Dynamic Redundancy Management of Multisource Multipath Routing Integrated with Voting-based Intrusion Detection in Wireless Sensor Networks

Hamid H. Al-Hamadi

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Computer Science and Applications

Ing-Ray Chen, Chair
Dennis G. Kafura
Gregory W. Kulczycki
Chang-Tien Lu
Hazhir Rahmandad

April 02, 2014
Falls Church, Virginia

Keywords: Wireless sensor networks, multisource multipath routing, dynamic redundancy management, energy conservation, reliability, security, intrusion detection, intrusion tolerance, performance analysis.

Dynamic Redundancy Management of Multisource Multipath Routing Integrated with Voting-based Intrusion Detection in Wireless Sensor Networks

Hamid H. Al-Hamadi

ABSTRACT

Wireless sensor networks (WSNs) are frequently deployed unattended and can be easily captured or compromised. Once compromised, intrusion prevention methods such as encryption can no longer provide any protection, as a compromised node is considered a legitimate node and possesses the secret key for decryption. Compromised nodes are essentially inside attackers and can perform various attacks to break the functionality of the system. Thus, for safety-critical WSNs, intrusion detection techniques must be used to detect and remove inside attackers and fault tolerance techniques must be used to tolerate inside attackers to prevent security failure.

In this dissertation research, we develop a class of dynamic redundancy management algorithms for redundancy management of multisource multipath routing for fault and intrusion tolerance, and majority voting for intrusion detection, with the goal of maximizing the WSN lifetime while satisfying application quality-of-service and security requirements, for base station based WSNs, homogeneous clustered WSNs, and heterogeneous clustered WSNs. By means of a novel model-based analysis methodology based on probability theory, we model the tradeoff between energy consumption vs. reliability, timeliness and security gain, and identify the optimal multisource multipath redundancy level and intrusion detection settings for maximizing the lifetime of the WSN while satisfying application quality-of-service requirements. A main contribution of our research dissertation is that our dynamic redundancy management protocol design addresses the issues of "how many paths to use" and "what paths to use" in multisource multipath routing for intrusion tolerance. Another contribution is that we take an

integrated approach combining intrusion detection and tolerance in the protocol design to address the issue of "how much intrusion detection is enough" to prevent security failure and prolong the WSN lifetime time.

We demonstrate resiliency of our dynamic redundancy management protocol design for intrusion detection and tolerance against sophisticated attacker behaviors, including selective and random capture, as well as persistent, random, opportunistic and insidious attacks, by model-based performance analysis with results supported by extensive simulation based on ns3.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Advances in wireless sensor networks (WSNs) technology have made it an attractive choice for a wide variety of applications. WSNs can be deployed in military, civil, healthcare, and environmental applications [4, 74, 109, 118]. For security-critical applications such as homeland security and battlefield surveillance [46, 73, 88], security is a major concern especially since WSNs are usually deployed in unsupervised and hostile environments vulnerable to security compromise and susceptible to physical capture. Thus there is a need for efficient protocols to prevent, detect, and tolerate compromised nodes. Security design is especially challenging due to severe resource constraints in WSNs.

WSNs can be classified from three perspectives: architecture, data flow, and application. From the architecture perspective, WSNs can be classified as base station based (BS-based), or cluster based [4, 79]. BS-based WSNs rely on the existence of a BS or more to collect data from sensor nodes (SNs). The BS is powerful and is often assumed secure because of physical protection. A Cluster based WSNs consists of several clusters, where each cluster contains a cluster head (CH) responsible for collecting data from nearby sensor nodes Cluster based WSNs can be further differentiated based on if SNs are homogeneous or heterogeneous. In homogeneous clustered WSNs, all SNs are of the same capacity and serve as CHs on a rotational basis, so all SNs consume energy at about the same rate. In heterogeneous clustered WSNs, frequently two types of nodes exist: super SNs with more capacity and regular SNs with low capacity. The more powerful super SNs often assume the role of CHs. Figure 1-1 below shows the difference in topology between cluster Based and BS-Based WSN.

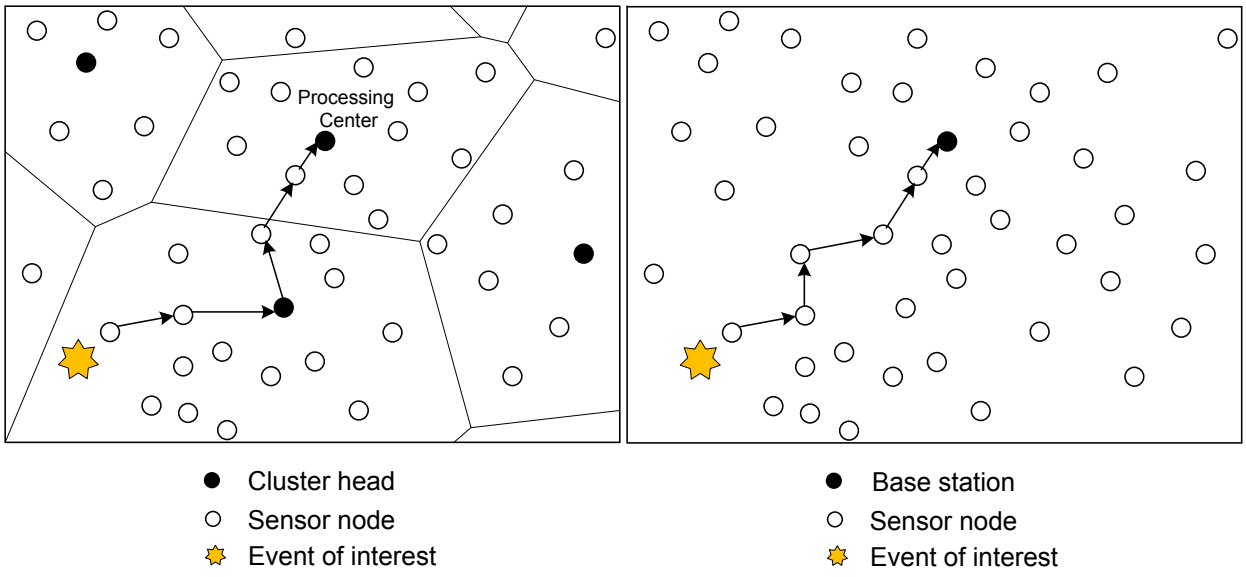


Figure 1-1: A Cluster-Based WSN (left) vs. A BS-Based WSN (right).

From the data flow perspective, WSNs can be classified as source-driven or query-based [38, 58]. In a source-driven WSN, SNs sense the environment at a fixed rate and periodically transmit sensing data to a sink node which can be a BS in a BS-based WSN or a CH in a clustered WSN. In query-based WSNs, a query is first received by a sink node, and then is sent to the feature areas to collect data requested. SNs in the feature areas collect data and forward data to the sink node in response to the query.

From the application perspective, WSNs can be classified based on the application domain, purpose, capability and structure as summarized in Table 1-1. One can see that all applications listed in Table 1-1 use either BS-based or clustered WSNs.

Our dissertation research addresses the security and dependability gain vs. energy consumption tradeoff issues for BS-based and clustered WSNs (from the architecture perspective) for maximizing the system lifetime, with the applicability and validity of our dissertation research focused on query-based WSN applications (from the data flow and application perspectives).

Table 1-1: Classifying WSNs based on Application Domain, Purpose, Capacity and Structure.

Application Domain	Purpose	Capability and Structure
Oil and gas [3, 127]	Monitor pipelines and equipment; detect gas leaks; monitor operation performance; monitor resources (e.g. reservoir status)	BS-based WSNs with homogeneous or heterogeneous sensors monitoring noise, vibration, humidity, electrical characteristics, temperature, radiation, toxic gases, etc.
Military [55, 56]	Monitor military infrastructure (both fixed/temporary); monitor battlefield	Clustered WSNs with homogeneous or heterogeneous sensors monitoring noise, vibration, temperature, radiation, toxic gases, speed, direction, size, location, etc.
Nuclear power plants [90]	Monitor equipment; monitor radiation	BS-based WSNs with homogeneous or heterogeneous sensors monitoring noise, vibration, humidity, temperature, electrical characteristics, and radiation
Forest [20, 92]	Ecological monitoring; fire detection	Clustered WSNs often with homogeneous sensors only monitoring ecological such as temperature and humidity
Smart City [95]	Natural disaster monitoring; environment monitoring such as emission data	A BS-based WSN with homogeneous or heterogeneous sensors monitoring motion, location, direction, size, temperature, humidity, radiation, etc.

Many WSNs are deployed in an unattended environment in which energy replenishment is difficult if not impossible. Due to limited resources, a WSN must not only satisfy the application specific Quality of Service (QoS) requirements such as reliability, and timeliness, but also minimize energy consumption to prolong the system useful lifetime. The tradeoff between energy consumption vs. QoS gain with the goal to maximize the WSN system lifetime has been well explored in the literature. However, no prior work exists to consider the tradeoff in the presence of malicious attackers.

The tradeoff issue between energy consumption vs. QoS gain becomes much more complicated when inside attackers are present as a path may be broken when a malicious node is on the path. Thus, very likely the system must employ an intrusion detection system (IDS) with the goal to detect and remove malicious nodes. While the literature is abundant in intrusion detection techniques for WSNs [21, 46, 49, 84, 129], the issue of how often intrusion detection should be invoked for energy reasons in order to remove potentially malicious nodes to prevent security failure (say to prevent a Byzantine failure [85]) is largely unexplored. The issue is especially critical for energy-constrained WSNs designed to stay alive for a long mission time.

Multipath routing is considered an effective mechanism for fault and intrusion tolerance to improve data delivery in WSNs. The basic idea is that the probability of at least one path reaching the sink node increases as we have more paths doing data delivery. While most prior research focused on using multipath routing to improve reliability [38, 58, 124], some attention has been paid to using multipath routing to tolerate insider attacks [54, 78, 93]. These studies, however, largely ignored the tradeoff between QoS gain vs. energy consumption which can adversely shorten the system lifetime. In the dissertation research, we leverage multipath routing as a form of redundancy control for achieving the desired level of fault and intrusion tolerance in WSNs. More importantly, we aim to identify the best balance between energy consumption and fault and intrusion tolerance strength so that the WSN lifetime is maximized.

It is well known that SNs close to the BS or CH are more critical in gathering and routing sensing data. The issue of protecting critical nodes against selective capture, i.e., critical SNs are targets of select capture attacks, is largely unexplored in the literature. We aim to design and validate multisource multipath routing protocols for intrusion tolerance as a countermeasure against selective capture.

1.1 Problem Definition and Goals

This dissertation research is motivated by the following research issues:

1. **Intrusion and Fault tolerance**: Many WSNs are deployed unattended in hostile and harsh environmental conditions. This makes them susceptible to hardware faults,

physical capture, and malicious attacks by compromised SNs. Thus, designing a system that tolerates faults and intrusions is important in WSNs. An algorithm for fault and intrusion tolerance must take into consideration the limited processing, storage, energy, and transmission capabilities of SNs. Moreover, it must also carry out its intended function in the presence of compromised nodes, taking into consideration physical capture and smart insider attacks. The algorithm design must also be resilient to single points of failures, which is a major threat to a safety critical WSN with high security demands.

2. **Intrusion Detection**: In harsh environments, it is necessary to apply IDS design to complement passive intrusion tolerance to result in safety critical WSNs. The intrusion detection design must be light-weight so as not to unnecessarily consume energy yet effective in detecting and evicting compromised nodes with low false alarm probability. The issue of how often intrusion detection should be invoked to trade detection strength with energy consumption is largely unexplored in the literature. The IDS must also be highly resilient to attacks including node capture and insider attacks.
3. **Tradeoff between QoS vs. Security:** A WSN often has stringent QoS requirements that must be maintained at all times such as timeliness and reliability of packet routing. However, there is often a tradeoff between QoS and security strength which ties to energy consumption. The issue of energy-aware redundancy management in WSNs for satisfying QoS requirements without excessive energy consumption requires a holistic approach of exploiting the tradeoff between intrusion detection/tolerance strength vs. energy consumption. This dissertation research explores multisource multipath routing as a mechanism to satisfy QoS requirements while applying redundancy management to minimize energy consumption and maximize the WSN lifetime.
4. **Wireless Sensor Network Lifetime Maximization:** Due to limited energy of WSNs, it is critical to implement energy-aware protocols that maximizes the WSN lifetime while satisfying the QoS requirements of the WSN. Energy consumption due to WSN functionality and security mechanisms could potentially decrease the system lifetime. Limited security might conserve energy but could result in a high density of compromised nodes breaking the basic functionality of the WSN and causing security failure. There is a tradeoff between energy conservation vs. QoS and security gain, and the system must find the best set of protocol settings to achieve QoS and security requirements and maximize lifetime.

Addressing all issues in tandem is little explored in the literature. Part of the difficulty comes from a lack of tradeoff analysis between energy consumption vs. QoS and se-

curity satisfaction. This dissertation research takes a holistic approach and aims to address all issues in tandem. Specifically, we aim to design and validate a class of multisource, multipath routing algorithms for fault and intrusion tolerance. We aim to identify the best way to combine it with IDS protocols for intrusion detection and eviction to satisfy QoS and security requirements and maintain basic functionality of the system, while maximizing the WSN lifetime. We aim to devise an analysis methodology allowing us to derive optimal configurations that maximize the lifetime of the WSN under attack while achieving QoS and security requirements. The analysis methodology must be generally applicable to different types of WSNs, including BS-based WSNs, and homogeneous and heterogeneous clustered WSNs. Lastly, we aim to develop adaptive redundancy engineering designs allowing the optimal configurations to be determined and applied dynamically at runtime in response to changing environment conditions including energy expenditure, capture and insider attacker behavior, and dynamic node behavior.

1.2 Research Contribution

We envision the following contributions from the dissertation research:

1. Untreated in the literature, we explore the tradeoff between energy consumption vs. QoS and security gain with the goal to maximize the lifetime of WSNs while satisfying application QoS and security requirements in the context of multisource multipath routing for query-based WSNs. The approach can be easily extended to source-driven WSNs. More specifically, we analyze the optimal amount of redundancy in terms of the number of source SNs sensing the same physical phenomena and the number of paths through which data are routed to the sink node in the presence of malicious nodes so that the query success probability is maximized while maximizing the WSN lifetime. We also consider an integrated approach for combining multisource multipath routing for intrusion tolerance with voting-based IDS for intrusion detection to remove malicious nodes from the WSN. Our contribution is to identify the optimal multisource multipath redundancy levels and intrusion detection settings for maximizing the WSN lifetime while satisfying application QoS requirements.
2. For the issue of intrusion tolerance through multisource multipath routing, there are two major problems to solve: (1) how many paths to use and (2) what paths to use. To the best of our knowledge, we are the first to address the “how many paths to use” problem. One main contribution of our research dissertation is that we decide “how many paths to use” in order to tolerate residual compromised nodes that survive intrusion detection, so as to maximize the WSN lifetime. For the “what paths to

use" problem, our approach is distinct from existing work in that we do not consider specific routing protocols (e.g., MDMP for WSNS [86] or AODV for MANETs [112]), nor the use of feedback information to solve the problem. Instead, for energy conservation, we employ distributed light-weight host-based and system-level IDS by which intrusion detection is performed locally. Nodes that are identified compromised are removed from the WSN. Only compromised nodes that survive detection have the chance to participate in routing.

3. We analyze the effect of *selective capture* on critical nodes in a WSN and develop and validate strategies for countering selective capture strategies by a smart attacker that targets critical nodes close to the BS or CHs to maximize the damage to a BS-based WSN. We analyze a protocol with 3 countermeasures in the protocol design: (1) dynamic radio range adjustment; (2) multisource multipath routing for intrusion tolerance; and (3) voting-based intrusion detection. We develop a probability model to reveal the tradeoff between energy consumption vs. reliability and security gain with the goal to maximize the lifetime of the WSN.
4. We analyze the effect of various attack strategies performed by inside attackers on the WSN including *random*, *opportunistic* and *insidious* attacks. We also analyze the effect on system lifetime and identify the best settings to counter these attacks.
5. We develop and validate the design notion of *dynamic redundancy management* to adaptively update the optimal settings of IDS (the trust/reputation update interval, number of voters, and IDS interval) and the optimal settings of multisource, multipath routing (how may paths and sources, and what paths to use) in response to dynamic condition changes such as query pattern, adversary behavior/status, node behavior/status, and energy status changes, to maximize lifetime while achieving security, and QoS requirements.
6. We develop a novel model-based analysis methodology based on probability model to analyze the best redundancy level to be applied for WSN lifetime maximization while best satisfying the application QoS requirements with simulation, using ns3, to support our analytical results. The analysis methodology is generic and applicable to a variety of WSNs including BS-based, and homogeneous and heterogeneous clustered WSNs. We also demonstrate the validity of our design by a comparative performance analysis with existing protocols through extensive simulation.

1.3 Thesis Organization

The rest of the dissertation is organized as follows. Chapter 2 provides a comprehensive survey of existing work related to the dissertation research. Chapter 3 presents the system model specifying the attack model, system assumptions, and applicability.

In Chapter 4 and Chapter 5, we develop analytical models based on probability theory for dynamic redundancy management of multisource multipath routing and intrusion detection for homogeneous clustered WSNs and heterogeneous clustered WSNs, respectively. In Chapter 6 we analyze selective capture, smart attacks, and countermeasures for BS-based WSNs with simulation to support our analytical results. Finally, Chapter 7 summarizes the dissertation research and outlines future research areas.

Chapter 2

Related Work

2.1 Intrusion Detection for WSNs

Wireless sensors are frequently deployed unattended and can be easily captured or compromised. Once compromised, intrusion prevention methods such as encryption can no longer provide any protection, as a compromised node is considered a legitimate node and possesses the secret key for decryption. Compromised nodes are essentially inside attackers and can perform various attacks to break the functionality of the system. Thus, intrusion detection techniques must be used to detect and remove inside attackers to prevent security failure.

Over the past few years, numerous protocols have been proposed to detect intrusion in WSNs. [21, 129] each provide an excellent survey of the subject. Existing IDS techniques proposed for WSNs can be categorized as centralized where the detection is based on a single BS or CH making the detection decision, and cooperative (decentralized) where many nodes collect information and reach a collective decision about a particular node. One advantage of cooperative IDS over centralized IDS is that there is no single point of failure. In cooperative IDS, many schemes are based on overhearing neighbor communication [49, 82, 96]. In [96] a watchdog scheme is proposed where the sender monitors the transmission of its next hop neighbor, and announces its maliciousness if it drops or fails to relay packets towards the destination. DICAS [82] is a protocol that detects and isolates malicious nodes in WSNs. Guard nodes are responsible for monitoring the activity of neighboring nodes. Once a target node's negative behavior count exceeds a threshold, guard nodes inform all the neighboring nodes, which in turn mark the node as revoked. In [49], a decentralized rule-based intrusion detection system is proposed by which monitor nodes are responsible for monitoring neighboring nodes. Their algorithm consists of a data acquisition phase which consists of the monitor nodes collecting messages from monitored neighbors, followed by a rule application phase where the stored messages are checked against predefined rules and failures are

raised if the messages fail the tests, and finally by an intrusion detection phase where the number of failures are examined and an alarm is raised if the number of failures exceeds a threshold value.

In [84], a collaborative approach is proposed for intrusion detection where the decision is based on majority voting of monitoring nodes. The IDS system is based on specification-based detection and aims to detect blackhole and selective forwarding attacks. If the majority of the monitoring nodes raise an alert for a specific neighboring node, then that node is deemed compromised and the system either takes direct action and revokes its communication with neighboring nodes or informs the base station which indirectly cuts it off from the network by degrading its path reliability. The authors in [120] define a framework for distributed cooperative failure detection, and specify methods with which the cooperating nodes can communicate effectively and efficiently. They rely on the creation of tree-based propagation-collection protocols. In [91], a voting-based distributed intrusion detection algorithm is proposed to detect insider attackers. The algorithm is purely localized with each sensor monitoring its immediate neighbors, and flagging abnormal behavior. Multiple features are examined and the final decision is based on majority voting. The result of the evaluation can also be combined with routing to ensure reliable communication. The majority voting prolongs the system lifetime since the energy consumption for intrusion detection is distributed among nodes.

In the dissertation research, we consider two levels of intrusion detection: host IDS and system IDS. The host-level IDS is based on monitoring, as done in [49, 82, 96]. We model the flaws of host IDS by a false positive probability (H_{pfp}) and a false negative probability (H_{pfn}). The system-level IDS is based on majority voting, as done in [84, 91]. The reason for having system IDS in addition to host IDS is to prevent node collusion as host IDS alone would break down when the monitor node itself is malicious. Specifically, our voting-based IDS approach extends from [46] by employing a number of verifiers randomly selected who cast their votes based on the host IDS results. The dissertation research investigates the best intrusion detection strength to apply in terms of the detection frequency and the number of verifiers, with considerations given to the tradeoff between energy consumption vs. security gain due to employment of voting-based IDS with the goal to prolong the WSN system lifetime.

Recently we have seen trust-based or reputation-based approaches being proposed in the literature for intrusion detection [13, 14, 23, 57, 61, 97, 98]. In [97], a mechanism to enforce node cooperation through collaborative monitoring is proposed. The reputation metrics are built based on observations and experiences of neighboring nodes. It is shown that denial of service attacks based on malicious broadcasting are prevented using this mechanism. The same authors propose a cooperative security scheme for de-

tecting selfish nodes is evaluated over the DSR routing protocol in [98]. The authors show that collaboration between nodes is an efficient method for detecting misbehaving selfish nodes. In [23], a reputation-based approach to detecting and isolating misbehaving nodes that refuse to cooperate with the network is proposed. Each node maintains information of neighboring nodes and their ratings based on own experiences, observations, and reported experiences. Each node contains a path manager component that is responsible for choosing secure paths based on available reputation information. The work uses the DSR routing protocol to illustrate their algorithm. Wang et al. [44] proposed to use the evidence chain for detecting misbehaviors of a node, and the trust fluctuation for reflecting the variability of a node's trust value over a time window. Ebinger et al. [45] proposed to split the reputation information into trust and confidence for reputation exchanges and then combined them into trustworthiness for intrusion detection. Our dissertation research lays the ground work to further extend prior work [13, 14] for trust-based intrusion detection at the host IDS level and for choosing trustworthy routing paths for multisource multipath routing.

In [113], the authors considered a centralized approach for fault and intrusion detection, relying on a powerful BS to perform computation and trace the identity of faulty nodes based on the information provided by the sensors in the network. Similarly, [76] proposed a method to overcome the response implosion problem caused by using a central diagnosing node. The central node becomes a bottleneck, and the authors proposed diffused computation mechanisms to address the problem. In the dissertation research, our host IDS and system IDS designs are based on distributed intrusion detection to avoid a single point of failure and performance bottleneck.

2.2 Capture and Attack Strategies in WSNs

Capture attacks in WSNs can be classified as either random or selective [12-14]. Selective capture attacks maximize the attack strength by targeting nodes whose capture will result in a high possibility of compromising the WSN. An intelligent attacker can strategically attack a specific area or a group of sensors to compromise the most number of keys that are not yet compromised [71, 72]. A clever adversary can strategically attack certain sensors so as to reveal the largest number of unknown pairwise keys [105]. In particular, [14] developed a framework to analyze the effect of selective attacks on performance of key pre-distribution protocols. However, in [12-14] selective capture was about key compromises and the focus was on key pre-distribution protocol design for achieving resiliency against key compromise attacks. Our dissertation research considers the presence of smart attackers capable of performing strategic capture of "critical nodes" near the BS to block data delivery. We note that in the literature, various approaches [52, 53] have been proposed to masquerade and hide CHs and critical SNs.

However, energy consumption is generally a concern for SNs in these approaches. Our approach is dynamic redundancy management of countermeasures. We demonstrate the effectiveness of our dynamic redundancy management protocol against selective capture of critical nodes to create black holes near the BS to maximize its attack strength.

In [1], the authors utilized ARM TrustZone and ARM secure boot to provide a secure boot process in which only secure system code is booted. The boot-up firmware cannot be modified or replaced, since it resides in a memory that is programmed only once. Keys and passwords are protected in memory locations which are secured from unauthorized access (on the processor chip itself). Hash values of images to be loaded are stored in secure memory for comparison at time of loading. While the authors approach seems promising, they do not assess the system against attacks that target vulnerabilities in the running code. The authors in [60] showed how permanent code injection attacks can be easily performed on Harvard-based architecture devices (e.g. Mica motes). The attacker can exploit vulnerabilities (e.g. buffer overflow) in the running code to run a sequence of instructions which in turn injects malicious code into the sensor. In [17] the authors classified attacks based on the time and resources needed by the attacker , ranging from minutes using plug-in connections to days using more invasive techniques to erase security bits using UV light and the usage of high cost equipment. The authors mentioned that long periods of inactivity are strong indications of tampering by an attacker, and suspicious nodes can be revoked by the network. While a secure boot process makes it harder for an attacker to compromise a node, a clever attacker can target a vulnerability which could lead to bypassing security measures. In this dissertation, we assume that the attackers have access to advanced equipment which enables invasive attacks to be performed, and once a node is captured, it is compromised. Our analytical model, however, can be easily extended to the case in which tamper proof hardware can successful prevent the attacker from turning a captured node into a compromised inside attacker.

Once a node is compromised (through capture) it becomes an inside attacker. A smart inside attacker can employ various attack strategies to maximize its attack strength [47]. In [10], the authors consider attackers that do not launch direct attacks in order to avoid detection. Instead, the captured nodes behave normally but aim to harm the network through injecting false data to the data collector. In [121], the authors consider the existence of malicious nodes that can decrease their attack rate in order to disguise themselves and avoid being detected by intrusion detection. A malicious node can drop incoming packets with the risk of being identified. The impact of attacker behaviors on a behavior-rule based intrusion detection design for smart grids is investigated in [103]. The authors consider two attack behaviors; reckless (constantly attacking to impair the system) and random (attacking while avoiding intrusion detection). Similar

to [103], we consider inside attackers that can perform persistent and random attacks. Furthermore, we consider smart attackers that can perform opportunistic attacks while perform random attacks to evade intrusion detection until a critical mass of compromised node population is reached after which "all in" attacks are performed to cripple the system. In a BS-based WSN, because of selective capture, the inside attacker strength can be a function of both time and distance from the BS.

2.3 Redundancy Management of Intrusion Detection in WSNs

Intrusion detection protocols for WSNs must take into consideration the limited resource available to SNs including limited energy supply, processing, and communication capabilities. In the literature, several studies exist attempting to achieve energy-efficient intrusion detection.

Ling, et al. [123] proposed Intrusion Detection Optimal Scheduling (IDOS). Their proposed IDS system is composed of nodes that sense activity (called collecting nodes), and transmit suspicious events to sink nodes. The collecting nodes follow a sleep schedule to save energy. Their algorithm aims to find the best schedule to minimize energy and detection delay, but no consideration is given to minimizing the system lifetime. Our dissertation research aims to find the best IDS and redundancy management parameter settings that minimize energy usage while satisfying QoS and security requirements to maximize the system lifetime.

In [99], the authors proposed a self-learning system used by SNs to detect malicious packets. A SN follows a sampling rate to sample packets in an attempt to detect malicious packets and drop them. The sampling rate is controlled such that it matches the level of compromise exhibited in the system, thus resulting in an energy-efficient system. Our work also aims to determine the best rate at which to invoke intrusion detection; however, we aim to find the best balance of reliability and security gain vs. energy consumption to maximize the system lifetime.

In [25], the authors proposed a hierarchal model for WSNs with cluster heads responsible for reporting intrusion related data to the BS, thus avoiding expensive transmission from SNs to the BS. This approach however drains energy of CHs faster than SNs, thus may adversely shorten the lifetime span of CHs which are critical to the WSN. eHIP was proposed in [116] to combine intrusion prevention based on authentication with collaborative-based intrusion detection by which a CH monitors member SNs. A CH isolates SNs that behave suspiciously. CHs themselves are monitored by SN groups that accumulate suspicious activity up to a threshold, after which the CH is evicted. SN groups are formed on a rotating basis for energy conservation purposes. In [70], a protocol for optimal selection and activation of intrusion detection agents was proposed.

Every node is assigned a trust value by its CH. Only agents with sufficient trust are allowed to activate the monitor agent and participate in the cooperative IDS by sending alerts to its CH. CHs enforce this rule and lower trust levels of misbehaving agents. Energy is conserved by limiting the number of active monitoring agents, and avoiding duplicate alerts to CHs, which in turn enhances the system lifetime. Bao, et al. [13, 14] also considered the use of trust in a hierarchically structured WSN such that a CH collects trust evaluations of SNs toward their peer SNs in the same cluster for intrusion detection in an effort to reduce energy consumption.

Relative to the work cited above, our dissertation research takes a more holistic approach by considering energy consumption vs. security gain not only for intrusion detection by means of adjusting IDS strength in response to attacker strength, but also for intrusion tolerance by means of redundancy management of multisource multipath routing with the goal to maximize the WSN lifetime.

2.4 Multipath Routing for Intrusion and Fault Tolerance

A common strategy to tolerate the compromise or failure of nodes in WSNs is the use of multiple paths to transmit packet to the intended destination. [114] provides an excellent survey in this topic. The use of single-path routing is highly susceptible to the single failure or compromise of a node on the path. Furthermore, once the path fails, the WSN needs to promptly expend energy to find a new path to maintain its operation. The use of multiple paths can further improve the WSN lifetime by avoiding the use of the same path during routing.

In [78] multiple paths are used to route traffic to the destination using geographic routing, aiming to increase packet delivery ratio in the presence of packet dropping attacks (through blackhole and selective forwarding). A trust based approach is taken by which a sender uses overhearing to monitor if the next nodes forward its packets. Similarly in [80] the authors propose a probabilistic routing algorithm called ARRIVE that is robust to link failures and patterned node failures. The algorithm is based on localized information. For communication, multiple packets are transmitted over diverse paths to increase the end-to-end reliability. The algorithm uses link reliability and node reputation to determine the next hop in routing. However, no energy consideration is given. In [54] the authors considered a disjoint multipath routing protocol that aims to tolerate intrusion by using multiple redundant paths to send a message to a destination. It aims to operate correctly in the presence of undetected intruders. However, it relies on the existence of a powerful base station to plan multipath routing, which is normally not available in WSNs, or otherwise would be a single point of failure.

Lee and Choi [87] presented a secure alternate path routing protocol called SeRINS. The alternate paths formed make routing resilient to selective forwarding attacks. Sensors are connected with parents in a tree-based formation, with every node having multiple parents which it utilizes for forwarding in a round robin manner. A malicious node that injects false routing information is identified by its neighbors which report the identity of the malicious node to the BS. The BS in turn revokes the malicious node's keys and evicts it from the WSN. The proposed protocol, however, does not address how to detect malicious nodes for other types of attacks, nor does it give considerations to energy consumption and lifetime maximization.

In [110], packets are sent over randomized dispersive multipath routes with the aim to avoid black holes resulting from compromised nodes performing packet dropping and/or denial of service attacks. A packet is split into n shares based on coding theory so that if k out of n shares are received, then the packet can be reconstructed. The randomized multipath routes generated are dispersive to avoid black holes and to enhance the probability of at least k out of n shares can reach the receiver. The approach, however, does not consider intrusion detection to detect compromised nodes, nor energy consideration. H-SPREAD [93] is another multipath protocol that relies on splitting a message into N shares using a secret sharing scheme, where each share is forwarded using a distinct path from source node to the base station. The base station can recover the message if T out of N shares is successfully received.

Relative to the work cited above, our dissertation research considers multipath multisource routing as well to circumvent black hole attacks for intrusion tolerance. In addition, we consider intrusion detection to detect and evict compromised nodes. More importantly, we consider the best way to perform multipath routing and intrusion detection to best tradeoff energy consumption vs. security and reliability gain to maximize the WSN system lifetime.

Geographic routing is the prevalent routing algorithm for WSNs due to its simplicity in routing, and its independence of routing tables [78], [122] which makes it immune to routing table attacks. The authors in [122] proposed a family of secure protocols with varying security strengths at the expense of more shared states. The protocols are built on a base protocol, IGF, which depends on geographic routing and does not use routing tables. This prevents attacks such as state corruption, wormholes, and HELLO floods. Attacks such as black hole, selective forwarding, and Sybil are defended by the use of reputation and cryptography. The authors showed that there is a tradeoff between security and shared states. We also adopt geographic routing for its desirable security properties. Further, we use multisource multipath routing for intrusion tolerance, and distributed IDS for intrusion detection against the above mentioned attacks plus collusion attacks with the goal to maximize the system lifetime.

Multipath routing has also been used for fault tolerance [9]. In [62], a multipath routing protocol based on Directed Diffusion [75] was proposed. The protocol maintains multiple paths between a source and a sink with one being primary and the remaining ones being backup alternative paths. The paths need not be disjoint; the authors considered the use of both disjoint and braided paths. The backup paths are kept alive by sending keep-alive messages. In the case of the primary path failing, a quick recovery is made to make one of the existing backup paths take over the primary path of communication. REAR [66] is another protocol that relies on maintaining a backup path and switching to it in the event of the primary path failure. This protocol differs in that nodes located on both paths reserve a fixed amount of energy for the communication between the source node and the sink node. The protocol takes into account the amount of energy reserved by nodes when allocating paths, in order to balance the energy consumed by the paths in the WSN. Our work differs from the above work in that we do not rely on a recovery protocol involving switching of allocated paths. Instead, we adopt multisource multipath routing with dynamic redundancy management to best balance energy consumption vs. security gain. Additionally, we use multipath routing to deal for both fault and intrusion tolerance.

2.5 Redundancy Management of Multipath Routing

In this section, we survey redundancy management issues for multipath routing. Particularly, we survey existing work addressing the issues of "how many paths" and "what paths" should be used for multipath routing.

SEEM [104] is a multipath routing protocol that relies on a powerful BS to perform route discovery, maintenance, and route selection. The BS takes into account the remaining energy of nodes when selecting routing paths between source and sink. SEEM shows improvement over directed diffusion in certain performance metrics such as network throughput, communication overhead, and network lifetime. Furthermore, it has some resistance against false routing path attacks since the routing paths are centrally selected by the BS. However, it does not consider the existence of malicious nodes and there is no consideration given to detect attacks. In [24], a fault-tolerant algorithm for heterogeneous WSNs with super SNs and regular SNs was proposed. Every regular SN is connected to a set of super SNs through k-paths, while minimizing the transmission range (and thus the transmission power). This k-vertex super SN connectivity can tolerate the failure of up to k-1 super SNs, thus providing a fault-tolerant WSN. In [119], an energy efficient adaptive routing protocol was proposed. The protocol creates multiple paths between a source and the sink node and chooses paths based on node energy levels and signal strengths. Furthermore, the protocol aims to provide an environment with high reliability and low energy consumption by efficiently balancing the total en-

ergy consumption of the nodes, which in turn increases the WSN lifetime. In [38][6] the issue of how many paths for fault tolerance was addressed in the context of multipath routing from a source node to a sink node. The authors identified the best path redundancy to apply to best trade energy consumption for reliability gain to maximize the system lifetime. However, no presence of malicious nodes was considered. This dissertation research is the first work that addresses the issue of "how many paths" to use for both intrusion tolerance and fault tolerance in the context of multisource multipath routing in WSNs.

Relative to the work cited above, our dissertation work uses multiple paths for both fault tolerance and intrusion tolerance. Further, we consider dynamic redundancy management to adaptively adjust the redundancy level for multisource multipath routing in response to changing environment conditions to trade reliability and security gain vs. energy consumption so as to maximize the WSN lifetime. Taking into consideration of the attacker behavior, our redundancy management design considers not only redundancy management of multipath routing, but also redundancy management of intrusion detection strength to counter the attacker strength, so as to best balance security gain vs. energy consumption to maximize the system lifetime. Unlike existing work, we address both "how many paths to use" and "what paths to use" issues.

2.6 Lifetime Maximization of Wireless Sensor Networks

Over the past few years, many protocols exploring the tradeoff between energy consumption and QoS gain particularly in reliability gain in WSNs have been proposed. In [115], the optimal communication range and communication mode were derived to maximize the WSN lifetime. In [111], the authors devised intra-cluster scheduling and inter-cluster multi-hop routing schemes to maximize the network lifetime. They considered a hierarchal WSN with CH nodes having larger energy and processing capabilities than normal SNs. The solution is formulated as an optimization problem to balance energy consumption across all nodes with their roles. In [128], the authors investigated the tradeoff between reliability versus lifetime in heterogeneous WSNs. SNs are grouped into sleep sets, and follow a sleep schedule that enables them to conserve energy. They derived the lifetime of the WSN as a function of the number of sleep sets, and proved that an optimal number of sleep sets can be found given a failure rate as input. In [83] the tradeoff between using data redundancy and maintaining data accuracy was studied. The authors identified the best protocol parameter settings for data aggregation and estimation quality such that the WSN lifetime is maximized while satisfying the estimation quality constraints. In [117], a strategy to minimize power consumption of queries and maximize the WSN system lifetime was proposed. The system considers queries with QOS requirements of power and accuracy, and finds the optimal sampling rate at

which the cost is minimized with the QOS requirements satisfied. The work however assumes fixed power transmission for nodes and does not consider radio range adjustment to maintain connectivity. In [94], the authors considered a two-tier WSN with the objective of maximizing network lifetime while fulfilling power management and coverage objectives. They determined the optimal density ratio of the SNs in the two tiers to maximize the system lifetime. Relative to the work cited above, we consider the presence of malicious nodes and explore the tradeoff between energy consumption vs. QoS and security gain to maximize the system lifetime.

MMSPEED [58] is a multipath multispeed routing protocol that provides QOS guarantees in both timeliness and reliability domains. End-to-end requirements are guaranteed in a localized way without global network information or a priori path setup. It also adopts geographic forwarding for packet delivery. However there is no consideration given to the presence of compromised nodes. Our solution of satisfying timeliness and reliability requirements of a query is totally distributed, which in some sense follows the design principle of MMSPEED. Contrast to MMSPEED, our work considers not only timeliness and reliability but also security issues, with the multipath multisource routing problem being formulated as a WSN lifetime optimization problem.

HEED [126] employs an energy-efficient distributed clustering approach for homogeneous clustered WSNs. The protocol aims to extend the lifetime of all the nodes in the network by distributing the energy consumption across the nodes. The role of cluster head is periodically changed based on residual energy and node proximity between sensor nodes such that energy consumption is distributed evenly among all sensors. Our dissertation research adopts HEED as the clustering algorithm for homogeneous clustered WSNs and extends it to addressing the issue of dynamic redundancy management for intrusion detection as well as fault and intrusion tolerance for system lifetime maximization.

Chapter 3

System Model

Our approach to maximizing lifetime using the proposed secure multipath routing protocol can be applied to a wide array of WSNs, including homogeneous clustered WSNs, heterogeneous clustered WSNs, and BS-based WSNs. Specific system models for these WSNs will be given in detail in Chapter 4, Chapter 5, and Chapter 6, respectively, where protocols and assumptions in each specific network system are discussed. In this chapter we discuss system assumptions generically applicable to all system models, including the system failure definition, the attacker model, and the applicability.

3.1 System Failure Definition

Our target system is a query-based WSN. We define the total number of queries the system can answer correctly until it fails as the *lifetime* or the *mean time to failure* (the MTTF) of the system, which can be translated into the actual system lifetime span given the query arrival rate. A failure occurs when the system ceases to provide responses to a query.

There are several causes for failing to provide response delivery. One cause is due to the lack of node availability (or connectivity). Nodes could cease to operate within the network due to either being evicted by the IDS (justly or unjustly), or due to energy exhaustion. This lack of connectivity could cut off the communication between sensor nodes that send query/response data to the final destination. While nodes can dynamically increase their transmission range to maintain connectivity, there is a maximum range after which nodes can no longer communicate with each other.

Another cause is due to the presence of inside attackers who could intentionally bring down the network by causing lack of connectivity between sensors and the BS/CH. It can perform slandering attacks by recommending a good node as a bad node, and a bad node as a good node when participating in intrusion detection activities. As a

result, slandering attacks can cause good nodes being misdiagnosed and evicted from the system, and bad nodes being missed and remained in the system. This effectively creates a 'black hole' area with a high concentration of bad nodes, especially for the critical SN area in a BS-based WSN with selective capture. Inside attackers can also perform packet dropping attacks, directly resulting in the loss of the data being forwarded toward the destination.

Hardware failure and transmission failure due to noise and interference can also contribute to loss of node availability or connectivity. Multipath routing is considered as an effective way to deal with transmission failure.

3.2 Attacker Model

3.2.1 Capture Attacks

All sensors are subject to capture attacks, i.e., they are vulnerable to physical capture by the adversary (humans or robots) after which their code is compromised and they become inside attackers. In the literature, capture attacks in WSNs can be classified as either random or selective [71, 72, 105]. We consider both types of capture in this dissertation research. In random capture, there is no distinction between CHs and SNs. In this case, the adversary performs arbitrary capture compromising nodes at random and turning them into inside attackers. In selective capture, the adversary tends to strategically capture CHs and "critical" SNs (those SNs close to the CH or the BS) whenever it can. Selective capture attacks maximize the attack strength by targeting nodes whose capture will result in a high possibility of compromising the WSN. An intelligent attacker can also strategically attack a specific area or a group of sensors to compromise the most number of keys that are not already compromised [71, 72]. Lastly, a clever adversary can strategically attack certain sensors so as to reveal the largest number of unknown pairwise keys [105].

In clustered WSNs, critical nodes are those SNs close to the CH because they are in the critical path to route packets to the CH. Therefore, capturing critical nodes will create black holes around the CH and block all traffic to the CH. We consider the presence of a smart attacker capable of performing strategic capture of critical nodes to maximize the attack strength. Our countermeasure designs for defending against selective capture include (1) dynamic radio range adjustment; (2) multisource multipath routing for intrusion tolerance; and (3) voting-based intrusion detection. This subject is treated in Chapter 6.

3.2.2 Insider Attacks

After a node is compromised it becomes an inside attacker. Inside attackers perform active attacks including packet dropping [81], packet modification, and bad-mouthing attacks to disrupt the operation of the network. A compromised node performs bad-mouthing attacks by recommending a good node as a bad node, and a bad node as a good node when participating in the voting-based distributed IDS as a voter. As a result, bad-mouthing attacks can cause good nodes being misdiagnosed and evicted from the system, and bad nodes being missed and stayed in the system.

Attacks in WSNs have been widely studied by the research community [49, 50, 81, 122, 129]. Below we summarize insider attacks which can be dealt with by intrusion detection and tolerance countermeasure designs proposed in the dissertation research.

1. *Blackhole/Selective forwarding*

   A malicious node may refuse to forward a packet upon reception and drop it instead. Selective forwarding is when the malicious node drops some but not all the packets that pass it through routing. When an adversary drops all packets, it forms a black hole. A selective forwarding attack is harder to detect than a black hole attack as it could be mistaken for an occasional message loss. On the other hand, a message loss could be falsely identified as an intentional packet drop by an attacker. We rely on monitoring and overhearing techniques in host IDS design to identify black hole and selective forwarding attacks.

2. *Sybil attacks*

   In a Sybil attack an adversary pretends to be multiple entities. To counteract a Sybil attack and prevent impersonation, we assume the use of cryptographic measures between sensor nodes, where nodes communicate with each other using pairwise keys. Furthermore, inconsistent information from the adversary performing Sybil attacks can be identified by neighboring nodes participating in voting-based intrusion detection. Trust-based IDS can furthermore increase the detection accuracy of multiple fake identities through the use of both local observations and recommendations.

3. *Flood attacks*

   In a flood attack an adversary flood the network with multiple packets in order to disrupt communication between nodes and deplete network resources of the WSN. This attack can be detected by monitoring and overhearing techniques in host IDS design.

4. *Bad-mouthing attacks*

A good node could be recommended unjustly as a bad node. Similarly a bad node could be recommended as a good node. We rely on majority voting at the system IDS level to cope with bad-mouthing attacks by inside attacker. By comparing votes cast by a malicious node against those cast by good nodes, we can also detect a malicious node at the host IDS level. A trust system can further help tolerate bad-mouthing attacks by imposing a trust threshold to filter out non-trustworthy voters.

5. *Routing state corruption*

   Altering or replaying routing messages can corrupt the routing state of nodes and result in message loss and routing loops. We use geographic routing in our protocol design, where sensors send packets towards the direction of the final destination. There is no routing state to be shared by SNs, thus this attack is prevented in our system.

6. *Wormhole attacks*

   An adversary tunnels a message over a low latency link from one part of the network to another distant part of the network. Wormhole attacks can greatly disrupt the route discovery process, since routes will most likely be attracted to a wormhole that leads to a location close to the BS. This attack can be detected in WSNs using geographic routing since neighboring nodes know the physical locations between them and can relate to the expected geographic distance. Additionally anomalous traffic toward a node can be detected by host IDS design to detect wormhole attacks.

7. *Sinkhole attack*

   A sinkhole is similar to a wormhole in that they both aim to attract incoming traffic, except that a sinkhole aims to subsequently tamper with data or drop incoming packets (blackhole/selective forwarding). Sinkhole attacks are hard to carry out in a WSN using geographic routing for the same reasons as wormhole attacks. One can also detect sinkhole arracks by abnormal traffic directed to a node at the host IDS level.

8. *HELLO floods*

   A HELLO flood attack is when a node uses long distance transmissions to trick a node into thinking that it is within close proximity to it. This attack is similar to wormhole in that it can be coped with in WSNs using geographic routing.

At the host IDS level, most attacks such as blackhole and selective forwarding can be detected through monitoring and overhearing. More sophisticated attacks such as Sybil

and bad-mouthing attacks may not be easily detectable. We model the detection error by a false negative probability (missing a bad node as a good node) and a false positive probability (misidentifying a good node as a bad node). Inside attackers can be persistent or random. A persistent attacker performs attacks at full force, and does not attempt to conceal itself. A random attacker performs attacks only randomly to elude detection. This dissertation research investigates the effect of inside attacker behavior on dynamic redundancy management of multisource multipath routing for intrusion tolerance and voting-based IDS for intrusion detection in WSNs.

Chapter 4

Redundancy Management of Multisource Multipath Routing for Intrusion and Fault Tolerance in Homogeneous Clustered WSNs

Multisource multipath data routing to a remote sink node is an effective way to cope with unreliable and malicious nodes in autonomous WSNs. In this chapter we analyze the optimal amount of redundancy in terms of the number of source SNs sensing the same physical phenomena and the number of paths through which data are routed to a remote sink node in the presence of unreliable and malicious nodes so that the query success probability is maximized while maximizing the WSN lifetime. Our dynamic multisource multipath routing algorithm design integrates with a voting-based distributed intrusion detection algorithm to remove malicious nodes from the WSN. By controlling the redundancy level for multisource multipath and intrusion detection settings dynamically with energy considerations as prescribed by our algorithm, we demonstrate that the lifetime of a query-based autonomous WSN is maximized in response to changing environment conditions including node density, radio range, and node capture rate. This chapter is based on our work published in [8].

4.1 System Model

We consider a WSN with low-power SNs distributed in a geographic area through air-drop. SNs are homogenous with the same initial energy (E_o). The deployment area of the WSN is of size A^2. SNs are distributed according to a homogeneous spatial Poisson process with intensity λ. We assume the domain is relatively free of obstacles and the WSN is dense enough so that the length of a path connecting two SNs can be approximated by the straight line distance divided by r. The transmission power is kept to a minimum such that one-hop radio range (r) is used for transmission. Thus, any communication between two nodes with a distance greater than r between them would require a multi-hop. The one-hop radio range can be adjusted to maintain connectivity as

the network becomes less dense because of node failures at the expense of more energy consumption.

Environment conditions which could cause a SN to fail with a certain probability include hardware failure (q), and transmission failure due to noise and interference (e). Moreover, the WSN is vulnerable to sensor captures, i.e., SNs may be captured and compromised. Because of random deployment of SNs (e.g., air drop), we assume all SNs have equal chances of capture with the capture time characterized by a distribution function $F_c(t)$ based on historical data and knowledge about the application environment.

The WSN we consider in this chapter is cluster-based, where CHs are elected periodically using an energy-saving clustering algorithm (e.g., [68, 81, 126]), and form clusters with non-CH nodes. Each node elects itself to become a tentative CH with probability p. If a node decides to become a CH it broadcasts its residual energy by a CH announcement message. This happens in multiple iterations with every node doubling its p every iteration if it is not a CH already, or it has received a message from a CH with a higher residual energy than itself. After a specified number of iterations, the cluster-join process occurs where every non-CH will send a message to a CH informing that it will join the cluster. The CH will send back an acknowledgement to the SN. This clustering algorithm ensures that the energy spent due to being the CH is distributed fairly evenly among nodes by performing a fair rotation of the CH role among SNs.

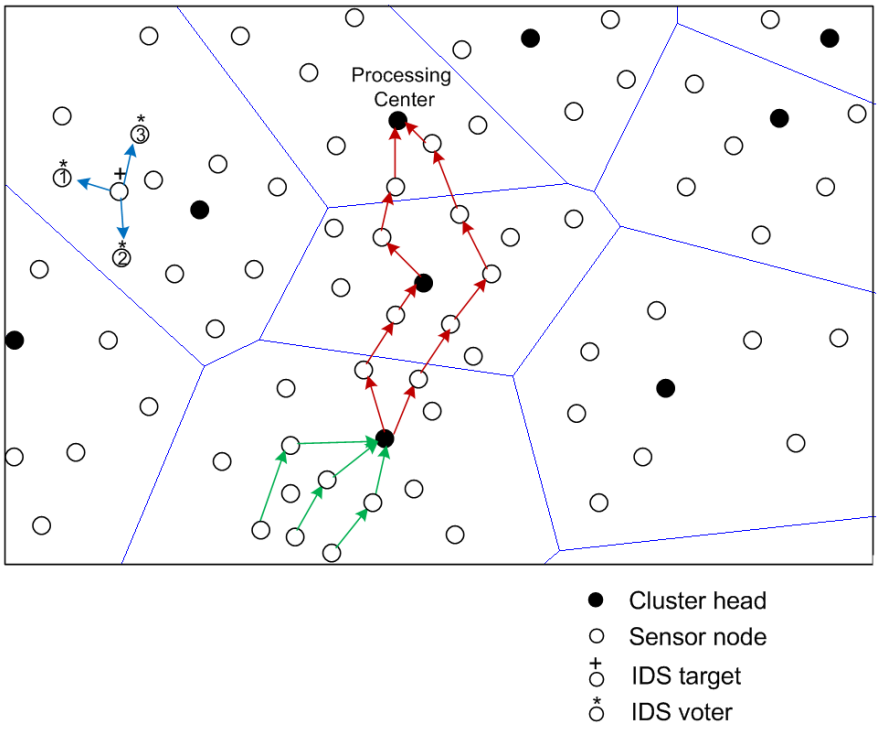


Figure 4-1: A Homogeneous Clustered WSN with Multisource Multipath Routing and Voting-based IDS.

Queries can be issued by a mobile user (while moving) and can be issued anywhere in the WSN through a nearby CH. A CH which takes a query to process is called a query processing center (PC). Each query has a strict timeliness requirement (T_{req}) before which the query must be delivered; otherwise, the query fails.

Figure 4-1 (bottom middle) shows an example of multisource multipath routing with sensing data being relayed from 3 source SNs to the source CH (m_s=3) and then from the source CH to the PC over 2 paths (m_p=2). Each source SN forms a disjoint path to the source CH. That is, one SN is chosen to relay the sensing data in each hop. Thus, there are a total of m_s paths from m_s source SNs. The source CH in turn creates m_p disjoint paths between the source CH and the PC. The m_p disjoint paths are formed by randomly choosing distinct m_p SNs in the first hop and then randomly choosing only one SN in each of the subsequent hops. Each query has a unique id, so an intermediate SN selected for packet forwarding can check the query id associated with the packet routing request to ensure that it is only committed once for a query. It has been reported that the number of edge-disjoint paths between nodes is equal to the average node degree with a very high probability [51]. Therefore, when the density is sufficiently high such that the average number of one-hop neighbors is sufficiently larger than m_p and m_s, we can effectively result in m_p disjoint paths for path redundancy and m_s disjoint paths from m_s sensors for source redundancy.

Geographic forwarding is used to route the information between nodes; thus, no path information is maintained. Only the location of the destination SN needs to be known to correctly forward a packet. As part of clustering, a CH knows the locations of SNs within its cluster, and vice versa.

We assume that the WSN executes a pairwise key establishment protocol (e.g., [65]) in a secure interval after deployment. Each node establishes pairwise keys with its k-hop neighbors, where k is large enough to cover a cluster area. Thus, upon electing a new CH, the CH will have pairwise keys with the SNs joining its cluster. Since every SN shares a pairwise key with its CH, a SN can encrypt data sent to the CH for confidentiality and authentication purposes.

We assume that SNs operate in power saving mode (e.g. [22, 107]). Thus, a SN is either active (transmitting or receiving) or in sleep mode. For the transmission and reception energy consumption of sensors, we adopt the energy model in [126].

Multisource multipath routing is achieved through two forms of redundancy: (a) source redundancy by which m_s SNs sensing a physical phenomenon in the same feature zone are used to forward sensing data to their CH, referred to as a source CH; (b) path redundancy by which m_p paths are used to relay packets from the source CH to the PC. It has been reported that the number of edge-disjoint paths between nodes is equal to the average node degree with a very high probability [19, 49, 108]. Therefore, when the density is sufficiently high such that the average number of one-hop neighbors is sufficiently larger than m_p and m_s, we can effectively result in m_p redundant paths for path redundancy and m_s distinct paths from m_s sensors for source redundancy.

Geographic forwarding is used to route the information between nodes; thus, no path information is maintained. Only the location of the destination SN needs to be known to correctly forward a packet. As part of clustering, a CH knows the locations of SNs within its cluster, and vice versa. We assume that SNs operate in power saving mode (e.g. [22, 107]). Thus, a SN is either active (transmitting or receiving) or in sleep mode. For the transmission and reception energy consumption of sensors, we adopt the energy model in [81].

We assume that the WSN executes a pairwise key establishment protocol (e.g., [65, 130]) in a secure interval after deployment. Each node establishes pairwise keys with its k-hop neighbors, where k is large enough to cover a cluster area. Thus, upon electing a new CH, the CH will have pairwise keys with the SNs joining its cluster. Since every SN shares a pairwise key with its CH, a SN can encrypt data sent to the CH for confidentiality and authentication purposes. Due to limited resources, we assume that when a node is compromised, it only performs two most energy conserving attacks, namely, bad-mouthing attacks (recommending a good node as a bad node and a bad node as a

good node) when serving as a recommender, and packet dropping attacks when performing packet routing to disrupt the operation of the network.

To detect and remove malicious nodes from the system, a voting-based system IDS is applied periodically in every T_{IDS} time interval. How often should T_{IDS} be is a design issue which we aim to identify in this paper. Every node runs a simple rule-based host IDS to detect if its neighbors exhibit suspicious behavior or comply with the prescribed protocol design, including the packet forwarding protocol, the voting-based system IDS protocol, and the clustering algorithm. An example is that when A selects B to forward a packet, A can monitor if B actually forwards the packet or not through overhearing based on the packet broadcast by B. If a neighbor node selected for packet forwarding does not forward as intended, then it is counted as a negative experience. Another example is that when A and B are involved in a voting-based system IDS activity on C, if A is sure C is good but B insists on C being bad, then A can view it as a negative experience against B. To conserve energy, a node does not promiscuously monitor all neighbors, but just uses packets received or overheard during protocol execution according to the rules defined in the host-IDS design (e.g., [19, 49]). The flaw of the host IDS is characterized by a false positive probability (H_{pfp}) and a false negative probability (H_{pfn}), which are assumed known at deployment time. In each interval, m neighbor nodes around a target node will be chosen randomly as voters to decide if the target node is still a good node.

Figure 4-1 (upper left) shows an example of voting-based intrusion detection with m = 3 voters. The m voters share their votes through secure transmission using their pairwise keys. How big should m be is another design issue which we aim to identify in this paper. When the majority of voters come to the conclusion that a target node is bad, then the target node is evicted. There is a system-level false positive probability (P_{fp}) that the voters can incorrectly identify a good node as a bad node. There is also a system-level false negative probability (P_{fn}) that the voters can incorrectly misidentify a bad node as a good node. These two system-level IDS probabilities will be derived based on the attack model in the paper.

To provide a unifying metric that considers the above two design tradeoffs, we define the total number of queries the system can answer correctly until it fails as the *lifetime* or the *mean time to failure* (MTTF) of the system which can be translated into the actual system lifetime span based on the query arrival rate.

Figure 4-2 shows a scenario where the percentage of compromised nodes is low relative to that of Figure 4-3. In Figure 4-2 we illustrate that we only need to choose a source redundancy of 2 (m_s = 2) and a path redundancy of 2 (m_p = 2) to maintain the source-SN-to-PC connectivity for query response delivery. As the bad node population increases as

in Figure 4-3, we need to use m_s = 3 and m_p = 3 to maintain the source-SN-to-PC connectivity.

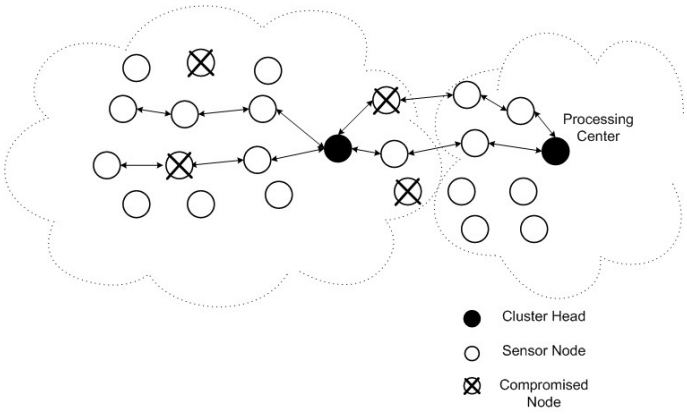


Figure 4-2: Multisource Multipath Routing with a Low Population of Compromised Nodes.

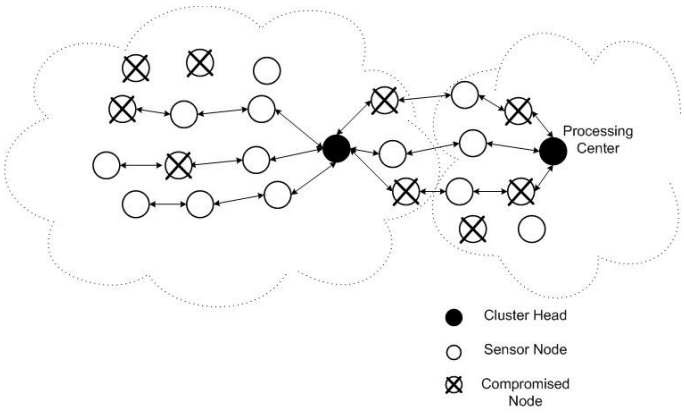


Figure 4-3: Multisource Multipath Routing with a High Population of Compromised Nodes.

Here we note that increasing source or path redundancy enhances reliability and security of source-SN-to-PC connectivity. However, it also increases the energy consumption, thus contributing to the decrease of the system lifetime. Thus, there is a tradeoff between reliability and security gain vs. energy consumption. The distributed IDS design attempts to detect and evict compromised nodes from the network without unnecessarily wasting energy so as to maximize the query success probability and the system lifetime. The effectiveness of the IDS depends on its parameters (T_{IDS} and m). While a shorter T_{IDS} or a higher m can result in low P_{fp} and P_{fn}, it also consumes more energy from the WSN nodes. Thus, this is another design tradeoff. To provide a unifying metric that considers the above two design tradeoffs, we define the total number of queries the

system can answer correctly until it fails as the *lifetime* or the *mean time to failure* (MTTF) of the system which can be translated into the actual system lifetime span based on the query arrival rate. A failure occurs when no response is received before the query deadline. The cause could be due to energy exhaustion, packet dropping by malicious nodes, channel/node failure, or insufficient transmission speed to meet the timeliness requirement. Our aim is to find both the optimal redundancy levels and IDS settings under which the MTTF is maximized, when given a set of parameters characterizing the operational and environment conditions.

4.2 Probability Model

Table 4-1: Parameter List.

Parameter	Meaning	Type
A	Length of each side of a square sensor area (meter)	input
n_b	Size of a data packet (bit)	input
E_{elec}	Energy dissipation to run the transmitter and receiver circuitry (J/bit)	input
E_{amp}	Energy used by the transmit amplifier to achieve an acceptable signal to noise ratio (J/bit/m^2)	input
E_o	Initial energy per SN (Joule)	input
E_{init}	Initial energy of the WSN (Joule)	derived
$E_{clustering}(t)$	Energy consumed for executing the clustering algorithm at time t (Joule)	derived
$E_{IDS}(t)$	Energy consumed for executing the IDS algorithm at time t (Joule)	input
$E_q(t)$	Energy consumed for executing a query at time t (Joule)	derived
$R_q(t)$	Probability that a query reply at time t is delivered successfully by the deadline	derived
r	Wireless radio communication range (meter)	input
p	Probability of a SN becoming a CH	derived
q	SN hardware failure probability	input
e_j	Transmission failure probability of SN_j	input
$N(t)$	Number of SNs in the WSN at time t	input
$n(t)$	Number of neighbor SNs at time t	derived
$n_{good}(t)$	Number of good neighbor SNs at time t	derived
$n_{bad}(t)$	Number of bad neighbor SNs at time t	derived
N_q	Maximum number of queries before energy exhaustion	derived
$N_{iteration}$	Number of iterations in clustering for CH election	derived
m_p	Path redundancy level: Number of paths from a source CH to the sink	design
m_s	Source redundancy level: Number of SNs per cluster in response to a query	design

f	Fraction of neighbor SNs that will forward data	input
$\lambda(t)$	SN population density (nodes/meter^2) at time t	derived
λ	SN population density at deployment time	input
λ_q	Query arrival rate (times/sec)	input
S_{jk}	Progressive transmission speed between SN_j and SN_k (meter/sec)	derived
$T_{clustering}$	Time interval for executing the clustering algorithm (sec)	input
T_{req}	Query deadline requirement (sec)	input
λ_c	Node capture rate	input
α	Ratio of IDS execution rate to query arrival rate	input
β	Ratio of clustering rate to query arrival rate	input
m	Number of voters selected for executing distributed IDS	design
H_{pfp}	Probability of host IDS false positive	input
H_{pfn}	Probability of host IDS false negative	input
Pfp	Probability of distributed IDS false positive	derived
Pfn	Probability of distributed IDS false negative	derived
T_{IDS}	IDS interval time (sec)	design
MTTF	Lifetime of a query-based WSN	output

Table 4-1 lists the parameter list along with the physical meaning and range. To detect and remove malicious nodes from the system, a voting-based distributed IDS is applied periodically in every T_{IDS} time interval. How often should T_{IDS} be is a design issue which we aim to identify in this dissertation research. Every node runs a simple *host IDS* using. In this section we develop a probability model to estimate the MTTF of an autonomous WSN using multisource multipath data forwarding to answer queries issued from a mobile user roaming in the WSN area. The basic idea of our MTTF formulation is that we first deduce the maximum number of queries, N_q, the system can possible handle before running into energy exhaustion for the best case in which all queries are processed successfully. Because the system evolves dynamically, the amount of energy spent per query also varies dynamically. Given the query arrival rate λ_q as input, we can reasonably estimate the amount of energy spent due to query processing and intrusion detection for query j based on the query arrival time $t_{Q,j}$. Next we derive the corresponding query success probability $R_q(t_{Q,j})$, that is, the probability that the response to query j arriving at time $t_{Q,j}$ is delivered successfully before the query deadline expires. Finally, we compute MTTF as the probability-weighted average of the number of queries the system can handle without experiencing any deadline, transmission, or security failure. More specifically, the MTTF is given by:

$$MTTF = \sum_{i=1}^{N_{q-1}} i\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)\left(1 - R_q(t_{Q,i+1})\right) + N_q \prod_{j=1}^{N_q} R_q(t_{Q,j}) \quad (4.1)$$

Here $\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)\left(1 - R_q(t_{Q,i+1})\right)$ accounts for the probability of the system being able to successfully execute i consecutive queries but failing the $i+1^{th}$ query. The second term is for the best case in which all queries are processed successfully without experiencing any failure for which the system will have the longest lifetime span.

4.2.1 Network Dynamics

Initially at deployment all SNs are good nodes. Assume that the capture time of a SN follows a distribution function $F_c(t)$ which can be determined based on historical data and knowledge about the target application environment. Then, the probability that a SN is compromised at time t, given that it was a good node at time t-T_{IDS}, denoted by P_c, is given by:

$$\begin{aligned} P_C &= 1 - P\{X > t \mid X > t - T_{IDS}\} \\ &= 1 - \frac{P\{X > t, X > t - T_{IDS}\}}{P\{X > t - T_{IDS}\}} = 1 - \frac{1 - F_c(t)}{1 - F_c(t - T_{IDS})} \end{aligned} \quad (4.2)$$

We note that P_c is time dependent. For the special case in which the capture time is exponential distributed with rate λ_c, $P_c = 1 - e^{-\lambda_c \times T_{IDS}}$. Recall that the voting-based distributed IDS executes periodically with T_{IDS} being the interval. At the i^{th} IDS execution time (denoted by $t_{I,i}$), a good node may have been compromised with ity P_c since the previous IDS execution time $(t_{I,i-1})$. Let $n_{good}(t)$and $n_{bad}(t)$ denote the numbers of good and bad neighbor nodes at time t, respectively, with $n_{good}(t) + n_{bad}(t) = n(t)$. Then, the population of good and bad neighbor nodes at time $t_{I,i}$ just prior to IDS execution can be recursively estimated from the population of good and bad neighbor nodes at time $t_{I,i-1}$:

$$\begin{aligned} n_{good}(t_{I,i}) &= n_{good}(t_{I,i-1}) - n_{good}(t_{I,i-1}) \times P_c \\ n_{bad}(t_{I,i}) &= n_{bad}(t_{I,i-1}) + n_{good}(t_{I,i-1}) \times P_c \end{aligned} \quad (4.3)$$

With $n_{good}(t)$ and $n_{bad}(t)$ in hand, the system-level false positive probability (P_{fp}) and false negative probability (P_{fn}) as a resulting of executing voting-based IDS are as follows:

(4.4)

$$P_{fp}\ or\ P_{fn} \equiv \sum_{i=0}^{m-m_{maj}} \left[\frac{C\binom{n_{bad}}{m_{maj}+i} \times C\binom{n_{good}}{m-(m_{maj}+i)}}{C\binom{n_{bad}+n_{good}}{m}} \right] + \sum_{i=0}^{m-m_{maj}} \left[\frac{C\binom{n_{bad}}{i} \times \sum_{j=m_{maj}-i}^{m-i} \left[C\binom{n_{good}}{j} \times \omega^{j} \times C\binom{n_{good-j}}{m-i-j} \times (1-\omega)^{m-i-j} \right]}{C\binom{n_{bad}+n_{good}}{m}} \right]$$

where m_{maj} is the minimum majority of m, e.g., 3 is the minimum majority of 5, and ω is H_{pfp} for calculating P_{fp} and H_{pfn} for calculating P_{fn}. We explain Equation 4.4 for the false positive probability at time t below. The explanation to the false negative probability is similar. A false positive results when the majority of the voters vote against the target node (which is a good node) as compromised. The first term in Equation (4.4) accounts for the case in which more than 1/2 of the voters selected from the target node's neighbors are bad sensors who, as a result of performing bad-mouthing attacks, will always vote a good node as a bad node to break the functionality of the WSN. Here the denominator is the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select at least m_{maj} bad voters out of n_{bad} nodes and the remaining good voters out of n_{good} nodes. The second term accounts for the case in which more than 1/2 of the voters selected from the neighbors are good nodes but unfortunately some of these good nodes mistakenly misidentify the target nodes as a bad node with probability H_{pfp}, resulting in more than 1/2 of the voters (some of those may be bad nodes) voting against the target node. Here the denominator is again the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select i bad voters not exceeding the majority m_{maj}, j good voters who diagnose incorrectly with $i + j \geq m_{maj}$, and the remaining $m - i - j$ good voters who diagnose correctly.

After the voting-based IDS is executed, some good nodes will be misidentified as bad nodes with probability P_{fp} and will be mistakenly removed from the WSN. Consequently, we need to adjust the population of good nodes after IDS execution. Let

$\overline{n_{good}(t)}$ be the number of good neighbor nodes at time t right after IDS execution. Then,

$$\overline{n_{good}(t_{I,\iota})} = n_{good}(t_{I,i}) - n_{good}(t_{I,i}) \times P_{fp} \tag{4.5}$$

On the other hand, some bad nodes will remain in the system because the voting-based IDS fails to identify them with probability P_{fn}. Let $\overline{n_{bad}(t)}$ be the number of bad neighbor nodes at time t right after IDS execution. Then,

$$\overline{n_{bad}(t_{I,\iota})} = n_{bad}(t_{I,i}) - n_{bad}(t_{I,i}) \times (1 - P_{fn}) \tag{4.6}$$

As the capture attack is totally random, the probability that any neighbor node is a bad node at time t, denoted by $Q_{c,j}(t)$, thus is given by:

$$Q_{c,j}(t_{I,i}) = \frac{\overline{n_{bad}(t_{I,\iota})}}{\overline{n_{bad}(t_{I,\iota})} + \overline{n_{good}(t_{I,\iota})}} \tag{4.7}$$

$Q_{c,j}(t)$ derived above provides critical information as a bad node can perform packet dropping attacks if it is on a path from source SNs to the PC. Here we note that the node population density is evolving because of some nodes being compromised and some being detected and evicted by the IDS dynamically. The node population remains the same until the next IDS execution (after T_{IDS} seconds) because the IDS only detects and evicts nodes periodically (as typically node hardware/software failure happens less frequently than security failure). Denote the node population density at time t by $\lambda(t)$ with $\lambda(0)= \lambda$. Then, $\lambda(t)$ can be computed by:

$$n(t_{I,i}) = \overline{n_{bad}(t_{I,\iota})} + \overline{n_{good}(t_{I,\iota})} \tag{4.8}$$

$$\lambda(t_{I,i}) = \frac{n(t_{I,i})}{\pi r^2} \tag{4.9}$$

4.2.2 Query Success Probability

There are three ways by which data forwarding from SN_j to SN_k could fail: (a) transmission speed violation; (b) sensor/channel failures; and (c) SN_j is compromised.

The first source of failure, transmission speed violation, accounts for query deadline violation. To know the failure probability due to transmission speed violation, we first derive the minimum hop-by-hop transmission speed required to satisfy the query deadline T_{req}. Let $d_{SN\text{-}CH}$ be the *expected* distance between a SN (selected to report sensor readings) and its CH and $d_{CH\text{-}PC}$ be the *expected* distance between the source CH and the PC accepting the query result. Given a query deadline T_{req} as input, a data packet from a SN through its CH to the PC must reach the PC within T_{req}. Thus, the minimum hop-by-hop transmission speed denoted by S_{req} is given by:

$$S_{req} = \frac{d_{SN-CH} + d_{CH-PC}}{T_{req}} \tag{4.10}$$

Since a SN becomes a CH with probability p and all the sensors are distributed in the area in accordance with a spatial Poisson process with intensity λ, CHs and non-CH SNs will also be distributed in accordance with a spatial Poisson process with rates $p\lambda$ and $(1\text{-}p)\lambda$ respectively. Non-CH SNs thus would join the closest CH to form a Voronoi cell [12] corresponding to a cluster in the WSN. It can be shown that the average number of non-CH SNs in each Voronoi cell is $(1\text{-}p)/p$ and the expected distance from a SN to its CH is given by $d_{SN-CH} = 1/2(p\lambda)^{1/2}$. On the other hand, since a query may be issued from anywhere by the mobile user to a CH (which serves as the PC) and the source CH requested by the query also can be anywhere in the WSN, $d_{CH\text{-}PC}$ essentially is the average distance between any two CHs in the WSN. Given location randomness of CHs in the square area A^2, it can be shown geometrically that the average distance between any two CHs is $d_{CH\text{-}PC}$=0.382A. With the knowledge of $d_{SN\text{-}CH}$ and $d_{CH\text{-}PC}$, we can also estimate the average numbers of hops to forward data from a SN to the source CH, denoted by N_{SC}^h, and the average numbers of hops to forward data from the source CH to the PC, denoted by N_{CP}^h, by $N_{SC}^h = d_{SN-CH}/r$ and $N_{cp}^h = d_{CH-PC}/r$ where r is radio range.

Let $Q_{t,jk}$ denote the probability that the forwarding speed from SN_j to SN_k would violate the minimum speed requirement, thus leading to a query deadline violation failure. To calculate $Q_{t,jk}$ we need to know the transmission speed S_{jk} from SN_j to SN_k. This can be dynamically measured by SN_j following the approach described in [58]. If S_{jk} is above S_{req} then $Q_{t,jk}$ = 0; otherwise, $Q_{t,jk}$ = 1. In general S_{jk} is not known until runtime. If S_{jk} is uniformly distributed within a range [a, b], then $Q_{t,jk}$ can be computed as:

$$Q_{t,jk} = cdf(S_{jk} \le S_{req}) = \frac{S_{req} - a}{b - a} \tag{4.11}$$

The second source of failure is due to sensor failure or channel failure. Let $Q_{r,j}$ denote the probability of failure due to sensor failure or channel failure. Since q is the hardware failure probability and e_j is transmission failure probability of node j, given as input, $Q_{r,j}$ can be estimated by:

$$Q_{r,j} = 1-[(1-q)(1-e_j)] \tag{4.12}$$

The third source of failure is due to node j being compromised and thus the packet is dropped. We make use of $Q_{c,j}(t)$ derived earlier in Equation (4.7). By combining these three failure probabilities we obtain $Q_{rtc,jk}$, the probability of SN_j failing to relay a data packet to a one-hop neighbor SN_k because of either speed violation, sensor/channel failure, or SN_j being compromised, as:

$$Q_{rtc,jk} = 1-[(1-Q_{r,j})(1-Q_{t,jk})(1-Q_{c,j})] \tag{4.13}$$

By using this one-hop failure probability, we next compute the success probability for SN_j to transmit a packet to at least one next-hop SN neighbor along the direction of the destination node as:

$$\theta_j = 1-\prod_{k=1}^{f\times n} Q_{rtc,jk} \tag{4.14}$$

where f =1/4 to account for the fact that only neighbor SNs in the quadrant toward the destination node can perform geographic forwarding; n is the number of neighbor SNs of node j as given in Equation (4.8).

Since on average there will be N_{CP}^h hops on a path from the source CH to the PC, a data packet transmitted along the path is successfully delivered only if it is delivered successful hop-by-hop without experiencing any speed violation failure, hardware/channel failure, or packet dropping failure, for N_{CP}^h hops. Consequently, the probability of a single path between the source CH and the PC being able to deliver data successfully is given by:

$$\Theta(N_{CP}^h) = (\prod_{j=1}^{N_{CP}^h-1}\theta_j)\times(1-Q_{rtc,\ N_{CP}^h(N_{CP}^h+1)}) \tag{4.15}$$

For path redundancy, we create m_p paths between the source CH and the PC. The m_p paths are formed by choosing m_p SNs in the first hop and then choosing only one SN in each of the subsequent hops. The source CH will fail to deliver data to the PC if one of the following happens: (a) none of the SNs in the first hop receives the message; (b) in the first hop, i $(1 \le i < m_p)$ SNs receive the message, and each of them attempts to form a path for data delivery; however, all i paths fail to deliver the message because the subsequent hops fail to receive the broadcast message; (c) in the first hop, at least m_p SNs receive the message from the source CH from which m_p SNs are randomly selected to forward data, but all m_p paths fail to deliver the message because the subsequent hops fail to receive the message. Summarizing above, the probability of the source CH failing to deliver data to the PC is given by:

$$\begin{aligned} Q_{fp}^{m_p} &= 1-\theta_1 + \\ &\sum_{|I|<m_p} \{[\prod_{i\in I}(1-Q_{rtc,1i})](\prod_{i\notin I} Q_{rtc,1i})\}\{\prod_{i\in I}[1-\Theta_i(N_{CP}^h-1)]\} + \\ &\sum_{|I|\ge m_p} \{[\prod_{i\in I}(1-Q_{rtc,1i})](\prod_{i\notin I} Q_{rtc,1i})\}\{\prod_{\substack{i\in M,\\ M\subseteq I,\\ |M|=m_p}}[1-\Theta_i(N_{CP}^h-1)]\} \end{aligned} \tag{4.16}$$

Following the same derivation to Equation (4.15), the success probability of a single path from a SN to its CH is given by:

$$\Theta(N_{SC}^h) = (\prod_{j=1}^{N_{SC}^h-1}\theta_j)\times(1-Q_{rt,\,N_{SC}^h(N_{SC}^h+1)}) \tag{4.17}$$

For source redundancy we use m_s SNs to report query responses to their source CH. The probability that all m_s SNs fail to deliver data to their CH is given by:

$$Q_{fs}^{m_s} = \prod_{i=1}^{m_s}[1-\Theta_i(N_{SC}^h)] \tag{4.18}$$

Consequently, the failure probability of data delivery from m_s SNs to the CH, and subsequently using m_p paths to relay data from CH to PC, is given by:

$$Q_f = 1 - (1 - Q_{fp}^{m_p})(1 - Q_{fs}^{m_s}) \tag{4.19}$$

Therefore, the query success probability is given by:

$$R_q = 1 - Q_f \tag{4.20}$$

Note that in the above derivation we omit time for brevity. More precisely, R_q derived above should be $R_q(t_{Q,i})$ since the query success probability is a function of time, depending on the node count (Equation (4.8)) and population density (Equation (4.9)) at the i^{th} query's execution time (i.e., at time $t_{Q,i}$).

4.2.3 Energy Consumption

We estimate the amounts of energy spent during a query interval [$t_{Q,i}$, $t_{Q,i+1}$], an IDS interval [$t_{I,i}$, $t_{I,i+1}$], and a clustering interval [$t_{c,i}$, $t_{c,i+1}$], so as to estimate N_q, the maximum number of queries the system can possible handle before running into energy exhaustion. Our energy model follows [126]. Because of the randomness introduced in our protocol in CH election, IDS detection, and query processing, all SNs consume energy at about the same rate. Hence it suffices to consider the overall system energy level instead of individual SN energy levels for calculating the amount of time it takes for the system to exhaust energy. To normalize energy consumption over N_q queries, let α be the ratio of the IDS execution rate to the query arrival rate and let β be the ratio of the clustering rate to the query arrival rate so that αN_q and βN_q are the numbers of IDS cycles and clustering cycles, respectively, before system energy exhaustion. Then, we can estimate N_q by the fact that the total energy consumed due to intrusion detection, clustering and query processing is equal to the system energy as follows:

$$E_{init} = \sum_{i=1}^{\alpha N_q} E_{IDS}(t_{I,i}) + \sum_{i=1}^{\beta N_q} E_{clustering}(t_{C,i}) + \sum_{i=1}^{N_q} E_q(t_{Q,i}) \tag{4.21}$$

Below we outline how to calculate $E_{IDS}(t_{I,i})$, $E_{clustering}(t_{C,i})$ and $E_q(t_{Q,i})$. We first estimate energy consumed by transmission and reception over wireless link. The energy spent by a SN to transmit an encrypted data packet of length n_b bits over a distance r is estimated as [126]:

$$E_T = n_b(E_{elec} + E_{amp} r^2) \tag{4.22}$$

Here E_{elec} is the energy dissipated to run the transmitter and receiver circuitry, E_{amp} is the energy used by the transmit amplifier, and r is the transmission radio range. The energy spent by a SN to receive an encrypted message of length n_b bits is given by:

$$E_R = n_b E_{elec} \tag{4.23}$$

The energy consumed for processing the i^{th} query, $E_q(t_{Q,i})$, is the sum of the energy consumed through m_p paths for the communication between CH and PC, denoted by $E_{CP}(t_{Q,i})$, and the energy consumed for the communication between m_s source SNs and the CH, denoted by $E_{SC}(t_{Q,i})$, i.e.,

$$E_q(t_{Q,i}) = E_{CP}(t_{Q,i}) + E_{SC}(t_{Q,i}) \tag{4.24}$$

The energy consumed for the communication between CH and PC is due to setting up m_p paths in the first hop and subsequently transmitting data over the m_p paths, i.e.,

$$E_{CP}(t_{Q,i}) = \{ E_T + n(t_{Q,i}) E_R \} + \{ m_p (N_{CP}^h - 1) [E_T + n(t_{Q,i}) E_R] \} \tag{4.25}$$

Here the first term accounts for the transmission energy consumed by the source CH and the reception energy consumed by its 1-hop SNs for setting up the m_p paths, and the second term accounts for the energy consumed for data transmission over the m_p paths in the remaining $N_{CP}^h - 1$ hops. We note that the number of neighbor SNs at time t, $n(t) = \lambda(t) \times \mathrm{p}r^2$ by Equation (4.9), depends on the SN population density at time t, i.e., $\lambda(t)$.

The energy consumed for the communication between source SNs and the CH is due to transmitting data over the m_s paths each with N_{SC}^h hops, i.e.,

$$E_{SC}(t_{Q,i}) = m_s N_{SC}^h [E_T + n(t_{Q,i}) E_R] \tag{4.26}$$

For clustering, the system would consume energy for broadcasting the announcement message and for the cluster-join process. Since p is the probability of a SN becoming a CH, there will be $p \times N(t_{c,i})$ SNs that would be broadcasting the announcement message where $N(t_{c,i}) = \lambda(t_{c,i}) \times A^2$ is the number of SNs in the WSN at time $t_{c,i}$. This an-

nouncement message will be received and retransmitted by each SN to the next hop until the TTL of the message reaches the value 0, i.e., the number of hops equals N_{SC}^{h}. Thus, the energy required for broadcasting is $pN(t_{C,i})[N_{SC}^{h}n(t_{C,i})(E_T + E_R)]$. The cluster-join process will require a SN to send a message to the CH informing that it will join the cluster and the CH to send an acknowledgement to the SN. Since there are $p\ N(t_{c,i})$ CHs and $(1-p)\ N(t_{c,i})$ SNs in the system, the energy for this is $N(t_{c,i})\ (E_T + E_R)$. Let $N_{iteration}$ be the number of iteration required to execute the clustering algorithm. Then, the energy required for executing the clustering algorithm at time $t_{c,i}$, $E_{clustering}(t_{c,i})$, is given by:

$$E_{clustering}(t_{c,i}) = pN(t_{c,i})N_{iteration}\left[N_{SC}^{h}n(t_{c,i})(E_T + E_R)\right] + N(t_{c,i})(E_T + E_R) \quad (4.27)$$

Lastly, for intrusion detection every node is evaluated by m voters in an IDS cycle, and each voter sends its vote to the other m -1 voters. Hence, the energy spent in each voting-based IDS cycle is given by:

$$E_{IDS}(t_{I,i}) = N(t_{I,i-1})[m(m-1)]\left[E_T + n(t_{I,i-1})E_R\right] \quad (4.28)$$

Once we obtain $E_{IDS}(t_{I,i})$, $E_{clustering}(t_{C,i})$ and $E_q(t_{Q,i})$ from Equations (4.28), (4.27) and(4.24), respectively, we calculate N_q from Equation (4.21). The knowledge of N_q along with $R_q(t_{Q,i})$ in Equation (4.20) allows us to calculate the system MTTF given by Equation (4.1).

4.3 Generalizations

4.3.1 Coping with data modification attacks

In this section, we extend our analytical model to the case in which data modification attacks may occur. This is possible for homogeneous WSNs because any node can become a CH and a SN may be far away from the destination PC, thus making it infeasible to establish a pairwise key between a SN and the PC.

To tolerate packet modification attacks, a majority of paths must return successfully to the PC. That is, unless a majority of paths return successfully with correct SN readings, the packet modification attack will not be recognized by the PC. Thus, the probability of the source CH failing to deliver data to the PC taking into account data modification attacks is given in Equation (4.29) below which replaces Equation (4.16).

$$
\begin{aligned}
Q_{fp}^{m_p} &= 1-\theta_1 + \\
&\sum_{|I|<m_p} \{[\prod_{i\in I}(1-Q_{rtc,1i})](\prod_{i\notin I} Q_{rtc,1i})\} + \\
&\sum_{|I|\geq m_p} \{[\prod_{i\in I}(1-Q_{rtc,1i})](\prod_{i\notin I} Q_{rtc,1i})\} \{\prod_{\substack{i\in M,\\ M\subseteq I,\\ |M|=m_p}} [1-\Theta_i(N_{CP}^h-1)]\}
\end{aligned}
\tag{4.29}
$$

Following a similar derivation, Equation (4.30) below replaces Equation (4.18).

$$
\begin{aligned}
Q_{fs}^{m_s} = & \sum_{|I|\geq \left\lceil \frac{m_s}{2} \right\rceil} \{\prod_{i\in I}[1-\Theta_i(N_{SC}^h)]\} \{\prod_{i\notin I}\Theta_i(N_{SC}^h)\} \\
& + \sum_{|I|\geq \left\lceil \frac{m_s}{2} \right\rceil} \{\prod_{i\notin I}[1-\Theta_i(N_{SC}^h)]\} \{\prod_{i\in I}\Theta_i(N_{SC}^h)\} \\
& \times \prod_{i=1}^{|I|}[1-\Theta_i(N_{intra}^h)]
\end{aligned}
\tag{4.30}
$$

4.3.2 Fail-safe hardware

In this section, we extend the analytical model to the case in which secure hardware can effectively prevent a captured node from turning into an inside attacker. To model the usage of tamper-proof fail-safe sensor nodes, we replace Equation (4.3) with Equation (4.31) below:

$$
\begin{aligned}
n_{good}(t_{I,i}) &= n_{good}(t_{I,i-1}) - n_{good}(t_{I,i-1}) \times P_c \\
n_{bad}(t_{I,i}) &= n_{bad}(t_{I,i-1}) + n_{good}(t_{I,i-1}) \times P_c \times P_t
\end{aligned}
\tag{4.31}
$$

Here P_t is the probability of successful tampering. Tamper-proof-fail-safe sensors can be modeled by setting $P_t = 0$. This will result in captured nodes being removed from the good SN population, but not being added into the bad SN population. Setting $P_t = 1$ models the deployment of SNs without fail-safe capabilities.

4.4 Algorithm for Dynamic Multisource Multipath Routing

The objective of dynamic multisource multipath routing is to dynamically identify and apply the best redundancy level in terms of path redundancy (m_p) and source redundancy (m_s), as well as the best intrusion detection settings in terms of the number of voters (m) and the intrusion invocation interval (T_{IDS}) to maximize MTTF, in response to environment changes including node density (λ), radio range (r), and node capture rate (λ_c).

Our algorithm is distributed in nature. The algorithm specifies control actions taken by individual SNs and CHs in response to dynamically changing environments as follows:

1: ***CH Execution***:
2: *Get next event*
3: ***if*** *event is T_D timer* ***then***
4: *determine radio range to maintain CH connectivity*
5: *determine optimal T_{IDS}, m, m_s, m_p by table lookup based on the current estimated density, CH radio range and compromise rate*
6: *notify SNs within the cluster of the new optimal settings of T_{IDS} and m*
7: ***else if*** *event is query arrival* ***then***
8: *trigger multipath routing using m_s and m_p*
9: ***else if*** *event is $T_{clustering}$ timer* ***then***
10: *perform clustering*
11: ***else if*** *event is T_{IDS} timer* ***then***
12: ***For*** *each neighbor CH*
13: ***if*** *selected as a voter* ***then***
14: *execute voting based intrusion detection*
15: ***else*** *// event is data packet arrival*
16: *follow multipath routing protocol design to route the data packet*
17:
18: ***SN Execution***:
19: *Get next event*
20: ***if*** *event is T_D timer* ***then***
21: *determine radio range to maintain SN connectivity within a cluster*
22: ***else if*** *event is control packet arrival from CH* ***then***
23: *Change the optimal settings of T_{IDS}, and m*
24: ***else if*** *event is $T_{clustering}$ timer* ***then***
25: *perform clustering*
26: ***else if*** *event is T_{IDS} timer* ***then***
27: ***For*** *each neighbor SN*
28: ***if*** *selected as a voter* ***then***
29: *execute votingbased intrusion detection*
30: ***else*** *// event is data packet arrival*
31: *follow multipath routing protocol design to route the data packet*

Figure 4-4: Algorithm for Dynamic Redundancy Management.

All nodes in the system act periodically to a "T_D timer" event to adjust the optimal parameter setting in response to changing environments. This is indicated on line 3 for a CH and line 20 for a SN. The optimal design settings in terms of optimal $T_{IDS}, m, m_s,$ and m_p are determined at static design time (described in Section 4.4) and pre-stored in a table over perceivable ranges of input parameter values. As there is no base station in the system, the duty of performing a table lookup operation with interpolation and/or extrapolation techniques applied to determine the optimal design parameter settings will be assumed by CHs. The action performed by a CH upon a T_D timer event includes (a) adjusting radio range to maintain connectivity (line 4); (b) determining $T_{IDS}, m, m_s,$ and m_p (line 5) based on the sensed environmental conditions at runtime; and (c) notifying SNs within the cluster of the new T_{IDS} and m settings. The action performed by a SN upon this T_D timer event is to adjust its radio range to maintain connectivity within a cluster (line 21). The action taken upon receiving the control packet from its CH is to update the new T_{IDS} and m settings (line 23) for intrusion detection. When the T_{IDS} timer event happens, each node in the system uses it current T_{IDS} and m settings to perform intrusion detection. The T_{IDS} timer event and the action taken are specified on lines 11-14 for a CH and lines 26-29 for a SN.

When a CH acting as a query processing center (PC) receives a query from a user, it triggers multisource multipath routing for intrusion tolerance using the current optimal m_s and m_p settings to prolong the system useful lifetime. This query arrival event and the action taken are specified on lines 7-8. When a data packet arrival event occurs, each node simply follows the prescribed multipath routing protocol to route the packet (lines 15-16 for a CH and lines 30-31 for a SN). Finally each node periodically performs clustering as prescribed by the cluster algorithm, i.e., when a $T_{clustering}$ timer event occurs, each node executes clustering (lines 9-10 for a CH and lines 24-25 for a SN) for periodic cluster formation.

4.5 Performance Evaluation

In this section, we present numerical results obtained from the evaluation of our probability model given in Section 4.4. Without loss of generality, we consider an example WSN consisting of 1500 nodes deployed in a square area of A^2 (400m×400m). Nodes are distributed in the area following a Poisson process with density λ =15 nodes/(40×40 m^2) at deployment time. The radio range r is 40m. So initially a SN has n=$\lambda \times \pi r^2$=15 neighbor SNs. The probability of a SN becoming a CH is p=1%. So initially a cluster has 1/p=100 nodes and there are 15 clusters in the system. Each SN has an initial

energy level E_o = 10 Joules. The energy parameters used by the radio module are adopted from [68, 81, 126]. The energy dissipation E_{elec} to run the transmitter and receiver circuitry is 50 nJ/bit. The energy used by the transmit amplifier to achieve an acceptable signal to noise ratio (ε_{amp}) is 10 pJ/bit/m^2. The query arrival rate λ_q is a variable and is set to 1 query/sec to reveal points of interest. The query deadline T_{req} is strict and set to between 0.3 and 1 sec. The inter-arrival time in between captures (T_{comp}) is between 4 and 28 days, corresponding to a capture rate (λ_c) of once per 4 days to once per 28 days. The host IDS false positive probability and false negative probability (H_{pfp} and H_{pfn}) vary between 1% and 5% to reflect the host intrusion detection strength as in [49].

The set of parameter values characterizing the operating and environment conditions of our example WSN derives from prior works in the research community and real world projects. Our model is a large scale deployment comparable to the GreenOrbs project [97] for ecological forest monitoring and the CitySee project [97] for city CO2 monitoring with 1000+ and 4000+ sensor nodes respectively. The node density parameter values follow the work of He, et al. [77] where routing performance of LEACH, BCDCP, and DMSTRP protocols were compared. Our energy, sensor transmission, and clustering parameter values follow those in HEED [66] and LEACH [68, 126] both of which have been exhaustively explored and well cited by the research community. The timeliness requirement (T_{req}) values follow those in MMSPEED [58]. We adopt the parameter value ranges for the hardware failure (q) and transmission failure (e_j) from [38]. Finally, we follow [17] which concluded that an attacker could take days to reveal the secret keys of a captured node, depending on the node hardware and the tools needed by the attacker along with the strategy of attack. We therefore adopt the results and set the range of the compromise time parameter (T_{comp}) in the range of [4-28] days.

We summarize our example WSN by a set of input parameter values/ranges listed in Table 4-2. Our objective is to identify the best setting in terms of m_p (path redundancy), m_s (source redundancy), m (the number of voters for intrusion detection) and T_{IDS} (the intrusion detection interval) to maximize MTTF.

Table 4-2: Input Parameter Values Characterizing a Homogeneous Clustered WSN.

Parameter	Default Value
N	1500
p	0.01
q	10^{-6}
e_j	[0.0001 – 0.1]

r	40 m
f	¼
λ	15 nodes/(40 x 40 m^2)
λ_q	1 query/sec
T_{comp} (or 1/λ_c)	[4-28] days
A	400m
n_b	50 bits
E_{elec}	50 nJ/bit
ε_{amp}	10 pJ/bit/m^2
E_o	10 Joule
$N_{iteration}$	3
$T_{clustering}$	60 sec
T_{req}	[0.3 – 1.0] sec
H_{pfp} , H_{pfn}	[0.01-0.05]

Figure 4-5 shows a high level description of the computational procedure to determine the optimal redundancy level (m_p, m_s) for maximizing MTTF. The MTTF Equation (Equation (4.1) is embedded on lines 14-20 and 30-31 in Figure 4-5. The accumulation of queries is shown on line 12. The value of N_q is computed on line 32. The computational procedure essentially has a complexity of O($m_p \times m_s$) as it exhaustively searches for the best (m_p, m_s) pair, given a set of input parameter values as listed in Table 4-2 (above) as well as instance values of m (the number of voters for intrusion detection) and T_{IDS} (the intrusion detection interval) characterizing a query-based WSN.

Input: Table 42 input parameters
Output: $optimalMttf, optimal\ (m_p, m_s)$

1: **for** $m_s \leftarrow 1$ **to** $maxMs$ **do**
2: **for** $m_p \leftarrow 1$ **to** $maxMp$ **do**
3: $num_q \leftarrow 0$ $where\ num_q\ is\ the\ query\ counter$
4: $E_{init} \leftarrow N(t) \times E_o$ $at\ t = 0$
5: $Compute\ \lambda, R_q, E_{clustering}, E_q, E_{IDS}$ $at\ \ t = 0$

```
6:      Compute arrival time for next clustering, query, and IDS events
7:      while E_init > E_threshold do
8:          ev ← next event
9:          if ev is clustering event then
10:             E_init ← E_init − E_clustering
11:         else if ev is query event then
12:             num_q ← num_q + 1
13:             E_init ← E_init − E_q
14:             if num_q = 1 then //first query
15:                 rq_muls ← rq_muls × R_q
16:                 temp ← num_q × rq_muls
17:             else //terminate previous query in MTTF calculation
18:                 tempMttf ← tempMttf + temp × (1 − R_q)
19:                 rq_muls ← rq_muls × R_q
20:                 temp ← num_q × rq_muls
21:         else // ev is an IDS event
22:             Update distribution of good and bad nodes
23:             Compute Pfp/Pfn
24:             Update E_IDS
25:             E_init ← E_init − E_IDS
26:             Remove Bad caught and Good misidentified nodes
27:             Compute Q_c
28:             Update λ, N
29:             Update R_q, E_clustering, E_q
30:     tempMttf ← tempMttf + temp
31:     Mttf ← tempMttf
32:     N_q ← num_q
33:     if Mttf > optimalMttf then
34:         optimalMttf ← Mttf
35:         optimal (m_p, m_s) ← (m_p, m_s)
36:
37: return optimalMttf and optimal (m_p, m_s)
```

Figure 4-5: Computational Procedure to Determine Optimal (m_p, m_s) for Maximizing MTTF.

In Figure 4-6, we show MTTF vs. (m_p, m_s) for three cases: (a) there are no malicious nodes and no intrusion detection (the top curve); (b) there are malicious nodes but there is no intrusion detection (the bottom curve); (c) there are malicious nodes and there is intrusion detection (the middle two curves). In each case we observe the existence of an optimal (m_p, m_s) value under which MTTF is maximized. When there are no malicious nodes (the top curve), the optimal (m_p, m_s) is (3,3). When there are malicious nodes, and no intrusion detection is used, the optimal (m_p, m_s) value becomes (7,7) because using higher redundancy in multisource multipath routing is necessary to cope with malicious nodes. When intrusion detection is used (middle curves), there exists an optimal

m value to maximize MTTF. In Figure 4-6, m=5 yields a higher MTTF value than m=3 because in this scenario the attack rate is relatively high (once a week), so a higher number of voters is needed to cope with and detect bad nodes more effectively. We observe that the maximum MTTF is sensitive to T_{comp} and m. Table 4-3 below summarizes the effect of T_{comp} and m on optimal (m_p, m_s) values under which MTTF is maximized. As the number of voters in intrusion detection (m) increases, the optimal (m_p, m_s) redundancy level decreases. This is because increasing m has the effect of detecting and evicting bad nodes more effectively, thus requiring a lower level of redundancy in (m_p, m_s) to cope with packet dropping attacks by bad nodes.

On the other hand, when given a T_{comp} there exists an optimal m value that will maximize MTTF. Table 4-4 summarizes the effect of T_{comp} on the optimal m value at which MTTF is maximized. When the node capture rate increases from once per 3 weeks (T_{comp} = 3 weeks) to once a week (T_{comp} = 1 week), the optimal m value goes from 3 to 7. The reason is that as the capture rate increases, there are more and more malicious nodes in the system, so using more voters (e.g. m =7) can help identify and evict malicious nodes, thus increasing the query success probability and consequently increasing the MTTF value. Again the system is better off this way to cope with increasing malicious node population for lifetime maximization even though more energy is consumed due to more voters being used.

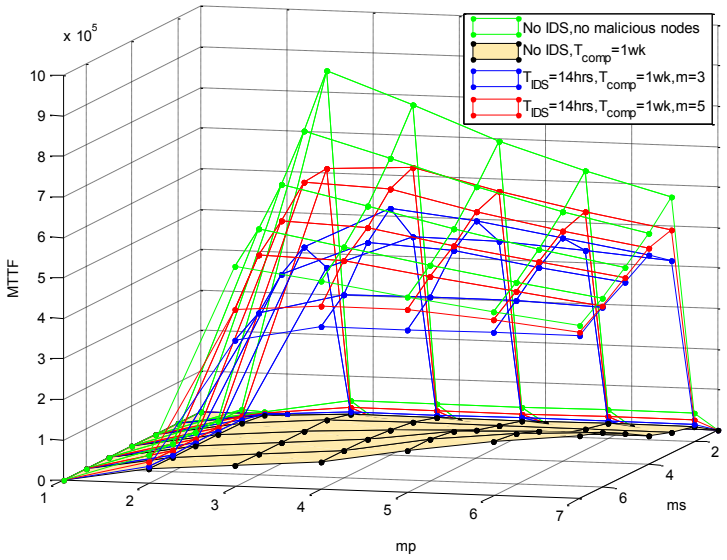
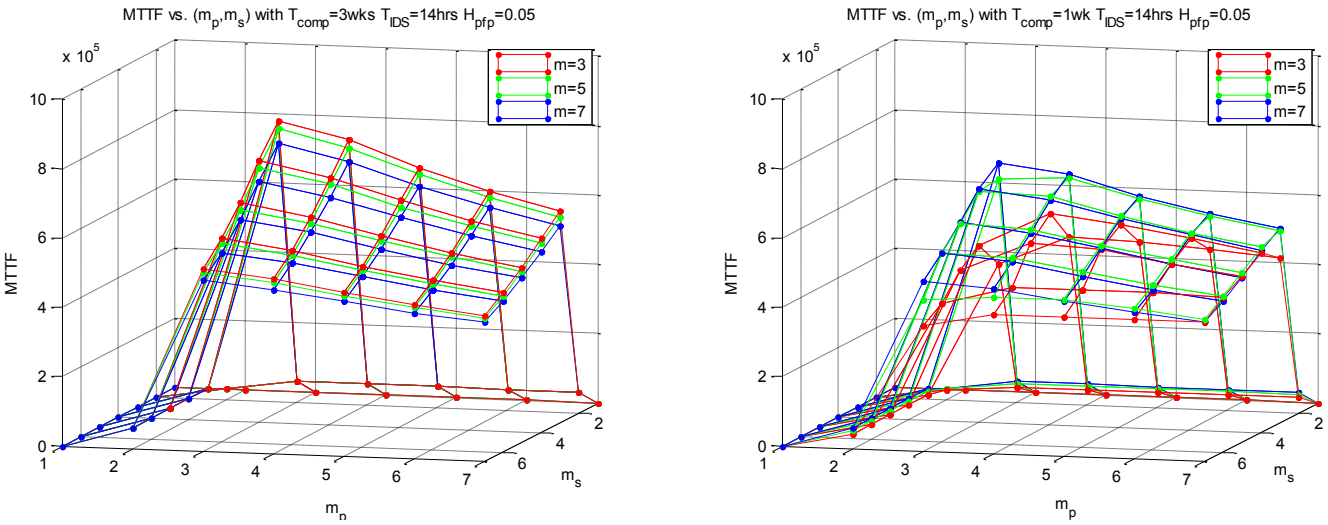


Figure 4-6: MTTF vs. (m_p, m_s): (3, 4) and (4, 4) are Optimal when m=5 and 3, Respectively.

We run the computational procedure to analyze the effect of T_{comp}, m and T_{IDS} on optimal (m_p, m_s). Figure 4-7 shows MTTF vs. (m_p, m_s) with varying T_{comp} and m values.

The left graph is for the case in which T_{comp} is large while the right graph is for the case in which T_{comp} is small. By comparing these two graphs, we observe a trend that as the capture rate increases (i.e., going from the left graph to the right graph), the optimal (m_p, m_s) redundancy level increases. For instance, when the capture rate increases from once in three weeks (T_{comp} = 3 weeks) to once a week (T_{comp} = 1 week), the optimal (m_p, m_s) redundancy level changes from (3, 3) to (4, 4). The reason behind this trend is that as more nodes are compromised in the system, a higher multisource multipath redundancy must be used to cope with packet dropping attacks. While increasing (m_p, m_s) consumes more energy, the gain towards increasing the query success probability (and thus towards increasing MTTF) outweighs the loss of lifetime due to energy consumption.



(a) Under Low Capture Rate

(b) Under High Capture Rate

Figure 4-7: Multisource Multipath Routing with a Low Population of Compromised Nodes.

Table 4-3: Optimal (m_p, m_s) with varying T_{comp} and m.

	m=3	5	7
T_{comp}=4 days	(m_p, m_s) =(5,7)	(4,6)	(4,5)
1 week	(4,4)	(3,4)	(3,3)
3 weeks	(3,3)	(3,3)	(3,3)

Table 4-4: Optimal m with varying T_{comp} and T_{IDS}.

	T_{IDS}=1hr	4hrs	14hrs	20hrs	24hrs
T_{comp}=4 days	5	7	7	7	7
1 week	5	5	7	7	7
2 weeks	5	5	5	5	7
3 weeks	3	3	3	5	5

Next we analyze the effect of T_{IDS} on MTTF. Figure 4-8 and Figure 4-9 show MTTF vs. T_{IDS} with varying m under low capture rate (T_{comp} = 3 weeks) and high capture rate (T_{comp} = 1 week), respectively. We first observe that there exists an optimal T_{IDS} value under which MTTF is maximized. Furthermore, the optimal T_{IDS} value increases as m increases. For example, in Figure 4-8 as m increases from 3, 5 to 7 we see that correspondingly the optimal T_{IDS} at which MTTF is maximized increases from 15, 32 to 46 hours. The reason is that as the number of voters increases so the intrusion detection capability increases per invocation, there is no need to invoke intrusion detection too often so as not to waste energy and adversely shorten the system lifetime. We also observe two general trends. One trend is that as T_{IDS} increases, the optimal m value increases. The reason is that when T_{IDS} is small so intrusion detection is invoked frequently, we don't need many voters per invocation so as not to waste energy unnecessarily to adversely shorten the system lifetime. The second trend shown in Figure 4-8 and Figure 4-9 is that as the node capture rate increases, the optimal m value increases in order to cope with more compromised nodes in the system. These two trends correlate well those summarized in Table 4-4 earlier.

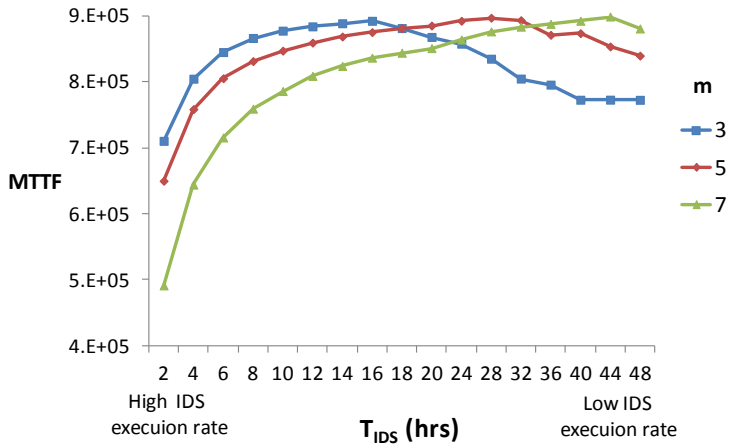


Figure 4-8: Effect of T_{IDS} on MTTF with varying m under Low Capture Rate.

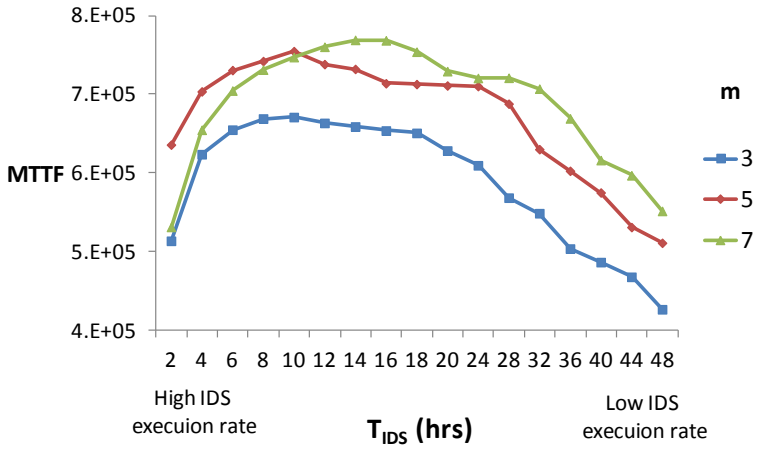


Figure 4-9: Effect of T_{IDS} on MTTF with varying m under High Capture Rate.

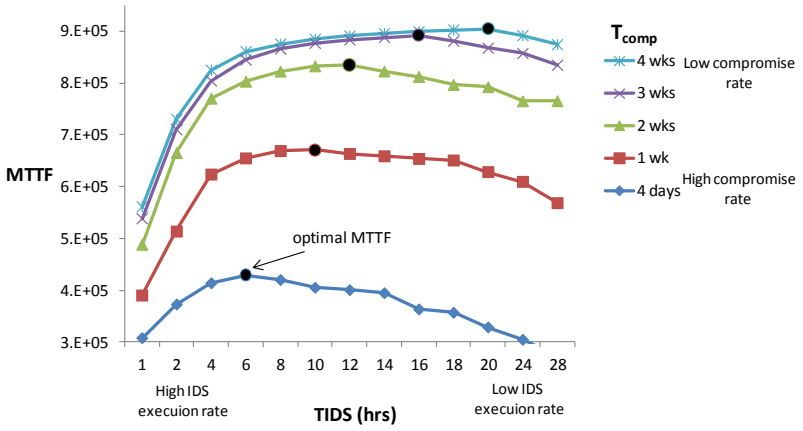


Figure 4-10: Effect of Capture Rate on MTTF.

Lastly T_{IDS} is also a tunable parameter to maximize MTTF. Figure 4-10 shows MTTF vs. T_{IDS} with varying T_{comp} values. It exhibits the trend that as the capture rate increases (a smaller T_{comp} value), the optimal T_{IDS} at which MTTF is maximized must decrease to cope with malicious attacks. For example, in Figure 4-10 the optimal T_{IDS} is 20 hours when T_{comp} = 4 weeks and reduces to 6 hours when T_{comp} = 4 days. Furthermore, the optimal T_{IDS} value increases as m increases. The reason is that as the number of voters (m) increases so the intrusion detection capability increases per invocation, there is no need to invoke intrusion detection too often so as not to waste energy and adversely shorten the system lifetime.

Table 4-5 summarizes the effect of T_{comp} and m on the optimal T_{IDS} value at which MTTF is maximized.

Table 4-5: Optimal T_{IDS} with varying T_{comp} and m.

	m =3	5	7
T_{comp}=4days	6	6	10
1 week	10	10	14
2 weeks	12	20	28
3 weeks	16	28	44

Table 4-3, Table 4-4, and Table 4-5 presented above are numerical solutions generated from evaluating the analytical equations derived in Section 4.2, given node density λ, radio range r, and node capture rate λ_c as input. As the system evolves, all these input parameter values may change, that is, λ will decrease as described by Equation (4.9) , radio range r will increase to maintain connectivity as more nodes fail or are evicted from the system, and λ_c may evolve depending on the instantaneous attacker strength. Lookup tables such as Table 4-3, Table 4-4, and Table 4-5 are built at static time, covering a wide range of (λ, r, λ_c) values as input. Our dynamic multisource multipath routing algorithm then utilizes these lookup tables built at static time to perform a simple lookup operation to decide the optimal settings of (m_p, m_s, m, T_{IDS}) to maximize MTTF at runtime.

4.6 Comparative Performance Analysis

We perform a comparative analysis of our dynamic redundancy management algorithm against AFTQC [38]. AFTQC is capable of dynamically identifying and applying the best (m_p, m_s) setting for query processing to maximize the WSN lifetime. However, it is designed for fault/intrusion tolerance only with no consideration given to intrusion detection of compromised nodes. For fair comparison, we tailor the probability model developed in Section 4.2 without IDS for AFTQC to identify the optimal (m_p, m_s). In Figure 4-11, we show MTTF vs. λ_c under our dynamic redundancy management algorithm against AFTQC, both operating at the optimal setting. The optimal setting in terms of the optimal (m_p, m_s) for each data point is labeled in the graph.

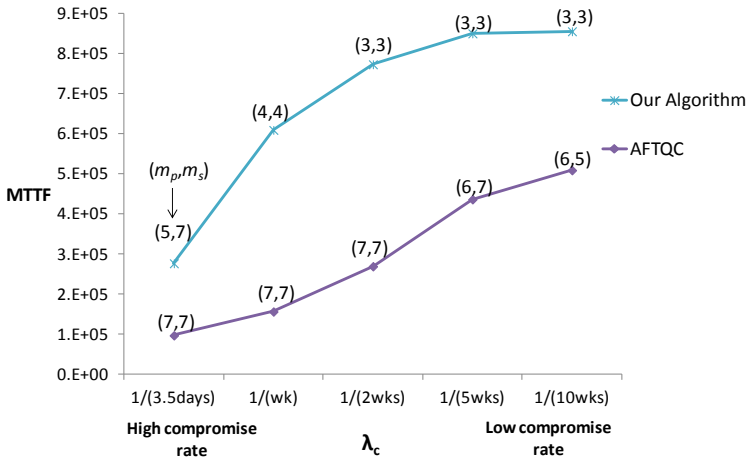


Figure 4-11: Performance Comparison with AFTQC.

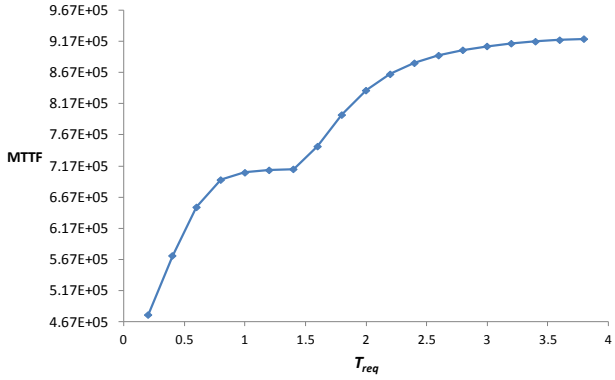
Our redundancy management algorithm significantly outperforms AFTQC in terms of MTTF obtainable under a wide range of capture rate values, the effect is especially pronounced when the capture rate is high. The result demonstrates the effectiveness of

our dynamic redundancy management algorithm which considers both fault intrusion tolerance through multipath routing and intrusion detection through voting, as opposed to AFTQC which considers fault/intrusion tolerance only through multipath routing.

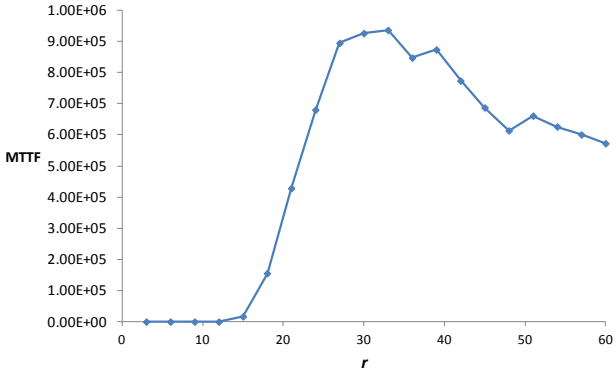
Analyzing the optimal (mp, ms) settings reveals insight why our algorithm performs better than AFTQC. That is, the optimal (mp, ms) value under our algorithm is consistently lower than that under AFTQC, given the same capture rate. The intrusion detection activity in our algorithm consumes extra energy. However the energy consumption is minimized as it operates under the identified best setting in terms of the detection interval (T_{IDS}) and the number of voters (m). Moreover, because bad nodes are removed effectively, the system does not need to use excessive redundancy in terms of (mp, ms) to cope with bad nodes performing packet dropping attacks. Consequently, the MTTF obtainable by our algorithm is substantially higher than that of AFTQC.

4.7 Sensitivity Analysis

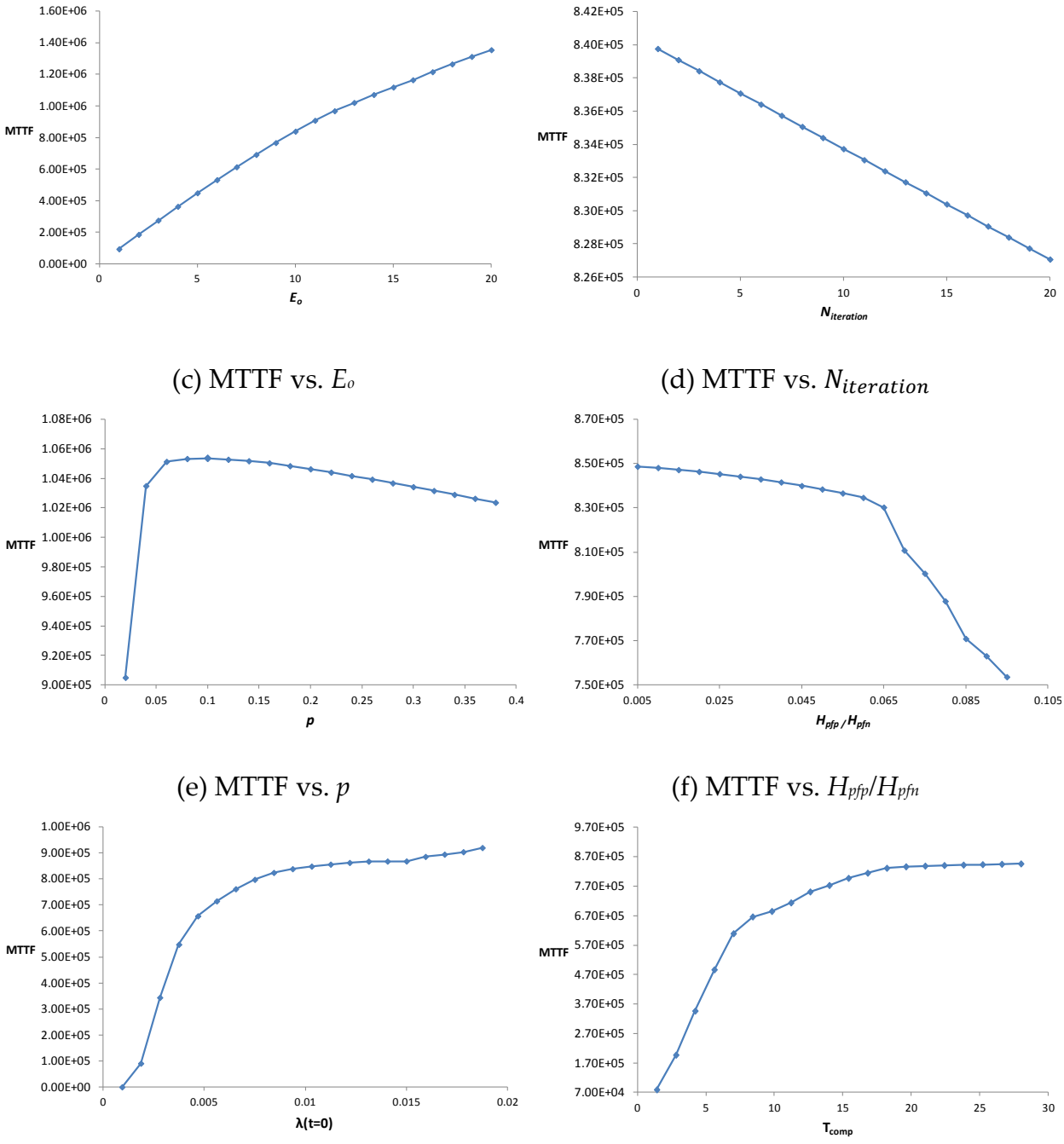
In this section we perform sensitivity analysis of MTTF obtained with respect to a selective set of input parameters, namely T_{req}, r, E_o , $N_{iteration}$, p, H_{pfp}/H_{pfn}, λ, and T_{comp}. Figure 4-12 (a)-(h) show unidimensional sensitivity analysis results by varying one parameter while keeping all the others fixed (as in Table 4.2).



(a) MTTF vs. T_{req}



(b) MTTF vs. r



(c) MTTF vs. E_o

(d) MTTF vs. $N_{iteration}$

(e) MTTF vs. p

(f) MTTF vs. H_{pfp}/H_{pfn}

(g) MTTF vs. λ

(h) MTTF vs. T_{comp}

Figure 4-12: Sensitivity Analysis of Input Parameters.

Figure 4-12 (a) shows the effect of T_{req} on MTTF. We see that increasing T_{req} relaxes the timeliness requirement of the queries which in turn increases the query reliability and thus increases MTTF.

The effect of the fixed radio range parameter r is shown in Figure 4-12 (b), where we observe that an optimal radio range exists that maximizes the system lifetime. While the radio range is considered as an input parameter in this study, it can be a design parameter. That is, one can adjust the radio range dynamically to maintain network connectivity in response to attacks [5, 6]. We treat radio range as a design parameter in Chapter 6 and use it as a counterattack against capture attacks to maximize MTTF.

In Figure 4-12 (c) we observe that MTTF increases linearly with the initial energy given to nodes at deployment. This is expected as increasing node energy increases the ability a SN is able to handle clustering, IDS, and query events until it reaches energy exhaustion.

Figure 4-12 (d) and (e) show the sensitivity of MTTF with respect to clustering parameters including the number of clustering iterations ($N_{iteration}$) and the probability of a node becoming a CH (p). Figure 4-12 (d) shows the effect of $N_{iteration}$ on MTTF, given all other parameters fixed at their default values as in Table 4.2. We see that MTTF in general decreases linearly as the number of iterations needed increases for cluster election. The reason is that energy consumption for clustering (Equation (4.27) is linearly proportional to $N_{iteration}$ and hence the time at which the system exhausts its energy decreases approximately linearly with $N_{iteration}$. Figure 4-12 (e) shows the effect of p on MTTF. We see that there exists an optimal value at which MTTF is maximized. This result correlates with that reported in [81]. The reason is that if the CH rotating probability is too high then clusters tend to be too small and too much energy is consumed for inter-cluster communication. Furthermore, each SN has a high probability of being a CH and will consume much energy performing CH duties. Conversely, if p is too low then clusters tend to be too large and too much energy is consumed for intra-cluster communication. As a result, an optimal p exists to best balance intra-cluster vs. inter-cluster communication to maximize the system lifetime.

The effect of the H_{pfp}/H_{pfn} is shown in Figure 4-12 (f) where we observe that increasing the host false positive or false negative probability decreases the system lifetime. The reason behind this trend is that the higher the false positive and false negative probabilities, the more likely good nodes will be falsely evicted and bad nodes will be falsely missed by the IDS. This in turn increases the bad node population and decreases the good node population, causing a lower MTTF.

Figure 4-12 (g) and (h) show MTTF increases with the increase of the node deployment density and node compromise interval, respectively. In Figure 4-12 (g), increasing the node population density (λ) increases the query reliability as the number of neighbors available for forwarding a query to the next hop increases. In Figure 4-12 (h), in-

creasing the capture inter-arrival time (T_{comp}) results in a higher MTTF, as less nodes will turn into bad nodes, and thus less badmouthing and packet dropping attacks.

In summary, MTTF is highly sensitive to r and p. This dictates a finer granularity be used for these two parameters when devising dynamic table lookup (to be discussed in Section 4.6 below). Furthermore, the capture inter-arrival time T_{comp} and node energy E_o are central to our model, thus should always be taken as input when searching for the best setting of design parameters to maximize MTTF.

4.8 Dynamic Table Lookup

In this Section we discuss our dynamic table lookup approach. Our objective is to dynamically identify the best setting of design parameters to maximize MTTF based on sensed input parameter values from the sensor environment. The best design parameter values given a set of input parameter values as input are pre-computed at static time, thus avoiding the overhead of runtime calculation. The design parameters in this study include: path redundancy (m_p), source redundancy (m_s), the number of voters (m), and the intrusion invocation interval (T_{IDS}). The input parameters in this study include: T_{req}, r, E_o , $N_{iteration}$, p, H_{pfp}/H_{pfn}, λ, and T_{comp}.

As all deployed sensors have limited memory, it is important to ensure the feasibility of storing and accessing such a lookup table. The lookup table would store key-value pairs where the keys are combinations of input parameter values, and the values are the design parameter values that maximize MTTF under the input parameter values. This is shown in Figure 4-13 below. There is a tradeoff between the size of the lookup table and the accuracy of the best set of design parameter values that maximize the system lifetime. From the sensitivity analysis performed in Section 4.7, we observe that MTTF is highly sensitive to r and p especially they are certain value areas at which MTTF is maximized. Such areas would demand the lookup table to be populated with finer granularity input values (keys).

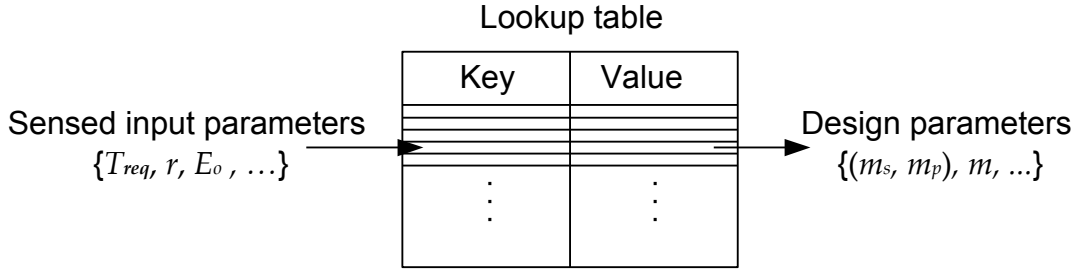


Figure 4-13: Lookup Table Mechanism.

If we consider storing 4 representative values per input variable in the lookup table, then the total number of combinations is 4^8 or 64K can be represented with 16 bits. Similarly our design parameter combinations for (m_s, m_p), m, and T_{IDS} have (7*7)*3*12 values which can be represented with 11 bits. Hence the memory storage requirement for the lookup table is 64k*11 bits, which is feasible with modern sensors in the market (e.g., CM5000 TelosB mote [2] has 1MB of Flash memory). An array, for example, can be used to implement the lookup table, by interpreting the key as an array index so that we can store the value associated with key i in array position i, which gives us O(1) lookup time. This is under the assumption that the nodes are homogeneous (as in Chapter 4), whereas in the case of heterogeneous WSNs (Chapter 5) or BS-Based WSNs (Chapter 6) the memory and processing limitation is less severe, as more capable nodes (e.g. CHs and BS) exist to hold the lookup table and be responsible to disseminate the best set of design parameters to less capable nodes in the network.

When there is less memory limitation, the number of key combinations and associated values can increase in order to increase the accuracy of the retrieved design parameter values at runtime. Finally, in case a large block of memory is scarce, one can use multiple lookup tables spread across the network, each saving distinct key-value combinations, similar to distributed hash tables (DHTs) which have been proposed for routing and data management in WSNs [59].

4.9 Summary

In this chapter we provided a solution to the issue of dynamic and adaptive multisource multipath routing for intrusion tolerance and lifetime maximization in autonomous homogeneous wireless sensor networks. We developed a novel probability model to analyze the best multisource multipath redundancy level in terms of path redundancy (m_p) and source redundancy (m_s), as well as the best intrusion detection settings in terms of the number of voters (m) and the intrusion invocation interval (T_{IDS}) under which the lifetime of a query-based wireless sensor network may be maximized in the presence of unreliable wireless communication and malicious nodes. Our dynamic multisource multipath routing algorithm utilizes the analysis result to determine the optimal system settings for redundancy and intrusion detection based on the sensed environmental conditions at runtime, thus resulting in the system achieving its maximum lifetime.

Chapter 5

Redundancy Management of Multisource Multipath Routing for Intrusion and Fault Tolerance in Heterogeneous Clustered WSNs

In this chapter we propose and analyze redundancy management of heterogeneous clustered WSNs (HWSNs), utilizing multisource multipath routing to answer user queries in the presence of unreliable and malicious nodes. The key concept of our redundancy management is to exploit the tradeoff between energy consumption vs. the gain in reliability, timeliness, and security to maximize the system useful lifetime. We formulate the tradeoff as an optimization problem for dynamically determining the best redundancy level to apply to multisource multipath routing for intrusion tolerance so that the query response success probability is maximized while prolonging the useful lifetime. Furthermore, we consider this optimization problem for the case in which a voting-based distributed intrusion detection algorithm is applied to detect and evict malicious nodes in a HWSN. We develop a novel probability model to analyze the best redundancy level in terms of path redundancy and source redundancy, as well as the best intrusion detection settings in terms of the number of voters and the intrusion invocation interval under which the lifetime of a HWSN is maximized, while satisfying the system reliability, timeliness and security requirements in the presence of unreliable wireless communication and malicious nodes. We then apply the analysis results obtained to the design of a dynamic redundancy management algorithm to identify and apply the best design parameter settings at runtime in response to environment changes, to maximize the HWSN lifetime. This chapter is based on our work published in [7] and [6].

5.1 System Model

We consider a HWSN with low-power SNs distributed in a geographic area. The WSN is characterized by the following system model:

1. There are two types of sensor nodes, CHs and SNs. CHs are superior to SNs in energy and computational resources. We use E_{init}^{CH} and $E_{init}^{SN}$to denote the initial energy levels of CHs and SNs, respectively.
2. The deployment area of the WSN is of size A^2.
3. CHs and SNs are deployed randomly and distributed according to homogeneous spatial Poisson processes with intensities λ_{CH} and λ_{SN}, respectively, with $\lambda_{CH} < \lambda_{SN}$. The radio ranges used by CH and SN transmission is denoted by r_{CH} and r_{SN}, respectively. The radio range and the transmission power of both CHs and SNs are dynamically adjusted throughout the system lifetime to maintain the connectivity between CHs and between SNs. Any communication between two nodes with a distance greater than single hop radio range between them would require a multihop. Due to limited energy, a packet is sent hop by hop without using acknowledgment or retransmission [58]
4. All sensors are subject to capture attacks, i.e., they are vulnerable to physical capture by the adversary after which their code is compromised and they become *inside* attackers. Since all sensors are randomly located in the operational area, the same capture rate applies to both CHs and SNs, and the compromised nodes are also randomly distributed in the operation area. Due to limited resources, we assume that when a node is compromised, it only performs two most energy conserving attacks, namely, *bad-mouthing attacks* (recommending a good node as a bad node and a bad node as a good node) when serving as a recommender, and *packet dropping attacks* [81] when performing packet routing to disrupt the operation of the network..
5. Environment conditions which could cause a node to fail with a certain probability include hardware failure (q), and transmission failure due to noise and interference (e). Moreover, the hostility to the WSN is characterized by a per-node capture rate of λ_c which can be determined based on historical data and knowledge about the target application environment. These probabilities are assumed to be constant and known at deployment time.
6. Queries can be issued by a mobile user (while moving) and can be issued anywhere in the WSN through a nearby CH. A CH which takes a query to process is called a query processing center (PC). Each query has a strict timeliness requirement (T_{req}). The query must be delivered within T_{req} seconds; otherwise, the query fails.
7. Redundancy management of multipath routing for intrusion tolerance is achieved through two forms of redundancy: (a) source redundancy by which m_s SNs sensing a physical phenomenon in the same feature zone are used to forward sensing

data to their CH (referred to as the source CH); (b) path redundancy by which m_p paths are used to relay packets from the source CH to the PC through intermediate CHs. Figure 5-1 shows a scenario with a source redundancy of 3 (m_s = 3) and a path redundancy of 2 (m_p = 2). It has been reported that the number of edge-disjoint paths between nodes is equal to the average node degree with a very high probability [51]. Therefore, when the density is sufficiently high such that the average number of one-hop neighbors is sufficiently larger than m_p and m_s, we can effectively result in m_p redundant paths for path redundancy and m_s distinct paths from m_s sensors for source redundancy.

8. Geographic forwarding is used to route the information between nodes; thus, no path information is maintained. The location of the destination node needs to be known to correctly forward a packet. As part of clustering, a CH knows the locations of SNs within its cluster, and vice versa. A CH also knows the location of neighbor CHs along the direction towards the processing center.
9. We assume that sensors operate in power saving mode (e.g. [22, 107]). Thus, a sensor is either active (transmitting or receiving) or in sleep mode. For the transmission and reception energy consumption of sensors, we adopt the energy model in [126] for both CHs and SNs.

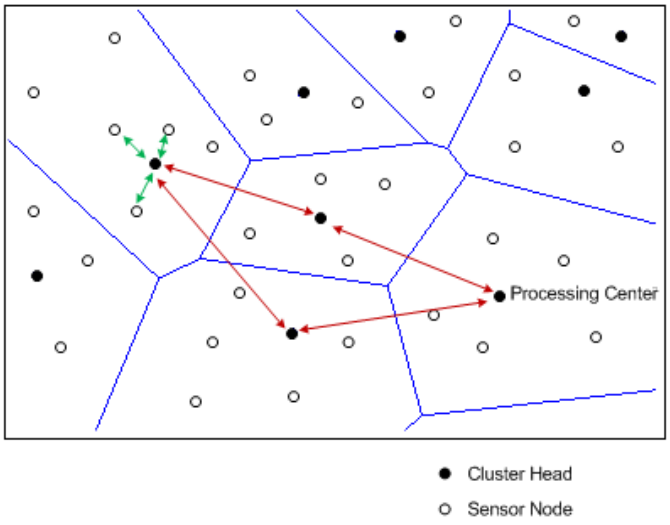


Figure 5-1: Source and Path redundancy for a Heterogeneous WSN

10. We assume that the WSN executes a pairwise key establishment protocol (e.g., [65, 130]) in a secure interval after deployment. Each node establishes pairwise keys with its k-hop neighbors, where k is large enough to cover a cluster area. Thus,

when SNs join a new cluster, the CH node will have pairwise keys with the SNs joining its cluster. Since every SN shares a pairwise key with its CH, a SN can encrypt data sent to the CH for confidentiality and authentication purposes. Every CH also creates a pairwise key with every other CH. Thus a pairwise key exists for secure communication between CHs. This mechanism is useful to prevent outside attackers, not inside attackers.

11. To detect compromised nodes, every node runs a simple *host IDS* to assess its neighbors. Our host IDS is light-weight to conserve energy. It is also generic and does not rely on the feedback mechanism tied in with a specific routing protocol (e.g., MDMP for WSNS [86] or AODV for MANETs [112]). It is based on local monitoring. That is, each node monitors its neighbor nodes only. Each node uses a set of anomaly detection rules such as a high discrepancy in the sensor reading or recommendation has been experienced, a packet is not forwarded as requested, as well as interval, retransmission, repetition, and delay rules as in [14, 19, 49, 108]. If the count exceeds a system-defined threshold, a neighbor node that is being monitored is considered compromised. The imperfection of monitoring due to environment noise or channel error is modeled by a "host" false positive probability (H_{pfp}) and a "host" false negative probability (H_{pfn}) which are assumed known at deployment time.
12. To remove malicious nodes from the system, a voting-based distributed IDS is applied periodically in every T_{IDS} time interval. A CH is being assessed by its neighbor CHs, and a SN is being assessed by its neighbor SNs. In each interval, m neighbor nodes (at the CH or SN level) around a target node will be chosen randomly as voters and each cast their votes based on their host IDS results to collectively decide if the target node is still a good node. The m voters share their votes through secure transmission using their pairwise keys. When the majority of voters come to the conclusion that a target node is bad, then the target node is evicted. For both CHs and SNs, there is a system-level false positive probability P_{fp}that the voters can incorrectly identify a good node as a bad node. There is also a system-level false negative probability P_{fn}that the voters can incorrectly misidentify a bad node as a good node. These two system-level IDS probabilities will be derived based on the *bad-mouthing* attack model in this chapter.

Here we note that increasing source or path redundancy enhances reliability and security. However, it also increases the energy consumption, thus contributing to the decrease of the system lifetime. Thus, there is a tradeoff between reliability/security gain vs. energy consumption. The distributed IDS design attempts to detect and evict compromised nodes from the network without unnecessarily wasting energy so as to maximize the query success probability and the system lifetime. The effectiveness of the IDS depends on its parameters (T_{IDS} and m). While a shorter T_{IDS} or a higher m can result in

low P_{fp} and P_{fn}, it also consumes more energy from the WSN nodes. Thus, this is another design tradeoff.

To provide a unifying metric that considers the above two design tradeoffs, we define the total number of queries the system can answer correctly until it fails as the *lifetime* or the *mean time to failure* (MTTF) of the system, which can be translated into the actual system lifetime span given the query arrival rate. A failure occurs when no response is received before the query deadline. The cause could be due to energy exhaustion, packet dropping by malicious nodes, channel/node failure, or insufficient transmission speed to meet the timeliness requirement. Our aim is to find both the optimal redundancy levels and IDS settings under which the MTTF is maximized, when given a set of parameters characterizing the operational and environment conditions.

5.2 Probability Model

Table 5-1: Parameter List.

Parameter	Meaning	Type
A	Length of each side of a square sensor area (meter)	input
n_b	Size of a data packet (bit)	input
E_{elec}	Energy dissipation to run the transmitter and receiver circuitry (J/bit)	input
E_{amp}	Energy used by the transmit amplifier to achieve an acceptable signal to	input
E_o	Initial energy per node (Joule)	input
E_{init}	Initial energy of the WSN (Joule)	derived
$E_{clustering}(t)$	Energy consumed for executing the clustering algorithm at time t (Joule)	derived
$E_{IDS}(t)$	Energy consumed for executing the IDS algorithm at time t (Joule)	input
$E_q(t)$	Energy consumed for executing a query at time t (Joule)	derived
$R_q(t)$	Probability that a query reply at time t is delivered successfully by the	derived
r	Wireless radio communication range (meter)	input
q	node hardware failure probability	input
e_j	Transmission failure probability of node j	input
$N(t)$	Number of nodes in the WSN at time t	input
$N_{CH}(t)$	Number of CHs in the WSN at time t	derived
$N_{SN}(t)$	Number of SNs in the WSN at time t	derived

$n(t)$	Number of neighbor nodes at time t	derived
$n_{good}(t)$	Number of good neighbor nodes at time t	derived
$n_{bad}(t)$	Number of bad neighbor nodes at time t	derived
N_q	Maximum number of queries before energy exhaustion	derived
m_p	Path redundancy level: Number of paths from a source CH to the sink	design
m_s	Source redundancy level: Number of SNs per cluster in response to a	design
f	Fraction of neighbor nodes that will forward data	input
$\lambda(t)$	Node population density (nodes/meter^2) at time t	derived
λ	Node population density at deployment time	input
λ_q	Query arrival rate (times/sec)	input
S_{jk}	Progressive transmission speed between node j and node k (meter/sec)	derived
$T_{clustering}$	Time interval for executing the clustering algorithm (sec)	input
T_{req}	Query deadline requirement (sec)	input
λ_c	Node capture rate	input
α	Ratio of IDS execution rate to query arrival rate	input
β	Ratio of clustering rate to query arrival rate	input
m	The query-based clustered WSN is characterized by a set of input pa-	design
H_{pfp}	Probability of host IDS false positive	input
H_{pfn}	Probability of host IDS false negative	input
Pfp	Probability of distributed IDS false positive	derived
Pfn	Probability of distributed IDS false negative	derived
T_{IDS}	IDS interval time (sec)	design
MTTF	Lifetime of a heterogeneous WSN	output

In this section we develop a probability model to estimate the MTTF of a HWSN using multipath data forwarding to answer queries issued from a mobile user roaming in the WSN area. Table 5-1 provides the parameter list along with the physical meaning. We use the same notation for both CHs and SNs, e.g., P_{fp} and P_{fn}. When differentiating a CH from a SN is necessary, we use the superscripts or subscripts "CH" and "SN", e.g., P_{fp}^{CH} and P_{fn}^{CH} for a CH, and P_{fp}^{SN} and P_{fn}^{SN} for a SN. A parameter is labeled as *input*, *derived*,

design or *output*. In particular, m_p (path redundancy), m_s (source redundancy), m (the number of voters for intrusion detection) and T_{IDS} (the intrusion detection interval) are design parameters whose values are to be identified to maximize MTTF, when given a set of input parameter values charactering the operational and environmental conditions. Derived parameters are those deriving from input parameters. There is only one output parameter, namely, MTTF. Note that most derived parameters are dynamic, i.e., as a function of time. For example, node density denoted by $\lambda(t)$ decreases over time because of node failure/eviction as time progresses. On the other hand, the radio ranges for CHs and SNs, denoted by r_{CH} and r_{SN}, increase over time to maintain network connectivity.

The basic idea of our MTTF formulation is that we first deduce the maximum number of queries, N_q, the system can possible handle before running into energy exhaustion for the best case in which all queries are processed successfully. Because the system evolves dynamically, the amount of energy spent per query also varies dynamically. Given the query arrival rate λ_q as input, the average interval between query arrivals is $1/\lambda_q$. So we can reasonably estimate the amount of energy spent due to query processing and intrusion detection for query j based on the query arrival time $t_{Q,j}$. Next we derive the corresponding query success probability $R_q(t_{Q,j})$, that is, the probability that the response to query j arriving at time $t_{Q,j}$ is delivered successfully to the PC before the query deadline expires. Finally, we compute MTTF as the probability-weighted average of the number of queries the system can handle without experiencing any deadline, transmission, or security failure. More specifically, the MTTF is computed by:

$$MTTF = \sum_{i=1}^{N_q-1} i\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)\left(1 - R_q(t_{Q,i+1})\right) + N_q \prod_{j=1}^{N_q} R_q(t_{Q,j}) \qquad (5.1)$$

Here $\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)\left(1 - R_q(t_{Q,i+1})\right)$ accounts for the probability of the system being able to successfully execute i consecutive queries but failing the $i+1^{th}$ query. The second term is for the best case in which all queries are processed successfully without experiencing any failure for which the system will have the longest lifetime span.

5.2.1 Network Dynamics

Initially at deployment all SNs are good nodes. Assume that the capture time of a SN follows a distribution function $F_c(t)$ which can be determined based on historical data and knowledge about the target application environment. Then, the probability that a

SN is compromised at time t, given that it was a good node at time t-T_{IDS}, denoted by P_c, is given by:

$$\begin{aligned} P_C &= 1 - P\{X > t \mid X > t - T_{IDS}\} \\ &= 1 - \frac{P\{X > t, X > t - T_{IDS}\}}{P\{X > t - T_{IDS}\}} = 1 - \frac{1 - F_c(t)}{1 - F_c(t - T_{IDS})} \end{aligned} \tag{5.2}$$

We note that P_c is time dependent. For the special case in which the capture time is exponential distributed with rate λ_c, $P_c = 1 - e^{-\lambda_c \times T_{IDS}}$. Recall that the voting-based distributed IDS executes periodically with T_{IDS} being the interval. At the i^{th} IDS execution time (denoted by $t_{I,i}$), a good node may have been compromised with ity P_c since the previous IDS execution time $(t_{I,i-1})$. Let $n_{good}(t)$and $n_{bad}(t)$ denote the numbers of good and bad neighbor nodes at time t, respectively, with $n_{good}(t) + n_{bad}(t) = n(t)$. Then, the population of good and bad neighbor nodes at time $t_{I,i}$ just prior to IDS execution can be recursively estimated from the population of good and bad neighbor nodes at time $t_{I,i-1}$:

$$\begin{aligned} n_{good}(t_{I,i}) &= n_{good}(t_{I,i-1}) - n_{good}(t_{I,i-1}) \times P_c \\ n_{bad}(t_{I,i}) &= n_{bad}(t_{I,i-1}) + n_{good}(t_{I,i-1}) \times P_c \end{aligned} \tag{5.3}$$

With $n_{good}(t)$ and $n_{bad}(t)$ in hand, the system-level false positive probability (P_{fp}) and false negative probability (P_{fn}) as a resulting of executing voting-based IDS are as follows:

(5.4)

$$P_{fp} \text{ or } P_{fn} \equiv \sum_{i=0}^{m-m_{maj}} \left[\frac{C\binom{n_{bad}}{m_{maj}+i} \times C\binom{n_{good}}{m-(m_{maj}+i)}}{C\binom{n_{bad}+n_{good}}{m}} \right] + \sum_{i=0}^{m-m_{maj}} \left[\frac{C\binom{n_{bad}}{i} \times \sum_{j=m_{maj}-i}^{m-i} \left[C\binom{n_{good}}{j} \times \omega^j \times C\binom{n_{good-j}}{m-i-j} \times (1-\omega)^{m-i-j} \right]}{C\binom{n_{bad}+n_{good}}{m}} \right]$$

where m_{maj} is the minimum majority of m, e.g., 3 is the minimum majority of 5, and ω is H_{pfp} for calculating P_{fp} and H_{pfn} for calculating P_{fn}. We explain Equation (5.4) for the false positive probability at time t below. The explanation to the false negative probability is similar. A false positive results when the majority of the voters vote against the target node (which is a good node) as compromised. The first term in Equation (5.4) accounts for the case in which more than 1/2 of the voters selected from the target node's neighbors are bad sensors who, as a result of performing bad-mouthing attacks, will always vote a good node as a bad node to break the functionality of the WSN. Since more than 1/2 of the m voters vote no, the target node (which is a good node) is diagnosed as a bad node in this case, resulting in a false positive. Here the denominator is the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select at least m_{maj} bad voters out of n_{bad} nodes and the remaining good voters out of n_{good} nodes. The second term accounts for the case in which more than 1/2 of the voters selected from the neighbors are good nodes but unfortunately some of these good nodes mistakenly misidentify the target nodes as a bad node with host false positive probability H_{pfp}, resulting in more than 1/2 of the voters (although some of those are good nodes) voting no against the target node. Since more than 1/2 of the m voters vote no, the target node (which is a good node) is also diagnosed as a bad node in this case, again resulting in a false positive. Here the denominator is again the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select i bad voters not exceeding the majority m_{maj}, j good voters who diagnose incorrectly with $i + j \geq m_{maj}$, and the remaining $m - i - j$ good voters who diagnose correctly. We apply intrusion detection to both CHs and SNs, i.e., m CH voters would be used to asses a target CH and m SN voters will be used to assess a target SN node. Here we note that more voters do not necessarily provide better detection accuracy since it depends on the percentage of bad node population. That is, if more bad nodes exist than good nodes in the neighborhood, or good nodes have high host false positive probability (H_{pfp}) and host false negative probability (H_{pfn}), then more voters will provide less detection accuracy.

After the voting-based IDS is executed, some good nodes will be misidentified as bad nodes with probability P_{fp} and will be mistakenly removed from the WSN. Consequently, we need to adjust the population of good nodes after IDS execution. Let $\overline{n_{good}(t)}$ be the number of good neighbor nodes at time t right after IDS execution. Then,

$$\overline{n_{good}(t_{I,i})} = n_{good}(t_{I,i}) - n_{good}(t_{I,i}) \times P_{fp} \tag{5.5}$$

On the other hand, some bad nodes will remain in the system because the voting-based IDS fails to identify them with probability P_{fn}. Let $\overline{n_{bad}(t)}$ be the number of bad neighbor nodes at time t right after IDS execution. Then,

$$\overline{n_{bad}(t_{I,\iota})} = n_{bad}(t_{I,i}) - n_{bad}(t_{I,i}) \times (1 - P_{fn}) \tag{5.6}$$

As the capture attack is totally random, the probability that any neighbor node is a bad node at time t, denoted by $Q_{c,j}(t)$, thus is given by:

$$Q_{c,j}(t_{I,i}) = \frac{\overline{n_{bad}(t_{I,\iota})}}{\overline{n_{bad}(t_{I,\iota})} + \overline{n_{good}(t_{I,\iota})}} \tag{5.7}$$

$Q_{c,j}(t)$ derived above provides critical information because a bad node can perform packet dropping attacks if it is on a path from source SNs to the PC. Here we note that the node population density is evolving because of some nodes being compromised and some being detected and evicted by the IDS dynamically. The node population remains the same until the next IDS execution (after T_{IDS} seconds) because the IDS only detects and evicts nodes periodically (as typically node hardware/software failure happens less frequently than security failure). The remaining nodes comprise both good nodes that pass the IDS evaluation and bad nodes that are undetected by the IDS. Denote the node population density at time t by $\lambda(t)$ with $\lambda(0)= \lambda$. Then, $\lambda(t)$ can be computed by:

$$n(t_{I,i}) = \overline{n_{bad}(t_{I,\iota})} + \overline{n_{good}(t_{I,\iota})} \tag{5.8}$$

$$\lambda(t_{I,i}) = \frac{n(t_{I,i})}{\pi r^2} \tag{5.9}$$

With Equation 5.8 we compute the neighbor node populations for CHs and SNs, denoted by $n_{CH}(t_{I,i})$ and $n_{SN}(t_{I,i})$, respectively. With Equation (5.9) we derive the CH and SN node densities at time t, $\lambda_{CH}(t_{I,i})$ and $\lambda_{SN}(t_{I,i})$, respectively. The total numbers of CHs and SNs in the system, denoted by $N_{CH}(t_{I,i})$ and $N_{SN}(t_{I,i})$, respectively, can be computed by:

$$N_{CH}(t_{I,i}) = \lambda_{CH}(t_{I,i}) \times A^2 \tag{5.10}$$

$$N_{SN}(t_{I,i}) = \lambda_{SN}(t_{I,i}) \times A^2 \tag{5.11}$$

We define the required network connectivity as the required redundancy level for effecting multipath routing defined by (m_p, m_s). Thus, the radio range at time $t_{I,i}$ is adjusted by:

$$r_{CH}(t_{I,i}) = \sqrt{\frac{m_p}{\pi f \lambda_{CH}(t_{I,i})}} \; ; \; r_{SN}(t_{I,i}) = \sqrt{\frac{m_s}{\pi f \lambda_{SN}(t_{I,i})}} \tag{5.12}$$

5.2.2 Query Success Probability

We will use the notation SN_j to refer to SN j and CH_j to refer to CH j. There are three ways by which data forwarding from CH_j to CH_k could fail: (a) transmission speed violation; (b) sensor/channel failures; and (c) CH_j is compromised. The first source of failure, transmission speed violation, accounts for query deadline violation. To know the failure probability due to transmission speed violation, we first derive the minimum hop-by-hop transmission speed required to satisfy the query deadline T_{req}. Let $d_{SN\text{-}CH}$ be the *expected* distance between a SN (selected to report sensor readings) and its CH and $d_{CH\text{-}PC}$ be the *expected* distance between the source CH and the PC accepting the query result. Given a query deadline T_{req} as input, a data packet from a SN through its CH to the PC must reach the PC within T_{req}. Thus, the minimum hop-by-hop transmission speed denoted by S_{req} is given by:

$$S_{req} = \frac{d_{SN-CH} + d_{CH-PC}}{T_{req}} \tag{5.13}$$

We follow the Voronoi theory [12] that during clustering SNs will join the closest CH to form a Voronoi cell and that the expected distance from a SN to its CH is given by $d_{SN-CH} = 1/2(\lambda_{CH})^{1/2}$. On the other hand, since a query may be issued from anywhere by the mobile user to a CH (which serves as the PC) and the source CH requested by the query also can be anywhere in the WSN, $d_{CH\text{-}PC}$ essentially is the average distance between any two CHs in the WSN. Given location randomness of CHs in the square area A^2, it can be shown geometrically that the average distance between any two CHs is $d_{CH\text{-}PC}$=0.382A. With the knowledge of $d_{SN\text{-}CH}$ and $d_{CH\text{-}PC}$, we can estimate the average numbers of hops to forward data from a SN to the source CH, denoted by N_{SC}^h, and the average numbers of hops to forward data from the source CH to the PC, denoted by N_{CP}^h. Specifically,

$$N_{SC}^{h} = d_{SN-CH}/r_{SN}; N_{cp}^{h} = d_{CH-PC}/r_{CH.} \quad (5.14)$$

Let $Q_{t,jk}$ denote the probability that the forwarding speed from CH_j to CH_k would violate the minimum speed requirement, thus leading to a query deadline violation failure. To calculate $Q_{t,jk}$ we need to know the transmission speed S_{jk} from CH_j to CH_k. This can be dynamically measured by CH_j following the approach described in [58]. If S_{jk} is above S_{req} then $Q_{t,jk} = 0$; otherwise, $Q_{t,jk} = 1$. In general S_{jk} is not known until runtime. If S_{jk} is uniformly distributed within a range $[a, b]$, then $Q_{t,jk}$ can be computed as:

$$Q_{t,jk}^{SN}(t) = cdf(S_{jk} \le S_{req}) = \frac{S_{req} - a}{b - a} \quad (5.15)$$

The second source of failure is due to node failure or channel failure. Let $Q_{r,j}$ denote the probability of failure due to node failure or channel failure. Since q is the hardware failure probability, given as input, and e_j is the per-hop transmission failure probability of node j, measured by node j at runtime, $Q_{r,j}$ can be estimated by:

$$Q_{r,j}^{CH} = 1 - [(1-q)(1-e_j)] \quad (5.16)$$

The third source of failure is due to node j being compromised and thus the packet is dropped. We make use of $Q_{c,j}^{CH}(t)$ derived earlier in Equation (5.7). By combining these three failure probabilities we obtain $Q_{rtc,jk}^{CH}(t)$, the probability of CH_j failing to relay a data packet to a one-hop neighbor CH_k because of either speed violation, sensor/channel failure, or CH_j being compromised, as:

$$Q_{rtc,jk}^{CH}(t) = 1 - \left[\left(1 - Q_{r,j}^{CH}\right)\left(1 - Q_{t,jk}^{CH}(t)\right)\left(1 - Q_{c,j}^{CH}(t)\right)\right] \quad (5.17)$$

By using this one-hop failure probability, we next compute the success probability for CH_j to transmit a packet to at least one next-hop CH neighbor along the direction of the destination node as:

$$\theta_j^{CH} = 1 - \prod_{k=1}^{f*n_{CH}} Q_{rtc,jk}^{CH}(t) \quad (5.18)$$

with f=1/4 to account for the fact that only neighbor CHs in the quadrant toward the destination node can perform geographic forwarding; n_{CH} is the number of neighbor CHs of CH_j as derived from Equation (5.8).

Since on average there will be N_{CP}^h hops on a path from the source CH to the PC, a data packet transmitted along the path is successfully delivered only if it is delivered successful hop-by-hop without experiencing any speed violation failure, hardware/channel failure, or packet dropping failure, for N_{CP}^h hops. Consequently, the probability of a single path between the source CH and the PC being able to deliver data successfully is given by:

$$\Theta(N_{CP}^h) = (\prod_{j=1}^{N_{CP}^h-1} \theta_j^{CH}) \times (1 - Q_{rtc,\ N_{CP}^h(N_{CP}^h+1)}^{CH}) \times [(1-q)(1 - Q_{c,PC}^{CH})] \tag{5.19}$$

For redundancy management, we create m_p paths between the source CH and the PC for *path redundancy*. The m_p paths are formed by choosing m_p CHs in the first hop and then choosing only one CH in each of the subsequent hops. The source CH will fail to deliver data to the PC if one of the following happens: (a) none of the CHs in the first hop receives the message; (b) in the first hop, i ($1 \le i < m_p$) CHs receive the message, and each of them attempts to form a path for data delivery; however, all i paths fail to deliver the message because the subsequent hops fail to receive the broadcast message; or (c) in the first hop, at least m_p CHs receive the message from the source CH from which m_p CHs are randomly selected to forward data, but all m_p paths fail to deliver the message because the subsequent hops fail to receive the message. Summarizing above, the probability of the source CH failing to deliver data to the PC is given by:

$$
\begin{aligned}
Q_{fp}^{m_p} &= 1-\theta_1^{CH} + \\
&\sum_{|I|<m_p}\left[\begin{array}{l}\{[\prod_{i\in I}(1-Q_{rtc,1i}^{CH}\ )]\times(\prod_{i\notin I}Q_{rtc,1i}^{CH})\} \\ \qquad\times\{\prod_{i\in I}[1-\Theta_i(N_{CP}^h-1)]\}\end{array}\right]+ \\
&\sum_{|I|\geq m_p}\left[\begin{array}{l}\{[\prod_{i\in I}(1-Q_{rtc,1i}^{CH}\ )]\times(\prod_{i\notin I}Q_{rtc,1i}^{CH})\} \\ \qquad\times\{\prod_{\substack{i\in M,\\ M\subseteq I,\\ |M|=m_p}}[1-\Theta_i(N_{CP}^h-1)]\}\end{array}\right]
\end{aligned}
\tag{5.20}
$$

Following the same derivation to Equation (5.19), the success probability of a single path from a SN to its CH is given by:

$$
\Theta(N_{SC}^h) = (\prod_{j=1}^{N_{SC}^h-1}\theta_j^{SN})\times(1-Q_{rt,\ N_{SC}^h(N_{SC}^h+1)}^{SN}) \tag{5.21}
$$

We use m_s SNs to report query responses to their source CH for *source redundancy*. The probability that all m_s SNs fail to deliver data to their CH is thus given by:

$$
Q_{fs}^{m_s} = \prod_{i=1}^{m_s}[1-\Theta_i(N_{SC}^h)] \tag{5.22}
$$

Consequently, the failure probability of data delivery from m_s SNs to the CH, and subsequently using m_p paths to relay data from source CH to PC, is given by:

$$
Q_f = 1-(1-Q_{fp}^{m_p})(1-Q_{fs}^{m_s}) \tag{5.23}
$$

Therefore, the query success probability is given by:

$$
R_q = 1-Q_f \tag{5.24}
$$

Note that in the above derivation we omit time for brevity. More precisely, R_q derived above should be $R_q(t_{Q,i})$ since the query success probability is a function of time,

depending on the node count (Equation (5.8)) and population density (Equation (5.9)) at the i^{th} query's execution time (i.e., at time $t_{Q,i}$).

5.2.3 Energy Consumption

Now we estimate the amounts of energy spent during a query interval $[t_{Q,i}, t_{Q,i+1}]$, an IDS interval $[t_{I,i}, t_{I,i+1}]$, and a clustering interval $[t_{C,i}, t_{C,i+1}]$, so as to estimate N_q, the maximum number of queries the system can possible handle before running into energy exhaustion. To normalize energy consumption over N_q queries, let α be the ratio of the IDS execution rate to the query arrival rate and let β be the ratio of the clustering rate to the query arrival rate so that αN_q and βN_q are the numbers of IDS cycles and clustering cycles, respectively, before system energy exhaustion. Then, we can estimate N_q by the fact that the total energy consumed due to intrusion detection, clustering and query processing is equal to the system energy as follows:

$$E_{init} = \sum_{i=1}^{\alpha N_q} E_{IDS}(t_{I,i}) + \sum_{i=1}^{\beta N_q} E_{clustering}(t_{C,i}) + \sum_{i=1}^{N_q} E_q(t_{Q,i}) \tag{5.25}$$

Below we outline how to calculate $E_{IDS}(t_{I,i})$, $E_{clustering}(t_{C,i})$ and $E_q(t_{Q,i})$. We first estimate energy consumed by transmission and reception over wireless link. The energy spent by a SN to transmit an encrypted data packet of length n_b bits over a distance r is estimated as [126]:

$$E_t = n_b(E_{elec} + E_{amp} r^x) \tag{5.26}$$

Here E_{elec} is the energy dissipated to run the transmitter and receiver circuitry, E_{amp} is the energy used by the transmit amplifier, and r is the transmission radio range. We use the current r_{CH} and r_{SN} to derive E_T^{CH} and E_T^{SN}. We set E_{amp}= 10 pJ/bit/m^2 and $x = 2$ when $d < d_0$, and E_{amp}= 0.0013 pJ/bit/m^4 and $x = 4$ otherwise. The energy spent by a node to receive an encrypted message of length n_b bits is given by:

$$E_R = n_b E_{elec} \tag{5.27}$$

The energy consumed for processing the i^{th} query, $E_q(t_{Q,i})$, is the sum of the energy consumed through m_p paths for the communication between CH and PC, denoted by $E_q^{CH}(t_{Q,i})$, and the energy consumed for the communication between m_s source SNs and the source CH, denoted by $E_q^{SN}(t_{Q,i})$, i.e.,

$$E_q(t_{Q,i}) = E_q^{CH}(t_{Q,i}) + E_q^{SN}(t_{Q,i}) \tag{5.28}$$

The energy consumed for the communication between CH and PC is due to setting up m_p paths in the first hop and subsequently transmitting data over the m_p paths, i.e.,

$$\begin{aligned} E_q^{CH}(t_{Q,i}) = &\left\{ E_T^{CH} + n_{CH}(t_{Q,i}) E_R^{CH} \right\} + \\ &\left\{ m_p (N_{CP}^h - 1) \left[E_T^{CH} + n_{CH}(t_{Q,i}) E_R^{CH} \right] \right\} \end{aligned} \tag{5.29}$$

Here the first term accounts for the transmission energy consumed by the source CH and the reception energy consumed by its 1-hop CHs for setting up the m_p paths, and the second term accounts for the energy consumed for data transmission over the m_p paths in the remaining$N_{CP}^h - 1$ hops. We note that the number of neighbor CHs at time t, $n_{CH}(t) = \lambda_{CH}(t) \times \pi r_{CH}^2$ by Equation (5.9), depends on the CH population density at time t, i.e., $\lambda_{CH}(t)$.

The energy consumed for the communication between source SNs and the CH is due to transmitting data over the m_s paths each with N_{SC}^h hops, i.e.,

$$E_q^{SN}(t_{Q,i}) = m_s N_{SC}^h \left[E_T^{SN} + n_{SN}(t_{Q,i}) E_R^{SN} \right] \tag{5.30}$$

For clustering, the system would consume energy for broadcasting the announcement message and for the cluster-join process. There will be $N_{CH}(t_{c,i})$ CHs each broadcasting the announcement message at time $t_{c,i}$, for a total energy consumption of $N_{CH}(t_{c,i}) \times E_T^{CH}$. There will be $N_{SN}(t_{c,i})$ SNs each on average receiving $n_{CH}(t_{c,i})$ announcement messages from CHs within radio broadcast range $r_{CH}(t_{l,i})$ at time $t_{c,i}$, for a total energy consumption of $N_{SN}(t_{c,i}) \times n_{CH}(t_{c,i}) \times E_R^{SN}$. The cluster-join process will require a SN to send a message to the CH through multi-hop routing informing that it will join the cluster. There will be $N_{SN}(t_{c,i})$ SNs each transmitting a join packet to the CH it selects to join, for a total energy consumption of $N_{SN}(t_{c,i}) \times E_R^{CH}$ for packet reception by the selected CHs plus $N_{SN}(t_{c,i}) \times N_{SC}^h(E_T^{SN} + n_{SN}(t_{c,i}) E_R^{SN})$ for multi-hop routing to the selected CHs. Summarizing above, the energy consumed for executing the clustering algorithm by CHs and SNs at time $t_{c,i}$, denoted by $E_{clustering}(t_{C,i})$, is given by:

$$\begin{aligned} E_{clustering}(t_{C,i}) = &N_{CH}(t_{c,i}) \times E_T^{CH} + N_{SN}(t_{c,i}) \times n_{CH}(t_{c,i}) \times E_R^{SN} \\ &+ N_{SN}(t_{c,i}) \times E_R^{CH} + N_{SN}(t_{c,i}) \times N_{SC}^h(E_T^{SN} + n_{SN}(t_{c,i}) E_R^{SN}) \end{aligned} \tag{5.31}$$

Lastly, for intrusion detection every node is evaluated by m voters in an IDS cycle, and each voter sends its vote to the other m - 1 voters. Hence, the energy spent in each voting-based IDS cycle, denoted by $E_{IDS}(t_{I,i})$, is given by:

$$\begin{aligned} E_{IDS}(t_{I,i}) &= E_{IDS}^{CH}(t_{I,i}) + E_{IDS}^{SN}(t_{I,i}) \\ E_{IDS}^{CH}(t_{I,i}) &= N_{CH}(t_{I,i-1})[m(m-1)]\left[E_T^{CH} + n_{CH}(t_{I,i-1})E_R^{CH}\right] \\ E_{IDS}^{SN}(t_{I,i}) &= N_{SN}(t_{I,i-1})[m(m-1)]\left[E_T^{SN} + n_{SN}(t_{I,i-1})E_R^{SN}\right] \end{aligned} \quad (5.32)$$

After we obtain $E_{IDS}(t_{I,i})$, $E_{clustering}(t_{C,i})$ and $E_q(t_{Q,i})$ from Equations (5.32), (5.31) and (5.28), respectively, we calculate N_q from Equation 5.25. The knowledge of N_q along with $R_q(t_{Q,i})$ in Equation (5.24) allows us to calculate the system MTTF given by Equation (5.1).

5.3 Algorithm for Dynamic Multisource Multipath Routing

The objective of dynamic redundancy management is to dynamically identify and apply the best redundancy level in terms of path redundancy (m_p) and source redundancy (m_s), as well as the best intrusion detection settings in terms of the number of voters (m) and the intrusion invocation interval (T_{IDS}) to maximize MTTF, in response to environment changes to input parameters including SN/CH node density ($\lambda_{SN}/\lambda_{CH}$), SN/CH radio range ($r_{SN}/r_{SN}$), and SN/CH capture rate (λ_c).

Our algorithm for dynamic redundancy management of multipath routing is distributed in nature. Figure 5-2 describes our dynamic redundancy management algorithm for managing multipath routing for intrusion tolerance to maximize the system lifetime. The algorithm specifies control actions taken by individual SNs and CHs in response to dynamically changing environments.

1: ***CH Execution***:
2: *Get next event*
3: ***if*** *event is T_D timer* ***then***
4: *determine radio range to maintain CH connectivity*
5: *determine optimal T_{IDS}, m, m_s, m_p by table lookup based on the current estimated density, CH radio range and compromise rate*
6: *notify SNs within the cluster of the new optimal settings of T_{IDS} and m*
7: ***else if*** *event is query arrival* ***then***
8: *trigger multipath routing using m_s and m_p*
9: ***else if*** *event is $T_{clustering}$ timer* ***then***
10: *perform clustering*
11: ***else if*** *event is T_{IDS} timer* ***then***
12: ***For*** *each neighbor CH*
13: ***if*** *selected as a voter* ***then***

```
14:              execute voting based intrusion detection
15:      else // event is data packet arrival
16:          follow multipath routing protocol design to route the data packet
17:
18:      SN Execution:
19:      Get next event
20:      if event is T_D timer then
21:          determine radio range to maintain SN connectivity within a cluster
22:      else if event is control packet arrival from CH then
23:          Change the optimal settings of T_IDS, and m
24:      else if event is T_clustering timer then
25:          perform clustering
26:      else if event is T_IDS timer then
27:          For each neighbor SN
28:              if selected as a voter then
29:                  execute votingbased intrusion detection
30:      else // event is data packet arrival
31:          follow multipath routing protocol design to route the data packet
```

Figure 5-2: Algorithm for Dynamic Redundancy Management.

All nodes in the system act periodically to a "T_D timer" event to adjust the optimal parameter setting in response to changing environments. This is indicated on line 3 for a CH and line 20 for a SN. The optimal design settings in terms of optimal $T_{IDS}, m, m_s,$ and m_p are determined at static design time (described in Section 5.3) and pre-stored in a table over perceivable ranges of input parameter values. As there is no base station in the system, the duty of performing a table lookup operation with interpolation and/or extrapolation techniques applied to determine the optimal design parameter settings will be assumed by CHs. The action performed by a CH upon a T_D timer event includes (a) adjusting radio range to maintain connectivity (line 4); (b) determining $T_{IDS}, m, m_s,$ and m_p (line 5) based on the sensed environmental conditions at runtime; and (c) notifying SNs within the cluster of the new T_{IDS} and m settings. The action performed by a SN upon this T_D timer event is to adjust its radio range to maintain connectivity within a cluster (line 21). The action taken upon receiving the control packet from its CH is to update the new T_{IDS} and m settings (line 23) for intrusion detection. When the T_{IDS} timer event happens, each node in the system uses it current T_{IDS} and m settings to perform intrusion detection. The T_{IDS} timer event and the action taken are specified on lines 11-14 for a CH and lines 26-29 for a SN.

When a CH acting as a query processing center (PC) receives a query from a user, it triggers multisource multipath routing for intrusion tolerance using the current optimal m_s and m_p settings to prolong the system useful lifetime. This query arrival event and the action taken are specified on lines 7-8. When a data packet arrival event occurs, each node simply follows the prescribed multipath routing protocol to route the packet (lines

15-16 for a CH and lines 30-31 for a SN). Finally each node periodically performs clustering as prescribed by the cluster algorithm, i.e., when a $T_{clustering}$ timer event occurs, each node executes clustering (lines 9-10 for a CH and lines 24-25 for a SN) for periodic cluster formation.

The cost of executing the dynamic redundancy management algorithm described above, including periodic clustering, periodic intrusion detection, and query processing through multipath routing, in terms of energy consumption has been considered in Section 5.2.3 Energy Consumption. The extra messaging overhead for each CH to notify SNs within its cluster of the new T_{IDS} and m settings (line 6) can be eliminated if the CH notifies the optimal settings to its SNs at the time periodic clustering is performed.

5.4 Performance Evaluation

In this section, we present numerical data obtained as a result of applying Equation (5.1). Table 5-2 lists the set of input parameter values characterizing a clustered HWSN. Our example HWSN consists of 3000 SN nodes and 100 CH nodes, deployed in a square area of A^2 (200m×200m). Nodes are distributed in the area following a Poisson process with density λ_{SN} =30 nodes/(20×20 m^2) and λ_{CH} =1 nodes/(20×20 m^2) at deployment time. The radio ranges r_{SN} and r_{CH} are dynamically adjusted between 5m to 25m and 25m to 120m respectively to maintain network connectivity. The initial energy levels of SN and CH nodes are $E_0^{SN} = 0.8$ Joules and $E_0^{CH} = 10$ Joules so that they exhaust energy at about the same time. The energy parameters used by the radio module are adopted from [68, 126]. The energy dissipation E_{elec} to run the transmitter and receiver circuitry is 50 nJ/bit. The energy used by the transmit amplifier to achieve an acceptable signal to noise ratio (E_{amp}) is 10 pJ/bit/m^2 for transmitted distances less than the threshold distance d_0 (75m) and 0.0013 pJ/bit/m^4 for distances greater than d_0. The query arrival rate λ_q is a variable and is set to 1 query/sec to reveal points of interest. The query deadline T_{req} is strict and set to between 0.3 and 1 sec. The SN capture time is exponential distributed with rate λ_c such that $P_c = 1 - e^{-\lambda_c \times T_{IDS}}$. We test the effect of λ_c by varying the inter-arrival time in between attacks (T_{comp}) from 4 to 28 days, corresponding to an attack rate (λ_c) of once per 4 days to once per 28 days. The host IDS false positive probability and false negative probability (H_{pfp} and H_{pfn}) vary between 1% and 5% to reflect the host intrusion detection strength as in [49].

Table 5-2: Input Parameter Values Characterizing a Heterogeneous Clustered WSN.

Parameter	Default Value
N_{SN}	3000
N_{CH}	100
λ_{SN}	30 nodes/(20 x 20 m^2)
λ_{CH}	1 node/(20 x 20 m^2)
E_0^{SN}	0.8 Joules
E_0^{CH}	10 Joules
r_{SN}	[5-25] m
r_{CH}	[25-120] m
$T_{clustering}$	60 sec
q	10^{-6}
e_j	[0.0001 – 0.1]
f	¼
λ_q	1 query/sec
T_{comp} (or $1/\lambda_c$)	[4-28] days
A	200m
n_b	50 bits
E_{elec}	50 nJ/bit
E_{amp}	10 pJ/bit/m^2
d_0	75m
T_{req}	[0.3 – 1.0] sec
H_{pf}, H_{pfn}	[0.01-0.05]

Figure 5-3 shows a high level description of the computational procedure to determine the optimal redundancy level (m_p, m_s) for maximizing MTTF. The MTTF Equation (Equation (5.1)) is embedded on lines 15-21 and 30-31 in Figure 5-3. The accumulation of queries is shown on line 13. The value of N_q is computed on line 32. Lines 7 and 8 contain the conditions the system must hold to remain alive while computing an MTTF value for a specific redundancy level. The computational procedure essentially has a complexity of O($m_p \times m_s$) as it exhaustively searches for the best (m_p, m_s) pair, given a set of input parameter values as listed in Table 5-2 (above) as well as instance values of m (the number of voters for intrusion detection) and T_{IDS} (the intrusion detection interval) characterizing a HWSN.

Input: Table 52 input parameters
Output: $optimal\ MTTF, optimal\ (m_p, m_s)$

for $m_s \leftarrow 1$ **to** $maxMs$ **do**
  for $m_p \leftarrow 1$ **to** $maxMp$ **do**
    $num_q \leftarrow 0$ $where\ num_q\ is\ the\ query\ counter$
    $E_{init}^{SN} \leftarrow N_{SN}(t) \times E_0^{SN}$, $E_{init}^{CH} \leftarrow N_{CH}(t) \times E_0^{CH}$ $where\ t = 0$
    $Compute \lambda_{SN}, \lambda_{CH}, R_q, E_{clustering}^{SN}, E_{clustering}^{CH},$
      $E_q^{SN}, E_q^{CH}, E_{IDS}^{SN}, E_{IDS}^{CH}, at\ \ t = 0$
    $Compute\ arrival\ time\ for\ next\ clustering,$
      $query, and\ IDS\ events\ \ at\ \ t = 0$
    while [$E_{init}^{SN} > E_{threshold}^{SN}$ and $E_{init}^{CH} > E_{threshold}^{CH}$ and
      $f * n_{SN}$ has m_s nodes and $f * n_{ch}$ has m_p nodes] **do**
      $ev \leftarrow next\ event$
      if $ev\ is\ clustering\ event$ **then**
        $E_{init}^{SN} = E_{init}^{SN} - E_{clustering}^{SN}, E_{init}^{CH} = E_{init}^{CH} - E_{clustering}^{CH}$
      else if $ev\ is\ query\ event$ **then**
        $num_q \leftarrow num_q + 1$
        $E_{init}^{SN} = E_{init}^{SN} - E_q^{SN}, E_{init}^{CH} = E_{init}^{CH} - E_q^{CH}$
        if $num_q = 1$ **then** //first query
          $rq\_muls \leftarrow rq\_muls \times R_q$
          $temp \leftarrow num_q \times rq\_muls$
        else //terminate previous query
          $tempMttf \leftarrow tempMttf + temp \times (1 - R_q)$
          $rq\_muls \leftarrow rq\_muls \times R_q$
          $temp \leftarrow num_q \times rq\_muls$
      else // ev is an IDS event
        $Update\ distribution\ of\ good\ and\ bad\ nodes$
        $Compute\ Pfp\ and\ Pfn$
          $E_{init}^{SN} = E_{init}^{SN} - E_{IDS}^{SN}, E_{init}^{CH} = E_{init}^{CH} - E_{IDS}^{CH}$
       $Remove\ Bad\ caught\ and\ Good\ misidentified\ nodes$
        $Compute\ Q_c^{SN}, Q_c^{CH}$
          $Update\ \lambda_{SN}, \lambda_{CH}, N_{SN}, N_{CH}, r_{SN}, r_{CH}$
          $Update\ R_q, E_{clustering}^{SN}, E_{clustering}^{CH}, E_q^{SN}, E_q^{CH}$
          $tempMttf \leftarrow tempMttf + temp$
          $Mttf \leftarrow tempMttf$
          $N_q \leftarrow num_q$
    if $Mttf > optimalMttf$ **then**
      $optimalMttf \leftarrow Mttf$
      $optimal\ (m_p, m_s) \leftarrow\ (m_p, m_s)$
return $optimalMttf\ and\ \ optimal\ (m_p, m_s)$

Figure 5-3: Computational Procedure to Determine Optimal (m_p, m_s) for Maximizing MTTF.

A query response propagates over SNs for source redundancy (m_s) and over CHs for path redundancy (m_p). Hence, m_s directly affects energy consumption of SNs and m_p directly affects energy consumption of CHs. Figure 5-4 to Figure 5-6 summarize the effect of (m_p, m_s) on the CH/SN energy, query reliability, and CH/SN radio range, respectively, for the case in which T_{comp} = 4 days and T_{IDS}= 10 hrs. In Figure 5-4, a relatively high m_p leads to quick energy depletion of a CH node. Similarly, a relatively high m_s leads to quick energy depletion of a SN. While energy determines the number of queries the system is able to execute, the system lifetime largely depends on query reliability. Figure 5-5 shows the effect of (m_p, m_s) on query reliability. The combination of (4, 3) has the highest query reliability over other combinations of (2, 5) or (5, 2) in this test scenario.

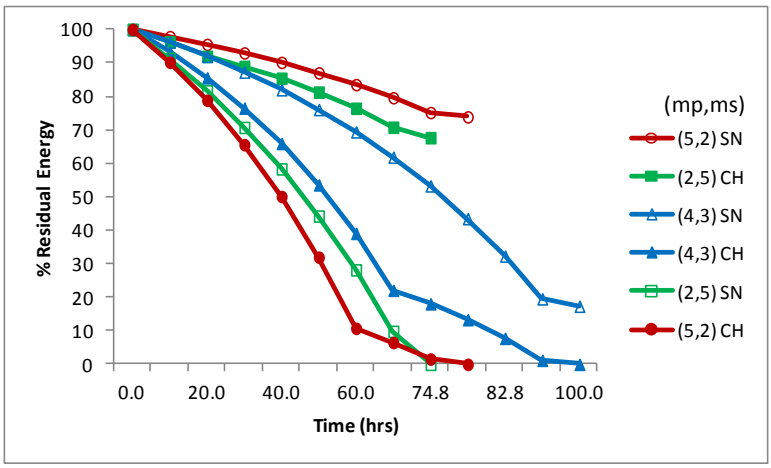


Figure 5-4: Effect of (m_p, m_s) on Energy of CHs and SNs.

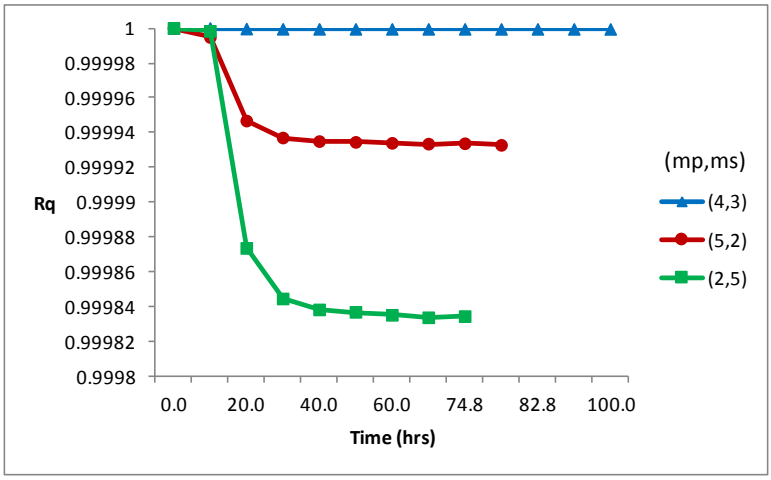


Figure 5-5: Effect of (m_p, m_s) on Query Reliability (Rq).

The system dynamically adjusts the radio range of CHs and SNs to maintain network connectivity based on Equation (5.10) as nodes are being removed from the system because of failure or eviction. Figure 5-6 shows that the rates at which radio ranges of CHs and SNs increase are highly sensitive to m_p and m_s, respectively. A sharp increase of the radio range affects the energy consumption rate and thus the system lifetime. Overall, Figure 5-4 to Figure 5-6 indicate that there exist an optimal combination of (m_p, m_s) that will maximize the system lifetime. Figure 5-7 confirms that among three (m_p, m_s) combinations, (4, 3) results in the highest MTTF, since it has the highest query reliability without consuming too much energy per query execution.

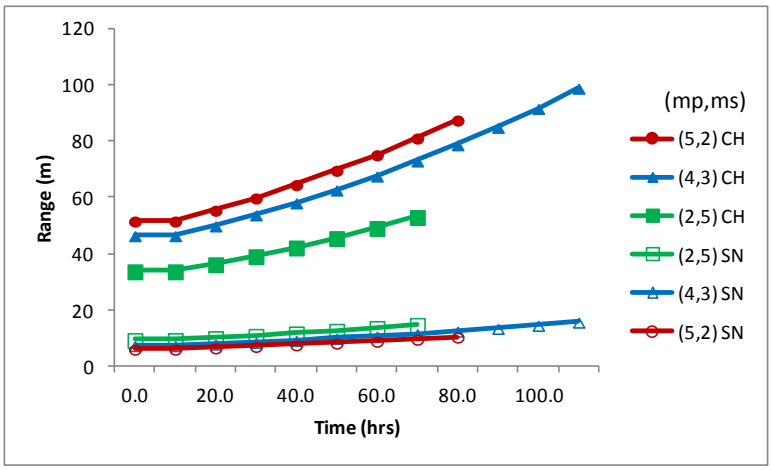


Figure 5-6: Effect of (m_p, m_s) on Radio Range of CHs and SNs.

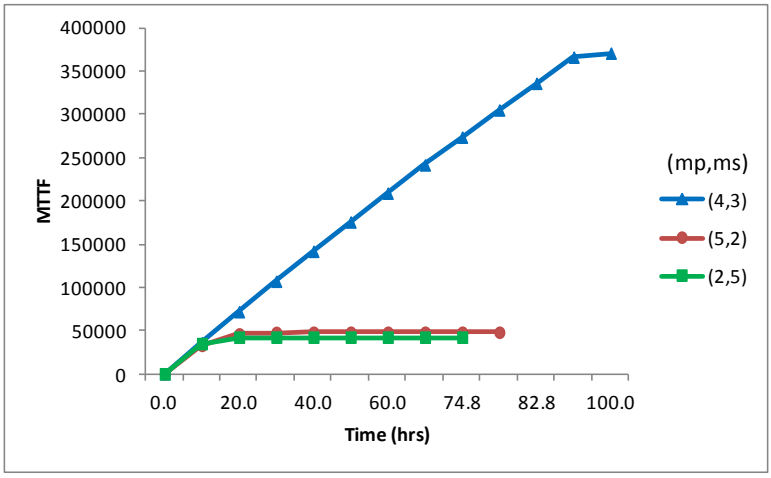


Figure 5-7: Effect of (m_p, m_s) on MTTF.

Next we analyze the effect of T_{comp}, m and T_{IDS} on optimal (m_p, m_s). Figure 5-8 and Figure 5-9 show MTTF vs. (m_p, m_s) under low and high attack rates, respectively. First of all, in both graphs, we observe the existence of an optimal (m_p, m_s) value under which MTTF is maximized. Secondly, there exists an optimal m value (the number of voters) to maximize MTTF. In Figure 5-9, m=7 yields a higher MTTF value than m=3 because in this scenario the attack rate is relatively high (one in four days), so a higher number of voters is needed to cope with and detect bad nodes more effectively, to result in a higher query success rate and thus a higher MTTF. Comparing these two graphs, we observe a trend that as the capture rate increases (i.e., going from the left graph to the right graph), the optimal m value level increases. The reason is that as the capture rate increases, there are more and more malicious nodes in the system, so using more voters (e.g. m=7) can help identify and evict malicious nodes more effectively, thus increasing the query success probability and consequently the MTTF value. The system is better off this way to cope with increasing malicious node population for lifetime maximization even though more energy is consumed due to more voters being used. By comparing these two graphs, we observe a trend that as the capture rate increases (i.e., going from Figure 5-8 to Figure 5-9), the optimal (m_p, m_s) redundancy level increases. When the capture rate increases from once in three weeks (T_{comp} = 3 weeks) to once in four days (T_{comp} = 4 days), the optimal m changes from m=3 to m=7. We also observe that the optimal (m_p, m_s) redundancy level changes from (3, 3) to (4, 4) when m=3. The reason behind this trend is that as more nodes are compromised in the system, a higher redundancy must be used to cope with packet dropping attacks. While increasing (m_p, m_s) consumes more energy, the gain towards increasing the query success probability (and thus towards increasing MTTF) outweighs the loss of lifetime due to energy consumption.

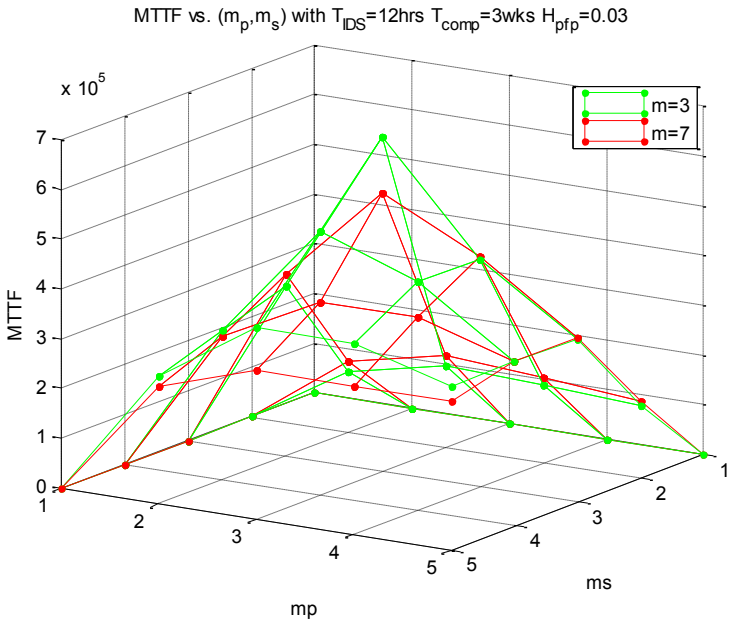


Figure 5-8: MTTF vs. (m_p, m_s) under Low Capture Rate.

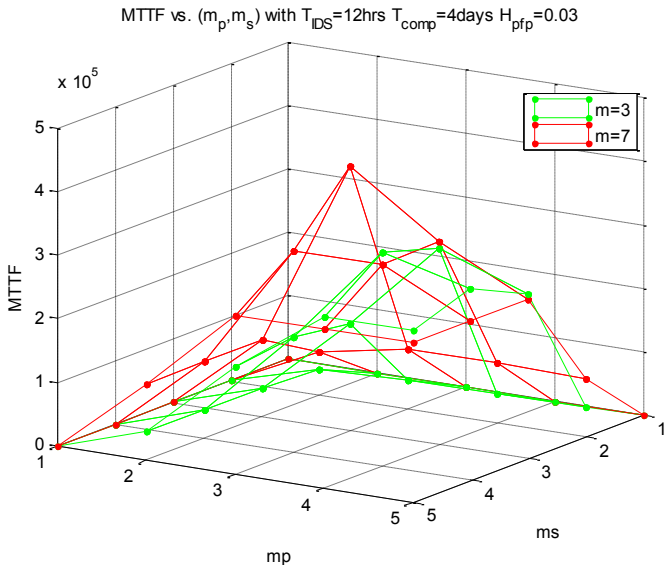


Figure 5-9: MTTF vs. (m_p, m_s) under High Capture Rate.

Another trend exhibited in Figure 5-8 and Figure 5-9 is that as the number of voters in intrusion detection (m) increases, the optimal (m_p, m_s) redundancy level decreases. This is because increasing m has the effect of detecting and evicting bad nodes more effectively, thus requiring a lower level of redundancy in (m_p, m_s) to cope with packet dropping attacks by bad nodes. In Figure 5-9, when m=3, the optimal (m_p, m_s) = (4, 4) while when m=7 the optimal (m_p, m_s) = (3, 3).

In Figure 5-10, we compare MTTF vs. (m_p, m_s) under three cases: (a) there are no malicious nodes and no intrusion detection (the top curve); (b) there are malicious nodes but there is no intrusion detection (the bottom curve); (c) there are malicious nodes and there is intrusion detection (the middle two curves). First of all, in each case we observe the existence of an optimal (m_p, m_s) value under which MTTF is maximized. Secondly, for the special case in which there are no malicious nodes (the top curve), the optimal (m_p, m_s) is (3, 3). When there are malicious nodes, however, the optimal (m_p, m_s) value becomes (7, 7) because using higher redundancy in multisource multipath routing is necessary to cope with malicious nodes that perform insider attacks. By applying intrusion detection, the MTTF value of the system under attack is increased. Figure 5-10 reflects the IDS case in Figure 5-8.

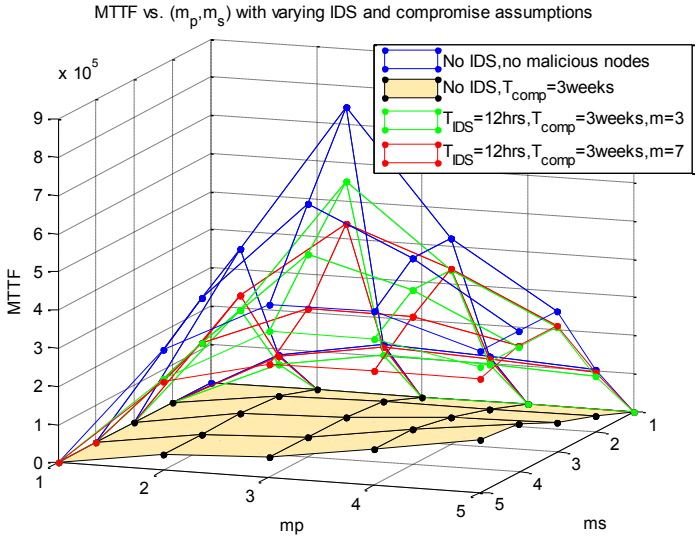


Figure 5-10: MTTF vs. (m_p, m_s) for Three Cases.

We summarize the effect of T_{comp} and m on the optimal (m_p, m_s) value in Table 5-3 and the effect of T_{comp} on the optimal m value at which MTTF is maximized in Table 5-4. Table 5-3 shows that as the capture rate increases (i.e., a smaller T_{comp}), the optimal (m_p, m_s) redundancy level should increase in order to cope with more bad nodes in the system performing packet dropping attacks. Also, as the number of voters (m) decreases and thus the detection strength decreases, the optimal (m_p, m_s) redundancy level should increase to cope with more bad nodes in the system. Table 5.4 shows that as the capture rate increases (i.e., a smaller T_{comp}), the optimal m value level should increase so as to strengthen intrusion detection to remove more bad nodes from the system. Also as T_{IDS} decreases so that the detection strength increases, the optimal m value should decrease so as not to waste energy unnecessarily to adversely affect the system lifetime.

Table 5-3: Optimal (m_p, m_s) with varying T_{comp} and m.

T_{comp}	m=3	m=5	m=7
4 days	(5,4)	(4,4)	(4,4)
1 week	(4,4)	(3,3)	(3,3)
3weeks	(3,3)	(3,3)	(3,3)

Table 5-4: Optimal m with varying T_{comp} and T_{IDS}.

T_{comp}	T_{IDS}=4hrs	8hrs	14hrs	18hrs	24hrs
4 days	3	5	7	7	7
1 week	3	5	5	7	7
2 weeks	3	3	5	5	7
3 weeks	3	3	5	5	5

Next we analyze the effect of T_{IDS} on MTTF. Figure 5-11 and Figure 5-12 show MTTF vs. T_{IDS} with varying m under low capture rate (T_{comp} = 3 weeks) and high capture rate (T_{comp} = 1 week), respectively. We first observe that there exists an optimal T_{IDS} value under which MTTF is maximized. Furthermore, the optimal T_{IDS} value increases as m increases. For example, in Figure 5-12 as m increases from 3, 5 to 7 we see that correspondingly the optimal T_{IDS} at which MTTF is maximized increases from 8, 10 to 16 hours. The reason is that as the number of voters increases so the intrusion detection capability increases per invocation, there is no need to invoke intrusion detection too often so as not to waste energy and adversely shorten the system lifetime. We also observe two general trends. One trend is that as T_{IDS} increases, the optimal m value increases. The reason is that when T_{IDS} is small so intrusion detection is invoked frequently, we don't need many voters per invocation so as not to waste energy unnecessarily to adversely shorten the system lifetime. The second trend shown in Figure 5-11 and Figure 5-12 is that as the node capture rate increases, the optimal m value increases in order

to cope with more compromised nodes in the system. These two trends correlate well with those summarized in Table 5-4 earlier.

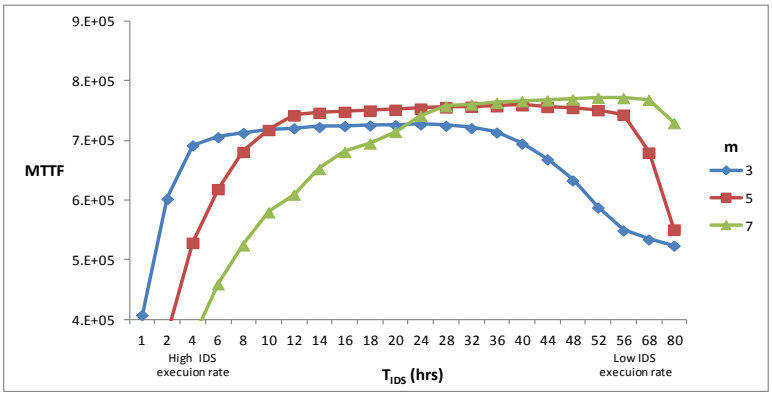


Figure 5-11: Effect of T_{IDS} on MTTF under Low Capture Rate.

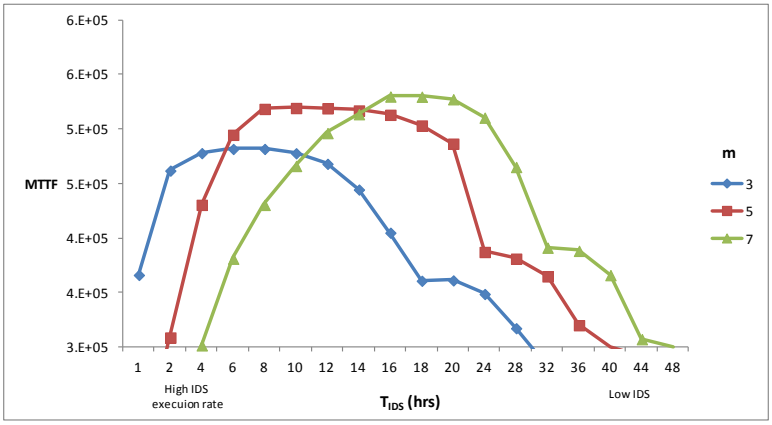


Figure 5-12: Effect of T_{IDS} on MTTF under High Capture Rate.

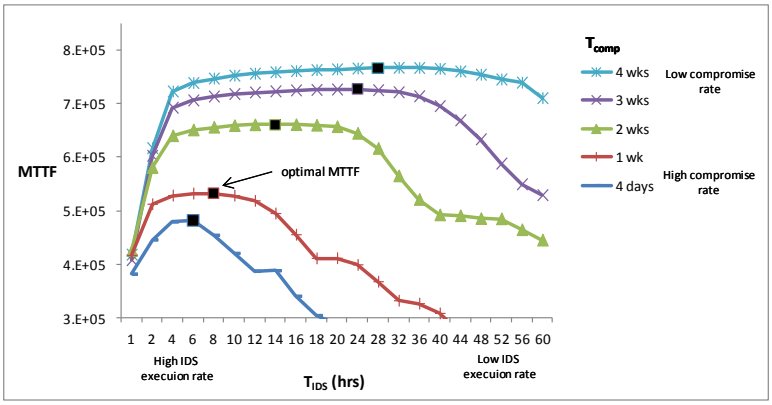


Figure 5-13: Effect of Capture Rate on Optimal T_{IDS}.

Lastly, we examine the sensitivity of the optimal T_{IDS} to the capture rate. Figure 5-13 shows MTTF vs. T_{IDS} with varying T_{comp} values. It exhibits the trend that as the capture rate increases (a smaller T_{comp} value), the optimal T_{IDS} at which MTTF is maximized must decrease to cope with malicious attacks. For example, in Figure 5-13 the optimal T_{IDS} is 24 hours when T_{comp} = 4 weeks and reduces to 6 hours when T_{comp} = 4 days. The reason is that when the capture rate is low and hence the malicious node population is low, the negative effects of wasting energy for IDS execution (through evicting falsely identified nodes and executing the voting mechanism) outweighs the gain in the query success probability, so the system is better off by executing intrusion detection less often. On the other hand, when the capture rate is high and the malicious node population is high, the gain in the query success probability because of evicting malicious nodes often outweighs the energy wasted because of frequent IDS execution, so the system is better off by executing intrusion detection often. Table 5-5 summarizes the effect of T_{comp} and m on the optimal T_{IDS} value at which MTTF is maximized. Table 5-6 further summarizes the effect of T_{comp} on the maximum radio range required to maintain network connectivity in terms of the optimal redundancy level to prolong the system lifetime.

Table 5-5: Optimal T_{IDS} with varying T_{comp} and m.

T_{comp}	m=3	m=5	m=7
4 days	6 hrs	10	14
1 week	8	10	16
2 weeks	14	24	36
3 weeks	24	40	52

Table 5-6: Effect of Capture Rate on Maximum Radio Range to Maintain Connectivity.

T_{comp}	r_{SN}	r_{CH}
4 days	21.5m	117.7m
1 week	15.9	82.2
2 weeks	11.4	62.4

3 weeks	10.9	60

5.5 Summary

In this chapter we performed a tradeoff analysis of energy consumption vs. QoS gain in reliability, timeliness, and security for redundancy management of heterogeneous clustered wireless sensor networks utilizing multipath routing to answer user queries. We developed a novel probability model to analyze the best redundancy level in terms of path redundancy (m_p) and source redundancy (m_s), as well as the best intrusion detection settings in terms of the number of voters (m) and the intrusion invocation interval (T_{IDS}) under which the lifetime of a heterogeneous wireless sensor network is maximized while satisfying the reliability, timeliness and security requirements of query processing applications in the presence of unreliable wireless communication and malicious nodes. Finally, we applied our analysis results to the design of a dynamic redundancy management algorithm to identify and apply the best design parameter settings at runtime in response to environment changes to prolong the system lifetime.

Chapter 6

Adaptive Network Management for Countering Smart Attack and Selective Capture in Wireless Sensor Networks

In this chapter we analyze adaptive network management for countering smart attack and selective capture which aim to cripple the basic data delivery functionality of a wireless sensor network. We consider 3 countermeasures in the protocol design: (1) dynamic radio range adjustment; (2) multisource multipath routing for intrusion tolerance; and (3) voting-based intrusion detection. We identify the best protocol settings in terms of the best redundancy level used for multisource multipath routing, and the best number of voters and the intrusion invocation interval used for intrusion detection under which the sensor network lifetime is maximized in the presence of selective capture which turns nodes into malicious nodes capable of performing random, opportunistic and insidious attacks to evade detection and maximize their chance of success. This chapter is based on our work published in [5].

6.1 System Model

6.1.1 WSN Environments

We consider a base station (BS) based WSN with low-power SNs distributed in a geographic area. There is a base station assigned to the WSN that interconnects the WSN to the outside world and that fields queries from the outside world for sensing results. Queries arrive at the system following a Poisson process with rate λ_q. A query failure is considered as a critical system failure. The initial energy of each SN is E_o^{SN}. The deployment area of the WSN is assumed circular with radius r^{BS}. We consider random deployment where SNs are deployed randomly (e.g., through air drop) and distributed according to homogeneous spatial Poisson processes with density λ_o^{SN}. The total number of SNs initially in the system thus is $N_o^{SN} = \lambda_o^{SN}\pi(r^{BS})^2$.

6.1.2 Selective Capture and Smart Attack Model

All SNs are subject to capture attacks. With "selective capture," the adversaries (humans or robots) strategically capture SNs and turn them into *inside* attackers. We represent the capture rate of a SN at a distance x away from the BS at time t by $\lambda_c^{SN}(x,t)$. To test our idea of countermeasures against selective capture attacks, we consider a linear function form, i.e., the capture rate drops linearly as the SN is further away from the base station:

$$\lambda_c^{SN}(x,t) = \lambda_c^{max} - \frac{x}{r^{BS}}(\lambda_c^{max} - \lambda_c^{min}) \tag{6.1}$$

where λ_c^{max} is the maximum capture rate the adversary can possibly have; and λ_c^{min} is the minimum capture rate. A baseline case against which this linear selective capture case will be compared is "random capture" by which the adversary, given the same energy and capacity, randomly performs capture attacks, i.e., $\lambda_c^{random} = (\lambda_c^{max} + \lambda_c^{min})/2$ at all distances. We note that these two capture models have the same overall capture rate accounting for the overall capability of the capturers in the system.

After a node is compromised it becomes an inside attacker. An inside attacker can perform packet dropping [81] to block data delivery. It can also perform slandering attacks by recommending a good node as a bad node, and a bad node as a good node when participating in intrusion detection activities. As a result, slandering attacks can cause good nodes being misdiagnosed and evicted from the system, and bad nodes being missed and remained in the system. This effectively creates an area with a high concentration of bad nodes, especially for critical SN areas with a high capture rate under selective capture.

In the literature it is often assumed that an inside attacker performs attacks constantly, without giving consideration to evade intrusion detection. We characterize a smart attacker by its capability to perform *random*, *opportunistic* and *insidious* attacks. First of all, a smart attacker can perform random or on-off attacks, i.e., attacking with a random probability p_a, to evade intrusion detection. Second, a smart attacker can perform opportunistic attacks, i.e., it attacks only when it sees opportunities which can lead to a successful attack while still eluding detection. Finally, it can perform insidious attacks, i.e., it can perform on-off attacks to evade intrusion detection until a critical mass of compromised node population is reached after which it performs "all in" attacks ($p_a = 1$) to cripple the system totally.

6.1.3 Countermeasures against Attacks

Our first countermeasure against selective capture and smart attack is dynamic radio range adjustment. With random deployment, the initial radio range is denoted by r_o^{SN} such that a SN is able to connect to n_0 neighbors for maintaining network connectivity. A SN adjusts its radio range dynamically throughout its lifetime to maintain connectivity such that the average number of 1-hop neighbor SNs remains at n_0. Thus, SNs closer to the BS may have to increase radio range more than SNs away from the BS to counter selective capture. Any communication between two nodes with a distance greater than single hop radio range between them would require a multi-hop.

Our second countermeasure is multipath routing for intrusion tolerance. This is achieved through two forms of redundancy: (a) source redundancy by which m_s SNs sensing a physical phenomenon in the same feature zone are used to forward sensing data to the BS; (b) path redundancy by which m_p paths are used to relay packets from a source SN to the BS. We assume geographic forwarding is being used to packet routing; thus, no path information is maintained.

While data delivery could fail due to hardware failure and transmission failure because of noise and interference [10], we only consider failure caused by compromised nodes performing packet drop attacks and data modification attacks. We assume that SNs operate in power saving mode (e.g. [22, 107]). Thus, a SN is either active (transmitting or receiving) or in sleep mode. For the transmission and reception energy consumption of sensors, we adopt the energy model in [126] for SNs. We assume that the BS will have pairwise keys with the SNs. A SN also has a pairwise key with each of its neighbors, up to a few hops away for future expandability. Thus, a SN can encrypt data for confidentiality and authentication purposes.

Our last countermeasure is voting-based intrusion detection system (IDS) mechanisms to detect and evict compromised nodes. Every SN runs a simple *host IDS* to assess its neighbors. The host IDS is light-weight to conserve energy. It is also generic and does not rely on the feedback mechanism tied in with a specific routing protocol (e.g., MDMP for WSNS [86] or AODV for MANETs [112]). It is based on local monitoring. That is, each node monitors its neighbor nodes only. Each node uses a set of anomaly detection rules such as a high discrepancy in the sensor reading or recommendation has been experienced, a packet is not forwarded as requested, as well as interval, retransmission, repetition, and delay rules as in [14, 19, 49, 108]. If the count exceeds a system-defined threshold, a neighbor node that is being monitored is considered compromised.

The imperfection of monitoring due to ambient noise and channel error is modeled by a monitoring error probability $p_{err.}$ The host IDS false positive probability (H_{pfp}) for misidentifying a good node as a bad node is due to this monitoring error, so $H_{pfp} = p_{err.}$

On the other hand, the host IDS false negative probability (H_{pfn}) for misidentifying a bad node as a good node is due to this monitoring error as well as how often a bad node performs attacks (with probability p_a). Hence, $H_{pfn} = (1 - p_a) + p_{err}$ upper bounded by 1. A voting-based distributed IDS is applied periodically in every T_{IDS} time interval. A SN is being assessed by its neighbor SNs. In each interval, m neighbor SNs around a target SN will be chosen randomly as voters and cast their votes based on their host IDS results to collectively decide if the target SN is still a good node. The m voters share their votes through secure transmission using their pairwise keys. When the majority of voters come to the conclusion that a target node is bad, then the target node is evicted. There is a system-level false positive probability P_{fp}^{SN} that the voters can incorrectly identify a good node as a bad node. There is also a system-level false negative probability P_{fn}^{SN} that the voters can incorrectly misidentify a bad node as a good node. In this chapter, we will derive the two system-level IDS probabilities based on slandering attacks performed by smart attackers exhibiting random, opportunistic and insidious attack behaviors. Finally we note that the system's intrusion detection strength is modeled by the detection interval T_{IDS} (the shorter the stronger) and the number of neighbor voters m (the more the stronger).

6.1.4 Mean Time to Failure as the Performance Metric

To provide a unifying metric that considers the above two design tradeoffs, we define the total number of queries the system can answer correctly until it fails as the *lifetime* or the *mean time to failure* (the MTTF) of the system, which can be translated into the actual system lifetime span given the query arrival rate. A failure occurs when no response toward a query is received. The cause could be due to energy exhaustion, or packet dropping by malicious nodes. Our aim is to find both the optimal redundancy levels and IDS settings under which the MTTF is maximized.

6.2 Problem Definition, Solution, and Adaptive Network Management Algorithm Description

The problem we are solving is that given a set of input parameters values characterizing the BS-based WSN operational environment, selective capture and smart attack behaviors as defined in Section 6.1, we want to dynamically apply the best decision variable settings to maximize the lifetime of the WSN in terms of its MTTF. The decision variables are those defined for the 3 countermeasures against selective capture and smart attacks, namely, the connectivity degree parameter (n_0)for dynamic radio range adjustment, the path redundancy (m_p) and source redundancy (m_s) for multisource multipath routing, and the number of voters (m) and intrusion invocation interval (T_{IDS}) for

voting-based intrusion detection. Our solution methodology is to model network dynamics by means stochastic processes (in Section 6.3) to yield a closed form solution for the MTTF as a function of these decision variables. The optimal decision variable settings obtained at design time are stored in a table and then applied at runtime by means of O(1) table lookup to implement adaptive network management.

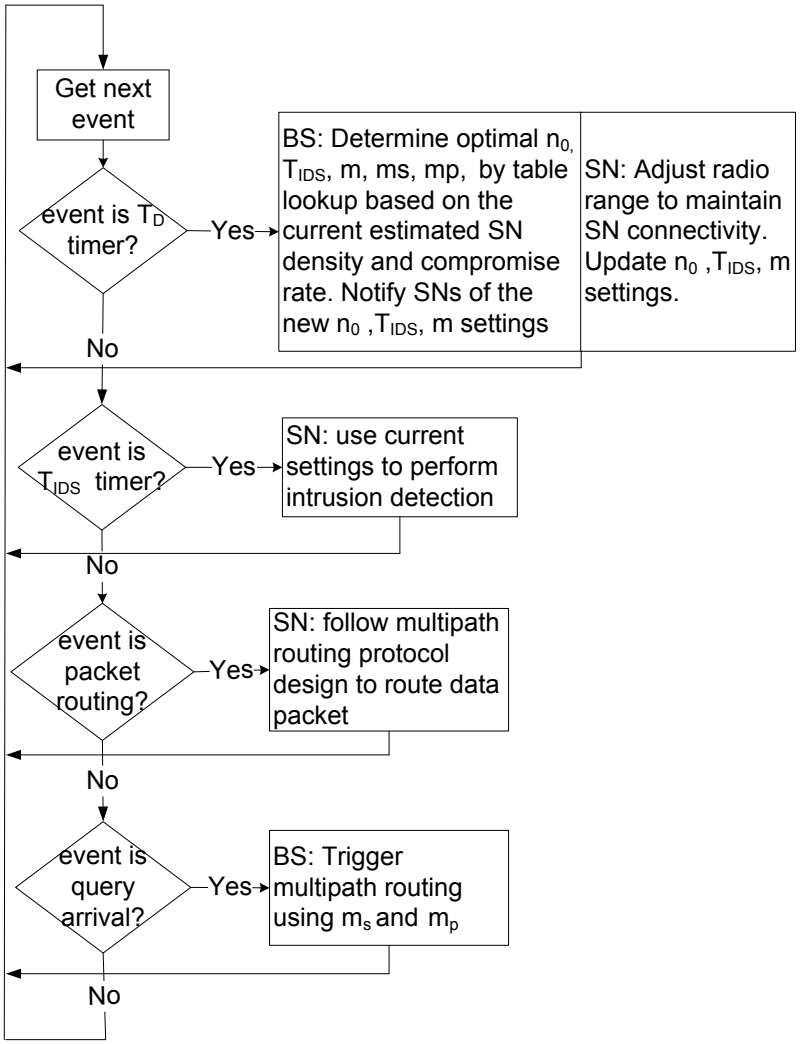


Figure 6-1: Adaptive network management algorithm flowchart.

We use a flowchart as shown in Figure 6-1 to describe our adaptive network management algorithm. All nodes in the system act periodically to a "T_D timer" event to adjust the optimal parameter setting in response to changing environments. This is indicated by an "event is T_D timer" box. The optimal design settings in terms of optimal $n_0, T_{IDS}, m, m_s,$ and m_p are determined at design time and pre-stored in a table over perceivable ranges of input parameter values. The BS applies a table lookup operation with extrapolation techniques [18, 67, 131] to determine the optimal design parameter set-

tings. The complexity is O(1) because of the table lookup technique employed. The action performed by a BS upon a T_D timer event includes (a) determining $n_0, T_{IDS}, m, m_s,$ and m_p based on runtime knowledge of node density and capture rate experienced; and (b) notifying SNs of the new n_0, T_{IDS} and m settings. The action performed by a SN upon this $T_D$timer event is to adjust its radio range to maintain SN connectivity and to update its n_0, T_{IDS} and m settings. These actions are specified in the two action boxes to the right of the "event is T_D timer" box, with "BS" and "SN" labeling the agents involved.

When the T_{IDS} timer event happens (indicated by the "event is T_{IDS} timer" box), each SN uses it current T_{IDS} and m settings to perform intrusion detection. This action is specified in the single action box to the right of the "event is T_{IDS} timer" box, with "SN" labeling the agents involved.

When a data packet arrival event occurs (indicated by the "event is packet routing" box), each SN simply follows the prescribed multipath routing protocol to route the packet. This action is specified in the single action box to the right of the "event is packet routing" box, with "SN" labeling the agents involved.

When the BS receives a query from a user (indicated by the "event is query arrival" box), it triggers multipath routing for intrusion tolerance using the current optimal (m_s, m_p) setting to prolong the system useful lifetime. The complexity is also O(1) for the BS. This action is specified in the single action box to the right of the "event is query arrival" box, with "BS" labeling the agent involved.

6.3 Probability Model

Table 6-1: Parameter List.

Parameter	Meaning	Type
λ_c^{max}	maximum capture rate the adversary can have	input
λ_c^{min}	minimum capture rate	input
λ_c^{random}	random capture rate	derived
$\lambda_c^{SN}(x,t)$	capture rate of a SN at a distance x away from the BS at time t	derived
$\lambda_{good}^{SN}(x,t_{I,i})$	Density of good SNs at distance x from the BS at time t	derived
$\lambda_{bad}^{SN}(x,t_{I,i})$	Density of bad SNs at distance x from the BS at time t	derived
p_{all-in}	Insidious behavior threshold percentage for "all-in" attacks	input
$P_{bad}(x,t)$	the percentage of malicious nodes at location x and time t	derived

p_a	Probability of attack	input
p_{err}	monitoring error prob. due to ambient noise and channel error	input
n_0	neighbors for maintaining network connectivity	input
r^{BS}	Radius of circular sensor area (meter)	input
n_b	Size of a data packet (bit)	input
E_{elec}	Energy dissipation to run the transmitter and receiver circuitry (J/bit)	input
E_{amp}	Energy used by the transmit amplifier to achieve an acceptable signal	input
E_o^{SN}	Initial energy per node (Joule)	input
$E_{IDS}^{SN}(x, t_{I,i})$	Energy consumed by a SN located at x in the i^{th} IDS cycle (Joule)	derived
$E_q^{SN}(x, t_{Q,i})$	Energy consumed for executing the ith query at time t (Joule)	derived
$R_q(t)$	Probability that a query reply at time t is delivered successfully	derived
r_o^{SN}	initial Wireless radio communication range (meter)	input
$r^{SN}(x,t)$	radio range of SN at distance x from its BS at time t	derived
$n^{SN}(x,t)$	number of neighbors at distance x from BS at time t	derived
$\check{n}^{SN}(x,t)$	number of forwarding neighbors at distance x from BS at time t	derived
$n_{good}^{SN}(x,t)$	number of good neighbors at distance x from BS at time t	derived
$n_{bad}^{SN}(x,t)$	number of bad neighbors at distance x from BS at time t	derived
N_q	Maximum number of queries before energy exhaustion	derived
m_p	Path redundancy level: Number of paths from a source SN to the sink	design
m_s	Source redundancy level: Number of source SNs per query response	design
f	Fraction of neighbor nodes that will forward data	input
$\lambda^{SN}(x,t)$	the density of SNs at distance x from the BS at time t	derived

λ_q	Query arrival rate (times/sec)	input
α	Ratio of IDS execution rate to query arrival rate	input
m	The query-based clustered WSN is characterized by a set of input pa-	design
H_{pfp}	Probability of host IDS false positive	input
H_{pfn}	Probability of host IDS false negative	input
Pfp	system-level false positive probability	derived
Pfn	system-level false negative probability	derived
T_{IDS}	IDS interval time (sec)	design
MTTF	Lifetime of a WSN	output

In this section we develop a probability model to estimate the MTTF of a query-based WSN built with the three countermeasure mechanisms in the protocol design. The novelty lies in the way we estimate the densities of good nodes, "active" bad nodes, and "inactive" bad node as a function of *location* and *time*, given a set of input parameter values charactering the operational and environmental conditions, and adversary behaviors. This allows us to estimate if a bad node is performing attacks or not at location x and time t, and, consequently, if it will perform packet dropping and slandering attacks. Consequently, we can reasonably predict if data delivery through a node at location x and time t will succeed or fail.

Table 6-1 provides the parameters list along with the physical meaning. A model parameter in our formulation can be an *input*, *derived*, *design* or *output* parameter. Specifically, m_p (path redundancy), m_s (source redundancy), m (the number of voters for intrusion detection) and T_{IDS} (the intrusion detection interval) are design parameters whose values are to be identified to maximize the system MTTF. Derived parameters are those deriving from input parameters. There is only one output parameter, namely, the MTTF. Note that most derived parameters are dynamic, i.e., as a function of time. For example, SN density, denoted by $\lambda^{SN}(x,t)$, decreases over time because of node failure/eviction as time progresses. On the other hand, radio range, denoted by $r^{SN}(x,t)$, increases over time to maintain connectivity.

The basic idea of our MTTF formulation is that we first deduce the maximum number of queries, N_q, the system can possible handle before running into energy exhaustion for the best case in which all queries are processed successfully. Because the system evolves dynamically, the amount of energy spent per query also varies dynamically. Given the average interval between query arrivals being $1/\lambda_q$, we can reasonably esti-

mate the amount of energy spent due to query processing and intrusion detection for query j based on the query arrival time $t_{Q,j}$. We then derive the corresponding query success probability $R_q(t_{Q,j})$, that is, the probability that the response to query j arriving at time $t_{Q,j}$ is delivered successfully to the BS. Finally, we compute the MTTF as the probability-weighted average of the number of queries the system can handle without experiencing any failure. More specifically, the MTTF is computed by:

$$MTTF = \sum_{i=1}^{N_{q-1}} i\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)\left(1 - R_q(t_{Q,i+1})\right) + N_q \prod_{j=1}^{N_q} R_q(t_{Q,j}) \tag{6.2}$$

Here $(\prod_{j=1}^{i} R_q(t_{Q,j}))\left(1 - R_q(t_{Q,i+1})\right)$ accounts for the probability of the system being able to successfully execute i consecutive queries but failing the i+1th query. The second term is for the best case in which all queries are processed successfully without experiencing any failure for which the system will have the longest lifetime span.

6.3.1 Modeling Network Dynamics due to Capture

Let $\lambda^{SN}(x,t)$ represent the density of SNs at distance x from the BS at time t. Initially at deployment time all SNs are good nodes, so $\lambda^{SN}(x,0) = \lambda_0^{SN}$ for all x's.

As time progresses some SNs are captured and turned into compromised nodes. Moreover, some SNs may be diagnosed as bad nodes and get evicted from the system. Let T be the capture time of a SN following a distribution function $F_c(t)$. Then, the probability that a SN at location x away from the BS is compromised at time t, given that it was a good node at time t-T_{IDS}, denoted by $P_c^{SN}(x,t)$, is given by:

$$\begin{aligned} P_c^{SN}(x,t) &= 1 - P\{T > t | T > t - T_{IDS}\} \\ &= 1 - \frac{P\{T > t, T > t - T_{IDS}\}}{P\{T > t - T_{IDS}\}} = 1 - \frac{1 - F_c(t)}{1 - F_c(t - T_{IDS})} \end{aligned} \tag{6.3}$$

In the special case in which the capture time is exponentially distributed, $P_c^{SN}(x,t) = 1 - e^{-\lambda_c^{SN}(x,t)\times T_{IDS}}$ for a SN at distance x from its BS. Recall that the voting-based distributed IDS executes periodically with T_{IDS} being the interval. At the ith IDS execution time (denoted by $t_{I,i}$), a good SN at distance x from its BS may be compromised with probability $P_c^{SN}(x, t_{I,i})$ since the previous IDS execution time $(t_{I,i-1})$. Let $\lambda_{good}^{SN}(x,t)$ and $\lambda_{bad}^{SN}(x,t)$ denote the densities of good and bad SNs at distance x from the BS at time t, respectively. Then, the densities of good and bad SNs at time $t_{I,i}$ just prior to IDS execution can be recursively estimated from the densities of good and bad SNs at time $t_{I,i-1}$ by:

$$\begin{aligned}\lambda_{good}^{SN}(x,t_{I,i}) &= \lambda_{good}^{SN}(x,t_{I,i-1}) \\ &\quad - P_c^{SN}(x,t_{I,i})\lambda_{good}^{SN}(x,t_{I,i-1}) \\ \lambda_{bad}^{SN}(x,t_{I,i}) &= \lambda_{bad}^{SN}(x,t_{I,i-1}) \\ &\quad + P_c^{SN}(x,t_{I,i})\lambda_{good}^{SN}(x,t_{I,i-1})\end{aligned} \tag{6.4}$$

The boundary conditions are $\lambda_{good}^{SN}(x,0) = \lambda_0^{SN}$ and $\lambda_{bad}^{SN}(x,0) = 0$ for all x's.

As our first countermeasure design, every SN dynamically adjusts its radio range for maintaining connectivity with its peers such that on average the number of 1-hop neighbor nodes is n_o to support its intended functions including routing and participating in majority voting IDS as a verifier. In particular, critical SNs (i.e., nodes that are close to the BS) must increase radio range more due to more nodes being evicted as a result of more intensive capture and slandering attacks toward critical SNs.

Let $r^{SN}(x,t)$ denote the radio range of a SN at distance x from its BS at time t so it can find n_o SNs within radio range. Since the SN density is a function of the distance (x) away from the BS, we have to solve $r^{SN}(x,t)$ by integration of the SN population from x-$r^{SN}(x,t)$ to x+$r^{SN}(x,t)$. Let X and Y be two variables denoting the X and Y coordinates in the X-Y coordinate system. Since $Y^2 = r^{SN}(x,t)^2 - X^2$ and $\int_0^{r^{SN}(x,t)} Y dX$ gives the area of the upper semicircle, the expected number of SNs covered by radio range, denoted by $r^{SN}(x,t)$, can be obtained by solving the following equation:

$$2\int_{-r^{SN}(x,t)}^{r^{SN}(x,t)} \lambda^{SN}(x+X,t)\ \sqrt{r^{SN}(x,t)^2 - X^2}dX = n_o \tag{6.5}$$

where the integral gives the expected number of SNs (accounting for density variation along X) located in the upper or lower half circle.

For notational convenience, let $n^{SN}(x,t)$ be the number of neighbor SNs of a SN located at distance x from the BS at time t, $\check{n}^{SN}(x,t)$ be the number of forwarding neighbors (with f=1/4 for geographical routing), $n_{good}^{SN}(x,t)$ be the number of good neighbors, and $n_{bad}^{SN}(x,t)$ be the number of bad neighbors at time t. Since we know the densities of good and bad nodes at time $t_{I,i}$ just prior to IDS execution, we have:

$$n^{SN}(x,t) = 2\int_{-r^{SN}(x,t)}^{r^{SN}(x,t)} \lambda^{SN}(t)(x+X,t)\ \sqrt{r^{SN}(x,t)^2 - X^2}dX \tag{6.6}$$

$$\check{n}^{SN}(x,t) = \int_{-r^{SN}(x,t)}^{0} \lambda^{SN}(t)(x+X,t)\ \sqrt{r^{SN}(x,t)^2 - X^2}dX$$

$$n_{good}^{SN}(x,t) = 2\int_{-r^{SN}(x,t)}^{r^{SN}(x,t)} \lambda_{good}^{SN}(x+X,t)\ \sqrt{r^{SN}(x,t)^2 - X^2}dX$$

$$n_{bad}^{SN}(x,t) = 2\int_{-r^{SN}(x,t)}^{r^{SN}(x,t)} \lambda_{bad}^{SN}(x+X,t)\ \sqrt{r^{SN}(x,t)^2 - X^2}dX$$

6.3.2 Modeling Insidious Attacks

Under insidious attacks, when a malicious node at location x and time t sees more than a threshold percentage, p_{all-in}, of its peers are malicious, it will perform "all-in" attacks. This is modeled by setting p_a=1. Let $P_{bad}(x,t)$ be the percentage of malicious nodes at location x and time t, defined as follows:

$$P_{bad}(x,t) = \frac{\lambda_{bad}^{SN}(x,t)}{\lambda_{bad}^{SN}(x,t) + \lambda_{good}^{SN}(x,t)} \tag{6.7}$$

More specifically, under the insidious attack, if $P_{bad}(x,t) > p_{all-in}$then $p_a = 1$. This affects the false positive probability, false negative probability and the probability that a node at location x and time t will perform packet dropping attacks, as derived below.

6.3.3 Modeling Random Attacks

We first derive the false positive probability (P_{fp}^R) and false negative probability (P_{fn}^R) at distance x and time t for this case in which bad nodes perform random attacks with probability p_a. Later we extend the derivation to opportunistic attack behavior. Equation (6.8) provides the closed-form solutions for $P_{fp}^R(x,t)$ and $P_{fn}^R(x,t)$ (with x and t omitted for brevity) where n_{bad}^a and n_{bad}^i are the numbers of "active" and "inactive" bad nodes, given by $n_{bad}^{SN}(x,t) \times p_a$ and $n_{bad}^{SN}(x,t) \times (1-p_a)$, respectively; m_{maj} is the minimum majority of m, e.g., 3 is the minimum majority of 5; and ω is H_{pfp} for calculating P_{fp}^R and H_{pfn} for calculating P_{fn}^R.

P_{fp}^R *or* P_{fn}^R = (6.8)

$$\sum_{i=0}^{m-m_{maj}} \left[\frac{C\left(\begin{matrix} n_{bad}^{a} \\ m_{maj}+i \end{matrix}\right) \times C\left(\begin{matrix} n_{good}+n_{bad}^{i} \\ m-(m_{maj}+i) \end{matrix}\right)}{C\left(\begin{matrix} n_{bad}^{a}+n_{bad}^{i}+n_{good} \\ m \end{matrix}\right)} \right]$$
$$+ \sum_{i=0}^{m-m_{maj}} \left[\frac{C\left(\begin{matrix} n_{bad}^{a} \\ i \end{matrix}\right) \times \sum_{j=m_{maj}-i}^{m-i} \left[C\left(\begin{matrix} n_{good}+n_{bad}^{i} \\ j \end{matrix}\right) \times \omega^{j} \times C\left(\begin{matrix} n_{good}+n_{bad}^{i}-j \\ m-i-j \end{matrix}\right) \times (1-\omega)^{m-i-j} \right]}{C\left(\begin{matrix} n_{bad}^{a}+n_{bad}^{i}+n_{good} \\ m \end{matrix}\right)} \right]$$

Recall that the imperfection of monitoring due to ambient noise and channel error is modeled by a monitoring error probability p_{err}, so $H_{pfp} = p_{err}$, and $H_{pfn} = (1 - p_a) + p_{err}$.

We explain Equation (6.8) for the false positive probability P_{fp}^{R}at time t below. The explanation to the false negative probability $P_{fn}^{R}$is similar. A false positive results when the majority of the voters vote against the target node (which is a good node) as compromised. The first term in Equation (6.8) accounts for the case in which more than 1/2 of the voters selected from the target node's neighbors are "active" bad nodes who, as a result of actively performing slandering attacks, will always vote a good node as a bad node. Since more than 1/2 of the m voters vote no, the target node (which is a good node) is diagnosed as a bad node in this case, resulting in a false positive. Here the denominator is the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select at least m_{maj} bad voters out of n_{bad} nodes and the remaining good voters out of n_{good} nodes. The second term accounts for the case in which more than 1/2 of the voters selected from the neighbors are good nodes but unfortunately some of these good nodes mistakenly misidentify the target nodes as a bad node with host IDS false positive probability $H_{pfp,}$ resulting in more than 1/2 of the voters (although some of those are good nodes) voting no against the target node. Since more than 1/2 of the m voters vote no, the target node (which is a good node) is also diagnosed as a bad node in this case, again resulting in a false positive. Here the denominator is again the total number of combinations to select m voters out of all neighbor nodes, and the numerator is the total number of combinations to select i "active" bad voters not exceeding the majority m_{maj}, j good or "inactive" bad voters who diagnose incorrectly with $i + j \geq m_{maj}$, and the remaining $m - i - j$ good or "inactive" voters who diagnose correctly. Here we note that an inactive "bad" voter acts as if it is a good node to evade detection. Also note that more voters do not necessarily provide better detection accuracy since it depends on the percentage of bad node population. That is, if more bad nodes exist than good nodes in the neighborhood, or good nodes have high host false positive probability (H_{pfp}) and host false negative probability (H_{pfn}), then more voters actually will provide less detection accuracy.

6.3.4 Modeling Opportunistic Attacks

Under opportunistic attacks, when a malicious voter sees that the majority of voters for intrusion detection of a target node at location x and time t being malicious nodes (active or inactive), it will collude with other malicious nodes and they together (active and inactive) will perform slandering attacks during voting, resulting in $P_{fp} = 1$ and $P_{fn} = 1$. On the other hand, if it does not see the majority being malicious nodes, it will just perform random attacks as usual, resulting inP_{fp}^{R} and P_{fn}^{R}as given in Equation (6.8). Summarizing above, the system-level false positive probability (P_{fp}) and false negative probability (P_{fn}) at time t under opportunistic attacks are given by:

$$P_{fp}(x,t) = \begin{cases} 1 & if\ \ P_{bad}(x,t) \geq 0.5 \\ P_{fp}^{R}(x,t) & if\ \ P_{bad}(x,t) < 0.5 \end{cases}$$

$$P_{fn}(x,t) = \begin{cases} 1 & if\ \ P_{bad}(x,t) \geq 0.5 \\ P_{fn}^{R}(x,t) & if\ \ P_{bad}(x,t) < 0.5 \end{cases} \tag{6.9}$$

6.3.5 Modeling Network Dynamics due to Intrusion Detection

Our 3rd countermeasure is voting-based IDS to detect and evict suspicious nodes. After the voting-based IDS is executed, however, a good node may be misidentified as a bad node with probability P_{fp} (Equation (6.9)) and mistakenly removed from the WSN. On the other hand, a bad node may be missed with probability P_{fn}(Equation (6.9)) and remained in the system. Consequently, we need to adjust the population of good and bad nodes after IDS execution. Let $\overline{\lambda_{good}^{SN}(x,t_{I,\iota})}$and $\overline{\lambda_{bad}^{SN}(x,t_{I,\iota})}$ denote the densities of good and bad SN nodes located at distance x from the BS, respectively, after IDS execution at time t. Then:

$$\overline{\lambda_{good}^{SN}(x,t_{I,\iota})} = \lambda_{good}^{SN}(x,t_{I,i}) - \lambda_{good}^{SN}(x,t_{I,i}) \times P_{fp}(x,t_{I,i})$$

$$\overline{\lambda_{bad}^{SN}(x,t_{I,\iota})} = \lambda_{bad}^{SN}(x,t_{I,i}) - \lambda_{bad}^{SN}(x,t_{I,i}) \times (1 - P_{fn}(x,t_{I,i})) \tag{6.10}$$

Therefore, the probability that node j located at distance x from its BS is an "active" bad SN performing packet dropping attacks at time $t_{I,i}$,denoted by $Q_{c,j}^{SN}(x,t_{I,i})$, is given by:

$$Q_{c,j}^{SN}(x,t_{I,i}) = \frac{\overline{\lambda_{bad}^{SN}(x,t_{I,\iota})}}{\overline{\lambda_{bad}^{SN}(x,t_{I,\iota})} + \overline{\lambda_{good}^{SN}(x,t_{I,\iota})}} \times p_a \tag{6.11}$$

The first term on the right hand side is the probability node j located at x is a bad node, and the 2nd term is the probability that it is performing attacks (p_a). $Q_{c,j}$ derived above provides critical information because an "active" bad node can perform packet dropping attacks causing a path to be broken if it is on a path from source SNs to the BS.

We note that the good/bad node density will remain the same until the next IDS execution (after T_{IDS} seconds) because the IDS only detects and evicts nodes periodically (given that typically node hardware/software failure happens less frequently than security failure). The remaining nodes are good nodes that pass the IDS evaluation and bad nodes that are undetected by the IDS. Thus, $\overline{\lambda_{good}^{SN}(x, t_{I,i-1})}$ and $\overline{\lambda_{bad}^{SN}(x, t_{I,i-1})}$ obtained at time $t_{I,i-1}$ essentially become $\lambda_{good}^{SN}(x, t_{I,i} - 1)$ and $\lambda_{bad}^{SN}(x, t_{I,i-1})$, respectively, for the next round of IDS execution at time $t_{I,i}$.

We can also estimate the number of SNs in the WSN at time t as:

$$N^{SN}(t) = \int_0^{r^{BS}} \lambda^{SN}(x,t)\ 2\pi x\ dx \tag{6.12}$$

6.3.6 Query Success Probability

We will use the notation SN_j to refer to SN j responsible to relay the packet for the *jth* hop from the source SN to the BS. Also we will use the notation $x(j)$ to refer to the distance from SN_j to its BS.

Let $D_{SN\text{-}BS}$ be the distance between a SN (selected to report sensor readings) and its BS, which on average is $r^{BS}/2$. Then the average numbers of hops to forward data from a source SN to the BS, denoted by N_{hop}^{SN}, can be estimated as follows:

$$\sum_{j=1}^{N_{hop}^{SN}} r^{SN}(x(j), t) = D_{SN-BS} \tag{6.13}$$

The equation above equates the sum of hop distances with the source-destination distance.

The success probability for SN_j to transmit a packet to at least p next-hop SN neighbors (with indices k=1, 2, … p) along the direction of the destination node based on *geographical routing* is given by:

$$\theta_j^{SN}(p) = \sum_{i=p}^{\check{n}^{SN}(x(j),t)} \left[\binom{\check{n}^{SN}(x(j),t)}{i} \prod_{k=1}^{i} (1 - Q_{k,c}^{SN}(x(k),t)) \prod_{k=i+1}^{\check{n}^{SN}(x(j),t)} Q_{k,c}^{SN}(x(k),t) \right] \quad (6.14)$$

where $Q_{k,c}^{SN}(x(k),t)$ is the probability that SN_k is compromised as derived in Equation (6.11), and $\check{n}^{SN}(x(j),t)$ is the number of forwarding neighbor SNs for SN_j as derived from Equation (6.6).

A path starting at SN_j to the BS is successful if in each hop there is at least one healthy next-hop SN neighbor found. Thus, the success probability of a path starting from SN_j (a source node has index j=1) to the BS is given by:

$$\varphi_j^{SN} = \prod_{l=j}^{N_{hop}^{SN}-1} \theta_l^{SN}(1) \quad (6.15)$$

Our 2nd countermeasure is to create m_p paths between a source SN and the BS for *path redundancy*. The m_p paths are formed by choosing m_p SNs in the first hop and then choosing only one SN in each of the subsequent hops. The source SN will fail to deliver data to the SN if one of the following happens: (a) none of the SNs in the first hop receives the message; (b) in the first hop, i $(1 \leq i < m_p)$ SNs receive the message, and each of them attempts to form a path for data delivery; however, all i paths fail to deliver the message because the subsequent hops fail to receive the broadcast message; or (c) in the first hop, at least m_p SNs receive the message from the source SN from which m_p SNs are randomly selected to forward data, but all m_p paths fail to deliver the message because the subsequent hops fail to receive the message. Summarizing above, the probability of a source SN (with index j=1) failing to deliver data to the BS through multipath routing is given by:

$$Q_1^{SN}(p) = 1 - \theta_1^{SN}(1) + \sum_{p=1}^{m_p} \theta_1^{SN}(p) \left[1 - \varphi_2^{SN}\right]^p \quad (6.16)$$

Consequently, the failure probability of data delivery to the BS from m_s source SNs, each utilizing m_p paths to relay data, is given by:

$$Q_f = \left[1 - (1 - Q_1^{SN})\left(1 - Q_{c,1(x(1),t)}^{SN}\right)\right]^{m_s} \quad (6.17)$$

Therefore, the query success probability is given by:

$$R_q = 1 - Q_f \tag{6.18}$$

Note that in the above derivation we omit time for brevity. More precisely, R_q derived above should be $R_q(t_{Q,i})$ since the query success probability is a function of time, depending on the node count and population density at the i^{th} query's execution time (i.e., at time $t_{Q,i}$).

6.3.7 Energy Consumption

Now we estimate the amounts of energy spent by a SN located at distance x away from the BS during a query interval $[t_{Q,i}, t_{Q,i+1}]$ and an IDS interval $[t_{I,i}, t_{I,i+1}]$ so as to estimate N_q, the maximum number of queries this SN can possible handle before running into energy exhaustion. When all SNs at distance x consumes all their energy, a 'black ring' at distance x away from the BS is formed. Nodes at distance greater than x will have to increase their radio range in order to maintain connectivity with the BS but eventually the system ceases to function. When selective capture is in effect, one can see that a black ring can more easily develop for nodes close to the BS.

To normalize energy consumption over N_q queries, let α be the ratio of the IDS execution rate to the query arrival rate so that αN_q is the numbers of IDS cycles before SN energy exhaustion. Then, we can estimate N_q by the fact that the SN energy consumed due to intrusion detection, and query processing is equal to the initial SN energy as follows:

$$E_o^{SN} = \sum_{i=1}^{N_q} E_q^{SN}(x, t_{Q,i}) + \sum_{i=1}^{\alpha N_q} E_{IDS}^{SN}(x, t_{I,i}) \tag{6.19}$$

Below we outline how to calculate $E_q^{SN}(x, t_{Q,i})$ and $E_{IDS}^{SN}(x, t_{I,i})$. We first estimate energy consumed by transmission and reception over wireless link. The energy spent by a SN to transmit an encrypted data packet of length n_b bits over a distance r is estimated as [126]:

$$E_T^{SN}(r) = n_b\left(E_{elec} + E_{amp} r^z\right) \tag{6.20}$$

Here E_{elec} is the energy dissipated to run the transmitter and receiver circuitry, E_{amp} is the energy used by the transmit amplifier, and r is the transmission radio range. We use the current SN radio range to derive E_T^{SN}. We set E_{amp}= 10 pJ/bit/m^2 and $z = 2$ when the

radio range is less than a threshold distance d_0 (75m) and E_{amp}= 0.0013 pJ/bit/m^4 and $z = 4$ otherwise[126]. The energy spent by a node to receive an encrypted message of length n_b bits is given by:

$$E_R^{SN} = n_b E_{elec} \tag{6.21}$$

The energy consumed by a SN located at x for processing the i^{th} query, $E_q^{SN}(x, t_{Q,i})$, conditioning on it is being a data delivery path with probability $P_q^{SN}(x, t_{Q,i})$is the energy consumed for reception (except when it is a source SN) and transmission, i.e.,

$$E_q^{SN}(x, t_{Q,i}) = P_q^{SN}(x, t_{Q,i}) \times [E_R^{SN} + E_T^{SN}(r^{SN}(x, t_{Q,i}))] \tag{6.22}$$

Since source SNs are randomly picked to answer a query, the probability that a SN at distance x away from the BS is on the data path $P_q^{SN}(x, t_{Q,i})$ is estimated as the probability of a SN at x is needed for data delivery, $(r^{BS} - x)/r^{BS}$, multiplied with the probability that this particular sensor is needed, $m_p m_s / N^{SN}(x, t_{Q,i})$. Here $N^{SN}(x, t_{Q,i})$ calculated by $\int_{x-r^{SN}(x,t_{Q,i})}^{x+r^{SN}(x,t_{Q,i})} \lambda^{SN}(X,t)\ 2\pi X\, dX$ is the total number of SNs within the radio range of SNs at distance x.

For intrusion detection every node is evaluated by m voters in an IDS cycle, and each voter sends its vote to the other m - 1 voters. Hence, the energy spent by a SN located at x in the i^{th} IDS cycle, $E_{IDS}^{SN}(x, t_{I,i})$, conditioning on it serving as a voter with probability $P_{IDS}^{SN}(x, t_{I,i})$ for each of its $n^{SN}(x, t_{I,i})$ neighbors is the energy consumed for reception of m-1 votes and transmission of its vote to other m-1 voters, i.e.,

$$E_{IDS}^{SN}(x, t_{I,i}) = P_{IDS}^{SN}(x, t_{I,i}) \times n^{SN}(x, t_{I,i}) \times (m-1)[E_R + E_T^{SN}\left(r^{SN}(x, t_{Q,i})\right)] \tag{6.23}$$

Here the probability that a SN at distance x serves as a voter for a neighbor SN, $P_q^{SN}(x, t_{Q,i})$, is estimated as $m/n^{SN}(x, t_{I,i})$.

The system fails when a SN at distance $x = r_{max}^{SN}$ (SN maximum radio range) depletes its energy since there is no way to maintain connectivity even by dynamic range adjustment. That is, we set $x = r_{max}^{SN}$to obtain $E_q(t_{Q,i})$, and $E_{IDS}(t_{I,i})$ from Equation (6.22), and Equation (6.23), respectively, and then we calculate N_q from Equation (6.19). The knowledge of N_q along with $R_q(t_{Q,i})$ in Equation (6.18) allows us to calculate the system MTTF given by Equation (6.2).

6.4 Performance Evaluation

Table 6-2: Input Parameter Values Characterizing a BS-Based WSN.

Parameter	Default Value	Parameter	Default Value
N_{SN}	1500	r^{BS}	300m
λ_0^{SN}	1500 nodes/($\pi\times300^2$)	n_b	50 bits
E_0^{SN}	2 Joules	E_{elec}	50 nJ/bit
p_{err}	0.01	E_{amp}	10 pJ/bit/m^2
p_a	0.0-1.0	d_0	75m
p_{all-in}	0.0-0.3	f	¼
λ_c^{min}	28days	λ_q	10^{-2} to 1 query/sec
λ_c^{max}	0.5-14days	n_0	7

In this section, we present numerical results. Our reference WSN consists of N_o^{SN}= 1500 SN nodes initially deployed with density $\lambda_o^{SN}$with the BS sitting at the center of a circular area with radius r^{BS}=300m. The selective SN capture time is assumed to be exponentially distributed following the linear model described by Equation (6.1), with λ_c^{min} being once per 4 weeks and λ_c^{max} varying in the range of once per half day to once per 2 weeks. The radio range r_{SN} is dynamically adjusted to maintain network connectivity of n_0= 7 to support basic multipath routing and voting-based IDS functions. The initial energy level of a SN is $E_0^{SN} = 2$ Joule. The energy parameters used by the radio module are adopted from [68, 126]. The energy dissipation E_{elec} to run the transmitter and receiver circuitry is 50 nJ/bit. The energy used by the transmit amplifier to achieve an acceptable signal to noise ratio (E_{amp}) is 10 pJ/bit/m^2 for transmitted distances less than the threshold distance d_0 (75m) and 0.0013 pJ/bit/m^4 otherwise. The query arrival rate λ_q is a variable ranging from 10^{-2} to 1 query/sec to reveal points of interest. The imperfection of host IDS monitoring due to ambient noise and channel error is modeled by a monitoring error probability p_{err}= 1%.

6.4.1 Analyzing the Effect of Selective Capture on MTTF

In this section we analyze the effects of selective capture on MTTF. To isolate out the effect of smart attack (which we will consider later), we only consider persistent attackers that always attack with probability 1 ($p_a = 1.0$). Our objective is to identify the best protocol setting of our countermeasures against selective capture which converts good nodes into bad nodes. This includes the radio range to be adjusted dynamically by indi-

vidual SNs, the best redundancy level used for multipath routing, as well as the best redundancy level in terms of the number of voters and the best intrusion invocation interval used for intrusion detection to maximize the WSN lifetime. We use random capture as a baseline case for performance comparison.

6.4.1.1 Countermeasures against Selective Capture

We first examine the effect of the compromise rate and intrusion detection interval T_{IDS} on the optimal (m_p, m_s) to maximize the WSN lifetime. Tables I and II summarize the optimal (m_p, m_s) values to maximize the WSN lifetime under selective capture and random capture attacks, respectively, with m fixed at 3 (i.e., the number of voters is 3). The maximum compromise rate λ_c^{max} on the 1st column specifies the magnitude of the compromise rate (with λ_c^{min} being fixed at once per 4 weeks). The intrusion detection interval T_{IDS} on the 1st row specifies the IDS interval. For example, when λ_c^{max} =1/(0.5 day) and T_{IDS}=4hrs, the optimal (m_p, m_s) is (2,5) under selective capture in Table 1 and is (1,5) under random capture in Table 2.

We first observe that there exists an optimal (m_p, m_s) setting under which the MTTF is maximized for either case. Furthermore, a higher (m_p, m_s) is needed when the capture strength λ_c^{max} increases. Also under selective capture attacks, the system must use a higher redundancy level to maximize the MTTF. For example when T_{IDS} = 4 hrs and λ_c^{max}= 1/(0.5 day), the optimal (m_p, m_s) setting is (2, 5) under selective capture (in Table 6-3) but is only (1, 5) under random capture attacks (in Table 6-4). This is because selective capture requires the system to apply more redundancy to cope with more critical nodes being compromised. The system is better off in this case to use higher redundancy to ensure secure routing at the expense of more energy consumption to maximize the system MTTF.

We also observe from Table 6-3 and Table 6-4 that the optimal MTTF is more likely to be achieved using a redundancy setting with a high m_s as opposed to a high m_p. While the same number of total paths can be achieved using various (m_p, m_s) combinations, e.g., 6 paths can be achieved by (1, 6), (2, 3), (3, 2) or (6, 1), increasing m_s rather than increasing m_p can more effectively increase the query success probability because the failure of a single source SN results in a system failure, even if the source SN is connected to the BS via m_p paths. On the other hand, the failure of a single path is less damaging to query success. Furthermore, we expect little difference in terms of energy consumption when the number of paths is the same. As a result, the optimal (m_p, m_s) setting favors a high m_s over a high m_p whenever possible.

Table 6-3: Optimal (m_p, m_s) under Selective Capture with varying λ_c^{max} and T_{IDS} Values.

λ_c^{max}	T_{IDS}=1hr	2hrs	4hrs	6hrs	8hrs
1/(8 hrs)	(1,3)	(1,4)	(5,5)	(5,5)	(5,5)
1/(0.5 day)	(1,2)	(1,4)	(2,5)	(2,5)	(2,5)
1/(0.75 days)	(1,2)	(1,2)	(1,4)	(2,5)	(2,5)
1/day	(1,2)	(1,2)	(1,2)	(1,4)	(2,5)
1/(2 days)	(1,2)	(1,2)	(1,2)	(1,2)	(1,2)

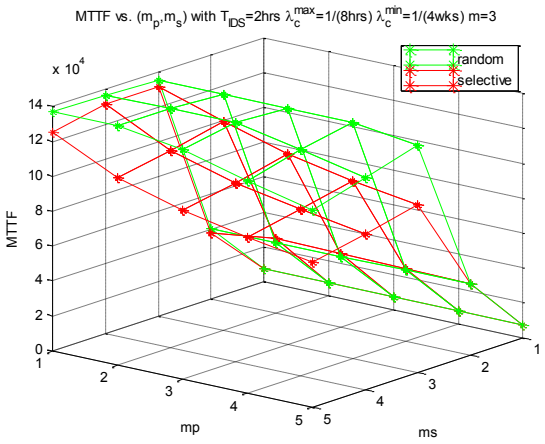
Table 6-4: Optimal (m_p, m_s) under Random Capture with varying λ_c^{max} and T_{IDS} Values.

λ_c^{max}	T_{IDS}=1hr	2hrs	4hrs	6hrs	8hrs
1/(8 hrs)	(1,3)	(1,4)	(1,5)	(5,5)	(5,5)
1/(0.5 day)	(1,2)	(1,2)	(1,5)	(1,5)	(2,5)
1/(0.75 days)	(1,2)	(1,2)	(1,2)	(1,5)	(1,5)
1/day	(1,2)	(1,2)	(1,2)	(1,2)	(1,5)
1/(2 days)	(1,2)	(1,2)	(1,2)	(1,2)	(1,2)

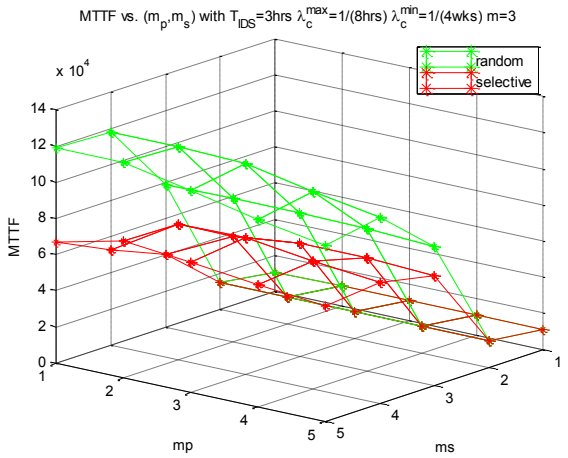
We next analyze the effect of the intrusion detection interval T_{IDS} (representing the intrusion detection strength) on the system MTTF. Whether to use a small or large T_{IDS} depends on the capture strength λ_c^{max}. When the capture strength is high (i.e., when λ_c^{max} is high), as evidenced by the frequency at which bad nodes are detected by the IDS and evicted, we must counter it with high detection strength (a small T_{IDS}).

Conversely, when the capture strength is low, a large T_{IDS}could be used to save energy to maximize the MTTF.

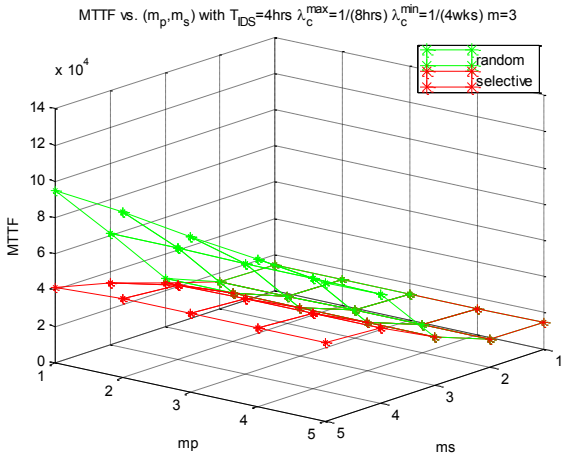
Figure 6-2(a), Figure 6-2 (b), and Figure 6-2 (c) show MTTF vs. (m_p, m_s) under small (T_{IDS}=2 hrs), medium (T_{IDS}=3 hrs), and large (T_{IDS}=4 hrs) detection intervals, respectively, for the case when the capture strength is high, i.e., $\lambda_c^{max} = 1/(8\text{ hrs})$. We again set m=3 to isolate its effect. We observe that at the optimal (m_p, m_s) setting, the MTTF under T_{IDS}=2 hrs (Figure 6-2 (a)) is much higher than the MTTF under T_{IDS}=4 hrs (Figure 6-2 (c)). This is because when the system is subject to a high capture rate, the system is better off to apply high detection strength (a small T_{IDS} at 2 hrs) at the expense of more energy consumption to quickly detect and evict compromised nodes, instead of applying low detection strength (a large T_{IDS} at 4 hrs), so as to increase the MTTF. This trend applies to both selective and random capture attacks. We also see that the MTTF under selective capture is much lower than that under random capture because with selective capture critical nodes are more easily compromised and black holes can more easily form near the BS to cause a system failure. Lastly we note that optimal (m_p, m_s) setting is highly situation dependent. The optimal (m_p, m_s) settings under random capture and selective capture in Figure 6-2 (a), Figure 6-2 (b) and Figure 6-2 (c) are (1,4)(1,4), (1,5)(2,5), and (1,5)(5,5) respectively.



(a) High detection strength ($\boldsymbol{T_{IDS}}$= 2hr)



(b) Medium Detection strength ($\boldsymbol{T_{IDS}}$= 3hrs)



(c) Low detection strength ($\boldsymbol{T_{IDS}}$= 4hrs)

Figure 6-2: MTTF vs. ($\boldsymbol{m_p, m_s}$) with varying Detection Strength in the presence of High Capture Strength.

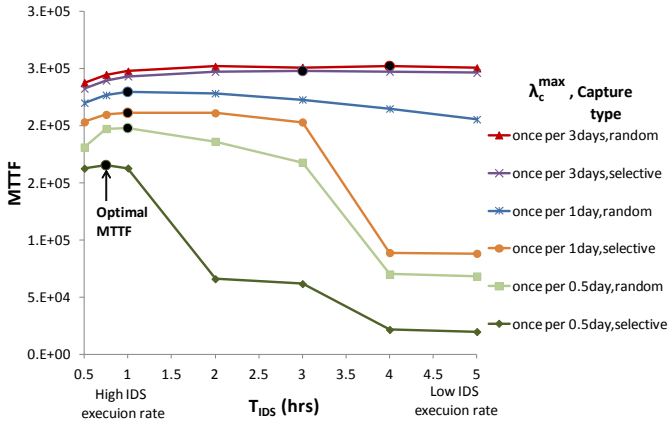


Figure 6-3: Effect of T_{IDS} on System MTTF under Random Capture vs. under Selective Capture.

Figure 6-3 compares the effect of T_{IDS} on the MTTF under random capture vs. selective capture at the optimal (m_p, m_s) setting under random capture vs. selective capture. We again observe that there exists an optimal T_{IDS}value (marked by a black dot) at which the MTTF is maximized. Furthermore, the optimal T_{IDS}value under selective capture in general is smaller than that under random capture because the system has to increase detection strength to cope with selective capture which creates more compromised critical nodes.

In Figure 6-4 we summarize the damaging effect of selective capture attacks compared with random capture attacks. It shows that selective capture has a devastating effect on the MTTF compared with random capture. The effect is especially pronounced when the capture strength λ_c^{max} is high (left end of the graph). The MTTF at the optimal (m_p, m_s) setting under selective capture is relatively low compared with that under random capture because the success probability for a node to transmit a packet to at least p next-hop SN neighbors (Equation (6.13)) is low as the node is close to the BS, as many critical nodes are compromised due to selective capture.

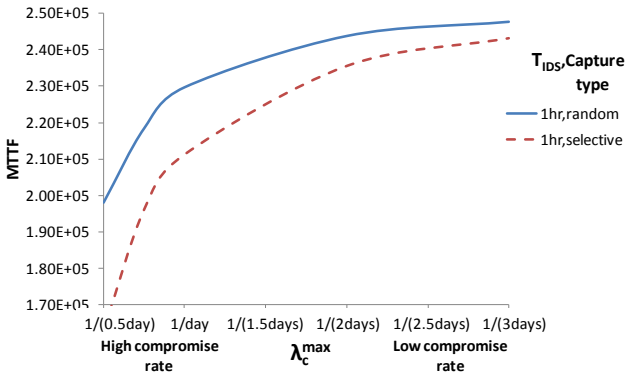


Figure 6-4: WSN lifetime under Random Capture vs. under Selective Capture with varying λ_c^{max}.

Figure 6-5 vividly displays how the "good SN density" $\lambda_{good}^{SN}(x,t)$ evolves over time under selective capture vs. under random capture. It confirms that $\lambda_{good}^{SN}(x,t)$ decreases over time because of capture, and the rate at which $\lambda_{good}^{SN}(x,t)$ declines for SNs with $x <$ 1/2 under selective capture is higher than that under random capture. The effect of selective capture on good node population is especially pronounced for critical nodes near the BS (i.e., when x=1/16 or 1/8).

Figure 6-6 displays how a SN at distance x dynamically adjusts its radio range to counter selective capture so as to maintain sufficient network connectivity and improve packet delivery reliability. It confirms that with the "dynamic radio range adjustment" countermeasure, a SN increases its radio range over time to maintain network connectivity. Further, under selective capture because critical nodes (i.e., when x is small) are more likely compromised, and subsequently detected and evicted from the system, a critical node must increase its radio range more rapidly to maintain network connectivity and improve packet delivery reliability to effectively counter selective capture. Figure 6-6 demonstrates that critical SNs (e.g., when x=1/16 or 1/8) are able to more rapidly adjust radio range to maximize the system MTTF.

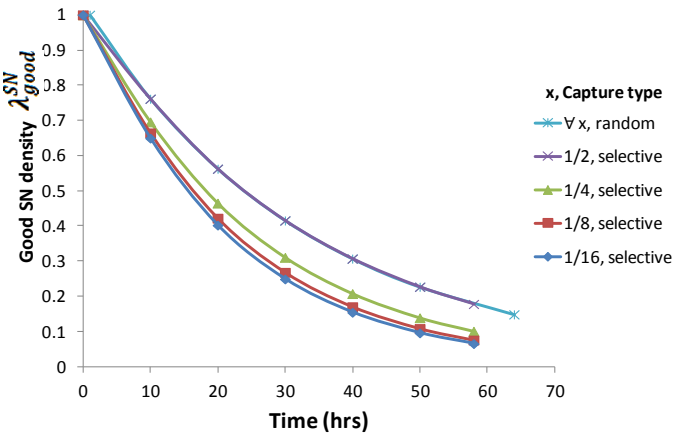


Figure 6-5: Density of Good SNs at Distance *x* vs. Time.

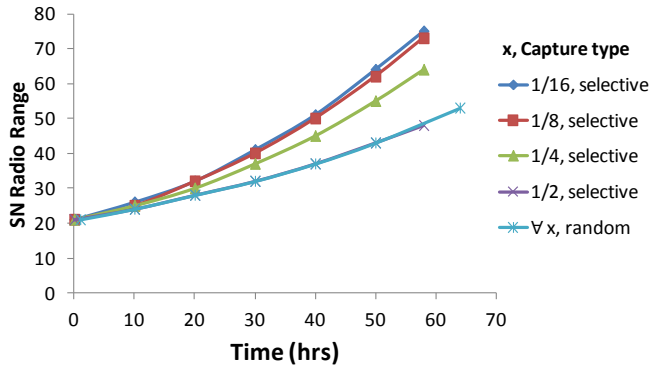
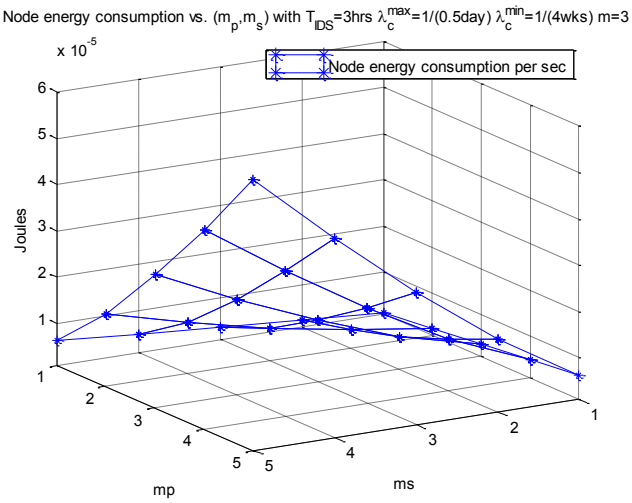


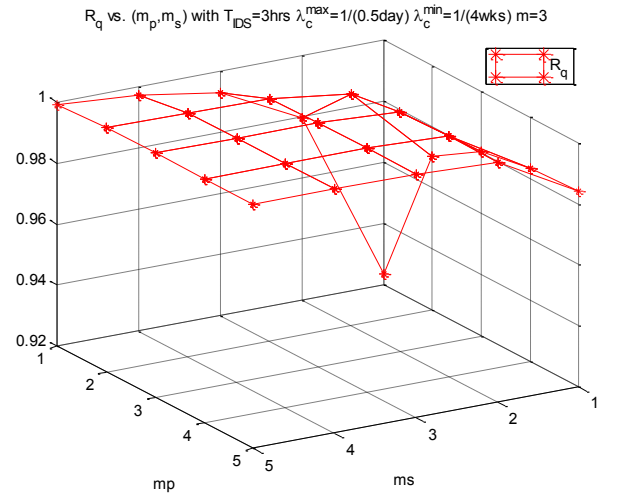
Figure 6-6: Adjusting Radio Range at Distance *x* vs. Time.

6.4.1.2 Tradeoff Analysis between Energy and Reliability

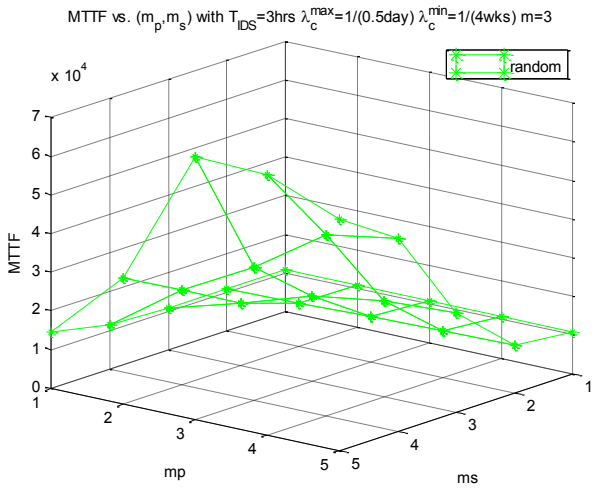
The optimal (m_p, m_s) setting identified to maximize MTTF is a result of the tradeoff between query success probability Equation (6.18) vs. energy consumption per second Equation (6.19). This is illustrated in Figure 6-7 for a selective capture case where Figure 6-7(a) shows the average node energy consumption rate vs. (m_p, m_s) (for both query processing and IDS execution), Figure 6-7 (b) correspondingly shows the query success probability (R_q) vs. (m_p, m_s), and Figure 6-7(c) shows the MTTF vs. (m_p, m_s) as a result of this tradeoff. Here we take data collected over the lifetime for Figure 6-7 (a) and Figure 6-7 (b), since the first two quantities are time dependent. While the query success probability is maximized with (m_p, m_s)=(5,5), this setting consumes the most energy which adversely shortens the system lifetime. As a result, the optimal (m_p, m_s) setting that maximizes MTTF in this example is (3,5).



(a) Node energy consumption rate vs. (m_p, m_s).



(b) Query success probability vs. (m_p, m_s).



(c) MTTF vs. (m_p, m_s).

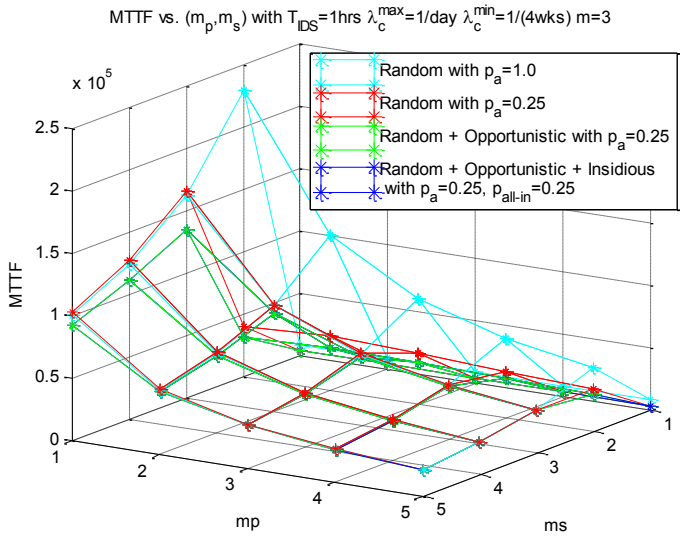
Figure 6-7: An Example showing (a) Node Energy Consumption per second, (b) Query Success Probability, and (c) MTTF vs. (m_p, m_s).

6.4.2 Analyzing the Effect of Smart attack on MTTF

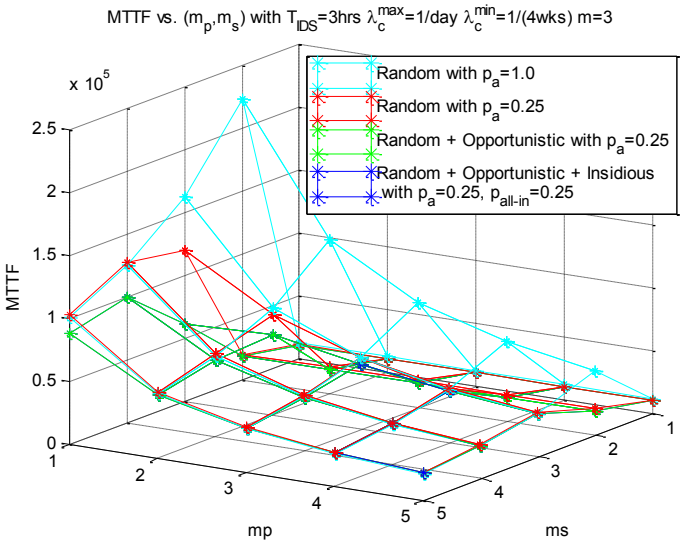
In this section, we analyze the effect of smart attack on MTTF. We consider 3 smart attack behavior models: (1) random, (2) random + opportunistic, and (3) random + opportunistic + insidious. We use persistent attacks as the baseline case for performance comparison. Also for each case, we differentiate selective capture from random capture as smart attack will very likely exploit environment vulnerability due to selective capture which creates more malicious nodes in areas nearer to the BS.

6.4.2.1 Smart attack under Random Capture

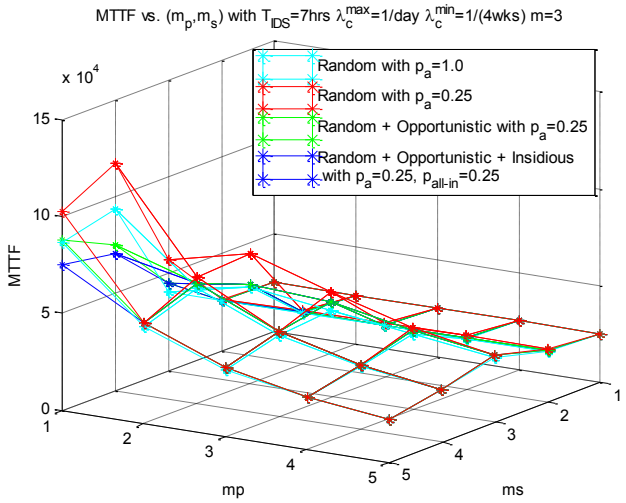
Figure 6-8(a)-(c) show MTTF vs. (m_p, m_s) for a WSN under smart attack and random capture when the detection strength goes from high to low (T_{IDS}= 1hr, 3hrs to 7hrs). For random attacks, $p_a = 0.25$. For insidious attacks, $p_{all-in} = 0.25$. There are several general observations. First, we observe that there exists an optimal (m_p, m_s) setting under which the MTTF is maximized for each attack behavior model. Further, more damaging attacks in general require higher redundancy to achieve the optimal (m_p, m_s) setting. For example, in Figure 6-8(a), the optimal (m_p, m_s) settings for the persistent attack model ($P_a = 1.0$) and the random + opportunistic + insidious attack model are (1,2) and (1,3), respectively.



(a) High detection strength ($\boldsymbol{T_{IDS}}$= 1hr)



(b) Medium Detection strength ($\boldsymbol{T_{IDS}}$= 3hrs)



(c) Low detection strength ($\boldsymbol{T_{IDS}}$= 7hrs)

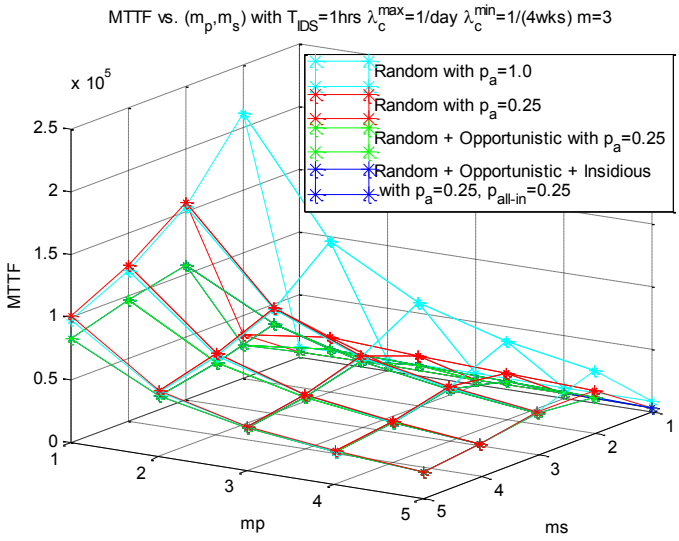
Figure 6-8: MTTF vs. (m_p, m_s) under Smart attack and Random Capture.

Second, we observe that smart attack that consider all strategies, i.e., random + opportunistic + insidious, are more damaging than those using random + opportunistic or just random attacks, because given the same level of hiddenness as provided by random attacks, opportunistic and insidious attacks are "sure" and effective attacks.

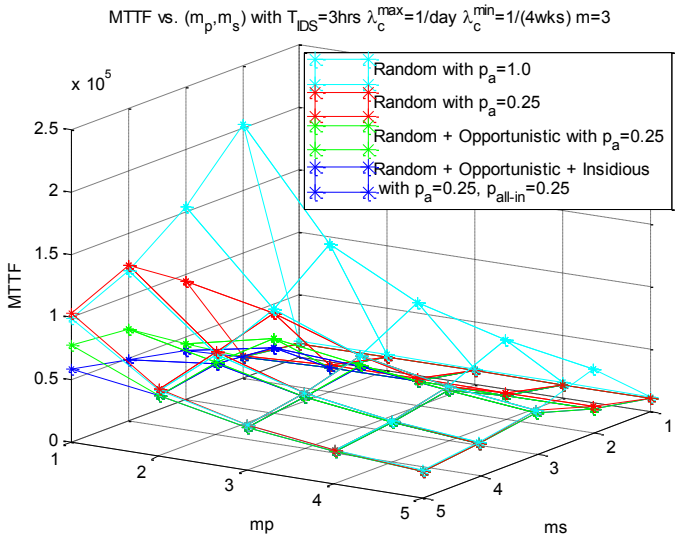
Third, from the smart attacker's perspective, the best attack strategy depends on the defender's detection strength. We observe that when the detection strength is high (Figure 6-8 (a)) the best adversary strategy is to attack randomly with a low random attack probability so as to evade detection and wait opportunistically for the best chance to come by to attack. This is observed in Figure 6-8 (a) where we see that in this case attacking persistently with $p_a = 1$ (the top curve) results in the attackers being detected and removed from the system, and consequently yields the highest MTTF among all cases. Conversely, when the detection strength is low (Figure 6-8 (c)) the best adversary strategy is to attack with a high random attack probability to maximize the damage. This is observe in Figure 6-8 (c) where we see that attacking with a low random attack probability $p_a = 0.25$ (the top curve) actually results in the highest MTTF among all, since in this case intrusion detection is ineffective; thus, attacking persistently (with $p_a = 1$) is more effective than random attacks with a low random attack probability $p_a = 0.25$.

Lastly, we observe that the effect of insidious attacks is pronounced when the detection strength is low. In Figure 6-8 (a) where the detection strength is high, its effect is insignificant. This is evidenced in Figure 6-8 (a) that MTTF under the random + opportunistic + insidious attack model is almost the same as that under the random + opportunistic attack model (the bottom curve). However, when the detection strength de-

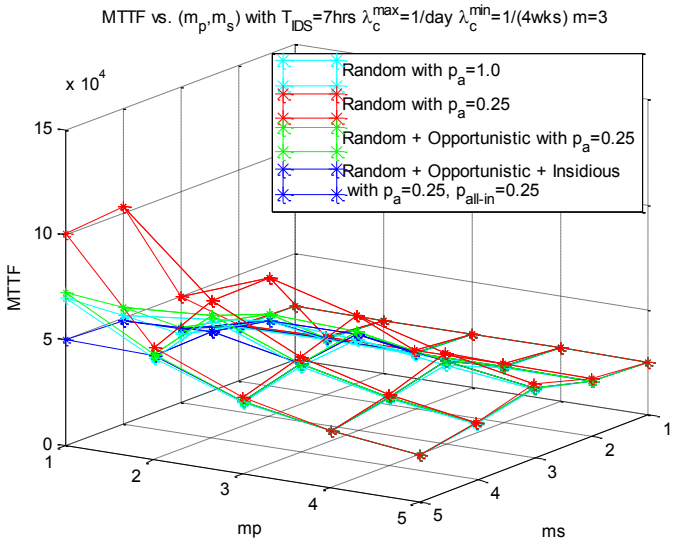
creases, its effect becomes more manifested because of a much higher chance of many malicious nodes accumulated in the system due to weak detection. This is evidenced in Figure 6-8 (c) that MTTF under the random + opportunistic + insidious attack model (the bottom curve) is much lower than that under the random + opportunistic attack model (the 3rd bottom curve).



(a) High detection strength ($\boldsymbol{T_{IDS}}$= 1hr)



(b) Medium detection strength ($\boldsymbol{T_{IDS}}$= 3hrs)



(c) Low detection strength ($\boldsymbol{T_{IDS}}$= 7hrs)

Figure 6-9: MTTF vs. (m_p, m_s) under Smart attack and Selective Capture.

6.4.2.2 Smart attack under Selective Capture

We repeat the same analysis as Section 6.4.2.1 above except that we analyze the effect of smart attack on MTTF of a WSN with selective capture. We summarize the results by Figure 6-9(a)-(c) showing MTTF vs. (m_p, m_s) under smart attack and selective capture, with the detection strength going from high to low (T_{IDS}= 1hr, 3hrs to 7hrs), respectively. Comparing Figure 6-8 and Figure 6-9, we see that the same general observations drawn from Figure 6-8 in Section V-B-(a) apply. However, there are two striking differences. First, the MTTF value obtained, given the same detection strength and attack behavior model under selective capture, is lower than that under random capture. Second, as the intrusion detection strength decreases, the difference in MTTF between random capture and selective capture widens (e.g., Figure 6-8(c) and Figure 6-9(c)) because of the damaging effect of selective capture which causes more malicious nodes accumulated near the BS.

We further compare selective capture with random capture to examine the environment vulnerability created by selective capture. Figure 6-10 and Figure 6-11 compare MTTF obtainable under selective capture vs. that under random capture for high and low detection strengths, respectively, using the random + opportunistic + insidious attack behavior model as the test case. The results demonstrate that selective capture has a far more damaging effect on MTTF because of its ability to create malicious nodes near the BS. The effect is especially pronounced when the detection strength is low. Table 6-5 and Table 6-6 summarize the optimal (m_p,m_s) setting to maximize MTTF obtain-

able under selective capture and random capture, respectively, with random + opportunistic + insidious attacks. We see that the system will have use higher redundancy in terms of (m_p, m_s) to cope with selective capture, when compared with random capture.

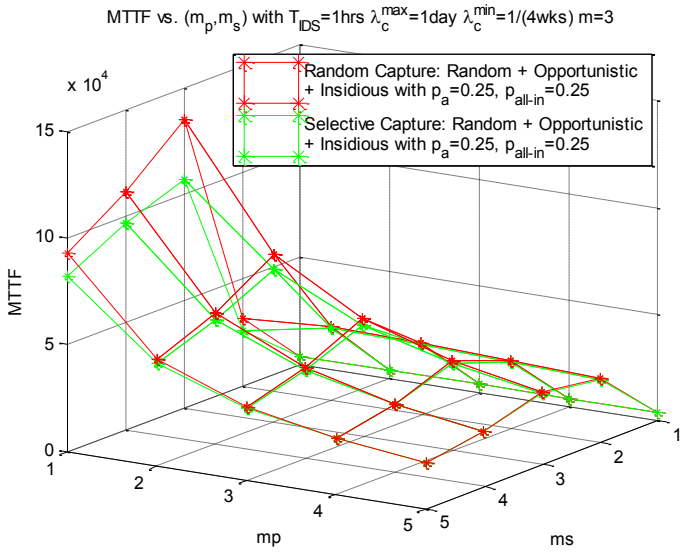


Figure 6-10: Comparing MTTF obtainable under Selective Capture vs. Random Capture with Random + Opportunistic + Insidious Attacks (High Detection Strength).

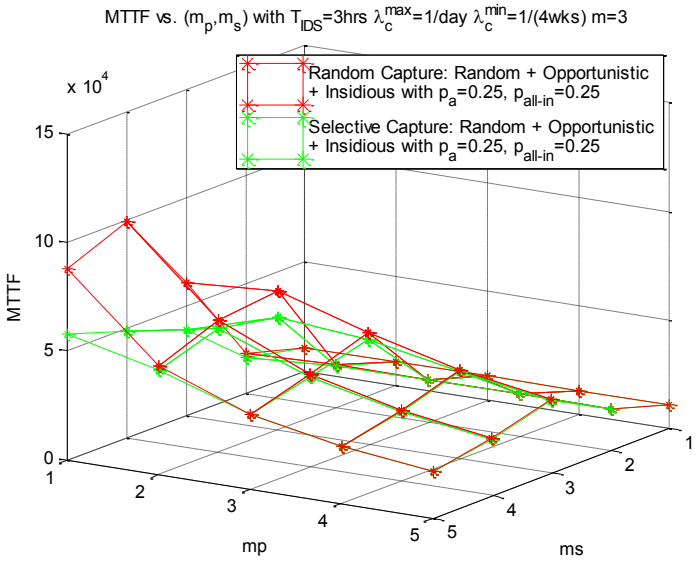


Figure 6-11: Comparing MTTF obtainable under Selective Capture vs. Random Capture with Random + Opportunistic + Insidious Attacks (Low Detection Strength).

Table 6-5: Optimal (m_p, m_s) under selective capture and Random + Opportunistic + Insidious attacker, with varying λ_c^{max} and T_{IDS} .

	T_{IDS}=1hr	3hrs	5hrs
$\lambda_c^{max} = 1/(0.5\text{ day})$	(2,4)	(2,5)	(3,5)
$\lambda_c^{max} = 1/(1\text{ day})$	(1,3)	(1,5)	(2,5)

Table 6-6: Optimal (m_p, m_s) under random capture and Random + Opportunistic + Insidious attacker, with varying λ_c^{max} and T_{IDS} .

	T_{IDS}=1hr	3hrs	5hrs
$\lambda_c^{max} = 1/(0.5\text{ day})$	(1,4)	(2,5)	(2,5)
$\lambda_c^{max} = 1/(1\text{ day})$	(1,3)	(1,4)	(1,5)

6.4.2.3 Analyzing Intrusion Detection Latency

In this section we analyze intrusion detection latency, i.e. how long it takes for our voting-based IDS to detect a compromised node. The detection latency for a node compromised at time t depends on the detection probability $(1 - P_{fn})$ in consecutive intrusion detection intervals past t. The probability that the compromised node is detected at the *ith* intrusion detection interval past time t is given by $\left[1 - P_{fn}(t + i \times T_{IDS})\right] \times \prod_{j=1}^{i-1} P_{fn}(t + j \times T_{IDS})$. This is so because it was not detected in the prior (*i-1*) intervals, the probability of which is $\prod_{j=1}^{i-1} P_{fn}(t + j \times T_{IDS})$ and it is then detected at the *ith* interval, the probability of which is $1 - P_{fn}(t + i \times T_{IDS})$. Hence, the detection latency of a node compromised at time t can be computed by conditioning on the time (i.e., the *ith* IDS interval) at which the compromised node is detected, as follows:

$$DL(t) = \sum_{i=1}^{\frac{MTTF-t}{T_{IDS}}} \left\{ (i \times T_{IDS}) \times \left[1 - P_{fn}(t + i \times T_{IDS})\right] \times \prod_{j=1}^{i-1} P_{fn}(t + j \times T_{IDS}) \right\} \quad (6.24)$$

Since the capture rate of a SN at a distance x away from the BS is $\lambda_c^{SN}(x)$ (defined in Equation 6.1 with t removed as the capture rate is time independent), the probability that a SN at x is compromised at time t is given by $\pi_c^{SN}(x,t) = 1 - e^{-\lambda_c^{SN}(x) \times t}$ assuming exponential distribution. Therefore, the average expected detection latency can be obtained by weighting on the probability of the node being compromised at time t, i.e.,

$$E(DL) = \frac{\sum_{i=1}^{\frac{MTTF}{T_{IDS}}} \{(DL(i \times T_{IDS}) \times \pi_c^{SN}(x,t)\}}{\sum_{i=1}^{\frac{MTTF}{T_{IDS}}} \{\pi_c^{SN}(x,t)\}} \quad (6.25)$$

Equations (6.24) and (6.25) are applied to a compromised SN at distance x from the BS to calculate its expected detection latency. Since the IDS detection probability, i.e., $1 - P_{fn}$, for detecting a compromised SN decreases as x decreases since there are more nodes captured and compromised near the BS due to selective capture, we expect a higher detection latency as x decreases.

Figure 6-12 and Figure 6-13 demonstrate the effect of selective capture on detection latency. The two diagrams display the average detection latency obtained from Equation 6.25 vs. distance from BS (x) under P_a=1.0 and P_a=0.25, respectively. The parameter values are as given in Table 6-2 with T_{IDS} =3hrs and λ_c^{max} =1/(0.5 day). We see clearly that as x decreases (i.e., as a compromised SN is closer to the BS), the detection latency increases, because selective capture populates more bad nodes near the BS, thus affecting the accuracy of voting IDS to detect a bad node. We also observe that the detection latency under random attack behavior (Figure 6-13) is higher than that under persistent attack behavior (Figure 6-12), because the IDS can better detect and thus evict bad nodes that attack all the time. This prevents bad nodes from accumulating in the system and bad-mouthing good nodes in future voting IDS executions.

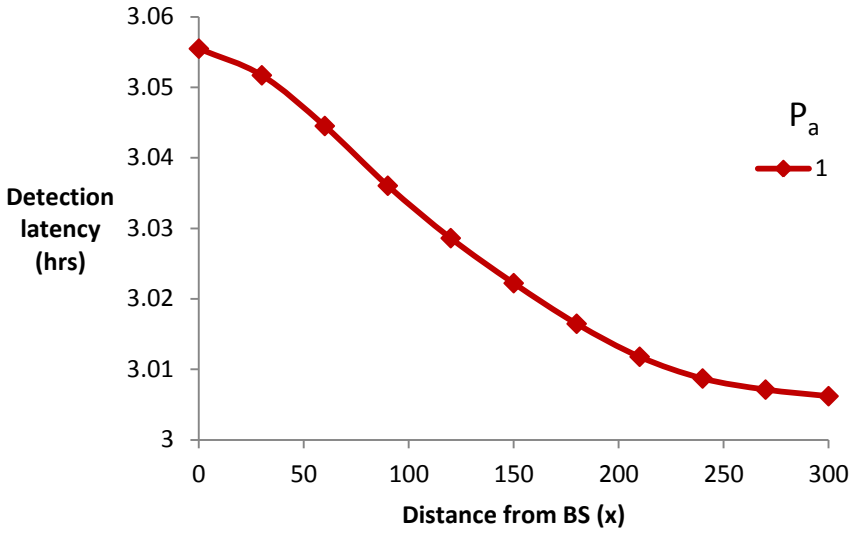


Figure 6-12: Detection Latency vs. Distance from BS under P_a=1.0, T_{IDS}=3hrs and λ_c^{max} =1/(0.5 day).

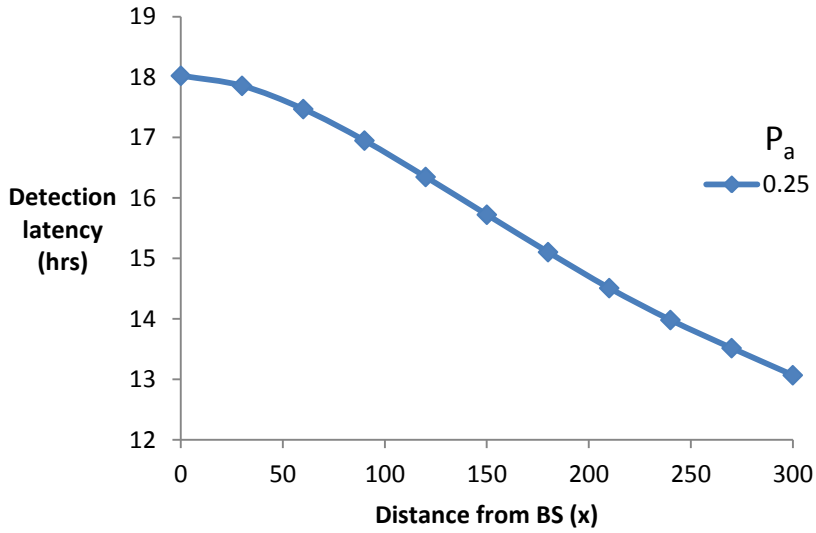


Figure 6-13: Detection Latency vs. Distance from BS under P_a=0.25, T_{IDS}=3hrs and λ_c^{max} =1/(0.5 day).

6.4.2.4 Countering Selective Capture and Smart attack

Lastly we analyze the effectiveness of our countermeasures against selective capture with random + opportunistic + insidious attacks which we model by λ_c^{max}, λ_c^{min}, p_a, and p_{all-in}. Figure 6-14 shows the optimal T_{IDS} interval (representing detection strength) to counter such attackers with varying capture strengths (λ_c^{max}) and random attack probability (p_a). In Figure 6-14, we fix λ_c^{min} to once per 4 weeks and vary λ_c^{max} in the range of once per half day to once per 2 weeks. We also fix $p_{all-in} = 0.25$ to isolate its effect.

We observe from Figure 6-14 that a low p_a demands a high detection rate (i.e., a small T_{IDS} interval). The reason is that a low p_a will result in a high per-host false negative probability H_{pfn}. Consequently, to cope with many hidden bad nodes missed by intrusion detection, the system will have to use a small T_{IDS} interval for high detection strength. Another observation is that as λ_c^{max} increases, T_{IDS} decreases (or the detection strength increases) to counter the increasing capture rate.

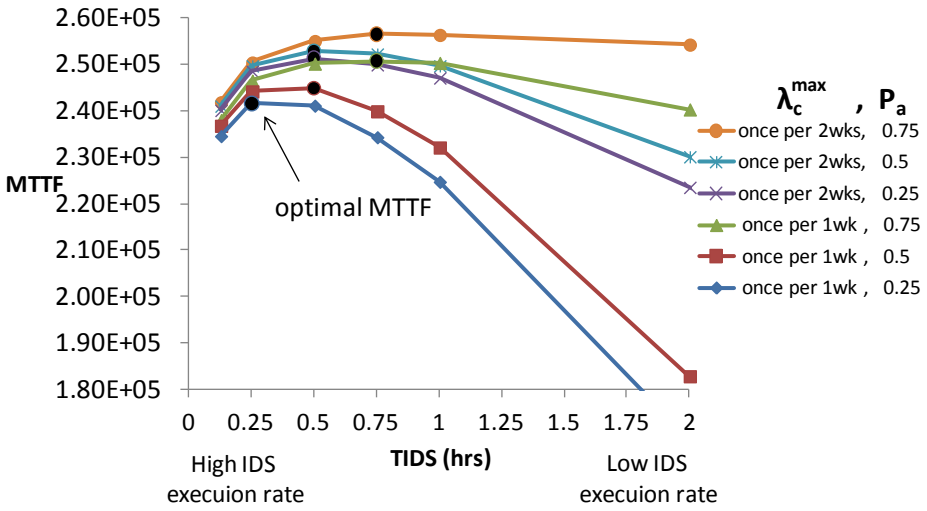


Figure 6-14: Countering Selective Capture with Random + Opportunistic + Insidious Attacks with varying λ_c^{max} and P_a by Adjusting Detection Strength Parameter T_{IDS}.

Table 6-7: Optimal (m_p, m_s) under low P_a (0.25), with varying λ_c^{max} and T_{IDS}.

	T_{IDS}= 0.25hr	0.5hrs	1hr	2hrs	4hrs
λ_c^{max} = 1/(0.5 day)	(1,3)	(1,4)	(2,4)	(2,5)	(3,5)
λ_c^{max}= 1/(1 day)	(1,3)	(1,3)	(1,3)	(1,5)	(2,4)
λ_c^{max}= 1/(7 days)	(1,2)	(1,2)	(1,2)	(1,3)	(1,3)
λ_c^{max}= 1/(14 days)	(1,2)	(1,2)	(1,2)	(1,2)	(1,3)

Table 6-7 and Table 6-8 summarize the optimal (m_p,m_s) settings (representing multipath multisource redundancy for intrusion tolerance) to maximize MTTF obtainable under selective capture with random + opportunistic + insidious attacks, for low and high $p_a$settings respectively. We observe that the more hidden the inside attackers are, that is, as $p_a$decreases, the more (m_p, m_s) redundancy is required to cope with the bad node population accumulated due to misses in intrusion detection. This is evidenced by comparing optimal (m_p,m_s) settings listed in

Table 6-7 and Table 6-8 for the same $\lambda_c^{max}$and T_{IDS}.

Table 6-8: Optimal (m_p, m_s) under high P_a (0.75) with varying λ_c^{max} and T_{IDS}.

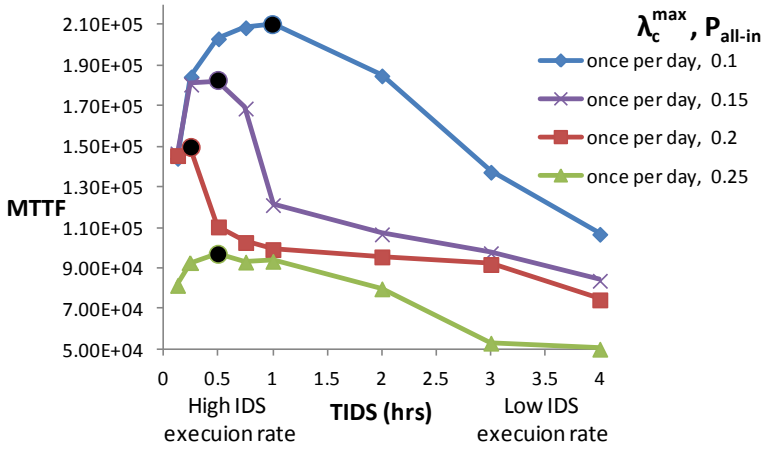
	T_{IDS}= 0.25hr	0.5hrs	1hr	2hrs	4hrs
λ_c^{max}= 1/(0.5 day)	(1,3)	(1,3)	(1,3)	(1,5)	(2,5)
λ_c^{max}= 1/(1 day)	(1,2)	(1,2)	(1,3)	(1,3)	(1,5)
λ_c^{max}= 1/(7 days)	(1,2)	(1,2)	(1,2)	(1,2)	(1,2)

λ_c^{max}= 1/(14 days)	(1,2)	(1,2)	(1,2)	(1,2)	(1,2)

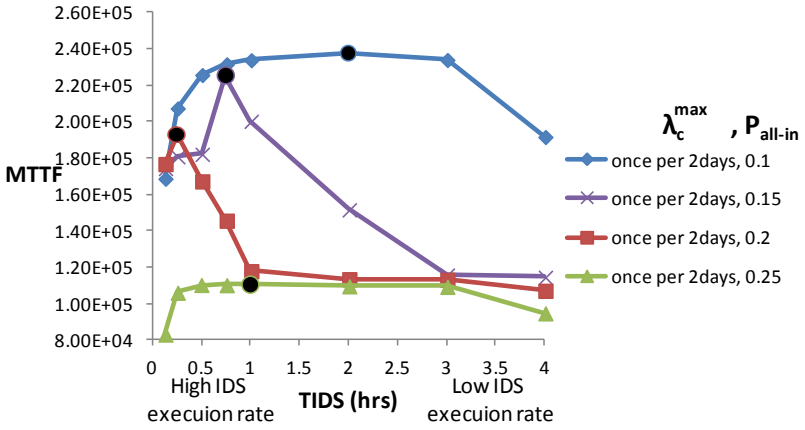
Lastly we analyze the effect of p_{all-in}(the all-in attack percentage population threshold) on MTTF of a WSN subject to selective capture with random + opportunistic + insidious attacks. We first note that a small p_{all-in} means that malicious nodes will perform all-in attacks early on (i.e., setting p_a=1) as soon as it senses the small percentage population threshold is reached. On the other hand, a large p_{all-in}means that malicious nodes will stay hidden until the large percentage population threshold is reached.

Figure 6-15 shows the optimal T_{IDS} interval (representing detection strength) to counter selective capture with random + opportunistic + insidious attacks with varying capture strengths (λ_c^{max}) and all-in attack percentage population threshold (p_{all-in}). We fix $p_a = 0.1$ to isolate out its effect. Figure 6-15 (a) is for the case in which $\lambda_c^{max} =$ once per day, while Figure 6-15 (b) is for the case in which $\lambda_c^{max} =$ once per 2 days. These two diagrams exhibit the same trend, namely, as p_{all-in}initially increases, the optimal T_{IDS} setting (at which MTTF is maximized) decreases. However, as p_{all-in}continues to increase past a high threshold, the optimal T_{IDS} setting increases again.

Table 6-9 summarizes this trend, covering a wider range of λ_c^{max}values. This seemingly odd trend has a logical explanation. That is, when $p_{all-in}$is very small (say 0-10%), $p_a$will be set to 1 early on for all-in attacks, so malicious nodes can be easily detected and the system should just use moderate detection strength to balance energy consumption with detection rate. As p_{all-in}increases further (say 10-20%), malicious nodes stay hidden until the all-in attack percentage population threshold is reached, so the system should use high detection strength to remove malicious nodes to prevent a critical mass from being formed. Finally as p_{all-in}continues to increase past a high threshold (say >25%), insidious attacks will be ineffective since it is unlikely such a high critical mass can be reached. The system in this case is better off using just moderate detection strength to balance energy consumption with detection rate.



(a) High compromise rate (λ_c^{max}= once per day).



(b) Lower compromise rate (λ_c^{max}= once per 2 days).

Figure 6-15: **Countering Selective Capture with Random + Opportunistic + Insidious Attacks with varying λ_c^{max} and $P_{all\text{-}in}$ by Adjusting Detection Strength Parameter T_{IDS}.**

Table 6-9: Optimal T_{IDS} under varying λ_c^{max} and $P_{all\text{-}in}$.

	p_{all-in} =0.0	0.1	0.15	0.2	0.25
λ_c^{max}= 1/(0.5 day)	0.75 hrs	0.75	0.25	0.125	0.5

λ_c^{max}= 1/(1 day)	1	1	0.5	0.25	0.5
λ_c^{max}= 1/(2 days)	2	2	0.75	0.25	1
λ_c^{max}= 1/(3 days)	3	3	0.5	0.25	3

Table 6-10: Optimal (m_p, m_s) under varying λ_c^{max} and $P_{all\text{-}in}$ with fixed T_{IDS} (0.5hr)

	p_{all-in} =0.0	0.1	0.2	0.3
λ_c^{max} = 1/(0.5 day)	(1,2)	(1,3)	(1,3)	(2,4)
λ_c^{max}= 1/(1 day)	(1,2)	(1,2)	(1,3)	(1,5)
λ_c^{max}= 1/(2 days)	(1,2)	(1,2)	(1,2)	(2,2)
λ_c^{max}= 1/(3 days)	(1,2)	(1,2)	(1,2)	(2,2)

Finally, Table 6-10 summarizes the optimal (m_p,m_s) settings (representing multipath multisource redundancy for intrusion tolerance) to maximize MTTF obtainable under varying p_{all-in}settings. We fix T_{IDS} to 0.5 hrs to isolate its effect. We can see from Table VIII that as p_{all-in} increases attackers become more hidden, and, consequently, a higher level of (m_p,m_s) redundancy is required to cope with the bad node population accumulated due to misses in intrusion detection. This finding is consistent with that drawn from Table 6-7 and Table 6-8.

6.5 Simulation

In this section, we conduct a simulation study to support the trends predicted by the analytical model. We use the ns-3 discrete-event network simulator [69] as our simulation framework.

The ns-3 implementation of a BS-based WSN closely follows our analytical model in Sections 6.1-6.3 except that we allow backtracking. Backtracking is difficult to model analytically, yet it is an important feature for a real world deployment that can be more easily modeled using simulation. That is, in case a forwarding node that can bring the packet nearer to the BS cannot be found, the packet may be routed to a node that is farther away from the BS, in the hope that a path will eventually be found to reach the BS. In the simulation, we limit backtracking to go backward for a maximum of 3 hops after which the path is considered failed.

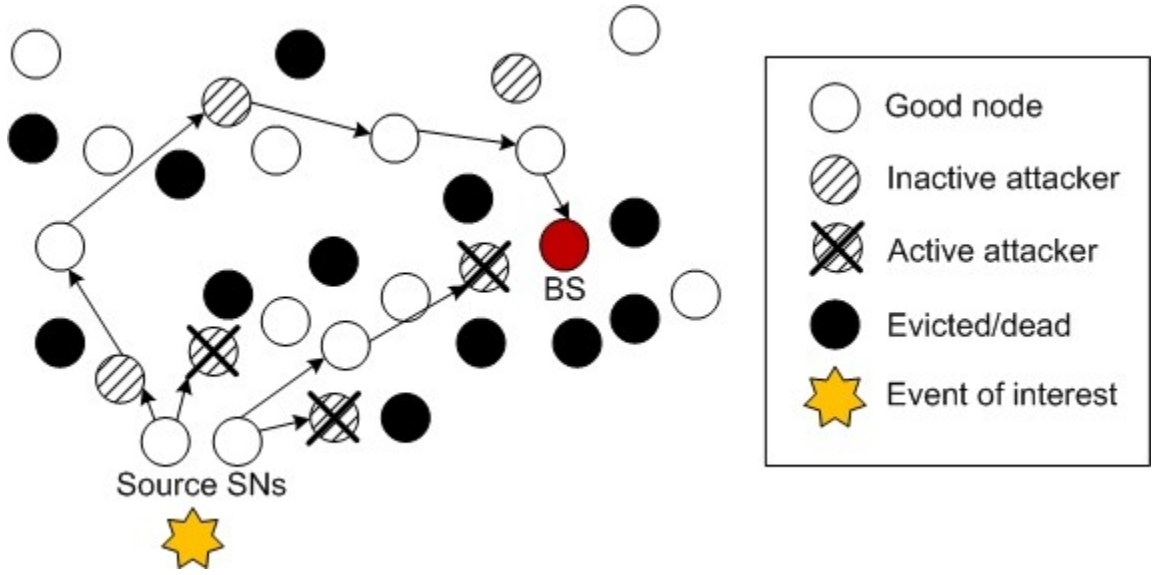


Figure 6-16: Multisource multipath routing in a BS-based WSN.

We illustrate how we simulate backtracking by Figure 6-16 for a case in which m_s=2 and m_p=2. In this figure, the left path of the left source SN backtracks 2 hops and eventually the packet reaches the BS using a total of 6 hops. All other paths fail due to the presence of malicious nodes on the path performing packet dropping with probability p_a. The query response is still delivered successfully since one path still successfully reaches the BS. The figure illustrates how backtracking could help finding a path to the BS at the expense of more energy consumption.

Below we report simulation results. Each data point (MTTF) is generated from 100 simulation runs. In each run, an observation of MTTF is collected when a system failure condition defined in Section 6.1.4 is satisfied.

Figure 6-17 and 6-16 compare simulation results with analytical results for MTTF vs. (m_p, m_s) under random and selective capture attacks, respectively. We see that there is a remarkable match between simulation and analytical results, both showing the same optimal (m_p, m_s) under which MTTF is maximized. The MTTF values generated from

the analytical model are consistently higher than those generated from simulation. We attribute this discrepancy to two reasons. The first reason is that a path in simulation in general has a longer overall length than that in the analytical model which assumes a straight line distance from the source SN to the BS. Consequently there are more intermediate nodes on a path and the path failure probability is higher. The second reason is due to the assumption of disjoint paths in the analytical model. This assumption in general is justified when there are sufficient nodes. However as nodes are evicted because of intrusion detection, it may not be justified. While the analytical model continues to use disjoint path assumption to do MTTF calculation, the simulation must use backtracking to cope with lack of nodes for forming m_p paths, especially when the frontier of a path is close to the BS which is affected the most by selective capture. This increases the path length and, consequently, reduces the path reliability. Despite these two factors contributing to a shorter MTTF value compared with the counterpart analytical MTTF value, we see that analytical model still accurately predicts the best (m_p, m_s) under which MTTF is maximized.

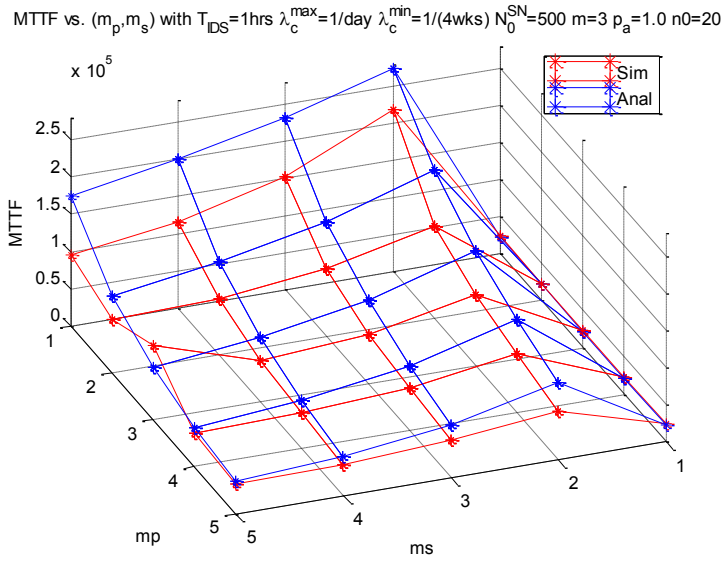


Figure 6-17: Simulation vs. Analytical Results under Random Capture.

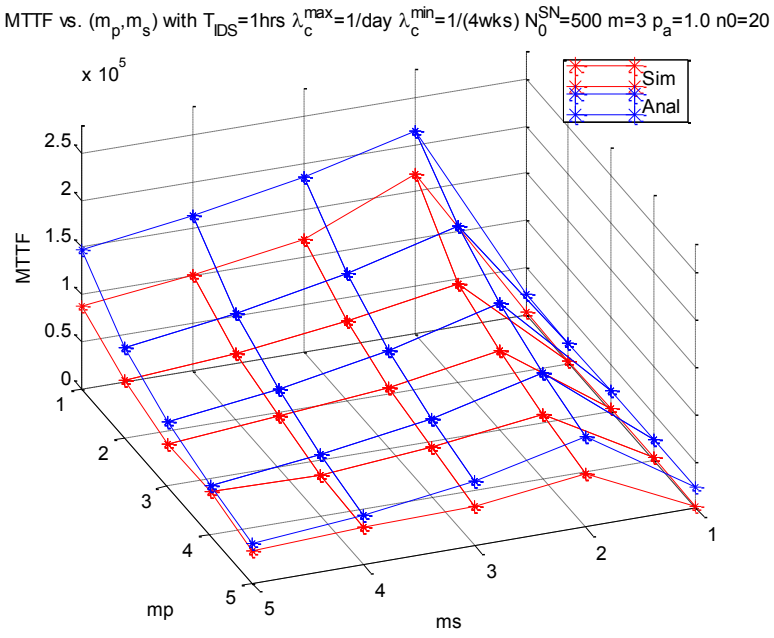
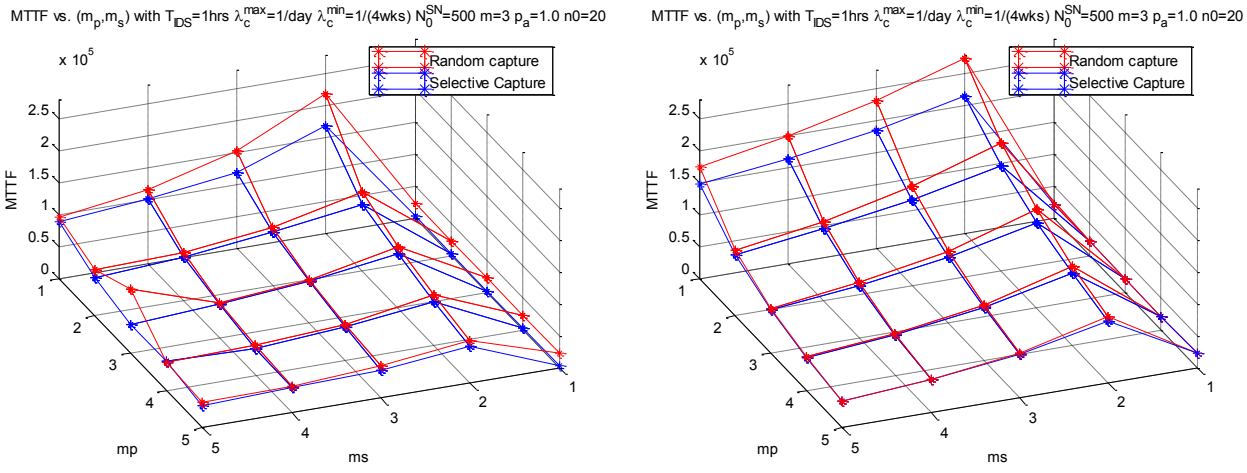


Figure 6-18: Simulation vs. Analytical under Selective Capture.

Figure 6-19 compares simulations results (case a on the left) vs. analytical results (case b on the right) for MTTF vs. (m_p, m_s) with high detection strength in the presence of high capture strength. We again see a remarkable match in trend between simulation and analytical results, both showing the same optimal (m_p, m_s) under which MTTF is maximized. We conclude that simulation results support the trends predicted by the analytical model.



(a) Simulation Results **(b) Analytical Results**

Figure 6-19: Comparing MTTF vs. (m_p, m_s) with High Detection Strength in the Presence of High Capture Strength: (a) Simulation Results and (b) Analytical Results.

6.6 Analyzing Distance-based Intrusion Detection and Distance-based Attacks

In this section we analyze *adaptive distance-based intrusion detection* (from the system defense perspective) by which the intrusion detection interval (T_{IDS}) is dependent on the distance (x) away from the BS. As a preliminary study, given that the selective capture rate increases linearly with decreasing x, we adjust T_{IDS} such that it decreases linearly with decreasing x (thus increasing the IDS detection rate) so as to counter selective capture. This "distance-based" IDS rate strategy is to be contrasted with the "fixed" IDS rate strategy.

We also analyze *adaptive distance-based attack* (from the adversary perspective) by which malicious nodes can make adaptive attacks to gain maximum benefit, given knowledge of intrusion detection/tolerance strength detected at runtime through broadcasting control messages in the WSN at runtime. The parameter which the adversary can control is the random attack probability (p_a). This "distance-based" random attack probability strategy is to be contrasted with the "fixed" random attack probability strategy.

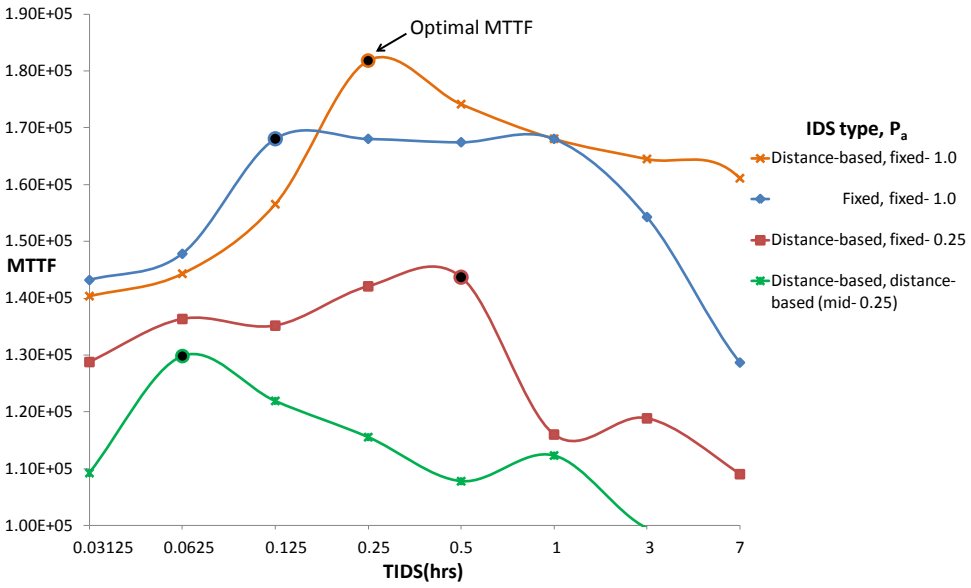


Figure 6-20: Effect of Distance-based Intrusion Detection and Distance-based Attack.

Figure 6-20 shows ns3 simulation results averaged from 100 simulation runs in a BS-based WSN characterized by a set of parameter values as given in Table 6-2. The 4 curves from top to bottom are for the distance-based IDS + fixed random attack proba-

bility ($p_a = 1$), fixed IDS + fixed random attack probability ($p_a = 1$), distance-based IDS + fixed random attack probability ($p_a = 1/4$), and distance-based IDS + distance-based random attack, respectively. The X coordinate is the fixed T_{IDS}value for the fixed IDS case, and is the average T_{IDS} value at x=1/2 from the BS for the distance-based IDS case.

The effect of using distance-based IDS vs. fixed IDS against fixed random attack probability on MTTF can be seen by comparing the top two curves. First we observe that the optimal MTTF achieved under distance-based IDS (top curve) is greater than the optimal MTTF under fixed IDS (second top curve), since increasing the IDS rate near the BS effectively detect and evict highly populated compromised insiders near the BS. This effectively counters selective capture which quickly populates compromised nodes near the BS. Second, we observe that distance-based IDS may produce a lower MTTF than that by fixed IDS. This is especially true when the detection rate is excessively high such that the loss in excessive energy consumption of the critical nodes outweighs the gain in detecting and evicting compromised critical nodes.

From the adversary perspective, we observe in Figure 6-20 that using a low random attack probability ($p_a = 0.25$) is more effective in this environment setting, since it can cause more damage to the system while avoiding detection. This is shown by comparing the top curve ($p_a = 1.0$) with the third curve ($p_a = 0.25$). Hence the adversary can counter distance-based IDS by adjusting p_a based on the distance x to the BS. The effect of using distance-based attack to counter distance-based IDS on MTTF can be seen by comparing the bottom two curves. Here we see that the bottom curve with distance-based attack with $p_a = 0.25$ at x=1/2 from the BS can significantly reduce the MTTF of the system compared with fixed attack with $p_a = 0.25$ throughout. The reason is that distance-based attack can effectively counter distance-based IDS to result in a low detection rate throughout. Especially, composed nods near the BS will use a very low attack probability to counter the very high intrusion detection rate being applied to critical nodes near the BS. As a result, the system will waste energy of critical nodes but miss detecting hidden compromised nodes.

For the defense perspective, nevertheless, dynamically adjusting the detection strength in term of the intrusion detection interval (T_{IDS}) as in this study and the number of verifiers (m) is the best defense mechanism that the system can leverage to maximize the system MTTF. Figure 6-20 clearly demonstrates that there exists an optimal detection strength setting under which the system MTTF is maximized, given some knowledge of the attacker strategies such as the fixed or distance-based random attack probability.

6.7 Summary

In this chapter we proposed and analyzed adaptive network management with three countermeasures for coping with selective capture and smart attack aiming to create holes near the base station in a wireless sensor network to block data delivery. We demonstrated that our countermeasures are effective against selective capture and smart attack. There exist best protocol settings in terms of the best radio adjustment, the best redundancy level for multipath routing, the best number of voters, and the best intrusion invocation interval used for intrusion detection to maximize the system lifetime. Leveraging the analysis techniques proposed in this chapter, one can obtain optimal protocol settings at static time, store them in a table, and apply a simple table lookup operation at runtime as discussed in Section 4.6 to determine optimal settings for adaptive network management to maximize the BS-based WSN lifetime without high runtime complexity.

Chapter 7

Conclusion

7.1 Completed Work

In this dissertation research, we have developed a class of dynamic redundancy management algorithms for redundancy management of multisource multipath routing for fault and intrusion tolerance, and voting-based IDS for intrusion detection, with the goal of maximizing the WSN lifetime while satisfying application QoS and security requirements, for homogeneous clustered WSNs (Chapter 4), heterogeneous clustered WSNs (Chapter 5), and BS-based WSNs (Chapter 6). By means of a novel analytical models based on probability theory, we explored the tradeoff between energy consumption vs. reliability, timeliness and security gain and identified the optimal multisource multipath redundancy level and intrusion detection settings for satisfying application QoS requirements while maximizing the lifetime of the WSN. We have addressed the design issue of "how many paths to use" to tolerate residual compromised nodes that survive our IDS, so as to maximize the WSN lifetime. We have also modeled smart attack behaviors and selective capture and investigated dynamic radio range adjustment, multisource multipath routing, and voting-based intrusion detection as countermeasure designs with ns-3 simulation. The dissertation work thus far has resulted in three conference publications, one journal publication, and two journal submissions listed below.

<u>Papers Published:</u>

- H. Al-Hamadi and I. R. Chen, "Dynamic Multisource Multipath Routing for Intrusion Tolerance and Lifetime Maximization of Autonomous Wireless Sensor Networks," *IEEE 11th International Symposium on Autonomous Decentralized Systems,* Mexico City, Mexico, pp. 10-16, March 2013. (Chapter 4.)

- H. Al-Hamadi and I. R. Chen, "Energy vs. QoS Tradeoff Analysis of Multipath Routing Protocols for Intrusion Tolerance in Heterogeneous Wireless Sensor Networks," *IEEE 10th International Symposium on Parallel and Distributed Processing with Applications*, Madrid, Spain, pp. 387-394, July 2012. (Chapter 5.)
- H. Al-Hamadi and I. R. Chen, "Redundancy Management of Multipath Routing for Intrusion Tolerance in Heterogeneous Wireless Sensor Networks," *IEEE Transactions on Network and Service Management,* vol. 10, no. 2, pp. 189-203, 2013. (Chapter 5.)
- H. Al-Hamadi and I. R. Chen, "Adaptive Network Management for Countering Selective Capture in Wireless Sensor Networks," *9th International Conference on Network and Service Management,* Zurich, Switzerland, pp. 203-210, October 2013. (Chapter 6.)

Papers Submitted:

- H. Al-Hamadi, and I. R. Chen, "Dynamic Redundancy Management of Integrated Intrusion Detection and Tolerance in Homogeneous Clustered Sensor Networks," submitted to *ACM Transactions on Sensor Networks (TOSN),* June 2013 and revised Dec. 2013. (Chapter 4.)
- H. Al-Hamadi, and I. R. Chen, "Adaptive Network Management for Countering Smart Attack and Selective Capture in Wireless Sensor Networks," submitted to *IEEE Transactions on Computers,* February 2014. (Chapter 6.)

7.2 Future Work

There are several future research directions that can be extended further from this dissertation research:

1. **Developing Trust-Based Intrusion Detection**: The proposed WSN in this dissertation can leverage trust management for WSNs [13, 14] to implement IDS functions with the goal to reduce false positive rate and increase detection rate without excessively waste energy. The effect of trust-based IDS designs on false positive rate, detection rate, and system lifetime, by exploiting the tradeoff between energy consumption vs. security gain for using trust-based IDS can be further analyzed utilizing more sophisticated techniques such as stochastic Petri nets [16, 30-33, 37, 40, 41, 46, 63, 64, 89]. Finally, performance enhancement of trust-based IDS vs. voting-based IDS against more sophisticated attacks can be explored.
2. **Developing Trust-Based Multipath Routing**: Developing and analyzing trust-based multipath routing algorithms to solve the "what paths to use" research issue can be further investigated. A SN can leverage its trust knowledge toward neighbors to select the most trustworthy carriers for packet forwarding in multisource multipath

routing. This will complement our solution toward the "how many paths to use" problem into a complete solution for multisource multipath routing for intrusion tolerance in WSNs. In the future, we plan to explore trust management in other fields such as [26, 27, 35, 45, 47, 48] for performing intrusion detection to address the issue of what paths one should use to avoid untrustworthy, malicious nodes to further enhance WSN survivability. This may involve the use of trust-based admission control strategies [36, 42-44, 125] to increase the probability of path success probability for data delivery. Further, we plan to consider fuzzy failure criteria [15, 28, 29] such that the WSN system fails after suffering a number of query failures instead of the binary failure criterion considered in the dissertation research.

3. **Malicious Node Behavior Modeling**: This dissertation research scratches the surface of malicious node behavior modeling by considering *random*, *opportunistic* and *insidious* attack behaviors by inside attackers in a wireless sensor network. In the future we plan to consider more sophisticated attacker models, e.g., a smart adversary that can perform more targeted attacks, capture certain strategic nodes with higher probability, alternate between benign and malicious behavior and collude with other attackers to avoid intrusion detection. In particular, we plan to investigate possible attacker strategies from the attacker perspective when given knowledge of intrusion detection/tolerance strength detected at runtime which is readily obtainable by insiders. The preliminary result performed in the dissertation research (in Section 6.6) sets the stage for more future research in this area.
4. **Extension to Sensor-Enriched Service-Oriented Systems:** The design principles developed and research outcomes discovered in the dissertation research can be applied to next-generation sensor-enriched service-oriented systems such as application-specific cyber physical systems [100-103] and the social Internet of things (IoT) systems [11, 34, 106]. In such systems every device is sensor-enriched and thus behaves similar to a sensor in a WSN. The same design issue regarding the tradeoff between energy consumption vs. reliability, timeliness and security gain remains the same. New design issues arise because every device may be mobile [39, 106] and can dynamically join and leave the network. A future research direction is to apply design principles developed in the dissertation research to these systems for dealing with smart adversary, with the objective to prolong the system lifetime while satisfying the system service goals.

Appendix – Notation and Acronym

Notations

A	Length of each side of a square sensor area (meter)
n_b	Size of a data packet (bit)
E_{elec}	Energy dissipation to run the transmitter and receiver circuitry (J/bit)
E_{amp}	Energy used by the transmit amplifier to achieve an acceptable signal to noise ratio (J/bit/m^2)
E_o	Initial energy per node (Joule)
E_{init}	Initial energy of the WSN (Joule)
$E_{clustering}(t)$	Energy consumed for executing the clustering algorithm at time t (Joule)
$E_{IDS}(t)$	Energy consumed for executing the IDS algorithm at time t (Joule)
$E_q(t)$	Energy consumed for executing a query at time t (Joule)
$R_q(t)$	Probability that a query reply at time t is delivered successfully by the deadline
r	Wireless radio communication range (meter)
q	node hardware failure probability
e_j	Transmission failure probability of node j
$N(t)$	Number of nodes in the WSN at time t

$N_{CH}(t)$	Number of CHs in the WSN at time t
$N_{SN}(t)$	Number of SNs in the WSN at time t
$n(t)$	Number of neighbor nodes at time t
$n_{good}(t)$	Number of good neighbor nodes at time t
$n_{bad}(t)$	Number of bad neighbor nodes at time t
N_q	Maximum number of queries before energy exhaustion
$N_{iteration}$	Number of iterations in clustering for CH election
m_p	Path redundancy level: Number of paths from a source CH to the sink
m_s	Source redundancy level: Number of SNs per cluster in response to a query
f	Fraction of neighbor nodes that will forward data
$\lambda(t)$	Node population density (nodes/meter^2) at time t
λ	Node population density at deployment time
λ_q	Query arrival rate (times/sec)
S_{jk}	Progressive transmission speed between node j and node k (meter/sec)
$T_{clustering}$	Time interval for executing the clustering algorithm (sec)
T_{req}	Query deadline requirement (sec)
λ_c	Node capture rate
α	Ratio of IDS execution rate to query arrival rate
β	Ratio of clustering rate to query arrival rate
m	Number of voters selected for executing distributed IDS
H_{pfp}	Probability of host IDS false positive
H_{pfn}	Probability of host IDS false negative

Pfp	Probability of distributed IDS false positive
Pfn	Probability of distributed IDS false negative
T_{IDS}	IDS interval time (sec)
$MTTF$	Lifetime of a WSN

Acronyms

BS	Base Station
CH	Cluster Head
IDS	Intrusion Detection System
MANET	Mobile Ad-Hoc Network
MTTF	Mean Time To Failure
PC	Processing Center
QoS	Quality of Service
SN	Sensor Node
WSN	Wireless Sensor Network

Bibliography

[1] L. Adnan, Y. Yussoff, and H. Hashim, "Secure boot process for wireless sensor node," *International Conf. on Computer Applications and Industrial Electronics (ICCAIE)*, 2010, pp. 646-649.

[2] Advanticsys. (2013, Nov. 03). *Products: Wireless Sensor Network Kits*. Available: http://www.advanticsys.com/shop/wireless-sensor-networks-wireless-sensor-networks-kits-c-7_10.html

[3] M. R. Akhondi, A. Talevski, S. Carlsen, and S. Petersen, "Applications of wireless sensor networks in the oil, gas and resources industries," *24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, 2010, pp. 941-948.

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

[5] H. Al-Hamadi and I. R. Chen, "Adaptive Network Management for Countering Selective Capture in Wireless Sensor Networks," *9th International Conference on Network and Service Management*, Zurich, Switzerland, 2013, pp. 203-210.

[6] H. Al-Hamadi and I. R. Chen, "Redundancy Management of Multipath Routing for Intrusion Tolerance in Heterogeneous Wireless Sensor Networks," *IEEE Transactions on Network and Service Management*, vol. 10, no. 2, pp. 189-203, 2013.

[7] H. Al-Hamadi and I. R. Chen, "Energy vs. QoS Tradeoff Analysis of Multipath Routing Protocols for Intrusion Tolerance in Heterogeneous Wireless Sensor Networks," *IEEE 10th Int. Symp. on Parallel and Distributed Processing with Applications (ISPA)*, Madrid, Spain, July 2012, pp. 387-394.

[8] H. Al-Hamadi and I. R. Chen, "Dynamic Multisource Multipath Routing for Intrusion Tolerance and Lifetime Maximization of Autonomous Wireless Sensor Networks," *IEEE 11th Int. Symp. on Autonomous Decentralized System (ISADS)*, Mexico City, Mexico, March 2013, pp.10-16.

[9] H. Alwan and A. Agarwal, "A Survey on Fault Tolerant Routing Techniques in Wireless Sensor Networks," *Third International Conference on Sensor Technologies and Applications*, 2009, pp. 366-371.

[10] I. M. Atakli, H. Hu, Y. Chen, W. S. Ku, and Z. Su, "Malicious node detection in wireless sensor networks using weighted trust evaluation," *Spring simulation multiconference*, 2008, pp. 836-843.

[11] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks,* vol. 54, no. 15, pp. 2787-2805, 2010.

[12] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," *22nd Conf. of IEEE Computer and Communications*, 2003, pp. 1713-1723.

[13] F. Bao, I. R. Chen, M. Chang, and J.-H. Cho, "Trust-Based Intrusion Detection in Wireless Sensor Networks," *2011 IEEE International Conference on Communications (ICC)*, 2011, pp. 1-6.

[14] F. Bao, I. R. Chen, M. Chang, and J. H. Cho, "Hierarchical Trust Management for Wireless Sensor Networks and its Applications to Trust-Based Routing and Intrusion Detection," *IEEE Transactions on Network and Service Management,* vol. 9, no. 2, pp. 161-183, 2012.

[15] F. B. Bastani, I. R. Chen, and T. Tsao, "Reliability of Systems with Fuzzy-Failure Criterion," *Annual Reliability and Maintainability Symposium*, Anaheim, California, USA, 1994, pp. 442–448.

[16] F. B. Bastani, I. L. Yen, and I. R. Chen, "A class of inherently fault tolerant distributed programs," *IEEE Transactions on Software Engineering,* vol. 14, no. 10, pp. 1432-1442, 1988.

[17] A. Becher, Z. Benenson, and M. Dornseif, *Tampering with motes: Real-world physical attacks on wireless sensor networks*: Springer, 2006.

[18] L. E. Bengtsson, "Lookup Table Optimization for Sensor Linearization in Small Embedded Systems," *Journal of Sensor Technology,* vol. 2, no. 4, pp. 177-184, 2012.

[19] V. Bhuse and A. Gupta, "Anomaly intrusion detection in wireless sensor networks," *J. High Speed Netw.*, vol. 15, no. 1, pp. 33-51, 2006.

[20] C. Bo, D. Ren, S. Tang, X.-Y. Li, X. Mao, Q. Huang, L. Mo, Z. Jiang, Y. Sun, and Y. Liu, "Locating sensors in the forest: A case study in greenorbs," *Proc. IEEE INFOCOM*, 2012, pp. 1026-1034.

[21] S. Bo, L. Osborne, X. Yang, and S. Guizani, "Intrusion detection techniques in mobile ad hoc and wireless sensor networks," *IEEE Wireless Commun.*, vol. 14, no. 5, pp. 56-63, 2007.

[22] G. Bravos and A. G. Kanatas, "Energy consumption and trade-offs on wireless sensor networks," *16th IEEE Int. Symp. on Personal, Indoor and Mobile Radio Communications*, 2005, pp. 1279-1283.

[23] S. Buchegger and J.-Y. L. Boudec, "Nodes bearing grudges: towards routing security, fairness, and robustness in mobile ad hoc networks," *Proc. of 10th Euromicro conference on Parallel, distributed and network-based processing*, Canary Islands, Spain, 2002.

[24] M. Cardei, Y. Shuhui, and W. Jie, "Algorithms for Fault-Tolerant Topology in Heterogeneous Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems,* vol. 19, no. 4, pp. 545-558, 2008.

[25] I. Chatzigiannakis and A. Strikos, "A decentralized intrusion detection system for increasing security of wireless sensor networks," *IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2007, pp. 1408-1411.

[26] I. R. Chen, F. Bao, M. Chang, and J. H. Cho, "Trust Management for Encounter-Based Routing in Delay Tolerant Networks," *IEEE Global Communications Conf.*, Miami, Florida, USA, Dec. 2010, pp. 1-6.

[27] I. R. Chen, F. Bao, M. Chang, and J. H. Cho, "Dynamic trust management for delay tolerant networks and its application to secure routing," *IEEE Transactions on Parallel and Distributed Systems,* 2014.

[28] I. R. Chen and F. B. Bastani, "Effect of artificial-intelligence planning-procedures on system reliability," *IEEE Transactions on Reliability*, vol. 40, no. 3, pp. 364-369, 1991.

[29] I. R. Chen, F. B. Bastani, and T. W. Tsao, "On the reliability of AI planning software in real-time applications," *IEEE Transactions on Knowledge and Data Engineering,* vol. 7, no. 1, pp. 4-13, 1995.

[30] I. R. Chen, T. M. Chen, and C. Lee, "Performance evaluation of forwarding strategies for location management in mobile networks," *The Computer Journal,* vol. 41, no. 4, pp. 243-253, 1998.

[31] I. R. Chen, T. M. Chen, and C. Lee, "Agent-based forwarding strategies for reducing location management cost in mobile networks," *Mobile Networks and Applications,* vol. 6, no. 2, pp. 105-115, 2001.

[32] I. R. Chen and B. Gu, "Quantitative analysis of a hybrid replication with forwarding strategy for efficient and uniform location management in mobile wireless networks," *IEEE Transactions on Mobile Computing,* vol. 2, no. 1, pp. 3-15, 2003.

[33] I. R. Chen, B. Gu, S. E. George, and S. T. Cheng, "On failure recoverability of client-server applications in mobile wireless environments," *IEEE Transactions on Reliability,* vol. 54, no. 1, pp. 115-122, 2005.

[34] I. R. Chen, J. Guo, and F. Bao, "Trust Management for Service Composition in SOA-based Internet of Things Systems," *IEEE Wireless Communications and Networking Conference (WCNC 2014)*, Istanbul, Turkey, 2014.

[35] I. R. Chen, J. Guo, F. Bao, and J. H. Cho, "Integrated Social and Quality of Service Trust Management of Mobile Groups in Ad Hoc Networks," *9th IEEE Conference on Information, Communications and Signal Processing*, Tainan, Taiwan, Dec. 2013.

[36] I. R. Chen and T. H. Hsi, "Performance analysis of admission control algorithms based on reward optimization for real-time multimedia servers," *Performance Evaluation,* vol. 33, no. 2, pp. 89-112, 1998.

[37] I. R. Chen, N. A. Phan, and I. L. Yen, "Algorithms for supporting disconnected write operations for wireless web access in mobile client-server environments," *IEEE Transactions on Mobile Computing,* vol. 1, no. 1, pp. 46-58, 2002.

[38] I. R. Chen, A. P. Speer, and M. Eltoweissy, "Adaptive Fault-Tolerant QoS Control Algorithms for Maximizing System Lifetime of Query-Based Wireless Sensor Networks," *IEEE Trans. on Dependable and Secure Computing,* vol. 8, no. 2, pp. 161-176, 2011.

[39] I. R. Chen and N. Verma, "Simulation study of a class of autonomous host-centric mobility prediction algorithms for wireless cellular and ad hoc networks," *ACM 36th Annual Symp. on Simulation*, Orlando, Florida, USA, 2003, pp. 65-72.

[40] I. R. Chen and D. C. Wang, "Analysis of replicated data with repair dependency," *The Computer Journal,* vol. 39, no. 9, pp. 767-779, 1996.

[41] I. R. Chen and D. C. Wang, "Analyzing dynamic voting using petri nets," *15th IEEE Symp. on Reliable Distributed Systems.*, Niagra Falls, Canada, 1996, pp. 44-53.

[42] I. R. Chen, O. Yilmaz, and I. L. Yen, "Admission control algorithms for revenue optimization with QoS guarantees in mobile wireless networks," *Wireless Personal Communications,* vol. 38, no. 3, pp. 357-376, 2006.

[43] S. T. Cheng, C. M. Chen, and I. R. Chen, "Dynamic quota-based admission control with sub-rating in multimedia servers," *Multimedia systems,* vol. 8, no. 2, pp. 83-91, 2000.

[44] S. T. Cheng, C. M. Chen, and I. R. Chen, "Performance evaluation of an admission control algorithm: dynamic threshold with negotiation," *Performance Evaluation,* vol. 52, no. 1, pp. 1-13, 2003.

[45] J. H. Cho and I. R. Chen, "On the tradeoff between altruism and selfishness in MANET trust management," *Ad Hoc Networks,* vol. 11, no. 8, pp. 2217-2234, 2013.

[46] J. H. Cho, I. R. Chen, and P. G. Feng, "Effect of Intrusion Detection on Reliability of Mission-Oriented Mobile Group Systems in Mobile Ad Hoc Networks," *IEEE Transactions on Reliability,* vol. 59, no. 1, pp. 231-241, 2010.

[47] J. H. Cho, A. Swami, and I. R. Chen, "Modeling and analysis of trust management with trust chain optimization in mobile ad hoc networks," *Journal of Network and Computer Applications,* vol. 35, no. 3, pp. 1001-1012, 2012.

[48] J. H. Cho, A. Swami, and I. R. Chen, "Modeling and analysis of trust management for cognitive mission-driven group communication systems in mobile ad hoc networks," *IEEE International Conference on Computational Science and Engineering*, Vancouver, BC, Canada, August 2009, pp. 641-650.

[49] A. P. R. da Silva, M. H. T. Martins, B. P. S. Rocha, A. A. F. Loureiro, L. B. Ruiz, and H. C. Wong, "Decentralized intrusion detection in wireless sensor networks," *1st ACM Workshop on Quality of Service & Security in Wireless and Mobile Networks*, Montreal, Quebec, Canada, 2005.

[50] P. De, Y. Liu, and S. K. Das, "Deployment-aware modeling of node compromise spread in wireless sensor networks using epidemic theory," *ACM Trans. on Sen. Netw. (TOSN),* vol. 5, no. 3, pp. 1-33, 2009.

[51] B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: reliable information forwarding using multiple paths in sensor networks," *28th IEEE Local Computer Networks*, Bonn, Germany, 2003, pp. 406-415.

[52] J. Deng, R. Han, and S. Mishra, "Intrusion tolerance and anti-traffic analysis strategies for wireless sensor networks," *International Conference on Dependable Systems and Networks*, 2004, pp. 637-646.

[53] J. Deng, R. Han, and S. Mishra, "Countermeasures against traffic analysis attacks in wireless sensor networks," *First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, 2005, pp. 113-126.

[54] J. Deng, R. Han, and S. Mishra, "INSENS: Intrusion-tolerant routing for wireless sensor networks," *Computer Communications,* vol. 29, no. 2, pp. 216-230, 2006.

[55] S. M. Diamond and M. G. Ceruti, "Application of wireless sensor network to military information integration," *5th IEEE International Conference on Industrial Informatics*, 2007, pp. 317-322.

[56] M. P. Durisic, Z. Tafa, G. Dimic, and V. Milutinovic, "A survey of military applications of wireless sensor networks," *Mediterranean Conference on Embedded Computing (MECO)*, 2012, pp. 196-199.

[57] P. Ebinger and N. Bissmeyer, "TEREC: Trust Evaluation and Reputation Exchange for Cooperative Intrusion Detection in MANETs," *7th Annu. Communication Networks and Services Research Conf.*, 2009, pp. 378-385.

[58] E. Felemban, L. Chang-Gun, and E. Ekici, "MMSPEED: multipath Multi-SPEED protocol for QoS guarantee of reliability and. Timeliness in wireless sensor networks," *IEEE Trans. Mobile Comput.,* vol. 5, no. 6, pp. 738-754, 2006.

[59] G. Fersi, W. Louati, and M. B. Jemaa, "Distributed Hash table-based routing and data management in wireless sensor networks: a survey," *Wireless networks,* vol. 19, no. 2, pp. 219-236, 2013.

[60] A. Francillon and C. Castelluccia, "Code injection attacks on harvard-architecture devices," *Proc. of the 15th ACM conf. on Computer and communications security*, 2008, pp. 15-26.

[61] W. Furong, H. Chen, Z. Jing, and R. Chunming, "IDMTM: A Novel Intrusion Detection Mechanism Based on Trust Model for Ad Hoc Networks," *22nd Int. Conf. on Advanced Information Networking and Applications*, 2008, pp. 978-984.

[62] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," *Proc. of 2nd ACM international symp. on Mobile ad hoc networking & computing*, Long Beach, CA, USA, 2001.

[63] S. E. George, I. R. Chen, and Y. Jin, "Movement-based checkpointing and logging for recovery in mobile computing systems," *Proceedings of the 5th ACM international workshop on Data engineering for wireless and mobile access*, 2006, pp. 51-58.

[64] B. Gu and I. R. Chen, "Performance analysis of location-aware mobile service proxies for reducing network cost in personal communication systems," *Mobile Networks and Applications,* vol. 10, no. 4, pp. 453-463, 2005.

[65] C. Haowen and A. Perrig, "PIKE: peer intermediaries for key establishment in sensor networks," *24th Annu. Joint Conf. of the IEEE Computer and Communications Societies.*, 2005, pp. 524-535.

[66] H. Hassanein and L. Jing, "Reliable Energy Aware Routing In Wireless Sensor Networks," *Second IEEE Workshop on Dependability and Security in Sensor Networks and Systems*, 2006, pp. 54-64.

[67] J. Hedley, C. Roelfsema, and S. R. Phinn, "Efficient radiative transfer model inversion for remote sensing applications," *Remote Sensing of Environment,* vol. 113, no. 11, pp. 2527-2532, 2009.

[68] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wireless Commun.,* vol. 1, no. 4, pp. 660-670, 2002.

[69] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, "Network simulations with the ns-3 simulator," *SIGCOMM demonstration,* 2008.

[70] H. T. Hoang and H. E. Nam, "Optimal Selection and Activation of Intrusion Detection Agents for Wireless Sensor Networks," *Future Generation Communication and Networking*, 2007, pp. 350-355.

[71] D. Huang and D. Medhi, "Secure pairwise key establishment in large-scale sensor networks: An area partitioning and multigroup key predistribution approach," *ACM Trans. Sen. Netw.,* vol. 3, no. 3, p. 16, 2007.

[72] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Location-aware key management scheme for wireless sensor networks," *Proc. of 2nd ACM workshop on Security of ad hoc and sensor networks*, Washington DC, USA, 2004.

[73] M. A. Hussain and K. K. Sup, "WSN research activities for military application," *11th Int. Conf. on Advanced Communication Technology*, 2009, pp. 271-274.

[74] M. Ilyas and I. Mahgoub, *Handbook of sensor networks: compact wireless and wired sensing systems*: CRC, 2004.

[75] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," *Proc. of 6th annu. international conf. on Mobile computing and networking*, Boston, Massachusetts, United States, 2000.

[76] C. Jaikaeo, C. Srisathapornphat, and C. C. Shen, "Diagnosis of sensor networks," *IEEE International Conference on Communications*, 2001, pp. 1627-1632 vol.5.

[77] H. Jing, Z. Yanchun, H. Guangyan, and S. Yong, "Network Lifetime of Application-Specific Randomly Deployed Wireless Sensor Networks in Arbitrary

Sensor Density," *6th IEEE/ACIS International Conference on Computer and Information Science (ICIS)*, 2007, pp. 352-357.

[78] K. D. Kang, K. Liu, and N. Abu-Ghazaleh, "Securing Geographic Routing in Wireless Sensor Networks," *9th Annu. Cyber Security Conf. on Information Assurance*, Albany, NY, USA, 2006.

[79] H. Karl and A. Willig, *Protocols and architectures for wireless sensor networks*: Wiley-Interscience, 2007.

[80] C. Karlof, Y. Li, and J. Polastre, "ARRIVE: Algorithm for Robust Routing in Volatile Environments," Technical Report, UCB//CSD-03-1233, University of California at Berkeley, 2003.

[81] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," *1st IEEE Int. Workshop on Sensor Network Protocols and Applications*, 2003, pp. 113-127.

[82] I. Khalil, S. Bagchi, and C. Nina-Rotaru, "DICAS: Detection, Diagnosis and Isolation of Control Attacks in Sensor Networks," *First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, 2005, pp. 89-100.

[83] I. Koutsopoulos and M. Halkidi, "Measurement aggregation and routing techniques for energy-efficient estimation in wireless sensor networks," *8th Int. Symp. on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, 2010, pp. 1-10.

[84] I. Krontiris, T. Dimitriou, and F. C. Freiling, "Towards intrusion detection in wireless sensor networks," *13th European Wireless Conference*, Paris, France, 2007.

[85] L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Trans. Programming Languages and Systems,* vol. 4, no. 3, pp. 382-401, 1982.

[86] Y. Lan, L. Lei, and G. Fuxiang, "A multipath secure routing protocol based on malicious node detection," *Chinese Control and Decision Conference*, 2009, pp. 4323-4328.

[87] S. B. Lee and Y. H. Choi, "A secure alternate path routing in sensor networks," *Comput. Commun.,* vol. 30, no. 1, pp. 153-165, 2006.

[88] S. H. Lee, S. Lee, H. Song, and H. S. Lee, "Wireless sensor network design for tactical military applications: remote large-scale environments," *IEEE Military Communications Conference*, 2009, pp. 1-7.

[89] Y. Li and I. R. Chen, "Design and performance analysis of mobility management schemes based on pointer forwarding for wireless mesh networks," *IEEE Transactions on Mobile Computing,* vol. 10, no. 3, pp. 349-361, 2011.

[90] R. Lin, Z. Wang, and Y. Sun, "Wireless sensor networks solutions for real time monitoring of nuclear power plant," *WCICA 2004. Fifth World Congress on Intelligent Control and Automation*, 2004, pp. 3663-3667.

[91] F. Liu, X. Cheng, and D. Chen, "Insider attacker detection in wireless sensor networks," *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, 2007, pp. 1937-1945.

[92] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, and X. Li, "Does wireless sensor network scale? A measurement study on GreenOrbs," *IEEE Trans. on Parallel and Distributed Systems,* vol. 24, no. 10, pp. 1983-1993, 2013.

[93] W. Lou and Y. Kwon, "H-SPREAD: a hybrid multipath scheme for secure and reliable data collection in wireless sensor networks," *IEEE Trans. Veh. Technol.,* vol. 55, no. 4, pp. 1320-1330, 2006.

[94] R. Machado, N. Ansari, G. Wang, and S. Tekinay, "Adaptive density control in heterogeneous wireless sensor networks with and without power management," *IET Communications,* vol. 4, no. 7, pp. 758-767, 2010.

[95] X. Mao, X. Miao, Y. He, X.-Y. Li, and Y. Liu, "CitySee: urban CO 2 monitoring with sensors," *Proc. IEEE INFOCOM*, 2012, pp. 1611-1619.

[96] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," *Proc. of 6th annu. int. conf. on Mobile computing and networking*, Boston, Massachusetts, United States, 2000.

[97] P. Michiardi and R. Molva, "Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," *Proc. of IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security*, 2002.

[98] P. Michiardi and R. Molva, "Simulation-based analysis of security exposures in mobile ad hoc networks," *European Wireless Conference*, 2002.

[99] S. Misra, P. V. Krishna, and K. I. Abraham, "Energy efficient learning solution for intrusion detection in wireless sensor networks," *Second International Conference on Communication Systems and Networks*, 2010, pp. 1-6.

[100] R. Mitchell and I. R. Chen, "Effect of Intrusion Detection and Response on Reliability of Cyber Physical Systems," *IEEE Transactions on Reliability,* vol. 62, no. 1, pp. 199-210, 2013.

[101] R. Mitchell and I. R. Chen, "Adaptive Intrusion Detection for Unmanned Aircraft Systems based on Behavior Rule Specification," *IEEE Transactions on Systems Man and Cybernetics,* pp. 1-10, 2014.

[102] R. Mitchell and I. R. Chen, "A Survey of Intrusion Detection Techniques in Cyber Physical Systems," *ACM Computing Survey,* 2014.

[103] R. Mitchell and I. R. Chen, "Behavior-Rule Based Intrusion Detection Systems for Safety Critical Smart Grid Applications," *IEEE Transactions on Smart Grid,* vol. 4, no. 3, pp. 1254-1263, Sept. 2013.

[104] N. Nasser and Y. Chen, "SEEM: Secure and energy-efficient multipath routing protocol for wireless sensor networks," *Computer Communications,* vol. 30, no. 11-12, pp. 2401-2412, 2007.

[105] J. Ni, L. Zhou, and C. V. Ravishankar, "Dealing with random and selective attacks in wireless sensor systems," *ACM Trans. Sen. Netw.,* vol. 6, no. 2, pp. 1-40, 2010.

[106] M. Nitti, R. Girau, and L. Atzori, "Trustworthiness Management in the Social Internet of Things," *IEEE Transactions on Knowledge and Data Management,* 2014.

[107] S. Qun, "Power Management in Networked Sensor Radios A Network Energy Model," *IEEE Sensors Applications Symp.*, 2007, pp. 1-5.

[108] S. Rajasegarar, C. Leckie, and M. Palaniswami, "Anomaly detection in wireless sensor networks," *IEEE Wireless Communications,* vol. 15, no. 4, pp. 34-40, 2008.

[109] L. Shi, Q. Miao, and D. Jinglin, "Architecture of Wireless Sensor Networks for Environmental Monitoring," *ETT and GRS*, 2008, pp. 579-582.

[110] T. Shu, M. Krunz, and S. Liu, "Secure Data Collection in Wireless Sensor Networks Using Randomized Dispersive Routes," *IEEE Trans. Mobile Comput.,* vol. 9, no. 7, pp. 941-954, 2010.

[111] I. Slama, B. Jouaber, and D. Zeghlache, "Optimal Power management scheme for Heterogeneous Wireless Sensor Networks: Lifetime Maximization under QoS and Energy Constraints," *Third International Conference on Networking and Services (ICNS)* 2007, pp. 69-69.

[112] D. Somasundaram and R. Marimuthu, "A Multipath Reliable Routing for detection and isolation of malicious nodes in MANET," *International Conference on Computing, Communication and Networking*, 2008, pp. 1-8.

[113] J. Staddon, D. Balfanz, and G. Durfee, "Efficient tracing of failed nodes in sensor networks," *Proc. of 1st ACM international workshop on Wireless sensor networks and applications*, Atlanta, Georgia, USA, 2002.

[114] E. Stavrou and A. Pitsillides, "A survey on secure multipath routing protocols in WSNs," *Comput. Netw.,* vol. 54, no. 13, pp. 2215-2238, 2010.

[115] H. Su and X. Zhang, "Network Lifetime Optimization for Heterogeneous Sensor Networks With Mixed Communication Modes," *IEEE Wireless Communications and Networking Conference*, 2007, pp. 3158-3163.

[116] W. T. Su, K. M. Chang, and Y. H. Kuo, "eHIP: An energy-efficient hybrid intrusion prohibition system for cluster-based wireless sensor networks," *Computer Networks,* vol. 51, no. 4, pp. 1151-1168, 2007.

[117] J. Sun, "Trade-off Consideration in Query Processing for Wireless Sensor Networks," *IEEE International Conference on Networking, Sensing and Control*, 2008, pp. 1453-1458.

[118] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," *Communications of the ACM,* vol. 47, no. 6, pp. 34-40, 2004.

[119] R. Vidhyapriya and P. Vanathi, "Energy efficient adaptive multipath routing for wireless sensor networks."

[120] G. Wang, W. Zhang, G. Cao, and T. La Porta, "On supporting distributed collaboration in sensor networks," *IEEE Military Communications Conference*, 2003, pp. 752-757.

[121] W. Wang, M. Chatterjee, and K. Kwiat, "Coexistence with malicious nodes: A game theoretic approach," *International Conference on Game Theory for Networks*, 2009, pp. 277-286.

[122] A. D. Wood, L. Fang, J. A. Stankovic, and T. He, "SIGF: a family of configurable, secure routing protocols for wireless sensor networks," *Proc. of the fourth ACM workshop on Security of ad hoc and sensor networks*, Alexandria, Virginia, USA, 2006.

[123] W. XiaoLing, M. Dejun, L. Zheyuan, and Y. Libin, "Optimal Scheduling Mechanism of Intrusion Detection in Wireless Sensor Networking," *International Conference on Information Engineering and Computer Science*, 2009, pp. 1-4.

[124] Y. Yang, C. Zhong, Y. Sun, and J. Yang, "Network coding based reliable disjoint and braided multipath routing for sensor networks," *J. Netw. Comput. Appl.,* vol. 33, no. 4, pp. 422-432, 2010.

[125] O. Yilmaz and I. R. Chen, "Utilizing call admission control for pricing optimization of multiple service classes in wireless cellular networks," *Computer Communications,* vol. 32, no. 2, pp. 317-323, 2009.

[126] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans. Mobile Comput.,* vol. 3, no. 4, pp. 366-379, 2004.

[127] H. Yu and M. Guo, "An efficient oil and gas pipeline monitoring systems based on wireless sensor networks," *International Conference on Information Security and Intelligence Control (ISIC)*, 2012, pp. 178-181.

[128] G. Yuan and J. McNair, "Redundancy versus lifetime tradeoff analysis for environment monitoring using wireless sensor networks," *Second IEEE Workshop on Dependability and Security in Sensor Networks and Systems*, 2006, pp. 7 pp.-77.

[129] Y. Zhou, Y. Fang, and Y. Zhang, "Securing wireless sensor networks: a survey," *IEEE Communications Surveys & Tutorials,* vol. 10, no. 3, pp. 6-28, 2008.

[130] S. Zhu, S. Setia, and S. Jajodia, "LEAP: efficient security mechanisms for large-scale distributed sensor networks," *10th ACM conference on Computer and Communications Security*, Washington D.C., USA, 2003.

[131] T. Zhu, A. Mohaisen, Y. Ping, and D. Towsley, "DEOS: Dynamic energy-oriented scheduling for sustainable wireless sensor networks," *Proceedings IEEE INFOCOM*, 2012, pp. 2363-2371.