Redundancy Management of Multipath Routing for Intrusion Tolerance in Heterogeneous Wireless Sensor Networks

Hamid Al-Hamadi and Ing-Ray Chen, *Member, IEEE*

***Abstract*—In this paper we propose redundancy management of heterogeneous wireless sensor networks (HWSNs), utilizing 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 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. 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.**

***Index Terms*—Heterogeneous wireless sensor networks, multipath routing, intrusion detection, reliability, security, energy conservation.**

I. Introduction

MANY wireless sensor networks (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 QoS requirements such as reliability, timeliness and security, but also minimize energy consumption to prolong the system useful lifetime. The tradeoff between *energy* consumption vs. *reliability* 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.

It is commonly believed in the research community that clustering is an effective solution for achieving scalability, energy conservation, and reliability. Using homogeneous nodes which rotate among themselves in the roles of cluster heads (CHs) and sensor nodes (SNs) leveraging CH election protocols such as HEED [1] for lifetime maximization has been considered [2], [3]. Recent studies [4]–[6] demonstrated that using heterogeneous nodes can further enhance performance and prolong the system lifetime. In the latter case, nodes with superior resources serve as CHs performing computationally intensive tasks while inexpensive less capable SNs are utilized mainly for sensing the environment.

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. This is especially the case in heterogeneous WSN (HWSN) environments in which CH nodes may take a more critical role in gathering and routing sensing data. Thus, very likely the system would 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 [7]–[11], the issue of how often intrusion detection should be invoked for energy reasons in order to remove potentially malicious nodes so that the system lifetime is maximized (say to prevent a Byzantine failure [12]) 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 or base station increases as we have more paths doing data delivery. While most prior research focused on using multipath routing to improve reliability [2], [3], [13], some attention has been paid to using multipath routing to tolerate insider attacks [14]–[16]. These studies, however, largely ignored the tradeoff between QoS gain vs. energy consumption which can adversely shorten the system lifetime.

The research problem we are addressing in this paper is effective redundancy management of a clustered HWSN to prolong its lifetime operation in the presence of unreliable and malicious nodes. We address the tradeoff between energy consumption vs. QoS gain in reliability, timeliness and security with the goal to maximize the lifetime of a clustered HWSN while satisfying application QoS requirements in the context of multipath routing. More specifically, we analyze the optimal amount of redundancy through which data are routed to a remote sink in the presence of unreliable and

Manuscript received April 17, 2012; revised June 28, 2012 and January 30, 2013. The associate editor coordinating the review of this paper and approving it for publication was C. Hong.

The authors are with the Department of Computer Science, Virginia Tech, Falls Church, VA, 22043, USA (e-mail: {hhamadi, irchen}@vt.edu).

Digital Object Identifier 10.1109/TNSM.2013.043013.120282

malicious nodes, so that the query success probability is maximized while maximizing the HWSN lifetime. We consider this optimization problem for the case in which a voting-based distributed intrusion detection algorithm is applied to remove malicious nodes from the HWSN. Our contribution is a model-based analysis methodology by which the optimal multipath redundancy levels and intrusion detection settings may be identified for satisfying application QoS requirements while maximizing the lifetime of HWSNs.

For the issue of intrusion tolerance through 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. 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 [17] or AODV for MANETs [18]), nor the use of feedback information to solve the problem. Rather, for energy conservation, we employ a distributed light-weight IDS by which intrusion detection is performed only locally. Nodes that are identified compromised are removed from the HWSN. Only compromised nodes that survive detection have the chance to disturb routing. One main contribution of our paper is that we decide "how many paths to use" in order to tolerate residual compromised nodes that survive our IDS, so as to maximize the HWSN lifetime.

The rest of the paper is organized as follows. In Section II we discuss related work and contrast our approach with existing work on multipath routing for intrusion tolerance and reliability enhancement. In Section III, we define our system model with system assumptions given. In Section IV we derive an analytical expression for the system lifetime, considering factors such as query rate, attacker behavior, node capture rate, link reliability, and energy consumption. In Section V we present numerical data and provide physical interpretations of the results. In Section VI, we present a dynamic management algorithm for managing redundancy of multipath routing for intrusion tolerance to maximize the system lifetime while satisfying the system reliability, timeliness and security requirements in the presence of unreliable wireless communication and malicious nodes. Finally in Section VII we conclude the paper and outline some future research areas.

II. Related Work

Over the past few years, many protocols exploring the tradeoff between energy consumption and QoS gain particularly in reliability in HWSNs have been proposed. In [19], the optimal communication range and communication mode were derived to maximize the HWSN lifetime. In [20], the authors devised intra-cluster scheduling and inter-cluster multi-hop routing schemes to maximize the network lifetime. They considered a hierarchal HWSN 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 either work cited above, no consideration was given to the existence of malicious nodes. In [21], the authors considered a two-tier HWSN with the objective of maximizing network lifetime while fulfilling power management and coverage objectives. They determined the optimal density ratio of the two tier's nodes to maximize the system lifetime. Relative to [21] our work also considers heterogeneous nodes with different densities and capabilities. However, our work considers the presence of malicious nodes and explores the tradeoff between energy consumption vs. QoS gain in both security and reliability to maximize the system lifetime.

In the context of secure multipath routing for intrusion tolerance, [22] provides an excellent survey in this topic. In [15] the authors considered a multipath routing protocol to tolerate black hole and selective forwarding attacks. The basic idea is to use overhearing to avoid sending packets to malicious nodes. In [14] the authors considered a disjoint multipath routing protocol to tolerate intrusion using multiple disjoint paths in WSNs. Our work also uses multipath routing to tolerate intrusion. However, we specifically consider energy being consumed for intrusion detection, and both CHs and SNs can be compromised for lifetime maximization. In [23] a randomized dispersive multipath routing protocol is proposed to avoid black holes. The randomized multipath routes are dispersive to avoid the black hole and to enhance the probability of at least k out of n shares based on coding theory can reach the receiver. The approach, however, does not consider intrusion detection to detect compromised nodes. Relative to [23] our work also uses multipath routing to circumvent black hole attacks for intrusion tolerance. Moreover, we consider intrusion detection to detect and evict compromised nodes as well as the best rate to invoke intrusion detection to best tradeoff energy consumption vs. security and reliability gain to maximize the system lifetime.

Over the past few years, numerous protocols have been proposed to detect intrusion in WSNs. [7], [11] provide excellent surveys of the subject. In [10], a decentralized rule-based intrusion detection system is proposed by which monitor nodes are responsible for monitoring neighboring nodes. The monitor nodes apply predefined rules to collect messages and raise alarms if the number of failures exceeds a threshold value. Our host IDS essentially follows this strategy, with the flaws of the host IDS characterized by a false positive probability (H_{pfp}) and a false negative probability (H_{pfn}). In [10], however, no consideration is given about bad-mouthing attacks by compromised monitor nodes themselves, so if a monitor node is malicious, it can quickly infect others. In [8], a collaborative approach is proposed for intrusion detection where the decision is based on a majority voting of monitoring nodes. Their work, however, does not consider energy consumption issues associated with a distributed IDS, nor the issue of maximizing the WSN lifetime while satisfying QoS requirements in security, reliability and timeliness. Our voting-based IDS approach extends from [9] with considerations given to the tradeoff between energy loss vs. security and reliability gain due to employment of the voting-based IDS with the goal to prolong the system lifetime.

In general there are two approaches by which energy-efficient IDS can be implemented in WSNs. One approach especially applicable to flat WSNs is for an intermediate node to feedback maliciousness and energy status of its neighbor nodes to the sender node (e.g., the source or sink node) who can then utilize the knowledge to route packets to avoid nodes with unacceptable maliciousness or energy status [17], [24]. Another approach which we adopt in this paper is to use local host-based IDS for energy conservation (with SNs monitoring neighbor SNs and CHs monitoring neighbor CHs only), coupled with voting to cope with node collusion for implementing IDS functions (as discussed in Section III in the paper). Energy efficiency is achieved by applying the optimal detection interval to perform IDS functions. Our solution considers the optimal IDS detection interval that can best balance intrusion accuracy vs. energy consumption due to intrusion detection activities, so as to maximize the system lifetime.

Compared with existing works cited above, our work is distinct in that we consider redundancy management for both intrusion/fault tolerance through multipath routing and intrusion detection through voting-based IDS design to maximize the system lifetime of a HWSN in the presence of unreliable and malicious nodes.

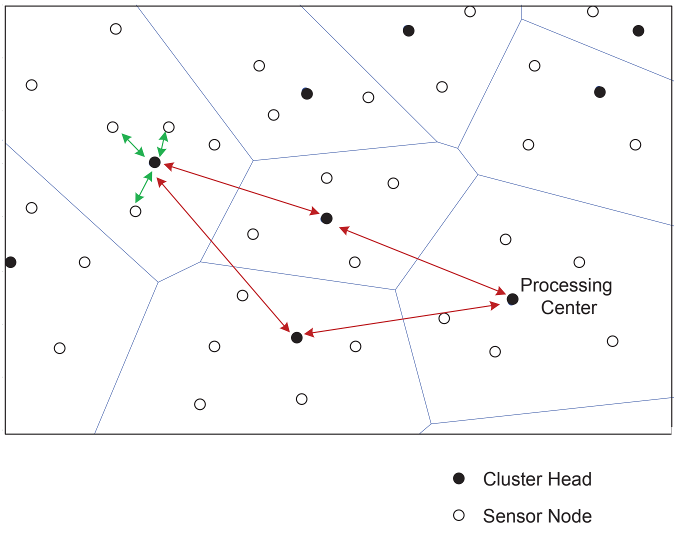

Fig. 1. Source and path redundancy for a heterogeneous WSN.

III. System Model

A HWSN comprises sensors of different capabilities. We consider two types of sensors: 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. While our approach can be applied to any shape of the operational area, for analytical tractability, we assume that the deployment area of the HWSN is of size A^2. CHs and SNs are distributed in the operational area. To ensure coverage, we assume that 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 multi-hop routing. Due to limited energy, a packet is sent hop by hop without using acknowledgment or retransmission [2].

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, as a result, 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* [25] when performing packet routing to disrupt the operation of the network.

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 HWSN 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.

Queries can be issued by a mobile user (while moving) and can be issued anywhere in the HWSN through a nearby CH. A CH which takes a query to process is called a query processing center (PC). Many mission critical applications, e.g., emergency rescue and military battlefield, have a deadline requirement. We assume that each query has a strict timeliness requirement (T_{req}). The query must be delivered within T_{req} seconds; otherwise, the query fails.

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. Fig. 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 [26]. 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.

We assume that geographic routing [18], a well-known routing protocol for WSNs, 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.

We assume that sensors operate in power saving mode (e.g. [27], [28]). 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

[1] for both CHs and SNs.

To preserve confidentiality, we assume that the HWSN executes a pairwise key establishment protocol (e.g., [29], [30]) 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.

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 [17] or AODV for MANETs [18]). 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 [10], [31]–[33]. 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.

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 the paper.

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 sensor nodes. Thus, this is another design tradeoff.

TABLE I
NOTATION OF SYMBOLS

Symbol	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 node (Joule)	input
E_{init}	Initial energy of the HWSN (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
q	node hardware failure probability	input
e_j	Transmission failure probability of node j	input
$N(t)$	Number of nodes in the HWSN at time t	input
$N_{CH}(t)$	Number of CHs in the HWSN at time t	derived
$N_{SN}(t)$	Number of SNs in the HWSN 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 query	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	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 HWSN	output

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.

IV. Probability Model

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 HWSN area. Table I provides the notation used for symbols and their physical meanings. 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:

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

Here $\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)(1-R_q(t_{Q,i+1}))$ accounts for the probability of the system being able to successfully execute i consecutive queries but failing the $i+1^{\text{th}}$ 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.

A. Network Dynamics

Initially at deployment time all nodes (CHs or 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:

$$\begin{aligned} P_c &= 1 - P\{X > t | X > t - T_{IDS}\} \\ &= 1 - \frac{P\{X > t, X > t - T_{IDS}\}}{P\{X > t - T_{IDS}\}} \\ &= 1 - \frac{1 - F_c(t)}{1 - F_c(t - T_{IDS})} \end{aligned} \tag{2}$$

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 probability 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 $t_{I,i-1}$ as follows:

$$\begin{aligned} n_{good}(t_{I,i}) &= n_{good}(t_{I,i-1}) - n_{good}(t_{I,i-1}) \times P_c \\ n_{bad}(t_{I,i}) &= n_{bad}(t_{I,i-1}) + n_{good}(t_{I,i-1}) \times P_c \end{aligned} \tag{3}$$

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 HWSN. 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,

$$\overline{n_{good}(t_{I,i})} = n_{good}(t_{I,i}) - n_{good}(t_{I,i}) \times P_{fp} \tag{5}$$

With $n_{good}(t)$ and $n_{bad}(t)$ in hand, we estimate the system-level false positive probability (P_{fp}) and false negative probability (P_{fn}) at time t as a resulting of executing the voting-based IDS as Eq. (4) at the top of the next page, 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 Eq. (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 Eq. (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 HWSN. 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

$$P_{fp} \text{ or } P_{fn} \equiv \sum_{i=0}^{m-m_{maj}} \left[\frac{C\begin{pmatrix} n_{bad} \\ m_{maj}+i \end{pmatrix} \times C\begin{pmatrix} n_{good} \\ m-(m_{maj}+i) \end{pmatrix}}{C\begin{pmatrix} n_{bad}+n_{good} \\ m \end{pmatrix}} \right] + \sum_{i=0}^{m-m_{maj}} \left[\frac{C\begin{pmatrix} n_{bad} \\ i \end{pmatrix} \times \sum_{j=m_{maj-1}}^{m-1} C\left[\begin{pmatrix} n_{good} \\ j \end{pmatrix} \times \omega^{i} \times C\begin{pmatrix} n_{good-j} \\ m-i-j \end{pmatrix} \times (1-\omega)^{m-i-j}\right]}{C\begin{pmatrix} n_{bad}+n_{good} \\ m \end{pmatrix}} \right] \quad (4)$$

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 assess 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.

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,

$$\overline{n_{bad}(t_{I,i})} = n_{bad}(t_{I,i}) - n_{bad}(t_{I,i}) \times (1-P_{fn}) \quad (6)$$

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:

$$Q_{c,j}(t_{I,i}) = \frac{\overline{n_{bad}(t_{I,i})}}{\overline{n_{bad}(t_{I,i})} + \overline{n_{good}(t_{I,i})}} \quad (7)$$

$Q_{c,j}(t)$ 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:

$$n(t_{I,i}) = \overline{n_{bad}(t_{I,i})} + \overline{n_{good}(t_{I,i})} \quad (8)$$

$$\lambda(t_{I,i}) = \frac{n(t_{I,i})}{\pi r^2} \quad (9)$$

With Eq. (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 Eq. (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:

$$N_{CH}(t_{I,i}) = \lambda_{CH}(t_{I,i}) \times A^2 \quad (10)$$

$$N_{SN}(t_{I,i}) = \lambda_{SN}(t_{I,i}) \times A^2 \quad (11)$$

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:

$$r_{CH}(t_{I,i}) = \sqrt{\frac{m_p}{\pi f \lambda_{CH}(t_{I,i})}}; r_{SN}(t_{I,i}) = \sqrt{\frac{m_s}{\pi f \lambda_{SN}(t_{I,i})}} \quad (12)$$

B. 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-CH} be the *expected* distance between a SN (selected to report sensor readings) and its CH and d_{CH-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:

$$S_{req} = \frac{d_{SN-CH} + d_{CH-PC}}{T_{req}} \quad (13)$$

We follow the Voronoi theory [34] 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 $d_{SN-CH} = 1/2\left(\lambda_{CH}\right)^{1/2}$. 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 HWSN, d_{CH-PC} essentially is the average distance between any two CHs in the HWSN. 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 $d_{CH-PC} = 0.382A$. With the knowledge of d_{SN-CH} and d_{CH-PC}, we can estimate the average numbers of hops to forward data from a SN to the source CH, denoted by N^h_{SC}, and the average numbers of hops to forward data from the source CH to the PC, denoted by N^h_{CP}. Specifically,

$$N^h_{SC} = d_{SN-CH}/r_{SN}; N^h_{cp} = d_{CH-PC}/r_{CH}. \tag{14}$$

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 [2]. 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:

$$Q^{CH}_{t,jk}\left(t\right) = cdf\left(S_{jk} \le S_{req}\right) = \frac{S_{req} - a}{b - a} \tag{15}$$

The second source of failure is due to node failure or channel failure. Let Q_{rj} 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:

$$Q^{CH}_{r,j} = 1 - \left[\left(1 - q\right)\left(1 - e_j\right)\right] \tag{16}$$

The third source of failure is due to node j being compromised and thus the packet is dropped. We make use of $Q^{CH}_{c,j}\left(t\right)$ derived earlier in Eq. (7). By combining these three failure probabilities we obtain $Q^{CH}_{rtc,jk}\left(t\right)$, 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:

$$Q^{CH}_{rtc,jk}\left(t\right) = 1 - \left[\left(1 - Q^{CH}_{r,j}\right)\left(1 - Q^{CH}_{t,jk}\left(t\right)\right)\left(1 - Q^{CH}_{c,j}\left(t\right)\right)\right] \tag{17}$$

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:

$$\theta^{CH}_j = 1 - \prod_{k=1}^{f \times n_{CH}} Q^{CH}_{rtc,jk}\left(t\right) \tag{18}$$

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 Eq. (8).

Since on average there will be N^h_{CP} 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^h_{CP} hops. Consequently, the probability of a single path between the source CH and the PC being able to deliver data successfully is given by:

$$\begin{aligned}\Theta\left(N^h_{CP}\right) \;&=\; \left(\prod_{j=1}^{N^h_{CP}-1} \theta^{CH}_j\right) \times \left(1 - Q^{CH}_{rtc,N^h_{CP}\left(N^h_{CP}+1\right)}\right)\\ &\quad \times \left[\left(1 - q\right)\left(1 - Q^{CH}_{c,PC}\right)\right]\end{aligned} \tag{19}$$

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 Eq. (20) at the top of the next page.

Following the same derivation to Eq. (19), the success probability of a single path from a SN to its CH is given by:

$$\Theta\left(N^h_{SC}\right) = \left(\prod_{j=1}^{N^h_{SC}-1} \theta^{SN}_j\right) \times \left(1 - Q^{SN}_{rt,N^h_{SC}\left(N^h_{SC}+1\right)}\right) \tag{21}$$

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:

$$Q^{m_s}_{fs} = \prod_{i=1}^{m_s}\left[1 - \Theta_i\left(N^h_{SC}\right)\right] \tag{22}$$

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:

$$Q_f = 1 - \left(1 - Q^{m_p}_{fp}\right)\left(1 - Q^{m_s}_{fs}\right) \tag{23}$$

Therefore, the query success probability is given by:

$$R_q = 1 - Q_f \tag{24}$$

Note that in the above derivation we omit time for brevity. More precisely, R_q derived above should be $R_q\left(t_{Q,i}\right)$ since the query success probability is a function of time, depending on the node count (Eq. (8)) and population density (Eq. (9)) at the i^{th} query's execution time (i.e., at time $t_{Q,i}$).

$$\begin{aligned} Q_{fp}^{m_p} &= 1-\theta_1^{CH}+ \\ &\sum_{|I|<m_p}\left[\begin{array}{l}\left\{\left[\prod_{i\in I}\left(1-Q_{rtc,1i}^{CH}\right)\right]\times\left(\prod_{i\notin I}\left(Q_{rtc,1i}^{CH}\right)\right)\right\} \\ \times\left\{\prod_{i\in I}\left[1-\Theta_i\left(N_{CP}^h-1\right)\right]\right\}\end{array}\right]+ \\ &\sum_{|I|\geq m_p}\left[\begin{array}{l}\left\{\left[\prod_{i\in I}\left(1-Q_{rtc,1i}^{CH}\right)\right]\times\left(\prod_{i\notin I}\left(Q_{rtc,1i}^{CH}\right)\right)\right\} \\ \times\left\{\prod_{i\in M,M\subseteq I,|M|=m_p}\left[1-\Theta_i\left(N_{CP}^h-1\right)\right]\right\}\end{array}\right] \end{aligned} \tag{20}$$

C. 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:

$$E_{init}=\sum_{i=1}^{\alpha N_q}E_{IDS}\left(t_{I,i}\right)+\sum_{i=1}^{\beta N_q}E_{clustering}\left(t_{C,i}\right)+\sum_{i=1}^{N_q}E_q\left(t_{Q,i}\right) \tag{25}$$

Below we outline how to calculate $E_{IDS}\left(t_{I,i}\right)$, $E_{clustering}\left(t_{C,i}\right)$ and $E_q\left(t_{Q,i}\right)$. We first estimate the amount of 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 [1]:

$$E_t=n_b\left(E_{elec}+E_{amp}r^x\right) \tag{26}$$

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:

$$E_R=n_bE_{elec} \tag{27}$$

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

$$E_q\left(t_{Q,i}\right)=E_q^{CH}\left(t_{Q,i}\right)+E_q^{SN}\left(t_{Q,i}\right) \tag{28}$$

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.,

$$\begin{aligned} E_q^{CH}\left(t_{Q,i}\right)&=\left\{E_T^{CH}+n_{CH}\left(t_{Q,i}\right)E_R^{CH}\right\}+ \\ &\left\{m_p\left(N_{CP}^h-1\right)\left[E_T^{CH}+n_{CH}\left(t_{Q,i}\right)E_R^{CH}\right]\right\} \end{aligned} \tag{29}$$

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. We note that the number of neighbor CHs at time t, $n_{CH}\left(t\right)=\lambda_{CH}\left(t\right)\times\pi r_{CH}^2$ by Eq. (9), depends on the CH population density at time t, i.e., $\lambda_{CH}\left(t\right)$.

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.,

$$E_q^{SN}\left(t_{Q,i}\right)=m_sN_{SC}^h\left\lfloor E_T^{SN}+n_{SN}\left(t_{Q,i}\right)E_R^{SN}\right\rfloor \tag{30}$$

For clustering, the system would consume energy for broadcasting the announcement message and for the cluster-join process. There will be $N_{CH}\left(t_{c,i}\right)$ CHs each broadcasting the announcement message at time $t_{c,i}$, for a total energy consumption of $N_{CH}\left(t_{c,i}\right)\times E_T^{CH}$. There will be $N_{SN}\left(t_{c,i}\right)$ SNs each on average receiving $n_{CH}\left(t_{c,i}\right)$ announcement messages from CHs within radio broadcast range $r_{CH}\left(t_{c,i}\right)$ at time $t_{c,i}$, for a total energy consumption of $N_{SN}\left(t_{c,i}\right)\times n_{CH}\left(t_{c,i}\right)\times E_R^{SN}$. 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 $N_{SN}\left(t_{c,i}\right)$ SNs each transmitting a join packet to the CH it selects to join, for a total energy consumption of $N_{SN}\left(t_{c,i}\right)\times E_R^{CH}$ for packet reception by the selected CHs plus $N_{SN}\left(t_{c,i}\right)\times N_{SC}^h\left(E_T^{SN}+n_{SN}\left(t_{c,i}\right)E_R^{SN}\right)$ 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 $E_{clustering}\left(t_{c,i}\right)$, is given by:

$$\begin{aligned} E_{clustering}\left(t_{c,i}\right)&=N_{CH}\left(t_{c,i}\right)\times E_T^{CH}+N_{SN}\left(t_{c,i}\right) \\ &\times n_{CH}\left(t_{c,i}\right)\times E_R^{SN} \\ &+N_{SN}\left(t_{c,i}\right)\times E_R^{CH}+N_{SN}\left(t_{c,i}\right) \\ &\times N_{SC}^h\left(E_T^{SN}+n_{SN}\left(t_{c,i}\right)E_R^{SN}\right) \end{aligned} \tag{31}$$

```
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
```

Fig. 2. CH execution for dynamic redundancy management.

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:

$$\begin{aligned} E_{IDS}(t_{I,i}) &= E_{IDS}^{CH}(t_{I,i}) + E_{IDS}^{SN}(t_{I,i}) \\ E_{IDS}^{CH}(t_{I,i}) &= N_{CH}(t_{I,i-1})[m(m-1)] \\ &\quad \left[E_T^{CH} + n_{CH}(t_{I,i-1}) E_R^{CH}\right] \\ E_{IDS}^{SN}(t_{I,i}) &= N_{SN}(t_{I,i-1})[m(m-1)] \\ &\quad \left[E_E^{SN} + n_{SN}(t_{I,i-1}) E_R^{SN}\right] \end{aligned} \tag{32}$$

After we obtain $E_{IDS}(t_{I,i})$, $E_{clustering}(t_{C,i})$ and $E_q(t_{Q,i})$ from Eqs. (32), (31) and (28), respectively, we calculate N_q from Eq. (25). The knowledge of N_q along with $R_q(t_{Q,i})$ in Eq. (24) allows us to calculate the system MTTF given by Eq. (1).

V. Algorithm for Dynamic Redundancy Management of 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 $(\lambda_{SN}/\lambda_{CH})$, radio range (r_{SN}/r_{CH}), and SN/CH capture rate (λ_c).

Our algorithm for dynamic redundancy management of multipath routing is distributed in nature. Figs. 2 and 3 describe the CH and SN execution protocols, respectively, for managing multipath routing for intrusion tolerance to maximize the system lifetime. They specify control actions taken by individual SNs and CHs in response to dynamically changing environments.

```
1:  SN Execution:
2:    Get next event
3:    if event is $T_D$ timer then
4:          determine radio range to maintain SN connectivity
              within a cluster
5:    else if event is control packet arrival from CH then
6:          Change the optimal settings of $T_{IDS}$, and m
7:    else if event is $T_{clustering}$ timer then
8:          perform clustering
9:    else if event is $T_{IDS}$ timer then
10:         For each neighbor SN
11:             if selected as a voter then
12:                   execute votingbased intrusion detection
13:   else // event is data packet arrival
14:         follow multipath routing protocol design to route
              the data packet
```

Fig. 3. SN execution 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 in both Fig. 2 for a CH and Fig. 3 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 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 CH radio range to maintain CH connectivity (line 4 in Fig. 2); (b) determining T_{IDS}, m, m_s, and m_p (line 5 in Fig. 2) 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 SN connectivity within a cluster (line 4 in Fig. 3). The action taken by a SN upon receiving the control packet from its CH is to update the new T_{IDS} and m settings (line 6 in Fig. 3) 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 in Fig. 2 for a CH and lines 9–12 in Fig. 3 for a SN.

When a CH acting as a query processing center (PC) receives a query from a user, it triggers 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 in Fig.2. When a data packet arrival event occurs, each node simply follows the prescribed multipath routing protocol to route the packet (lines 15–16 in Fig. 2 and lines 13–14 in Fig. 3). 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 in Fig. 2 and lines 7–8 in Fig. 3) for periodic cluster formation.

The cost of executing the dynamic redundancy management algorithm described above, including periodic clustering,

TABLE II
INPUT PARAMETER VALUES CHARACTERIZING A QUERY-BASED CLUSTERED HWSN

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_{pfp}, H_{pfn}	[0.01-0.05]

periodic intrusion detection, and query processing through multipath routing, in terms of energy consumption has been considered in Section IV-C 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 in Fig. 2) can be eliminated if the CH notifies the optimal settings to its SNs at the time periodic clustering is performed.

VI. PERFORMANCE EVALUATION

In this section, we present numerical data obtained as a result of applying Eq. (1). Table II 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 $(200m \times 200m)$. Nodes are distributed in the area following a Poisson process with density $\lambda_{SN} = 30$ nodes/$(20 \times 20\ m^2)$ and $\lambda_{CH} = 1$ node/$(20 \times 20\ m^2)$ 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 [1], [35]. 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)

```
Input: Table II input parameters
Output: optimal MTTF, 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:      $E_{init}^{SN} \leftarrow N_{SN}(t) \times E_0^{SN}$, $E_{init}^{CH} \leftarrow N_{CH}(t) \times E_0^{CH}$   where $t = 0$
5:      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 $t = 0$
6:      Compute arrival time for next clustering,
            query, and IDS events at $t = 0$
7:      while [ $E_{init}^{SN} > E_{threshold}^{SN}$ and $E_{init}^{CH} > E_{threshold}^{CH}$ and
8:          $f * n_{SN}$ has $m_s$ nodes and $f * n_{ch}$ has $m_p$ nodes] do
9:        $ev \leftarrow$ next event
10:       if $ev$ is clustering event then
11:         $E_{init}^{SN} = E_{init}^{SN} - E_{clustering}^{SN}$, $E_{init}^{CH} = E_{init}^{CH} - E_{clustering}^{CH}$
12:       else if $ev$ is query event then
13:         $num_q \leftarrow num_q + 1$
14:         $E_{init}^{SN} = E_{init}^{SN} - E_q^{SN}$, $E_{init}^{CH} = E_{init}^{CH} - E_q^{CH}$
15:         if $num_q = 1$ then //first query
16:           $rq\_muls \leftarrow rq\_muls \times R_q$
17:           $temp \leftarrow num_q \times rq\_muls$
18:         else //terminate previous query
19:           $tempMttf \leftarrow tempMttf + temp \times (1 - R_q)$
20:           $rq\_muls \leftarrow rq\_muls \times R_q$
21:           $temp \leftarrow num_q \times rq\_muls$
22:       else // ev is an IDS event
23:         Update distribution of good and bad nodes
24:         Compute Pfp and Pfn
25:           $E_{init}^{SN} = E_{init}^{SN} - E_{IDS}^{SN}$, $E_{init}^{CH} = E_{init}^{CH} - E_{IDS}^{CH}$
26:       Remove Bad caught and Good misidentified nodes
27:         Compute $Q_c^{SN}, Q_c^{CH}$
28:          Update $\lambda_{SN}, \lambda_{CH}, N_{SN}, N_{CH}, r_{SN}, r_{CH}$
29:          Update $R_q, E_{clustering}^{SN}, E_{clustering}^{CH}, E_q^{SN}, E_q^{CH}$
30:          $tempMttf \leftarrow tempMttf + temp$
31:          $Mttf \leftarrow tempMttf$
32:          $N_q \leftarrow num_q$
33:     if $Mttf > optimalMttf$ then
34:       $optimalMttf \leftarrow Mttf$
35:       optimal $(m_p, m_s) \leftarrow (m_p, m_s)$
36:         return $optimalMttf$ and optimal $(m_p, m_s)$
```

Fig. 4. Computational procedure to determine optimal (m_p, m_s) for maximizing MTTF.

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 [10].

Fig. 4 shows a high level description of the computational procedure to determine the optimal redundancy level (m_p, m_s) for maximizing MTTF. The MTTF Eq. (Eq. (1)) is embedded on lines 15–21 and 30–31 in Fig. 4. 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

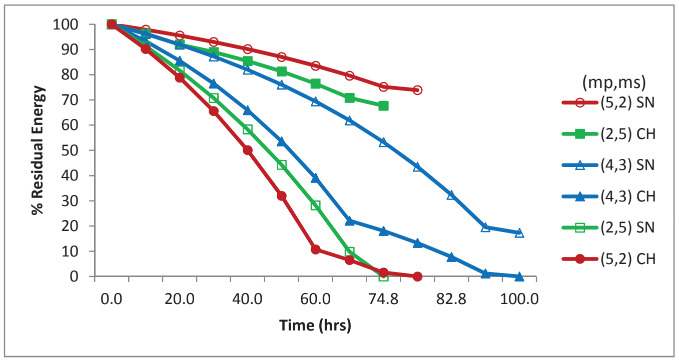

Fig. 5. Effect of (m_p, m_s) on energy of CHs and SNs.

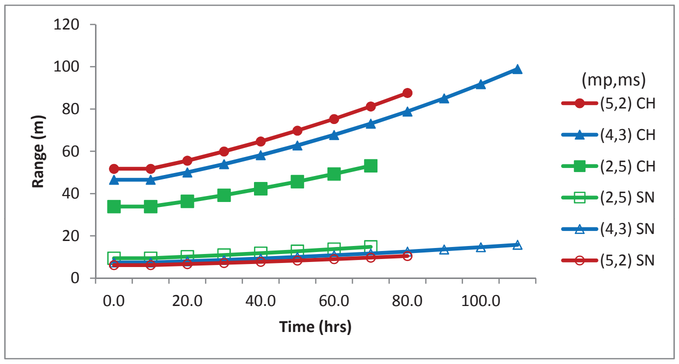

Fig. 7. Effect of (m_p, m_s) on radio range of CHs and SNs.

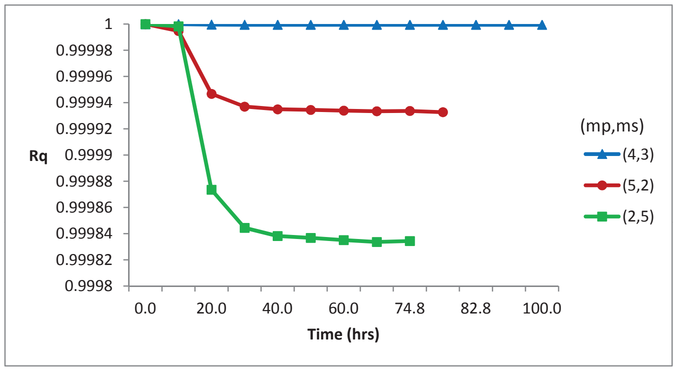

Fig. 6. Effect of (m_p, m_s) on query reliability (R_q).

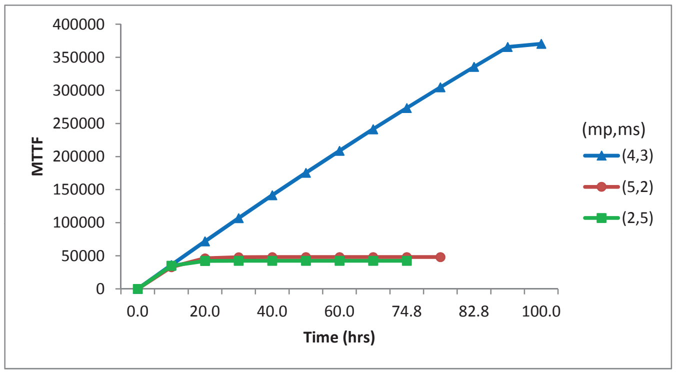

Fig. 8. Effect of (m_p, m_s) on MTTF.

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 II (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.

Below we present numerical data to provide evidence of the correctness of our analysis and to provide physical interpretations of the results. 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. Figs. 5–7 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 Fig. 5, 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. Fig. 6 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.

The system dynamically adjusts the radio range of CHs and SNs to maintain network connectivity based on Eq. (10) as nodes are being removed from the system because of failure or eviction. Fig. 7 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, Figs. 5–7 indicate that there exist an optimal combination of (m_p, m_s) that will maximize the system lifetime. Fig. 8 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.

The correctness of our protocol design is evidenced by the effect of T_{comp}, m, and T_{IDS} on optimal (m_p, m_s). Figs. 9 and 10 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 Fig. 10, $m = 7$ 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

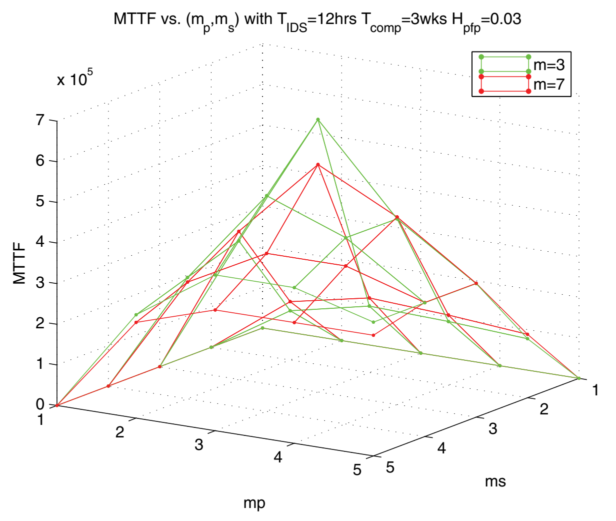

Fig. 9. MTTF vs. (m_p, m_s) under low capture rate.

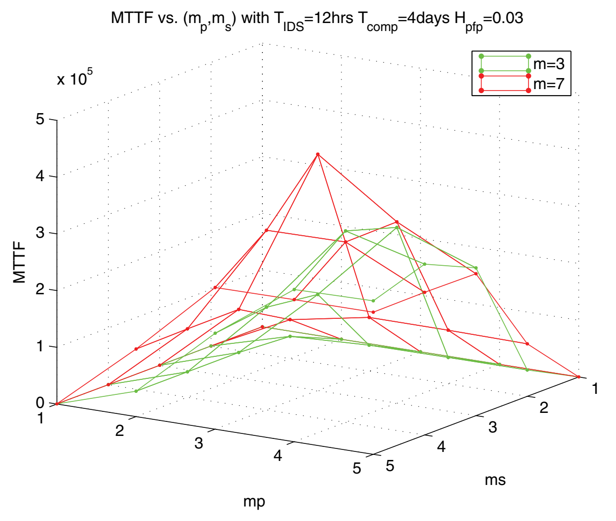

Fig. 10. MTTF vs. (m_p, m_s) under high capture rate.

used. By comparing these two graphs, we observe a trend that as the capture rate increases (i.e., going from Fig. 9 to Fig. 10), 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.

Another trend exhibited in Figs. 9 and 10 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 Fig. 10, when $m = 3$, the optimal $(m_p, m_s) = (4, 4)$ while when $m = 7$ the optimal $(m_p, m_s) = (3, 3)$.

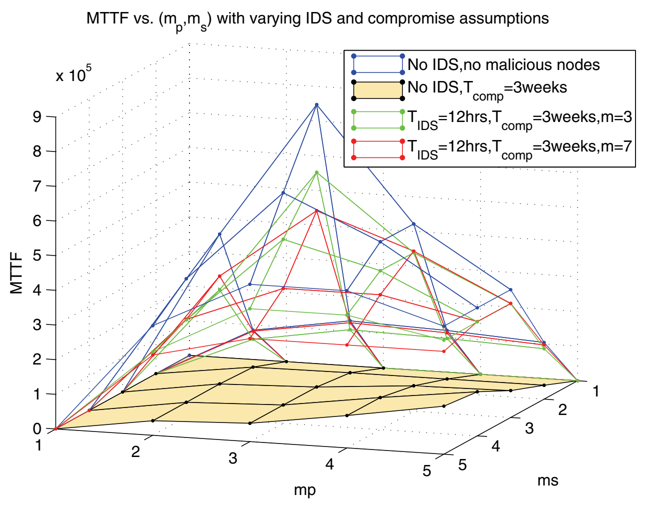

Fig. 11. MTTF vs. (m_p, m_s) for three cases.

TABLE III
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)

In Fig. 11, we compare MTTF vs. (m_p, m_s) under three cases: (a) there are no malicious nodes and no intrusion detection, considering using multipath routing for fault tolerance only as in [8] (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) value to maximize MTTF is (3, 3) and the MTTF is the highest among three cases because there are no malicious nodes. When there are malicious nodes, however, the optimal (m_p, m_s) value becomes (5, 5) 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. Fig. 11 reflects the IDS case in Fig. 9.

We summarize the effect of T_{comp} and m on the optimal (m_p, m_s) value in Table III and the effect of T_{comp} on the optimal m value at which MTTF is maximized in Table IV. Table III 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 IV shows that as the capture rate increases (i.e., a smaller T_{comp}), the optimal

TABLE IV
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

TABLE V
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

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.

We further ascertain the correctness of our analysis by the effect of T_{IDS} (IDS detection interval) on MTTF. Figs. 12 and 13 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 Fig. 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 Figs. 12 and 13 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 IV earlier.

Lastly, we examine the sensitivity of the optimal T_{IDS} to the capture rate. Fig. 14 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 Fig. 14 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

TABLE VI
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

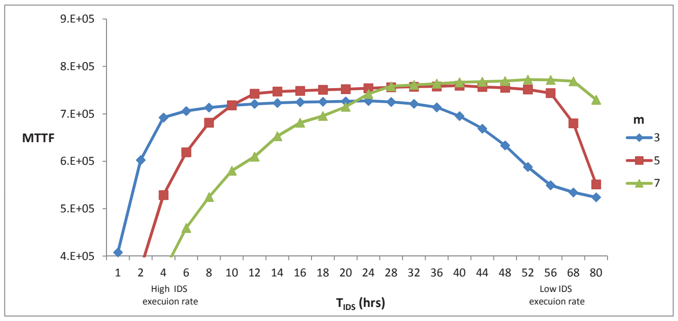

Fig. 12. Effect of T_{IDS} on MTTF under low capture rate.

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

VII. CONCLUSIONS

In this paper we performed a tradeoff analysis of energy consumption vs. QoS gain in reliability, timeliness, and security for redundancy management of clustered heterogeneous 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

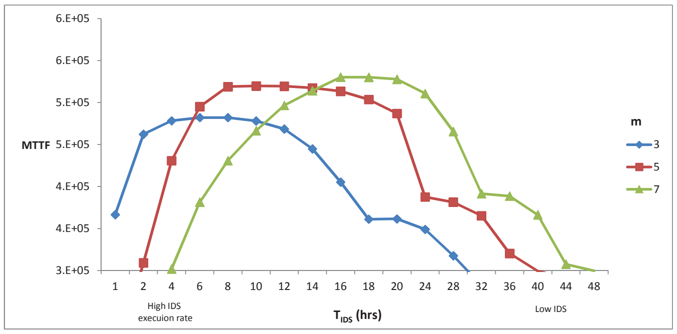

Fig. 13. Effect of T_{IDS} on MTTF under high capture rate.

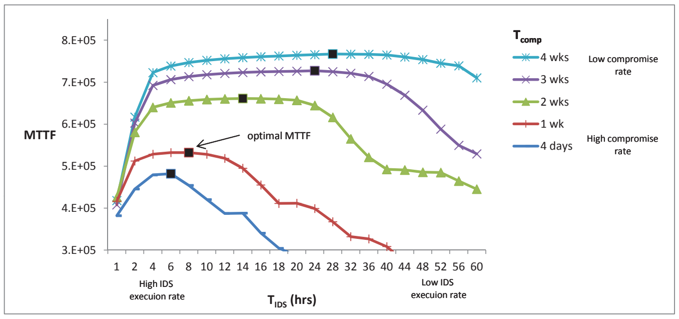

Fig. 14. Effect of capture rate on optimal T_{IDS}.

apply the best design parameter settings at runtime in response to environment changes to prolong the system lifetime.

For future work, we plan to explore more extensive malicious attacks in addition to packet dropping and bad mouthing attacks, each with different implications to energy, security and reliability, and investigate intrusion detection and multipath routing based tolerance protocols to react to these attacks. Another direction is to consider smart and insidious attackers which 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. Lastly, we plan to investigate the use of trust/reputation management [33], [36], [37] to strengthen intrusion detection through "weighted voting" leveraging knowledge of trust/reputation of neighbor nodes, as well as to tackle the "what paths to use" problem in multipath routing decision making for intrusion tolerance in WSNs. In situations where concurrent query traffic is heavy, we plan to explore trust-based admission control [38]–[40] to optimize application performance.

Acknowledgement

This material is supported in part by the U.S. Army Research Office under contract number W911NF-12-1-0445.

References

[1] 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.

[2] 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.

[3] 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. Dependable Secure Computing*, vol. 8, no. 2, pp. 161–176, 2011.

[4] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, "Exploiting heterogeneity in sensor networks," in *Proc. 2005 IEEE Conf. Computer Commun.*, vol. 2, pp. 878–890.

[5] H. M. Ammari and S. K. Das, "Promoting heterogeneity, mobility, and energy-aware Voronoi diagram in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 7, pp. 995–1008, 2008.

[6] X. Du and F. Lin, "Improving routing in sensor networks with heterogeneous sensor nodes," in *Proc. 2005 IEEE Veh. Technol. Conf.*, pp. 2528–2532.

[7] S. Bo, L. Osborne, X. Yang, and S. Guizani, "Intrusion detection techniques in mobile ad hoc and wireless sensor networks," *IEEE Wireless Commun. Mag.*, vol. 14, no. 5, pp. 560–563, 2007.

[8] I. Krontiris, T. Dimitriou, and F. C. Freiling, "Towards intrusion detection in wireless sensor networks," in *Proc. 2007 European Wireless Conf.*

[9] 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 Trans. Reliab.*, vol. 59, no. 1, pp. 231–241, 2010.

[10] 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," in *Proc. 2005 ACM Workshop Quality Service Security Wireless Mobile Netw.*

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

[12] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Trans. Programming Languages Syst.*, vol. 4, no. 3, pp. 382–401, 1982.

[13] 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.

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

[15] K. D. Kang, K. Liu, and N. Abu-Ghazaleh, "Securing geographic routing in wireless sensor networks," in *Proc. 2006 Cyber Security Conf. Inf. Assurance*.

[16] 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.

[17] Y. Lan, L. Lei, and G. Fuxiang, "A multipath secure routing protocol based on malicious node detection," in *Proc. 2009 Chinese Control Decision Conf.*, pp. 4323–4328.

[18] D. Somasundaram and R. Marimuthu, "A multipath reliable routing for detection and isolation of malicious nodes in MANET," in *Proc. 2008 Int. Conf. Computing, Commun. Netw.*, pp. 1–8.

[19] H. Su and X. Zhang, "Network lifetime optimization for heterogeneous sensor networks with mixed communication modes," in *Proc. 2007 IEEE Wireless Commun. Netw. Conf.*, pp. 3158–3163.

[20] I. Slama, B. Jouaber, and D. Zeghlache, "Optimal power management scheme for heterogeneous wireless sensor networks: lifetime maximization under QoS and energy constraints," in *Proc. 2007 Int. Conf. Netw. Services*, pp. 69–69.

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

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

[23] 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.

[24] Y. X. Jiang and B. H. Zhao, "A secure routing protocol with malicious nodes detecting and diagnosing mechanism for wireless sensor networks," in *Proc. 2007 IEEE Asia-Pacific Service Comput. Conf.*, pp. 49–55.

[25] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," in *Proc. 2003 IEEE Int. Workshop Sensor Netw. Protocols Appl.*, pp. 113–127.

[26] B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: reliable information forwarding using multiple paths in sensor networks," in *Proc. 2003 IEEE Conf. Local Computer Netw.*, pp. 406–415.

[27] G. Bravos and A. G. Kanatas, “Energy consumption and trade-offs on wireless sensor networks,” in *Proc. 2005 IEEE Int. Symp. Pers., Indoor Mobile Radio Commun.*, pp. 1279–1283.

[28] S. Qun, “Power management in networked sensor radios a network energy model,” in *Proc. 2007 IEEE Sensors Appl. Symp.*, pp. 1–5.

[29] C. Haowen and A. Perrig, “PIKE: peer intermediaries for key establishment in sensor networks,” in *Proc. 2005 IEEE Conf. Computer Commun.*, pp. 524–535.

[30] S. Zhu, S. Setia, and S. Jajodia, “LEAP: efficient security mechanisms for large-scale distributed sensor networks,” in *Proc. 2003 ACM Conf. Computer Commun. Security*.

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

[32] S. Rajasegarar, C. Leckie, and M. Palaniswami, “Anomaly detection in wireless sensor networks,” *IEEE Wireless Commun. Mag.*, vol. 15, no. 4, pp. 34–40, 2008.

[33] F. Bao, I. R. Chen, M. Chang, and J. Cho, “Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection,” *IEEE Trans. Netw. Service Manage.*, vol. 9, no. 2, pp. 161–183, 2012.

[34] S. Bandyopadhyay and E. J. Coyle, “An energy efficient hierarchical clustering algorithm for wireless sensor networks,” in *Proc. 2003 Conf. IEEE Computer Commun.*, pp. 1713–1723.

[35] 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.

[36] C. J. Fung, Z. Jie, I. Aib, and R. Boutaba, “Dirichlet-based trust management for effective collaborative intrusion detection networks,” *IEEE Trans. Netw. Service Manage.*, vol. 8, no. 2, pp. 79–91, 2011.

[37] S. Ozdemir, “Secure and reliable data aggregation for wireless sensor networks,” in *Proc. 2007 Int. Conf. Ubiquitous Comput. Syst.*

[38] 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.

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

[40] 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.

Hamid Al-Hamadi received his bachelor degree in information technology from Griffith University, Brisbane, Australia, in 2003, and a master’s of information technology from Queensland University of Technology, Brisbane, Australia, in 2005. His research interests include security, computer networks, wireless networks, wireless sensor networks, and reliability and performance analysis. Currently, he is pursuing his Ph.D. degree in the Computer Science Department at Virginia Tech.

Ing-Ray Chen received the B.S. degree from National Taiwan University, Taipei, Taiwan, and the M.S. and Ph.D. degrees in computer science from the University of Houston. He is a professor in the Department of Computer Science at Virginia Tech. His research interests include mobile computing, wireless systems, security, trust management, data management, real-time intelligent systems, and reliability and performance analysis. Dr. Chen currently serves as an editor for IEEE COMMUNICATIONS LETTERS, the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, *Wireless Personal Communications*, *Wireless Communications and Mobile Computing*, *The Computer Journal*, and *Security and Network Communications*. He is a member of the IEEE and ACM.