compared to the pairwise case, except its precondition factor. As discussed earlier, the occurrence of a false alarm is expected to be roughly around L_f almost surely. While in the quorum scheme, the false alarm event is achieved subject to the probability of false alarm $P_{F.A.}^{(q)}$. In order to calculate this metric, we need to determine the distribution of the residual time-to-false-alarm given in any state, T_s in the underlying pairwise scheme and the corresponding *pmf* P_s. These quantities are derived from Eqs. 3.16 and 3.9, respectively. Fig. 3.11 shows the residual time-to-false-alarm T_s given in state s for the corresponding pairwise case with the parameters described in Eq. 3.26.

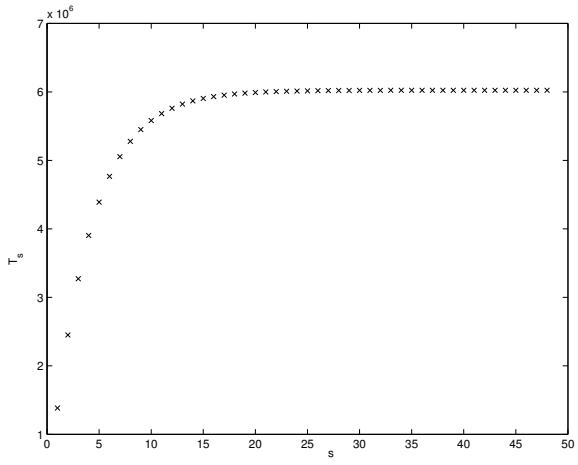
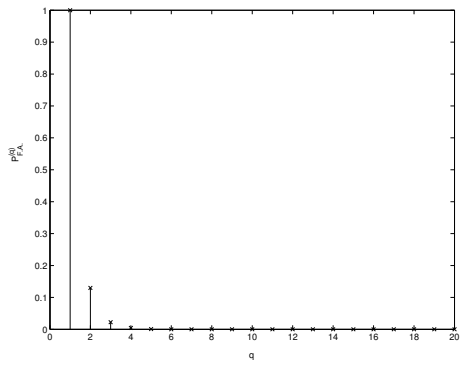


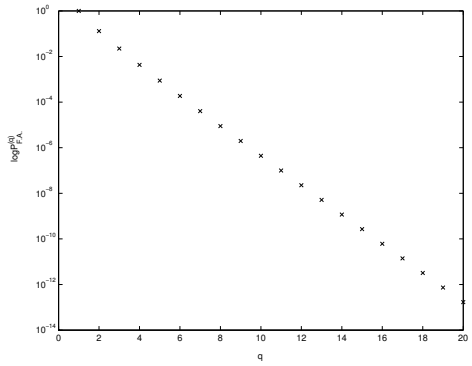
Figure 3.11: Residual time-to-false-alarm T_s given in state s in Scenario 1.

Using the above values of T_s and finding their appropriate *q-tuples* with respect to T, we can find the possible *q-tuples* masses and find their sum to get the probability of false alarm denoted by $P_{F.A.}^{(q)}$. Fig. 3.12 depicts this quantity. Again, for clarification we put the actual values on the left graph and the logarithmic values on the right graph to get a sense of its reduction rate with respect to q.

Similar to our argument on the missed detection case, we will explain the trend of this graph for the quorum scheme and its comparison with its pairwise counterpart. In the case of false alarm events, the precondition is that the target node is legitimate and not captured while the neighbors reach a consensus regarding the captured status of the target node. Now, consider the quorum scheme for which the quorum size q is set to 1 and there are 20 neighbors in each target node's communication range. In this case, if at least 1 out of 20 neighbors of the target node raise a flag against the target node,



(a) Actual value of $P_{F.A.}^{(q)}$ versus q.



(b) Logarithm of $P_{F.A.}^{(q)}$ versus q.

Figure 3.12: Probability of false alarm $P_{F.A.}^{(q)}$ for quorum pinging scheme with various quorum size q.

the consensus will be achieved and the revocation flag will be raised by the participating neighbors in the quorum. This would cause a false alarm and lead to the revocation of the legitimate target node by mistake. The only way to skip false alarms in this scenario is for all 20 neighbors not to raise a flag against the target node which is actually the correct thing for them to do. In contrast, if we set q to be 5 then for reaching a consensus among the neighbors, the scheme is at least required to collect 5 flags out of 20 neighbors. If

there is less than or equal to 4 neighbors which raised flags against the target node, the consensus cannot be achieved and the false alarm does not occur. Comparing the latter case with the former one, we can easily see that by increasing q, the chances for false alarm events will be reduced, which is exactly the same observations we get from Fig. 3.12. So, as far as the false alarm rate is concerned we need to have higher values of the quorum size q in order to get a lower probability of false alarm $P_{F.A.}^{(q)}$. The benefit of imposing the quorum scheme with respect to false alarms is significant once we have $q \geq 4$ in this example. Note that in the right graph of Fig. 3.12, the reduction in the probability of false alarm $P_{F.A.}^{(q)}$ has a linear trend, which seems to suggest that its actual value decreases exponentially by increasing q.

Overall, the achieved gain of using the quorum scheme as opposed to the initial pairwise pinging scheme is governed by our proper choice of the quorum size q which is $4 \leq q \leq 13$ in this example. This selection of q would lead to a negligible probability of a missed detection and also a negligible probability of false alarm even though the conditional residual time-to-false-alarm remains in the same order as the one in its pairwise counterpart. Eventually, in the limit sense when the probability of false alarm tends toward zero, we can interpret that as if the residual time-to-false-alarm approaches infinity.

Scenario 2. In this example we try to discover the effects of weak physical protection of the nodes, weak obfuscation techniques, and a high skill level of the adversary in capturing a target node in a WSN. All these circumstances would lead to a shorter time for the adversary to complete the capture process of a target node. Compared to Scenario 1, when the adversary

completes its capture faster, it is intuitively obvious that the scheme needs to work harder to combat the adversary. The immediate influence of such circumstances on the parameters of our scheme is on the reduced value of η_x. Here we consider an adversary which is capable of completing the capture process of its target node on average 4 times faster than the one of Scenario 1. In summary, the parameters of the underlying pairwise pinging scheme for this scenario are as follows:

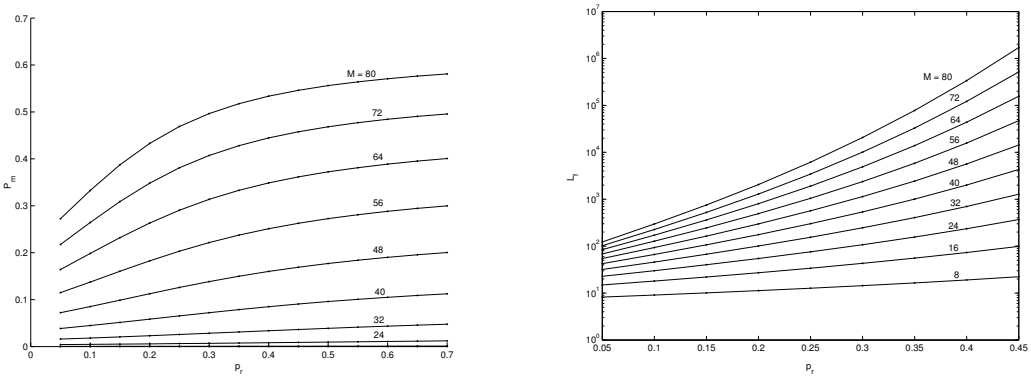
$$\begin{aligned} d &= 20 \\ p_l &= 10^{-3} \\ T_e &= 5 \ sec \\ \eta_x &= 100 \ sec \end{aligned} \tag{3.29}$$

Using the same performance requirements as in Scenario 1 in terms of the time-to-false-alarm and probability of a missed detection would not lead to any result in the pairwise pinging scheme under the current parameters in Eq. 3.29. Therefore, in order to get a feasible result, we relax the desired performance requirements and expect to have a possible outcome from the underlying pairwise pinging scheme. Here we set our required objectives as follows:

$$\begin{aligned} P_m &\leq 0.25 \\ L_f &\geq 10^4 \,\text{sec} \approx 2.8 \ hours \end{aligned} \tag{3.30}$$

Following the results depicted in Fig. 3.13 leads us to have 56 states and a 0.37 pinging rate for the proposed pairwise pinging scheme which is shown in Eq. 3.31.

$$\begin{aligned} M &= 56 \\ p_r &= 0.37 \end{aligned} \tag{3.31}$$



(a) Probability of a missed detection P_m w.r.t. p_r and M.

(b) Expected residual time-to-false-alarm L_f w.r.t. p_r and M.

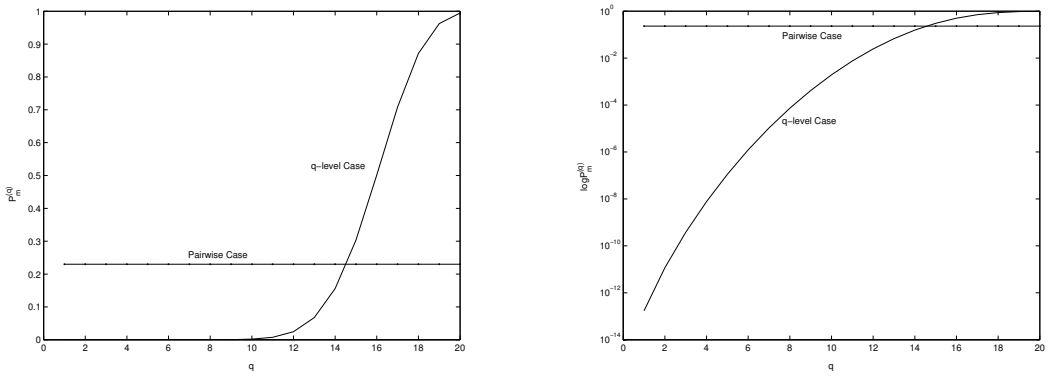
Figure 3.13: Relevant graphs to design the pairwise pinging scheme parameters for Scenario 2.

Selecting the parameters in Eq. 3.31 satisfies the inequalities in Eq. 3.30 and leads to a 0.23 probability of a missed detection, i.e., $P_m = 0.23$.

Note that even though we have relaxed the performance objectives significantly compared to the ones in Scenario 1, we still have to have more states and ping more often. The reason for this extra cost comes from the fact that the adversary in this case gets faster and the detection scheme has to spend relatively more energy combating the adversary.

Again imposing the *q-node* quorum pinging scheme with the corresponding parameters for its underlying pairwise scheme derived from Eq. 3.31 shows the significant gain compared to the pairwise case.

Fig. 3.14 shows the value of $P_m^{(q)}$ for different values of the quorum size q. The left graph is the actual values of $P_m^{(q)}$ for the corresponding value of q and the right graph is the logarithmic values of $P_m^{(q)}$ to show a better picture for the smaller values of q. Similar to the discussion given in Scenario 1, we have the same trend of increasing $P_m^{(q)}$ by increasing the quorum size q. Note that as Fig. 3.14 shows, we have some gain in the probability of a missed detection for the cases in which $q \leq 14$ and in particular for $q \leq 10$, this measure is reasonably negligible.



(a) Actual value of $P_m^{(q)}$ versus q.

(b) Logarithm of $P_m^{(q)}$ versus q.

Figure 3.14: Probability of a missed detection for *q-node* quorum pinging scheme w.r.t. the one in the pairwise pinging scheme for various quorum size q.

To demonstrate the projected gain for the false alarm rate in the quorum scheme, we try to find the probability of false alarm $P_{F.A.}^{(q)}$ based on the same argument in Scenario 1. To do so, we need to have the distribution of the residual time-to-false-alarm for any state, T_s in the underlying pairwise scheme and their corresponding *pmf* P_s. Again, these quantities are derived from Eqs.

3.16 and 3.9, respectively. Fig. 3.15 depicts the residual time-to-false-alarm T_s given in state s for the corresponding pairwise case with the parameters described in Eq. 3.29.

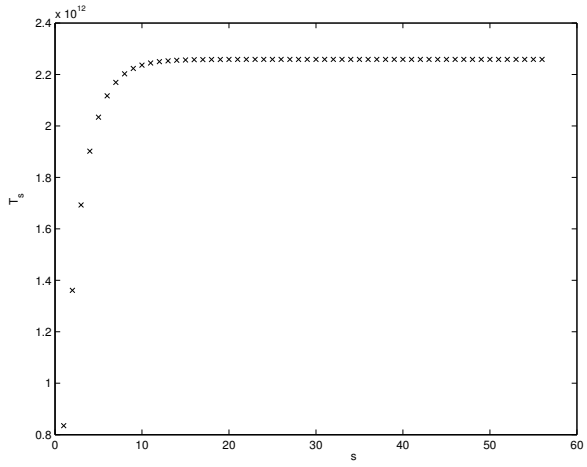
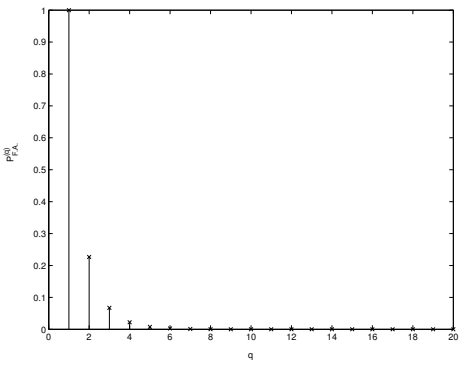


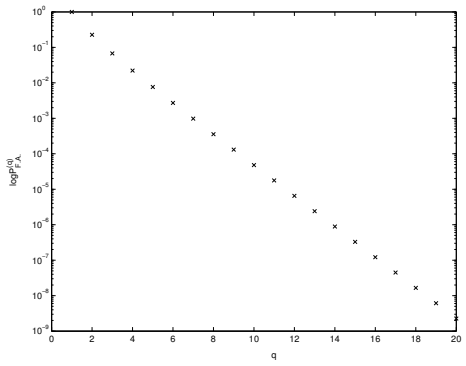
Figure 3.15: Residual time-to-false-alarm T_s given in state s in Scenario 2.

Fig. 3.16 illustrates the probability of false alarm $P_{F.A.}^{(q)}$. Again, for clarification we put the actual values on the left graph and the logarithmic values on the right graph to get a sense of its reduction rate with respect to q.

Again the achieved gain with respect to false alarm is significant once we have $q \geq 5$ in this example. Therefore, the gain of using the quorum scheme as opposed to the initial pairwise pinging scheme is governed by our proper choice of the quorum size q which is $5 \leq q \leq 10$ in this example in order to get a negligible probability of a missed detection and also a negligible probability of false alarm.



(a) Actual value of $P_{F.A.}^{(q)}$ versus q.



(b) Logarithm of $P_{F.A.}^{(q)}$ versus q.

Figure 3.16: Probability of false alarm $P_{F.A.}^{(q)}$ for quorum pinging scheme with various quorum size q.

Scenario 3. In this example we try to discover the effects of enhanced physical protection of nodes, strong obfuscation techniques, or low skill level of an adversary in capturing a target node in a WSN. All these circumstances would lead to a longer time for the adversary to complete its capture process on a target node.

Compared to Scenario 1, when the adversary takes longer to complete its capture, we expect that the situation would be easier for combating the adversary. The immediate influence of such circumstances on the parameters of our scheme is on the increased value of η_x. Here we consider an adversary which is capable of completing the capture process of its target node on average 4 times slower than the one of Scenario 1. In summary, the parameters of the underlying pairwise pinging scheme for this Scenario are as follows:

$$
\begin{aligned}
d &= 20 \\
p_l &= 10^{-3} \\
T_e &= 5 \ sec \\
\eta_x &= 800 \ sec
\end{aligned}
\tag{3.32}
$$

Here we consider the same performance requirements as in Scenario 1 in terms of the time-to-false-alarm and the probability of a missed detection.

$$
\begin{aligned}
P_m &\leq 0.2 \\
L_f &\geq 10^7 \,\text{sec} \approx 116 \ days
\end{aligned}
\tag{3.33}
$$

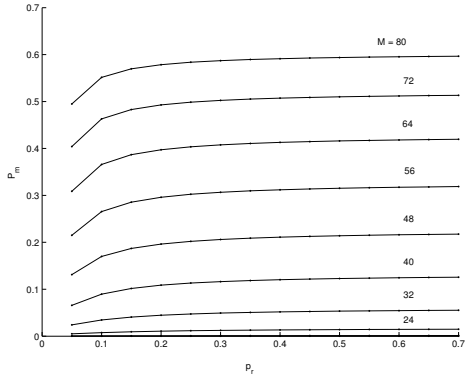
Following the results depicted in Fig. 3.17 leads to 48 states and a 0.17 pinging rate for the proposed pairwise pinging scheme which is shown in Eq. 3.34.

$$
\begin{aligned}
M &= 48 \\
p_r &= 0.17
\end{aligned}
\tag{3.34}
$$

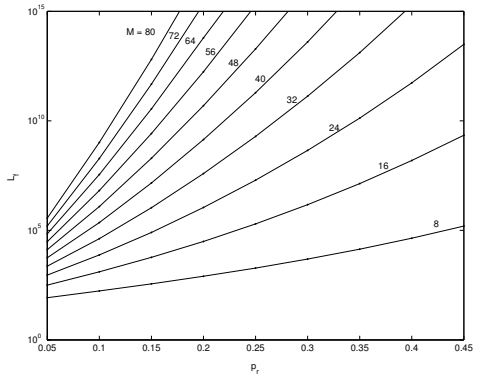
Selecting the parameters in Eq. 3.34 meets the inequalities in Eq. 3.33 and leads to a 0.18 probability of a missed detection precisely, i.e., $P_m = 0.18$. This result shows that the pairwise scheme needs to spend less energy in combating the slower adversary compared to the one in Scenario 1.

Imposing the *q-node* quorum pinging scheme with the corresponding parameters for its underlying pairwise scheme derived from Eq. 3.34 shows the significant gain compared to the pairwise case.

Fig. 3.18 shows the value of $P_m^{(q)}$ for different values of quorum size q.



(a) Probability of a missed detection P_m w.r.t. p_r and M.

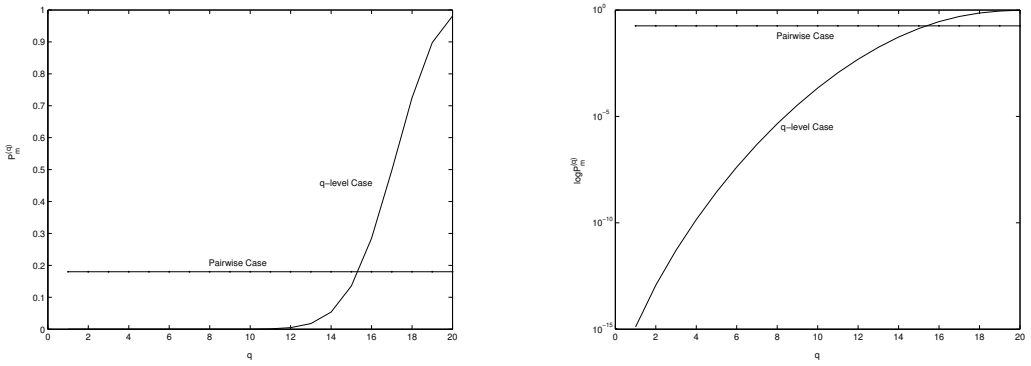


(b) Expected residual time-to-false-alarm L_f w.r.t. p_r and M.

Figure 3.17: Relevant graphs to design pairwise pinging scheme parameters for Scenario 3.

The left graph is the actual values of $P_m^{(q)}$ for corresponding q and the right graph is the logarithmic values of $P_m^{(q)}$ to show a better picture for smaller values of q. Similar to the discussion given in Scenario 1, we have the same trend of increasing $P_m^{(q)}$ by increasing quorum size q. Note that as Fig. 3.18 shows, we have some gain in the probability of a missed detection for the cases in which $q \leq 15$ and in particular for $q \leq 11$, this measure is reasonably negligible.

To demonstrate the projected gain for the false alarm rate in the quorum scheme, we try to find the probability of false alarm $P_{F.A.}^{(q)}$ based on the same argument in Scenario 1. To do so, we need to have the distribution of residual time-to-false-alarm given in any states T_s in the underlying pairwise scheme and their corresponding *pmf* P_s. Again, these quantities are derived from Eqs. 3.16 and 3.9, respectively. Fig. 3.19 depicts the residual time-to-false-alarm T_s given in state s for the corresponding pairwise case with the



(a) Actual value of $P_m^{(q)}$ versus q.

(b) Logarithm of $P_m^{(q)}$ versus q.

Figure 3.18: Probability of a missed detection for *q-node* quorum pinging scheme w.r.t. the one in the pairwise pinging scheme for various quorum size q.

parameters described in Eq. 3.32.

Fig. 3.20 illustrates the probability of false alarm $P_{F.A.}^{(q)}$. Again, for clarification we put the actual values on the left graph and the logarithmic values on the right graph to get a sense of its reduction rate with respect to q.

Again the achieved gain with respect to false alarm is significant once we have $q \geq 3$ in this example. Therefore, the gain of using the quorum scheme as opposed to the initial pairwise pinging scheme is governed by our proper choice of the quorum size q, which is $3 \leq q \leq 11$ in this example, in order to get a negligible probability of a missed detection and also a negligible probability of false alarm.

Scenario 4. Now we try to investigate the cost-performance trade-offs of the pairwise pinging scheme and performance compensation using the quorum

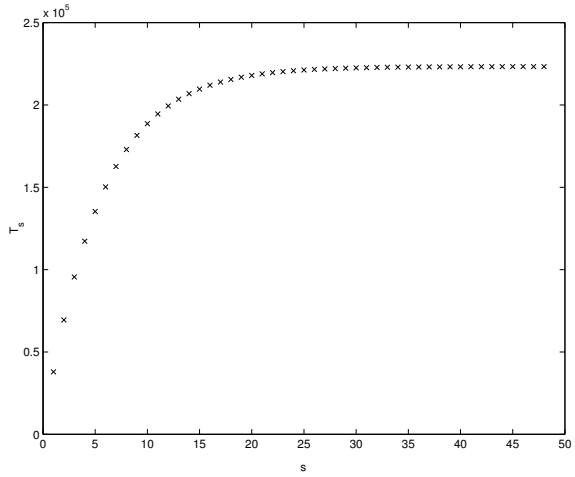


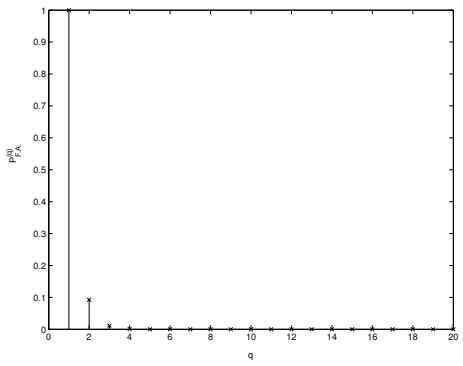
Figure 3.19: Residual time-to-false-alarm T_s given in state s in Scenario 3.

pinging scheme. It is clear that by relaxing the performance objectives, we can reduce the costs associated with the pairwise pinging scheme in order to meet those requirements. To show this issue, we set the same parameters for the pairwise scheme that we did in Scenario 1 as shown in Eq. 3.35.

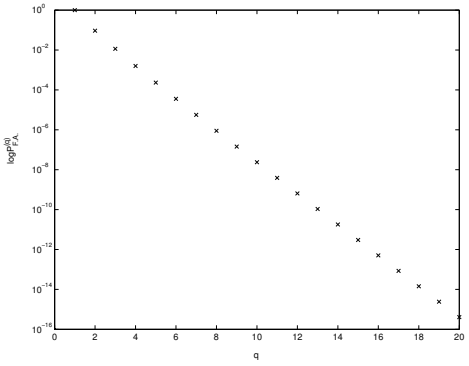
$$\begin{aligned} d &= 20 \\ p_l &= 10^{-3} \\ T_e &= 5 \ sec \\ \eta_x &= 400 \ sec \end{aligned} \tag{3.35}$$

Instead we have relaxed the performance measures including the probability of a missed detection and expected residual time-to-false-alarm compared to the ones in Scenario 1 as follows:

$$\begin{aligned} P_m &\leq 0.25 \\ L_f &\geq 10^4 \,\text{sec} \approx 2.8 \ hours \end{aligned} \tag{3.36}$$



(a) Actual value of $P_{F.A.}^{(q)}$ versus q.



(b) Logarithm of $P_{F.A.}^{(q)}$ versus q.

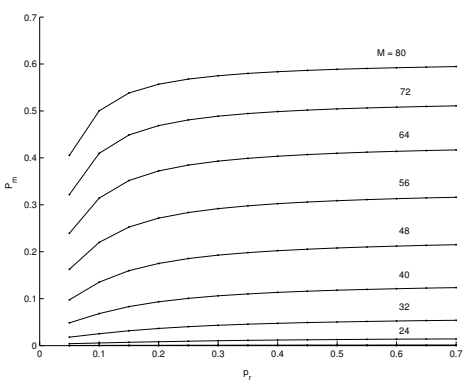
Figure 3.20: Probability of false alarm $P_{F.A.}^{(q)}$ for the quorum pinging scheme with various quorum size q.

Following the results depicted in Fig. 3.21 leads to 56 states and a 0.09 pinging rate for the proposed pairwise pinging scheme which is shown in Eq. 3.37.

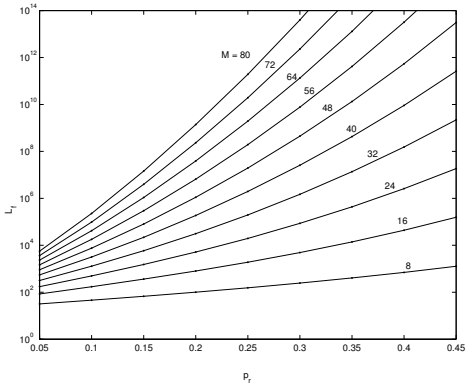
$$\begin{aligned} M &= 56 \\ p_r &= 0.09 \end{aligned} \tag{3.37}$$

Selecting the parameters in Eq. 3.37 meets the inequalities in Eq. 3.36 and leads to a 0.21 probability of a missed detection, i.e., $P_m = 0.21$. Observe that by relaxing the performance measures, we have gained significant cost reduction associated with the pairwise pinging scheme. This is because while the required number of states is increased from 48 to 56, the pinging rate is dropped from 0.23 to 0.09. Note that the majority of cost is directly associated with the required pinging rate p_r.

Again imposing the *q-node* quorum pinging scheme with the correspond-



(a) Probability of a missed detection P_m w.r.t. p_r and M.



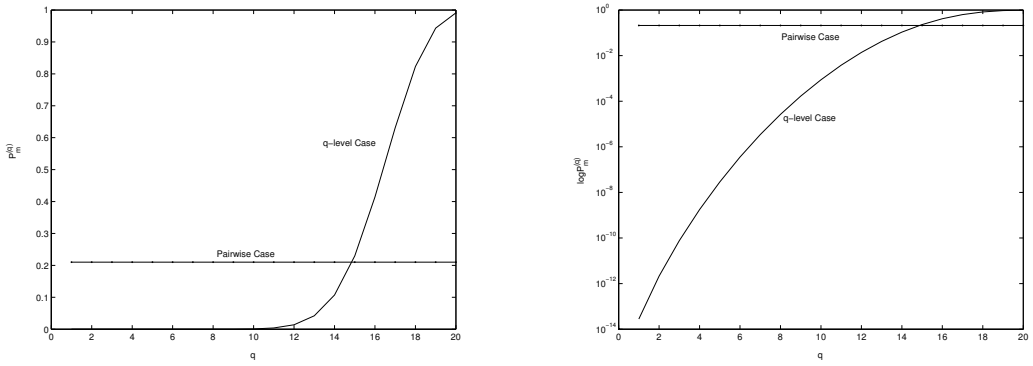
(b) Expected residual time-to-false-alarm L_f w.r.t. p_r and M.

Figure 3.21: Relevant graphs to design pairwise pinging scheme parameters for Scenario 4.

ing parameters for its underlying pairwise scheme derived from Eq. 3.37 shows the significant gain compared to the pairwise case.

Fig. 3.22 shows the value of $P_m^{(q)}$ for different values of quorum size q. The left graph is the actual values of $P_m^{(q)}$ for its corresponding q and the right graph is the logarithmic values of $P_m^{(q)}$ to show a better picture for smaller values of q. Similar to the discussion given in Scenario 1, we see the same trend of increasing $P_m^{(q)}$ by increasing quorum size q. Note that as Fig. 3.22 shows, we have some gain in the probability of a missed detection for the cases in which $q \leq 14$ and in particular for $q \leq 10$, this measure is reasonably negligible.

To demonstrate the projected gain for the false alarm rate in the quorum scheme, we try to find the probability of false alarm $P_{F.A.}^{(q)}$ based on the same argument in Scenario 1. To do so, we need to have the distribution of the



(a) Actual value of $P_m^{(q)}$ versus q.

(b) Logarithm of $P_m^{(q)}$ versus q.

Figure 3.22: Probability of a missed detection for the *q-node* quorum pinging scheme w.r.t. the one in the pairwise pinging scheme for various quorum size q.

residual time-to-false-alarm given in any state, T_s in the underlying pairwise scheme and their corresponding *pmf* P_s. Again, these quantities are derived from Eqs. 3.16 and 3.9, respectively. Fig. 3.23 depicts the residual time-to-false-alarm T_s given in state s for the corresponding pairwise case with the parameters described in Eq. 3.35.

Fig. 3.24 illustrates the probability of false alarm $P_{F.A.}^{(q)}$. Again, for clarification we put the actual values on the left graph and the logarithmic values on the right graph to get a sense of its reduction rate with respect to q.

Again the achieved gain with respect to false alarm is significant once we have $q \geq 3$ in this example. Therefore, the gain of using the quorum scheme as opposed to the initial pairwise pinging scheme is governed by our proper choice of the quorum size q, which is $3 \leq q \leq 10$ in this example, in order to get a negligible probability of a missed detection and also a negligible

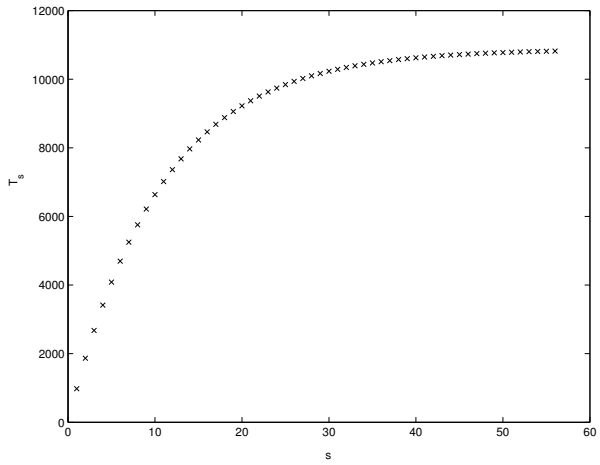
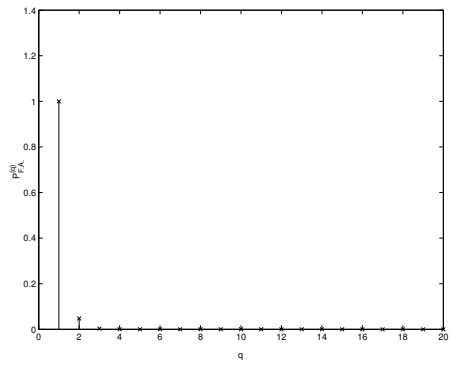


Figure 3.23: Residual time-to-false-alarm T_s given in state s in Scenario 4.

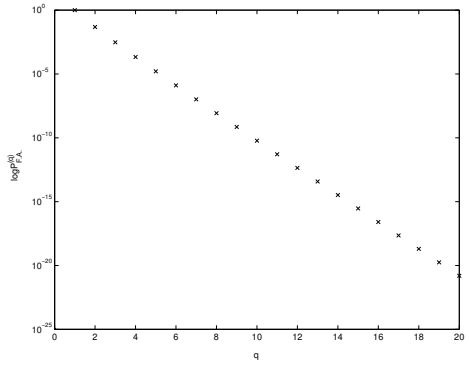
probability of false alarm.

As a result, we see that by relaxing the performance measures for the initial pairwise pinging scheme, we could save significantly in costs. While using the proper quorum size q in the quorum scheme, we could improve the overall performance measures including the probability of a missed detection $P_m^{(q)}$ and the probability of false alarm $P_{F.A.}^{(q)}$.

Remark 7 It is quite interesting to note that among all these different scenarios the required quorum size q is in the same relative range in order to meet the desired performance measures such as overall probability of a missed detection $P_m^{(q)}$ and probability of false alarm $P_{F.A.}^{(q)}$. This quorum size is proportional to the average node degree d of the network. As a rule of thumb we can pick q to be 0.25 d. This selection of q implies that we need about 25 *percent* of a node's neighbors reaching the same judgement about a suspicious node in order to have a reliable consensus in a quorum.



(a) Actual value of $P_{F.A.}^{(q)}$ versus q.



(b) Logarithm of $P_{F.A.}^{(q)}$ versus q.

Figure 3.24: Probability of false alarm $P_{F.A.}^{(q)}$ for the quorum pinging scheme for various quorum sizes q.

3.3 Cost Analysis

In this section, we address the costs imposed by the proposed schemes. These costs include communication, computation, and memory requirements of the corresponding schemes. Note that the parameter selections associated with each scheme is done offline and there is no need for the network nodes to perform the design procedures of the proposed protocols.

First, we consider the pairwise pinging scheme which can be run between each pair of neighbors independently. To perform this scheme, each node has to run a separate *Markov Chain* for each of its neighbors. This requires a ping message exchange between each pair within each epoch with the pinging probability p_r. This leads to a communication cost of $p_r d/T_e$ *messages per sec.* For each *Markov Chain* run against each neighbor of a node, the node has to keep a state counter associated with each neighbor resulting in d *reg-*

isters per node. Computation costs include a random variable generation per epoch, MAC generations and verifications. Within each epoch, a node has to generate a *uniform* random variable between $[0, 1]$ interval and compares it with p_r in order to decide whether it wants to ping its neighbors in the current epoch or not. Moreover, each node needs to generate d ping messages for all its neighbors with probability p_r within each epoch which includes d number of MAC generations. On the other hand, once the node decides to ping its neighbors it expects to receive the responses for all sent ping messages within the current epoch. However, it may or may not receive all the expected ping responses depending on the status of each neighbor. At most, it has to verify d number of MACs in the pinging response messages. It is worth noting here that attackers cannot launch bogus responses to deplete the resources of the node due to the fact that the node only verifies those responses which it initiates itself in the appropriate intervals of the current epoch. Here we put more loads from the scheme in the computational segments in exchange for reducing the required communication costs as the communication costs are more restrictive than computational ones.

Secondly, the costs of the *q-node* quorum pinging scheme are as follows. In this case even though each node runs an independent pairwise pinging scheme with each of its neighbor, the results of its judgements and the judgements of the other nodes in the neighborhood regarding their neighbors are recorded in counter registers dedicated for each of their neighbors and kept on each individual node. Once a node reaches a judgement about one of its neighbors, namely that, from its own perspective its neighbor has been captured, it raises a flag and broadcasts it to the neighborhood. Those neighbors

which receive this flag would increment the counter register associated with the target node. Each flag has a lifetime of T. In each of the neighbors of the target node, each flag is kept for its own lifetime interval and if the counter reaches the quorum size q, the neighbors that experience this event raise a revocation flag and commit to that flag and broadcast it across the network. Otherwise, once each flag is expired, each neighbor individually decrements the counter of the target node based on its own clock. The interesting property of the proposed quorum scheme lies in the following observation. Compared to the pairwise case, this scheme does not impose any additional communication costs other than the occasional broadcasting of the raised flags. Instead, we can get significant gains in performance compared to the pairwise scheme with the same parameters using emergent properties in the neighborhoods. Note that as far as cost is concerned, the communication costs are the dominant prohibitive factors in WSN applications. Here, we have more or less the same communication costs and relatively small computation costs for keeping track of raised and expired flags and the counter registers. To give a rough idea of the overall costs in a practical setting, we present the required costs for the case presented in Scenario 1. The quorum scheme imposes the communication load of about 0.92 messages per node per second and 40 registers per node for storing the state values and counters for its neighbors.

Chapter 4

Capture Strategies by Adversary's Agents

In contrast to the attack scenarios discussed in the previous chapter, which was launched by the adversary, here we consider the case in which the adversary deploys multiple agents in the network in order to capture as many active network nodes as possible. In the previous chapter the main objective was to detect the presence of a single adversary agent which is trying to capture a network node. This task can be performed by the participation of, at least one, and up to q neighbors of the target node under capture, as we seen before, by engaging into a particular variation of a so called pinging scheme. Now, we try to generalize our model in the sense that the adversary has multiple agents in hand and is able to target various network nodes using those agents.

In order for the adversary to gain a better capture result, namely more captured nodes, it has to use a specific strategy to distribute its agents against

the network. An ideal strategy is the one which guarantees that with a fixed number of agents, the adversary can capture the maximum possible number of nodes. So, the goal is to achieve the optimal capture strategy for adversary agents. This objective requires various classifications of capture strategies; including random versus targeted distribution of agents, and the order the network nodes should be captured (e.g., sequential, random, or simultaneously).

Definition 8 Collusion is communication among adversary agents which preside over targets under capture or over already captured targets.

The act of collusion includes any exchange of information among agents for the purpose of defeating the underlying pinging scheme. The ultimate goal of collusion is to capture more nodes successively through additional adversary's agents using the harvested information from colluders. The communication can be performed via hybrid channels as agents are physically present on the target nodes and have access to their own resources as well. Due to the way that the *q-node* quorum pinging scheme is set up, once an agent targets a specific node and starts to capture it, it can broadcast this event to the rest of the agents through those communication links. This exchange of information is called collusion, which can potentially improve the adversary's capture results.

We shall investigate whether agent collusion yields better capture results (i.e., more nodes) than independent, random capture. We will analyze our model and characterize the countermeasures that are necessary for addressing the adversary's strategies defined earlier.

4.1 Capture Strategy for Specific Targets

4.1.1 Modeling of Capture Strategy for Specific Targets

In this trivial case, the objective of each adversary's agent is to maximize the probability of success for capturing a specific target node. Obviously without loss of generality and for the sake of brevity, we will only analyze the single specific target case here.

The adversary's success to capture a target node is equivalent to missing a capture event in the underlying *q-node* quorum pinging scheme. Therefore, the probability of success for each adversary's agent is equal to $P_m^{(q)}$. $n_A^{(j)}$ denotes the number of collaborative agents accompanying the j^{th} adversary's agent. N_A represents the total number of adversary agents which can be written in terms of $n_A^{(j)}$ as follows:

$$N_A = \sum_j n_A^{(j)} \tag{4.1}$$

As defined earlier, the average node degree of each node is d and q is the quorum size of the underlying *q-node* quorum pinging scheme. Hence, we can evaluate the effective node degree of each node, d_{eff} as follows:

$$d_{eff} = d - n_A + 1 \tag{4.2}$$

Note that in Eq. 4.2, the dependency of each node index is implicitly reflected on the actual value of d. However, as we are here concerned about achieving an average sense of the behavior under this scenario, we ignore the

node level dependency. Later on when needed, we will discuss per node analysis.

Obviously the relative values of q and d_{eff} of the target node dictate the success event for the adversary's agents. If $q > d_{eff}$, then the adversary is successful in capturing the target node with probability 1, which in turn implies that the number of agents that capture successfully is maximized and equals the total number of agents that are initially engaged in the attack. However, if $q \ll d_{eff}$, then the adversary is unsuccessful with probability 1. Similarly, for the multiple agent case under this condition, all the participated agents will have failed in capturing their corresponding targets. This leads to a minimum number of captured targets, which in this case is 0.

4.1.2 Countermeasures on Capture Strategy for Specific Targets

Therefore, as a defense mechanism against an envisioned total number of adversary's agents, N_A that the network might encounter, we have to maximize the average node degree, d such that $q \ll d_{eff}$ holds in order to be able to counter the maximum probability of success for capturing specific target nodes by an adversary's agents.

4.2 Optimal Capture Strategy

4.2.1 Modeling of Optimal Capture Strategy

In this case, an adversary's objective is to maximize the expected number of agents that succeed in capturing nodes, given the total number of deployed agents is N_A.

Here, A_S denotes the number of agents that succeed in capturing their corresponding target nodes. So we can write the objective cost function as follows:

$$maxE[A_S] \tag{4.3}$$

In order to evaluate this cost function, we have to simplify it as much as possible. To do so, we first expand the expected value of A_S.

$$E[A_S] = [p(A_S = 0) * 0] + [p(A_S = 1) * 1] + [p(A_S = 2) * 2] + [p(A_S = 3) * 3] + ... \\ ... + [p(A_S = i) * i] + ... + [p(A_S = N_A) * N_A] \tag{4.4}$$

In Eq. 4.4, we see that for $i \neq 0$ the term $p(A_S = i)$ is proportional to $[P_m^{(q)}]^i$. This comes from the fact that the event where $A_S = i$ corresponds to i successful capture events performed by i agents independently, each having a probability $P_m^{(q)}$. This implies that for practical values of $P_m^{(q)}$, $[p(A_S = i) * i]$ goes to zero rapidly for $i \geq 2$. Thus, Eq. 4.4 can be rewritten as:

$$E[A_S] \approx p(A_S = 1) \tag{4.5}$$

Substituting Eq. 4.5 into the cost function of Eq. 4.3 results in the following optimization problem:

$$maxE[A_S] \approx max[p(A_S = 1)] \tag{4.6}$$

Now we need to find $p(A_S = 1)$ in Eq. 4.6 in order to be able to get the optimal capture strategy. $p(A_S = 1)$ means the probability of the event in which just one out of N_A adversary agents becomes successful in its capture process. This agent can be any one of the N_A deployed agents and the rest of them should necessarily be unsuccessful. Fig. 4.1 depicts a fraction of the network with 4 agents targeting different nodes of the network. Red links denote the communication links corresponding to possible collusion among agents.

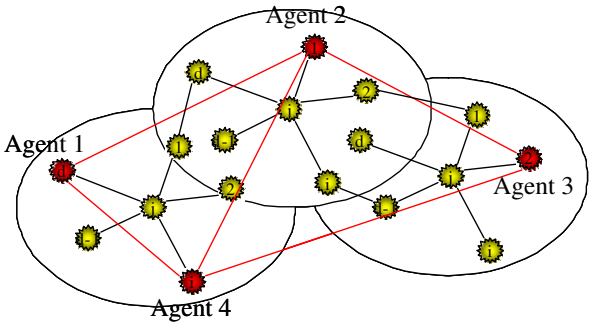


Figure 4.1: A fraction of a network with sensor nodes and 4 adversary agents.

By defining $P_m^{(q),(d_{eff}^j)}$ as the probability of a missed detection for the j^{th} adversary's agent as follows, we plan to evaluate $p(A_S = 1)$ in terms of the available measurable quantities, namely probabilities of missed detection.

$$P_m^{(q),(d_{eff}^j)} = \binom{d_{eff}^j}{d_{eff}^j - q + 1} P_m^{d_{eff}^j - q + 1} (1 - P_m)^{q-1} + \binom{d_{eff}^j}{d_{eff}^j - q + 2} P_m^{d_{eff}^j - q + 2} (1 - P_m)^{q-2} + ...$$

$$... + \binom{d_{eff}^j}{d_{eff}^j - q + i} P_m^{d_{eff}^j - q + i} (1 - P_m)^{q-i} + ... + \binom{d_{eff}^j}{d_{eff}^j} P_m^{d_{eff}^j} \qquad 1 \leq j \leq N_A \tag{4.7}$$

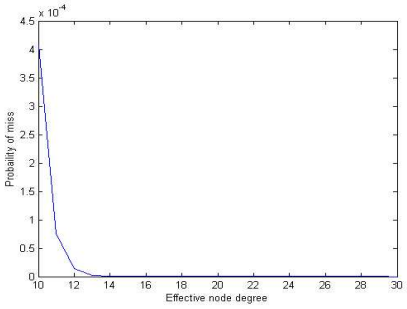
in which d_{eff}^j denotes the effective node degree of the node targeted by the j^{th} agent. $p(A_S = 1)$ can be written as the union of all possible choices of a success event for one of the agents and unsuccess events for the remaining agents. This can be evaluated as follows:

$$\begin{aligned} p[A_S = 1] = &\left\{ P_m^{(q),(d_{eff}^1)} [1 - P_m^{(q),(d_{eff}^2)}][1 - P_m^{(q),(d_{eff}^3)}]...[1 - P_m^{(q),(d_{eff}^{N_A})}] \right\} + \\ &\left\{ P_m^{(q),(d_{eff}^2)} [1 - P_m^{(q),(d_{eff}^1)}][1 - P_m^{(q),(d_{eff}^3)}]...[1 - P_m^{(q),(d_{eff}^{N_A})}] \right\} + \\ &... \\ &\left\{ P_m^{(q),(d_{eff}^{N_A})} [1 - P_m^{(q),(d_{eff}^1)}][1 - P_m^{(q),(d_{eff}^2)}]...[1 - P_m^{(q),(d_{eff}^{N_A - 1})}] \right\} \end{aligned} \tag{4.8}$$

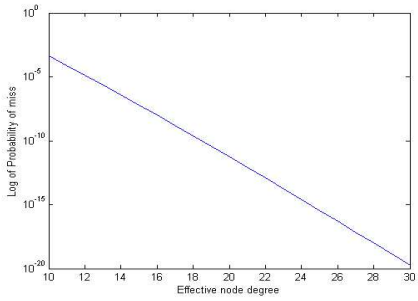
To elaborate on the behavior of $p(A_S = 1)$ with respect to its parameters, we try to find the dominant terms in Eq. 4.8 and simplify the evaluation of $p(A_S = 1)$. As $P_m^{(q),(d_{eff}^j)} \ll 1$, we can approximate Eq. 4.8 as follows:

$$p[A_S = 1] \approx P_m^{(q),(d_{eff}^1)} + P_m^{(q),(d_{eff}^2)} + ... + P_m^{(q),(d_{eff}^j)} + ... + P_m^{(q),(d_{eff}^{N_A})} \tag{4.9}$$

To proceed in our analysis, we track the variations of $P_m^{(q),(d_{eff}^j)}$ in terms



(a) Actual value of $P_m^{(q),(d_{eff}^j)}$ versus d_{eff}^j



(b) Logarithm of $P_m^{(q),(d_{eff}^j)}$ versus d_{eff}^j

Figure 4.2: Sensitivity of $P_m^{(q),(d_{eff}^j)}$ w.r.t. d_{eff}^j

of d_{eff}^j over its feasible range. The result is shown in Fig. 4.2 which, in turn, implies the exponential decaying dependency of $P_m^{(q),(d_{eff}^j)}$ with respect to d_{eff}^j, which means that the dominant terms in the summation in Eq. 4.9 correspond to the ones with the lowest d_{eff}^j. This observation leads us to the following conclusion:

Lemma 2 *Maximizer to Eq.4.6 is equal to the minimum effective node degree d_{eff}^j. In other words, the maximum is achieved when the adversary agent targets the node with the minimum effective node degree. This can be formulated as follows:*

$$arg\,\{maxE[A_S]\} = min(d_{eff}^j) \qquad 1 \leq j \leq N_A \tag{4.10}$$

Remark 8 Note that in our argument in this case, we deliberately drop the dependency of the effective node degree d_{eff}^j to each node index for the sake of notational simplification. It is quite clear that each node can have a different node degree d individually in Eq. 4.2 and consequently, the resulting effective

node degree d_{eff} would also be dependent on each individual node index. The unbalanced node degree among nodes can occur due to a variety of reasons ranging from the earlier capture of a neighbor of a particular node, to some dead neighbors of a particular node due to battery depletion, malfunction, etc.

Lemma 2 leads to the optimal capture strategy for the adversary's agents in a way that the adversary distributes its agents to the potential target nodes with the lowest effective node degree, i.e., lowest d_{eff}, and to their surrounding nodes simultaneously. Those nodes with the lowest effective node degree constitute the optimal targets for capture and their surrounding nodes collaborate in the capture process of optimal targets to increase their probability of success. The primary objective of the agents that are deployed in the neighborhood of the main target is to help the capture of the main target, though each of these collaborative agents can have their own chances of success in capture of their direct targets with the corresponding probabilities. Needless to say that these probabilities would be much smaller than the ones for the main targets due to the fact that for the latter the probability is boosted by the support of additional agents in the neighborhood while in the former case, the agents are left alone in capturing their prospective targets. In the next step, agents proceed sequentially to target other nodes based on the immediate lowest value of d_{eff} and continue to follow this procedure for the current target with lowest d_{eff} at a time. At each step of capturing a main target node, at least 1 agent is deployed on the main target and a few agents are needed per each main target deployed as a batch in its neighborhood. Therefore, this procedure continues until the adversary uses all its available agents. We will discuss the required number of collaborative agents in details

in the following paragraphs.

Remark 9 The initial phase of deployment of an adversary's agents can be executed either based on the local knowledge of the node degree distribution d or purely at random. Then for the successive deployment of agents, according to Eq. 4.2, the adversary gets the lowest available effective node degree d_{eff} in the neighborhood of the already deployed agents. This notion of collusion among an adversary's agents can be performed due to the fact that in the *q-node* quorum pinging scheme, once an agent starts its capture process on a target, the target is unable to interact with its immediate neighbors. Consequently, the effective node degree d_{eff} of all the neighbors of the target under capture will be affected and decremented by 1. This information can be transferred in advance to the rest of the agents prior to capture for the purpose of exploring the lowest immediate effective node degree d_{eff} for the next agent deployment. In this fashion, the already deployed agents gather and exchange their information and try to help new agents to get better results.

Remark 10 The knowledge of node degree assumption provides a worst case analysis of the model. As shown in Lemma 2, once the adversary knows the node degree distribution of the network, it can target the node with the lowest degree, which is the most vulnerable. In contrast, if the adversary does not know this information, there is a possibility that it will target some nodes with higher degrees, which in turn reduces the adversary's probability of success. Therefore, the availability of node degree distribution implies the worst case analysis. Rest assured, if we design the system in a way that under such a scenario the desired measures can be met, then for any other cases the system definitely outperforms.

To show the collaborative effect of a concentrated capture phase among the targeted neighbors on capturing the main target with the lowest effective node degree, we focus on the *right-hand-side* of Eq. 4.10. Minimizing d_{eff} for any given node using Eq. 4.2 implies increasing the number of agents, n_A on its surroundings. To that effect, the adversary tries to distribute as many additional collaborative agents as possible around the main target node with lowest d_{eff}, in order to maximize the probability of success of capture, which can be written as follows:

$$P_m^{(q),(d_{eff})} = \binom{d_{eff}}{d_{eff}-q+1} P_m^{d_{eff}-q+1}(1-P_m)^{q-1} + \binom{d_{eff}}{d_{eff}-q+2} P_m^{d_{eff}-q+2}(1-P_m)^{q-2} + ...$$

$$... + \binom{d_{eff}}{d_{eff}-q+i} P_m^{d_{eff}-q+i}(1-P_m)^{q-i} + ... + \binom{d_{eff}}{d_{eff}} P_m^{d_{eff}} \quad \text{s.t.} \quad d_{eff} = d - n_A + 1 \tag{4.11}$$

Note that for a desired value of the probability of success for an adversary in a node capture, it may not necessarily require using all its resources, i.e., all its agents deployed for one target node. Rather, it might be sufficient enough to meet the desired $P_m^{(q),(d_{eff})}$ by partially distributing some of its agents for capturing the most immediate target, namely, the one with the lowest d_{eff} and its surroundings, and reserve the remaining agents for capturing the next immediate targets.

Comparing Eqs. 4.7 and 4.11, we realize that the two behave the same, and consequently we can use Fig. 4.2 as a reference of the behavior of $P_m^{(q),(d_{eff})}$ with respect to its parameter d_{eff}. So, we know that $P_m^{(q),(d_{eff})}$ is a convex function of d_{eff} and due to its exponential regime, $f(a+b) \gg f(a) + f(b)$. This

implies that, in terms of the probability of success for capture, the adversary gains exponentially by concentrating its agents on the surrounding of a desired main target, rather than scattering them sparsely across various targets. We show this in the following example.

Suppose we have two distinct targets fairly far from each other in the network. Each of them has a node degree of 15. The adversary has 6 agents available in its hand for deployment. One strategy for the adversary is to try to capture just one of its targets. To do so, it deploys one of its agents on the main target and the remaining 5 agents on the target's neighbors. In this way the effective node degree of the main target would be $d_{eff} = 15 - 5 = 10$. The other adversary's strategy is to capture both targets by deploying one agent on each of them and the remaining 4 agents equally distributed on their neighbors. Following this strategy, each main target ends up having the effective node degree of $d_{eff} = 15 - 2 = 13$. Using the corresponding values of $P_m^{(q),(d_{eff})}$ in terms of d_{eff} from Fig. 4.2, we find the expected value of the number of successful agents as follows:

$$E[A_S] = p(A_S = 1) = 10^{-3} \qquad \text{for the first strategy} \tag{4.12}$$

while,

$$E[A_S] = p(A_S = 1) = 10^{-6} + 10^{-6} = 2 * 10^{-6} \qquad \text{for the second strategy} \tag{4.13}$$

Obviously, the numerical results shown on Eqs. 4.12 and 4.13 justifies the significant exponential gain for the adversary in distributing its agents as concentrated as possible to its desired targets similar to the one discussed in the first strategy of the above example. However, we have to point out that once the desired adversary's objective in terms of probability of success is met, it is unnecessary to overinvest and deploy extra agents in the main target's neighborhood. It would be wise to save the remaining agents to capture the next immediate targets.

Based on the detailed analysis of the optimal capture strategy presented in this section, Table 4.1 summarizes the required steps as to how and in what order the adversary needs to distribute its agents on the network nodes in order to capture them.

Desired probability of success $P_m^{(q),(d_{eff})}$ for an adversary is given. Maximum available number of adversary's agents is N_A. Distribution of node degree d among nodes is imposed by network topology. Adversary agent targets a node with the lowest d_{eff} to capture. Simultaneously a batch of agents accompany that agent in its neighborhood. The size of the batch depends on the improvement factor of $P_m^{(q),(d_{eff})}$. For successive capture, another agent targets a node with the next lowest d_{eff}. This procedure continues sequentially till all agents are used.

Table 4.1: Summary of optimal capture strategy procedure

4.2.2 Countermeasures on Optimal Capture Strategy

Again as a defense mechanism against the maximum envisioned number of an adversary's agents, the designer needs to push the effective node degree d_{eff} of network nodes higher. This should be done primarily for those node with a lower d_{eff} because they are more vulnerable to successful capture by an adversary's agents and for the same reason are subject to initial capture through an optimal strategy. However, such nodes may not be identifiable *a priori* and as a result we need to boost the average node degree across the network. Therefore, to counter against the adversary, the designer has to increase the average node degree d of the network, such that for the worst case scenario for which a node may encounter the lowest node degree and highest number of adversary's agents deployed in its neighborhood, the underling *q-node* quorum pinging scheme will still be able to meet the desired performance metric, namely the maximum tolerable probability of a missed detection $P_m^{(q),(d_{eff})}$, or equivalently the probability of success for an adversary's agent of capturing a node.

Chapter 5

Conclusions and Future Research

5.1 Summary of Main Contributions

A common threat in many networks is the capture of network devices by an adversary. Stajano's *big stick principle*, which states that whoever has physical control of a device is allowed to take it over, suggests that such an adversary is more powerful than the Dolev-Yao and traditional Byzantine adversaries, and hence difficult to counter. Wireless sensor networks (WSNs), mesh networks, and embedded networks pose unique security challenges due to the fact that their nodes operate in an unattended manner in potentially hostile environments. A particularly difficult problem not addressed to date is the handling of node capture by an adversary. In this work we addressed these security challenges in WSNs due to the captured node problem. A key goal here is to limit the damage caused by captured nodes. This is important since node capture cannot be prevented. Hence, the presence of the adversary within a WSN must be detected, and of course, the earlier the better. We identified the

shortcomings of proposed methodologies dealing with the captured nodes and defined the minimum requirements of a desired design. Adversary detection is predicated on the fact that access to a captured node's internal state incurs a nonzero time delay. Our treatment for adversary detection is focused on the *in-capture* detection phase, namely detection before the adversary gets a chance to access a node's internal state and do any network damage.

In *in-capture* detection phase, we proposed two probabilistic schemes called the pairwise pinging scheme and quorum pinging scheme, whereby the network continuously monitors itself in a distributed and self-organizing manner. The intuition behind our approach is based on the prohibitive cost burden of continuous monitoring, whereas by invoking random monitoring in local neighborhoods we can attain a trade-off between cost and risk of node capture. We investigated the trade-offs between the network cost-performance and security of this scheme via a *Markov Chain* model, and presented analytical solutions for both cases which allow us to choose appropriate performance parameters, such as the expected residual time-to-false-alarm and probability of false alarm, and security, such as the probability of a missed detection. The quorum scheme is a generalization of the pairwise scheme by invoking at least q neighbors of any target node to reach to a consensus regarding the status of the target node. This scheme introduces a novel emergent property by which performance of the proposed detection mechanism is improved exponentially in exchange of linear cost increase in terms of quorum size q. A full version of this work is to appear in [47].

Later on, we investigated optimal strategies for deploying multiple ad-

versary agents in order to capture multiple target nodes. These strategies include which target nodes to pick, in what order they should be captured, and what pattern the distribution of adversary agents should be (e.g., sequential, random, simultaneously). The objective of the capture strategy is to maximize the number of successful node capture through adversary agents using fixed number of agents. We showed that using collusion among adversary agents can significantly improve their chances in capturing their targets determined by the optimal capture strategy. Finally, we evaluated that agent collusion yields better capture results than independent, random capture. We analyzed our model and characterized the necessary countermeasures against the optimal capture strategy used by adversary to deploy its agents. The results show significant resiliency of the underlying quorum pinging scheme to detect node capture despite collusion among adversary agents.

5.2 Directions for Future Research and Projected Applications

In this work we assumed that agents preside over their corresponding targets after their initial engagements. We would like to extend this scenario to the case that agents can hop around different target nodes after completion of their capture. Doing so, we can introduce the notion of rate of node capture over time and assess that metric as a measure of effectiveness for various capture strategies. A more generalized scenario to consider is the one in which both network nodes and adversary agents in real time can be inserted with certain rates to the network. In such highly dynamic cases, evaluation of node capture

should be merely judged based on rate of node capture.

Potential applications for our proposed detection mechanisms rely on the core assumptions of our scheme, i.e., dense network with unattended nodes. A relatively broad range of applications can fit into this class. To name a few more interesting cases, we can mention applications such as smart home security in urban area, structural health monitoring of pipeline, electrical grid networks, and automated surveillance system.

Bibliography

[1] Adrian Perrig, Robert Szewczyk, J. D. Tygar, Victor Wen, and David E. Culler. Spins: Security protocols for sensor networks. In *Wireless Networks*, 8(5):521–534, Sep. 2002.

[2] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings 9th ACM Conference on Computer and Communications Security, CCS'02*, pages 41–47, Nov. 2002.

[3] Frank Stajano. Security in pervasive computing. In *Lecture Notes in Computer Science*, 2802:257–264, Feb. 2004.

[4] D. Dolev and A. C. Yao. On the security of public key protocols. In *IEEE Transactions on Information Theory (IT)*, 29:198–208, 1983.

[5] M. Castro and B. Liskov. Authenticated byzantine fault tolerance without public-key cryptography. In *Massachusetts Institute of Technology*, 1999.

[6] Adrian Perrig, John Stankovic, and David Wagner. Security in wireless sensor networks. In *Communications of the ACM, Special Issue on Wireless sensor networks*, 47(6):53–57, June 2004.

[7] Federal information processing standards publication 140-1: Security requirements for cryptographic modules. In *U.S. National Institute of Standards and Technology*, 1994.

[8] G. Edward Suh and Srinivas Devadas. Physical unclonable functions for device authentication and secret key generation. In *Proceedings of the 44th annual conference on Design automation, DAC'07*, pages 9–14, 2007.

[9] Leonid Bolotnyy and Gabriel Robins. Physically unclonable function-based security and privacy in rfid systems. In *Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications, PERCOM'07*, pages 211–220, 2007.

[10] Srinivas Devadas, Edward Suh, Sid Paral Richard Sowell, Tom Ziola, and Vivek Khandelwal. Design and implementation of puf-based unclonable rfid ics for anti-counterfeiting and security applications. In *IEEE International Conference on RFID*, pages 58–64, April 2008.

[11] G. Edward Suh, Charles W. O'Donnell, Ishan Sachdev, and Srinivas Devadas. Design and implementation of the aegis single-chip secure processor using physical random functions. In *ACM SIGARCH Computer Architecture News*, 33(2):25–36, May 2005.

[12] Abdulrahman Alarifi and Wenliang Du. Diversify sensor nodes to improve resilience against node compromise. In *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks, SASN'06*, pages 101–112, 2006.

[13] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistri-

bution schemes for sensor networks. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy, Oakland, CA, USA*, pages 197–213, May 2003.

[14] W. Du, J. Deng, Y. Han, and P. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of the Tenth ACM Conference on Computer and Communications Security, CCS'03*, pages 42–51, 2003.

[15] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the Tenth ACM Conference on Computer and Communications Security, CCS'03*, pages 52–61, 2003.

[16] R. Blom. An optimal class of symmetric key generation systems. In *Proceedings of the Eurocrypt 84 workshop on Advances in cryptology: theory and application of cryptographic techniques*, pages 335–338, 1985.

[17] Haowen Chan, Virgil D. Gligor, Adrian Perrig, and Gautam Muralidharan. On the distribution and revocation of cryptographic keys in sensor networks. In *IEEE Transactions on Dependable and Secure Computing*, 2(3):233–247, July 2005.

[18] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig. The sybil attack in sensor networks: analysis and defenses. In *Proceedings of the third international symposium on Information processing in sensor networks*, pages 259–268, 2004.

[19] Sencun Zhu, Sanjeev Setia, Sushil Jajodia, and Peng Ning. An interleaved hop-by-hop authentication scheme for filtering of injected false data in

sensor networks. In *IEEE Symposium on Security and Privacy*, pages 260–272, 2004.

[20] Harald Vogt. Increasing attack resiliency of wireless ad hoc and sensor networks. In *Proceedings of the Second International Workshop on Security in Distributed Computing Systems, (SDCS) (ICDCSW'05)*, 2:179–184, 2005.

[21] Christoph Krau, Frederic Stumpf, and Claudia Eckert. Detecting node compromise in hybrid wireless sensor networks using attestation techniques. In *Lecture Notes in Computer Science, Security and Privacy in Ad-hoc and Sensor Networks*, 4572:203–217, 2007.

[22] Yi Yang, Xinran Wang, Sencun Zhu, and Guohong Cao. Distributed software-based attestation for node compromise detection in sensor networks. In *26th IEEE Symposium on Reliable Distributed Systems, Beijing, China, SRDS'07*, pages 219–230, Oct. 2007.

[23] Patrick Tague, David Slater, Jason Rogers, and Radha Poovendran. Vulnerability of network traffic under node capture attacks using circuit theoretic analysis. In *Proceedings of the 27th IEEE International Conference on Computer Communications, INFOCOM'08*, pages 161 – 165, April 2008.

[24] R. Di Pietro, A. Mei, L. Mancini, A. Panconesi, and J. Radhakrishnan. Redoubtable sensor networks. In *ACM Transactions on Information and System Security (TISSEC)*, 11(3):1–22, Mar. 2008.

[25] Ross Anderson, Haowen Chan, and Adrian Perrig. Key infection: Smart

trust for smart dust. In *Proceedings of the 12th IEEE International Conference on Network Protocols, ICNP'04*, pages 206–215, Oct. 2004.

[26] Bruno Dutertre, Steven Cheung, and Joshua Levy. Lightweight key management in wireless sensor networks by leveraging initial trust. In *SRI-SDL-04-02 SDL Technical Report*, 2004.

[27] David Naccache, David Pointcheval, and Christophe Tymen. Monotone signatures. In *Lecture Notes in Computer Science, Proceedings of the 5th International Conference on Financial Cryptography, FC'01*, 2339:305–318, 2001.

[28] Johan Håstad, Jakob Jonsson, Ari Juels, and Moti Yung. Funkspiel schemes: an alternative to conventional tamper resistance. In *Proceedings of the 7th ACM conference on Computer and communications security, CCS'00*, pages 125–133, 2000.

[29] Torben Pryds, Pedersen, and Birgit Pfitzmann. Fail-stop signatures. In *SIAM Journal on Computing*, pages 291–330, 1997.

[30] Mihir Bellare and Bennet S. Yee. Forward-security in private-key cryptography. In *Lecture Notes in Computer Science, CT-RSA*, 2612:1–18, Jan. 2003.

[31] Gene Itkis and Peng Xie. Generalized key-evolving signature schemes or how to foil an armed adversary. In *International Conference on Applied Cryptography and Network Security ACNS'03, Springer's LNCS 2846*, pages 151–168, 2003.

[32] Ross Anderson. Two remarks on public key cryptology. In *Fourth Annual Conference on Computer and Communications Security*, 1997.

[33] Gene Itkis. Cryptographic tamper evidence. In *Proceedings of the 10th ACM conference on Computer and communications security, CCS'03*, pages 355–364, 2003.

[34] R. R. Brooks, P. Govindaraju, N. Vijaykrishnan, M. Kandemir, and M. Pirretti. On the detection of clones in sensor networks using random key predistribution. In *IEEE Transactions on Systems, Man, and CyberneticsPart C: Applications and Reviews*, 37(6):1246–1258, Nov. 2007.

[35] Bryan Parno, , Adrian Perrig, and Virgil D. Gligor. Distributed detection of node replication attacks in sensor networks. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 49–63, 2005.

[36] Huirong Fu, Satoshi Kawamura, Ming Zhang, and Liren Zhang. Replication attack on random key pre-distribution schemes for wireless sensor networks. In *Computer Communications*, 31(4):842–857, Mar. 2008.

[37] Haowen Chan, Adrian Perrig, Bartosz Przydatek, and Dawn Song. Sia: Secure information aggregation in sensor networks. In *Journal of Computer Security, Special Issue on Security of Ad-hoc and Sensor Networks*, 15(1):69–102, Jan. 2007.

[38] Lingxuan Hu and David Evans. Secure aggregation for wireless networks. In *Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT-W'03 Workshops)*, pages 384–394, 2003.

[39] Matt Welsh. Exposing resource tradeoffs in region-based communication abstractions for sensor networks. In *Proceedings of the 2nd ACM Workshop on Hot Topics in Networks, SIGCOMM Computer Communication Review*, 34(1):119–124, Jan. 2004.

[40] David Wagner. Resilient aggregation in sensor networks. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 78–87, 2004.

[41] R. Merkle. Protocols for public key cryptosystems. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 122–134, 1980.

[42] Yi Yang, Xinran Wang, Sencun Zhu, and Guohong Cao. Sdap: a secure hop-by-hop data aggregation protocol for sensor networks. In *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing, MobiHoc'06*, pages 58–64, April 2006.

[43] Ronald W.Wolff. Stochastic Modelling and the Theory of Queues. 1989.

[44] Averill M. Law and W. David Kelton. Simulation Modeling and Analysis. 2000.

[45] H. Vincent Poor. An Introduction to Signal Detection and Estimation. 1994.

[46] Virgil D. Gligor. Security of emergent properties in ad-hoc networks. In *Proceedings of the 12th International Workshop on Security Protocols, Cambridge, UK, LNCS 3957, Springer-Verlag*, pages 256–266, April 2004.

[47] S. Farshad Bahari and Virgil D. Gligor. Handling new adversaries in ad-hoc networks. In *Proceedings of the 16th International Workshop on Security Protocols, Sidney Sussex College, Cambridge, UK, to appear in Springer-Verlag*, April 2008.