Pers Ubiquit Comput
DOI 10.1007/s00779-012-0529-y

ORIGINAL ARTICLE

A novel intrusion detection framework for wireless sensor networks

Ashfaq Hussain Farooqi · Farrukh Aslam Khan · Jin Wang · Sungyoung Lee

Received: 20 September 2011 / Accepted: 2 February 2012

Abstract Vehicle cloud is a new idea that uses the benefits of wireless sensor networks (WSNs) and the concept of cloud computing to provide better services to the community. It is important to secure a sensor network to achieve better performance of the vehicle cloud. Wireless sensor networks are a soft target for intruders or adversaries to launch lethal attacks in its present configuration. In this paper, a novel intrusion detection framework is proposed for securing wireless sensor networks from routing attacks. The proposed system works in a distributed environment to detect intrusions by collaborating with the neighboring nodes. It works in two modes: online prevention allows safeguarding from those abnormal nodes that are already declared as malicious while offline detection finds those nodes that are being compromised by an adversary during the next epoch of time. Simulation results show that the proposed specification-based detection scheme performs extremely well and achieves high intrusion detection rate and low false positive rate.

A. H. Farooqi
Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Pakistan
e-mail: ashfaq.farooqi@nu.edu.pk

F. A. Khan (✉)
Center of Excellence in Information Assurance (CoEIA), King Saud University, Riyadh, Saudi Arabia
e-mail: fakhan@ksu.edu.sa

J. Wang · S. Lee
Department of Computer Engineering, Kyung Hee University, Yongin, South Korea
e-mail: wangjin@oslab.khu.ac.kr

S. Lee
e-mail: sylee@oslab.khu.ac.kr

Keywords Vehicle cloud · Wireless sensor network · Network security · Intrusion detection framework (IDF) · Threat prevention · Offline attack detection

1 Introduction

Cars are one of the most essential parts of our daily life. People want to minimize their traveling time and like to have something that provides them safety as well as entertainment. *Vehicle cloud* (VCloud) [8] is a novel idea that uses the benefits of wireless sensor networks (WSNs) and the concept of cloud computing to provide better services to the community. Sensor nodes are used to guide about the environmental condition inside the car; to detect the behavior of the driver whether he/she is driving normally, and to take care of the physical condition of the driver. VCloud model differs from other research works [1–6] as it combines different networking paradigms such as mobile ad hoc networks, wireless sensor networks, vehicular ad hoc networks, and cloud computing and provides new trends in telecommunication, transportation and healthcare systems. The model provides a framework for the future *intelligent transportation system* (ITS) that assures safety of the vehicles from accidents on the road, determines the condition of vehicle or driver using sensors and supplies better assistance in case of abnormality, provides possible healthcare services to the passengers during traveling, discovers shortest reliable routes to the destination, and provides entertainment.

Security is among the major challenges for all kinds of networking paradigms whether they are wired, wireless or newly emerging network models [7]. The flow of data in VCloud is shown in Fig. 1. VCloud can achieve better performance and increase reliability once it secures its

Published online: 06 June 2012

wireless sensor networks [8]. Wireless networks use wireless medium and provide ad hoc access to resources. Hence, these networks become more vulnerable to threats than wired networks [9]. Wireless sensor networks are usually composed of small sized sensor nodes. These nodes can be homogeneous or heterogeneous [10] and have less computation capacity and small memory. These nodes are deployed or installed in an area called sensor field to sense the surroundings. These nodes work in an infrastructure-less and dynamically changing environment [11] and route the collected data to the base station (BS) or sink node for further interpretation. Sensor nodes are self-controlled and an easy target for attacks from adversaries.

Wood et al. [12] discuss different attacks that cause denial of service (DoS) at various layers of sensor nodes and also provide countermeasures against each threat. In another work, Karlof et al. [13] focus on routing protocol attacks, such as homing, selective forwarding, black-hole, sink-hole attacks and others. According to them, these attacks degrade the performance of sensor networks. Roosta et al. [14] present a detailed survey of security threats on wireless sensor network. They also discuss countermeasures for some of them to cater these attacks efficiently. In [15], Bojkovic et al. analyze the working of wireless sensor networks under several attacks and discuss the role of key distribution protocols in threat scenario.

Intrusion detection system (IDS) is considered as a front-line solution for inside attacks that audits the working of nodes and determines those entities that are performing maliciously [16–18]. In WSNs, the unit that performs this activity is called *IDS Agent*. IDS agent is installed in all sensor nodes or monitor nodes or only at base station (BS) [19]. It collects network data from sensor nodes and applies detection policy to determine abnormal or compromised node(s). Rajasegarar et al. [20] conducted a survey on anomaly detection mechanisms in wireless sensor networks. Xie et al. [21] also conducted a survey on anomaly detection schemes. It provides detailed studies of various approaches that are used for detecting intrusions in sensor networks. Farooqi and Khan [22] discussed various lethal attacks that degrade the overall performance of a sensor network and also explained different intrusion-based detection schemes.

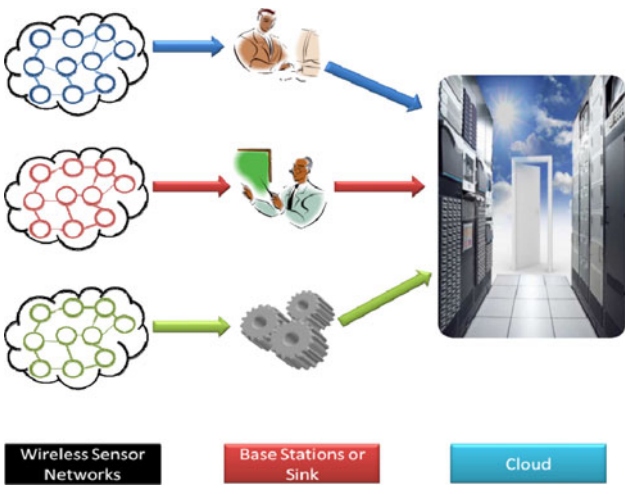

Fig. 1 Transmission of data between different entities of VCloud

Mostly, security is not considered during the designing of the routing protocols [23]. Therefore, most of the routing protocols are vulnerable to security threats. Krontiris et al. [24] presented a model to detect sink-hole attack for MintRoute routing protocol, Loo et al. [25] formulated different rules to detect various routing attacks for AODV routing protocol, and Su et al. [26] presented a security scheme for LEACH protocol. However, these approaches are not applicable to secure other routing protocols. Hence, there is a requirement of a security model that can be added to insecure routing protocols to make them resilient against routing attacks. In this paper, we present an intrusion detection methodology that can be added to such routing protocols to make them more secure. We propose a *novel intrusion detection framework* (IDF) that works in a distributed environment using a specification-based detection policy to detect intrusions by collaborating with neighboring nodes. It works in two modes: *Online prevention* allows safeguarding from those abnormal nodes that are already declared while *offline detection* finds those nodes that are being compromised by an adversary during the next epoch of time. In this framework, we propose a security model that suits distributed detection policies. It should be installed in every sensor node. The proposed security framework differs from other works in the sense that it includes the IDS agent installation; the way intrusions are detected; generation of claim about the malicious nodes having maliciousness level greater than certain threshold using cognitive decision making; collaboration with neighboring nodes to make final decision about the claimed nodes, and finally the consolation.

We also formulate a *specification-based detection scheme* for a flat wireless sensor network scenario and test it using a simulator that is implemented in C#. Results show that it achieves high detection rate and receives low false positive rate. The results also show that a centralized distributed approach cannot figure out the actual condition of the network properly. Therefore, a purely distributed security system is more appropriate for wireless sensor networks.

The rest of the paper is organized as follows: related work is discussed in Sect. 2 while Sect. 3 contains the description about proposed intrusion detection framework. Section 4 presents the experiments and discussion on the results. Section 5 concludes the paper.

2 Related work

Sensor nodes have a simple architecture. These nodes work in dynamic environment. Hence, they need a methodology that works in distributed way to safeguard them from various threats. We discussed in our previous work [22] the various lethal attacks that degrade the overall performance of a sensor network. We also briefly explained different intrusion-based detection schemes for wireless sensor networks. Table 1 depicts various IDS-based security methodologies that are presented for wireless sensor networks found in the literature.

Roman et al. [27] present an IDS agent architecture for sensor nodes, which is composed of two elements: data structures and detection entities. According to them, sensor nodes should contain two types of information: (1) knowledge about the security and (2) knowledge about the environment; and two detection entities (1) *Local agents* are responsible to analyze the behavior of sensor nodes by auditing against the local resources (2) *Global agents* work like a *spontaneous watchdog*. The proposed scheme just focused on the working of these entities while did not provide any detail about the detection of various attacks and the way appropriate action may be taken.

A purely distributed detection system is presented by Krontiris et al. [28] that cooperates with neighboring node to take decision about the maliciousness of the sensor nodes. They improve the initial security framework and formulate a more encouraging IDS agent architecture called *lightweight intrusion detection architecture* (LIDeA) [24]. They present some rules to detect sink-hole attack and further propose an encryption mechanism to secure network from outside attacks. Their work focuses on *MintRoute* routing protocol, and the proposed approach cannot be applied to other routing protocols such as *LEACH protocol* etc.

A well-known specification-based distributed–centralized security mechanism is discussed by Silva et al. [36]. It is a simplest architecture designed for monitor node (A node in which IDS agent is installed). It works in three phases: (1) *Data acquisition* filters the packets and collects information that is required for next phase; (2) *Rule application* applies predefined rules on the acquired information; (3) Lastly, alert is generated by intrusion detection if the failure counter is more than the expected value. Stetsko et al. [29] presented a *neighbor-based detection scheme* for securing sensor networks by analyzing the behavior of neighboring node with itself. They discussed the way a node detects the neighboring node if it performs abnormally to the set parameters during normal condition. A distributed–centralized detection technique is discussed by Phuong et al. [38]. It detects three types of attacks by an anomaly detection algorithm called *Cumulative Summation* (CUSUM). These are as follows: (1) compromised node attracts the attention of other nodes and (2) affects the data of the messages, and (3) compromised node floods packets to exhaust resources of other nodes. CUSUM algorithm is not simulated or tested, so it is difficult to analyze the effectiveness of this algorithm. The proposed approaches do not provide any idea about the decision making whether it will be done by individual sensor node or by collaboration with the neighboring nodes. Further, they did not guide about the way network will be secured once the intrusion is detected.

Table 1 Previous security schemes

Proposed approach	IDS agent installation	Detection policy	Attacks
Spontaneous Watchdog [27]	Purely distributed	Any	–
Cooperative local auditing [28]	Purely distributed	Specification based	Routing
LIDeA [24]	Purely distributed	Specification based	Routing
Fixed-width clustering [25]	Purely distributed	Anomaly based	Routing
Neighbor-based intrusion detection [29]	Purely distributed	Specification based	Routing
Artificial immune system [30]	Purely distributed	Anomaly based	MAC/routing
Intrusion aware validation algorithm [31]	Purely distributed	Anomaly based	–
Pair-based approach [32]	Purely distributed	Misuse and anomaly based	–
Group-based detection scheme [33]	Purely distributed	Anomaly based	Routing
ANDES algorithm [34]	Purely centralized	Anomaly based	Phy./routing
Application-independent framework [35]	Purely centralized	Anomaly based	–
Decentralized intrusion detection model [36]	Distributed–centralized	Specification based	Trans./routing
Hybrid intrusion detection system [37]	Distributed–centralized	Misuse and anomaly based	Routing
Cumulative summation [38]	Distributed–centralized	Anomaly based	Trans./routing
Hierarchical intrusion detection model [39]	Distributed–centralized	Anomaly based	Routing

Loo et al. [25] propose a *fixed-width clustering approach* for AODV routing protocol. Drozda et al. [30] present an *artificial immune system* (AIS) for wireless sensor networks. These are anomaly detection schemes; hence, they are expensive as considered to wireless sensor networks. They also do not provide the detail about the collaboration with the neighboring nodes for making decision and further lack the way network should be secured after detection of the attacks.

Shaikh et al. [31] discuss the limitation of purely distributed detection schemes with respect to individualized decision making. According to them, sensor nodes should collaborate with the neighbors to make final decision and proposed a mechanism for them called *intrusion aware validation algorithm*. The proposed scheme is just an enhancement to the purely distributed schemes to make them more effective.

Ahmed et al. [32] propose a *pair-based abnormal node detection* while Li et al. [33] propose a *group-based detection scheme*. These both schemes require additional burden on the application layer for the formation of pairs and groups. Hence, they are expensive with respect to computation and communication.

Gupta et al. [34] present an anomaly detection scheme called *ANDES*. It works in two phases: collection of information from the sensor nodes and detection of abnormal nodes. Zhang et al. [35] propose an *application-independent framework* for identifying the source of information, whether it is reliable one or compromised. These are purely centralized detection schemes as base station is responsible for the detection of intrusions. Purely centralized approaches are energy efficient but they add additional burden on the base station. They require additional routing protocol to perform their activities, and in many cases, these approaches cannot be able to judge the system perfectly.

3 Intrusion detection framework

Wireless sensor networks are used in various applications to make decision about the area in consideration. These applications range from military to healthcare. Vehicle cloud is among those applications that use the benefits of wireless sensor networks and the concept of cloud computing to provide better services to the community. It is important to secure sensor network to achieve better performance of the vehicle cloud. Hence, it is eminent to have a system that secures these networks from adversaries. Researchers often do not focus on security aspects while designing a new routing protocol [23]. Their prime target is to come up with an energy efficient routing scheme that consumes low energy. Therefore, such routing protocols are vulnerable to security threats. Hence, they require a security framework that makes these protocols resilient against routing attacks. We propose a novel intrusion detection scheme that can provide a solution for securing these approaches.

Our proposed intrusion detection framework (IDF) works in a distributed environment as it is a purely distributed detection system. Figure 2 illustrates the key modules of our approach. It works in two modes: online prevention and offline detection. Online prevention secures the sensor network from those nodes that are already declared as malicious while offline detection applies intrusion detection scheme to find those nodes that are not working properly or that are being compromised by the adversary after installation.

IDF works in a promiscuous mode. It listens to every kind of traffic and after that it takes decision whether to process it or send it to next hop (act like a router). Whenever a node senses any message, it is collected by two modules: local auditing and data collection. *Local auditing* module verifies whether it is destined to it and comes from the legitimate neighbor. If its status is clear then the sensor node processes that message and performs normal task. In the mean time, *Data collection* unit forwards the received packets to *content suppression* unit. This unit interprets the header to acquire required information. Once the data are being processed, *intrusion detection* policy is applied. The result of this unit is transmitted for *cognitive decision making*. If the failure level is above certain expected value, an alert is generated. After communication with neighboring IDS agents, it is finally declared as abnormal node or normal node. If it is declared malicious, an action is taken against it.

3.1 Online prevention

Whenever a node senses any message, *online prevention* validates the packet whether it is coming from legitimate neighboring node or not. If it is received from the normal node, the sensor node performs normal task otherwise it discards it immediately. The general flow is depicted in Fig. 3.

Following is the description of different elements that play a part in online prevention as shown in Fig. 2.

3.1.1 Data repository

Sensor nodes do not contain data about the already declared malicious nodes [27]. There should be a container that holds this information. Here, it is called *data repository*. It has two lists: one regarding neighboring nodes and other contains a list of malicious nodes.

Sensor nodes have a small memory. Data repository module should not take too much space. Sensor nodes

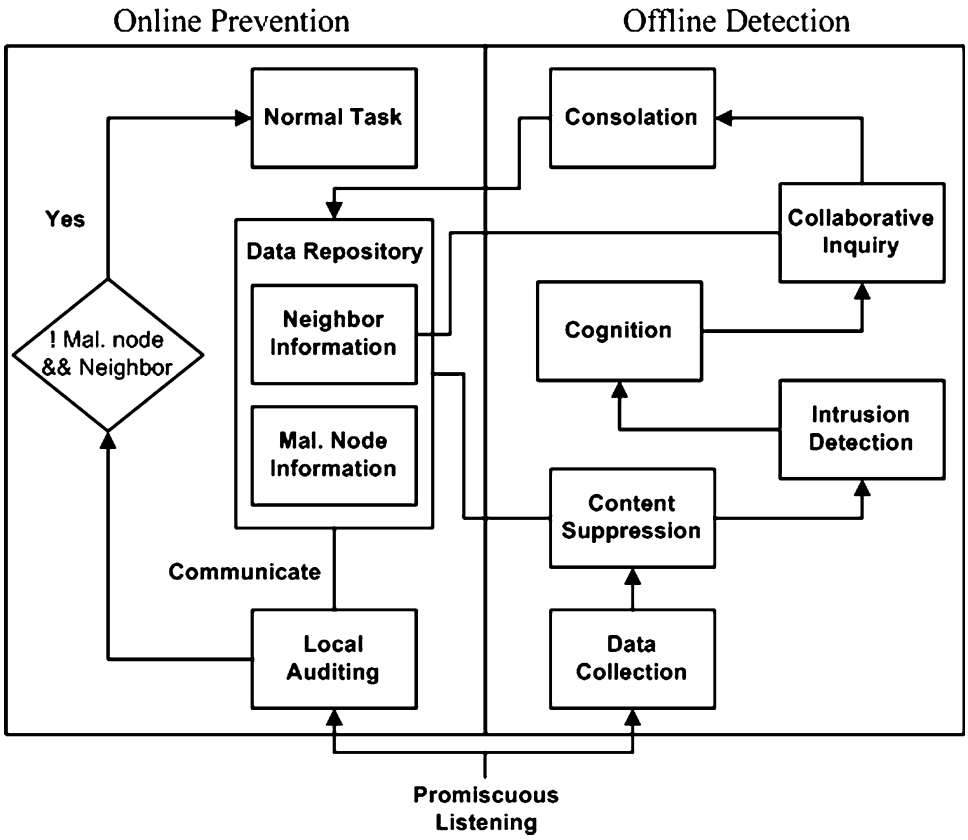

Fig. 2 Proposed intrusion detection framework

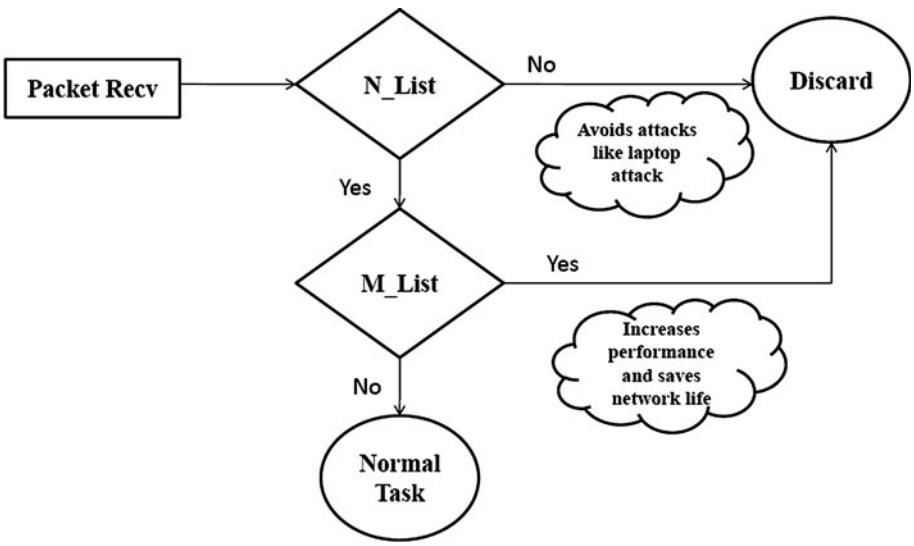

Fig. 3 Online prevention

join and leave the network. The *neighborhood list* (N_List) should update on each instance. The format of N_List is shown in Table 2. It contains three fields: neighbor node ID, time stamp of the last received packet from that node and the status whether that node is normal.

Second list called the *malicious nodes list* (M_List) holds information about those nodes that are other than the neighboring nodes but are declared abnormal nodes. The format of M_List is shown in Table 3. It has two fields: node ID and its maliciousness level (see Sect. 3.2.4).

Node_ID is indexed by taking the hash of actual ID and other fields are populated respectively. Let a fixed size array data structure is used for N_List, then a suitable hash function helps to place the values and retrieve too. It is expensive with respect to memory but efficient with respect to computation. In best case, the computation time complexity is $O(1)$. The worst case time complexity deals with

Table 2 Neighborhood list (N_List)

Node_ID	Time stamp	Status (normal or mal.)
C	C_New	N or M
I	I_New	N or M

Table 3 Malicious node list (M_List)

Node_ID	Status
D	Mal_Level

the way two similar hashing outputs are handled, i.e., open chaining, etc.

3.1.2 Local auditing

The *local auditing* unit verifies and validates the incoming packets as mentioned in Algorithm 1. It consults with data repository module and takes decision whether to discard it or forward it for further processing. Here "processing" means to perform normal activities. It purely depends on the configuration and application of the sensor node. The systematic working of this module is discussed in Algorithm 1.

Algorithm 1: Local Auditing at node J

Input: Packets from other nodes
Output: Validation of packet (discard or accepted)
Begin
  If received packet is destined for node J
    If it is from neighbor (Tally N_List)
      If neighbor is not malicious
        accept it for further processing
      Else
        discard it
      End If
    Else
      discard it
    End If
  Else
    discard it
  End If
End

It works in promiscuous listening mode as discussed earlier. It listens to all the communications that takes place in the radio range of that node. Whenever it receives any packet, it makes decision whether to process it or drop it. Firstly, the packet should be destined to it. After that, it checks the resident of arrival. This should be the address or ID of any neighbor node (N_List). It is discarded if it does not belong to the neighborhood. Here, this approach takes care of laptop-class attack as well. Let the received packet is from one of the neighboring node, then the status of that neighbor is checked. The received message is processed further if it is from the normal node.

This unit consults regularly with the data repository. Hence, the performance of this unit is directly associated with the implementation of data repository. If it is array based, then in best case, the time complexity while tallying is $O(1)$, and if it is implemented using linked list, then it would be $O(n)$ where n is the length of the list.

3.2 Offline detection

Offline detection finds those nodes that are being compromised by an adversary after the installation of the sensor network. It is composed of various elements as shown in Fig. 2. It works in a promiscuous mode. Data collection unit listens to every kind of traffic and forwards the received packets to content suppression unit. This unit interprets the header to acquire required information. Once the data are being processed, detection policy is applied. The result of this unit is transmitted for cognitive decision making. If the failure level is above certain expected value, an alert is generated. After communication with neighboring IDS agents, it is finally declared as abnormal node or normal node. If it is declared malicious, an action is taken against it by consolation unit.

Consider a *flat wireless sensor network* of 24 nodes (A–X). They communicate with the base station using some routing protocol. We have made few assumptions about the sensor network that help to understand the proposed mechanism. First, sensor network is a static one. Second, sensor nodes cannot join after some time interval called *initialization phase*. It is the time period in which nodes make a topology after communicating with their neighbors to find a route to the sink or BS. Here, the initial routes of the sensor nodes to the BS are shown in Fig. 4. Sensor nodes send data messages after some random time interval. Lastly, nodes should initiate *route discovery* after some specific time interval. It must be equal or greater than the time required for IDF to make some decision.

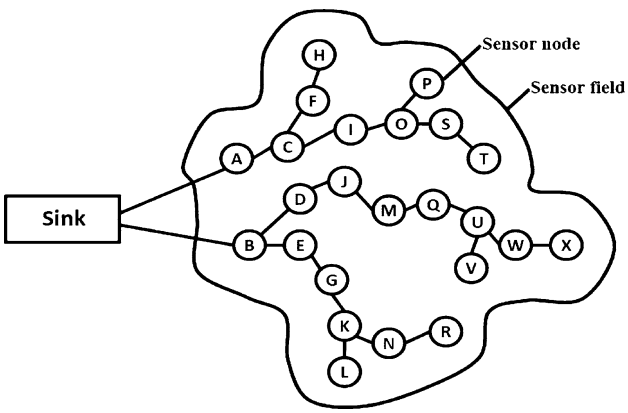

Fig. 4 A network of 24 sensor nodes that are communicating with base station

We explain the working of different elements of offline detection through the network topology shown in Fig. 4. Consider node J as an example node having seven neighbors (C, D, E, F, I, O and M).

3.2.1 Data collection

Sensor nodes usually listen promiscuously to the communication between neighboring nodes that reside in its radio range. In our proposed framework, data collection unit simply listens to these packets and transmits them to data processing unit. It does not store these packets. It is just like a channel between outside world and inner detection body.

3.2.2 Content suppression

Whenever a packet is received from data collection unit, its header is interpreted to analyze the actual transaction and values are updated in *audit data list* (A_List). It is a list that holds useful data that is utilized by intrusion detection unit to get maliciousness level of the surrounding nodes. The format of this list for the above-mentioned example scenario is shown in Table 4.

Let node J senses a packet. It interprets the header and gets that it is sent by node I to node C. So node J increments its A_List against sent and received fields of node I and node C, respectively. Consider that node C does not forward that packet further for some time "*t*," then node I retransmits the same packet again. Hence, two values change in A_List of node J: one for node I; the retransmit field, and one for node C; the received field. Now let node C forward that packet which it received from node I, then A_List updates only one field of node C i.e., the forward field.

The implementation of A_List is similar to that of N_List because it is updated for each instance of the surroundings. The length of A_List depends on the number of nodes from which the particular node is listening messages. Hence, we can make some assumption about the length of A_List if we know the density of the network. It is clear from the above discussion of A_List that no packet is stored but some fields of every packet are checked and then the packet is discarded.

The above process continues till some time epoch. After this, A_List is cleared by removing the entries of the already declared as malicious by tallying with M_List. The final A_List is sent to intrusion detection unit. Once it is communicated, A_List is refreshed and content suppression starts again.

3.2.3 Intrusion detection

Specification-based detection schemes are considered more favorable [27, 28, 36] for wireless sensor networks because *misuse detection* approaches cannot cater with unknown attacks while *anomaly detection* techniques are computationally expensive. In this module, some specific rules are applied that are designed for a routing protocol to detect routing attacks by validating the data collected from content suppression unit. These rules are formulated according to routing protocols; for example, Krontiris et al. [24] discussed various rules to detect sink-hole attack for MintRoute routing protocol and Loo et al. [25] formulated different rules to detect various routing attacks for AODV routing protocol. These rules are designed after the analysis of the normal working of the network and the way network behaves after some specific attacks are launched.

We explained the launching affect of different routing attacks for the example of a flat wireless sensor network in our previous work [22]. We made rules to detect various routing attacks that are inspired from [29, 36–38] but it differs the way they are formulated for the presented network scenario. Here, the detection scheme sets thresholds after normal execution of the flat wireless sensor network. If sensor node's behavior violates these thresholds during next epoch of time, then a particular flag is set against the respective field in the *flag list* (F_List).

The structure (shown in Table 5) and implementation of flag list are similar to those of A_List but it contains some flags in respective field positions. These are given as follows:

- *N* (miN): If value is less than the minimum threshold value and shows any attack pattern.
- *X* (maX): If value is greater than the maximum threshold value and shows any attack pattern.
- *L* (normaL): If value is between N and X or less/greater than threshold value but does not show any attack pattern.

Table 4 Audit data list (A_List) at node J

Node_ID	Packet sent (A_snt)	Packet received (A_rec)	Packet forward (A_fwd)	Packet retransmit (A_rtm)
C	C1	C2	C3	C4
I	I1	I2	I3	I4
…	…	…	…	…

Table 5 Flag list (F_List)

Node_ID	Packet sent (F_snt)	Packet received (F_rec)	Packet forward (F_fwd)	Packet retransmit (F_rtm)
C	N | X |L	N | X |L	N | X |L	N | X |L
I	N | X |L	N | X |L	N | X |L	N | X |L
…	…	…	…	…

Threshold values may be set by using any specific algorithm or any stochastic process that includes some intelligence. As far as the present sensor network scenario is concerned, these values may be set by executing the sensor network normally. In other words, consider a simulator that runs normally and calculate these values accordingly. These values are stored in a *threshold list* (T_List) shown in Table 6.

There are two ways through which T_List can be maintained. Firstly, take the average of obtained values of all the nodes for each field. Secondly, simulate the sensor network for n number of times and then calculate thresholds for each node, by taking the averages of obtained values for each node. T_List contains single value for all the nodes in the first type of implementation while it has more than one in second one. The second case seems more realistic because it suits the dynamic nature of sensor network.

Algorithm 2 explains the detection policy. There are two inputs, A_List and T_List, for this algorithm. These lists are analyzed to populate F_List.

Algorithm 2: Detection Policy

Input: Audit Data List (A_List), Threshold List (T_List)
Output: Flag List (F_List)
Begin
Case I: *(Packet Sent)*
  If Node_ID.A_snt < Node_ID.N_Snt
    Node_ID.F_snt == N
  Else If Node_ID.A_snt > Node_ID.X_Snt
    Node_ID.F_snt == X
  Else
    Node_ID.F_snt == L
  End If
Case II: *(Packet Receive)*
  If Node_ID.A_rec < Node_ID.N_Rec
    Node_ID.F_rec == L
  Else If Node_ID.A_rec > Node_ID.X_Rec
    Node_ID.F_rec == X
  Else
    Node_ID.F_rec == L
  End If
Case III: *(Packet Forward)*
  If Node_ID.A_fwd < Node_ID.N_Fwd
    Node_ID.F_fwd == N
  Else If Node_ID.A_fwd > Node_ID.X_Fwd
    Node_ID.F_fwd == L
  Else
    Node_ID.F_fwd == L
  End If
Case IV: *(Packet Retransmit)*
  If Node_ID.A_rtm < Node_ID.N_Rtm
    Node_ID.F_rtm == L
  Else If Node_ID.A_rtm > Node_ID.X_Rtm
    Node_ID.F_rtm == X
  Else
    Node_ID.F_rtm == L
  End If
End

The values that are stored against each node ID in A_List are compared with relative field value of T_List to find whether it is less, equal or more than that value. Following is the explanation of the Algorithm 2.

A. *Case I (Sending rate analysis)*

- Less than miN: Node might be damaged or exhausted.
- More than maX: Flooding attack or any other routing attack that compromises the node to send many packets.

B. *Case II (Receiving rate analysis)*

- Less than miN: Not affected. It might be due to the other compromised node.
- More than maX: Transport or routing attack; collision, flooding, worm-hole, sink-hole, black-hole, selective forwarding attack.

C. *Case III (Forward rate analysis)*

- Less than miN: Routing attack (node compromised with homing, selective forwarding or black-hole attack).
- More than maX: Not affected. It might be due to the other compromised node.

D. *Case IV (Retransmission rate analysis)*

- Less than miN: Not affected. It might be due to the other compromised node.
- More than maX: Collision attack.

We have implemented *a simulator* in Visual Studio.NET 2008 using C#. The basic purpose of developing a simulator is to make a test-bed that can be used to test the efficiency of a specification-based detection policy. Result shows that the proposed detection mechanism receives high *intrusion detection rate* and achieves *low false positive rate*.

3.2.4 Cognition

Cognition module is responsible for making decision about the behavior of the sensor nodes. Cognitive decision making starts once F_List is updated from A_List and T_List. We propose three postulates for this procedure:

1. If numbers of L are less than or equal to two, then its maliciousness level is considered *high (HIG)*.
2. If numbers of L are three, then its maliciousness level is considered *medium (MED)*.
3. Sensor node is a normal one if the node behavior does not follow any one of the above. Its maliciousness level is considered *low (LOW)*.

At the end of this phase, a list is populated that contains maliciousness information of each node called

Table 6 Threshold list (T_List)

Node_ID	N_Snt	X_Snt	N_Rec	X_Rec	N_Fwd	X_Fwd	N_Rtm	X_Rtm
C	CN_Snt	CX_Snt	CN_Rec	CX_Rec	CN_Fwd	CX_Fwd	CN_Rtm	CX_Rtm
I	IN_Snt	IX_Snt	IN_Rec	IX_Rec	IN_Fwd	IX_Fwd	IN_Rtm	IX_Rtm
…	…	…	…	…	…	…	…	…

maliciousness level list (ML_List). The format of this list is shown in Table 7.

Suppose node C violates many rules while node D violates few of them and node I does not violate any; hence, their maliciousness level is HIG, MED and LOW, respectively.

3.2.5 Collaborative inquiry

Ideally, sensor nodes should make decision on their own without collaborating with their neighborhood but this seems to be unrealistic. Because, they do not contain the whole picture of the network and they cannot detect the compromised node by individual analysis in most of the cases. Authors favor the collaboration of the node with its neighboring nodes in [24, 27, 40, 41]. According to our model, sensor node consults with the neighbors for those nodes only whose maliciousness level is MED.

Shaikh et al. [31] propose a consensus-based validation mechanism for distributed IDS methodology to incorporate cooperation in these approaches. It identifies compromised node(s) and takes care from already declared malicious node during decision making. In our model, the collaboration module is inspired from this work but it differs too. Their work is expensive due to the following:

- Sensor node requires neighbor information of each malicious node.
- It finds common neighbors of claiming node and claimed node.
- It eliminates the already declared malicious node from the common node list.
- After that, claiming node sends claim_packet to "n" number of neighbors according to the maliciousness level.
- When it gets the response from the consulting nodes, it performs decision making by validating.

Our proposed methodology also works in two phases: *consensus phase* and *validation phase*. It differs in several ways: (1) It does not include the neighbor list of claimed node; (2) It does not look for common normal nodes for consulting; (3) It communicates with the neighboring nodes to find the status of the claimed nodes; and (4) Its computational complexity is very low because it does not perform consensus for each claimed node at a time.

Table 7 Maliciousness level list (ML_List)

Node_ID	Maliciousness level
C	HIG
D	MED
I	LOW

Consensus phase In this phase, monitor node communicates with the neighboring nodes to find the status of the claimed nodes. It sends a message containing a list of those nodes that have medium maliciousness level called *claim list* (C_List). It is acquired from the ML_List.

An initial *status list* (S_List) is maintained that contains the IDs of malicious nodes, and their claim status is "1" as shown in Table 8.

Algorithm 3: Consensus Phase

Input: Maliciousness level list (ML_List) => Claim List (C_List)
Output: Status List (S_List)
Begin
  For i=1 to (n.N_List) / 2 // n.N_List = Total number of Neighbors
    If rand(Node_ID.N_List) is Normal
      F1--> send C_List (rand(Node_ID.N_List))
    Else
      decrement i
    End If
  End For
F2-->Receive S_List
F3-->Update S_List
End

Note: F1, F2 and F3 are three functions.

During consensus phase, claiming sensor node updates its S_List after receiving S_List from other neighboring nodes according to Algorithm 3. Algorithm 3 states that if the received S_List is from the normal neighbor, then it should be used otherwise it should be discarded. The respective N_Claim in S_List should be updated against each corresponding node, whenever it receives the message in response of C_List that contains S_List.

Validation phase The next phase validates the maliciousness of the claimed node by analyzing the N_Claim of the final S_List. It performs a check that whether the N_Claim number is less than the validation threshold and

Table 8 Initial Status List (S_List) of MED level Malicious Nodes

Malicious node Node_ID	Number of claims N_Claim
D	1

updates the ML_List accordingly. If it is less than this value, then its maliciousness level status is updated to LOW. In other case, if it is more than that value, it is declared as abnormal node and its maliciousness level status is updated to HIG.

Let node J receive S_List from its neighbors. It updates its S_List accordingly and we assume that half of the neighboring nodes declare node D as a malicious entity. Hence, it maliciousness status is updated to HIG.

3.2.6 Consolation

The last phase of our proposed framework is *consolation*. It works on the basis of final ML_List, and only those nodes that have HIG maliciousness level are considered. It differs from [24] and works according to the following steps:

- *Update N_List*. The neighbor nodes that have high maliciousness level should be declared as malicious.
- *Update M_List*. Nodes that are not neighbors but are malicious should be highlighted.
- *Apply route discovery*. Find new routes that do not contain any malicious node as intermediate node.
- *Notify the sink*. Make a message that contains the list of malicious nodes and send it to the sink through a secure channel.

Here ML_List contains two nodes: node C and node D. Online prevention takes care from these nodes in future and does not allow them to affect the data aggregation and other application dependent functions.

4 Experiments and analysis

A simulator is implemented in Visual Studio.NET 2008 using C# to test the efficiency of the purely distributed specification-based detection scheme. The main focus of our test is to provide an insight about centralized–distributed approaches (security systems in which monitor nodes analyze the network and communicate with the base station using any secure communication mechanism) and show that they do not figure out the actual condition of the network properly.

4.1 Trace list

Let there be a sensor node X having n number of neighboring nodes. The numbers of nodes vary and are equal to 20, 40, 60, 80 or 100. A trace list (Trace_List) is randomly produced for 10,000 instances. The format of this list is shown in Table 9.

In our experiment, 100 different trace files are generated. Hence, there are 100 audit lists, which are used to find minimum and maximum sending rates. These are placed in T_List. Once the Trace_List is populated, the A_List is formalized by counting the number of send, receive, forward and retransmit packets for each node. Audit lists help in adjusting the threshold values in T_List.

Table 9 Trace list

Transaction type	Node X	Node Y
Send	A	B
Forward	C	D
Retransmit	E	F

4.2 Attack scenario (AS)

The proposed strategy is tested by launching four types of attack scenarios. The plotted values are acquired by taking average after running the simulation for 10 times in each case. In most of the cases, the average value of all the nodes that is calculated in attacked scenarios is nearby the average value of normal execution. But the average value of attacker nodes is higher or lower according to the AS type. These are discussed below.

4.2.1 Increased sending rate (AS-I)

During flood attack, the attacker sends more number of packets. Hence, the sending rate of the attacker nodes is increased by some fraction. There are nA number of attackers that are randomly selected and their sending rates are increased.

Figure 5 provides sending rate analysis. It shows that attacker nodes are sending more number of packets than normal nodes.

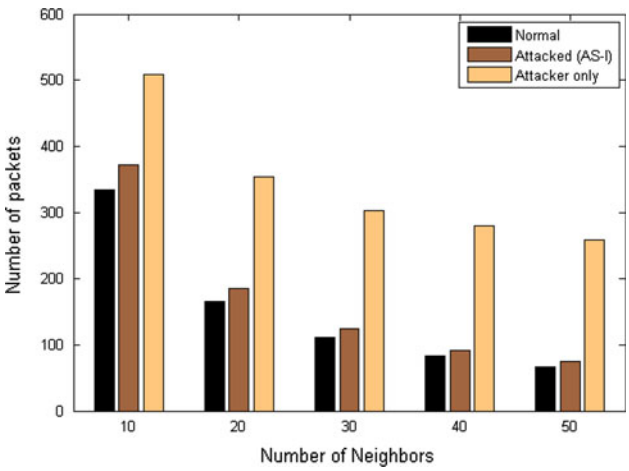

Fig. 5 Sending rate analysis

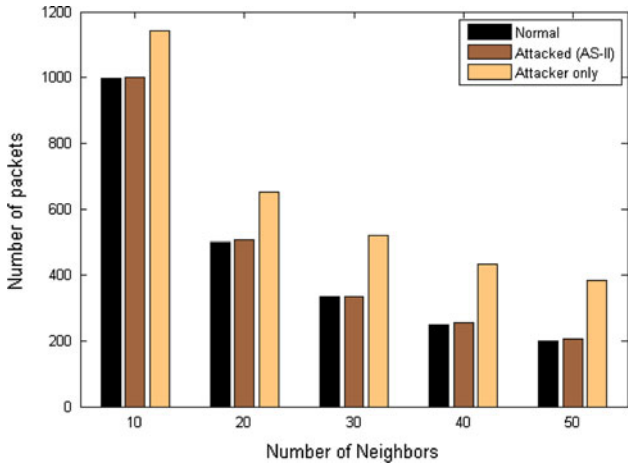

Fig. 6 Receiving rate analysis

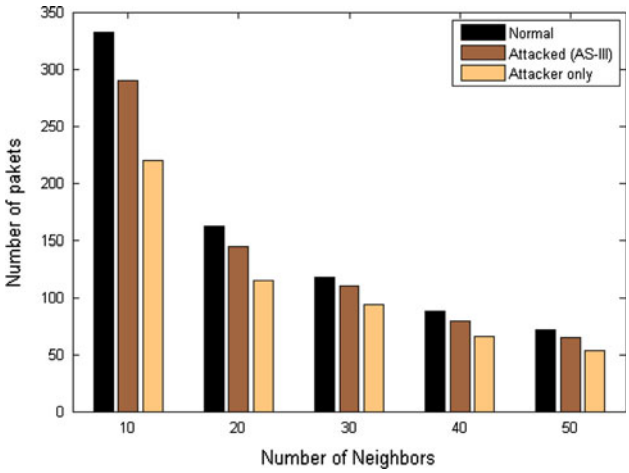

Fig. 7 Forwarding rate analysis

4.2.2 *Increased receiving rate (AS-II)*

In black-hole, sink-hole or worm-hole attack, the attacker receives more number of packets. Hence, the receiving rate of the attacker nodes is increased by some fraction. There are nA number of attackers that are randomly selected and their receiving rates are increased.

Figure 6 provides receiving rate analysis. It shows that the attacker nodes are receiving more number of packets than normal nodes.

4.2.3 *Decreased forwarding rate (AS-III)*

During selective forwarding, black-hole or sink-hole attack, the attacker forwards less number of packets. Hence, the forwarding rate of the attacker nodes is decreased by some fraction. There are nA number of attackers that are randomly selected and their forwarding rates are decreased.

Figure 7 provides forwarding rate analysis. It shows that the attacker nodes are forwarding less number of packets than normal nodes.

4.2.4 *Increased retransmission rate (AS-IV)*

In collision attack, the node that is attacked by a compromised node retransmits more number of same packets. Hence, the retransmission rate of the sender is increased by some fraction. There are nA number of such nodes that are randomly selected and their retransmission rates are increased.

Figure 8 provides retransmission rate analysis. It shows that the attacking nodes are retransmitting more number of packets than the normal nodes.

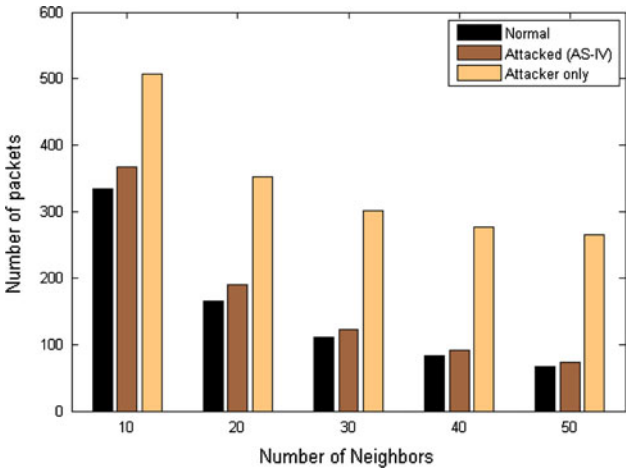

Fig. 8 Retransmission rate analysis

4.3 Discussion

The results show that if the node X sends the average value to the sink or base station to analyze the network whether it is in attack or not, it cannot figure out the actual scenario. But if the sensor node makes decision on its own and analyze the behavior of individual node, then it can detect the abnormal node efficiently. This shows that a centralized distributed approach cannot figure out the actual condition of the network properly. Therefore, a purely distributed security system is more appropriate for wireless sensor networks.

Here, an average audit list is maintained after generating 10 trace files for each attack pattern. These attack patterns vary from each other on the basis of the following parameters:

- Attack scenario (AS-I to AS-IV).
- Number of neighbors (10, 20… 50).

- Number of attackers (1, 3 or 5).

These are used to test two types of performance metrics to judge the effectiveness of the proposed scheme. These are intrusion detection rate and false positive rate.

4.3.1 Intrusion detection rate (IDR)

100 % detection rate means that the applied technique detects all the nodes that are compromised or not working properly. The formula for detection rate is mentioned below:

$$\mathrm{IDR} = \frac{\mathrm{A}}{\mathrm{A}+\mathrm{B}} \begin{cases} \mathrm{A} = \text{true positive} \\ \mathrm{B} = \text{false negative} \end{cases}$$

If the node is normal and it is declared as normal as well by the detection policy then it is A, while if the node is abnormal but declared as normal then it is B.

Our proposed methodology is working after random generation of trace files that are used to set thresholds. The attack scenarios are generated as described in previous section. Results depicted in previous section clarify that the compromised nodes deviate from the normal behavior. The interpretation of Flag_List shows that B is almost zero for each case. Hence, intrusion detection rate is almost 100 %.

4.3.2 False positive rate (FPR)

False positive means that a node is normal but wrongly declared as abnormal. The formula that is used to find the false positive rate of a system is mentioned below:

$$\mathrm{FPR} = \frac{\mathrm{C}}{\mathrm{C}+\mathrm{D}} \begin{cases} \mathrm{C} = \text{false positive} \\ \mathrm{D} = \text{true negative} \end{cases}$$

The average false positive rate of various AS for different number of neighboring nodes shows that the false positive rate of the proposed detection scheme is below 0.06 in most cases.

5 Conclusion

Vehicle cloud assures safety of the vehicles from accidents on the road, determines vehicle or driver condition using sensors and supply better assistance in case of abnormality, provide possible healthcare services to the passengers during traveling, discovers shortest reliable routes to the destination, and provide entertainment. It is important to secure the sensor network to achieve better performance of the vehicle cloud.

In this work, we have presented a novel intrusion detection framework to secure wireless sensor networks from routing attacks. The proposed approach is explained thoroughly using a flat wireless sensor network scenario. We test the specification-based detection scheme proposed for the presented example using a simulator that is implemented in C#. The results show that the specification-based detection scheme achieves higher detection rate and receives low false positive rate. These results also guide that each node should be treated independently in WSNs and centralized distributed detection schemes may fail to identify the network behavior whether it is normal or it is under any attack. Therefore, a purely distributed security system is more appropriate for WSNs.

In future we plan to apply the proposed intrusion detection framework to a clustering hierarchical routing protocol for WSNs such as LEACH to show its effectiveness with respect to throughput and energy efficiency.

Acknowledgments This research was supported by MKE (The Ministry of Knowledge Economy), Korea, under IT/SW Creative research program supervised by the NIPA (National IT Industry Promotion Agency)" (NIPA-2011-(C1090-1121-0003) and HEC (Higher Education Commission, Pakistan) under IRSIP (International Research Support Initiative Program) and indigenous fellowship program.

References

1. Abuelela M, Olariu S (2010) Taking VANET to the clouds. In: 8th International conference on advances in mobile computing and multimedia. Paris, France, November 2010
2. Bernstein D, Vidovic N, Modi S (2010) A cloud PAAS for high scale, function, and velocity mobile applications. In: Fifth IEEE international conference on systems and networks communications, November 2010, pp 117–123
3. Park P, Yim H, Moon H, Jung J (2009) An OSGi based in-vehicle gateway platform architecture for improved sensor extensibility and interoperability. In: 33rd Annual IEEE international computer software and applications conference, pp 140–147
4. Chen G, Fraichard T (2007) A real-time navigation architecture for automated vehicles in urban environments. In: IEEE intelligent vehicles symposium, Istanbul, Turkey
5. Cheikh FB, Mastouri MA, Hasnaoui S (2010) Implementing a real-time middleware based on DDS for the cooperative vehicle infrastructure systems. In: Sixth international conference on wireless and mobile communications, pp 492–497
6. Chen MC, Chen JL, Chang TW (2011) Android/OSGi-based vehicular network management system. Elsevier Comput Commun 34:169–183
7. Xie B, Kumar A, Zhao D, Reddy R, He B (2010) On secure communication in integrated heterogeneous wireless networks. Int J Inform Technol Commun Converg I(1):4–23
8. Farooqi AH, Khan FA, Wang J, Lee S (2011) Security requirements for a cyber physical community system: a case study. In: 4th International symposium on applied sciences in biomedical and communication technologies, Barcelona, Spain, pp 1–5
9. Imani M, Taheri M, Naderi M (2010) Security enhanced routing protocol for ad hoc networks. J Converg I(1):35–42
10. Kumar D, Aseri TC, Patel RB (2011) Multi-hop communication routing (MCR) protocol for heterogeneous wireless sensor networks. Int J Inform Technol Commun Converg I(2):130–145

11. Akyildiz IF, Su W, Sankarsubramaniam Y, Cayirci E (2002) A survey on sensor networks. IEEE Commun Mag 102–114
12. Wood AD, Stankovic JA (2002) Denial of service in sensor networks. Computer 54–62
13. Karlof C, Wagner D (2003) Secure routing in wireless sensor networks: attacks and countermeasures. In: The first IEEE international workshop on sensor network protocols and applications, pp 113–127
14. Roosta T, Shieh S, Sastry S (2006) Taxonomy of security attacks in sensor networks and countermeasures. In: First international conference on system integration and reliability improvements, Hanoi, Vietnam
15. Bojkovic ZS, Bakmaz BM, Bakmaz MR (2008) Security issues in wireless sensor networks. Int J Commun II(1):106–115
16. Ponomarchuk Y, Seo DW (2010) Intrusion detection based on traffic analysis and fuzzy inference system in wireless sensor networks. J Converg I(1):35–42
17. Teodoroa PG, Verdejo JD, Fernandez GM, Vazquez E (2009) Anomaly-based network intrusion detection: techniques, systems and challenges. Comput Sec 28:18–28
18. El-Semary AM, Mostafa MGM (2010) Distributed and scalable intrusion detection system based on agents and intelligent techniques. J Inform Process Syst 6(4):481–500
19. Farooqi AH, Khan FA (2009) Intrusion detection systems for wireless sensor networks: a survey. In: Communication and networking. Springer, Berlin, pp 234–241 (Jeju Island, Korea)
20. Rajasegarar S, Leckie C, Palansiwami M (2008) Anomaly detection in wireless sensor networks. IEEE Wirel Commun 15(4):34–40
21. Xie M, Han S, Tian B, Parvin S (2011) Anomaly detection in wireless sensor networks: a survey. J Netw Comput Appl 34:1302–1325
22. Farooqi AH, Khan FA (2012) A survey of intrusion detection systems for wireless sensor networks. Int J Ad Hoc Ubiq Comput 9(2):69–83
23. Krontiris I, Giannetsos T, Dimitriou T (2008) Launching a Sinkhole attack in wireless sensor networks; the intruder side. In: International conference on wireless and mobile computing networking and communications, pp 526–531
24. Krontiris I, Dimitriou T, Giannetsos T (2008) LIDeA: a distributed lightweight intrusion detection architecture for sensor networks. In: ACM secure communication, Istanbul, Turkey
25. Loo CE, Ng MY, Leckie C, Palaniswami M (2006) Intrusion detection for routing attacks in sensor networks. Int J Distrib Sensor Netw II(4):313–332
26. Su CC, Chang KM, Kuo YH, Horng MF (2005) The new intrusion prevention and detection approaches for clustering-based sensor networks. In: IEEE wireless communications and networking conference, pp 1927–1932
27. Roman R, Zhou J, Lopez J (2006) Applying intrusion detection systems to wireless sensor networks. In: 3rd IEEE consumer communications and networking conference, pp 640–644
28. Krontiris I, Dimitriou T (2007) Towards intrusion detection in wireless sensor networks. In: 13th European wireless conference, Paris
29. Stetsko A, Folkman L, Vashek M (2010) Neighbor-based intrusion detection for wireless sensor networks. In: 6th IEEE international conference on wireless and mobile communications, Valencia, pp 420–425
30. Drozda M, Schaust S, Szczerbicka H (2007) AIS for misbehaviour detection in wireless sensor networks: performance and design principles. In: Congress on evolutionary computation, Singapore, pp 3719–3726
31. Shaikh RA, Jameel H, Auriol BJ, Lee S, Song YJ (2008) Trusting anomaly and intrusion claims for cooperative distributed intrusion detection schemes of wireless sensor networks. In: The 9th international conference for young computer scientists, Hunan, pp 2038–2043
32. Ahmed KR, Ahmed K, Munir S, Asad A (2008) Abnormal node detection in wireless sensor network by pair based approach using IDS secure routing methodology. Int J Comput Sci Netw Sec VIII(12):339–342
33. Li G, He J, Fu Y (2008) A group based intrusion detection scheme in wireless sensor networks. In: The 3rd international conference on grid and pervasive computing—workshop, pp 286–291
34. Gupta S, Zheng R, Cheng AMK (2007) ANDES: an anomaly detection system for wireless sensor networks. In: International conference on mobile ad hoc and sensor systems, pp 1–9
35. Zhang Q, Yu T, Ning P (2008) A framework for identifying compromised nodes in wireless sensor networks. ACM Trans Inform Syst Sec XI(3):1–37
36. Da Silva APR et al (2005) Decentralized intrusion detection in wireless sensor networks. In: Proceedings of the 1st ACM international workshop on quality of service and security in wireless and mobile networks, Quebec, Canada, pp 16–23
37. Hai TH, Khan F, Huh EN (2007) Hybrid intrusion detection system for wireless sensor networks. In: Computational science and its applications, 4706th edn. Springer, Berlin
38. Phuong TV, Hung LX, Cho SJ, Lee YK, Lee S (2006) An anomaly detection algorithm for detecting attacks in wireless sensor networks. In: Intelligent and security informatics, San Diego, pp 735–736
39. Li L, Li Y, Fu D, Wan M (2010) Intrusion detection model based on hierarchical structure in wireless sensor networks. In: IEEE international conference on electrical and control engineering, Wuhan, pp 2816–2819
40. Marchang N, Datta R (2008) Collaborative techniques for intrusion detection in mobile ad-hoc networks. Ad Hoc Netw VI:508–523
41. Kim H, Chitti BR, Song J (2011) Handling malicious flooding attacks through enhancement of packet processing technique in mobile ad hoc networks. J Inform Process Syst 7(1):137–150