
Muhammad Usman, Vallipuram Muthukkumarasamy, and Xin-Wen Wu
School of Information and Communication Technology, Griffith University,
Gold Coast Campus, Queensland 4222, Australia
Email: muhammad.usman3@griffithuni.edu.au, v.muthu@griffith.edu.au, x.wu@griffith.edu.au

Abstract—Sensor readings are vulnerable to in situ and in transit anomalies. A well designed anomaly detection system should be able to identify the source of anomalies through in situ verification of suspicious behavior of sensor nodes. One approach for in situ verification is physical diagnosis of sensor nodes, which is a cumbersome and time-consuming task in medium-to-large-scale networks. Therefore, we propose a novel method for in situ verification of malicious sensor nodes using mobile agents. We employ Coordinated Resource Management mechanism-based observations to detect first-order anomalies and perform in situ verification. Since mobile agents cannot be frequently transmitted over the resource constrained Wireless Sensor Networks (WSN) due to the expensive nature of the communication operation as compared to the processing operation, we propose a method which exploits the historical information of anomalous behavior of the sensor node to optimize mobile agent transmission. The performance of the proposed system is investigated via simulation, experiments on MICAz mote, and comparative study. The results and analysis demonstrate the effectiveness and efficiency of the proposed system, in comparison with other existing schemes.

Index Terms—Agent transmission optimization, Anomaly detection system, First-order anomalies, In situ verification, Anomaly agent, Wireless sensor networks

I. INTRODUCTION

The emergence of innovative networking frameworks such as Internet of Things and Shared Sensor Networks has increased the deployment of the Wireless Sensor Networks (WSN) for a number of application domains such as wireless smart home sensor networks, built infrastructure monitoring, smart cities, and health monitoring [1]. The built infrastructure monitoring is one of the promising WSN applications, which monitors water, electricity, and gas consumption, and carbon dioxide (CO₂) emission in the built environments [2]. The performance of such data-centric applications is highly dependent on the accuracy of the received sensed data. However, sensor nodes and their sensor readings are vulnerable to in situ and in transit anomalies due to several factors such as faults, errors, and attacks. An anomaly detection system can be deployed to detect such anomalies in order to mitigate them. Effective mitigation can only be performed after correct identification of the source of anomalies. Existing anomaly detection schemes, however, focus only on detection of anomalies rather than the identification of the source of anomalies [3]-[7]. This limits the effectiveness of the existing schemes.

A robust anomaly detection system should be able to identify the source of anomalies, after their timely detection. This may be accomplished through in situ verification of malicious sensor nodes. Typically, in situ verification of a sensor node can be performed through physical diagnosis. However, it may not always be possible to gain frequent physical access to sensor nodes, particularly when they are deployed on difficult terrains or within the vicinity of private built infrastructure (such as houses). Over the years, the research community has proposed several software mobile agent-based anomaly detection schemes, which have employed mobile agents for different roles such as transmission of control messages among different entities and random sampling of sensed data over the sensor network [3]-[7]. However, these schemes have not considered mobile agents for in situ verification of sensor node in pursuit of identification of the source of anomalies.

In this study, we propose a method that enables mobile agents to use Coordinated Resource Management (CRM) mechanism-based information to perform in situ diagnosis of the sensor nodes. The CRM mechanism enables sensor nodes to share their resource status with corresponding cluster heads or the base station for network resource management. The TinyOS facilitates CRM mechanism through low-level interfaces to share and manage the hardware state of the node over the network [8], [9]. In addition to the traditional network resource management, our hypothesis is to employ CRM-based information to detect several types of first-order anomalies and to perform an in situ verification process. To the best of our knowledge, this is the first study to employ CRM mechanism for both these purposes.

In the proposed system, we exploit the statistical association among different features of the CRM-based observations to define first-order joins over their underlying dis-

Manuscript received June 8, 2014; revised October 24, 2014.
Correspondence e-mails: manilasani@yahoo.com, muhammad.usman3@griffithuni.edu.au

A preliminary version containing initial work of this study has appeared in the proceedings of the 9th International Conference on Ubiquitous Intelligence and Computing, and 9th International Conference on Autonomic and Trusted Computing, pp. 322-329, September 2012.
contributions. This enables our proposed Anomaly Detection System (ADS) to detect a range of first-order anomalies, which are caused by denial of sleep attack, battery exhaustion attack, and sensor node faults. The ADS further uses information received from CRM-based observations to verify the source of anomalies using mobile agents. However, mobile agents cannot be frequently transmitted due to the expensive nature of communication overhead in terms of energy consumption [10]. To address this research challenge, we propose a method that optimizes mobile agent transmission by taking into account the weighted sum of anomalous instances of historical and current observations. This approach assists the system administrator in controlling the transmissions of mobile agents, which prolongs the network lifetime as compared to the existing approaches [3]-[7].

In summary, the main contributions of this paper are as stated below:

- The proposed mobile agent-based anomaly detection and verification system with internal structure and algorithms, which is not only capable of detecting a range of first-order anomalies, but also performs in situ verification of the cluster member sensor nodes using information gathered through the CRM mechanism.

- A mobile agent transmission optimization method is proposed that incorporates the past and current behaviors of the sensor node to optimize the agent transmission process in order to prolong the network lifetime.

- The proposed algorithms are implemented on a real sensor node to assess the feasibility of the proposed methods on the low-resource sensor nodes. We also carried out a thorough analysis of the proposed model through a simulation study, to estimate the detection rate of anomalies and energy consumption.

The remainder of this paper is organized as follows: Section II provides a brief overview of the related work. The architecture of the proposed anomaly detection system is explained in Section III. Algorithms of the proposed system are presented in Section IV. Section V provides analysis of the results obtained through implementation, simulation, and comparative study. Section VI concludes the paper.

II. RELATED WORKS

The use of anomaly detection in communication and networking systems to discover outliers, exceptions, or contaminants caused by in situ or in transit errors, faults, or attacks was started as early as in 1987 with the pioneer work carried out by Denning [11]. Her hypothesis advocates the use of abnormal pattern in the audit record to detect anomalies in the system. Her proposition later on served as a base to develop several real-time anomaly detection systems in various network types and applications. On the basis of the position of the anomaly detection component within the WSN, the existing anomaly detection schemes can be classified into three categories, namely, centralized, semi-centralized, and distributed [12]. The central authority, a base station, is responsible for carrying out the anomaly detection process in the centralized class of schemes. The cluster heads are responsible for detection in the semi-centralized schemes. The distributed schemes have a cooperative nature of working among cluster heads and cluster members nodes to accomplish the anomaly detection task. Based on this taxonomy, the research community has proposed several notable anomaly detection schemes for WSNs over the last decade [13]-[16]. However, none of these centralized, semi-centralized, or distributed schemes have focused on the identification of the source of anomalies, which is an important service to effectively rectify the causes of anomalies.

Over the years, several attempts have been made to use the mobile agent technology for anomaly detection in addition to other tasks such as localization, parallelism, and distributed data fusion [3]-[7]. One of the initial studies which advocates the use of mobile agent technology for anomaly detection was undertaken by Krugel et al. [3]. Their work suggests the dispatch of mobile agents as guards to roam among different nodes to accomplish random sampling. The comprehensive detection is initiated after the identification of the anomalous activities. Although this proposal reduces dispatch and arrival costs of the mobile agents, nodes are vulnerable to attacks in the absence of guards.

Ketel suggested the use of several static and mobile agents for distributed anomaly detection application in WSNs [4]. In his work, Static Agents (SA) are located at each cluster head and transmit the anomaly report to the Mobile Agent Server (MAS) after observing anomalous activities. In response, MAS triggers a mobile agent to the respective monitoring node. The mobile agents are further divided into thick and thin agents associated with rich and low resource nodes, respectively. The Victim Node List (VNL) is responsible for generating the itinerary of MA. However, the author has offered no explanation about the internal details of the MAS repository and working of the VNL. In another study, Pugliese and colleagues proposed a rule-based method to detect network-layer anomalies by using mobile agents [5]. The formalism based on Weak Process Models (WPM), a non-parametric version of Hidden Markov Model, is employed to reduce the rules of reachability. The threat model is classified into low and high types of attacks. However, the authors have only reported the early implementation results. The thorough performance analysis of the proposed method is required to establish its effectiveness for low resource WSNs.

Eludiora et al. investigated the use of mobile agents for a distributed intrusion detection application in which sensor nodes directly communicate with the base station instead of the cluster heads [6]. The mobile agents are employed to communicate among different base stations. This model is based on two algorithms. The first algorithm detects DoS attacks and consequently updates the status of the sensor node to the base station. The second algorithm
calculates the probability of the failure of the base station to discover anomalies. The major drawback of this study is its heavy reliance on the non-realistic assumption of one hop distance among sensor nodes and the base station. Khanum et al. presented an architecture of mobile agent based hierarchical IDS for WSN [7]. Their work employs three agents for anomaly detection: Analyzer Agent, Management Agent, and Coordinating Agent. The detection module is installed at each cluster head. Then anomalies are detected at two levels: node and network. Nevertheless, in their work no attempt has been made to quantify the proposed architecture through experimental or analytical study.

The analysis of the above-mentioned studies provides the evidence that none of the existing schemes is specifically designed for the CRM-based observations. Similarly, no scheme has considered the verification of source of anomalies that is desired to effectively counter them. Furthermore, existing mobile agent-based anomaly detection schemes have not considered space and transmission costs of mobile agents. Therefore, these schemes have overlooked an important aspect that the mobile agents cannot be freely used and transmitted over the network because of the memory and energy restrictions of WSNs. To overcome these issues, this study presents a detailed mobile agent-based anomaly detection system after due consideration of space and transmission costs. We have exploited association among the CRM-based normal profile features to detect several types of first-order anomalies. We have also proposed a method which employs mobile agents to perform the in situ verification in order to confirm the source of anomalies. Furthermore, the transmissions of the mobile agents are optimized by taking into account the anomalous instances of the historical and current observations. Unlike existing schemes, we have also performed the thorough theoretical and empirical analyses to investigate the efficiency and effectiveness of the proposed system.

III. ANOMALY DETECTION AND VERIFICATION ARCHITECTURE

In this section, we describe the network model and explain the role of the components of the anomaly detection module in the proposed system.

A. Network Model

We consider a network region with a large number of low resource MICAz sensor nodes. The network region is virtually segmented into several clusters which are governed by the resource rich Cluster Head (CH) nodes. The cluster member nodes (SN) periodically sense their ambient environment and transmit data to their respective CHs. The CH nodes keep track of the resource status of the SNs through the CRM-mechanism [9]. The CH nodes are also aware of the actions performed by the SNs such as sensing from the ambient environment and going into sleep and wake up modes. The CH nodes detect anomalies and take appropriate actions after filtering incoming data packets that are received from the SNs. The mobile agents are positioned at CH nodes that can transmit them for in situ verification of the SN. In addition to other traditional operations such as code and data dissemination, localization, and distributed data fusion. A resource rich node, that is, a personal digital assistant or laptop class device, acts as a Base Station (BS). The BS manages the overall network through a user application. The BS is a main decision-making authority, whereas CH nodes act as regional chiefs. The generalized structure of the mobile agent-based network model is depicted in Figure 1.

The overall course of the main events in the network model can be formally described as given below:

- \( <E_0, SN_R>: \) In the event \( E_0 \), the sensor reading \( R \) is collected by the sensor node \( SN \) from its ambient environment.
- \( <E_1, R>/<E_1, R^\prime>: \) In the event \( E_1 \), the normal reading \( R \) or the anomalous reading \( R^\prime \) is sent to the cluster head \( CH \).
- \( <E_2, Anly>: \) In the event \( E_2 \), the CH performs analysis on \( <E_1, R>/<E_1, R^\prime> \).
- \( <E_3, Anm_Agt>: \) In the event \( E_3 \), the anomaly agent \( Anm_Agt \) is transmitted to the SN for in situ verification.
- \( <E_4, Ins_V>: \) In the event \( E_4 \), the \( Anm_Agt \) performs in situ verification of the SN.
- \( <E_5, Ins_R>: \) In the event \( E_5 \), the SN transmits results of in situ verification to the CH.
- \( <E_6, Agg_R>/<E_6, Anm>: \) In event \( E_6 \), the aggregated data or anomaly report is forwarded to the BS.
- \( <E_7, Decp>: \) In the event \( E_7 \), the BS makes a decision on \( <E_6, Agg_R>/<E_6, Anm> \).
- \( <E_8, Dec_R>: \) In the event \( E_8 \), the report of \( <E_7, Decp> \) is sent to the CH.
- \( <E_9, Act>: \) In the event \( E_9 \), the CH takes an action on the sensor node.

Note that the order of the aforementioned events is only for the first cycle of the system at the time of deployment. During normal operation of the system, event \( E_9 \) may not occur. However, if BS wishes to update CH regarding the anomaly detection process, then event \( E_9 \) may take place. To have a better picture of the working of the overall system, the above-mentioned course of events can
be formally defined as illustrated below:

\[ S \in SN, CH & BS \]

\[ S_0 \in SN \]

\[ \sum = \{<E_0, SN_R>, <E_1, R>/<E_1, \overline{R}> <E_2, Anly>, <E_3, Anm_Agnt>, <E_4, InsY>, <E_5, Ins_R>, <E_6, Agg_R>/<E_6, Anm>, <E_7, Dec_P>, <E_8, Dec_R>, <E_9, Act>\} \]

\[ \wedge \in 0, 1 \]

\[ T : S \times \sum \rightarrow S = (SN \times <E_0, SN_R> \rightarrow SN), (SN \times <E_1, R> \rightarrow CH), (SN \times <E_1, \overline{R}> \rightarrow CH), (CH \times <E_2, Anly> \rightarrow CH), (CH \times <E_3, Anm_Agnt> \rightarrow SN), (SN \times <E_4, InsY> \rightarrow SN), (SN \times <E_5, Ins_R> \rightarrow CH), (CH \times <E_6, Agg_R> \rightarrow BS), (BS \times <E_7, Dec_P> \rightarrow BS), (BS \times <E_8, Dec_R> \rightarrow CH), (CH \times <E_9, Ins_Y> \rightarrow SN) \} \]

\[ G : S \rightarrow \wedge = \{ (SN \rightarrow 0,1), (CH \rightarrow 0,1), (BS \rightarrow 0,1) \} \]

In the above definition, “0” and “1” denote normal and anomalous states, respectively. Note that we have assumed CH and BS as secure, however, they are prone to faults, thus they can have values of either 0 or 1.

**B. Anomaly Detection Module**

Each cluster head is equipped with an Anomaly Detection Module (ADM). The ADM performs several key jobs including first-order anomaly detection, in situ verification using mobile agents, and agent transmission optimization. The ADM is composed of three components: Coordination Unit, Anomaly Agent, and Repository. Figure 2 illustrates the structure of the ADMs positioned on the different CH nodes.

1) **Coordination Unit**: The Coordination Unit (CU) is a core element of the ADM. It performs intra-ADM components coordination and anomaly detection related communication with other entities of the network such as BS. The CU extracts values of “features of interest” from incoming network traffic to perform anomaly detection as per anomaly detection criteria. The process of anomaly detection is described in Section IV-B. The normal sensor reading is forwarded to the Aggregation Unit (AU) after the anomaly detection process. The AU is a logical unit within the cluster head that aggregates sensor readings and forwards them to the BS after a pre-defined period of time. On the other hand, if received observation is found anomalous, then the CU may take the following action(s) against the anomalous node: (a) trigger anomaly agent for in situ verification, (b) generate alarm to the BS, (c) announce node as malicious or faulty to other nodes in the cluster, and (d) minimize communication with suspicious node.

2) **Anomaly Agent**: The anomaly agent (a software mobile agent) is composed of four parts, viz., identity, itinerary, code, and data. Each anomaly agent has a unique identity which links it with target cluster member sensor node. The itinerary of the agent contains the address of the destination node. The code segment is made up of the code for in situ verification process and data portion consists of the values of the historical observations that are used for in situ verification of anomalous sensor node. Figure 3 depicts the internal structure of the anomaly agent.

The CU may trigger an anomaly agent for the in situ verification of the suspicious sensor node after detecting anomalies in received observations. One of three situations may arise after transmission of an anomaly agent: (a) the successful execution of the anomaly agent on the suspicious sensor node and transmission of the in situ verification result to the CH, (b) the suspicious node may defend the investigation of the anomaly agent and as a consequence does not send any result to the CH, and (c) migration of the anomaly agent to the suspicious sensor node is vulnerable to the agent execution manipulation attack. In case (a), the anomaly agent will send the desired result to the CH. In case (b), if the suspicious...
node defends the investigation of the anomaly agent and does not send any result to the cluster head, this will confirm its anomalous status, which is the ultimate goal of sending the anomaly agent. In case (c), the agent execution integrity protection mechanism can be employed to avoid the execution manipulation attack [17]. Note that case (c) is only applicable to those situations where SNs are vulnerable to more sophisticated attacks and the adversary takes complete control of the node. The agent execution integrity protection mechanism is not required for anomalies caused by faults or other types of attacks. The algorithm for the in situ verification of SN using anomaly agent is elucidated in Section IV-C.

C. Repository

The repository, Rep, stores normal profile features and other related data to perform anomaly detection and mobile agent transmission processes in the form of five tuples. The structure of the repository is an extension to the initial idea reported in [11]. The tuples of the proposed system can be expressed as shown below.

\[ \text{Rep} = \langle N_{id}, \text{Res}_{set}, \text{Fet}_{set}, \text{Anm}_{obs}, \text{Act}_{set} \rangle \] (1)

In the above equation, \( N_{id} \) is a column vector which stores identities of the cluster member SNs. The tuple \( \text{Res}_{set} \) denotes resource status of SNs. Each SN has multiple resources such as battery and memory, which are stored as \( m \times n \) matrix, where \( m \) denotes a number of SNs in the cluster and \( n \) represents their resources. The \( \text{Fet}_{set} \) tuple is based on “features of interest” which are used for anomaly detection. By default, the sensor nodes with a similar role in the network have single \( \text{Fet}_{set} \) in order to optimize the overall memory consumption of the cluster head. However, different values of \( \text{Fet}_{set} \) of similar nodes can also be accommodated depending on the security requirements and available memory space. The basic structure of the \( \text{Fet}_{set} \) is given below.

\[ \text{Fet}_{set} = \langle \lambda, T, \varphi, \upsilon, F \rangle \] (2)

In the above equation, \( \lambda \) defines minimum to maximum bounds on sensor reading values. For example, during summer daytimes, feature \( \lambda \) can hold values between 15°C - 35°C to define the normal behavior of the SN. The feature \( T \) denotes the time interval to monitor the activities of the SN for a certain period of time. This feature keeps track of the normal behavior of the SN with respect to other features such as resource status and sensor reading. The feature \( \varphi \) stores values for entitled actions performed by the SN such as sensing, sleeping, wake-up, and transmission of observations. The features \( \upsilon \) and \( F \) denote the status of the resources of the SN and received packet count, respectively.

The tuple \( \text{Anm}_{obs} \) is \( w \times n \) matrix which stores \( w \) number of historical anomalous observations to facilitate the computation of threshold values to optimize anomaly agent transmission. The agent transmission optimization method is described in Section IV-D.

The structure of \( \text{Anm}_{obs} \) can be expressed as stated below.

\[ \text{Anm}_{obs} = \text{Fet}’_{set} \] (3)

The \( \text{Fet}’_{set} \) holds values for \( \lambda’, T’, \varphi’, \upsilon’, \) and \( F’ \), where \( \lambda’ \) represents anomalous sensor reading values which lie outside the normal bounds, \( T’ \) denotes anomalous activities of the SN with respect to time, \( \varphi’ \) represents unauthorized actions performed by the SN, and \( F’ \) denotes the abnormal frequency of received packets.

The tuple \( \text{Act}_{set} \) defines the set of actions which facilitates functionality of the anomaly detection system. It is composed of two types of actions, namely, \( \kappa \) and \( \tau \).

\[ \text{Act}_{set} = \langle \kappa, \tau \rangle \] (4)

In the above equation, \( \kappa \) denotes the anomaly detection action, which performs the anomaly detection after arrival of each observation. After the anomaly detection process, the anomaly detection module forwards the normal sensor reading to the aggregation unit. If an anomalous observation is received, the anomaly detection module would transmit the anomaly agent to the SN or take different actions such as generating an alarms to the BS, announcing the SN as malicious or faulty to other nodes in the cluster, and minimizing communication with the suspicious node for those observations that lie in the tolerance zone. The tuple \( \tau \) represents tuning actions such as updates in first-order join bounds, and also in normal, tolerated, and anomalous zones for the \( \text{Fet}_{set} \) of the SNs. It is submitted that the action \( \kappa \) is executed automatically by the ADM on the CH after arrival of each observation, whereas the action \( \tau \) is initiated by the network administrator from the BS. Tables I and II represent definitions of sample set of actions. Accordingly, Figures 4 and 5 show flows of these actions.
IV. ALGORITHMS AND ANALYSIS

The workflow of the proposed anomaly detection system consists of four processes, namely, features collection, anomaly detection, in situ verification, and anomaly agent transmission optimization. The notations used for the pseudocodes of the proposed algorithms and their analysis are listed in Table III.

A. Features Collection on Sensor Node

At time $t_1$, the $g_{th}$ sensor node, $SN_g$, in the $q_{th}$ cluster, collects values of the features $f_i \in Fet_{set}(1)$. The $Fet_{set}(1) = \{\lambda, \varphi, \nu\}$, $Fet_{set}(2) = \{T, F\}$, and $Fet_{set} = Fet_{set}(1) \cup Fet_{set}(2)$, where $f_i$ denotes the value of each individual feature and $\lambda, \varphi, \nu, T, F$ denote sensor reading, resource status, entitled action, time interval, and packet count, respectively, as described in Section III-C. The $Fet_{set}(1)$ is stored by the $SN_g$ in its stack memory and transmitted to the cluster head, $CH_q$, as observation $Obs_j$. The values of $Fet_{set}(1)$ are used by the anomaly agent for the in situ verification process. The previous values of the features of $Fet_{set}(2)$ are computed on the $CH_q$ after arrival of the $Obs_j$ and used for detection of the first-order anomalies along with the received values of the $Fet_{set}(1)$. The pseudocode for the features collection process on the $SN_g$ is illustrated in Algorithm 1.

Algorithm 1 Features, $Fet_{set}(1) = \{\lambda, \varphi, \nu\}$, collection on $SN_g$

1: At time $t_1$
2: while $j < m$ do
3: $SN_g$ collects $f_i \in Fet_{set}(1)$
4: PUSH $f_i$ in stack of $SN_g$
5: $Obs_j \leftarrow Obs_j + f_i$
6: $j \leftarrow j + 1$
7: endwhile
8: TRNSMT $Obs_j$ to $CH_q$

B. Anomaly Detection on Cluster Head

One of the objectives of the proposed anomaly detection system is to maximize the use of received CRM-based observation, $Obs_j$, values and computed information of $Fet_{set}(2) = \{T, F\}$ for the anomaly detection process, where $Obs_j = Fet_{set}(1) = \{\lambda, \varphi, \nu\}$. To this end, we exploit the statistical association among features of interest, $Fet_{set} = \{\lambda, \varphi, \nu, T, F\}$, to detect certain types of group anomalies, which are caused by in situ fault or attack, resource exhaustion attack, denial of sleep attack, and faults on node. We define the first-order join as a two-dimensional association among two features by defining bounds on each feature to compute normal region. On the basis of this definition, we establish a first-order join between $\lambda$ and $T$ features to detect anomalies caused by in situ fault or attack. The symptom of such anomalies on the $CH_q$ is the receipt of the erroneous values of the sensor readings with respect to the time intervals. The combined normal region for $\lambda$ and $T$ features can be computed from the following equation.
The first-order joins of the feature set, corresponding anomalous events and their description are provided in Table IV.

To detect the above mentioned anomalies, the CHq receives the Obsj from the SNq. Then the coordination unit extracts the values of the \( F_{set(1)} = \{ \lambda, \varphi, \nu \} \) from the received Obsj to perform the anomaly detection process using first-order joins. If the \( SN_q \) is found normal on the basis of received Obsj, then the sensor reading is forwarded to the aggregation unit. On the other hand, if the \( SN_q \) is found anomalous, then the anomaly detection algorithm invokes the agent transmission optimization procedure (Phase 2 of Algorithm 4) that returns the \( SN_q \)'s behavior as anomalous (\( Beh_{Anm} \)), tolerated category 1 (\( Beh_{Tol_1} \)), tolerated category 2 (\( Beh_{Tol_2} \)), or tolerated category 3 (\( Beh_{Tol_3} \)). For anomalous behavior, the anomaly detection module transmits the anomaly agent to the \( SN_q \) to perform the in situ verification. On the other hand, for tolerated categories 1, 2, and 3, the anomaly detection module announces the \( SN_q \) as anomalous to the other cluster member nodes and cluster head nodes, minimizes the communication with the \( SN_q \), and generates an alarm to the base station, respectively.

The pseudocode for the anomaly detection process is given in Algorithm 2.

Algorithm 2 Anomaly detection on \( CH_q \)

1: \( CH_q \) receives \( Obs_j \) from \( SN_q \)
2: CU extract \( F_{set(1)} = \{ \lambda, \varphi, \nu \} \) from \( Obs_j \)
3: Compute \( F_{set(2)} = \{ T, F \} \)
4: \( CHK (Anm_{1,rd}) = \int_{\lambda_1}^{\lambda_2} \int_{T_j}^{T_f} f(\lambda, T) \, dT \, d\lambda \)
AND \( N(\lambda, T) = \int_{T_j}^{T_f} \int_{\nu_1}^{\nu_2} f(T, \nu) \, d\nu \, dT \)
AND \( N(\varphi, \nu) = \sum_{\varphi_i}^{} \int_{\nu_1}^{\nu_2} f(\varphi, \nu) \, d\nu \)
AND \( N(\varphi, T) = \sum_{\varphi_i}^{} \int_{T_j}^{T_f} f(\varphi, T) \, dT \)
AND \( N(F, T) = \sum_{F_1}^{F_i} \int_{T_j}^{T_f} f(F, T) \, dT \)
5: if \( CHK (Anm_{1,rd}) \) \( \equiv \) TRUE then
6: \( Fwd (Aggr) \) // sensor reading is forwarded to the aggregation unit
7: else
8: CALL \( Beh = Agnt_{Opt}(F_{set(1)}) \) // invoke Algorithm 4 Phase 2
9: if \( CHK (Beh = Beh_{Anm}) \) \( \equiv \) TRUE then
10: \( Transmt \) \( Anm_{Agnt} \) to \( SN_q \)
11: else if \( CHK (Beh = Beh_{Tol_1}) \) \( \equiv \) TRUE then
12: \( Aggr_{Unf} \) store \( SR \) AND \( CU \) // announce the \( SN_q \) as anomalous to the cluster member nodes and other \( CHs \)
13: else if \( CHK (Beh = Beh_{Tol_2}) \) \( \equiv \) TRUE then
14: \( Aggr_{Unf} \) store \( SR \) AND \( CU \) // minimizes the communication with the \( SN_q \)
15: else if \( CHK (Beh = Beh_{Tol_3}) \) \( \equiv \) TRUE then
16: \( Aggr_{Unf} \) store \( SR \) AND \( CU \) // generates an alarm to the BS
17: end if
18: end if
C. Suspicious Node Verification

The sensor node, $SN_g$, receives the anomaly agent and stores it in its memory portion which is specified for the anomaly agent. The memory portion has a dedicated stack memory segment that is used to store the data which is brought by the anomaly agent for the in situ verification process. The length of the stack memory portion is customizable. The low memory sensor nodes may store the values of only few observations as compared to the rich memory nodes. However, the minimum size of the stack memory should be large enough to accommodate the anomaly agent arrival and comparison of values for the in situ verification process. The arrival of the anomaly agent is dependent on two factors: time taken by the cluster head for the anomaly detection and the anomaly agent trip time from the cluster head to the $SN_g$. The anomaly agent brings values of three features (i.e., sensor reading, resource status, and actions performed) for the in situ verification of the $SN_g$. After the arrival of the anomaly agent, the comparison between the values of the $SN_g$ stack and anomaly agent data is performed. If all values are matched, then the sensor node is considered as normal and result “0” is transmitted to the cluster head. Otherwise, in the case of anomaly, result “1” is sent to the corresponding cluster head. Algorithm 3 describes the pseudocode for the in situ verification process.

Algorithm 3 In situ Verification

1: $SN_g$ receive $Ann_Agent$ from $CH_g$
2: $PUSH$ $Ann_Agent$ in $Agent_stck$
3: while $Stack\_SN_g! = \varepsilon$ do
4: $CMP(Stack\_SN_g[i], Agent\_stck[k])$
5: if $CMP$ is $TRUE$ then
6: $COUNT$ $Stack\_SN_g[i]by 1$
7: else
8: $Vrf\_rs\_SN_g \leftarrow 1$
9: Break
10: end if
11: $Vrf\_rs\_SN_g \leftarrow 0$
12: end while
13: $TRNSMT$ $Vrf\_rs\_SN_g$

D. Anomaly Agent Transmission Optimization

The anomaly detection module may tolerate the anomalous behavior of the sensor node, $SN_g$, to some extent to optimize anomaly agent transmission for the in situ verification process. A relatively naive approach for agent transmission optimization can be performed by the examination of each feature, $f_i$, for its presence in the normal (i.e., $−\sigma \leq f_i \leq \sigma$) tolerated (i.e., $1\sigma < f_i \leq 2\sigma OR −2\sigma \leq f_i \leq −2\sigma$) or anomalous (i.e., $f_i > 2\sigma OR f_i < −2\sigma$) zones. Then, the anomaly agent transmission can be curtailed for the tolerated zone and triggered only for the anomalous zone [26], [27]. However, this approach does not consider the past behavior of the sensor node and transmitting the anomaly agent merely based on the presence of the current anomalous observation, may cause excessive transmission of the anomaly agent. Therefore, we optimize the anomaly agent transmission by taking into account the weighted sum of the instances of the historical and current observations.

We define the $SN_g$’s historical observations score, $S_{SN_g}$, as stated below.

$$S_{SN_g} = \alpha_1\frac{\Omega_{11}}{h} + \alpha_2\frac{\Omega_{12}}{h}$$ (10)

In the above equation, $\alpha_1$ and $\alpha_2$ are the weighting factors for the tolerated and anomalous instances of $f_i$, respectively. The weighting factors $\alpha_1 + \alpha_2 = 1$ and $\alpha_2 > \alpha_1$. The weighting factor $\alpha_2$ is assigned a higher value due to the fact that it is associated with anomalous instances of the $f_i$.

In the equation (10), the $S_{SN_g}$ has value in the unit interval $[0, 1]$ and is evaluated as a function of the two main parameters:

- $\Omega_{11}$: The number of $f_i$ instances from $h$ number of historical observations which satisfy the inequality condition $\sigma < f_i \leq 2\sigma OR −1\sigma > f_i \geq −2\sigma$.
- $\Omega_{12}$: The number of $f_i$ instances from $h$ number of historical observations which satisfy the inequality condition $f_i > 2\sigma OR f_i < −2\sigma$.

By the above definitions, it is clear that the parameters $\Omega_{11} + \Omega_{12} = h$. In order to optimize the values of the weighting factors parameters $\alpha_1$ and $\alpha_2$, the equality $\alpha_1 + \alpha_2 = 1$ may be rearranged as $\alpha_2 = 1 − \alpha_1$, and may be fixed as constant $c$, and $\Omega_{11} + \Omega_{12} = h$ may be rearranged as $\Omega_{12} = c − \Omega_{11}$. As a consequence, the equation 10 can be rewritten as given below.

$$S_{SN_g} = \alpha_1\frac{\Omega_{11}}{c} + \alpha_2\frac{c − \Omega_{11}}{c}$$ (11)

Since $\alpha_1 + \alpha_2 = 1$ and $\alpha_2 > \alpha_1$, thus $\alpha_1 \in [0, 0.5)$. In the following discussion, to simplify the procedure we fix the window size of the historical observations, i.e., $h$ as $c$, where $c = 10$, but the procedure is straightforward to generalize. This implies $\Omega_{11} \in [1, 10]$, where $\Omega_{11} = 1$ means there exists only a single observation in $h$ that lies in the tolerance zone. On the other hand, $\Omega_{11} = 10$ shows that there exists all historical observations in the tolerance zone and no observation in the anomalous zone.

For optimization of the weighting parameters, we choose the mean values of the objective functions and identify the corresponding mean value of the first parameters: $\alpha_1$ and $\beta_1$, which are associated with the tolerated instances of $f_i$. The maximum value of the first parameters are not chosen due to the fact that the second parameters: $\alpha_2$ and $\beta_2$ have high priority due to their association with the anomalous instances of $f_i$.

To optimize the parameters $\alpha_1$ and $\alpha_2$ such that the $S_{SN_g}$ achieves the mean value, we define the following objective function, which is derived from the equation (11), where $w$ and $x$ represents $\alpha_1$ and $\Omega_{11}$, respectively.

$$f(w, x) = (1 − w) + (2w − 1)\frac{x}{10}$$ (12)
Let $w \in [0, 0.5]$ and $x \in [1, 10]$ based on the facts that $\alpha_1 \in [0, 0.5)$ and $\Omega_1 \in [1, 10]$ correspondingly. This gives the mean value of the objective function $f(w, x) = 0.47$. The corresponding values of $w$ that yields the mean value of the objective function $f(w, x)$ lies in the interval $[0.35, 0.47]$ with 0.005 deviation on the both sides of the mean value. The mean of the interval yields $0.415 \approx 0.42$, which we take as the optimum value for $w$ for the above stated scenario. Figure 6 depicts the three-dimensional view of the objective function $f(w, x)$.

From the above mathematical procedure, we obtained the optimized value of $\alpha_1 = 0.42$. By the implication of the relation $\alpha_2 \approx 1 - \alpha_1$, the value for the parameter $\alpha_2$ can be deduced as 0.58.

The final score for the anomaly agent transmission can be derived from the following equation.

$$S_{MA_{trn}} = \frac{S_{SN_y}}{2} + \beta_1 u + \beta_2 v$$

(13)

In the above equation, the parameters $\beta_1$ and $\beta_2$ are the weighting factors for the current observation which lies in $1\sigma < f_i \leq 2\sigma \ OR \ -1\sigma > f_i \geq -2\sigma$, and $f_i > 2\sigma \ OR \ f_i < -2\sigma$ zones, respectively. Note that the weighting factors sets $\{\alpha_1, \alpha_2\}$ and $\{\beta_1, \beta_2\}$ are independent of each other. The weighting factors $\beta_1 + \beta_2 = 1$ and $\beta_2 > \beta_1$. The weighting factor $\beta_2$ holds the higher value because it is associated with the current anomalous instance of the $f_i$.

In the equation (13), the $S_{MA_{trn}}$ has value in the unit interval $[0, 1]$ and is estimated as a function of three parameters, namely, $S_{SN_y}$, $u$, and $v$. The $S_{SN_y}$ is obtained from the equation 10, whereas $u$ and $v$ are defined below.

- $u = 1$, if and only if current $f_i$’s instance satisfy the inequality condition $1\sigma < f_i \leq 2\sigma \ OR \ -1\sigma > f_i \geq -2\sigma$, otherwise $u = 0$.
- $v = 1$, if and only if current $f_i$’s instance satisfy the inequality condition $f_i > 2\sigma \ OR \ f_i < -2\sigma$, otherwise $v = 0$.

The relation between parameters $u$ and $v$ is defined as $v = 1 - u$. The weighting factors equality $\beta_1 + \beta_2 = 1$ may be rearranged as $\beta_2 = 1 - \beta_1$. As a consequence, the equation 13 can be rewritten as shown below.

$$S_{MA_{trn}} = \frac{S_{SN_y}}{2} + 2\beta_1 u - \beta_1 - u$$

(14)

Since $\beta_1 + \beta_2 = 1$ and $\beta_2 > \beta_1$, thus $\beta_1 \in [0, 0.5)$ and $S_{SN_y} \in [0, 1]$.

In order to optimize the parameters $\beta_1$ and $\beta_2$ such that $S_{MA_{trn}}$ achieves the mean value, we define the following objective function, which is obtained from the equation (14), where $y$ and $z$ represent $\beta_1$ and $S_{MA_{trn}}$, respectively.

$$f(y, z) = \begin{cases} \frac{y}{2} + 1 - y, & u = 0, \\ \frac{y}{2} + y, & u = 1. \end{cases}$$

(15)

To compute the mean value of the objective function $f(y, z)$, let $y \in [0, 0.5)$ and $z \in [0, 1]$ analogous to $\beta_1 \in [0, 0.5)$ and $S_{SN_y} \in [0, 1]$, respectively. The objective function $f(y, z)$ has two cases, viz., case 1: $u = 0$ and case 2: $u = 1$. In case 1, the min, max, and mean values are 1, 1.01, and 1.0049, respectively. On the other hand, in case 2, the min, max and mean values are 0, 0.99, and 0.5048, respectively. The corresponding value for the parameter $y$ for both cases lies in the interval $[0.2401, 0.2499]$. The mean of the interval yields $0.245 \approx 0.25$, which we consider as an optimum value for the parameter $y$.

Based on the computation on the objective function $f(y, z)$, we estimated the optimized value of parameter $\beta_1 = 0.25$. Again by the implication of the relation $\beta_2 = 1 - \beta_1$, the value for the parameter $\beta_2$ is obtained as 0.75.

The derived anomaly agent transmission score should be greater than a pre-defined threshold $\psi$ to transmit an anomaly agent and $\psi \in (0, 1)$. Note that we set the upper bound clipping level of the overall anomaly agent transmission score, i.e., $S_{MA_{trn}}$, as 1. This will, however, not affect the anomaly agent transmission decision, since any value above $\psi$ is deemed as suitable for anomaly agent transmission. Furthermore, if the anomaly agent transmission score is lower than $\psi$, then the anomaly detection module will take other routine actions such as making an announcement of $SN_y$ as anomalous to the cluster member nodes and other cluster head nodes, minimizing communication with the $SN_y$, and generating an alarm to the base station for tolerated categories 1, 2, and 3, respectively (as described in Section IV-B). This approach causes less frequent transmissions of the anomaly agents which reduces energy consumption and increases overall network lifetime. The pseudocode for the anomaly agent transmission optimization method is illustrated in Algorithm 4. Note that Phase 1 in Algorithm 4 is computed only at the time of the system deployment and whenever tuning action $\tau$ is performed by the system administrator, whereas Phase 2 is executed by the anomaly detection module for each received anomalous observation.
Algorithm 4 Zone Computation and Anomaly Agent Transmission Optimization

Phase 1: Compute zones (Fet<sub>set</sub>)

1: At t<sub>k</sub> time
2: for each f<sub>i</sub> from Fet<sub>set</sub> do
3: Compute −1σ, 1σ, −2σ, 2σ // threshold values
4: Update (Nor<sub>zone</sub>) = −1σ ≤ f<sub>i</sub> ≤ 1σ
5: Update (Tol<sub>zone</sub>) = 1σ < f<sub>i</sub> ≤ 2σ AND −1σ > f<sub>i</sub> ≥ −2σ
6: Update (Ann<sub>zone</sub>) = f<sub>i</sub> > 2σ AND f<sub>i</sub> < −2σ
7: end for

Phase 2: Transmission optimization (Beh = \textit{Agn}_{\text{Opt}}(Fet<sub>set</sub>))

1: for each f<sub>i</sub> from Fet<sub>set</sub> do
2: Compute SN<sub>y</sub> = α₁(\frac{Ω}{α₂}) + α₂(\frac{Ω}{α₃})
3: Compute SM<sub>Ann</sub> = SN<sub>y</sub> / 2 + β₁γ + β₂z
4: if SM<sub>Ann</sub> ≥ ϱ then
5: Beh ← Beh<sub>Ann</sub>
6: break
7: else if SM<sub>Ann</sub> < ϱ AND ≥ ζ then
8: Beh ← Beh<sub>Tol</sub>γ
9: break
10: else if SM<sub>Ann</sub> < ζ AND ≥ γ then
11: Beh ← Beh<sub>Tol</sub>γ
12: break
13: else if SM<sub>Ann</sub> < γ AND ≥ η then
14: Beh ← Beh<sub>Tol</sub>η
15: break
16: else if SM<sub>Ann</sub> < η then
17: Beh ← Beh<sub>Nrm</sub>
18: else
19: f<sub>i</sub> + +
20: end if
21: end for
22: return Beh to

E. Space and time complexities

Theorem 1: (a) The space complexity for the features collection process on SN<sub>y</sub> is bounded from above by l[m], (b) for anomaly detection process on CH<sub>y</sub> it is C<sub>u</sub>, (c) for in situ verification process of SN<sub>y</sub>, space complexity is l[y], and (d) for agent optimization process on CH<sub>y</sub>, space complexity is constant C<sub>v</sub>.

Proof: (a) Let Fet<sub>set</sub>(1) = \{λ, ϕ, v, T, F\}, the Fet<sub>set</sub>(2) = \{λ, ϕ, v, T, F\}. The Fet<sub>set</sub>(2) takes l[j] space in the memory. Therefore, the total memory space taken by the n number of features of Fet<sub>set</sub> becomes l[n] = l[m] ∪ l[j], where l[m] < l[j]. The CH<sub>y</sub> takes constant memory spaces C₁, C₂, and C₃ to store threshold values for the first-order joins, aggregated value, and result of agent optimization process, respectively. The anomaly agent consumes C₄ and C₅ memory spaces to store the code and data of the anomaly agent, respectively. Thus, the total memory space taken by the anomaly detection process is ∪<sub>n=1</sub>C₅ + l[n].

(c) Let SN<sub>y</sub>[y] be the length of the stack memory of the SN<sub>y</sub> to accommodate the in situ verification process. The SN<sub>y</sub>[y] should satisfy the two conditions: (i) l[n] < SN<sub>y</sub>[y] and (ii) C₆ < SN<sub>y</sub>[y]. where C₆ = C₄ ∪ C₅. This means that the total memory of the SN<sub>y</sub> should accommodate the collected values of the Fet<sub>set</sub>(1) and the code and data of the anomaly agent for the in situ verification process. Considering l[y] as the upper bound for the collective memory of Fet<sub>set</sub>(1) for both l[n] and C₆, the space complexity for the in situ verification of the suspicious SN<sub>y</sub> is l[y].

(d) The CH<sub>y</sub> takes constant memory spaces C₇, C₈, C₉, and C₁₀ to store the values of the weighting factors α₁, α₂, β₁, and β₂, respectively. The memory spaces C₁₁ and C₁₂ are used by the CH<sub>y</sub> to store the values of the instances of f<sub>i</sub> from w number of historical tolerated and anomalous observations. Similarly, the CH<sub>y</sub> takes C₁₃ space in the memory to store the value of the tolerated or anomalous instance of the current received observation. The memory spaces C₁₄, C₁₅, C₁₆, and C₁₇ are taken by the CH<sub>y</sub> to store the historical observations score of the SN<sub>y</sub>, the agent transmission score, the agent transmission threshold value, and the behavior status of the SN<sub>y</sub>, respectively. Considering C₉ = ∪<sub>v=1</sub>C₉, the space complexity of the agent transmission process is constant C<sub>v</sub>.

Theorem 2: (a) The time complexity for the features collection process on the SN<sub>y</sub> is O(m), (b) the anomaly detection process on the CH<sub>y</sub> runs in a constant time D for normal observations and has time complexity of O(n) for anomalous observations, (c) the time complexity for the in situ verification process of the SN<sub>y</sub> is O(x), and (d) for the anomaly agent transmission optimization process on the CH<sub>y</sub> is O(y).

Proof: (a) The time complexity for the features collection process is mainly based on three features of the Fet<sub>set</sub>(1) = \{λ, ϕ, v\}, whereas the Fet<sub>set</sub>(2) = \{T, F\} is computed on the CH<sub>y</sub> after receiving values of Fet<sub>set</sub>(1) from the SN<sub>y</sub>. Let us assume that the SN<sub>y</sub> consumes m time to collect Fet<sub>set</sub>(1) from its ambient environment and to store in its memory. The SN<sub>y</sub> takes constant time D₁ to transmit the m number of features to the CH<sub>y</sub>. Considering the upper bound case, the complexity of the features collection process is O(m).

(b) The CH<sub>y</sub> consumes constant time D<sub>2</sub> to receive the observation from the SN<sub>y</sub>, D<sub>3</sub> time to retrieve Fet<sub>set</sub>(1) values, and D<sub>4</sub> time to compute the Fet<sub>set</sub>(2) values. To
aggregate the sensor reading, the \( SN_g \) consumes \( D_8 \) time to transmit the result to the BS for analysis. The \( CH_q \) takes \( D_9 \) time to perform the anomaly detection process using first-order joins and assigning relevant behavior to the \( SN_g \), which is retrieved after the computation performed by the agent transmission optimization process, where \( D_9 > D_5 \). The \( CH_q \) takes \( D_7 \) time for the anomaly agent transmission to the \( SN_g \), where the \( SN_g \) is anomalous in this case. Thus, considering \( D = \sum_{i=2}^{\infty} D_i \), the algorithm for the anomaly detection process runs in a constant time \( D \). Moreover, Algorithm 2 calls agent transmission optimization phase for anomalous observations which has \( O(n) \) time complexity (as shown in the proof of Theorem 2(d)).

(c) The \( SN_g \) receives the anomaly agent and returns the in situ verification result to the \( CH_q \) in \( D_8 \) and \( D_9 \) times, respectively. The \( SN_g \) consumes the \( x \) time to perform comparison operation between the values of \( Agnt_{stick} \) and \( SN_g_{Stick} \). Therefore, by considering the upper bound on the time taken by the comparison process, the time complexity for the in situ verification algorithm is \( O(x) \).

(d) The \( CH_q \) takes \( y \) time for the examination of different zones for the values of \( n \) number of features to classify them as normal, tolerated, or anomalous to perform the agent transmission optimization process. By considering the upper bound case, the running time for the agent transmission optimization process becomes \( O(y) \).

Note that the above proofs should satisfy the relation \( O(y) < O(m) < O(x) < O(n) \) due to the fact that Algorithms 2 and 3 involve the transmission, receiving, and processing of anomaly agent, whereas Algorithms 1 and 4 merely include processing on the \( F_{elset} \).

V. PERFORMANCE EVALUATION

The performance of the proposed system is examined through simulation study, implementation of the proposed algorithms on MICAz mote, and comparison with the competing schemes.

A. Simulation Study

In order to evaluate the performance of the proposed system, we developed simulation scenarios in the object-oriented and discrete event environment, which simulates events in the chronological order [24]. The simulation environment is based on the following models and procedures.

- **Network model:** The simulation plane is based on the \( Wd \times Lg \) square meter area. The \( k \) number of nodes are randomly deployed in the different simulation scenarios. The base station is placed at \( (a, b) \) position in the simulation field.

- **Node model:** The resource capability of the MICAz mote is used as a node model [18]. The program, the SRAM, and the flash data logger memories of the sensor nodes are set as \( M_P, M_{SRAM}, M_{FLASH} \), respectively. At the time of deployment, the total energy of the node is considered as \( N_E \). The energy consumed by the sensor node during sleep mode is assumed as \( N_{E_{slp}} \).

- **Node Categorization:** In order to investigate the energy consumption, we considered the wireless smart home sensor network scenario. We categorized the sensor nodes as security and non-security devices. The security related sensors may include motion detector, door sensor, and light detector. On the other hand, humidity, temperature, and pressure detectors are a few examples of the non-security sensors. On the basis of this classification, we set a tight bound, \( \sigma \), for the security nodes. On the other hand, we set a relatively loose bound on the non-security devices, which is \( 2\sigma [25] \).

- **Communication energy dissipation model:** We used a well-known model for the radio hardware energy consumption [19]. The reason for using simplified model is highly stochastic nature of radio wave propagation, which makes it hard to model. The following formulas are used to estimate the energy dissipation by the radio hardware to transmit \( l \)-bits over the distance \( d \).

\[
E_{Tx}(l, d) = E_{Tx-elec}(l) + E_{Tx-amp}(l, d) \quad (16)
\]

\[
P_U(u) = \begin{cases} 
    l E_{elec} + l f_s d^2 & d < d_0, \\
    l E_{elec} + l m_p d^4 & d \geq d_0.
\end{cases} \quad (17)
\]

Similarly, to receive \( l \)-bits, the energy dissipation is estimated from the following equation.

\[
E_{Rx}(l) = E_{Rx-elec}(l) + l E_{elec} \quad (18)
\]

In the above equations, the \( E_{elec} \) illustrates the electronic energy dissipation based on multiple factors such as filtering, modulation, digital coding, and signal spreading. Similarly, \( f_s d^2 \) and \( m_p d^4 \) are the amplifier energies, which are based on several factors such as acceptable bit-error rate and distance between sender and receiver. Note that the \( E_{elec} \) is equivalent to \( E_{TX} + E_{DA} \) for cluster heads and \( E_{TX} \) for other nodes for transmission, where \( E_{TX} \) denotes the transmission energy and \( E_{DA} \) represents the data aggregation energy. On the other hand, the value of the \( E_{elec} \) is equivalent to \( E_{Rx} \) for cluster heads and other nodes while receiving data packets.

- **Network lifetime:** We employed the “first node die” approach for our experiments [22].

- **Propagation model:** The free space (\( d^2 \) power loss) and multipath fading (\( d^4 \) power loss) channel models are used for the propagation [20]. These models are based on the distance between the sender and the receiver. If the distance between the sender and the receiver is less than that of the threshold, \( d_0 \), then the free space, \( f_s \), otherwise the multipath, \( m_p \), channel model is used.
Traffic model: The nodes are characterized to transmit the periodic data packets. The sensor nodes transmit the data packets after every $t$ seconds on average. Each transmitted data packet has $Pkt$ size including header and payload. The payload of the normal data packet, as depicted in Figure 7, is based on the values of $F_{\text{el}}(1)$, which are transmitted by the cluster member nodes. In our experiments, we consider features $\lambda$, $\nu$, and $T$ as continuous, whereas $\psi$ and $F$ as discrete random variables.

Cluster formation: The clusters are formed through the Low-Energy Adaptive Clustering Hierarchy (LEACH) algorithm [19]. LEACH is one of the most commonly used cluster formation algorithms for WSNs. Our scenarios are based on heterogeneous nodes and LEACH supports such nodes to form the cluster-based network topology.

Anomaly Agent: The size of the developed anomaly agent is 762 bytes (including identity, itinerary, code, and data). The anomaly agent cannot travel in one message due to its large size. Therefore, we segment the anomaly agent into multiple frames as per 802.15.4 specifications [21]. According to the 802.15.4 specifications, the maximum permitted frame size is 127 bytes (i.e., payload = 102 bytes and header = 25 bytes). Therefore, we segment the anomaly agent into 8 packets. The header size is $8 \times 25 = 200$ bytes. The payload size for the first 7 packets is $7 \times 102 = 714$ bytes and for the last packet $1 \times 48 = 48$ bytes. The total size of the first 7 packets, on the basis of the formula $((\text{number of packets} \times \text{payload size}) + (\text{number of packet} \times \text{header size}))$, is $(7 \times 102) + (7 \times 25) = 714 + 175 = 889$ bytes. Similarly, the size of the last packet is $(1 \times 48) + (1 \times 25) = 48 + 25 = 73$ bytes. Thus, the total anomaly agent size after segmentation is 962 bytes. The size of the packet which carries the result of the in situ verification process is 27 bytes (i.e., 25 bytes header and 2 bytes payload), because this message just sends the result of the verification process in the form of either “1” or “0” representing “anomalous” or “normal” status, respectively. The data packet structure of the anomaly agent and the verification process is illustrated in Figure 8 to Figure 10.

Anomalous traffic: We randomly generated 25% anomalous traffic by anomalous sensor nodes in the simulation plane. The normal and the anomalous data traffics are collected and subsequently used for building the anomaly detection rules.

Agent optimization thresholds: The threshold values for $-2\sigma$, $-1\sigma$, $1\sigma$, and $2\sigma$ are based on six sigma rule to optimize anomaly agent transmission [25]. The values for the weighting factors are set as $\alpha_1$, $\alpha_2$, $\beta_1$, and $\beta_2$. The threshold for the anomaly agent transmission is fixed as $\psi$, whereas the tolerance zones are defined as $Tol_\psi$, $Tol_c$, and $Tol_c$.

Number of iterations: The results given in the next subsection are average of the 30 simulated experiments.
Following seeds are used to build the simulation scenarios: $W_d = 100$, $L_g = 100$, $k = 30 – 150$, $a = 50$, $b = 50$, $E_{T_X} = 50 \times 10^{-9}$ Joules, $E_{R_X} = 50 \times 10^{-9}$ Joules, $e_{f_x} = 10 \times 10^{-12}$ Joules/bit/m$^2$, $e_{mp} = 1.3 \times 10^{-3} \times 10^{-12}$ Joules/bit/m$^4$, $E_{DA} = 5 \times 10^{-9}$ Joules/bit/signal, $M_p = 128$ kb, $M_{SRAM} = 4$ kb, $M_{flash} = 512$ kb, $N_E = 10^4$ nJ, $t = 0.1$, $Pkt = 31$ bytes, $N_{E_{alg}} = 1 n/J$, $\epsilon = 0.68$, $\lambda = 0.95$, $\alpha = 12$° C, $T = 0.1$, 2.3, 4.5 seconds, $\varphi = 1$ for entitled action performed and 0 for non-entitled action performed, $v = 100\%$ (full battery) $\rightarrow 0\%$ (empty battery), $F = 1$ for each timely received and 0 for every delayed received packet, $h = 10$, $\alpha_1 = 0.42$, $\alpha_2 = 0.58$, $\beta_1 = 0.25$, $\beta_2 = 0.75$, $\psi = 0.55$, $\eta = 0.50$, $\gamma = 0.45$, and $\zeta = 0.40$.

1) Results and Analysis: We measured the comprehensive performance of the proposed anomaly detection and verification system in terms of following performance statistics:

- **Anomaly detection rate:** This statistic gives the percentage of the anomalies detected from the total number of anomalies.
- **Energy dissipation estimation:** The energy dissipation is estimated for the network traffic which is comprises of both anomalous and normal data packets (observations). This allows to estimate the impact of using mobile agents for in situ verification on the energy budget of resource constrained WSN.
- **Number of anomaly agents transmitted:** This statistic facilitates measuring the numbers of anomaly agents transmitted with and without employing anomaly agent transmission optimization algorithm.

We performed the first set of experiments to estimate the detection rate of the first-order anomalies which are caused by the in situ faults (refer to the first first-order join $N(\varphi, T)$, as discussed in Section IV-B). In this set of experiments, cluster member nodes continuously transmitted sensed $F_{el_{set}(1)}$ values instead of periodic transmissions to the respective cluster heads. We generated network traffic based on 5000 data packets, which were transmitted by 30 cluster member nodes to the corresponding cluster heads. In this case, the anomaly detection module detected 99% of the anomalies. For the network traffic comprised of 7000, 9000, 11000, and 13000 observations transmitted by 60, 90, 120, and 150 cluster member nodes, the anomaly detection rates were 98.80%, 98.40%, 98.20%, and 98%, respectively. The experiments results show that the detection rate of the first-order anomalies caused by the in situ faults was in the range of 98% to 99%.

The second and third first-order anomalies are related to the resource consumption by the cluster member nodes. We enforced a resource exhaustion attack, in which cluster member nodes transmitted low battery status instead of expected values of the residual battery status (refer to the second first-order join $N(T, \nu)$, as discussed in Section IV-B). The overall detection rate of the anomalies caused by the resource exhaustion attack varied between 98.60% and 99%. The detection rates for 5000, 7000, 9000, 11000, and 13000 observations were 99%, 98.80%, 98.80%, 98.60%, and 98.60%, respectively. Next, the anomaly detection modules on the cluster heads were enabled to detect the anomalies caused by the faulty nodes (refer to the third first-order join $N(\varphi, \nu)$, as discussed in Section IV-B). In these experiments, the energy budget of the faulty cluster member nodes dissipated quickly due to the unauthorized actions performed by them. As a consequence, the nodes transmitted low battery status instead of expected values to the corresponding cluster heads. In these experiments, the detection rates were 98.2%, 98.80%, 99%, 97.80%, and 98.80% for 5000, 7000, 9000, 11000, and 13000 observations, respectively.

Next, we experimented with the scenario of the faulty nodes (refer to the fourth first-order join $N(\varphi, T)$, as discussed in Section IV-B), in which faulty cluster member nodes transmitted anomalous values of the entitled actions with respect to the duration of time. In these cases, the anomaly detection modules, installed on each cluster head, detected 98.2%, 98.80%, 99%, 97.80%, and 98.80% of first-order anomalies for 5000, 7000, 9000, 11000, and 13000 numbers of observations, respectively. Finally, we induced the denial of sleep attack scenarios (refer to the fifth first-order join $N(F, T)$, as discussed in Section IV-B). In these scenarios, cluster member nodes continuously transmitted the sensed observations instead of periodic transmissions. In this set of experiments, the detection rates for 5000, 7000, 9000, 11000, and 13000 observations were 98%, 98.60%, 99%, 97.80%, and 98.80%, respectively. In all of the above cases, anomaly agents were capable of identifying the source of anomalies that have occurred in situ or in transit. The graph shown in Figure 11 summarizes the anomaly detection results. It can be observed that the overall detection accuracy is high for all first-order joins as it varies between 97.80% to 99.20%.

We now analyze the energy dissipation cost of both situations, when normal and anomalous data packets are transmitted in varying node density scenarios. The normal data traffic comprises of data packets that are transmitted from cluster member sensor nodes to their corresponding cluster heads. On the other hand, the anomalous data
traffic includes data packets of the anomaly agents that are transmitted from the cluster heads to the suspicious member nodes for in situ verification and results of in situ verification process that are transmitted back to the cluster heads. It is evident from Figure 12 that the line which denotes the energy dissipation by transmission of 5000, 7000, 9000, 11000, and 13000, normal data packets that are transmitted by 30, 60, 90, 120, and 150 cluster member nodes to their respective cluster heads has a steady growth in the positive direction along the x-axis. This shows the gradual increase in the energy dissipation as numbers of packets increase in the network traffic. On the contrary, the energy dissipation by the anomalous network traffic that is based on the anomaly agents and in situ verification data packets tend to variate and has a relatively non-steady growth. This is due to the variations in the detection accuracy of the corresponding cluster heads and also randomness involved in the generation of underlying anomalous traffic.

Next, we analyze the packet reception cost in terms of energy dissipation. Figure 13 depicts the energy dissipation results for the reception of both normal and anomalous data packets. The energy dissipation caused by receiving data packets follows the same trend as shown by data packets transmission with a slightly lower cost. The data traffic based on 5000 to 7000 data packets caused the dissipation of 0.0686 to 0.177 J energy for transmission and 0.0682 to 0.176 J energy for receiving normal traffic. Similarly, for transmitting and receiving anomalous traffic, the network dissipated 0.517 to 1.592 J energy and 0.51215 to 1.591 J energy, respectively. The relatively low energy dissipation while receiving data traffic is due to the fact that the transmission operation also involves the distance factor in addition to the fixed amount of energy consumed by the transceiver circuitry for data communication. Another key fact which is evident from Figure 12 and 13 is the difference in the magnitude of the energy dissipation by normal data traffic with that of anomalous data traffic. This difference of energy dissipation results for the reception of both normal and anomalous data traffic down to 42%-52%, as shown in Figure 14. This reduced the energy dissipation by both transmission and reception of anomalous data traffic over the network down to 42%-52%, as shown in Figure 15 and 16, respectively.

The simulation results and analysis show that the proposed anomaly detection and verification system is not only capable of detecting a range of first-order anomalies at high detection rate, but also capable of successfully performing in situ verification of suspicious nodes. Furthermore, the anomaly agent transmission optimization method, which considers both the current and historical
observations to optimize the agent transmission, can prolong the network life time by as much as around 50%.

B. Implementation

In order to measure the impact of the proposed scheme on the low resource real sensor nodes, we implemented algorithms on the MICAz mote, running TinyOS 2.1.1 [8]. The MICAz mote consists of a microprocessor (Atmega128L) [18]. The program flash memory, the configuration EEPROM, and the flash data logger memories of MICAz sensor nodes have 128, 4, and 512 kb storage spaces, respectively. The proposed algorithms are programmed using network embedded systems C (nesC) [23], a component-based event-driven programming language, used for building applications in the TinyOS. The objective of the implementation is an estimation of memory and energy dissipation along with processing time taken by the proposed algorithms. Note that we executed the worst case scenarios (i.e., all conditions were executed) to estimate the impact of the proposed algorithms on the resources of MICAz mote.

1) Memory consumption: The programming language nesC generates a memory consumption report of programs during their compilation stage. This report indicates the status of ROM, RAM, .data, .bss, and .text segments of the MICAz mote’s memory in terms of bytes. The total memory consumption (i.e., RAM and ROM together) by the proposed Algorithms 1, 2, 3, and 4 are 1567, 2559, 3130, and 1206 bytes, respectively. This result indicates that the memory consumption of Algorithms 2, and 3, is higher than that of the Algorithms 1 and 4 due to the fact that these Algorithms involve processing and transmission of an anomaly agent. The proposed Algorithms 2 and 4 are executed on resource rich cluster head nodes. Therefore, high memory consumption can easily be managed by these nodes. The executions of Algorithms 1 and 3 on the non-cluster head sensor nodes consume 1567 and 3130 bytes of memory, respectively, which can be easily managed by the memory subsystem of the MICAz sensor node family. Furthermore, the execution of Algorithm 3 is not very frequent. It executes only when a suspicious sensor node is found in a cluster and the cluster head triggers the anomaly agent for the in situ verification of the suspicious sensor node after due consideration of the agent transmission optimization method. It should be noted that the anomaly agent occupies the memory space only for a short period of time. As such, the anomaly agent is killed and clears the occupied memory space as soon as it transmits the in situ verification result to the cluster head. Therefore, the impact of the anomaly agent on the cluster member node’s memory is rather short term.

2) Processing time: We now examine the processing time taken by the proposed algorithms, as this is an important performance metric for the anomaly detection application. The processing times for the Algorithms 1, 2, 3, and 4 are 6.93, 37.28, 7.65, and 4.13 ms, respectively. Algorithms 2 and 3 perform the jobs of the anomaly detection and in situ verification processes and their combined elapsed time is 44.93 ms. Algorithms 2 and 3 consume high processing time as compared to Algorithms 1 and 4 due to the involvement of the processing of an anomaly agent. This outcome of the processing time is consistent with the theoretical results reported in Section IV-D. Furthermore, if the cluster head also executes the agent transmission optimization process, then the whole procedure takes 49.06 ms time, which is quite efficient.

3) Energy consumption: The process of features collection by the sensor node consumes 55.60 J of energy in each epoch. Algorithms 2 and 4 consume 4039.91 J and 40.02 J of energy, respectively. These algorithms are executed on the resource rich cluster head nodes which can easily manage this amount of energy consumption. The energy consumption by Algorithm 3 is 1570.96 J. However, this energy is only consumed by the suspicious cluster member node for the in situ verification process. The summary of processing time, memory, and energy consumption by the proposed algorithms is given in Table V.

C. Comparative study and discussion

The proposed scheme is compared with three recent related schemes proposed by Ketel [4], Eludiora et al. [6], and Khanum et al. [7]. The comparison is based on the following five aspects: (a) role of mobile agent, (b) nature
of anomalies, (c) number of agents per node, (d) detection time complexity, and (e) agent transmission optimization. In the proposed scheme, the nature of the agent is mobile and only one agent per node is triggered for the in situ verification of the suspicious behavior of the sensor node, using its resources. On the other hand, Ketel’s scheme employs three agents for the process of the anomaly detection, namely, static, mobile, and nodal agents [4]. Furthermore, the scheme presented by Eludiora et al. uses a mobile agent for the inter-base station control communication [6]. Also, the proposal by Khanum et al. uses two static agents: coordination and management, and a mobile agent to carry out the anomaly detection process [7]. Both Ketel and Khanum et al. employed three agents for the process of anomaly detection [4], [7]. The use of several agents not only increases the computational cost, it also requires additional computation for inter-mobile agent communication. The use of the multiple agents also increases the communication and processing load of the network.

Another important difference between the proposed and other related schemes is the transmission of the mobile agent by the particular type of the node. In our proposed scheme, the mobile agent is only triggered by the resource rich nodes (i.e., by the cluster heads) and received by the cluster member nodes as compared to other schemes, in which agents are transmitted by all sensor nodes. This approach of existing schemes can quickly dissipate the energy resources of the cluster member nodes. On the other hand, in the proposed scheme, the cluster member nodes receive the mobile agent only for the in situ verification process (i.e., not very often). This approach puts the least burden on the resource limited nodes. Furthermore, our scheme does not support the inter-cluster member nodes mobile agent movement (i.e., between non-cluster head nodes within or outside the cluster). This strategy effectively uses overall WSN resources without negating the role of the mobile agent in the proposed anomaly detection system.

The nature of the anomalies detected in the work presented by Ketel and Eludiora et al. are node and DoS attack based anomalies, respectively [4], [6]. The scheme presented by Khanum et al. only detects sensor reading anomalies [7]. On the other hand, our proposed scheme has the ability to detect several types of first-order anomalies which are caused by attacks such as denial of sleep attack, battery exhaustion attack, and so forth. The detection complexity of the scheme presented by Ketel cannot be estimated, as it is an architectural level scheme and no actual details of the anomaly detection process are provided [4]. The anomaly detection complexity of Eludiora’s et al. scheme is $O(n^2)$ [6]. Both, Khanum et al. [7] and the proposed scheme detects anomalies in $O(n)$ time.

Furthermore, none of the existing schemes has considered the optimization of mobile agent transmission [4], [6]-[7]. The proposed scheme optimizes mobile agent transmission by considering historical and current instances of the anomalous observations. This comparative study shows that the proposed scheme is efficient and effective in various aspects as compared to the several existing schemes. A summary of the comparative study is stated in Table VI.

### VI. Conclusions

This study has proposed a mobile agent-based anomaly detection and in situ verification system for cluster-based WSNs. In the proposed system, the mobile agent uses the resources of the victim node to verify if the node is compromised. This approach offers an additional service to the network administrator to confirm the source of the anomaly, before initiating appropriate action against the anomalous node. The proposed system is designed in a way that the majority of the processing is performed by the resource rich cluster head nodes. Furthermore, the itinerary of the mobile agent has only a single node, that
is, from cluster head to cluster member nodes. This design consideration saves the resources of the cluster member nodes, which are consumed in the case of multi-node itinerary. This study has also successfully employed the CRM mechanism to detect different types of first-order anomalies caused by in situ faults, denial of sleep attacks, and resource exhaustion attacks. Further, we have proposed a mobile agent transmission optimization method which prolongs the network lifetime. The results obtained through the detailed performance evaluation advocate the efficiency and effectiveness of the proposed system for resource constrained wireless sensor networks.

In future work, we intend to extend our investigation to examine the effectiveness of the proposed system in large-scale multi-hop sensor networks against complex nature of anomalies.

ACKNOWLEDGMENT

The authors are thankful to the anonymous reviewers for their constructive feedback and suggestions to improve the presentation of this paper.

REFERENCES

Muhammad Usman received the M. Science (M. Phil) in Computer Science from the PMAS-AAUR Pakistan. He is currently associated with the School of Information and Communication Technology, Griffith University, Gold Coast, Australia, where he is working towards his PhD degree. Prior to joining the Griffith University, he served as a Lecturer at the King Khalid University, Saudi Arabia, and a Project Manager and Lecturer at the University of Central Punjab, Pakistan. He also worked as a System Administrator and Lecturer at the Pakistan Air Force Fazaia College, Pakistan. His current research interests include investigation of research issues in mobile agent-enabled distributed systems, formal modeling and verification of communication protocols, anomaly detection, and design and analysis of cross layer protocols.

Vallipuram Muthukkumarasamy obtained the BScEng with 1st Class Hons from the University of Peradeniya, Sri Lanka and obtained the PhD from the Cambridge University, England. He is currently attached to the School of Information and Communications Technology, Griffith University, Australia as a Senior Lecturer. His current research areas include investigation of security issues in wireless networks, sensor networks, trust management in MANETs, key establishment protocols and medical sensor networks. He is currently leading the Network Security research Group at the Institute for Integrated and Intelligent Systems at the Griffith University. He has also received a number of best teacher awards.

Xin-Wen Wu received the Ph.D. degree from the Chinese Academy of Sciences. He was with the Chinese Academy of Sciences, the University of California, San Diego (as a post-doctoral researcher), and the University of Melbourne (as a research fellow). He was affiliated with the School of Information Technology and Mathematical Science, University of Ballarat, Australia. In April 2010, he joined the Griffith University, Australia, as a faculty member of the School of Information and Communication Technology; and he is currently a Senior Lecturer. His research interests include cyber and data security, coding techniques, and information theory and its applications.