Ann. Telecommun. (2017) 72:265–282
DOI 10.1007/s12243-017-0566-7

Securing wireless sensor networks for improved performance in cloud-based environments

Ashfaq Hussain Farooqi[1] · **Farrukh Aslam Khan**[1,2]

Received: 13 March 2016 / Accepted: 28 January 2017 / Published online: 22 March 2017

Abstract Cloud computing has a great potential to assist in storing and processing data collected from sensors placed in any environment such as smart homes, vehicles, hospitals, enemy surveillance areas, volcanoes, oceans, etc. The sensors may be implanted in the form of a body sensor network or placed in the surroundings. The data recorded by these sensors may further be used for several applications implemented in the cloud as well as other services. Here, the data is acquired from sensors through the wireless medium. Recent studies show that wireless sensor networks (WSNs) are vulnerable to various kinds of security threats and there is a requirement of a security solution that safeguards them from lethal attacks. In this paper, we modify the low-energy adaptive clustering hierarchy (LEACH) protocol for WSNs and add the functionality of intrusion detection to secure WSNs from sinkhole, black hole, and selective forwarding attacks. The modified protocol is called LEACH++. We perform two types of analyses: numerical analysis to check the effect on throughput and energy, and simulations in Network Simulator-2 (NS-2) to prove the results found from the numerical analysis. The results are quite promising and favor LEACH++ over LEACH under attack with respect to throughput and energy consumption.

Keywords Wireless sensor networks (WSNs) · Cloud computing · Intrusion detection system (IDS) · Low-energy adaptive clustering hierarchy (LEACH) protocol

1 Introduction

In the near future, many businesses will shift toward the usage of cloud computing due to its decentralized processing and storage of data and online access to computing services. Cloud computing will play a major role in providing healthcare services to smart homes [4], healthcare institutes [30] [6], e-healthcare systems [46], and people at remote locations. Wireless medical sensor networks (MSNs) are considered to be the building blocks for remote health monitoring systems. To minimize the limitations of MSNs (computational power, data storage, and communication range/bandwidth) as standalone systems, these networks have been integrated with cloud computing environments [15]. Smart hospitals are equipped with a number of sensor devices that send critical information to the base station (BS) for monitoring the activities of the patients. The data collected from patients may be transferred to the cloud for high-performance computation, and this data would be accessible to the doctors from remote locations. This will increase the data utilization and will also help in providing prompt response by the healthcare experts. Similarly, cloud computing will facilitate the healthcare institutes to perform their tasks online with ease. A brief model is proposed in [6], [30] that motivates the use of IT in healthcare institutes to improve services and reduce the cost on manual processing. Vehicular cloud computing model combines different networking paradigms such as mobile ad hoc networks (MANETs), wireless sensor networks (WSNs), vehicular ad hoc networks (VANETs), and cloud computing to assist in facilitating new trends in traffic management, road safety,

✉ Farrukh Aslam Khan
fakhan@ksu.edu.sa

Ashfaq Hussain Farooqi
ashfaq.farooqi@nu.edu.pk

[1] National University of Computer and Emerging Sciences, A. K. Brohi Road, H-11/4, Islamabad, Pakistan

[2] King Saud University, Riyadh, Saudi Arabia

and intelligent transportation systems (ITS) [13]. The model provides a framework for the future transportation systems that ensure safety of vehicles from accidents on the road, determine the condition of vehicles or drivers using sensors, supply better assistance in case of emergency, provide possible healthcare services to the passengers during traveling, discover shortest reliable routes to the destination, and provide entertainment.

Cloud computing offers the capability to access shared resources and common infrastructure in a ubiquitous and pervasive manner. The computation and analysis performed at the cloud is based on the accuracy of the data received from the sensor nodes. Healthcare-based cloud products offer services in all the three layers of the cloud [29], [31], [43]. It may be used for recording the health-related data, examining of patients, and managing diseases. It enables to collaborate and communicate with the peers with ease and helps in the analysis of the gathered data. Healthcare organizations, ITS, smart home users, or others may use the cloud to analyze and facilitate the end user in a better way, as shown in Fig. 1. The sensors collect data from the devices (whether medical or other appliances) and send that data to a central location for storage and processing. These central locations are maintained in the cloud, which store and process all the data received from the sensors. Hence, the user may diagnose or analyze records from a remote location as well and assist the patient accordingly.

It is also observed that the patient's health could be seriously threatened by a malicious adversary, and using traditional security techniques may not be suitable in such a case [7]. To provide cloud services to smart homes, vehicle passengers, mobile users, healthcare institutes, etc., there is a requirement of a reliable network that ensures zero level of tolerance toward the loss of sensors' data. WSNs use wireless medium and provide ad hoc access to resources; therefore, they are more vulnerable to security threats than wired networks. Sensor nodes are self-controlled and an easy target for attacks from the adversaries [33]. Wood and Stankovic [40] discuss different attacks that cause denial of service (DoS) attacks at various layers of the sensor nodes. They also provide countermeasures against each threat, while Karloff and Wagner [21] focus on routing protocol attacks like homing, selective forwarding, black hole, sinkhole, and others. Roosta et al. [35] provide a detailed survey of security threats on WSNs. They also discuss countermeasures for some of them to cater these attacks efficiently. Bojkovic et al. [5] analyze the working of WSNs under several attacks and discuss the role of key distribution protocols in threat scenarios.

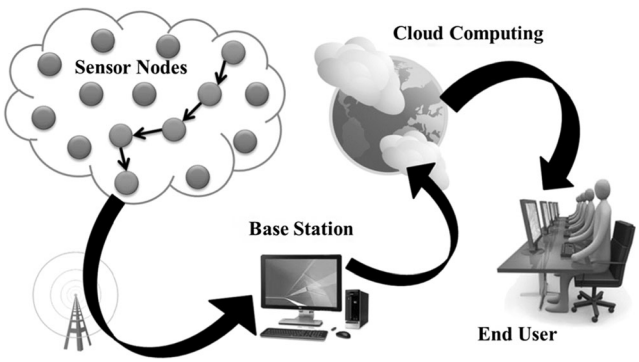

Fig. 1 Data transmission from a sensor network to the cloud for processing and analysis

Heinzelman et al. [18] presented an energy-efficient hierarchical routing protocol for WSNs called low-energy adaptive clustering hierarchy (LEACH). LEACH protocol works in rounds, and each round is composed of two phases, i.e., setup phase and steady phase. Setup phase takes less time while steady phase is for longer periods of time. In the setup phase, sensor nodes make clusters and cluster heads (CHs) set for each cluster node [17]. Different clusters use different timings to avoid collision among the cluster nodes. Sensor nodes send data message to their CHs in the steady phase and these CHs deliver the message to the BS after aggregation. LEACH avoids extra usage of energy, but it is considered vulnerable to different types of attacks [14], [21], [36], [42], [45]. Literature review shows that most of the work has been done in providing privacy and authentication to the data while few papers are available on the reliable transmission of data[37]. To ensure reliable delivery of data to the BS, which is further transferred to the cloud, a secure and reliable routing protocol is required. Most of the routing protocols are designed without considering security as a key feature but focus more on energy efficiency and throughput. Hence, they become vulnerable to routing attacks that may cause a fatal effect on the performance of these approaches.

In our previous work [13], we proposed an intrusion detection framework (IDF) for WSNs that 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 the adversary during the next epoch of time. In this paper, we add the proposed framework to the LEACH routing protocol to verify its effectiveness for WSNs. For that, we consider three attacks, i.e., sinkhole, black hole, and selective forwarding attacks. These attacks show a common feature: a compromised node tries to become the CH to behave maliciously. We call the modified LEACH protocol as LEACH++. Two kinds of tests are carried out to check the efficiency of this modified protocol. Numerical analysis guides us about the effect on overall energy utilization and throughput. Network Simulator-2 (NS-2) simulations help to validate the impact of LEACH++ during attack scenario with respect to throughput gained at the BS. The results show that LEACH++ receives more throughput than LEACH during attack, while it puts some burden on LEACH with respect to overall energy

consumption of the system. Normally, secure schemes bring some overhead than insecure solutions as they consume more energy due to computation and communication for the detection and collaboration [28]. NS-2 simulation results also favor LEACH++ because it achieves higher throughput than the LEACH under attack. The proposed approach provides a way to get more throughput even in the attack scenarios. It is better to adopt such a mechanism that gives access to the data to make better decisions in a cloud-based environment.

The remainder of the paper is organized as follows: In Section 2, the related work is discussed that covers the security requirements for sensor network based cloud computing and discusses various proposed solutions related to our work. Section 3 provides details about the proposed solution and the way it ensures reliable delivery of data by discarding the impact of malicious adversaries. Section 4 presents the numerical analysis with results, while Section 5 discusses the NS-2 simulations and results. Finally, Section 6 concludes the paper.

2 Related Work

WSNs are useful for analyzing the behavior of the surroundings without direct human interaction. These networks are easy to install and are used in various applications ranging from military to social life applications such as battlefield surveillance, smart homes, ubiquitous healthcare systems, smart cars, and others. Studies show that WSNs are vulnerable to several types of security threats [5], [21], [33], [35], [39], [44]. Medical WSNs also inhabit these threats and may lead to wrong output. Hence, there is a requirement of a security solution that can safeguard these networks from lethal attacks, which degrade the overall performance of the system [9], [11], [12], [23], [34], [38]. Routing protocols allow sensor nodes to communicate with the BS and vice versa. Akkaya and Younis [2] classify WSN routing protocols into three main categories; data-centric, hierarchical, and location-based routing protocols.

LEACH [17] is a well-known hierarchical routing protocol used for WSNs. LEACH protocol is vulnerable to different types of security threats, such as sinkhole, black hole and selective forwarding attacks [14], [21], [42], [45]. In black-hole attack, a compromised node uses the routing protocol to advertise wrong information to its neighbors to intercept more packets from the neighborhood, and then drops all the packets [3], [10]. Selective forwarding attack is considered to be more effective because the compromised node drops sensitive packets such as packets that report about the movement of the enemy in a particular time. Sinkhole attack is among the intelligent attacks in WSNs. In this attack, a compromised node makes itself more attractive to the neighboring nodes with respect to the routing metric and attracts as much traffic as possible [21], [24]. Hence, the compromised node receives most of the data messages and allows the adversary to launch severe attacks like dropping all or a fraction of packets and modifying them. In LEACH, a compromised node advertises ADV_CH message to become the CH in each round. Once it becomes the CH of some n number of nodes, it starts launching black-hole, selective forwarding, or sinkhole attacks.

There are several proposed schemes for securing LEACH protocol from insider attacks. Su et al. [36] proposed a security method that aimed to provide security from outsider as well as insider attacks. They presented a distributed detection policy where the CH enquires about cluster nodes and cluster nodes watchdog the working of the CH. They tested their approach for misbehavior in packet forwarding using LEACH protocol. The scheme works at two levels; cluster node and CH. It utilizes more energy and is computationally expensive. In another work, a secure cluster-based sensor network is proposed, which is an adaptive security design that safeguards against outsider attacks by authentication, while using centralized-distributed detection scheme for insider attacks [20]. Here, there is a response unit and trust evaluation unit. Response unit is responsible to decide about the intrusive behavior of the node in the network. The trust evaluation unit works at the monitor node where it maintains a trust status against each node and lowers the trust of the nodes if it receives any claim about it generated by the response unit. Lee et al. [25] proposed a specification-based centralized distributed detection scheme for LEACH protocol. According to them, the cluster member nodes and CHs act as monitor nodes and send claim message to the BS for taking proper action against the claimed nodes. The architecture presented by Zhang et al. [45] is used for intrusion detection system (IDS) agent that is installed at the CH, member nodes, and the BS. It is a distributed framework for mobile nodes that forms an ad hoc network and favors a cooperative approach for cooperation with neighboring IDS agents. Numerical analysis of the proposed specification-based distributed cooperative detection scheme is formulated to find the energy efficiency. Herbert et al. [19] proposed a framework for the detection of invariants or misbehavior in the installed system called hierarchical sensor network debugging (H-Send). This system is designed to help programmers to improve the system after determining faults so that the installed system is reprogrammed to have no bugs. That is why it is called a debugging tool. The system can detect sudden changes of the environments, while it is expensive in terms of communication, computation, and storage. Chen et al. [8] presented a security methodology to secure LEACH protocol from Sybil attack. The proposed scheme is based on received signal strength indicator from the nodes. It is a distributed-centralized approach in which sensor nodes detect the abnormal behavior of sensor nodes. According to the proposed approach, the IDS determines the Sybil attack once it receives more CH JOIN requests than a certain threshold.

Abdullah et al. [1] designed an IDS based on Stable Election Protocol for clustered heterogeneous WSNs. They propose to use K-Nearest Neighbor (KNN) classifier to find the outliers. They focus on LEACH routing protocol during the design of the IDS, but the results are prepared by applying KNN on a well-known dataset called KDDCup99. In a recent work, Sundararajan et al[37]. proposed a centralized IDS where the BS is responsible for calculating the intrusion ratio or maliciousness of the sensor nodes. Here, intrusion ratio is measured based on the transmission done or received by the CHs. In [13], the authors used a centralized approach for securing WSNs and discussed that the scheme does not suit well for networks that are infrastructure-less and where the nodes are self-organized. Masdari et al. [28] provided an analysis of secure LEACH-based clustering protocols in WSNs. These schemes focus to provide security to the data by using cryptographic solutions or by trust-based solutions that ensure authentication. In [37], Sundararajan et al. mention that there is no such technique that can be used for comparison of IDSs. They compare their proposed centralized security solution with MS-LEACH, which provides data confidentiality and authentication using shared pair-wise keys.

Roman et al. [34] focused on two elements to be included in a sensor node to have the ability to perform intrusion detection in a distributed environment. According to them, sensor nodes should have a data structure and detection entities. Data structure allows having information about neighboring nodes and existing malicious nodes. Detection entities perform local detection and global detection. The authors do not provide any simulation and analysis of the proposed idea. da Silva et al. [9] presented a simple architecture for monitor nodes to detect malicious activity of the neighboring nodes. According to them, a monitor node performs intrusion detection by listening in promiscuous mode. It makes a dictionary after listening to the communication of the neighboring nodes and applies predefined rules to this dictionary. If a node violates any rule, the failure counter increments. An alert is generated once the failure counter exceeds the expected value. The authors develop their own simulator to prove their method. Krontiris et al. [23] proposed a lightweight intrusion detection architecture to detect routing attacks. It is an online detection system that applies specifications to the promiscuously listened data. If a violation is detected, a response is initiated against that node.

3 Proposed IDS-based security solution

In our previous work [13], we proposed an IDF for securing WSNs from insider attacks, which ensured reliable delivery of data from the source to the destination. Here, we modify a well-known hierarchical routing protocol, i.e., LEACH, by adding features of the proposed framework to secure it from sinkhole, black-hole, and selective forwarding attacks. Literature study shows that LEACH is only concerned about the energy efficiency and lacks the security aspects required for securing WSNs [32]. In this section, we explain the working of the LEACH protocol and also discuss the proposed IDF for WSNs. We further describe our proposed modification to the LEACH protocol based on the proposed IDF.

3.1 Low-energy adaptive clustering hierarchy (LEACH)

LEACH protocol works in rounds, and each round is composed of two phases, as shown in Fig. 2.

- Setup phase: When the sensor nodes are deployed in a sensor field, they first elect the CH to make a hierarchy for sending data messages to the BS. Some CHs advertise ADV_CH message and tell other nodes to join their cluster. Sensor nodes calculate minimum distance towards BS with each choice of the CH. A sensor node selects one CH with whom it has minimum distance to the BS and sends a JOIN_REQ. CH makes a schedule and allows sensors to share data with it in a slotted time. Different clusters use different timings to avoid collisions among the cluster nodes.
- Steady phase: Once the sensor nodes select appropriate CH and become part of a cluster, they send data message to their CHs after sensing the environment in their slotted TDMA. CHs are responsible to deliver the data message after aggregation to the BS. Hence, LEACH avoids extra usage of energy.

LEACH protocol is energy-efficient as compared to multi-hop routing protocols. In LEACH, cluster nodes sense and send the sensed data to the CH in its allocated TDMA. CH aggregates the received data from all cluster nodes and transmits it to the BS. While in a multi-hop routing protocol, a sensor node finds a route to the BS and then uses it for communication whenever it senses anything. Thus, this process utilizes energy at each node and the overall energy consumption is more than that of a clustering routing protocol, as shown in Fig. 3.

3.2 Intrusion detection framework

Our proposed framework differs from the existing solutions presented in [9], [23], [34] in several ways. These existing solutions do not fulfill the requirements of a complete security framework. Moreover, these cannot be applied to a large category of routing protocols for WSNs such as hierarchical routing protocols. Our proposed IDF works in two modes: online prevention allows safeguarding from those abnormal nodes that are already declared as malicious, while offline

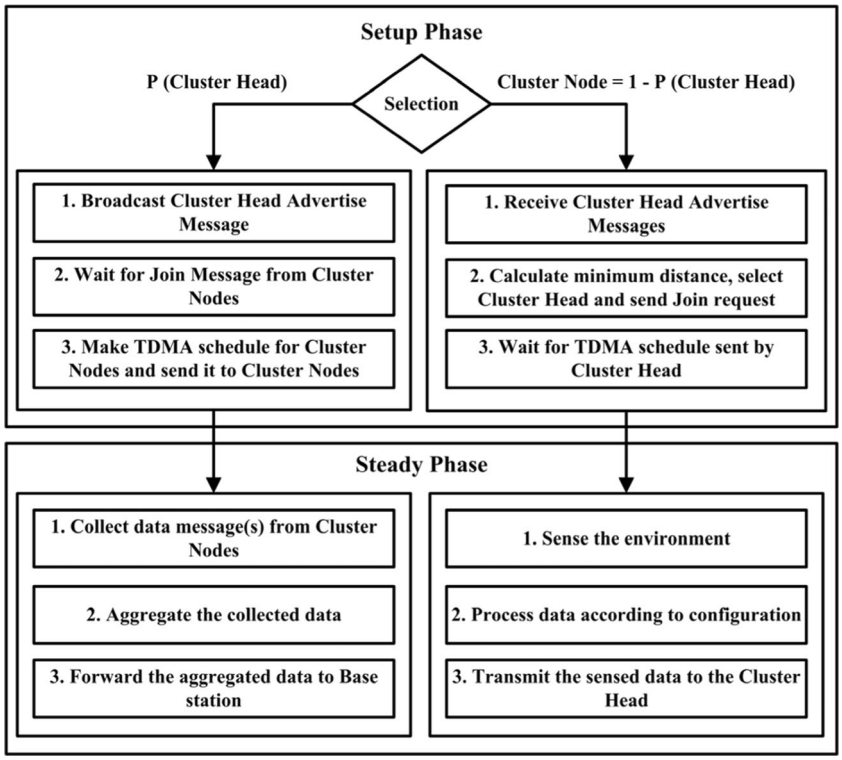

Fig. 2 Workflow of normal LEACH protocol

detection finds those nodes that are being compromised by the adversary during the next epoch of time. Offline detection applies distributed detection policy to find intrusions. It collaborates with neighboring nodes for making the final decision about the maliciousness of the claimed nodes. In this work, we illustrate the way IDF can be applied to a hierarchical routing protocol (i.e., LEACH) and prove its efficiency. The proposed framework is a distributed detection system. Figure 4 illustrates the key modules of the proposed IDF.

Some assumptions are made about the sensor network that helps in understanding the proposed mechanism. First, the sensor network should be static. Secondly, the 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. 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.

IDF works in promiscuous mode. It listens to every kind of traffic and then it takes the decision whether to process it or discard it. Whenever a node senses any message, it is collected by two modules: local auditing and data collection. Local auditing module verifies whether it is from a legitimate neighbor. If its status is clear, then the sensor node processes that message and performs the normal tasks. Data collection module forwards the received packets to content suppression unit

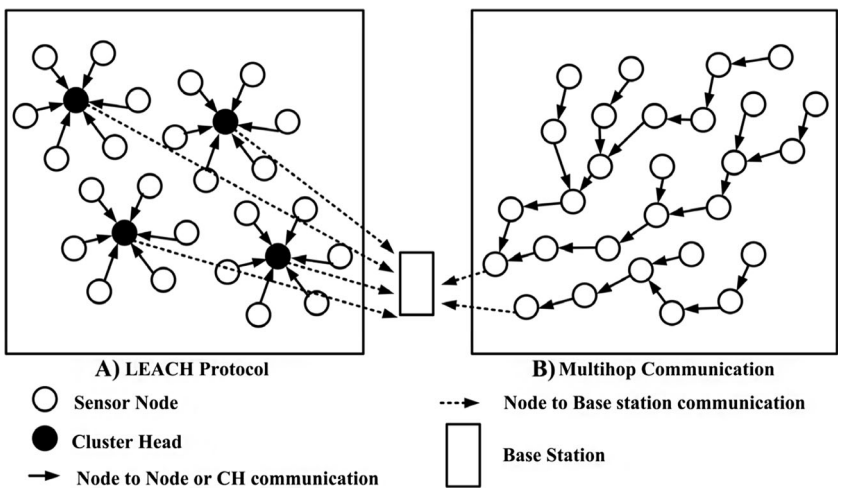

Fig.3 LEACH protocol vs. multi-hop routing protocol

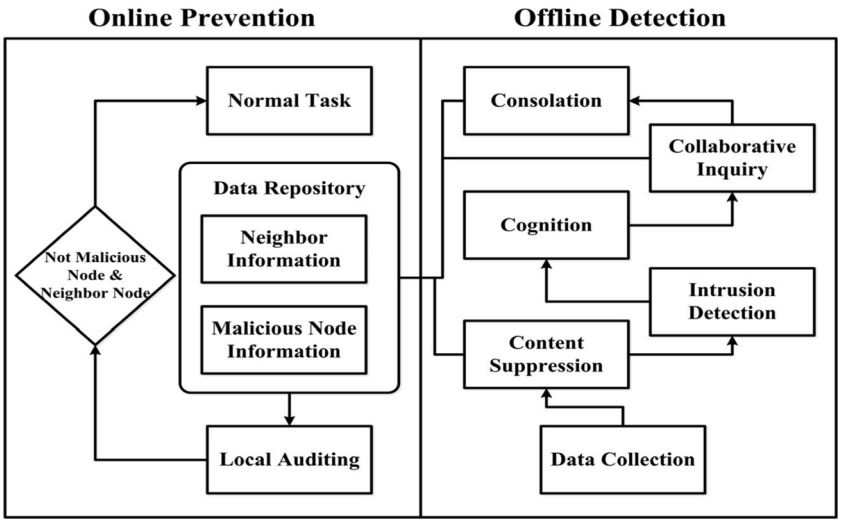

Fig. 4 Proposed intrusion detection framework

at the same time. This unit interprets the header to acquire the required information and process this data. Once the data is being processed, the intrusion detection policy is applied. The result of this unit is transmitted for cognitive decision-making. If the failure level is above a certain expected value, an alert is generated for collaborative inquiry. Nodes take final decision after communicating with neighboring nodes to declare the claimed node as malicious or normal. If it is declared malicious, an action is taken against it.

3.3 Securing LEACH

Sensor nodes develop a topology in each round to send the sensed data to the CHs. These CHs are responsible to transmit data to the BS. It is the responsibility of the LEACH protocol to assure that a node does not advertise ADV_CH in consecutive rounds. An adversary can launch sinkhole, black-hole, or selective forwarding attacks by modifying the LEACH configuration of a compromised node. Here, we assume that the compromised node becomes CH in each round and behaves maliciously.

In this sub-section, we explain the way the proposed IDF can be added to the LEACH protocol to secure it from sinkhole, black-hole, and selective forwarding attacks. Figure 5 shows how the IDF is installed in LEACH in both setup and steady phases. The modified protocol is called LEACH++.

3.3.1 Online prevention

Whenever a node receives ADV_CH message, the online prevention validates it by checking whether it is advertised by a legitimate node. If it is received from the normal node, then the sensor node adds it in the CH Choices table otherwise it discards it immediately. Sensor nodes do not have any knowledge about already declared malicious nodes. We propose that each sensor node should maintain a data repository, called Mal_List, for those nodes that are already declared as malicious. Local detection unit is responsible for validating the incoming ADV_CH. It works according to Algorithm 1 as follows:

Algorithm 1: Local Auditing at node X

```
Input: ADV_CH from other nodes
Output: Validation of packet (discarded or accepted)
Begin
  Foreach node ID in Mal_List
     If  nodeID of ADV_CH = nodeID
           discard it
      End If
   Otherwise
      Add in current CH Choices
End
```

Local detection units check the node ID of the received ADV_CH whether it is available in Mal_List or not. If it is available then it discards the ADV_CH, otherwise it adds the node ID in the current CH Choices table.

3.3.2 Offline detection

Sensor nodes do not maintain any record about CHs of the previous rounds. So, they are not capable of identifying whether they are making the same node as CH again and again. Offline detection finds those nodes that are trying to become CH in consecutive rounds. The following is the illustration of the key elements of IDF according to LEACH++.

Data collection Sensor node listens to the communication between neighboring nodes in promiscuous mode. In our proposed LEACH++ protocol, the data collection unit listens to

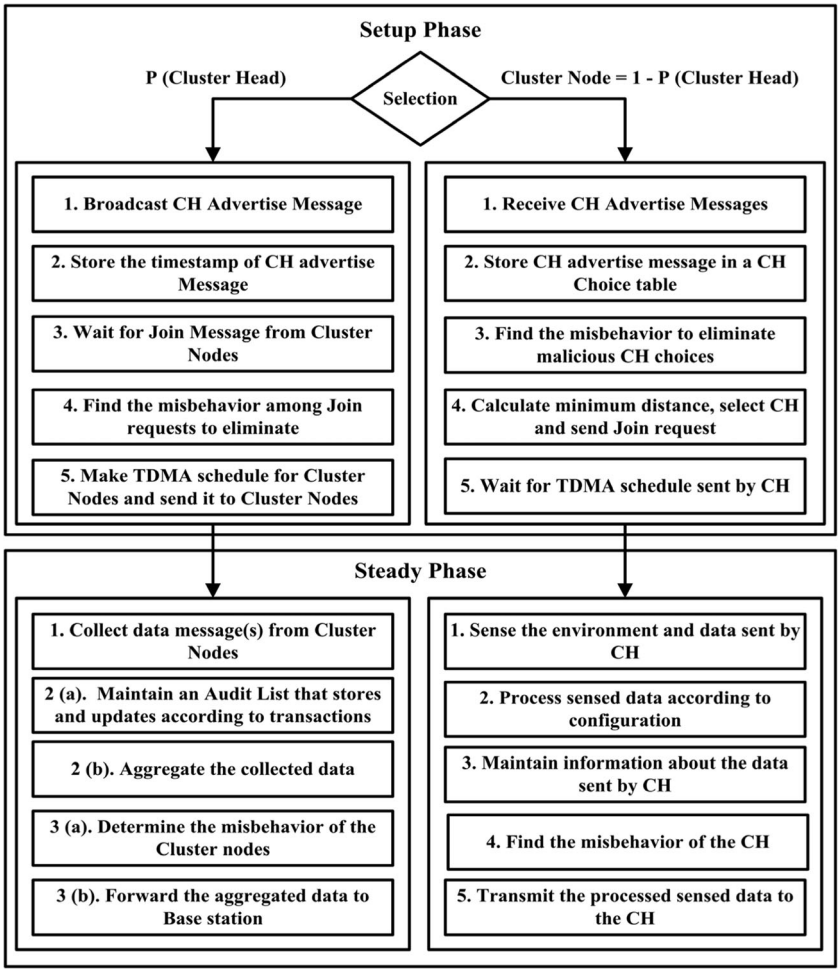

Fig. 5 Workflow of proposed LEACH++ protocol

ADV_CH packets and transmits them to the content suppression unit.

Content suppression Whenever some information is received from the data collection unit, the values are updated in an audit data list (A_List). Every node should maintain an A_List. It is a list that holds information about the past occurrences of the CHs. According to our analysis, after simulating LEACH protocol in NS-2, a node usually does not advertise itself as a CH in consecutive rounds. Hence, we propose that the length of A_List should be 3 * Number of CHs allowed in a round. It means that a node stores the CH information for three rounds. The values swap after the third round and so on to keep track of three consecutive rounds. Consider a sensor network having 100 sensor nodes with five CHs. Table 1 shows an example of the normal occurrences of CHs in three consecutive rounds A, B, and C.

After round C, the values stored for rounds B and C shift to rounds A and B respectively, while round C stores the CHs of the current round. This continues until the last round.

Suppose that node ID 13 is compromised by an adversary and it tries to become CH in each round. Table 2 shows the attack pattern in this scenario.

Intrusion detection Specification-based intrusion detection scheme is favored for WSNs by [9], [22], [34]. This detection technique is lightweight and has the ability to detect unknown attacks, while misuse detection mechanism cannot find unknown attacks and anomaly detection approach is expensive [11]. We propose some specifications for the detection of intrusions for the LEACH protocol, as follows:

- Step #1: The sensor node searches through A_List to

Table 1 An example of Audit List (A_List) for normal occurrences of CHs

	CH #1	CH #2	CH #3	CH #4	CH #5
Round A	Node 5	Node 13	Node 44	Node 71	Node 95
Round B	Node 12	Node 33	Node 56	Node 82	Node 91
Round C	Node 7	Node 17	Node 37	Node 66	Node 78

Table 2 An example of Audit List (A_List) during SBS-F attacks

	CH #1	CH #2	CH #3	CH #4	CH #5
Round A	Node 5	Node 13	Node 44	Node 71	Node 95
Round B	Node 13	Node 33	Node 56	Node 82	Node 91
Round C	Node 7	Node 13	Node 37	Node 66	Node 78

check whether the received ADV_CH packet is from the node that advertised the same packet in the previous consecutive rounds.

- Step #2: If the outcome of Step #1 is true, then it adds it to the claim list (C_List) and collaborates with other nodes to find its behavior at their sides. Otherwise, it follows the normal cluster formation steps.

The format of C_List is very simple. It contains node IDs of claimed nodes only. The proposed approach lies in purely distributed IDS category [11]. Every node in the network performs intrusion detection to find the compromised nodes.

Collaborative inquiry Sensor nodes advertise C_List once they detect any malicious node. If the rule in Step #2 fires, then this unit activates and shares the status of malicious nodes with other nodes. The authors in [23], [27], [34] favor the collaborative inquiry about the maliciousness of a node with other nodes. Here, collaborative inquiry unit works in two phases: consensus and validation.

(a) Consensus phase: In this phase, the monitor node advertises C_List to tell others about the detected malicious node(s). Once the other nodes detect the malicious nodes at their end, they advertise C_List as well in their slotted time. Sensor nodes receive C_List from other nodes in the setup phase and maintain the status list (S_List).
(b) Validation phase: Compromised nodes can act maliciously and can advertise wrong claims. Hence, the validation unit shows whether the received C_List is from a legitimate node or already declared malicious node.

S_List contains node IDs of claimed nodes and their status. Initially, the status is "1." Once a node receives C_List from other nodes, it updates S_List and increments the status values against those nodes that are already present in the S_List.

Let a node X detects two nodes that advertise ADV_CH in three consecutive rounds as shown in Table 3. Node X receives C_List from a node Y and it contains two node IDs: node 16 and node 13. Node X updates its S_List by adding node 16 and the status value "1," while it increments status value of node 13 by 1, as shown in Table 4.

Consolation Consolation performs actions to assure that the malicious node does not take part in further degradation of the system's throughput. We propose that if the status value of any node is above certain expected value, it should be added in the malicious nodes list (Mal_List). Here, the expected value can be any ratio of the total number of nodes and CHs. Hence, online prevention avoids these nodes to become CH in future.

Table 3 S_List at node X

Node ID	Status
Node 12	1
Node 13	1

4 Numerical Analysis

LEACH protocol works in two phases: setup phase and steady phase. These phases consume different amounts of energy. Setup phase takes less time while steady phase continues till the end of each round. We formulate equations to analyze the energy utilization and throughput by LEACH protocol in three scenarios: (1) normal execution, (2) attack launched by the adversary, and (3) LEACH++. Here, our focus is to find the effect of the proposed solution on the transmission of messages during collaborative inquiry, which increases the routing overhead. The energy used in the computation is very low as compared to the communication. The proposed solution increases the energy used in computation, but it is negligible as compared to the overall working of the LEACH protocol. Table 5 describes different notations that are used to derive the equations.

4.1 Normal execution

Sensor nodes develop a topology after their installation in a sensor field. This topology depends on the routing protocol. Here, sensor nodes use LEACH protocol to form a path that is used for transmission of information to the BS. The amount of energy these nodes consume in a normal scenario during the formation of the topology and data transmission is discussed below.

4.1.1 Energy used in setup phase

When a sensor network is deployed, ideally T_{CHeads} number of nodes advertise ADV_CH message to form clusters.

$$\text{Energy used in cluster setup} = T_{\text{CHeads}} * E_2 \quad (1)$$

Table 4 Updated S_List

Node ID	Status
Node 12	1
Node 13	2
Node 16	1

Table 5 Notations used in numerical analysis

Notation	Description
E_1	Amount of energy utilized by a node during sending a message to CH and vice versa. It is the energy consumed during the short-range communication
E_2	Amount of energy utilized when a CH sends message to the BS and vice versa. It is the energy consumed during long-range communication
E_{Setup}	Total amount of energy used by sensor nodes during the setup phase
E_{Steady}	Total amount of energy used by sensor nodes during the steady phase
T_{Nodes}	Total number of sensor nodes except CHs
T_{CHeads}	Total number of CHs
C_{CHeads}	Number of compromised sensor nodes
N_{Data}	Amount of data messages sent by each node to the CH
C_{Data}	Amount of data messages sent by CH to the BS
R_{Sec}	Length of a round in seconds. It is assumed that the rounds have fixed length

Sensor nodes reply with JOIN_REQ message to join the clusters and CHs respond with an acknowledgement.

$$\text{Energy used in cluster formation} = (T_{\text{Nodes}} * E_1) + (T_{\text{CHeads}} * E_1) \quad (2)$$

Hence, from Eqs. (1) and (2), we conclude that

$$E_{\text{Setup}}(\text{LEACH}) = E_2 * (T_{\text{CHeads}}) + E_1 * (T_{\text{Nodes}} + T_{\text{CHeads}}) \quad (3)$$

4.1.2 Energy used in steady phase

In the steady phase, sensor nodes start sending messages to the CHs, and CHs pass the information after aggregation. So, we track the number of data packets that are sent during this phase to calculate the energy usage. If the clusters have same size or they are of uniform size, then the number of nodes in a cluster is equal to (T_{Nodes}/T_{CHeads}). If there are 100 sensor nodes, then each cluster has 20 nodes. But here, we use the cluster formation algorithm to determine the number of nodes in each cluster, as shown in Algorithm 2.

Cluster size defines the number of clusters to be formed, while the number of nodes define the nodes that will be in these clusters. Initially, a random number is assigned to each cluster that sets the percentage of nodes to be present in that cluster, which ranges from 10 to 50. Here, we want that a cluster should have at least 10 nodes or maximum 50 nodes out of 100. In the next FOR loop, the number of nodes are assigned with reference to its percentage. The results of this algorithm show that most of the time, the clusters are of different sizes.

The next step is to determine the number of data packets received by a CH in one round. Here, we assume that a node sends about 1 to 10 data messages in one round to the CH. To achieve that, the following algorithm is used:

The clusters having different numbers of nodes are returned by the cluster formation algorithm. Each node in the cluster sends 1 to 10 data messages in a round known as $\mathbf{N}_{\mathbf{Data}}$, which is collectively considered as the cluster data $\mathbf{C}_{\mathbf{Data}}$. Here, we consider that once the CH receives 20 data messages, it sends an aggregated message to the BS. Hence, the energy consumption in the

Algorithm 2: Cluster formation

Input: Number of normal nodes and cluster size
Output: Clusters with different numbers of nodes
Begin
  For all cluster size **Do**
    Temp_Array: Assign random number from 10 to 50 to a temporary array
    Sum: It holds the sum of these numbers
  End For
  For all cluster size again **Do**
    Clusters: Assign value to cluster equal to (Temp_Array/Sum) * Number of normal nodes
  End For
End

steady phase is as follows:

$$E_{\text{Steady}}(\text{LEACH}) = E_1 * T_{\text{Data}}(R_{Sec}) + E_2 * T_{\text{Data}}(R_{Sec})/20 \quad (4)$$

4.1.3 Total energy and throughput

Total energy consumed by the LEACH protocol by executing in normal fashion can be achieved from Eqs. (3) and (4).

$$T_{\text{Energy}}(\text{LEACH}) = \sum_{i=0}^{n} E_{\text{Setup}}(\text{LEACH}) + E_{\text{Steady}}(\text{LEACH}) \quad (5)$$

Here, n is the number of rounds. Throughput is the amount of data packets received by the BS. From Algorithm 3, we calculate:

$$T_{\text{Throughput}}(\text{LEACH}) = \sum_{i=0}^{n} C_{\text{Data}}[i] * R_{Sec} \quad (6)$$

4.2 Attack launched by adversary

LEACH protocol shows resilience to a number of attacks due to its adaptive nature of the clustering mechanism. However, there are some attacks that can degrade the performance of the LEACH protocol. Black-hole, sinkhole, and selective forwarding attacks are among these attacks. In these attacks, a legitimate node can be compromised to perform the malicious activity. This malicious activity depends on the type of attack, such as dropping of packets, sending data packets containing wrong information, and dropping a fraction of packets. These attacks are considered as insider attacks because these are launched by nodes that are part of the current network.

Here, we assume that a compromised node becomes a CH in each round and does not send any message to the BS. It means that the attacking node drops all the packets that should be transmitted to the BS after taking aggregate of the data received from cluster nodes. Here, we called it ALEACH: LEACH under attack.

4.2.1 Energy used in setup phase

Total energy utilization for this phase remains the same as in Eq. (3) for the attack scenario because the compromised nodes work similar to the other nodes. In the attack scenario, the compromised nodes advertise ADV_CH message to assure that it becomes CH in each round. This does not affect the overall energy consumption in the setup phase.

4.2.2 Energy used in steady phase

In the attack scenario, the amount of data messages received by a CH remains the same as in Algorithm 3. This is because according to our assumption, a compromised node becomes the CH in each round, while the working of the other nodes remain the same.

The messages received by the CH are:

$$N_{\text{Data}}(R_{Sec}) = N_{\text{Data}} * T_{\text{Nodes}} * R_{Sec} \quad (7)$$

There are C_{CHeads} number of compromised nodes that form clusters. These nodes do not send messages to the BS. Hence, the amount of data packets received by a BS in a round during attack are:

$$C_{\text{Data}}(R_{\text{Sec}}) = C_{\text{Data}}\,\text{of\_normal\_node} * R_{\text{Sec}} - C_{\text{Data}}\,\text{of\_compromised\_node} * C_{\text{CHeads}} \quad (8)$$

Algorithm 3: Amount of Cluster data and Total data in a round

Input: Clusters returned by cluster formation algorithm and cluster size
Output: Each cluster data and total data of all clusters in a round
Begin
  For all cluster size **Do**
    For all cluster nodes of this cluster **Do**
      Node data (N_{Data}) = Node data + Random number between 1 to 10
    End For
  Cluster data of that cluster (C_{Data}) = Node data
  Total data (T_{Data}) = Total data + Cluster data of that cluster
  End For
End

Therefore, from Eqs. (7) and (8), we have:

$$E_{\text{Steady}}(\text{ALEACH}) = E_1 * N_{\text{Data}}\,(R_{Sec}) + E_2 * C_{\text{Data}}\,(R_{Sec})/20 \quad (9)$$

4.2.3 Total energy

Total energy consumed by the attacked LEACH by executing in a normal fashion can be achieved from Eqs. (3) and (9).

$$T_{\text{Energy}}\,(\text{ALEACH}) = \sum_{i=1}^{n} E_{\text{Setup}}\,(\text{ALEACH}) + E_{\text{Steady}}\,(\text{ALEACH}) \quad (10)$$

Here, n is the number of rounds in which the network is under attack. Throughput is the amount of data packets received by the BS

$$T_{\text{Throughput}}\,(\text{ALEACH}) = \sum_{i=1}^{n} C_{\text{Data}}[i] * R_{Sec} - C_{\text{Data}}\,\text{of\_compromised\_node}^{*}\,C_{\text{CHeads}} \quad (11)$$

4.3 LEACH++

In this sub-section, we calculate the effect of the proposed IDF to the original LEACH protocol with respect to energy consumption and throughput after securing from attacks.

4.3.1 Energy used in setup phase

The proposed mechanism may increase the energy usage in the setup phase. Let a node X detects that node Y advertises to become CH in consecutive rounds; node X advertises the claim list (C_List) to make the final decision. Meanwhile, other nodes also share their claim lists and nodes maintain the status of the claimed nodes as discussed in Section 3.3. If a node is declared as a malicious node, then the first node among the detectors advertises ADV_CH to become the CH.

Amount of energy used for detecting and changing CH is as follows:

$$E_{\text{ChangeCH}} = 2E_1 * \left(\frac{T_{\text{Nodes}}}{T_{\text{CHeads}}}\right) + E_2(T_{\text{Nodes}}) + E_2 \quad (12)$$

Here, $E_2(T_{Nodes})$ energy is used for sharing the claim list. E_2 amount of energy is used by a node that advertises ADV_CH message and $2E_1 * (T_{Nodes}/T_{CHeads})$ energy is again used to join the CH.

Hence, the overall energy utilization in the setup phase is the sum of Eqs. (3) and (12), as shown in Eq. (13).

$$E_{\text{Setup}}(\text{LEACH}++) = \{E_2 * (T_{\text{CHeads}}) + E_1 * (T_{\text{Nodes}} + T_{\text{CHeads}})\} + E_{\text{ChangeCH}} \quad (13)$$

4.3.2 Energy used in steady phase

The proposed approach does not play any role in the steady phase. The total energy that is consumed in this phase may differ from the LEACH protocol because the compromised node does not send any message during this phase due to its configuration. Hence, the number of nodes that send data to CHs differs by the number of compromised nodes.

$$E_{\text{Steady}}(\text{LEACH}++) = E_1 * \left(N_{\text{Data}}{}^{*}(\,T_{\text{Nodes}} - C_{\text{CHeads}}) * R_{Sec}\right) + E_2 * (C_{\text{Data}} * T_{\text{CHeads}} * R_{Sec}) \quad (14)$$

4.3.3 Total energy

The total energy consumed by LEACH++ protocol by executing in a normal fashion can be achieved from Eqs. (13) and (14).

$$T_{\text{Energy}}\,(\text{LEACH}++) = \sum_{i}^{n} E_{\text{Setup}}\,(\text{LEACH}++) + E_{\text{Steady}}\,(\text{LEACH}++) \quad (15)$$

Here, n is the number of rounds in which nodes detect any malicious activity. Throughput differs in different rounds based on the impact of the attackers on the network, the time when the malicious activity is detected, and the round in which isolation of the compromised node is achieved.

Data loss during the attack is high as the compromised node drops all or selectively forwards few data messages to the BS. Here, when the attacker becomes successful, it drops all the packets of that cluster. The impact of the attacker still remains at it does not provide its own information throughout its lifetime. Hence, even if the network detects the malicious node, there is still some data loss. This is measured using the Algorithm 4.

$$T_{\text{Throughput}}\,(\text{LEACH}++) = \sum_{i}^{n} C_{\text{Data}} * T_{\text{CHeads}} * R_{Sec} - D_{\text{Lose}} \quad (16)$$

Here, n is the number of rounds in which the nodes detect any malicious activity.

Algorithm 4: Data loss during and after detection of attack

```
Input: Clusters, Cluster Data, Number of attackers
Output: Data loss (D Loss)
Begin
    For all rounds Do
        If network is under attack Do
            For all successful attackers Do
                Data loss = Data loss + Cluster data of the compromised cluster
            End For
        Else If network recovered from attack Do
                Data loss = Data discarded by the attacking node
        End If
    End For
End
```

4.4 Discussion

We simulate and test the effect of the proposed approach on the LEACH protocol. Simulation parameters are shown in Table 6. We assume that the sensor nodes communicate in two ranges and consume different amounts of energies, e.g., nodes utilize three times more energy for long distance communication than for the short range communication. There are 10 rounds with 25 s for each round in a simulation of 250 s. There are 100 nodes and varying number of CHs. The amount of data a node sends to the CH in one round is a random number between 1 and 10 data packets, while the CH sends 1 packet once it receives 20 data messages from the cluster nodes.

Hierarchical routing protocols can minimize the overall impact of attacks on the throughput by increasing the number of CHs. We test the LEACH protocol by increasing the number of CHs for each round and launch different numbers of attacks. Figure 6 shows the results. The results are calculated after launching 1, 2, and 3 attackers in 10 different simulations by setting the number of CHs as 5 and 10. Here, we use the average value acquired from the results after simulation. First, CHs are set to 5 and attackers vary like 1, 2, and 3, and then CHs are set to 10 and attackers vary like 1, 2, and 3 for few simulations.

Table 6 Simulation parameters

Name	Value
Simulation time	250 s
Rounds	10
R_Sec	25 s
E1	1 Unit
E2	3 Unit
T_Nodes	100
T_CHeads	5, 10
C_CHeads	1, 3
N_Data	Random (1, 10)
C_Data	(Total Data)/20

Here, the normal throughput of LEACH having 5 and 10 clusters are depicted for 10 rounds by setting the simulation time as 250 s and round time as 25 s. The results are very close to each other while the results acquired by launching different numbers of attacks and taking the average shows less degradation in case of 5 clusters. The degradation for 5 CHs is about 37% while for 10 CHs, it is around 20%. Still the effect is high for both the cases as it degrades the performance of the output acquired from such missing data. However, there is a negative impact of increasing CHs in the LEACH protocol. This directly affects the energy utilization of the system.

Cluster size should be kept optimal to achieve high throughput and ensure low energy consumption. Consider the results shown in Fig. 7.

CHs take more energy to communicate with BS as compared to energy used by nodes to communicate with the CHs. In Fig. 7, we simulated for 10 times each by setting the CH as 1 to 10. The workload of an individual cluster in different sized clusters is depicted by averaging all the simulation results and rounds in it. If there are no clusters or only one cluster, then the load on the CH is very high and that helps it to die soon. The optimal cluster size here is 5, as the average throughput or energy utilization of the CH does not change more for latter cluster sizes. As the cluster size increases, it puts extra burden on the network. The above results also prove that increase in the number of CHs is directly proportional to a factor increase in energy utilization. The results depicted here show about 10% increase in the overall working of the network having 10 clusters, rather than having 5 clusters. In our experiments, we have mostly used the cluster size equal to 5.

Now, consider there are 100 nodes with 5 CHs. The pattern of energy utilization by LEACH and LEACH++ during each round acquired from the above numerical analysis is shown in Fig. 8. It shows that LEACH++ consumes more energy than LEACH. The factors that influence the results are discussed below.

The energy utilization of the previous round is added in the current round to have a clear look at the difference. In round 1, both consume the same amount of energy because the security

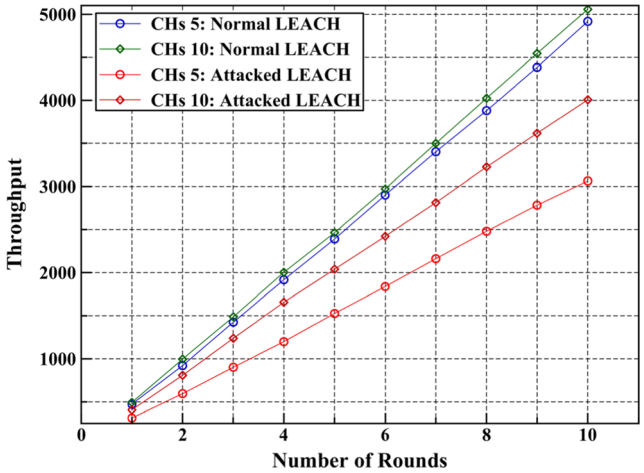

Fig. 6 Impact of sinkhole, black-hole, and selective forwarding attacks on normal LEACH with respect to throughput by varying the CHs

patch that is included in LEACH for LEACH++ works after an attack is detected. There is a change in energy consumption from round 3 onwards. We assume that the sensor network is attacked by the adversary at this point. Hence, the energy consumed by the network is according to Eq. (13). The compromised CH does not communicate with the BS so it does not take part in enhancing the energy. In round 4, LEACH++ performs its activity and utilizes more energy than LEACH. It is not a considerable amount by looking at the solution it provides to secure the LEACH from various attacks. From round 4 onwards, the network works according to LEACH because there are no more attacks for which any change in energy occurs. We ignore the energy utilization on the computation, as it is negligible as compared to communication overhead. We conclude that LEACH++ puts some burden on LEACH protocol but considering the positive impact of this overhead, it can be neglected.

In our next result, we show the importance of the security patch and show its impact on throughput. Figure 9 shows the effect of the attack on throughput for both LEACH and LEACH++ having 5 CHs and attacking nodes. The throughput achieved by LEACH++ as compared to the attacked LEACH shows that LEACH++ recovers from the attack while the performance of LEACH protocol degrades after an attack is launched. Here, we can conclude that LEACH++ can provide better throughput in case of attack on the LEACH protocol.

5 NS-2 implementation and testing

Heinzelman et al. [17] proposed the LEACH protocol and implemented it in NS-2.15b for their tests. This implementation is available online [16]. There is also a patch for LEACH available for NS-2.34. We use that patch to test our proposed approach. LEACH protocol is modified to launch attacks. It is further enhanced by adding the proposed solution to detect this attack and perform some appropriate action.

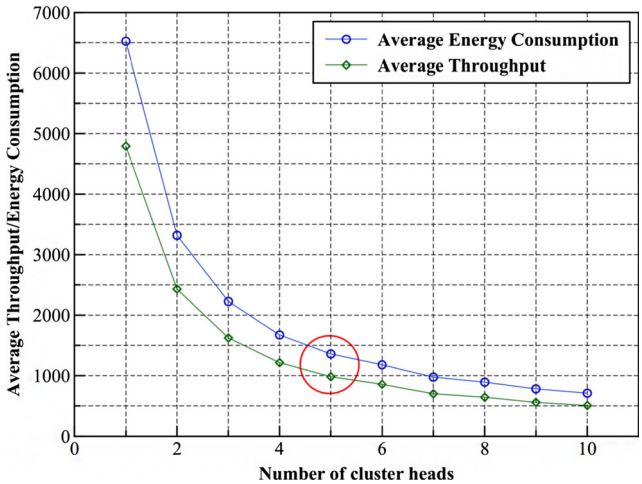

Fig. 7 Average energy and throughput utilization by LEACH protocol for a different number of CHs

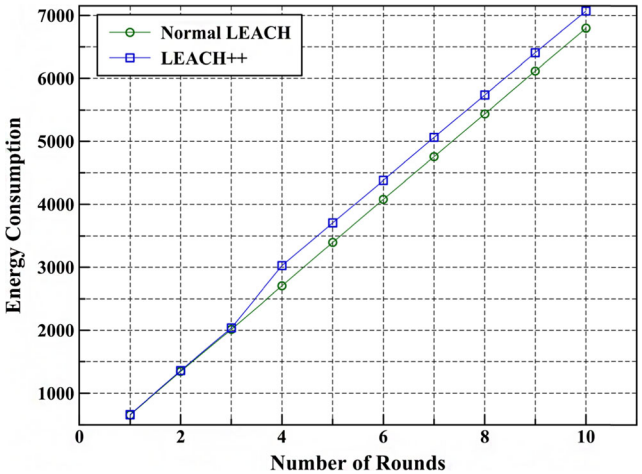

Fig. 8 Comparison of energy consumption between LEACH and LEACH++

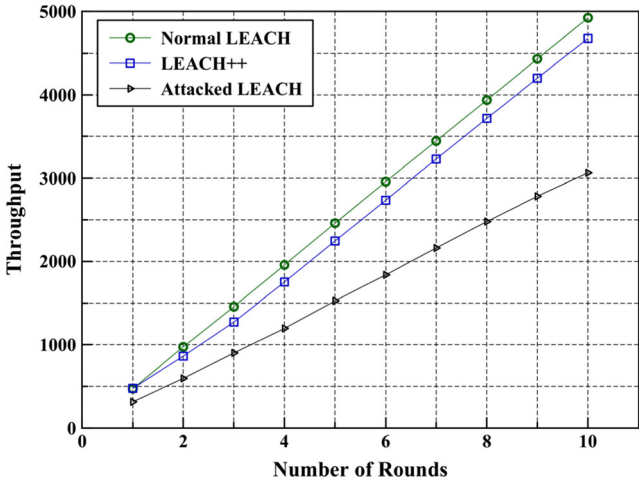

Fig. 9 Throughput analysis: LEACH++ performs better in attacked scenario

5.1 Introduction of a malicious node

Three changes are required to launch black-hole, selective forwarding, or sinkhole attack in LEACH protocol. These are (1) Increase the energy of the malicious node, (2) Make it a CH in each round, and (3) Drop all packets or selectively drop important messages. Here, we assume that once the compromised node becomes CH, it drops all the packets. The implementation details are as follows:

5.1.1 Increasing energy

Energy of the malicious node is increased so that it does not die throughout the simulation and shows others that it has more energy to play the role of a CH. We made some changes in a file "uamps.tcl" and modify the function "proc leach-create-mobile-node {id}" to increase the energy of the attacking node. It sets infinite amount of energy for the attacking node.

5.1.2 Making CH

Attacking node should be a part of cluster choices for each node in every round. There are different ways to achieve it. In our experiments, we append the malicious node request to the CH choices. Hence, it appends the node's ID of the attacking node in the list of cluster choices. So, there are some nodes that make it a CH in consecutive rounds.

5.1.3 Dropping packets

Once an attacking node becomes a CH, it drops all the packets. We made some modification that drops the packet received by the malicious node. It checks for the attacker node's ID to avoid sending data to the BS. It also updates the number of times it does not forward the message to record the amount of packets that are dropped.

5.2 LEACH++

Original implementation of the LEACH protocol is modified to add more security against insider attacks. The proposed approach works in two phases: online prevention and offline detection. In this section, we present the details of their implementation in NS-2.

5.2.1 Online prevention

Local detection engine validates and verifies the incoming packets whether these are from legitimate nodes or not. For LEACH protocol, it verifies the list of cluster choices whether it contains any node that is already declared as malicious. If it finds that node, then it eliminates it from the list of cluster choices.

The function that provides the list of possible CHs in a round is "Application/LEACH instproc recvADV_CH {msg}" available in "ns-leach.tcl". ClusterChoices_ is the array that contains the list of current choices for the CH. In function "findBestCluster {}", a list is introduced called malicious node list. This list contains node IDs of those nodes that are declared as malicious by offline detection. This avoids the malicious nodes from becoming CH again by removing the IDs of the malicious nodes from the cluster choices.

5.2.2 Offline detection

Intrusion detection works after collecting some amount of data. We simulate LEACH protocol for 100 nodes having 5 CHs for 25 times keeping the simulation time as 800 s to find whether nodes advertise ADV_CH in consecutive rounds or not. We find that nodes do not advertise ADV_CH in consecutive rounds. So, we track three consecutive rounds rather than judging for more rounds to make the decision about the maliciousness of a node.

In offline detection, data collection unit provides data to the content suppression unit to arrange the data for intrusion detection. "clusterChoices_" is the data that is provided to the content suppression unit. It makes a record list of cluster choices for three consecutive rounds. Here, if a node is selected as a CH by a node for third consecutive time; it is declared as malicious. The process of detecting intrusions continues until the last round to avoid the malicious node from becoming CH in continuous rounds.

Intrusion detection checks whether a node becomes a CH for three consecutive times. If it detects such a behavior, then it appends it in a maliciousNodes_ list. "ns-leach.tcl" executes for each node in every round. Therefore, maliciousNodes_ is visible for all the nodes. Hence, we do not need to implement

Table 7 Simulation parameters

Name	Value
Simulation time	800 s
Sensor nodes	100
Base station	1
Base station location	70, 100
Number of clusters	5
Initial energy	2 J
Dimension	1000 * 1000

the collaboration part here. But, in real scenarios, the collaborative inquiry will be required.

5.3 Results and Discussion

LEACH++ implementation shows that the modification does not put burden on the LEACH protocol with respect to overall memory utilization and computation. We simulate LEACH++ using NS-2.34 in Fedora 12. The simulation parameters used to carry out the tests are mentioned in Table 7. Simulation results are discussed below.

There are 100 sensor nodes and a BS while the initial energy of the sensor nodes is set to 2 J. The sensor network topology used for the simulation is available at [16], which is shown in Fig. 10. The BS is located at (70,100) colored as metallic black with 5 nodes randomly selected as CHs, which are colored in green. Here, we have kept 5 CHs as it is considered as an optimal number [39]. If the CH of high density is compromised, then the sensor network will be affected more than compromising any other CH.

Clusters shown in Fig. 10 are almost uniform as CHs are at a good distance from each other and close to 20% of nodes in each case. While in Fig. 11, CHs are very close to each other, which may cause heterogeneous clusters. In such a case, if a cluster having more density gets compromised, then the effect

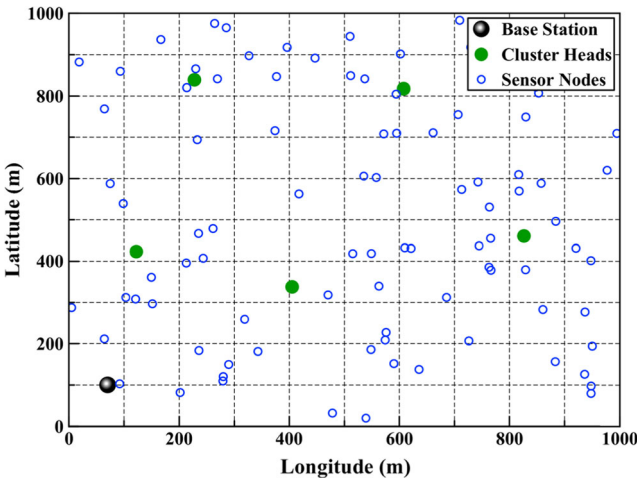

Fig. 10 Sensor network having 101 nodes with 5 CHs and 1 BS

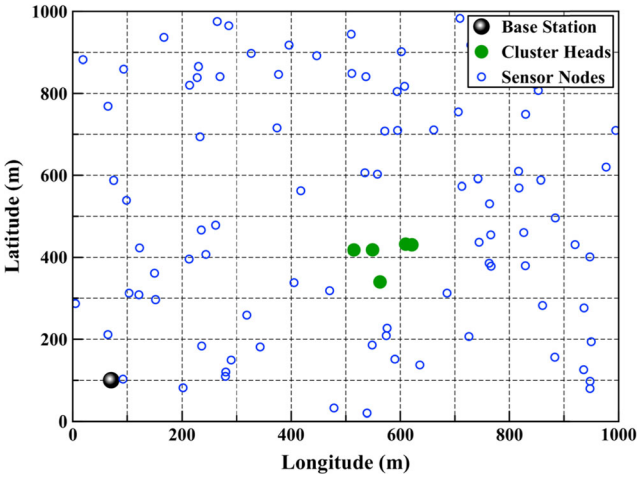

Fig. 11 Varying cluster size as LEACH supports heterogeneous cluster formation

is higher than the one having low cluster size. That is why we have simulated for more than 10 times for each scenario to cover this impact.

LEACH protocol supports heterogeneous clusters, and this may lead to the formation of worse clusters as shown in Fig. 11. Here, the five adjacent nodes are selected as CHs. So, the nodes may have a very minute difference among the distance from the CHs. This is the reason why we have performed different numbers of simulations and used the average value in our results. In our simulations, for attacked LEACH and LEACH++, we have changed the attacker node in each simulation to have a variety in the affected percentage. This can also be ensured automatically as CHs do not remain the same throughout the simulation.

5.3.1 Energy consumption in rounds

Energy utilization is a key feature to evaluate the performance of the routing protocols. LEACH utilizes less energy during its working but lacks the security aspects. LEACH++ adds some overhead to the normal LEACH as it includes security mechanism for reliable delivery of messages to the destination, as shown in Fig. 12.

Here, the sensor node consumes some energy for some computation steps and for the communication after detecting the malicious node. Each node consumes some extra energy due to the security policy during the setup phase in each round. This energy relates to the computation overhead enforced by the IDF on the LEACH protocol. However, this overhead is negligible as compared to the communication overhead, as shown in Fig. 12.

5.3.2 Data delivery in rounds

We have analyzed the data delivery at the BS that occurred during the normal LEACH, attacked LEACH, and

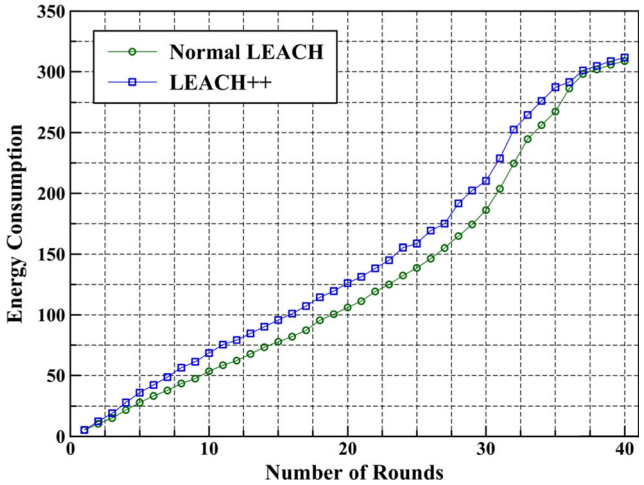

Fig. 12 Comparison between LEACH++ and normal LEACH with respect to energy utilization

LEACH++. The amount of data received at the BS depends on the data messages sent by the CHs. In the attack scenario, CH drops or selectively forwards the data communicated by the cluster nodes. In our simulation, there is a CH that drops all the packets. Hence, if there are 5 CHs in each round and one CH drops all the packets, then there will be about 20% reduction in the packets received at the BS as compared to the normal scenario. However, LEACH works in a random environment, so the number of messages sent in each round is not fixed for every simulation. So, the drop ratio calculated varies for each round for different simulations. We use the average of these simulations and show it in the following results in Fig. 13.

Sensor nodes will detect the malicious activity in the next round as per the detection policy set for the LEACH protocol, while the detection of the cluster node takes place in the current round by the CH. Here, the cluster nodes do not select a node as CH, which is selected by it in previous rounds, or the one who advertises CH JOIN request in every round. Sensor network remains affected for first two rounds and recovers from it in next rounds by avoiding the malicious node from becoming the CH. The impact of data received at the BS is shown in Fig. 13. Results show that about 22% of the data is saved by the LEACH++ protocol, which is compromised during the attacked scenario.

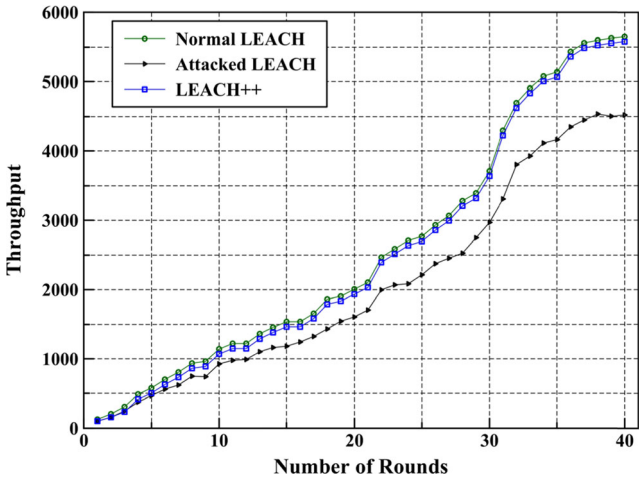

Fig. 13 Throughput comparison of LEACH, attacked LEACH, and LEACH++

6 Conclusion

Wireless sensor networks (WSNs), intelligent transportation systems (ITS), healthcare institutes, or smart homes that are integrated with the cloud computing environment require reliable wireless communication to ensure accurate results and better performance. Reliable delivery of data is the key requirement for applications that take sensor network data to the cloud for analysis or other user-centric services. In this paper, we modified the low-energy adaptive clustering hierarchy (LEACH) protocol for WSNs and added the functionality of intrusion detection to secure WSNs from sinkhole, black-hole, and selective forwarding attacks. We analyzed the energy consumption and throughput of the proposed LEACH++ protocol both numerically as well as using the NS-2 simulator. The results show that LEACH++ receives more throughput than LEACH during attack, while it does not affect the overall energy consumption of the system. The proposed intrusion detection framework (IDF) is lightweight and does not put much burden on the LEACH protocol with respect to memory utilization and computation. The proposed protocol can be effectively used for cloud-based WSN environments.

Acknowledgements The authors would like to extend their sincere appreciation to the Deanship of Scientific Research at King Saud University for its funding of this research through the Research Group Project no. RGP-214. The authors would also like to thank the Higher Education Commission (HEC), Pakistan, for its support through the indigenous PhD fellowship program.

References

1. Abdullah M, Alsanee E, Alseheymi N (2014) Energy efficient cluster-based intrusion detection system for wireless sensor networks. Int J Adv Comput Sci Appl 5(9):10–15
2. Akkaya, K., & Younis, M. (2005). A survey on routing protocols for wireless sensor networks. Elsevier Ad Hoc Networks , 325–349.
3. Al-Shurman, M., Yoo, S. M., & Park, S. (2004). Black hole attack in mobile ad hoc networks. 42nd Annual Southeast Regional Conference ACM-SE, (pp. 96–97).
4. Biren G. (2012). Cloud: An enabler for a smarter home. https://www.ibm.com/blogs/cloudcomputing/2012/03/cloud-an-enabler-for-a-smarter-home/
5. Bojkovic ZS, Bakmaz BM, Bakmaz MR (2008) Security issues in wireless sensor networks. International Journal of Communications II(1):106–115.
6. Doukas, C. & Maglogiannis, I. (2011). Managing wearable sensor data through cloud computing. Third IEEE International

Conference on Cloud Computing Technology and Science, (pp. 440–445).
7. Camara C, Peris-Lopez P, Tapiador J (2015) Security and privacy issues in implantable medical devices. Journal of Biomedical Informatics 55:272–289.
8. Chen, S., Yang, G., & Chen, S. (2010). A security routing mechanism against Sybil attack for wireless sensor networks. International Conference on Communications and Mobile Computing, (pp. 142–146).
9. Da Silva AP, Martins MH, Rocha BP, Loureiro AA, Ruiz LB, Wong WC (2005) Decentralized intrusion detection in wireless sensor networks. Proceedings of the 1st ACM international workshop on quality of service & security in wireless and mobile networks. ACM, Quebec, pp 16–23
10. Deng H, Li W, Agrawal DP (2002) Routing security in ad hoc networks. IEEE Communications Magazine, Special Topics on security in Telecommunication Networks 40(10):70–75
11. Farooqi, A. H., & Khan, F. A. (2012). A survey of intrusion detection systems for wireless sensor networks. International Journal of Ad Hoc and Ubiquitous Computing, 9(2):69–83.
12. Farooqi, A. H., & Khan, F. A. (2009). "Intrusion Detection Systems for Wireless Sensor Networks: A Survey. International Conference on Future Generation Communication and Networking, Jeju, South Korea. 234–241.
13. Farooqi AH, Khan FA, Wang J, Lee S (2013) A novel intrusion detection framework for wireless sensor networks. Pers Ubiquit Comput 17(5):907–919
14. Ferreira, A. C., Vilaça, M. A., Oliveira, L. B., Habib, E., Wong, H. C., & Loureiro, A. A. (2005). On the security of cluster-based communication protocols for wireless sensor networks. 4th IEEE International Conference on Networking, (pp. 449–458).
15. Hayajneh T, Mohd BJ, Imran M, Almashaqbeh G, Vasilakos AV (March 2016) Secure authentication for remote patient monitoring with wireless medical sensor networks. Sensors 16(424):1–25
16. Heinzelman, W. (2011). From Wendi Heinzelman Home page: http://www.ece.rochester.edu/~wheinzel/research.html#code
17. Heinzelman WB, Chandrakasan AP, Balakrishnan H (2002) An application-specific protocol architecture for wireless microsensor networks. IEEE Trans Wirel Commun 1(4):660–670
18. Heinzelman, W. B., Chandrakasan, A. P., & Balakrishnan, H. (2000). Energy-efficient routing protocols for wireless microsensor networks. 33rd Hawaii International Conference System Sciences. Maui, HI.
19. Herbert D, Sundaram V, Lu YH, Bagchi S, Li Z (2007) Adaptive correctness monitoring for wireless sensor networks using hierarchical distributed run-time invariant checking. ACM Transactions on Autonomous and Adaptive Systems 2(3):1–23
20. Hsieh MY, Huang YM, Chao HC (2007) Adaptive security design with malicious node detection in cluster-based sensor networks. Comput Commun 30(11–12):2385–2400
21. Karlof, C., & Wagner, D. (2003). Secure routing in wireless sensor networks: attacks and countermeasures. The first IEEE International Workshop on Sensor Network Protocols and Applications (pp. 113–127). IEEE.
22. Krontiris I, Dimitriou T (2007) Towards intrusion detection in wireless sensor networks. In: 13th European Wireless Conference. ENSTA and SEE, Paris
23. Krontiris I, Dimitriou T, Giannetsos T (2008a) LIDeA: a distributed lightweight intrusion detection architecture for sensor networks. ACM Secure Communication. Fourth International Conference on Security and Privacy for Communication Networks, Istanbul, Turkey
24. Krontiris, I., Giannetsos, T., & Dimitriou, T. (2008b). Launching a sinkhole attack in wireless sensor networks: the intruder side. International Conference on Wireless and Mobile Computing Networking and Communications, (pp. 526–531).
25. Lee S, Lee Y, Yoo SG (2012) A specification based intrusion detection mechanism for LEACH protocol. Inf Technol J 11(1):40–48
26. Lounis A, Hadjidj A, Bouabdallah A, Challal Y (2015) Healing on the cloud: secure cloud architecture for medical wireless sensor networks. Futur Gener Comput Syst 55:266–277
27. Marchang, N., & Datta, R. (2008). Collaborative techniques for intrusion detection in mobile ad-hoc networks. Ad Hoc Networks, 6(4): 508–523
28. Masdari M, Bazarchi SM, Bidaki M (2013) Analysis of secure LEACH-based clustering protocols in wireless sensor networks. J Netw Comput Appl 36(2013):1243–1260
29. Sultan, N. (2014). Making use of cloud computing for healthcare provision: Opportunities and challenges. International Journal of Information Management 34(2): 177–184
30. Rolim, C. O., Koch, F. L., Westphall, C. B., Werner, J., Fracalossi, A. & Salvador, G. S. (2010). A cloud computing solution for patient's data collection in health care institutions. Second IEEE International Conference on eHealth, Telemedicine, and Social Medicine, (pp. 95–99)
31. Zhang, R. & Liu, L. (2010). Security models and requirements for healthcare application clouds. 3rd IEEE International Conference on Cloud Computing, (pp. 268–275), Miami, FL.
32. Rahayu, T. M., Lee, S.-G., & Lee, H.-J. (2014, February). Survey on LEACH-based security protocols. 16th International Conference on Advanced Communication Technology, 304–309.
33. Rajasegarar S, Leckie C, Palansiwami M (2008) Anomaly detection in wireless sensor networks. IEEE Wirel Commun 15(4):34–40
34. Roman, R., Zhou, J., & Lopez, J. (2006). Applying intrusion detection systems to wireless sensor networks. 3rd IEEE Consumer Communications and Networking Conference (pp. 640–644). IEEE Communications Society.
35. Roosta, T., Shieh, S., & Sastry, S. (2006). Taxonomy of security attacks in sensor networks and countermeasures. First International Conference on System Integration and Reliability Improvements. Hanoi, Vietnam: IEEE.
36. Su, C. C., Chang, K. M., Kuo, Y. H., & Horng, M. F. (2005). The new intrusion prevention and detection approaches for clustering-based sensor networks. IEEE Wireless Communications and Networking Conference, (pp. 1927–1932).
37. Sundararajan RK, Arumugam U (2015) Intrusion detection algorithm for mitigating sinkhole attack on LEACH protocol in wireless sensor networks. Journal of Sensors 2015:1–12
38. Teodoroa PG, Verdejo JD, Fernandez GM, Vazquez E (2009) Anomaly-based network intrusion detection: techniques, systems and challenges. Computers and Security 28:18–28
39. Tian, L., ChangDu, H., & Huang, Y. (2012, June). The simulation and analysis of LEACH protocol for wireless sensor network protocol for wireless sensor network based on NS2. International Conference on System Science and Engineering, 530–533.
40. Wood, A. D., & Stankovic, J. A. (2002). Denial of service in sensor networks. Computer (pp. 54–62). IEEE Computer Society.
41. Wu, D., Hu, G., & Ni, G. (2008). Research and improve on secure routing protocols in wireless sensor networks. 4th IEEE International Conference on Circuits and Systems for Communications (pp. 853–856).
42. Wang, X. & Tan, Y. (2010). Application of cloud computing in the health information system. International Conference on Computer Application and System Modeling, (pp. 179–182).
43. Xie M, Han S, Tian B, Parvin S (2011) Anomaly detection in wireless sensor networks: a survey. J Netw Comput Appl 34: 1302–1325
44. Zhang, K., Wang, C., & Wang C. (2008). A secure routing protocol for cluster-based wireless sensor networks using group key management.

4th IEEE International conference on Wireless Communications, Networking and Mobile Computing (pp. 1–5).

45. Zhang Y, Lee W, Huang YA (2003) Intrusion detection techniques for mobile wireless networks. Wirel Netw 9(5):545–556

46. Zhou, J., Cao, Z., Dong, X., & Lin, X. (2015). Security and privacy in cloud-assisted wireless wearable communications: challenges, solutions and, future directions. IEEE Wireless Communications, 136–144.