CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2010; **00**:1–15
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

PAWN: A Payload-based mutual Authentication scheme for Wireless Sensor Networks

Mian Ahmad Jan[1]*, Priyadarsi Nanda[2]*, Muhammad Usman[2], Xiangjian He[2]*

[1]*Department of Computer Science,*
Abdul Wali Khan University Mardan, Pakistan
[2]*School of Computing and Communications, Faculty of Engineering and Information Technology,*
University of Technology Sydney, Australia

SUMMARY

Wireless Sensor Networks (WSNs) consist of resource-starving miniature sensor nodes deployed in a remote and hostile environment. These networks operate on small batteries for days, months and even years depending on the requirements of monitored applications. The battery-powered operation and inaccessible human-terrains make it practically infeasible to recharge the nodes unless some energy scavenging techniques are employed. These networks experience threats at various layers and as such, are vulnerable to a wide range of attacks. The resource-constrained nature of sensor nodes, in-accessible human terrains and error-prone communication links make it obligatory to design lightweight but robust and secured schemes for these networks. In view of these limitations, we aim to design an extremely lightweight payload-based mutual authentication scheme for a cluster-based hierarchical WSN. The proposed scheme, also known as PAWN, operates in two steps. First, an optimal percentage of cluster heads is elected, authenticated and allowed to communicate with neighbouring nodes. Second, each cluster head, in a role of server, authenticates the nearby nodes for cluster formation. We validate our proposed scheme using various simulation metrics which outperforms the existing schemes.
Copyright © 2010 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: Cluster Head, Cluster, Authentication, Lightweight, Payload-based, Client-Server Model

1. INTRODUCTION

Wireless Sensor Network (WSN) is a collection of miniature sensor nodes deployed to monitor, sense, capture and process the data about an application, i.e., phenomena of interest [1]. These nodes are resource-starving and as such are highly constrained on battery power, storage, computation, data rate and available bandwidth. Typically, they are deployed and left unattended in a remote and human-inaccessible terrain to perform monitoring and reporting tasks. As a result, the limited resources of these nodes need to be utilized efficiently to prolong the network lifetime. Due to their unique characteristics of self-healing and fault-tolarance, these networks have found their applications in various domains such as military surveillance, health care, industrial automation, home automation, agriculture, and environmental monitoring [2].

*Correspondence to: [1]Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan.[2]School of Computing and Communications, University of Technology, Sydney, Australia.
E-mail: [1]Mianjan@awkum.edu.pk
E-mail: [2]Xiangjian.He@uts.edu.au
E-mail: [2]Priyadarsi.Nanda@uts.edu.au

Applications deployed in hostile environment with little human interventions to manage them expose these networks to various threats and vulnerabilities. These networks are susceptible to not only physical tampering but also various remote attacks such as sink hole, worm hole, Sybil, replications, homing and distributed denial-of-service flooding [3][4]. As a result, the nodes in general and the network in particular need to be protected against these threats. Lightweight but highly efficient and robust authentication schemes need to be designed for these networks. The resource-constrained nature of the nodes limits the support and application of resource-intensive, highly robust and sophisticated authentication and privacy schemes to these networks. The existing security schemes require ample of resources in terms of computation, transmission, storage and available bandwidth. Their application to WSNs deteriorates the network performance and rapidly depletes the network lifetime. As a result, designing lightweight but highly intelligent and robust security solutions is a challenging task in these networks. The provisioning of security, privacy and trust need to be kept in mind while the prototype of a routing protocol is designed. Furthermore, the underlying operations of different routing protocols also influence the energy consumption which is linked directly with the application of any secured scheme. Among all the routing protocols, cluster-based hierarchical routing protocols efficiently utilize the limited resources of sensor nodes. These protocols are highly efficient in terms of data aggregation, energy consumption, collision avoidance, load balancing, fault-tolerance and network lifetime [5]. Unlike other protocols, a single cluster head collects data from multiple nodes in order to eliminate redundancy and improves the quality of aggregated data. These protocols use intra-cluster and inter-cluster communication to reduce long-haul transmissions to a base station.

Cluster-based hierarchical routing protocols perform only intelligent partitioning of the network to provide energy-efficient communication. The underlying network still suffers from various threats posed during set-up and steady-state phases. During set-up phase, an adversary may declare itself as a cluster head by broadcasting advertisement messages with much stronger signal strengths. It is also possible that an adversary may participate in cluster formation or prevent legitimate nodes from joining a cluster head. An attacker may eavesdrop on advertisement packets and/or join-request packets and replay in other parts of the network. Furthermore, the same attacker has the ability to maliciously manipulate the contents of these packets. An adversary may disrupt the ongoing operations in one or more clusters by constantly emitting jamming signals or launching Denial-of-Service (DoS) attack [6]. An adversary may deceive a cluster head by persuading it to create a false member list for its members and as a result, a wrong Time Division Multiple Access (TDMA) schedule is created for its members [7]. An adversary may launch a more sophisticated attack by colluding with other adversaries in their neighbourhood [8]. In [9], the authors analyzed the behaviour of various attacks such as, continuous election as cluster head, TDMA schedule obedience, aborting and/or no transmission of member/cluster head data and transmission of powerful signals. In [10], the authors investigated the behaviour of Sybil nodes in cluster-based hierarchical routing protocols. They studied the impact and consequences of Sybil nodes performing the role of cluster heads in these networks. A single Sybil node has the ability to breakdown the whole communication in a cluster-based WSN provided that it is elected as a cluster head.

In this paper, we present a payload-based mutual authentication algorithm, also known as PAWN, which uses a centralized cluster-based hierarchical WSN. The major contributions of our research are as follows.

1. The existing centralized cluster-based hierarchical protocols do not guarantee an optimal percentage of cluster heads. However, our proposed algorithm always elects an optimal percentage of such nodes so that it enhances the lifetime of the network and reduces the number of server nodes required for communication.
2. PAWN only initiate data communication when the base station, cluster heads and neighbouring nodes are authenticated. It uses extremely lightweight nomination packets and advertisement messages to perform base station/cluster head and cluster head/neighbouring nodes authentication.
3. A lightweight version of AES algorithm is used and the payload of each authentication message does not exceed 256 bits.

The rest of the paper is organized as follows. In Section II, we present the related works from literature. In Section III, we present our PAWN algorithm in terms of cluster head selection and cluster formation. In Section IV, experimental results of our proposed scheme are presented. Finally, the paper is concluded and future research directions are provided in Section V.

2. RELATED WORK

In this section, we provide an overview of cluster-based hierarchical routing protocols as they are used as the underlying platform by our proposed scheme. Next, various lightweight mutual authentication and key agreement schemes are discussed in the context of WSNs.

Low-Energy Adaptive Clustering Hierarchy (LEACH) [11] was designated as a pioneer protocol among clustering-based hierarchical routing protocols. LEACH partitions a sensor field into small geographical regions known as clusters. Each cluster has a cluster head node which collects and aggregates data from member nodes and transmits them to a base station. The protocol operates in rounds and nodes take turn to become cluster heads in subsequent rounds for uniform distribution of energy load. The problem with LEACH protocol is the probabilistic selection of cluster heads using random number generation. Each node, *i*, chooses a random number between 0 and 1. If the chosen number is less than the threshold value *T(i)*, the node is elected as a cluster head for the current round. The operation of LEACH for cluster head selection is shown in Equation 1.

$$T(i) = \begin{cases} \frac{k_{opt}}{1-k_{opt}(r mod(\frac{1}{k_{opt}}))}, & if\ i \in G, \\ 0, & otherwise. \end{cases} \tag{1}$$

Here, k_{opt} is the optimal percentage of cluster heads in each round, r is the current round and G is the set of nodes that have not been elected as cluster heads in the past $\frac{1}{k_{opt}}$ rounds. The probabilistic selection of cluster heads has a potential risk of low energy nodes being elected as cluster heads in subsequent rounds. Moreover, Equation 1 cannot guarantee an optimal number of cluster heads in each round. In [12], the authors argued that cluster heads need to be elected based on the residual energy of the nodes. They suggested the inclusion of residual energy of nodes in Equation 1. However, it does not solve the problem because cluster heads are still elected using a random number generation. To solve this problem, nodes need to be elected by a central controller, i.e., a base station. In [13], the authors proposed a centralized approach for cluster head selection. Nodes having remaining energy greater than the average residual energy are elected as cluster heads in each round. However, it is highly probable that there are a large number of such nodes in each round resulting in too many cluster heads. In [14], we proposed a centralized scheme which elected an optimal percentage of cluster heads (5% of total nodes). Each round results in balanced clusters that enhance network stability, scalability and data aggregation. Moreover, the proposed approach reduces network load, energy consumption and congestion. Irrespective of a centralized or randomly distributed cluster-based approach, each round consists of a set-up phase and a steady-state phase. During the set-up phase, cluster heads are elected, cluster are formed and schedules are created. In the steady-state phase, each cluster head collects data from its member nodes and transmits them to a centralized base station.

For secured transmission of data in cluster-based hierarchical networks, there exist various secured versions of LEACH in literature [15] [16][17][18]. Most of these protocols used a one-way key chain[19] for encryption and decryption purposes. The chain is generated using a one-way hash function $\mathbb{H}$ on the last key in each chain. In [20], a survey was conducted to analyze and evaluate the performances of various secured cluster-based hierarchical schemes. Most of these schemes increase the security of set-up and steady-state phases in particular and the overall operation of the network in general. In these schemes, LEACH-alike cluster hierarchies are protected against various attacks launched by adversaries from outside the network. These schemes make it hard to compromise various operations performed within the set-up and steady-state phases by deploying various key distribution mechanisms. The application of asymmetric encryption to miniature sensor nodes exhausts their resources quickly and rapidly reduce the lifetime of WSNs. As a result, most of the studies focus on symmetric encryption for security and privacy provisioning in cluster-based WSNs. In [21], SecLEACH protocol was proposed for securing cluster-based hierarchical WSNs. Prior to network deployment, SecLEACH generates a large pool of keys using a random key pre-distribution approach. Each node is allocated a small subset of these keys, also known as key ring, drawn pseudo randomly from the pool. Furthermore, each node is assigned a pairwise key which is shared with the base station and a group key common to each member of the network. Upon network deployment, each cluster head advertises itself by broadcasting a message containing information about the keys in its pre-defined key ring. In SecLEACH, the base station is responsible for authenticating the advertisement messages as it is a resource-rich entity and is highly trusted. Upon authentication by the base station, each neighbouring node forms a cluster around its nearest cluster head with whom they share the group key. Cryptographic-based LEACH variants are robust only against external attacks, however, they are not capable to tackle threats posed by internal adversaries, e.g., a compromised sensor. This is due to the fact that compromised sensors have already been allocated authentication keys and they have the same privileges which are required for transmission of authenticated information. As a result, conventional authentication algorithms fail to detect such an attack [22]. A centralized secure LEACH (CSLEACH) was proposed in [23]. Each neighbouring node and its gateway share a unique private key which is used for authentication of broadcasted advertisement messages. Each node possesses two keys, i.e., a gateway private key and a neighbor private key. A node waits for a message upon entering the network at the start of each round. A particular round is initiated by the gateway using a Round Start Message (RSM). These messages are mainly used for synchronization purposes. These messages distribute a network key and a session template for generation of Medium Access Control keys (KMAC) and session keys. Any communication between a neighbouring node and its gateway is encrypted by the session key. These keys also encrypt the communication between a cluster head and its member nodes. KMAC, on the other hand, encodes MAC for providing integrity in the network.

3. PAWN: PAYLOAD-BASED AUTHENTICATION FOR WSN

Our proposed scheme mainly consists of two important steps and are presented as follows.

1. An optimal percentage of cluster heads are elected and advertised to the network using a token-based authentication approach. The token is used for correlating cluster head advertisement messages with the corresponding acknowledgment messages.
2. A payload-based mutual authentication is used for cluster formation between neighbouring nodes and their nearest cluster head.

Both of these steps are part of the set-up phase during which cluster heads are elected and clusters are formed around them. In this section, first we present the token-based optimal election of cluster heads followed by a payload-based mutual authentication for a cluster-based hierarchical WSN. An optimal election of cluster heads enables us to nominate very few nodes as servers. The unelected nodes become the potential clients of cluster heads in the network.

3.1. Token-based Cluster Head Election

Our proposed network model consists of randomly deployed heterogeneous sensor nodes. There are 100 normal nodes and 5 higher-energy nodes in the network. To reduce the operational cost, normal nodes are equipped with initial energy of 1 joule each, and higher-energy nodes are equipped with 5 joules each. All nodes are deployed in a 100×100 sq. meter geographical region. The role of each node is specified as follows.

1. Only normal nodes can be elected as cluster heads.
2. Only normal nodes are eligible to participate in a client-server interaction model.
3. Higher-energy nodes assist the base station in cluster head election.

4. Higher-energy nodes exchange control packets with the base station while normal nodes transmit both data and control packets.

Upon network deployment, each normal node is assigned a token and a pre-shared secret (λ_i). Token is used for secure exchange of nomination packets and *ACKs* between a base station and a cluster head while λ_i is used for in-network mutual authentication between a neighbouring node and its prospective cluster head. Higher-energy nodes perform various administrative tasks such as route discovery, route maintenance and neighbourhood discovery. Each higher-energy node collects and disseminates control packets during upstream and downstream traffic.

Initially, each normal node, *i*, creates and broadcasts a control packet to its nearest higher-energy node as shown in Figure 1. This packet contains the residual energy (E_i) and identity (ID_i) of *i*. The location of a nearest higher-energy node with respect to *i* is computed by solving Euclidean distance ($d_{\mathbb{H}}$) of Equation 2.

$$d_{\mathbb{H}} = \sqrt{(x_{\mathbb{H}} - x_i)^2 - (y_{\mathbb{H}} - y_i)^2}. \tag{2}$$

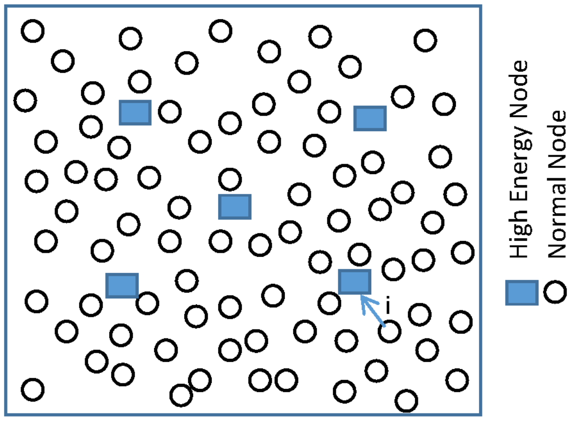


Figure 1. Network at the Time of Deployment

Each higher-energy node collects these packets in their neighbourhood, retrieves E_i and ID_i, and broadcasts a single control packet containing these values to a base station. After transmission of the packet to a base station, the higher-energy nodes go to sleep mode and will awake at the start of next round to initiate the aforementioned tasks. Upon reception, the base station retrieves E_i and ID_i from each control packet. At this stage, an average energy threshold is computed using Equation 3.

$$E_{avg} = \sum_{i=1}^{N} \frac{E_i}{N}. \tag{3}$$

Here, E_{avg} is the average energy threshold for a total of N normal nodes, where N is equal to 100.

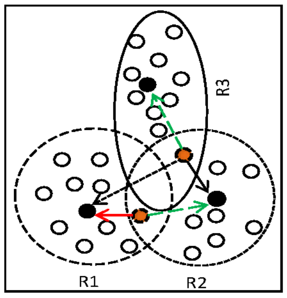
Any normal node *i* having E_i greater than or equal to E_{avg} in a particular round is eligible for cluster head selection, $\forall\ i \in \{1, 2, \cdots, N\}$. The only exception is the initial round immediately after network deployment, where all the nodes have nearly the same residual energy values, i.e., $E_i \approx E_{avg}$. All such nodes are eligible for cluster head selection because $\Delta E_{a,b} \approx 0$, where $\Delta E_{a,b}$ = E_a - E_b, $\forall\ a, b \in \{1, 2, , \cdots, N\}$. In our proposed scheme, base station has the ability to elect cluster heads based on extremely low values of ΔE, up to 4 decimal digits.

In cluster-based hierarchical routing protocols, it is highly probable that there may be a large number of nodes for whom $E_i \geq E_{avg}$ in various rounds. All such nodes are candidates, also known as nominees, for cluster heads. The only exception is towards the end of network lifetime where nominees are either equal to or less than k_{opt}. The base station is responsible to elect k_{opt} cluster heads among the nominees. We use the following criteria for an optimal election.

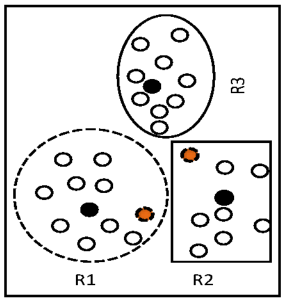
- E_i of a nominee, i, is greater than or equal to E_{avg}.
- i is not elected as cluster head during the past $\frac{1}{p}$ rounds.
- Two or more nominees located in a same geographical region are evaluated based on their energy values and previous history of election.

In our proposed scheme, k_{opt} is 5% for a network of N nodes. An optimal percentage of cluster heads influences the performance of cluster-based hierarchical networks. Energy of a cluster head is depleted rapidly due to resource-intensive tasks such as data aggregation, fusion and long-haul transmission to a base station. A small value of k_{opt} will deteriorate the performance of cluster-based hierarchical network because each cluster head will accommodate a large set of neighbouring nodes in its cluster. In this case, energy of each cluster head will be exhausted rapidly in the aforementioned resource-intensive operations. A large value of k_{opt} will make a cluster-based hierarchical network rather inefficient and ineffective due to low data aggregation and fusion [10].

In each round, the base station broadcasts nomination packets to the elected cluster heads containing their identities (forming the set ID_{CH}) and identities (forming the set ID_{NB}) of their neighbouring nodes. Both these types of identities are transmitted within the payload of nomination packets. Recall that the base station knows the identity of each node and its geographical location. To ensure secured transmission of ID_{CH} and ID_{NB}, each nomination packet has a 16-bit token in its header. The base station generates a token similar to the one possessed by an elected cluster head, i.e., one token per cluster head is generated. The number of tokens generated depends on the number of elected cluster heads. Upon reception, each cluster head, denoted by S, examines the appended token within the nomination packet. If it matches with the one possessed by it, the cluster head retrieves the appended identities ($ID_{\overline{\mathbb{NB}}}$) of its neighbouring nodes, where $ID_{\overline{\mathbb{NB}}} \subset ID_{NB}$. Each cluster head must acknowledge the receipt of a nomination packet. It creates an acknowledgment message (ACK), appends its token to the ACK and transmits the ACK to the base station. An adversary may intercept one or more nomination packets, however, it cannot reproduce the token required to retrieve $ID_{\overline{\mathbb{NB}}}$. An adversary would require 2^{16} attempts to retrieves $ID_{\overline{\mathbb{NB}}}$ from a nomination packet. Upon election and successful exchange of nomination packet/ACK, each cluster head advertises itself to the nearest neighbours. A neighbouring node may receive advertisement messages from multiple cluster heads, however, it associates itself with the one having the strongest signal strength among all. The neighbouring nodes in overlapping regions receive signals from multiple cluster heads as shown in Figure 2(a). In this figure, the circles and ovals do not represent clusters. Instead, they represent the regions where the signal strength of each cluster head, identified with black colour nodes, is detected. The signal strength experienced by a node is represented by an arrow, where a solid arrow represents a strong signal, dashed arrow represents a medium signal and dotted arrow represents a weak signal. Each neighbouring node pursues its connection with a cluster head having the strongest signal as shown in Figure 2(b). In this figure, the oval, circle and rectangle indicate the intention of each node for joining a prospective cluster head. Cluster formation is yet to take place at this point of time.



(a) Overlapping Regions



(b) RSS-based Clustering

Figure 2. Initiating Cluster Formation

Each neighbour node sends a join-request message to a cluster head having the strongest of signal strength for cluster formation. However, cluster formation is not a straight forward process. A neighbour needs to authenticate itself before association with a cluster head. Furthermore, each neighbour is not sure about the authenticity of a cluster head. As a result, not only the neighbours, but also the cluster head needs to authenticate themselves. To do so, each neighbour transmits a join-request message to a prospective cluster head as discussed in the next section.

3.2. Payload-based Mutual Authentication

Upon successful reception of advertisement messages, each neighbouring node initiates cluster formation. The aim of cluster head selection was to elect a feasible number of servers for neighbouring nodes. On the other hand, the aim of cluster formation is to allow authentic data transmission to a base station via the elected servers. The authentication process is accomplished using a 128-bit AES encryption algorithm. This mode of encryption requires less resources and is extremely beneficial for time-critical, delay-sensitive data generated by the nodes. Furthermore, it allows enough time for the nodes to offload their data before an attack is launched. AES-192 and AES-256, on the other hand, are highly computational, resource-consuming and require sophisticated hardware and software platforms. Such requirements are not fulfilled by most of the resource-constrained sensor nodes.

To meet the requirements of resource-starving miniature sensor nodes, we use a lightweight payload-based mutual authentication technique. In our proposed scheme, authentication is performed within the payload of each message. Each neighbouring node exchanges multiple handshake messages with a server as shown in Figure 3. The authentication process is completed in four extremely light-weighted handshake messages. Once authenticated, each neighbouring node is allowed to transmit its data to its respective cluster head. The authentication procedure is accomplished using the following four simple phases.

1. Session Initiation
2. Server Challenge
3. Client Response
4. Server Response

During session initiation, each neighbour, i, creates a join-request message which contains ID_i and ID_{CH_S}. Here, ID_i is the identity of the source node and ID_{CH_S} is the identity of the destination node of cluster head, S. Both the source and destination IDs are of 8-bits. This number of bits results in sufficiently light-weighted join-request messages. Irrespective of a legitimate or a malicious node, each neighbour can negotiate a maximum of four session initiation requests with a server. After these attempts, any further requests from the given neighbour are discarded and the server refrains from any further communication with it. During session initiation phase, ID_i is transmitted within the payload and ID_{CH_S} is transmitted within the header of each join-request message. The payload is followed by an optional Frame Check Sequence (FCS), which is appended as a trailer and is used for error detection and correction.

In the server challenge phase, each server retrieves ID_{CH_S} from the header and ID_i from the payload of a received join-request message. If ID_{CH_S} matches the server ID, it means that the join-request message was indeed intended for it. If a match is not found, the join-request message is discarded. To negotiate a session, ID_i must match with an identity within the pool of identities provided to a given server. A match can only be found if $ID_i \in ID_{\overline{\mathbb{NB}}}$. Each server retrieves ID_i from the payload and searches for a matching λ_i. Each identity ID_i within $ID_{\overline{\mathbb{NB}}}$ is associated with a λ_i for authentication purpose. If a match is found, the server responds back with an encrypted payload using an AES algorithm. At this point, the server creates a challenge for the client *i*. For successful authentication, this challenge needs to be deciphered by *i*.

To create a challenge, the server generates a pseudo-random nonce (η_{server}) and a potential session key (μ_{key}) of 128-bit each. Nonce is a temporary number used only once by a client/server in the entire cryptographic communication. Using μ_{key} and η_{server}, an encrypted payload is generated. First, an XOR operation is performed on λ_i and μ_{key} using Equation 4. This operation

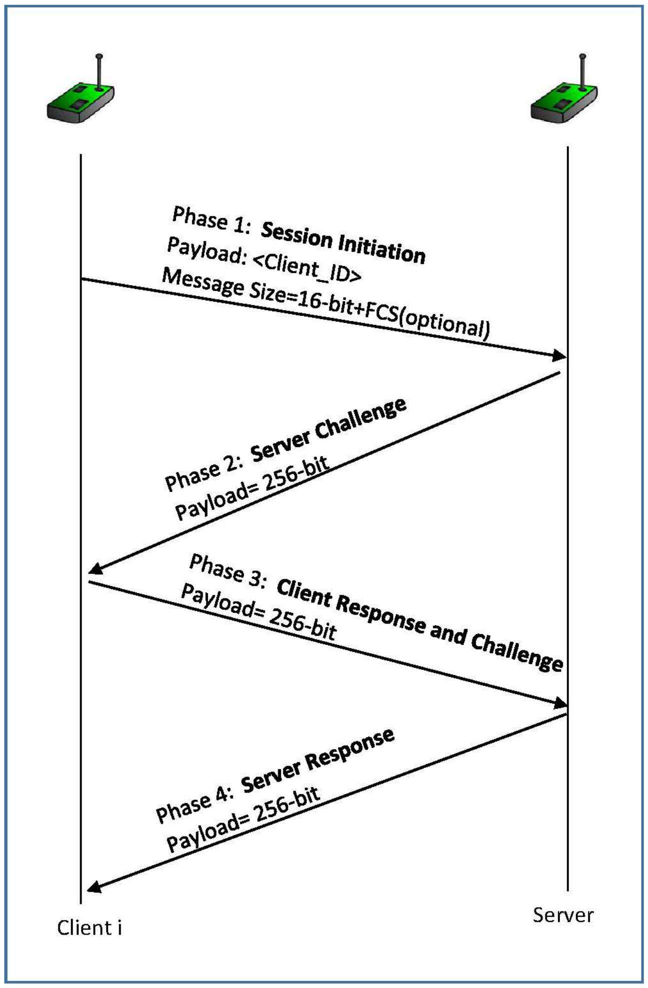

Phase 1: Session Initiation
Payload: <Client_ID>
Message Size=16-bit+FCS(optional)
Phase 2: Server Challenge
Payload= 256-bit
Phase 3: Client Response and Challenge
Payload= 256-bit
Phase 4: Server Response
Payload= 256-bit
Client i
Server

Figure 3. Payload-based Mutual Authentication

is computationally inexpensive and is extremely common as a component in complex ciphers. Moreover, this operation does not leak information about the original plaintext and applying it twice enables the retrieval of original plaintext. In our case, plaintext is the session key.

$$\psi_{resultant} = \lambda_i \oplus \mu_{key}. \tag{4}$$

The 128-bit $\psi_{resultant}$ is appended to η_{server} and encrypted with λ_i to generate a payload of 256-bit using Equation 5. This payload, i.e., $\gamma_{server\text{-}payload}$, is transmitted to the client as a challenge.

$$\gamma_{server\text{-}payload} = \{\lambda_i, (\psi_{resultant}|\eta_{server})\}AES128. \tag{5}$$

In the client response and challenge phase, the client needs to decipher the encrypted payload ($\gamma_{server\text{-}payload}$) to retrieve the potential session key μ_{key}. If the client is successful to do so, it will have the correct η_{server} and μ_{key}. The η_{server} and μ_{key} are known only to the server and λ_i belongs to a specific client. Only a legitimate client can decipher this challenge. An intruder can eavesdrop only on η_{server} and μ_{key}, but not on λ_i in accordance with the Internet Threat model [24]. The client uses its λ_i to decipher the payload. Upon successful deciphering, the client has authenticated itself. As mutual authentication requires both parties to be verified, the server also needs to authenticate itself. The client generates a new encrypted payload similar to the server. First, an XOR is performed on η_{server} and λ_i using Equation 6.

$$\psi_{resultant} = \eta_{server} \oplus \lambda_i. \tag{6}$$

The 128-bit $\psi_{resultant}$ is appended with η_{client} and encrypted with μ_{key} to generate an encrypted payload as shown in Equation 7. The 256-bit encrypted payload ($\gamma_{\textit{client-payload}}$) is transmitted to the server as a challenge.

$$\gamma_{\textit{client-payload}} = \{\mu_{key}, (\psi_{resultant}|\eta_{client})\}\textit{AES128}. \tag{7}$$

Finally, in the server response phase, the server deciphers the encrypted payload ($\gamma_{\textit{client-payload}}$) of the client challenge to observe η_{server} in it. If present, the server realizes that the client has successfully authenticated itself. The server retrieves η_{client} and creates an encrypted payload of its own by appending η_{client} to μ_{key} and encrypting with λ_i as shown in Equation 8. Next, the 256-bit encrypted payload ($\gamma_{\textit{server-payload}}$) is transmitted in response to the client challenge.

$$\gamma_{\textit{server-payload}} = \{\lambda_i, (\eta_{client}|\mu_{key})\}\textit{AES128}. \tag{8}$$

At this point of time, the status of ongoing session changes from *session negotiation* to *authenticated* at the server because client *i* is successfully authenticated and is eligible to transmit its data to the server. The client, however, has yet to verify the authenticity of the server. It decrypts the payload $\gamma_{\textit{server-payload}}$ and observe η_{client} in it. As the client is the one which generated η_{client}, the client comprehends that the response is from a legitimate server. Both the client and the server are mutually authenticated and they have agreed upon a common session key (μ_{key}) for exchanging the data packets. At this stage, both i and the server are mutually authenticated. The server creates a schedule for each member node i within its cluster. The server allocates TDMA slots to each member node within its cluster. These slots are used for contention-free communication and scheduling the duty-cycle of each node. The four phases of our payload-based mutual authentication handshaking are shown in Algorithm 1.

Once the client and server are authenticated, the set-up phase is completed and the steady-state phase is initiated. During this phase, each cluster head collects data from its member nodes, aggregates the data and transmits them to a base station located inside or outside the sensor field. In Figure 4, the complete set of operations performed by PAWN is shown.

4. EXPERIMENTAL RESULTS

In this section, we provide simulation results and experimental analysis of PAWN algorithm. For simulation, we set up our experiments using Matlab R2011a. PAWN network model consists of 100 normal nodes and 5 higher-energy nodes deployed in a 100×100 square meter area. The base station is located outside the sensor field and is a resource-rich entity.

4.1. *Security Analysis*

PAWN provides authenticity, confidentiality and freshness of data for node-to-node communication. In Table I, we compare PAWN against the existing schemes.

PAWN uses a two-step procedure for provisioning of authentication. First, a predefined procedure is used to elect the cluster heads. Upon election, a token-based technique is used for secured transmission of advertisement messages. Second, each cluster head, in a role of server, verifies the identities of potential clients, i.e., the neighbouring nodes. The payload-based authentication is used to validate the identities of nodes in the incoming packets. In comparison, LEACH-C does not provide any authentication as it only partitions the network into clusters. SecLEACH, on the other hand, uses Message Authentication Code (MAC) for authentication purpose. Each message is encrypted with a key using a key pool. After successful decryption of the message, the receiver knows that message is originated from a legitimate node in the sensor field.

PAWN uses a payload-based approach to provide message confidentiality. The payload of join-request messages and nomination packets are used for this purpose. LEACH-C, on the other hand, does not provide data confidentiality while SecLEACH uses the same MAC for message confidentiality. Freshness of messages is guaranteed by nonces in PAWN and SecLEACH. PAWN, as a mutual authentication scheme, includes both the client and server nonces because both parties

Algorithm 1 Payload-based Handshake Algorithm

```
1:  Initialization:
      1. Each neighbour, i, sends $ID_i$ to $S$, a prospective Server (Cluster Head).
      2. $S$ is provided with $ID_{\overline{\mathbb{NB}}}$ and $\lambda_i$
      3. Input: $\{ID_{\overline{\mathbb{NB}}}, \lambda_i\}$    ▷ $\lambda_i \in \{0, 1, \cdots, 2^{128}-1\}, \forall\, i \in \{1, 2, \cdots, N\}$
2:  for $i = 1 : N$ do    ▷ Nested For loop generates a two-column server table
3:      for $j = 1 : 2$ do
4:          input $A[i][j]$    ▷ $ID_i$ and $\lambda_i$ are stored in $A[i][0]$ and $A[i][1]$ respectively.
5:      end for
6:  end for
7:  Phase 1 [Session Initiation]: $i$-> $S$: {join-request with $ID_i$}
8:      ▷ $i$ sends a join-request message to $S$ containing $ID_i$ in the payload.
9:  Phase 2 [Server Challenge]: $S$ retrieves $ID_i$ to find a matching $\lambda_i$
10: if $ID_i$ == $A[i][0]$ then    ▷ A match is found
11:     Session negotiation is initiated between $i$ and $S$
12:     $S$ responds with an encrypted payload, $\{\lambda_i, (\lambda_i \oplus \mu_{key} | \eta_{server})\}AES128$
13:         ▷ where $\mu_{key}, \eta_{server} \in \{0, 1, \cdots, 2^{128}-1\}$
14: else
15:     $i$ is unauthorized and join-request is discarded
16: end if
17: Phase 3 [Client Response & Challenge]: $i$ deciphers challenge and responds with an encrypted
    payload, $\{\mu_{key}, (\eta_{server} \oplus \lambda_i | \eta_{client})\}AES128$,
18: Phase 4 [Server Response]: $S$ checks $\eta_{server}$ in the client challenge by comparing against the
    $\eta_{server}$ generated in phase 3
19: if Both matches then
20:     [$i$ becomes a member node of $S$]- Cluster is formed
21:     $S$ responds as $\{\lambda_i, (\eta_{client} | \mu_{key})\}AES128$
22: else
23:     [Session Negotiation Fails]-$i$ is unauthorized and barred from communication in cluster
        formation
24: end if
25: $i$ compares $\eta_{client}$ of phase 3 and 4.
26: if Both matches then
27:     $S$ becomes member node of $S$
28:     $S$ allocates TDMA slots to $i$
29:     [Data Exchange within a Cluster]:
30:         $i$->$S$: $\{d_i$, where $d_i$ represents data packets of $i\}$.
31:             ▷ $i$ waits for its turn and transmits data encryped with $\mu_{key}$ to $S$.
32: else
33:     $S$ is unauthorized
34: end if
```

Table I. Security Analysis

Security Services	LEACH-C	SecLEACH	PAWN
Authentication	No	MAC	Two-step
Confidentiality	No	MAC	Payload
Freshness	No	Nonces	Nonces

have to authenticate each other. Nonces are non-reproducible and are used only once in the entire authentication process. The presence of nonces ensure that stale messages are not replayed.

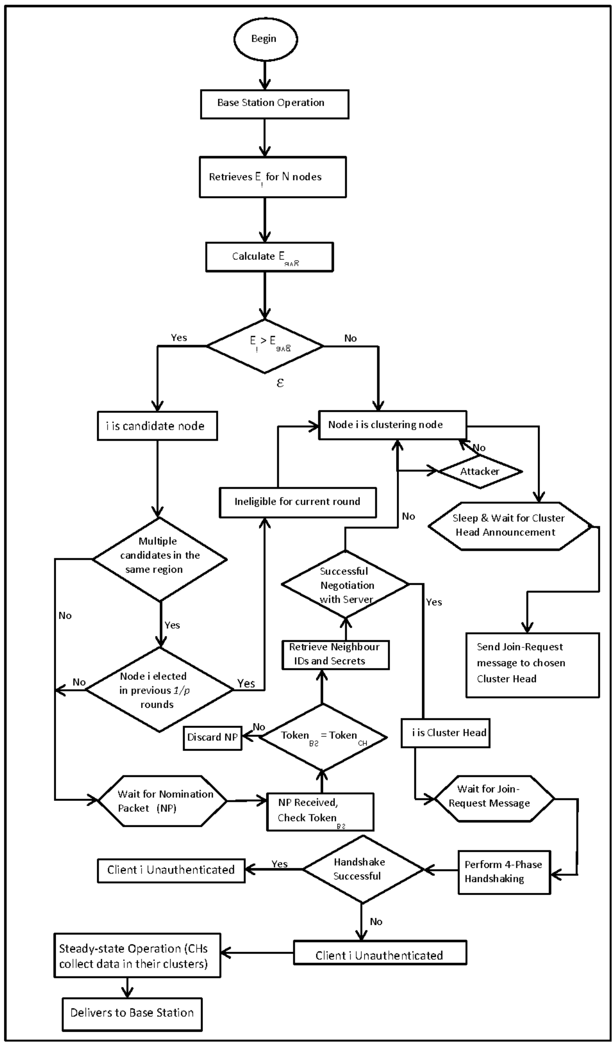

Begin
Base Station Operation
Retrieves E_i for N nodes
Calculate E_{TAR}
$E_i > E_{TAR}$
Yes
No
i is candidate node
Node i is clustering node
Attacker
No
Ineligible for current round
Sleep & Wait for Cluster Head Announcement
Multiple candidates in the same region
Successful Negotiation with Server
Yes
No
Send Join-Request message to chosen Cluster Head
Retrieve Neighbour IDs and Secrets
Node i elected in previous 1/p rounds
Discard NP
$Token_{B2} = Token_{CH}$
i is Cluster Head
Wait for Nomination Packet (NP)
NP Received, Check $Token_{B2}$
Wait for Join-Request Message
Client i Unauthenticated
Handshake Successful
Perform 4-Phase Handshaking
Steady-state Operation (CHs collect data in their clusters)
Client i Unauthenticated
Delivers to Base Station

Figure 4. Flowchart of PAWN Protocol

The freshness of all subsequent readings from sensor is guaranteed by nonces values which are incremented each time.

4.2. Robustness against various Attacks

The robustness of PAWN algorithm against various attacks and malicious activities is shown in Table II. In PAWN, η_{client} and η_{server} are generated by a pseudo-random number (R_i) and appended to a timer T_i. The combination of T_i and R_i assure that an adversary will find it extremely difficult to replay stale packets. The pseudo-random nature of T_i guarantees that η_{server} and η_{client} are non-reproducible while the timer T_i ensures that η_{server} and η_{client} are non-predictable. SecLEACH uses hash function (H) and ring keys (kr) for protection against replay attacks. LEACH-C, a pioneer protocol for cluster-based hierarchical architectures, does not provide any protection against any attack.

Apart from replay attack, PAWN is highly robust and defensive against resource exhaustion and DoS attacks. The proposed scheme uses λ_i for authenticating the session initiation request from any neighbouring node, also known as potential member node. The absence of λ_i in the server table ensures that any unauthenticated neighbouring node will not be able to establish one or more connection with a given server. As a result, the resources of a server remain intact. The nodes

Table II. Robustness against Network Attacks

Attacks	LEACH-C	SecLEACH	PAWN
Replay	No	Yes	Yes
Resource Exhaustion	No	No	Yes
Sybil	No	No	Yes

are temper-safe in order to avoid compromising the security primitives in accordance with the Internet Threat Model [24]. This model assumes that λ_i are hardcoded on the nodes at the time of manufacturing and deployment. In case, if an intruder attempts to temper with the hardware of a sensor node, an alarm is generated to notify about the security breach. A malevolent entity in a cluster head neighbourhood may fabricate its own identities. However, the absence of such fabricated identities and their matching λ_i in the server table will not enable this entity to conduct a Sybil attack. SecLEACH and LEACH-C, on the other hand, do not provide any defensive solution to tackle resource exhaustion and Sybil attacks.

4.3. Handshake Duration

The average handshake duration of PAWN is compared with SecLEACH in Table III for various cluster sizes, i.e., the number of potential member nodes in each cluster. The average handshake duration, in milliseconds, for SecLEACH is much higher than PAWN for various cluster sizes.

Table III. Handshake Duration (in ms)

Cluster Size	SecLEACH	PAWN
15	4331.44	2071.22
20	4991.08	2166.69
25	5311.02	2389.10

SecLEACH uses the underlying cluster hierarchy of LEACH protocol, a randomly distributed approach using a probabilistic threshold for cluster head selection as discussed in Equation 1. In SecLEACH, the absence of local knowledge of the neighbouring nodes requires much longer time for cluster heads to authenticate its nearest neighbours. PAWN, on the other hand, uses the local knowledge of its neighbours, i.e., the information provided by the base station about its neighbourhood, for authentication. Each cluster head only needs to look-up its table to validate/invalidate a session initiation request. The presence of λ_i enables each cluster head to enter into a four phase handshake negotiation for mutual authentication.

4.4. Average Energy Consumption

PAWN algorithm uses somewhat similar centralized approach for cluster formation as LEACH-C protocol. However, LEACH-C does not provide a secured transmission of data and is susceptible to a wide range of malicious attacks. In Figure 5, the average energy consumed by PAWN algorithm is shown in comparison with LEACH-C.

As depicted in Figure 5, the average energy consumed by PAWN is slightly higher (in order of millijoule) than LEACH-C in most of the rounds. However, the slight increase in energy consumption is negligible keeping in view that PAWN provides a complete set of operations for authentication and data protection. LEACH-C, on the other hand, does not provide any secured features and the election of cluster heads is too complex. LEACH-C elects cluster heads and forms clusters using the simulated annealing algorithm by solving the N-P hard problem of finding optimal clusters [13]. This technique of cluster formation and cluster head election incurs excessive delay and consumes too much energy. On the other hand, PAWN uses a simple technique for cluster

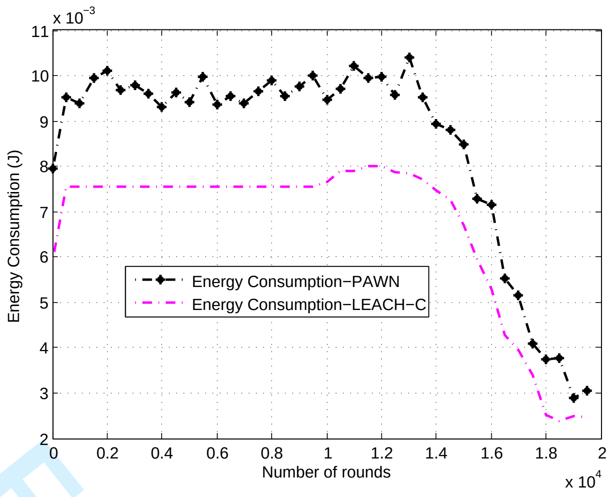


Figure 5. Average Energy Consumption

head selection which requires a much smaller amount of energy and at the same time provides a robust defence against various malicious attacks. The novelty and simplicity of cluster head election technique is the driving force behind energy minimization in PAWN algorithm.

4.5. *Average Network Throughput*

The average network throughput is calculated as the ratio of total number of packets successfully received to the total number of packets transmitted to a base station. The average network throughput of PAWN is shown in Figure 6

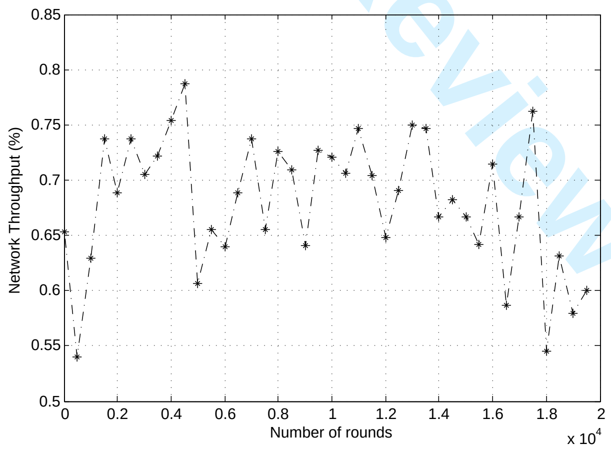


Figure 6. Average Network Throughput

The average network throughput of PAWN reaches up to 75% in most of the rounds despite its operation in hostile, remote locations and communication over error-prone noisy links. These factors cause too much distortion and attenuation of signals. Although PAWN provides a robust and defensive solution to the underlying network, the nature of deployed region and quality of the communication links also play an important role in packet loss and Quality of Service (QoS) degradation.

5. CONCLUSION

In this paper, we have proposed a lightweight mutual authentication scheme for a cluster-based hierarchical WSN. The proposed scheme achieves both energy-efficiency and security through implementing a lightweight mutual authentication scheme for cluster-based WSN. Our proposed algorithm, Payload-based mutual Authentication for Wireless Sensor NetworkS (PAWN), operates in two distinct phases and has been presented in detail in this paper. We have compared PAWN against state-of-the-art LEACH-C and SecLEACH in terms of robustness against attacks, handshake duration and average energy consumption. Unlike LEACH-C, PAWN is highly robust and resilient to various attacks such as the replay, DoS and Sybil attacks. SecLEACH, on the other hand, can detect only a handful of such attacks. The average handshake duration using PAWN is much shorter in comparison with SecLEACH. Depending on the cluster size, PAWN can establish an authenticated session with a time almost half of that required by SecLEACH. LEACH-C, on the other hand, lacks security features and as such does not establish an authenticated session. PAWN is also energy-efficient as compared with the existing schemes. We have implemented PAWN using AES-128 bit. We will test the performance of PAWN using other encryption algorithms in our future work. In this paper, PAWN performs base station/cluster head authentication using tokens. In our future work, we will perform the same task using alternative techniques to further strengthen the proposed scheme. In future, we will also investigate the performance of PAWN using a randomly distributed cluster-based hierarchy, similar to LEACH.

REFERENCES

1. P. Rawat, K. D. Singh, H. Chaouchi, and J. M. Bonnin, “Wireless sensor networks: a survey on recent developments and potential synergies,” *The Journal of supercomputing*, vol. 68, no. 1, pp. 1–48, 2014.
2. J. A. Stankovic, A. D. Wood, and T. He, “Realistic applications for wireless sensor networks,” in *Theoretical Aspects of Distributed Computing in Sensor Networks*. Springer, 2011, pp. 835–863.
3. Y. Yu, K. Li, W. Zhou, and P. Li, “Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures,” *Journal of Network and computer Applications*, vol. 35, no. 3, pp. 867–880, 2012.
4. S. T. Zargar, J. Joshi, and D. Tipper, “A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks,” *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 4, pp. 2046–2069, 2013.
5. M. A. Jan, “Energy-efficient routing and secure communication in wireless sensor networks,” Ph.D. dissertation, 2016.
6. M. A. Jan, P. Nanda, X. He, Z. Tan, and R. P. Liu, “A robust authentication scheme for observing resources in the internet of things environment,” in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2014 IEEE 13th International Conference on*. IEEE, 2014, pp. 205–211.
7. W. Xiao-yun, Y. Li-zhen, and C. Ke-fei, “Sleach: Secure low-energy adaptive clustering hierarchy protocol for wireless sensor networks,” *Wuhan University Journal of Natural Sciences*, vol. 10, no. 1, pp. 127–131, 2005.
8. D. Puthal, S. Nepal, R. Ranjan, and J. Chen, “A dynamic prime number based efficient security mechanism for big sensing data streams,” *Journal of Computer and System Sciences*, 2016.
9. Y.-H. Lee, J.-H. Kang, and S.-J. Lee, “A specification-based intrusion detection mechanism for leach protocol,” *The Journal of Korean Institute of Communications and Information Sciences*, vol. 37, no. 2B, pp. 138–147, 2012.
10. M. A. Jan, P. Nanda, X. He, and R. P. Liu, “A sybil attack detection scheme for a centralized clustering-based hierarchical network,” in *Trustcom/BigDataSE/ISPA, 2015 IEEE*, vol. 1. IEEE, 2015, pp. 318–325.
11. W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” in *Proceedings of the 33rd annual Hawaii international conference on System sciences*. IEEE, 2000, pp. 1–10.
12. M. A. Jan, P. Nanda, X. He, and R. P. Liu, “Enhancing lifetime and quality of data in cluster-based hierarchical routing protocol for wireless sensor network,” in *2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC)*. IEEE, 2013, pp. 1400–1407.
13. W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” *Wireless Communications, IEEE Transactions on*, vol. 1, no. 4, pp. 660–670, 2002.
14. M. A. Jan, P. Nanda, and X. He, “Energy evaluation model for an improved centralized clustering hierarchical algorithm in wsn,” in *Wired/Wireless Internet Communication*. Springer, 2013, pp. 154–167.
15. M. M. Morshed and M. R. Islam, “Cbsrp: Cluster based secure routing protocol,” in *Advance Computing Conference (IACC), 2013 IEEE 3rd International*. IEEE, 2013, pp. 571–576.
16. Y. Zhang, X. Li, J. Yang, Y. Liu, N. Xiong, and A. V. Vasilakos, “A real-time dynamic key management for hierarchical wireless multimedia sensor network,” *Multimedia tools and applications*, vol. 67, no. 1, pp. 97–117, 2013.
17. E. Shaaban *et al.*, “Enhancing s-leach security for wireless sensor networks,” in *Electro/Information Technology (EIT), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1–6.

18. M. Alshowkan, K. Elleithy, and H. AlHassan, “Ls-leach: a new secure and energy efficient routing protocol for wireless sensor networks,” in *Distributed Simulation and Real Time Applications (DS-RT), 2013 IEEE/ACM 17th International Symposium on*. IEEE, 2013, pp. 215–220.
19. R. Dutta, E.-C. Chang, and S. Mukhopadhyay, “Efficient self-healing key distribution with revocation for wireless sensor networks using one way key chains,” in *Applied Cryptography and Network Security*. Springer, 2007, pp. 385–400.
20. P. Schaffer, K. Farkas, Á. Horváth, T. Holczer, and L. Buttyán, “Secure and reliable clustering in wireless sensor networks: a critical survey,” *Computer Networks*, vol. 56, no. 11, pp. 2726–2741, 2012.
21. L. B. Oliveira, A. Ferreira, M. A. Vilaça, H. C. Wong, M. Bern, R. Dahab, and A. A. Loureiro, “SecleachâĂŤon the security of clustered sensor networks,” *Signal Processing*, vol. 87, no. 12, pp. 2882–2895, 2007.
22. Y. Cho, G. Qu, and Y. Wu, “Insider threats against trust mechanism with watchdog and defending approaches in wireless sensor networks,” in *Security and privacy workshops (SPW), 2012 IEEE Symposium on*. IEEE, 2012, pp. 134–141.
23. L. Yang, “Centralized security protocol for wireless sensor networks,” 2011.
24. E. Rescorla and B. Korver, “Guidelines for writing rfc text on security considerations,” 2003.

Reviewer 1 Comments	Comments Addressed
1. This paper is well-written, novel in terms of proposed idea and ease of understanding. The good thing about this paper is the use of cluster-based hierarchical protocols for security provisioning. Most of the existing literature on these protocols focused only on energy-efficient routing for network stability. However, the authors have used the underlying concept of these protocols in a very clever way to provide network security along with network stability. This paper can be approved further by incorporating the following comments.	Dear Reviewer, we would like to thank you for appreciating our work presented in this manuscript.
2. The data exchange within the cluster is part of steady-state phase. It would be better to rewrite the last step (Data exchange) of Algorithm 1 in an equation and/or mathematical form.	This comment has been addressed in the revised manuscript on line 30-31 in Algorithm 1, Page 10.
3. How many attempts are allowed from a client (either normal node or malicious) for session initiation before a client is blocked from further requests? It would be better to update the server with this information in order to refrain it from further connection negotiation with the given client.	Irrespective of a legitimate or a malicious node, each neighbor can negotiate a maximum of four session initiation requests with a server. After these attempts, any further requests from the given neighbor are discarded and the server refrains from any further communication with it. This comment has been addressed on Page 7 in the revised manuscript.

Reviewer 2 Comments	Comments Addressed
1. This paper is easy to understand, logical, up to the point, and matches the scope of the conference. The authors have used the distinguishing features of cluster-based hierarchy for addressing security issues faced by wireless sensor networks. Unlike most of the existing approaches, PAWN provides not only network	Dear Reviewer, we would like to thank you for appreciating our work presented in this manuscript.

stability but message authentication, integrity and in-network encryption as well. It would be appreciated if the authors could address the following questions for camera ready or most probably for an extended version of this paper to a journal as the authors are not left with enough space in the current paper.	
2. An intruder keeps sending initiation requests with valid object IDs to the server and forcing the server respond with challenges. Will the server suffer from resource exhaustion when the number of request grows extremely large?	The nodes are temper-safe in order to avoid compromising the security primitives in accordance with the Internet Threat Model. This model assumes that object ID or pre-shared secrets (λ_i) are hardcoded on the nodes at the time of manufacturing and deployment. In case, if an intruder attempts to temper with the hardware of a sensor node, an alarm is generated to notify about the security breach. From the above statement, it is clear that an intruder cannot retrieve the object ID of a legitimate node as it is in contrary to Internet Threat Model. This comment has been addressed in Page 12 of the revised manuscript.
3. An intruder keeps sending initiation requests with a valid object ID to the server. If these intrusive attempts will result in denial-of-service to the actual owner of the valid object ID? Authors have discussed the robustness of PAWN protocol against denial-of-service, resource exhaustion and denial of service attacks. How efficient it is against other type of attacks?	This comment has been addressed in Page 12 of the revised manuscript.

CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2010; **00**:1–15
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

PAWN: A Payload-based mutual Authentication scheme for Wireless Sensor Networks

Mian Ahmad Jan[1]*, Priyadarsi Nanda[2]*, Muhammad Usman[2], Xiangjian He[2]*

[1]*Department of Computer Science,*
Abdul Wali Khan University Mardan, Pakistan
[2]*School of Computing and Communications, Faculty of Engineering and Information Technology,*
University of Technology Sydney, Australia

SUMMARY

Wireless Sensor Networks (WSNs) consist of resource-starving miniature sensor nodes deployed in a remote and hostile environment. These networks operate on small batteries for days, months and even years depending on the requirements of monitored applications. The battery-powered operation and inaccessible human-terrains make it practically infeasible to recharge the nodes unless some energy scavenging techniques are employed. These networks experience threats at various layers and as such, are vulnerable to a wide range of attacks. The resource-constrained nature of sensor nodes, in-accessible human terrains and error-prone communication links make it obligatory to design lightweight but robust and secured schemes for these networks. In view of these limitations, we aim to design an extremely lightweight payload-based mutual authentication scheme for a cluster-based hierarchical WSN. The proposed scheme, also known as PAWN, operates in two steps. First, an optimal percentage of cluster heads is elected, authenticated and allowed to communicate with neighbouring nodes. Second, each cluster head, in a role of server, authenticates the nearby nodes for cluster formation. We validate our proposed scheme using various simulation metrics which outperforms the existing schemes.
Copyright © 2010 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: Cluster Head, Cluster, Authentication, Lightweight, Payload-based, Client-Server Model

1. INTRODUCTION

Wireless Sensor Network (WSN) is a collection of miniature sensor nodes deployed to monitor, sense, capture and process the data about an application, i.e., phenomena of interest [1]. These nodes are resource-starving and as such are highly constrained on battery power, storage, computation, data rate and available bandwidth. Typically, they are deployed and left unattended in a remote and human-inaccessible terrain to perform monitoring and reporting tasks. As a result, the limited resources of these nodes need to be utilized efficiently to prolong the network lifetime. Due to their unique characteristics of self-healing and fault-tolarance, these networks have found their applications in various domains such as military surveillance, health care, industrial automation, home automation, agriculture, and environmental monitoring [2].

*Correspondence to: [1]Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan.[2]School of Computing and Communications, University of Technology, Sydney, Australia.
E-mail: [1]Mianjan@awkum.edu.pk
E-mail: [2]Xiangjian.He@uts.edu.au
E-mail: [2]Priyadarsi.Nanda@uts.edu.au

Applications deployed in hostile environment with little human interventions to manage them expose these networks to various threats and vulnerabilities. These networks are susceptible to not only physical tampering but also various remote attacks such as sink hole, worm hole, Sybil, replications, homing and distributed denial-of-service flooding [3][4]. As a result, the nodes in general and the network in particular need to be protected against these threats. Lightweight but highly efficient and robust authentication schemes need to be designed for these networks. The resource-constrained nature of the nodes limits the support and application of resource-intensive, highly robust and sophisticated authentication and privacy schemes to these networks. The existing security schemes require ample of resources in terms of computation, transmission, storage and available bandwidth. Their application to WSNs deteriorates the network performance and rapidly depletes the network lifetime. As a result, designing lightweight but highly intelligent and robust security solutions is a challenging task in these networks. The provisioning of security, privacy and trust need to be kept in mind while the prototype of a routing protocol is designed. Furthermore, the underlying operations of different routing protocols also influence the energy consumption which is linked directly with the application of any secured scheme. Among all the routing protocols, cluster-based hierarchical routing protocols efficiently utilize the limited resources of sensor nodes. These protocols are highly efficient in terms of data aggregation, energy consumption, collision avoidance, load balancing, fault-tolerance and network lifetime [5]. Unlike other protocols, a single cluster head collects data from multiple nodes in order to eliminate redundancy and improves the quality of aggregated data. These protocols use intra-cluster and inter-cluster communication to reduce long-haul transmissions to a base station.

Cluster-based hierarchical routing protocols perform only intelligent partitioning of the network to provide energy-efficient communication. The underlying network still suffers from various threats posed during set-up and steady-state phases. During set-up phase, an adversary may declare itself as a cluster head by broadcasting advertisement messages with much stronger signal strengths. It is also possible that an adversary may participate in cluster formation or prevent legitimate nodes from joining a cluster head. An attacker may eavesdrop on advertisement packets and/or join-request packets and replay in other parts of the network. Furthermore, the same attacker has the ability to maliciously manipulate the contents of these packets. An adversary may disrupt the ongoing operations in one or more clusters by constantly emitting jamming signals or launching Denial-of-Service (DoS) attack [6]. An adversary may deceive a cluster head by persuading it to create a false member list for its members and as a result, a wrong Time Division Multiple Access (TDMA) schedule is created for its members [7]. An adversary may launch a more sophisticated attack by colluding with other adversaries in their neighbourhood [8]. In [9], the authors analyzed the behaviour of various attacks such as, continuous election as cluster head, TDMA schedule obedience, aborting and/or no transmission of member/cluster head data and transmission of powerful signals. In [10], the authors investigated the behaviour of Sybil nodes in cluster-based hierarchical routing protocols. They studied the impact and consequences of Sybil nodes performing the role of cluster heads in these networks. A single Sybil node has the ability to breakdown the whole communication in a cluster-based WSN provided that it is elected as a cluster head.

In this paper, we present a payload-based mutual authentication algorithm, also known as PAWN, which uses a centralized cluster-based hierarchical WSN. Our proposed algorithm operates in two steps. In the first step, an optimal percentage of cluster heads are elected to balance the network load by minimizing the energy consumption of the nodes. For security provisioning, a token-based authentication approach is adopted. A 16-bit token is used by the base station and the nominated cluster heads for correlating the advertisement messages with the corresponding acknowledgement messages. This step ensures that only legitimate cluster heads are elected by a legitimate base station. In the second step, each elected cluster head negotiates cluster formation with the nearby neighbouring nodes using four handshake messages which are encrypted with Advanced Encryption Standard (AES). Data authentication and secured exchange is guaranteed within the payload of each transmitted message. The major contributions of our research are as follow.

1. Unlike the existing centralized cluster-based hierarchical routing protocols, our proposed algorithm always elects an optimal percentage of cluster heads to enhance the lifetime of the network and reduce the number of server nodes required for communication.
2. Unlike the existing approaches which rely mostly on asymmetric algorithms, PAWN uses a simple 16-bit token and a lightweight symmetric algorithm, AES in this case, to provide encryption within the payload of each message. The use of AES in 128-bit mode incurs extremely small communication and processing overhead and at the same time, the payload of each message does not exceed 256 bits. For resource-starving sensor nodes, the use of asymmetric algorithms is a highly resource-intensive operation because most of the resources of a node are particularly dedicated to these algorithms.

The rest of the paper is organized as follows. In Section II, we present the related works from literature. In Section III, we present our PAWN algorithm in terms of cluster head selection and cluster formation. In Section IV, experimental results of our proposed scheme are presented. Finally, the paper is concluded and future research directions are provided in Section V.

2. RELATED WORK

In this section, we provide an overview of cluster-based hierarchical routing protocols as they are used as the underlying platform by our proposed scheme. Next, various lightweight mutual authentication and key agreement schemes are discussed in the context of WSNs.

Low-Energy Adaptive Clustering Hierarchy (LEACH) [11] was designated as a pioneer protocol among clustering-based hierarchical routing protocols. LEACH partitions a sensor field into small geographical regions known as clusters. Each cluster has a cluster head node which collects and aggregates data from member nodes and transmits them to a base station. The protocol operates in rounds and nodes take turn to become cluster heads in subsequent rounds for uniform distribution of energy load. The problem with LEACH protocol is the probabilistic selection of cluster heads using random number generation. Each node, *i*, chooses a random number between 0 and 1. If the chosen number is less than the threshold value *T(i)*, the node is elected as a cluster head for the current round. The operation of LEACH for cluster head selection is shown in Equation 1.

$$T(i) = \begin{cases} \frac{k_{opt}}{1-k_{opt}(r mod(\frac{1}{k_{opt}}))}, & if\ i \in G, \\ 0, & otherwise. \end{cases} \tag{1}$$

Here, k_{opt} is the optimal percentage of cluster heads in each round, r is the current round and G is the set of nodes that have not been elected as cluster heads in the past $\frac{1}{k_{opt}}$ rounds. The probabilistic selection of cluster heads has a potential risk of low energy nodes being elected as cluster heads in subsequent rounds. Moreover, Equation 1 cannot guarantee an optimal number of cluster heads in each round. In [12], the authors argued that cluster heads need to be elected based on the residual energy of the nodes. They suggested the inclusion of residual energy of nodes in Equation 1. However, it does not solve the problem because cluster heads are still elected using a random number generation. To solve this problem, nodes need to be elected by a central controller, i.e., a base station. In [13], the authors proposed a centralized approach for cluster head selection. Nodes having remaining energy greater than the average residual energy are elected as cluster heads in each round. However, it is highly probable that there are a large number of such nodes in each round resulting in too many cluster heads. In [14], we proposed a centralized scheme which elected an optimal percentage of cluster heads (5% of total nodes). Each round results in balanced clusters that enhance network stability, scalability and data aggregation. Moreover, the proposed approach reduces network load, energy consumption and congestion. Irrespective of a centralized or randomly distributed cluster-based approach, each round consists of a set-up phase and a steady-state phase. During the set-up phase, cluster heads are elected, cluster are formed and schedules are created. In the steady-state phase, each cluster head collects data from its member nodes and transmits them to a centralized base station.

For secured transmission of data in cluster-based hierarchical networks, there exist various secured versions of LEACH in literature [15] [16][17][18][21][22][27][28]. Most of these protocols used a one-way key chain[19] for encryption and decryption purposes. The chain is generated using a one-way hash function $\mathbb{H}$ on the last key in each chain. In [20], a survey was conducted to analyze and evaluate the performances of various secured cluster-based hierarchical schemes. Most of these schemes increase the security of set-up and steady-state phases in particular and the overall operation of the network in general. In these schemes, LEACH-alike cluster hierarchies are protected against various attacks launched by adversaries from outside the network. These schemes make it hard to compromise various operations performed within the set-up and steady-state phases by deploying various key distribution mechanisms. The application of asymmetric encryption to miniature sensor nodes exhausts their resources quickly and rapidly reduce the lifetime of WSNs. As a result, most of the studies focus on symmetric encryption for security and privacy provisioning in cluster-based WSNs. In [23], SecLEACH protocol was proposed for securing cluster-based hierarchical WSNs. Prior to network deployment, SecLEACH generates a large pool of keys using a random key pre-distribution approach. Each node is allocated a small subset of these keys, also known as key ring, drawn pseudo randomly from the pool. Furthermore, each node is assigned a pairwise key which is shared with the base station and a group key common to each member of the network. Upon network deployment, each cluster head advertises itself by broadcasting a message containing information about the keys in its pre-defined key ring. In SecLEACH, the base station is responsible for authenticating the advertisement messages as it is a resource-rich entity and is highly trusted. Upon authentication by the base station, each neighbouring node forms a cluster around its nearest cluster head with whom they share the group key. Cryptographic-based LEACH variants are robust only against external attacks, however, they are not capable to tackle threats posed by internal adversaries, e.g., a compromised sensor. This is due to the fact that compromised sensors have already been allocated authentication keys and they have the same privileges which are required for transmission of authenticated information. As a result, conventional authentication algorithms fail to detect such an attack [24]. A centralized secure LEACH (CSLEACH) was proposed in [25]. Each neighbouring node and its gateway share a unique private key which is used for authentication of broadcasted advertisement messages. Each node possesses two keys, i.e., a gateway private key and a neighbor private key. A node waits for a message upon entering the network at the start of each round. A particular round is initiated by the gateway using a Round Start Message (RSM). These messages are mainly used for synchronization purposes. These messages distribute a network key and a session template for generation of Medium Access Control keys (KMAC) and session keys. Any communication between a neighbouring node and its gateway is encrypted by the session key. These keys also encrypt the communication between a cluster head and its member nodes. KMAC, on the other hand, encodes MAC for providing integrity in the network.

3. PAWN: PAYLOAD-BASED AUTHENTICATION FOR WSN

Our proposed scheme mainly consists of two important steps and are presented as follows.

1. An optimal percentage of cluster heads are elected and advertised to the network using a token-based authentication approach. The token is used for correlating cluster head advertisement messages with the corresponding acknowledgment messages.
2. A payload-based mutual authentication is used for cluster formation between neighbouring nodes and their nearest cluster head.

Both of these steps are part of the set-up phase during which cluster heads are elected and clusters are formed around them. In this section, first we present the token-based optimal election of cluster heads followed by a payload-based mutual authentication for a cluster-based hierarchical WSN. An optimal election of cluster heads enables us to nominate very few nodes as servers. The unelected nodes become the potential clients of cluster heads in the network.

3.1. *Token-based Cluster Head Election*

Our proposed network model consists of randomly deployed heterogeneous sensor nodes. There are 100 normal nodes and 5 higher-energy nodes in the network. To reduce the operational cost, normal nodes are equipped with initial energy of 1 joule each, and higher-energy nodes are equipped with 5 joules each. All nodes are deployed in a 100×100 sq. meter geographical region. The role of each node is specified as follows.

1. Only normal nodes can be elected as cluster heads.
2. Only normal nodes are eligible to participate in a client-server interaction model.
3. Higher-energy nodes assist the base station in cluster head election.
4. Higher-energy nodes exchange control packets with the base station while normal nodes transmit both data and control packets.

Upon network deployment, each normal node is assigned a token and a pre-shared secret (λ_i). Token is used for secure exchange of nomination packets and *ACKs* between a base station and a cluster head while λ_i is used for in-network mutual authentication between a neighbouring node and its prospective cluster head. Higher-energy nodes perform various administrative tasks such as route discovery, route maintenance and neighbourhood discovery. Each higher-energy node collects and disseminates control packets during upstream and downstream traffic.

Initially, each normal node, *i*, creates and broadcasts a control packet to its nearest higher-energy node as shown in Figure 1. This packet contains the residual energy (E_i) and identity (ID_i) of *i*. The location of a nearest higher-energy node with respect to *i* is computed by solving Euclidean distance ($\mathrm{d}_{\mathbb{H}}$) of Equation 2.

$$d_{\mathbb{H}} = \sqrt{(x_{\mathbb{H}} - x_i)^2 - (y_{\mathbb{H}} - y_i)^2}. \tag{2}$$

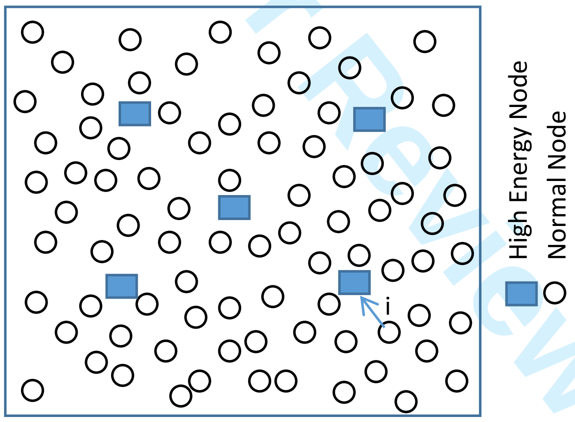


Figure 1. Network at the Time of Deployment

Each higher-energy node collects these packets in their neighbourhood, retrieves E_i and ID_i, and broadcasts a single control packet containing these values to a base station. After transmission of the packet to a base station, the higher-energy nodes go to sleep mode and will awake at the start of next round to initiate the aforementioned tasks. Upon reception, the base station retrieves E_i and ID_i from each control packet. At this stage, an average energy threshold is computed using Equation 3.

$$E_{avg} = \sum_{i=1}^{N} \frac{E_i}{N}. \tag{3}$$

Here, E_{avg} is the average energy threshold for a total of N normal nodes, where N is equal to 100.

Any normal node *i* having E_i greater than or equal to E_{avg} in a particular round is eligible for cluster head selection, $\forall\ i \in \{1, 2, \cdots, N\}$. The only exception is the initial round immediately after network deployment, where all the nodes have nearly the same residual energy values, i.e., E_i $\approx E_{avg}$. All such nodes are eligible for cluster head selection because $\Delta E_{a,b} \approx 0$, where $\Delta E_{a,b}$ = E_a - E_b, $\forall\, a, b \in \{1, 2, , \cdots, N\}$. In our proposed scheme, base station has the ability to elect cluster heads based on extremely low values of ΔE, up to 4 decimal digits.

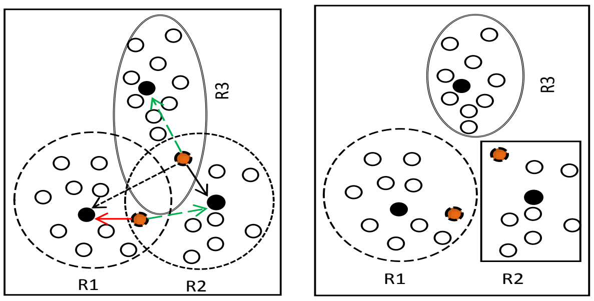
In cluster-based hierarchical routing protocols, it is highly probable that there may be a large number of nodes for whom $E_i \geq E_{avg}$ in various rounds. All such nodes are candidates, also known as nominees, for cluster heads. The only exception is towards the end of network lifetime where nominees are either equal to or less than k_{opt}. The base station is responsible to elect k_{opt} cluster heads among the nominees. We use the following criteria for an optimal election.

- E_i of a nominee, i, is greater than or equal to E_{avg}.
- i is not elected as cluster head during the past $\frac{1}{p}$ rounds.
- Two or more nominees located in a same geographical region are evaluated based on their energy values and previous history of election.

In our proposed scheme, k_{opt} is 5% for a network of N nodes. An optimal percentage of cluster heads influences the performance of cluster-based hierarchical networks. Energy of a cluster head is depleted rapidly due to resource-intensive tasks such as data aggregation, fusion and long-haul transmission to a base station. A small value of k_{opt} will deteriorate the performance of cluster-based hierarchical network because each cluster head will accommodate a large set of neighbouring nodes in its cluster. In this case, energy of each cluster head will be exhausted rapidly in the aforementioned resource-intensive operations. A large value of k_{opt} will make a cluster-based hierarchical network rather inefficient and ineffective due to low data aggregation and fusion [10].

In each round, the base station broadcasts nomination packets to the elected cluster heads containing their identities (forming the set ID_{CH}) and identities (forming the set ID_{NB}) of their neighbouring nodes. Both these types of identities are transmitted within the payload of nomination packets. Recall that the base station knows the identity of each node and its geographical location. To ensure secured transmission of ID_{CH} and ID_{NB}, each nomination packet has a 16-bit token in its header. The base station generates a token similar to the one possessed by an elected cluster head, i.e., one token per cluster head is generated. The number of tokens generated depends on the number of elected cluster heads. Upon reception, each cluster head, denoted by S, examines the appended token within the nomination packet. If it matches with the one possessed by it, the cluster head retrieves the appended identities ($ID_{\overline{\mathbb{NB}}}$) of its neighbouring nodes, where $ID_{\overline{\mathbb{NB}}} \subset ID_{NB}$. Each cluster head must acknowledge the receipt of a nomination packet. It creates an acknowledgment message (ACK), appends its token to the ACK and transmits the ACK to the base station. An adversary may intercept one or more nomination packets, however, it cannot reproduce the token required to retrieve $ID_{\overline{\mathbb{NB}}}$. An adversary would require 2^{16} attempts to retrieves $ID_{\overline{\mathbb{NB}}}$ from a nomination packet. Upon election and successful exchange of nomination packet/ACK, each cluster head advertises itself to the nearest neighbours. A neighbouring node may receive advertisement messages from multiple cluster heads, however, it associates itself with the one having the strongest signal strength among all. The neighbouring nodes in overlapping regions receive signals from multiple cluster heads as shown in Figure 2(a). In this figure, the circles and ovals do not represent clusters. Instead, they represent the regions where the signal strength of each cluster head, identified with black colour nodes, is detected. The signal strength experienced by a node is represented by an arrow, where a solid arrow represents a strong signal, dashed arrow represents a medium signal and dotted arrow represents a weak signal. Each neighbouring node pursues its connection with a cluster head having the strongest signal as shown in Figure 2(b). In this figure, the oval, circle and rectangle indicate the intention of each node for joining a prospective cluster head. Cluster formation is yet to take place at this point of time.

Each neighbour node sends a join-request message to a cluster head having the strongest of signal strength for cluster formation. However, cluster formation is not a straight forward process. A neighbour needs to authenticate itself before association with a cluster head. Furthermore, each neighbour is not sure about the authenticity of a cluster head. As a result, not only the neighbours, but also the cluster head needs to authenticate themselves. To do so, each neighbour transmits a join-request message to a prospective cluster head as discussed in the next section.



(a) Overlapping Regions (b) RSS-based Clustering

Figure 2. Initiating Cluster Formation

3.2. *Payload-based Mutual Authentication*

Upon successful reception of advertisement messages, each neighbouring node initiates cluster formation. The aim of cluster head selection was to elect a feasible number of servers for neighbouring nodes. On the other hand, the aim of cluster formation is to allow authentic data transmission to a base station via the elected servers. The authentication process is accomplished using a 128-bit AES encryption algorithm. This mode of encryption requires less resources and is extremely beneficial for time-critical, delay-sensitive data generated by the nodes. Furthermore, it allows enough time for the nodes to offload their data before an attack is launched. AES-192 and AES-256, on the other hand, are highly computational, resource-consuming and require sophisticated hardware and software platforms. Such requirements are not fulfilled by most of the resource-constrained sensor nodes.

To meet the requirements of resource-starving miniature sensor nodes, we use a lightweight payload-based mutual authentication technique. In our proposed scheme, authentication is performed within the payload of each message. Each neighbouring node exchanges multiple handshake messages with a server as shown in Figure 3. The authentication process is completed in four extremely light-weighted handshake messages. Once authenticated, each neighbouring node is allowed to transmit its data to its respective cluster head. The authentication procedure is accomplished using the following four simple phases.

1. Session Initiation
2. Server Challenge
3. Client Response and Challenge
4. Server Response

During session initiation, each neighbour, i, creates a join-request message which contains ID_i and ID_{CH_S}. Here, ID_i is the identity of the source node and ID_{CH_S} is the identity of the destination node of cluster head, S. Both the source and destination IDs are of 8-bits. This number of bits results in sufficiently light-weighted join-request messages. Irrespective of a legitimate or a malicious node, each neighbour can negotiate a maximum of four session initiation requests with a server. After these attempts, any further requests from the given neighbour are discarded and the server refrains from any further communication with it. During session initiation phase, ID_i is transmitted within the payload and ID_{CH_S} is transmitted within the header of each join-request message. The payload is followed by an optional Frame Check Sequence (FCS), which is appended as a trailer and is used for error detection and correction.

In the server challenge phase, each server retrieves ID_{CH_S} from the header and ID_i from the payload of a received join-request message. If ID_{CH_S} matches the server ID, it means that the join-request message was indeed intended for it. If a match is not found, the join-request message is discarded. To negotiate a session, ID_i must match with an identity within the pool of identities provided to a given server. A match can only be found if $ID_i \in ID_{\overline{\mathbb{NB}}}$. Each server retrieves ID_i from

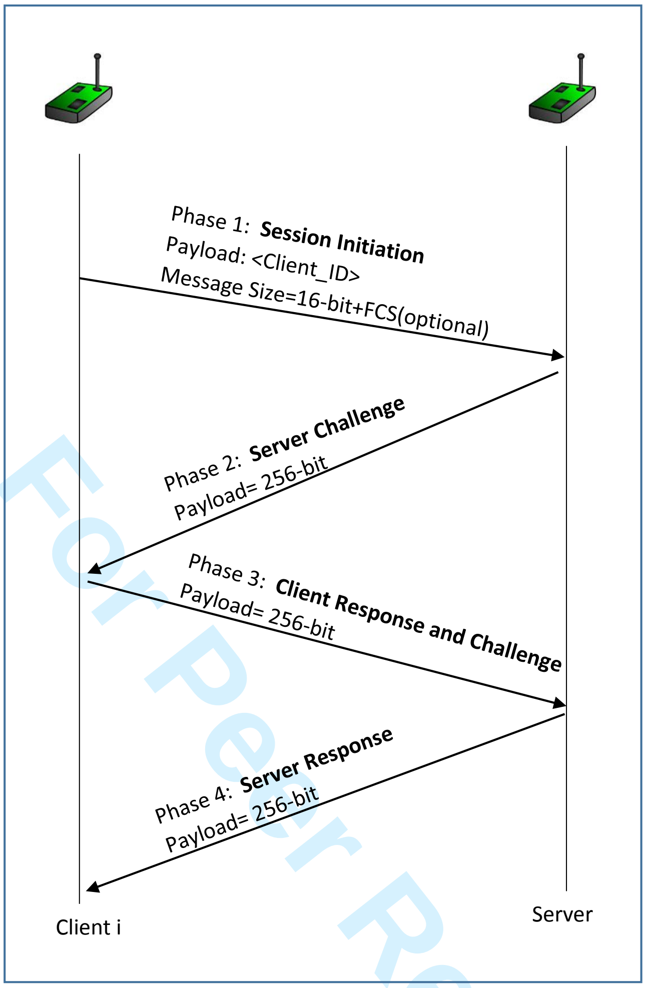

Phase 1: **Session Initiation**
Payload: <Client_ID>
Message Size=16-bit+FCS(optional)
Phase 2: **Server Challenge**
Payload= 256-bit
Phase 3: **Client Response and Challenge**
Payload= 256-bit
Phase 4: **Server Response**
Payload= 256-bit
Client i
Server

Figure 3. Payload-based Mutual Authentication

the payload and searches for a matching λ_i. Each identity ID_i within $ID_{\overline{\mathbb{NB}}}$ is associated with a λ_i for authentication purpose. If a match is found, the server responds back with an encrypted payload using an AES algorithm. At this point, the server creates a challenge for the client *i*. For successful authentication, this challenge needs to be deciphered by *i*.

To create a challenge, the server generates a pseudo-random nonce (η_{server}) and a potential session key (μ_{key}) of 128-bit each. Nonce is a temporary number used only once by a client/server in the entire cryptographic communication. Using μ_{key} and η_{server}, an encrypted payload is generated. First, an XOR operation is performed on λ_i and μ_{key} using Equation 4. This operation is computationally inexpensive and is extremely common as a component in complex ciphers. Moreover, this operation does not leak information about the original plaintext and applying it twice enables the retrieval of original plaintext. In our case, plaintext is the session key.

$$\psi_{resultant} = \lambda_i \oplus \mu_{key}. \tag{4}$$

The 128-bit $\psi_{resultant}$ is appended to η_{server} and encrypted with λ_i to generate a payload of 256-bit using Equation 5. This payload, i.e., $\gamma_{server\text{-}payload}$, is transmitted to the client as a challenge.

$$\gamma_{server\text{-}payload} = \{\lambda_i, (\psi_{resultant}|\eta_{server})\}AES128. \tag{5}$$

In the client response and challenge phase, the client needs to decipher the encrypted payload ($\gamma_{server\text{-}payload}$) to retrieve the potential session key μ_{key}. If the client is successful to do so, it will have the correct η_{server} and μ_{key}. The η_{server} and μ_{key} are known only to the server and λ_i belongs to a specific client. Only a legitimate client can decipher this challenge. An intruder can eavesdrop only on η_{server} and μ_{key}, but not on λ_i in accordance with the Internet Threat model [26]. The client uses its λ_i to decipher the payload. Upon successful deciphering, the client has authenticated itself. As mutual authentication requires both parties to be verified, the server also needs to authenticate itself. The client generates a new encrypted payload similar to the server. First, an XOR is performed on η_{server} and λ_i using Equation 6.

$$\psi_{\mathit{resultant}} = \eta_{server} \oplus \lambda_i. \tag{6}$$

The 128-bit $\psi_{resultant}$ is appended with η_{client} and encrypted with μ_{key} to generate an encrypted payload as shown in Equation 7. The 256-bit encrypted payload ($\gamma_{\mathit{client\text{-}payload}}$) is transmitted to the server as a challenge.

$$\gamma_{\mathit{client\text{-}payload}} = \{\mu_{key}, (\psi_{resultant}|\eta_{client})\}\mathit{AES128}. \tag{7}$$

Finally, in the server response phase, the server deciphers the encrypted payload ($\gamma_{\mathit{client\text{-}payload}}$) of the client challenge to observe η_{server} in it. If present, the server realizes that the client has successfully authenticated itself. The server retrieves η_{client} and creates an encrypted payload of its own by appending η_{client} to μ_{key} and encrypting with λ_i as shown in Equation 8. Next, the 256-bit encrypted payload ($\gamma_{\mathit{server\text{-}payload}}$) is transmitted in response to the client challenge.

$$\gamma_{\mathit{server\text{-}payload}} = \{\lambda_i, (\eta_{client}|\mu_{key})\}\mathit{AES128}. \tag{8}$$

At this point of time, the status of ongoing session changes from *session negotiation* to *authenticated* at the server because client *i* is successfully authenticated and is eligible to transmit its data to the server. The client, however, has yet to verify the authenticity of the server. It decrypts the payload $\gamma_{\mathit{server\text{-}payload}}$ and observe η_{client} in it. As the client is the one which generated η_{client}, the client comprehends that the response is from a legitimate server. Both the client and the server are mutually authenticated and they have agreed upon a common session key (μ_{key}) for exchanging the data packets. At this stage, both i and the server are mutually authenticated. The server creates a schedule for each member node i within its cluster. The server allocates TDMA slots to each member node within its cluster. These slots are used for contention-free communication and scheduling the duty-cycle of each node. The four phases of our payload-based mutual authentication handshaking are shown in Algorithm 1.

Once the client and server are authenticated, the set-up phase is completed and the steady-state phase is initiated. During this phase, each cluster head collects data from its member nodes, aggregates the data and transmits them to a base station located inside or outside the sensor field. In Figure 4, the complete set of operations performed by PAWN is shown.

4. EXPERIMENTAL RESULTS

In this section, we provide simulation results and experimental analysis of PAWN algorithm. For simulation, we set up our experiments using Matlab R2011a. PAWN network model consists of 100 normal nodes and 5 higher-energy nodes deployed in a 100×100 square meter area. The base station is located outside the sensor field and is a resource-rich entity.

4.1. Security Analysis

PAWN provides authenticity, confidentiality and freshness of data for node-to-node communication. In Table I, we compare PAWN against the existing schemes.

PAWN uses a two-step procedure for provisioning of authentication. First, a predefined procedure is used to elect the cluster heads. Upon election, a token-based technique is used for secured transmission of advertisement messages. Second, each cluster head, in a role of server, verifies the identities of potential clients, i.e., the neighbouring nodes. The payload-based authentication is

Algorithm 1 Payload-based Handshake Algorithm

```
1:  Initialization:
      1. Each neighbour, $i$, sends $ID_i$ to $S$, a prospective Server (Cluster Head).
      2. $S$ is provided with $ID_{\overline{\mathbb{NB}}}$ and $\lambda_i$
      3. Input: $\{ID_{\overline{\mathbb{NB}}}, \lambda_i\}$      ▷ $\lambda_i \in \{0, 1, \cdots, 2^{128} - 1\}, \forall\, i \in \{1, 2, \cdots, N\}$
2:  for $i = 1 : N$ do      ▷ Nested For loop generates a two-column server table
3:      for $j = 1 : 2$ do
4:          input $A[i][j]$      ▷ $ID_i$ and $\lambda_i$ are stored in $A[i][0]$ and $A[i][1]$ respectively.
5:      end for
6:  end for
7:  Phase 1 [Session Initiation]: $i$-> $S$: {join-request with $ID_i$}
8:      ▷ $i$ sends a join-request message to $S$ containing $ID_i$ in the payload.
9:  Phase 2 [Server Challenge]: $S$ retrieves $ID_i$ to find a matching $\lambda_i$
10: if $ID_i == A[i][0]$ then      ▷ A match is found
11:     Session negotiation is initiated between $i$ and $S$
12:     $S$ responds with an encrypted payload, $\{\lambda_i, (\lambda_i \oplus \mu_{key} | \eta_{server})\}AES128$
13:         ▷ where $\mu_{key}, \eta_{server} \in \{0, 1, \cdots, 2^{128} - 1\}$
14: else
15:     $i$ is unauthorized and join-request is discarded
16: end if
17: Phase 3 [Client Response & Challenge]: $i$ deciphers challenge and responds with an encrypted
    payload, $\{\mu_{key}, (\eta_{server} \oplus \lambda_i | \eta_{client})\}AES128$,
18: Phase 4 [Server Response]: $S$ checks $\eta_{server}$ in the client challenge by comparing against the
    $\eta_{server}$ generated in phase 3
19: if Both matches then
20:     [$i$ becomes a member node of $S$]- Cluster is formed
21:     $S$ responds as $\{\lambda_i, (\eta_{client} | \mu_{key})\}AES128$
22: else
23:     [Session Negotiation Fails]-$i$ is unauthorized and barred from communication in cluster
    formation
24: end if
25: $i$ compares $\eta_{client}$ of phase 3 and 4.
26: if Both matches then
27:     $S$ becomes member node of $S$
28:     $S$ allocates TDMA slots to $i$
29:     [Data Exchange within a Cluster]:
30:         $i$->$S$: $\{d_i$, where $d_i$ represents data packets of $i\}$.
31:         ▷ $i$ waits for its turn and transmits data encryped with $\mu_{key}$ to $S$.
32: else
33:     $S$ is unauthorized
34: end if
```

used to validate the identities of nodes in the incoming packets. In comparison, LEACH-C does not provide any authentication as it only partitions the network into clusters. SecLEACH, on the other hand, uses Message Authentication Code (MAC) for authentication purpose. Each message is encrypted with a key using a key pool. After successful decryption of the message, the receiver knows that message is originated from a legitimate node in the sensor field.

PAWN uses a payload-based approach to provide message confidentiality. The payload of join-request messages and nomination packets are used for this purpose. LEACH-C, on the other hand, does not provide data confidentiality while SecLEACH uses the same MAC for message confidentiality. Freshness of messages is guaranteed by nonces in PAWN and SecLEACH. PAWN, as a mutual authentication scheme, includes both the client and server nonces because both parties

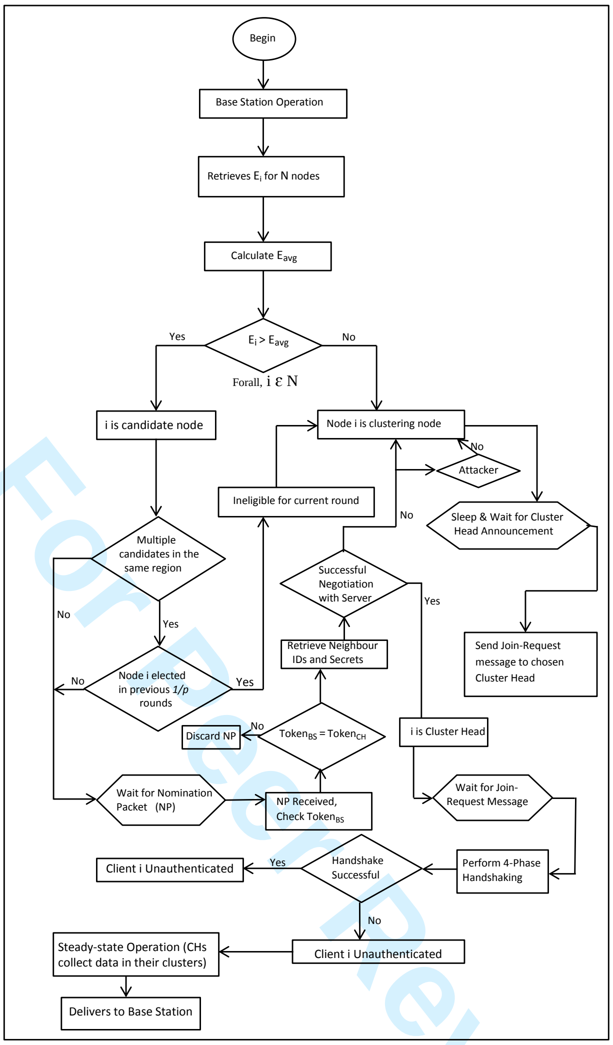

Begin
Base Station Operation
Retrieves E_i for N nodes
Calculate E_{avg}
$E_i > E_{avg}$
Forall, i ε N
Yes
No
i is candidate node
Node i is clustering node
No
Attacker
Ineligible for current round
No
Sleep & Wait for Cluster Head Announcement
Multiple candidates in the same region
Successful Negotiation with Server
Yes
No
Yes
Send Join-Request message to chosen Cluster Head
Retrieve Neighbour IDs and Secrets
No
Node i elected in previous 1/p rounds
Yes
Discard NP
No
$Token_{BS} = Token_{CH}$
i is Cluster Head
Wait for Nomination Packet (NP)
NP Received, Check $Token_{BS}$
Wait for Join-Request Message
Client i Unauthenticated
Yes
Handshake Successful
Perform 4-Phase Handshaking
No
Steady-state Operation (CHs collect data in their clusters)
Client i Unauthenticated
Delivers to Base Station

Figure 4. Flowchart of PAWN Protocol

Table I. Security Analysis

Security Services	LEACH-C	SecLEACH	PAWN
Authentication	No	MAC	Two-step
Confidentiality	No	MAC	Payload
Freshness	No	Nonces	Nonces

have to authenticate each other. Nonces are non-reproducible and are used only once in the entire authentication process. The presence of nonces ensure that stale messages are not replayed. The freshness of all subsequent readings from sensor is guaranteed by nonces values which are incremented each time.

4.2. *Robustness against various Attacks*

The robustness of PAWN algorithm against various attacks and malicious activities is shown in Table II. In PAWN, η_{client} and η_{server} are generated by a pseudo-random number (R_i) and appended to a timer T_i. The combination of T_i and R_i assure that an adversary will find it extremely difficult to replay stale packets. The pseudo-random nature of T_i guarantees that η_{server} and η_{client} are non-reproducible while the timer T_i ensures that η_{server} and η_{client} are non-predictable. SecLEACH uses hash function (H) and ring keys (kr) for protection against replay attacks. LEACH-C, a pioneer protocol for cluster-based hierarchical architectures, does not provide any protection against any attack.

Table II. Robustness against Network Attacks

Attacks	LEACH-C	SecLEACH	PAWN
Replay	No	Yes	Yes
Resource Exhaustion	No	No	Yes
Sybil	No	No	Yes

Apart from replay attack, PAWN is highly robust and defensive against resource exhaustion and DoS attacks. The proposed scheme uses λ_i for authenticating the session initiation request from any neighbouring node, also known as potential member node. The absence of λ_i in the server table ensures that any unauthenticated neighbouring node will not be able to establish one or more connection with a given server. As a result, the resources of a server remain intact. The nodes are temper-safe in order to avoid compromising the security primitives in accordance with the Internet Threat Model [26]. This model assumes that λ_i are hardcoded on the nodes at the time of manufacturing and deployment. In case, if an intruder attempts to temper with the hardware of a sensor node, an alarm is generated to notify about the security breach. A malevolent entity in a cluster head neighbourhood may fabricate its own identities. However, the absence of such fabricated identities and their matching λ_i in the server table will not enable this entity to conduct a Sybil attack. SecLEACH and LEACH-C, on the other hand, do not provide any defensive solution to tackle resource exhaustion and Sybil attacks.

4.3. Handshake Duration

The average handshake duration of PAWN is compared with SecLEACH in Table III for various cluster sizes, i.e., the number of potential member nodes in each cluster. The average handshake duration, in milliseconds, for SecLEACH is much higher than PAWN for various cluster sizes.

Table III. Handshake Duration (in ms)

Cluster Size	SecLEACH	PAWN
15	4331.44	2071.22
20	4991.08	2166.69
25	5311.02	2389.10

SecLEACH uses the underlying cluster hierarchy of LEACH protocol, a randomly distributed approach using a probabilistic threshold for cluster head selection as discussed in Equation 1. In SecLEACH, the absence of local knowledge of the neighbouring nodes requires much longer time for cluster heads to authenticate its nearest neighbours. PAWN, on the other hand, uses the local knowledge of its neighbours, i.e., the information provided by the base station about its neighbourhood, for authentication. Each cluster head only needs to look-up its table to validate/invalidate a session initiation request. The presence of λ_i enables each cluster head to enter into a four phase handshake negotiation for mutual authentication.

4.4. Average Energy Consumption

PAWN algorithm uses somewhat similar centralized approach for cluster formation as LEACH-C protocol. However, LEACH-C does not provide a secured transmission of data and is susceptible to a wide range of malicious attacks. In Figure 5, the average energy consumed by PAWN algorithm is shown in comparison with LEACH-C.

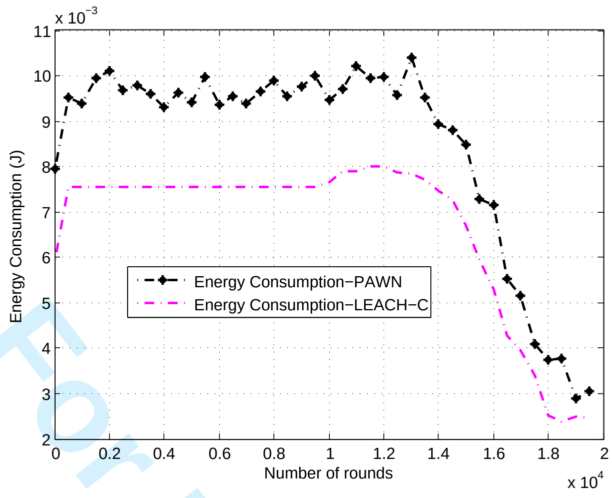


Figure 5. Average Energy Consumption

As depicted in Figure 5, the average energy consumed by PAWN is slightly higher (in order of millijoule) than LEACH-C in most of the rounds. However, the slight increase in energy consumption is negligible keeping in view that PAWN provides a complete set of operations for authentication and data protection. LEACH-C, on the other hand, does not provide any secured features and the election of cluster heads is too complex. LEACH-C elects cluster heads and forms clusters using the simulated annealing algorithm by solving the N-P hard problem of finding optimal clusters [13]. This technique of cluster formation and cluster head election incurs excessive delay and consumes too much energy. On the other hand, PAWN uses a simple technique for cluster head selection which requires a much smaller amount of energy and at the same time provides a robust defence against various malicious attacks. The novelty and simplicity of cluster head election technique is the driving force behind energy minimization in PAWN algorithm.

4.5. Average Network Throughput

The average network throughput is calculated as the ratio of total number of packets successfully received at the base station to the total number of transmitted packets. The average network throughput of PAWN is shown in Figure 6. In absence of an authentication scheme, the number of packets transmitted to a base station is much higher because each scheme incurs higher processing overhead and has a much higher buffer occupancy. These two factors result in packet processing delay, packet loss, queuing delay and increase the number of retransmission attempts. PAWN provides authentication not only at the cluster head election time but also during cluster formation. Even with the provisioning of authentication at two levels, the percentage of packets successfully received at the base station is much higher in most of the rounds. The average network throughput of PAWN reaches up to 75% in most of the rounds despite its operation in hostile, remote locations and communication over error-prone noisy links. These factors cause too much distortion and attenuation of signals. Although PAWN provides a robust and defensive solution to the underlying network, the nature of deployed region and quality of the communication links also play an important role in packet loss and Quality of Service (QoS) degradation.

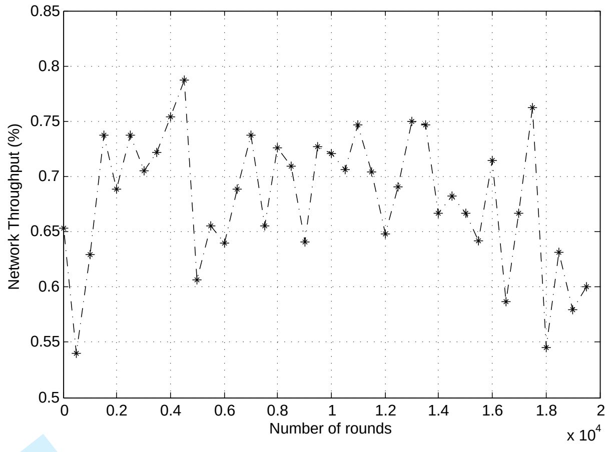


Figure 6. Average Network Throughput

5. CONCLUSION

In this paper, we have proposed a lightweight mutual authentication scheme for a cluster-based hierarchical WSN. The proposed scheme utilizes the energy-efficient nature of cluster-based hierarchical architecture for energy minimization and the lightweight features of the mutual authentication scheme for security provisioning. Our proposed algorithm, Payload-based mutual Authentication for Wireless Sensor NetworkS (PAWN), operates in two phases. First, an optimal percentage of cluster heads are elected and advertised by the base station, and broadcast nomination packets are used to ensure that only legitimate cluster heads receive those advertisements. Second, each cluster head uses a simple four phase mutual handshaking to authenticate any neighbouring node wishing to be a member node. If handshaking is successful, each cluster head forms a cluster and the authenticated neighbouring node becomes a member node of it. Next, each cluster head collects data from its member nodes and transmits to a centralized base station. We have compared PAWN against state-of-the-art LEACH-C and SecLEACH in terms of various performance metrics such as robustness against attacks, handshake duration and average energy consumption. Unlike LEACH-C, PAWN is highly robust and resilient to various attacks such as replay, DoS, Sybil and resource exhaustion. SecLEACH, on the other hand, can detect only a handful of such attacks. The average handshake duration of PAWN is much shorter as compared to SecLEACH. Depending on the cluster size, PAWN can establish an authenticated session in a time almost half of that required by SecLEACH. LEACH-C, on the other hand, lacks security features and as such does not establish an authenticated session. PAWN is also energy-efficient as compared to the existing schemes because it consumes much smaller energy despite provisioning of robust security features. We have implemented PAWN using AES-128 bit, however, it would be interesting to see its performance using other encryption algorithms. Moreover, apart from symmetric encryption, PAWN is open to public key encryption techniques provided that energy-efficient nature of the scheme and the resource-constrained nature of the nodes remain intact. Presently, PAWN performs base station/cluster head authentication using tokens, however, the same task can be achieved using some other alternative techniques to further minimize the energy consumption. PAWN uses the centralized cluster-based hierarchical architecture for cluster head election. In future, we are interested to see the performance of PAWN for a randomly distributed cluster-based hierarchy, similar to LEACH.

REFERENCES

1. P. Rawat, K. D. Singh, H. Chaouchi, and J. M. Bonnin, “Wireless sensor networks: a survey on recent developments and potential synergies,” *The Journal of supercomputing*, vol. 68, no. 1, pp. 1–48, 2014.

2. J. A. Stankovic, A. D. Wood, and T. He, “Realistic applications for wireless sensor networks,” in *Theoretical Aspects of Distributed Computing in Sensor Networks*. Springer, 2011, pp. 835–863.
3. Y. Yu, K. Li, W. Zhou, and P. Li, “Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures,” *Journal of Network and computer Applications*, vol. 35, no. 3, pp. 867–880, 2012.
4. S. T. Zargar, J. Joshi, and D. Tipper, “A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks,” *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 4, pp. 2046–2069, 2013.
5. M. A. Jan, “Energy-efficient routing and secure communication in wireless sensor networks,” Ph.D. dissertation, 2016.
6. M. A. Jan, P. Nanda, X. He, Z. Tan, and R. P. Liu, “A robust authentication scheme for observing resources in the internet of things environment,” in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2014 IEEE 13th International Conference on*. IEEE, 2014, pp. 205–211.
7. W. Xiao-yun, Y. Li-zhen, and C. Ke-fei, “Sleach: Secure low-energy adaptive clustering hierarchy protocol for wireless sensor networks,” *Wuhan University Journal of Natural Sciences*, vol. 10, no. 1, pp. 127–131, 2005.
8. D. Puthal, S. Nepal, R. Ranjan, and J. Chen, “A dynamic prime number based efficient security mechanism for big sensing data streams,” *Journal of Computer and System Sciences*, 2016.
9. Y.-H. Lee, J.-H. Kang, and S.-J. Lee, “A specification-based intrusion detection mechanism for leach protocol,” *The Journal of Korean Institute of Communications and Information Sciences*, vol. 37, no. 2B, pp. 138–147, 2012.
10. M. A. Jan, P. Nanda, X. He, and R. P. Liu, “A sybil attack detection scheme for a centralized clustering-based hierarchical network,” in *Trustcom/BigDataSE/ISPA, 2015 IEEE*, vol. 1. IEEE, 2015, pp. 318–325.
11. W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” in *Proceedings of the 33rd annual Hawaii international conference on System sciences*. IEEE, 2000, pp. 1–10.
12. M. A. Jan, P. Nanda, X. He, and R. P. Liu, “Enhancing lifetime and quality of data in cluster-based hierarchical routing protocol for wireless sensor network,” in *2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC)*. IEEE, 2013, pp. 1400–1407.
13. W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” *Wireless Communications, IEEE Transactions on*, vol. 1, no. 4, pp. 660–670, 2002.
14. M. A. Jan, P. Nanda, and X. He, “Energy evaluation model for an improved centralized clustering hierarchical algorithm in wsn,” in *Wired/Wireless Internet Communication*. Springer, 2013, pp. 154–167.
15. M. M. Morshed and M. R. Islam, “Cbsrp: Cluster based secure routing protocol,” in *Advance Computing Conference (IACC), 2013 IEEE 3rd International*. IEEE, 2013, pp. 571–576.
16. Y. Zhang, X. Li, J. Yang, Y. Liu, N. Xiong, and A. V. Vasilakos, “A real-time dynamic key management for hierarchical wireless multimedia sensor network,” *Multimedia tools and applications*, vol. 67, no. 1, pp. 97–117, 2013.
17. E. Shaaban *et al.*, “Enhancing s-leach security for wireless sensor networks,” in *Electro/Information Technology (EIT), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1–6.
18. M. Alshowkan, K. Elleithy, and H. AlHassan, “Ls-leach: a new secure and energy efficient routing protocol for wireless sensor networks,” in *Distributed Simulation and Real Time Applications (DS-RT), 2013 IEEE/ACM 17th International Symposium on*. IEEE, 2013, pp. 215–220.
19. R. Dutta, E.-C. Chang, and S. Mukhopadhyay, “Efficient self-healing key distribution with revocation for wireless sensor networks using one way key chains,” in *Applied Cryptography and Network Security*. Springer, 2007, pp. 385–400.
20. P. Schaffer, K. Farkas, Á. Horváth, T. Holczer, and L. Buttyán, “Secure and reliable clustering in wireless sensor networks: a critical survey,” *Computer Networks*, vol. 56, no. 11, pp. 2726–2741, 2012.
21. J. Long, A. Liu, M. Dong, and Z. Li, "An energy-efficient and sink-location privacy enhanced scheme for wsns through ring based routing." *Journal of Parallel and Distributed Computing*, 81:47–65, 2015.
22. M. A. Jan, P. Nanda, X. He, and R. P. Liu, "A sybil attack detection scheme for a forest wildfire monitoring application." *Future Generation Computer Systems*, 2016.
23. L. B. Oliveira, A. Ferreira, M. A. Vilaça, H. C. Wong, M. Bern, R. Dahab, and A. A. Loureiro, “SecleachâĂŤon the security of clustered sensor networks,” *Signal Processing*, vol. 87, no. 12, pp. 2882–2895, 2007.
24. Y. Cho, G. Qu, and Y. Wu, “Insider threats against trust mechanism with watchdog and defending approaches in wireless sensor networks,” in *Security and privacy workshops (SPW), 2012 IEEE Symposium on*. IEEE, 2012, pp. 134–141.
25. L. Yang, “Centralized security protocol for wireless sensor networks,” 2011.
26. E. Rescorla and B. Korver, “Guidelines for writing rfc text on security considerations,” 2003.
27. M. Dong, K. Ota, L. T. Yang, A. Liu and M. Guo, "Lscd: A low-storage clone detection protocol for cyber-physical systems." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(5):712–723, 2016.
28. J. Wu, K. Ota, M. Dong, and C. Li, "A hierarchical security framework for defending against sophisticated attacks on wireless sensor networks in smart cities." *IEEE Access*, 4:416–424, 2016.