Report Dropping and Tampering Detection in Sensor Networks: Enhancing Data Reliability

Mohit Virendra*, Qi Duan, Shambhu Upadhyaya

{virendra, qiduan, shambhu}@ cse.buffalo.edu
Department of Computer Science and Engineering,
State University of New York at Buffalo, Buffalo, NY 14260

***Abstract*—Achieving communication dependability in Sensor networks is difficult due to limitations on node power and wireless channel unreliability. This problem is further compounded by hard-to-detect attacks such as data-report tampering by malicious Cluster Heads, and dropping of data-reports by intermediate nodes en route from the CH to the Base Station (BS). Detecting these attacks coupled with node failure detection may result in achieving enhanced reliability (semi-reliability) in sensor networks data communication, depending on the granularity and accuracy of diagnosis. In this paper, we introduce consistency checking-based proxy-report schemes rooted at the BS for detecting such data report dropping and tampering. The merit of the solution methodology lies in its simplicity, comprehensive nature and practical usability. We consider a cluster-based sensor network model where sensor nodes within a cluster send their data to the CH which aggregates and forwards this data to the Base Station (BS) in the form of data reports. Our schemes use proxy reports to verify the accuracy and receipt of data reports sent by a CH to the BS. Proxy reports are periodically sent along non-primary paths, enabling the BS to detect any tampering or dropping of the data reports by malicious nodes on the primary paths with a certain probability. Proxy report transmission involves path-reuse and piggybacking on data reports from other clusters, making the scheme lightweight and resulting in minimal additional control and energy overhead for the energy constrained sensor nodes. Simulation results show the robustness of our schemes against random and patterned node failures.**

***Index Terms*— Malicious Cluster Heads, Packet Dropping, Report Tampering, Security, Sensor Networks.**

I. INTRODUCTION

Wireless sensor networks have found increased use in data collection and aggregation. They are especially important in hostile terrain or in wartime applications whereby aerial scattering (or any other similar installation) of sensor nodes and strategic positioning of Base Stations (BSs) allows rapid network deployment.

The data collected using sensor networks in such scenarios may be critical for decision making (e.g., tracking moving enemy targets). If this data is tampered with or destroyed before it reaches the BS, the underlying decision making process may be significantly affected. Thus, it is desirable to have some degree of reliability or dependability in data communication for such networks. However, achieving communication dependability in sensor networks is difficult due to (a) limitations on node power and wireless channel unreliability, and (b) Byzantine failures and possible attacks on the network.

Since sensor networks are often deployed in unattended or hostile environments, constraints mentioned in (a) above may magnify the impact of attacks and failures when compared with other wireless networks. For example, an adversary can compromise some nodes and the compromised nodes may deliberately drop data reports selectively, tamper with the reports, misroute them, or, inject false data into the network. These attacks result in inaccurate data aggregation at the BS and depletion of

the nodes' battery power. Often due to the adverse ambient conditions, detecting such attacks or Byzantine failures is difficult. However, detection is crucial in mission critical operations where the BS may have to make decisions based on the received data reports.

A. Motivation

Current sensor network security research mainly concentrates on Key Management, Secure Broadcast, Sybil (impersonation by assuming multiple identities) Attacks and False Data Injection Attacks [10], [11], [12], [13]. These schemes address key management in the network, and re-keying protocols if some nodes are compromised. Though most security schemes for sensor networks assume the reports sent to the base station (BS) to be end-to-end encrypted by the CHs, these reports may be: (a) tampered with by malicious CHs, and this tampering may go undetected if the cluster size is large, or (b) selectively dropped by malicious CHs, or by other intermediate nodes in the path to the BS (or the reports may be misrouted to a passively-present adversary). For example, in the protocols mentioned above, non-receipt of the reports at the BS due to report dropping (or misrouting) would be undetected. These schemes would not be able to differentiate between reports dropped as a result of in-network processing (or passive participation) and maliciously dropped reports. This is further discussed in related work in Sec. IX.A. Conventional networking protocols for reliable data delivery (e.g., TCP [14]) would not be viable in the sensor networks domain due to expensive communication, processing, and storage overheads. Besides, these protocols focus on communication reliability rather than security, and do not assume the presence of an adversary. Sec. IX.B. outlines the related research in the field and explains how schemes presented in this paper complement it.

B. Problem Statement

Thus, there is lack of detection and corrective techniques for: (a) report tampering by malicious CHs for large (more than 1-hop) clusters, and (b) malicious report-dropping by CHs or other nodes in the path to the BS. This is due to the absence of (i) any acknowledgement style response from the BS, primarily due to overhead and management issues, and (ii) verification of reports received at the BS (except for any error correction codes that might be appended to the report by the CH itself). This is especially relevant in mission critical applications where admissibly reliable data delivery is desirable, or where the BS may have to make decisions based on the received data reports.

This paper addresses detection of (a) report tampering and report dropping attacks by malicious CHs, and (b) report dropping attacks by any intermediate nodes on the route to the BS. The paper presents a lightweight BS-based monitoring scheme with minimal overhead that incorporates periodic verification of the data reports (checking for consistency between the main report and a proxy report) sent by the CHs to the BS. It also discusses in detail the design issues for such a scheme in the sensor network domain.

The merit of the solution methodology lies in this simplicity, comprehensive nature and practical usability. This paper is unique in that it considers sensor networks with varied cluster sizes and different data formats, discussing the full implementation details irrespective of the topologies. Using our schemes in conjunction with schemes to prevent false data injection [10], [12], could result in enhancing the dependability and achieving enhanced reliability (semi-reliability) in sensor networks data communication, depending on the granularity and accuracy of attack detection.

C. Paper Organization

The rest of this paper is organized as follows: Section II discusses the network and security assumptions. Section III gives a high level description of our scheme. Section IV presents the BS-Cluster communication model. Section V discusses the design

considerations for various report formats. Section VI describes report verification and consistency-check techniques adopted by the BS. In Section VII we analyze several key security features of our scheme. Section VIII presents experimental results. Related work in literature is outlined in Section IX. Finally, Section X analyzes the current status and future direction of this research in terms of security and performance, and concludes the paper with a discussion of its contributions and limitations.

II. Network Model and Security Assumptions

A. Network Model Assumptions

The paper considers a cluster-based sensor network model where sensor nodes are organized into clusters with each cluster having a unique cluster ID. The nodes within a cluster forward their data to the CH which aggregates and sends this data to the Base Station (BS) in the form of data reports. We assume the network to comprise of current generation of sensor motes, e.g., the Berkeley MICA/MICA2 Motes [1-6]. Cluster organization is static, but the role of the CH rotates according to some popular protocols [8], [9], and [15]. All links are bi-directional, though the cost of sending data may be different from CHs to BS than from BS to CHs [16], [17].

Routes from each cluster to the BS can be computed using some well known schemes [18] and route updates and path repair (after node failures) can be performed on-the-fly. Route computation, route maintenance and routing protocols are outside the scope of this paper, and are addressed extensively in the existing literature [40], [30], [31], [32], [33]. The network may maintain a multipath model (disjoint or braided multipaths [19]) with multiple paths between the BS and a CH. But at any given time, a CH will use exactly one primary route to the BS, and it will use this route for sending all traffic (reports and control data) to the BS [10]. The density of sensor nodes is relatively high such that each cluster is surrounded by several neighboring clusters, and nodes that are 1-hop neighbors, but not within the same cluster, can communicate with each other.

B. Security Assumptions

The aggregated data reports from a CH to the BS are end-to-end encrypted, thus they cannot be tampered or forged by any intermediate node on the route from CH to BS without detection. We assume the use of well known schemes for establishing and managing keys in the network [7], [20], [21]. These schemes assume that nodes are pre-loaded with some keying material which enables them to establish and update keys on-the-fly. Specifically, nodes in a cluster share a common group key with the BS which is used for encrypting the cluster's reports. Nodes also share pair-wise keys with their one-hop neighbors both within and outside their cluster, and can establish pair-wise keys with other nodes as required. Thus, we assume that non-malicious nodes have a way of securely communicating with each other (by renewing keys) even when a malicious node is detected in the network. New keys can be established and managed as and when old nodes are removed from the network. The BS is never compromised.

III. Technique Overview

Resiliency against report tampering by malicious CHs and malicious report-dropping attacks by CHs or other nodes in the path to the BS is obtained through the traditional *consistency-check* techniques. In our proposed scheme, the BS receives two versions of data reports from the sensor suite and performs consistency checking by comparing them. In addition to a CH aggregating the cluster nodes' data into a primary report, periodically a backup node within the cluster constructs a proxy report from the data of the nodes in its radio range. It forwards this proxy report to a node outside the cluster known as the proxy node. The proxy node forwards this proxy report to its own CH as specially marked data. This neighboring CH then sends this proxy-report to the BS

along its own primary path. This is shown in Fig. 1a where CH1 is the CH of the cluster under consideration.

The BS ascertains the correctness and authenticity of the primary reports with some probability by verifying them for integrity and reliability against the proxy reports received by it along backup paths from stand-by nodes (backup and proxy, in this case). This is in addition to the consistency checks built in the reports themselves to detect any transmission and channel errors, and any tampering with the reports by nodes en-route to the BS. The techniques employed for such consistency checks, and for the primary-proxy report verification depend on the nature of the data reports and the size of the cluster. Frequency of proxy report generation, governed by the BS, depends on the correct receipt of previous primary reports by it.

Report formats can be numeric values (e.g., temperature readings), or non-numeric data-string format. Clusters can be small (such that all nodes within a cluster are in the radio range of each other) or they can be larger. For small cluster sizes, reports sent to the BS contain XMAC values for verification. An XMAC value [10] is the XOR combination of encrypted message authentication codes (HMAC or encrypted-MAC of messages) sent by individual sensors. For large clusters with numeric report formats, Merkle hash tree [28, 29] based non-interactive function-specific proofs are sent to the BS along with the reports for verification. Finally, for large clusters with string data format, the reports sent to the BS contain the HMAC values with some compression function applied to the reports. The BS uses the XMAC values or the non-interactive proofs enclosed in the primary reports to ensure that the primary reports have been received accurately without any errors or en-route tampering. The BS compares the primary and proxy reports and verifies whether they are within certain error margins.

The paths from the CHs of neighboring clusters to the BS comprise the set of backup paths for a given cluster. Fig. 1b shows the primary path and few of the possible backup paths for cluster with CH1. Selecting a backup path from the set of available paths is non-trivial: To maximize the chances of a proxy report being delivered accurately at the BS, effects of any node failure or compromise on the primary path (or the primary report) should have minimal impact on the backup path (or the proxy report). In [22] the authors show that multipaths with least number of nodes and maximum disjointness between a pair of nodes in mobile ad-hoc networks are least likely to fail due to node mobility. Extending this logic to node failure in our model, backup paths maximally disjoint from the main path and having the fewest nodes are least likely to fail due to a node failure in the main path. The problem of finding backup paths that simultaneously satisfy the conditions of being shortest and maximally disjoint from primary path is NP-hard. This is addressed by us in a related paper [34] which presents a heuristic for selecting short paths that are sufficiently disjoint from the primary path. Presently, for the purpose of understanding the BS-Cluster communication model, it is sufficient to assume that the BS computes a heuristic measure called Degree of Disjointedness (DD) and the nodes in the Cluster utilize this heuristic for backup node and path selection.

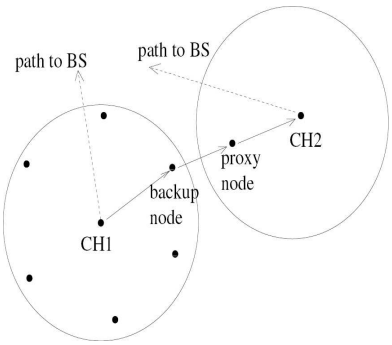

Fig. 1a. Backup and Proxy Nodes for cluster with CH1

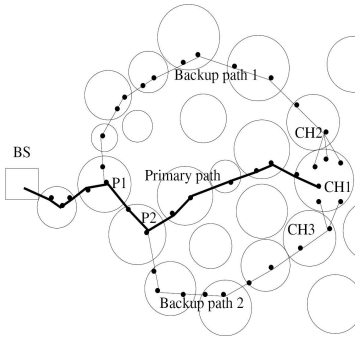

Fig. 1b. Primary and Backup Paths for cluster with CH1

A careful selection of backup and proxy nodes for every cluster is also in order. Backup nodes should be different from the CH, should not have sent more than a certain number of contiguous proxy reports to the BS in the immediate past, and their choice should be non-trivial to an adversary. This is to avoid malicious backup/proxy nodes repeatedly discrediting the primary data reports. The remaining battery power for the candidates should be greater than a certain threshold. Possible backup node selection techniques are:

• **Random Selection**: Nodes randomly elect to be backup nodes. The advantage of this scheme is its simplicity; however, a malicious node can repeatedly elect to be the backup node.

• **Static Selection**: Candidate nodes are arranged in an ordered set and sequentially assigned to be backup nodes. This can be done in a fashion such that the nodes are able to uniquely determine their order without exchanging any information [24]. The advantage over random selection is that a single malicious node cannot continually discredit the primary report at the BS.

• **Evaluation-based**: Here a simple metric N is evaluated for each candidate node, where N is a function of DD, the amount of traffic sent by the node, the remaining power, and the area of the cluster covered by that particular node (i.e., the proportion of nodes whose data it can include in the proxy report, see Sec. VI). This is more complex but a fine-grained selection of nodes is achievable.

IV. BS-Cluster Communication Model

This section presents BS-Cluster communication and associated computation for one cluster. The sequence is outlined in Fig. 2 and described below. Steps (1) and (2) describe the initial computations. Steps (3) to (6) describe the subsequent continuous functioning of the scheme. The following notations are used:

- p: primary path of the cluster under consideration
- P': set of backup paths (BS to neighboring CHs) for p
- p': backup path selected from P′ to send proxy report

(1) The BS computes DD values for the set of backup paths in P'. Initially, it selects a backup path p' from P' (usually the path with the highest DD value). It sends the set of DD values to the cluster along the paths p and p'. Thus all nodes in the cluster should have the set of DD values. These DD values can be updated by the BS incase of any changes to p or the routes in P'. The DD values are piggybacked on the acknowledgement digests (explained below) to the cluster.

(2) The BS acknowledges the last-t reports from a cluster through an acknowledgement digest, where t is an implementation dependent parameter. The value of t can be varied by the BS based on the number of previous reports received accurately, or it can be a function of the cluster size (e.g., log[n]). This is similar to *flow control* in TCP. Initially the BS sends this acknowledgement digest on both p and p'. Subsequently, a threshold may be used for the number of paths for sending the acknowledgement digest. For example, if more than half of the last t reports were correctly received, the BS sends the digest along p, else along both p and p'. Initially the choice of p' is made by the BS. Eventually the choice of p' is governed by the cluster nodes based on DD and local computations at the cluster. Once the cluster receives the DD, the process of proxy report generation is initiated.

(3) The cluster determines the next backup node to generate the proxy report, and the path p' to send the proxy report on.

(4) The backup node collects the data of the nodes in its radio range to construct the proxy report. It passes the proxy report to the proxy node, which in turn sends it to its own CH as specially marked data. The format of the proxy report, cluster size, and determining the accuracy of the main report from the proxy report are discussed in Sec. V and VI.

(5) The cluster also acknowledges the acknowledgement digest received from the BS to enhance the robustness of the scheme. Receipt of this acknowledgement by the BS helps in accurate assessment of the reliability of the paths. This acknowledgement can be sent on *p* along with the next report, and also on *p′* along with the proxy-report.

(6) The BS controls the frequency of generation of proxy reports by communicating it to the cluster along with the acknowledgement digests. This is similar to the contention window in IEEE 802.11 [23]. If the primary reports are being received with adequate accuracy, then the BS can request to reduce the rate of generation of proxy reports.

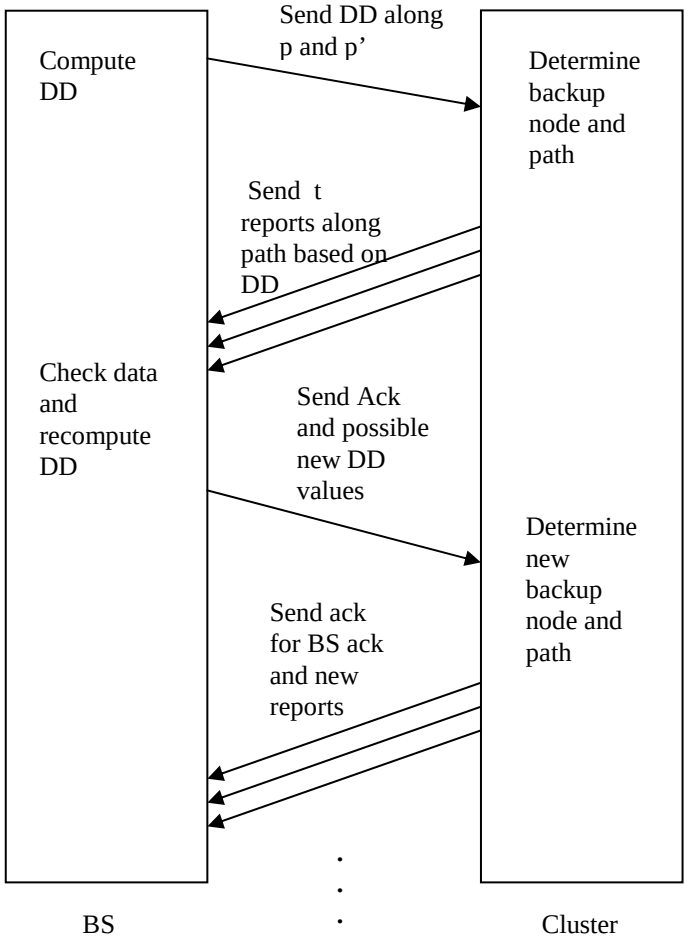

Fig. 2. BS-Cluster Communication for one cluster

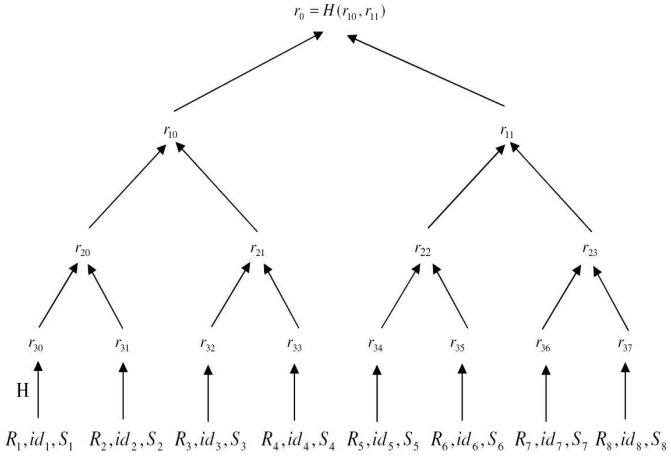

Fig. 3. 8-Node Merkel Hash Tree

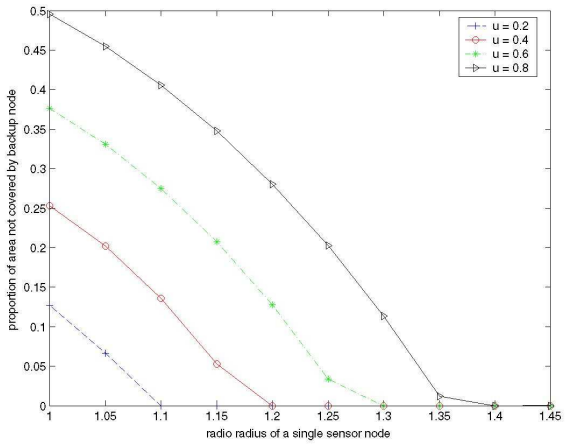

Fig. 4. Percentage of Cluster Area

V. REPORT FORMATS: DESIGN CONSIDERATIONS

This section describes different report formats for different data types and cluster size scenarios. There are four different cases pertaining to small and large sized clusters and number and string format reports, each having different security and reliability implications. Below, we first present the notations used in the remainder of this paper to describe the reports.

A. Report Format: Notations

- C is the id of the cluster.
- CH is the id of the cluster head.
- K_{BC} is the pairwise key between the base station and cluster head CH .

- K_{Bp} is the pairwise key between the base station and backup node P.
- K_{Bi} is the pairwise key between the base station and sensor node i.
- $R_1, R_2, \cdots, R_n$ are the data collected by node $1,2,\cdots,n$, respectively.
- $id_1, id_2, \cdots, id_n$ are the ids of node $1,2,\cdots,n$, respectively.
- Seq is the sequence number of the report.
- $Nonce$ is the random number used only for the report to guarantee message freshness.
- $E_K(M)$ is the symmetric encryption of message M with key K.
- $HMAC_K(M)$ is the encrypted message authentication code of message M with key K.
- $H(M)$ is the hash function of message M.

B. Report Format and Cluster Size: Different Cases

(1) Small Cluster Size: String Format Report

Primary Report:

$$Seq, Nonce, CH, C, E_{K_{BC}}((R_1, id_1) \| \cdots \| (R_n, id_n) \| XMAC \| Seq \| Nonce \| CH \| C)$$

Where $XMAC = HMAC_{K_{B1}}(R_1, id_1)\ XOR \cdots XOR\ HMAC_{K_{Bn}}(R_n, id_n)$

Proxy Report:

$Seq, Nonce, id_p, C, E_{K_{Bp}}((R_1', id_1') \| \cdots \| (R_k', id_k') \| XMAC \| Seq \| Nonce \| id_p \| C)$ Where: $XMAC = HMAC_{K_{B1}'}(R_1', id_1')\ XOR \cdots XOR\ HMAC_{K_{Bk}'}(R_k', id_k')$; $id_1', \cdots, id_k'$ are the ids of the nodes whose data was received by the backup node; $R_1', \cdots R_k'$ are the data values received from those nodes; $K_{B1}', \cdots, K_{Bk}'$ are the corresponding pair-wise keys between those nodes and BS. To reduce the size of the transmitted reports, the concatenation operator $\|$ in $Seq||Nonce||CH||C$ in the encrypted part can be replaced by the XOR operator. It is important to mention that a primary report and its corresponding proxy report would use the same sequence number *Seq*. However, the *Nonce* used would be unique for each report.

(2) Small Cluster Size: Numeric Format Report

This is identical to (1) above.

(3) Large Cluster Size, Numeric Format Report

This scenario uses Merkle hash trees [28], [29]. In a Merkle hash tree, every parent is the hash of the concatenation of its two children, and the root of the tree is the commitment of all leaf nodes. To authenticate a specific leaf node, (e.g. $R_4id_4S_4$) the prover needs to provide all the values from the leaf to the root ($r_{32}, r_{20}, r_{11}, r_0$) [25]. Fig. 3 shows the Merkle hash tree of 8 nodes. Merkle hash trees also have the following properties:

1. Given the root of the Merkle hash tree, it is not possible to recover any of the non-root nodes, including the leaf-nodes.
2. Changing the root of the Merkle hash tree will lead to failure in future commitment verification.
3. A single message can be the commitment of multiple messages.

Properties 1 and 2 ensure message security while 3 ensures reduced communication overhead for sensor networks.

Primary Report:

$Seq, Nonce, CH, C, E_{K_{BC}}(f(R_1, \cdots, R_n) \| r_0 \| proof \| Seq \| Nonce \| CH \| C)$ Where:

f is the function to be computed over collected data in the cluster (e.g., *min* or *max*); r_0 is the root of the Merkle hash tree of the collected data; $proof$ is the proof for the BS to verify the correctness of the report, which will be discussed in next section.

The leaves of the Merkle hash tree are in the format (R_i, id_i, S_i), where $S_i = HMAC_{K_{Bi}}(R_i, id_i)$.

Proxy Report:

$Seq, Nonce, id_p, C, E_{K_{Bp}}(f(R_1', \cdots, R_k') \| r_0 \| proof \| Seq \| Nonce \| id_p \| C)$

Again, a primary report and its corresponding proxy report would use the same sequence number *Seq*, but the *Nonce* used would be unique for each report. Also, the root of the Merkle Hash tree would be different in the primary and proxy reports.

The construction of *proof* will be dependent on the specific function that is being used. Two of the most commonly used functions are median and min (max). The details of constructing non-interactive *proofs* for median and min/max function are presented in the next section.

(4) Large Cluster Size, String Format Report

Primary Report:

$Seq, Nonce, CH, C, E_{K_{BC}}(Comp(R_1 \| \cdots \| R_n) \| XMAC \| Seq \| Nonce \| CH \| C)$ Where, $Comp$ is some lightweight compression function;

$XMAC = HMAC_{K_{B1}}(R_1, id_1)\ XOR \cdots XOR\ \ HMAC_{K_{Bn}}(R_n, id_n)$

Proxy Report:

$Seq, Nonce, id_p, C, E_{K_{Bp}}(Comp(R_1' \| \cdots \| R_k') \| XMAC \| Seq \| Nonce \| id_p \| C)$ Where $XMAC = HMAC_{K_{B1'}}(R_1', id_1')\ XOR \cdots XOR\ \ HMAC_{K_{Bk'}}(R_k', id_k')$

and other notations have the same meaning as in case (1).

Again, the concatenation operator ‖ in $Seq||Nonce||CH||C$ in the encrypted part can be replaced by the XOR operator for size reduction.

VI. Report Verification And Consistency Check At BS

The BS verifies the validity of each report that it receives. It also checks for consistency between the primary and proxy reports. We first describe how the BS manages each report- format/cluster-size scenario described above. Small cluster size implies that each node in a cluster is within every other node's radio range, making the consistency-check easier. Additional checks are required for large clusters with string and numeric format data reports, taking into consideration the area coverage of the backup nodes. This is discussed next. Finally, for large-size clusters with numeric format data reports, Merkle hash tree based non-interactive proofs for representative functions like median and min/max verification are also elaborated.

A. Report Formats and Cluster Sizes: Computation at BS

(1) Small Cluster Size: String Format Report

To verify the validity of a report (either primary or proxy), the BS needs to investigate the following:

- The sequence number, nonce and ids of cluster and CH (or backup node) in the plaintext part of the report are the same as that in the encrypted part.
- The nonce is fresh; it has not been used previously.

- The XMAC is correct. Since the BS has all the pairwise keys, it can independently compute the XMAC after receiving the report.

To check for consistency between the primary report and proxy report, the BS needs to ascertain the following:

- The sequence number and ids of the cluster in the two reports should be the same.
- The data values reported by the proxy report should be a subset of the values reported in the primary report.

(2) Small Cluster Size: Numeric Format Report

This is identical to (1) above.

(3) Large Cluster Size, Numeric Format Report

Report validation by the BS is similar to the first case. In addition, the legitimacy of the *proof* part inside a report needs to be ascertained, i.e., the BS needs to verify that the *proof* inside the report is an accurate representation of the reported data values. Details of *proof*s for some representative functions are presented in section VI.C below.

Besides, the BS cannot directly check if the reported data values in the proxy report are a subset of the values in the primary report. However, the BS can ascertain this (with some probability) by performing some additional function-specific consistency-checks for the two reports. This is discussed in section VI.B below.

(4) Large Cluster Size, String Format Report

The BS performs the same checks as in the first case, but it needs one more step to decompress the compressed part in both reports.

B. Additional Checks for Large Size Clusters: Area Coverage

Suppose the radio range of a single sensor node is r, the radius of the cluster is R, the distance between the backup node and the center of the cluster is u. Then the backup node can cover area (For simplicity, we assume that the cluster is a circle)

$$A = r^2 \arccos\left(\frac{r^2+u^2-R^2}{2ru}\right) + R^2 \arccos\left(\frac{R^2+u^2-r^2}{2Ru}\right) - 2\sqrt{s(s-u)(s-r)(s-R)}$$

Where $s = \frac{R+r+u}{2}$

So the area that is not covered by the backup node is $\pi R^2 - A$. If we take $R = 1$ and r is fixed, then the expected area that is not covered by the backup node (for a random backup node with random distribution in the cluster) will be $\int_0^1 (1 - A/\pi)du$

If we take $R = 1$ and $r = 1$, then the expected area not covered by the backup node is 0.9786, which accounts for about 31.15% of the whole area of cluster. In the worst case, if the backup node is located near the edge of the cluster, the backup node will cover about 39.1% of the area, which means the backup node will receive data from about 39.1% of the nodes in the cluster.

Figure 4 represents the values for $[1 - A/\pi]$ (which is the percentage of the cluster area not covered by the backup node) for different p and r, if we assume $R = 1$. From the figure it is clear that the proportion of area not covered by the backup node is almost inversely proportional to the radio range of a single node, and as long as the distance between the backup node and the CH is less the 0.8 times of the cluster radius, the proportion of area not covered by backup node will be less than one half.

C. Proofs for Representative Functions: Numeric Format Reports and Large Cluster Size

In our scheme, we attach a *proof* part with every report. The *proof* part in the report in our scheme depends on the specific function f. We show the construction of the *proof* part for two representative functions: median and min/max.

1. The *proof* for Median Function.

The *proof* part for the primary report can be adopted from the scheme in [25], which describes uniform sampling of $O(\log n/\varepsilon)$ elements (where n is the number of sensors in one cluster and ε is the approximation error which represents the tolerable error margin in reporting data). However, the approach adopted in [25] is to perform interactive proofs. Interactive protocols would be prohibitively expensive for sensor networks in terms of communication overhead, even if sub-linear rounds of communication are required (as described in [25]).

We suggest that uniform sampling can be done through some public random functions (one method to obtain the seed of the public random function can be from the clock in the sensor nodes), and the interactive proofs in the original scheme in [25] can be converted to non-interactive proofs, thus substantially saving on communication overhead and making the technique feasible for sensor networks.

The details of non-interactive proofs merit an entire paper and are outside the scope of the current discussion; however, we briefly mention a few possible techniques below:

Non-interactive Proofs: Our construction of the *proof* part requires some random bits to be used as seed for some pseudorandom functions. To guarantee safety, we must ensure that the CH and the backup node cannot control or predict the random bits. To achieve this, one of the following methods can be chosen.

- The first method is to install some tamper-proof hardware to generate random bits in the sensor nodes. This approach has no communication overhead but is expensive if the number of sensor nodes is very large.
- Another method is to use the current time as the random seed. This requires the sensor nodes to have precise time synchronization (time synchronization in sensor networks is a well researched problem). If the nature of the network requires time synchronization then this technique entails no additional overhead in terms of communication or hardware costs. Otherwise this technique would not be useful, especially when data aggregation needs to be performed frequently, resulting in frequent time synchronization requirements.
- The third method is to use the delayed verification technique. In this scheme, the attached *proof* part in a data packet is the evidence of validity of the data sent in the previous data packet (or several previous data packets). The random bits are included in the *Ack* packet sent by the BS before this round of data aggregation. In this approach, the CH or backup node needs to cache the data of the previous round or several previous rounds (depending on the frequency of *Ack* packets). Techniques to improve robustness of the scheme in case of packet loss are in order. We will investigate this further in our future research.

The *proof* part for the proxy report is the same as that in the primary report; the only difference is that the backup node may only collect data from $n_1 \le n$ nodes. Thus the sampling is done only over these l elements. According to the following theorem [25], the median of l elements should be close to the median of n data points with high probability if l is not too smaller than n.

Theorem 6.1 The median of uniform sample of l out n elements $a_1, a_2, \cdots, a_n$ with probability at least $1-(2/e^{2l\varepsilon^2})$ yields an element whose position in the sorted sequence $a_1, a_2, \cdots, a_n$ is within εn of $n/2$.

Remark 1: If the backup node can receive data from all the nodes in the cluster, then the median reported by the backup node should be identical to the median reported by the primary node. In case of large clusters, if the backup node is not within the radio range of every other cluster node, some inter-cluster communication scheme to send all data to the backup node is required. This may be too resource consuming for the sensor nodes. Thus, the scenario that the backup node only receives a subset of data elements is considered. This would not incur much overhead since receiving is less resource consuming than broadcasting.

Remark 2: In the event that the primary report is lost and only the proxy report is received, the BS can still obtain and check the median from the proxy report (though the value may be inaccurate if the backup node does not receive enough data).

Remark 3: If the BS receives both the primary and proxy reports, it performs the previously mentioned validity and consistency checks. It enumerates the difference between the two medians reported by the CH and the backup node. If the difference between the two medians is located more than $(1-k)n/2$ positions apart in the sorted sequence of elements in the primary report, then the BS detects that one of the two nodes (CH or backup) is cheating. Here k is the percentage of the cluster area covered by the backup node. The BS can estimate some minimum value of k and send this value to the cluster before this round of data aggregation. The nodes with values that are located $(1-k)n/2$ positions apart from the reported median can be included in the *proof* for the BS to check.

2. The *proof* for Min/Max Function

For the min/max function, we propose two approaches for constructing the *proof.*

Approach 1: We adopt the data propagation scheme for min/max function in [25]. First a spanning tree is constructed such that the root of the tree holds the minimum (maximum) element, and then it is checked if the tree was constructed properly. If we make the same assumption as in [25] that uncorrupted sensors form a connected component, then both the CH and backup node can receive the min/max value. The backup node may not receive the authenticated states from all nodes in the cluster while the CH can receive all of them. The CH constructs the *proof* part as described in [25], containing $O(1/\varepsilon)$ random samples of nodes and their corresponding paths to the root of the tree.

Since the backup node receives the authenticated states only from some of the nodes in the cluster, it cannot construct the *proof* part in the same manner as the CH does. As in [25] assume that each sensor node *i* maintains a tuple of state variables (p_i, v_i, id_i) where p_i, v_i and id_i are the parent of the node in the tree being constructed, received min/max value, and identity of the node *i*, respectively. Suppose the backup node receives the min/max value from k neighboring nodes $(n_1, n_2, \cdots, n_k)$ and the final state of node *i* is (p_i, v_i, id_i), for $i = 1,2,\cdots,k$. Then the backup node can construct the *proof* through t random samples of nodes from $n_1, n_2, \cdots, n_k$, where every sample of node i is the concatenation of (p_i, v_i, id_i) and the MAC of (p_i, v_i, id_i), computed using the shared key between node *i* and the BS.

Theorem 6.2 If no more than ε fraction of the sensors are corrupted, and the backup node reports a false min (max) value, then the BS will detect the false value with probability at least $1-\varepsilon^t$.

Proof: For every sample node *i*, no node except *i* can forge the authenticated code of (p_i, v_i, id_i). The backup node cannot cheat on this unless the node *i* is corrupted. Node *i* is corrupted with probability ε, since no more than ε fraction of the sensors are corrupted. For t random samples, the BS will detect the false value with probability at least $1-\varepsilon^t$.

Approach 2: The scheme to find min/max in [25] requires *d* steps of broadcasting in the cluster, where d is the diameter (in terms of radio range) of the cluster. This scheme is still too expensive for practical sensor networks. If the data commitment (root of the Merkle hash tree in the aggregate-commit-prove approach) is constructed from the sorted leaf elements, and random sampling is used to construct the *proof* part directly, then a simpler, admissibly secure *proof* is obtained.

In this case, the *proof* (in both the primary and proxy report) contains the random sample of $1/\varepsilon$ leaf elements and their corresponding hash in the Merkle hash tree. It is evident that if the CH (or backup node) chooses an element whose position is more than εn elements away from the true minimum (or maximum) in the sorted data list, the following condition evaluates to true. Considering both reports together, the probability that one of the sampled elements (in either primary or proxy report) is smaller than the reported minimum (or greater the reported maximum) is $1-(1-\varepsilon)^{2\varepsilon} > 1-e^{-2}$

This probability can be further improved if we consider that the "sorted list" property will be violated if the CH or backup node reports a non-minimum (or non-maximum) element.

VII. Security Analysis

This section analyzes several key security features of our scheme and evaluates their impact. We show that the reports (both primary and proxy) cannot be forged by any outsiders and the contents of the reports are confidential. By proving the NP-hardness of the minimum cost blocking problem, we demonstrate that it is difficult for any adversary to determine within an optimal cost, the subset of nodes to compromise such that the traffic from certain specific clusters can be completely blocked from reaching the BS.

A. Outsider Attacks

It is evident that if the pairwise key between the CH and BS is not compromised, then the contents of the data values in the reports cannot be decrypted by any outsiders. Also, no outsider can launch a replay attack since we use nonces and sequence numbers. False data injection attacks can also be prevented since all public parts in the reports (those items that are not encrypted in the report, such as sequence number, nonce, cluster id and node id) can be verified in the encrypted parts. Thus our scheme ensures safety against attacks by an outside adversary.

B. Hardness of Minimum Cost Blocking Problem

Suppose there are m clusters: $C_1, C_2, \cdots, C_m$. Every cluster C_i uses a primary path P_i and backup paths $P_{i1}, P_{i2}, \cdots, P_{ik}$ in k different rounds. One of the adversary's goals is to block the primary path and all backup paths in k rounds for some clusters, so no report from these clusters can reach the BS in these k rounds. Suppose the total number of sensor nodes is $N = nm$, where n is the number of nodes in one cluster. Every node has an associated cost c_i ($i = 1,2,\cdots,N$) to be compromised. The adversary's goal translates to finding the subset of nodes with minimum total cost such that blocking (or compromising) this subset of nodes can block all traffic between a subset of clusters and BS in all k rounds. We demonstrate below that finding such a subset of nodes is NP-hard.

Theorem 7.1 It is NP-hard to find a subset of nodes with minimum total cost such that blocking (or compromising) this subset of nodes blocks traffic between a subset of clusters and BS in all k rounds.

Proof It is evident that the problem is equivalent to finding a subset of nodes with minimum total cost to block all primary paths and backup paths in k rounds for the specified clusters. If we regard each path under consideration as an element, every node as a set contains some elements. If a node is in a path, then the path is contained in the set corresponding to the node. The problem becomes the weighted set cover problem [35], which is the generalization of set cover problem and is NP-hard.

VIII. Experimental Results

Though a qualitative approach was adopted for the security analysis, the performance of our scheme under node failures is evaluated through simulation experiments. We tested our schemes in a randomly generated network topology. In the simulation, we consider a total of 100 clusters where every cluster has 10 nodes. The simulation topology is a 1000 meter by 1000 meter square region which is divided into 100 small squares (every square has size 100 meter by 100 meter). Every cluster is located inside a small square, and all nodes belonging to a cluster are randomly distributed inside the small square. CH of every cluster is randomly chosen. We assume that all nodes are homogenous and the radio range of every node is 100 meters. When two nodes are in each other's radio range, we say that they are connected. The BS is located in a corner of the square region, and all routing is based on the criteria that the routes with the smallest number of hops will be selected. If any tie exists, then one of the routes will be randomly chosen.

We tested the performance of our scheme in the presence of two kinds of node failures: single node failures and patterned node failures. In case of single node failures, every node is assigned with a probability of it being compromised. In case of patterned failures, all nodes within a selected square region will be compromised. The center of this compromised square region is randomly assigned in the entire topology. We use MATLAB in our tests and all data points are the average of 100 runs.

Fig. 5 is the probability of the reports reaching the BS with some nodes compromised or failed (these nodes are evenly distributed over the network). From Fig 5 we see that when the number of compromised nodes increases, the probability that both reports reach BS decreases almost linearly. The probability that at least one of the reports reaches the BS also decreases almost linearly.

Fig. 6 shows the percentage of clusters from which neither the primary nor the proxy report reaches the BS when there is a 40m by 40m square patterned failure centered at a certain point. Fig. 7 represents the percentage of clusters from which only one of the primary or proxy reports reaches the BS when there is a 40m by 40m patterned failure centered at a certain point. Fig. 8 is the percentage of clusters from which neither the primary or proxy report reaches the BS when there is an 80m by 80m patterned failure centered at a certain point. Fig. 9 represents the percentage of clusters from which only one of the primary or proxy reports reaches the BS when there is an 80m by 80m patterned failure centered at a certain point.

From Figs. 6, 7, and 8 we can see that there is a sharp increase in number of affected clusters when the affected area is near the BS and when the affected area is near the diagonal of the entire region. The trend of increase in the number of affected clusters is similar for clusters losing one report and clusters losing both reports when the affected area is close to the BS and the diagonal. As expected, larger failure area has increased effect.

In reality, it may be more difficult to compromise the nodes closer to the BS. However, our experiments underline the fact that nodes near the BS need more protection as it would be easier for an adversary to have a large impact on the network by compromising fewer nodes proximal to the BS. This is due to the large number f routes passing through these nodes being compromised.

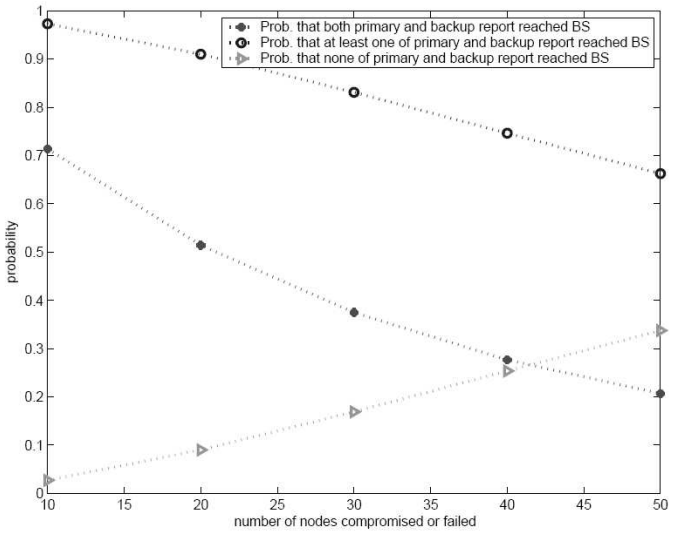

Fig. 5. Probability of Reports Reaching the BS

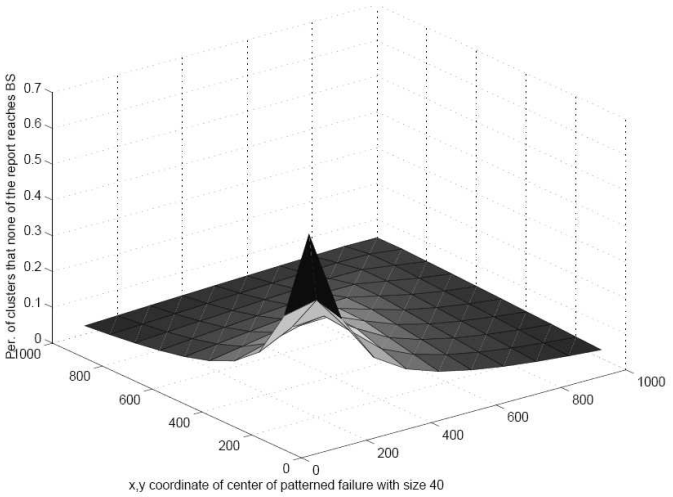

Fig. 6. 40m X 40m Patterned Node Failure: No Reports Reach BS

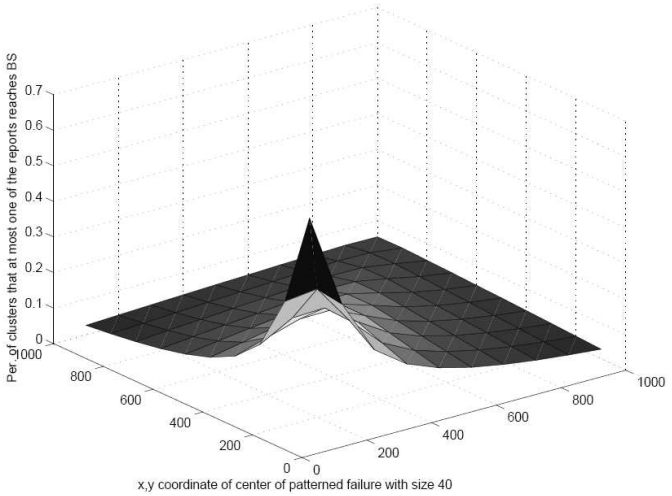

Fig. 7. 40m X 40m Patterned Node Failure: One Report Reaches BS

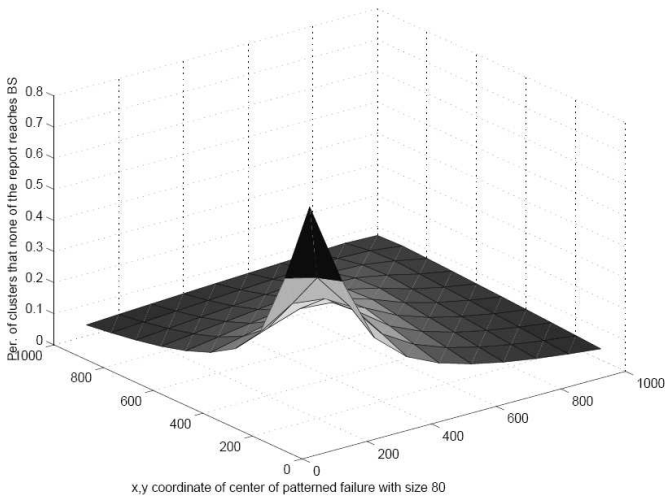

Fig. 8. 80m X 80m Patterned Node Failure: No Reports Reach BS

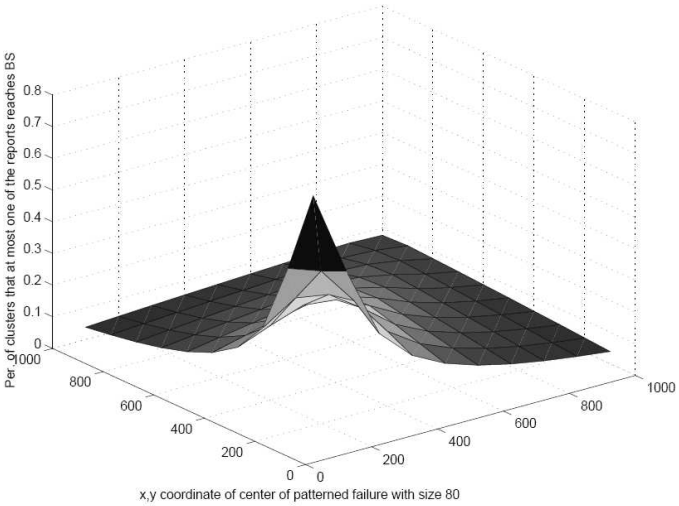

Fig. 9. 80m X 80m Patterned Node Failure: One Report Reaches BS

IX. Related Work

This section discusses related work in Sensor networks security and reliable data delivery, and compares it with our approach. Due to the different approaches adopted in security research and reliable data delivery, we mention these under different sub-sections.

A. Sensor Network Security

In [7], Zhu et al. describe a mechanism for establishing, updating and managing four types of keys for sensor nodes: individual (node and BS), pairwise (node and neighbor), cluster (all nodes in a cluster), and group keys (for the entire network). In [21], Ning et al. also describe similar key management schemes. In [10], Zhu et al. present an interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks. In [36], Perrig et al. present schemes for secure authentication in multicast communication, i.e., enabling receivers of multicast data to verify that the received data originated with the claimed source and was not modified en-route. In [1], the authors build on their work to address this issue specifically for networks with two component architecture: SNEP and μTESLA. SNEP includes: data confidentiality, two-party data authentication, and evidence of data freshness. μTESLA provides authenticated broadcast for severely resource-constrained environments. But our scheme addresses the issue of the CH modifying the data. In [10], the authors present a Random Key Pre-distribution scheme for sensor networks.

In [25] the authors present a Secure Information aggregation scheme for sensor networks based on random sampling mechanisms and interactive proofs. This work assumes the use of special class of nodes called aggregators which respond to data queries and uses statistical techniques like computing the mean and averages of the readings recorded and gives a good approximation on the sensor readings to the users. In [13], Perrig et al. present a scheme to deal with Sybil attacks on sensor networks in which a node illegitimately assumes the identities of multiple nodes in the network. In [39], the authors address the key distribution problem in environments with a partially present, passive adversary: a node wishing to communicate securely with other nodes simply generates a symmetric key and sends it in the clear to its neighbors. In [12], the authors present symmetric key distribution protocol for large scale sensor networks with low overhead that uses one or more intermediate sensor nodes as a trusted intermediary to facilitate key establishment. [37] (Distributed Detection of Node Replication Attacks in Sensor Networks) and [38] (Detection of Denial-of-Message Attacks on Sensor Network Broadcasts) are two other papers by same authors on related topics, but the manuscripts for these papers are not available at the time of writing this paper.

B. Reliable Data Delivery

In [41], Akyldiz et al. consider the problem of reliable downstream point-to-multipoint data delivery, from the sink to the sensors. In [42], Wan et al. claim to propose the first reliable transport protocol for wireless sensor networks by “pumping data at slow pace” in networks with light traffic. In [43] and [44], Deb et al. provide schemes for differential Quality of Service in Sensor Networks. In [45], the authors introduce SPROID (Scalable Protocol for RObust Information Dissemination), which tags data generated with a unique identifier and provides reliable data delivery to all the sensor nodes in the network.

As can be seen from this review, the conceivable problems of report tampering by a CH in a network with clusters of any size, and the detection of report dropping by en-route nodes, though important, have not been addressed by any prior work.

X. Conclusion

This scheme detects report tampering by malicious CHs and report dropping by CH or other malicious nodes on the route to the BS. It considers different cluster topologies and different report formats. The design of this scheme seeks to minimize the control

overhead for the energy constrained sensor nodes. Unlike TCP, our scheme can be tuned for obtaining a desired tradeoff between control overhead and the accuracy of detection. Thus, the degree of reliability in receiving data reports can be governed by the particular application. Using our scheme in conjunction with schemes to prevent false data injection can result in an end-to-end, semi-reliable data delivery model for sensor networks. Salient features of this scheme are: (a) Even if a backup node is malicious and deliberately sends an incorrect proxy report, the role of the backup node rotates such that no node serves as a backup for more than one consecutive proxy report. This minimizes the possibility of collusion attacks by colluding malicious nodes. (b) Sending the acknowledgement digest to the backup nodes makes it impossible for a malicious CH to go unnoticed.

Our continuing work focuses on determining the control overhead entailed by the scheme through qualitative analysis and verifying the results through simulation. We are also looking at techniques to minimize the size and contents of the proxy reports without affecting the accuracy of attack detection.

REFERENCES

[1] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. Tygar, "SPINS: Security Protocols for Sensor Networks", Seventh Annual ACM International Conference on Mobile Computing and Networks (Mobicom 2001), Rome Italy, July 2001.

[2] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister, "System Architecture Directions for Networked sensors", ASPLOS IX, 2000.

[3] CROSSBOWTECHNOLOGY INC., “Wireless sensor networks”, http://www.xbow.com/Products/ Wireless Sensor Networks.htm.

[4] UC Berkeley The EECS department, “Cotsbots: The mobile mote-based robots”, http://www-bsac.eecs.berkeley.edu/projects/cotsbots/

[5] W. Zhang, H. Song, S. Zhu, G. Cao, “Least Privilege and Privilege Deprivation: Towards Tolerating Mobile Sink Compromises in Wireless Sensor Networks”, ACM Mobihoc'05, UIUC, May 2005.

[6] http://www.ieee802.org/15/pub/TG4.html

[7] S. Zhu, S. Setia, and S. Jajodia, “LEAP: efficient security mechanisms for large-scale distributed sensor networks”, ACM CCS, Washington DC, 2003, pp. 62-72.

[8] W. Heinzelman, “Application-specific protocol architectures for wireless networks”, PhD Thesis, Massachusetts Institute of Technology, June 2000.

[9] S. Lindsey, C. S. Raghavendra, "PEGASIS: Power Efficient GAthering in Sensor Information Systems", 2002 IEEE Aerospace Conference, March 2002, pp. 1-6.

[10] S. Zhu, S. Setia, S. Jajodia, P. Ning, “An Interleaved Hop-by-Hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks”, IEEE Symposium on Security and Privacy 2004, pp. 259-271.

[11] H. Chan, A. Perrig, D. Song, “Random Key Predistribution Schemes for Sensor Networks”, IEEE Symposium on Security and Privacy 2003.

[12] H. Chan, A. Perrig, “PIKE: Peer Intermediaries for Key Establishment in Sensor Networks”, Infocom 2005.

[13] J. Newsome, E. Shi, D. Song, A. Perrig, “The Sybil Attack in Sensor Networks: Analysis and Defenses”, Third International Symposium on Information Processing in Sensor Networks (IPSN 2004).

[14] http://www.faqs.org/rfcs/rfc793.html

[15] J. Kulik, W. Rabiner, H. Balakrishnan, “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks”, 5th ACM/IEEE Mobicom Conference, Seattle, WA, August 1999.

[16] F. Stann, J. Heidemann, “RMST: Reliable Data Transport in Sensor Networks”, 1st IEEE International Workshop on Sensor Net Protocols and Applications (SNPA). Anchorage, AK, May 2003.

[17] C. Karlof, D. Wagner, “Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures”, Ad Hoc Networks, vol. 1, issues 2--3 (Special Issue on Sensor Network Applications and Protocols), pp. 293-315, Elsevier, September 2003.

[18] F. Stann, J. Heidemann, “BARD: Bayesian-Assisted Resource Discovery In Sensor Networks”, IEEE Infocom, Miami, FL, March 2005.

[19] D. Ganesan, R. Govindan, S. Shenker, D. Estrin, "Highly-Resilient, Energy-Efficient Multipath Routing in Wireless Sensor Networks", Mobile Computing and Communications Review, Vol. 1, Number 2, 2002.

[20] D. Liu, P. Ning, K. Sun, “Efficient self-healing group key distribution with revocation capability”, ACM Conference on Computer and Communications Security 2003, pp. 231-240.

[21] D. Liu, P. Ning, “Establishing pairwise keys in distributed sensor networks”, ACM Conference on Computer and Communications Security 2003, pp.

52-61.

[22] X. Li, L. Cuthbert, “Node-Disjointness-Based Multipath Routing for Mobile Ad hoc Networks”, International Conference on Mobile Computing and Networking, 1st ACM International Workshop on Performance Evaluation of Wireless Ad hoc, Sensor, and Ubiquitous Networks, Venezia, Italy , Oct 2004, pp. 23 - 29.

[23] http://grouper.ieee.org/groups/802/11/

[24] M. Virendra, S. Upadhyaya, V. Kumar, V. Anand, “SAWAN: A Survivable Architecture for Wilreless LANs”, 3rd IEEE International Workshop on Information Assurance (IWIA’05), Mar 2005.

[25] A. Perrig, B. Przydatek, D. Song, “SIA: Secure Information Aggregation in Sensor Networks”, ACM SenSys 2003.

[26] F. Ergun, S. Kannan, S.R. Kumar, R. Rubinfeld and M. Viswanathan. Spot-checkers. JCSS, 60:717-751. Preliminary version in Proc. STOC’98.

[27] F. Ergun, R. Kumar and R. Rubinfeld. "Fast approximate PCPs". In Proc. 31st STOC, pages 41-50, 1999.

[28] Ralph C. Merkle. "A certified digital signature". In Proc. Crypto’89, pages 218-238, 1989.

[29] Ralph C. Merkle. "Protocols for public key cryptosystems". In Proceedings of the IEEE Symposium on Research in Security and Privacy, pages 122-134, April 1980.

[30] B. Krishnamachari, D. Estrin and S. Wicker,"Modelling Data-Centric Routing in Wireless Sensor Networks",IEEE INFOCOM 2002.

[31] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks," in the Proceedings of the First Workshop on Sensor Networks and Applications (WSNA), Atlanta, GA, October 2002.

[32] C. Schurgers and M.B. Srivastava, "Energy efficient routing in wireless sensor networks", in the MILCOM Proceedings on Communications for Network-Centric Operations: Creating the Information Force, McLean, VA, 2001.

[33] Y. Xu, J. Heidemann, D. Estrin,"Geography-informed Energy Conservation for Ad-hoc Routing," In Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking 2001, pp. 70-84.

[34] M.Virendra, Q. Duan, S. Upadhyaya, "On the hardness of an optimal routing problem in wireless sensor networks", manuscript, April 2006.

[35] R. Gandhi, S. Khuller, A. Srinivasan, “Approximation Algorithms for Partial Covering Problems”, Lecture Notes in Computer Science, Springer-Verlag GmbH, ISSN: 0302-9743, Vol. 2076, pp. 225-236.

[36] Adrian Perrig, Ran Canetti, J. D. Tygar, Dawn Song, "The TESLA Broadcast Authentication Protocol", RSA Cryptobytes, Summer 2002.

[37] Bryan Parno, Adrian Perrig, Virgil Gligor, "Distributed Detection of Node Replication Attacks in Sensor Networks", IEEE Symposium on Security and Privacy 2005.

[38] Jonathan M. McCune, Elaine Shi Adrian, Perrig Michael K. Reiter,
"Detection of Denial-of-Message Attacks on Sensor Network Broadcasts ",IEEE Symposium on Security and Privacy 2005.

[39] Ross Anderson, Haowen Chan, Adrian Perrig,"Key Infection: Smart Trust for Smart Dust", 12th IEEE International Conference on Network Protocols (ICNP'04), pp. 206-215.

[40] Issa Khalil, Saurabh Bagchi, Cristina Nina-Rotaru, “DICAS: Detection, Diagnosis and Isolation of Control Attacks in Sensor Networks,” In the IEEE Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm). Athens, Greece from 5 - 9 September, 2005.

[41] Sankarasubramaniam Y. , Akan O. B., and Akyildiz I. F., ``ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," Proc. of the ACM MobiHoc Conference, Annapolis, Maryland, June 2003.

[42] Chieh-Yih Wan, Andrew T. Campbell, Lakshman Krishnamurthy. "PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks," WSNA 2002, Atlanta, GA.

[43] Budhaditya Deb, Sudeept Bhatnagar and Badri Nath, "Information assurance in sensor networks," in WSNA 2003, San Diego, CA.

[44] Budha Deb, Sudeept Bhatnagar, and Badri Nath, "REinform: reliable information forwarding using multiple paths in sensor networks", LCN 2003.

[45] Hari Rangarajan, J.J. Garcia-Luna-Aceves, "Reliable Data Delivery in Event-Driven Wireless Sensor Networks," The Ninth IEEE Symposium on Computers and Communications (ISCC2004).