ISSN:2348-2079

Volume-6 Issue-2

International Journal of Intellectual Advancements and Research in Engineering Computations

Enhanced Selective Encryption Method for Big Data Sensing Stream Using Hash Chain Algorithm

[1] Savitha S, [2]Logeswaran K, [3]Abinaya N, [4]Dhinakaran R, [5]Mahalakshmi S, [6]Dharamesh B

[1]Assistant Professor, Department of Computer Science and Engineering, K.S.R. College of Engineering

[2]Assistant Professor, Department of Computer Science and Engineering, Kongu Engineering College

[3-6]UG Students, Department of Computer Science and Engineering, K.S.R. College of Engineering

E-mail id: [1]infosavi@gmail.com, [2]klogesbtech@gmail.com, [3]selvaabi20@gmail.com, [4]dhinacse020@gmail.com, [5]mahashanmugam97@gmail.com, [6]dhamub95665@gmail.com

***Abstract*—Resource constrained sensing devices are used to build and deploy self-organizing wireless sensor networks for a variety of critical applications such as smart cities, smart health, precision agriculture and industrial control systems. A Data Stream Manager at the server collects the data streams often called big data to perform real time analysis and decision-making for these critical applications. A malicious adversary may access or tamper with the data in transit. One of the challenging tasks in such applications is to assure the trustworthiness of the collected data so that any decisions are made on the processing of correct data. Frequent pattern/itemset mining is a fundamental problem for many domains, thus has a number of applications. In this paper, we proposed a novel scheme to secure a multihop network programming protocol through the use of multiple one-way hash chains and an index-based algorithm that addresses the challenge and provides the exact answer. The one way hash chains scheme is shown to be lower in computational, power consumption and communication costs yet still able to secure multihop propagation of program images. The bigdata objects in these applications are often generated in a streaming fashion. The CP-Graph approach, a hybrid index of graph and inverted file structures. The CP-Graph computes the count of a given pattern and updates the answer while pruning unnecessary patterns.**

Keywords – Wireless sensor networks, Bigdata, Multihop network, one way hash chain, index based algorithm, CP-Graph.

I. INTRODUCTION

Network programming is becoming necessary for Wireless Sensor Networks (WSNs) because there is always a need to fix bugs in program images or insert new functionalities after WSN is deployed in an evolving, dynamic environment. Early network programming protocols concentrated on reliable program image dissemination and minimal end to end update latency using distribution methods having epidemic like characteristics. This method does not provide any authentication or security mechanisms. The absence of authentication of the broadcast of program image means that a malicious node could install arbitrary program images in the sensor nodes. An adversary could just capture one sensor node, inject malicious program images into the network, and thereby take control of the entire WSN. The main approach of this paper is to present a design and implementation for a new scheme to verify the authenticity and integrity of program updates in network programming protocols. The work is motivated by the following challenges. First as a significant class of WSN sensor nodes is resource impoverished, traditional cryptographic schemes are impractical. For example, the Tmote [6] has 10 KB RAM, 48 KB flash memory, 1 MB storage, and 250 kbps communication bandwidth. This is barely sufficient to execute traditional asymmetric cryptography (e.g., RSA or Diffie and Hellman). It is important that a security scheme for WSNs should be low in power consumption and have low computational overhead. The second challenge arises from the open wireless environment in which WSNs are typically deployed. Since program updates are also broadcast through the wireless medium, an adversary can readily intercept the program updates and attempt to forge a malicious program image while avoiding detection. Another complication arises from the way that sensor nodes are deployed. Tmote [6] has barely sufficient to execute traditional asymmetric cryptography (e.g., RSA or Diffie and Hellman). It is important that a security scheme for WSNs should be low in power consumption and have low computational overhead. The second challenge arises from the open wireless environment in which WSNs are typically deployed. Program updates are also broadcast through the wireless medium; an adversary can readily intercept the program updates and attempt to forge a malicious program image while avoiding detection. Another complication arises from the way that sensor nodes are deployed. It is possible to physically secure a sensor network node against theft or tampering by a variety of

means, but these physical approaches are outside the scope of this paper. This Paper presents a scheme that is resilient against brute-force attack and node compromise.

- To introduce a novel scheme to secure a multihop network programming protocol through the use of multiple oneway hash chains.
- To apply the scheme such that it should be lower in computational, power consumption, and communication costs yet still able to secure multihop propagation of program images.
- To demonstrate the use of this scheme and provide some results using a network programming protocol.
- To provide a cost-effective security scheme for network programming. The digital signature is not required to bootstrap the network programming process. Instead, multiple distinct one-way key chains are distributed to the nodes based on their hop distances to base station for packet authentication. This obviates the need for digital signature, resulting in efficient use of the limited computational power of each sensor node.

- Time synchronization among sensor nodes is not required, and consequently, the packets of a program update can be verified immediately as they are received to make the scheme extensible to other broadcast application in WSNs. To the best of knowledge, all existing methods to securing network programming [10] are data-dependent. The implication of this is that the number of packets in a base station broadcast must be known beforehand.

II. RELATED WORKS

The Dynamic Behavior of a Data Dissemination Protocol for Network Programming at Scale [3] the authors Jonathan W. Hui and David Culler to support network programming, presented Deluge, a reliable data dissemination protocol for propagating large data objects from one or more source nodes to many other nodes over a multihop, wireless sensor network. Wireless sensor networks (WSNs) represent a new class of computing with large numbers of resource-constrained computing nodes cooperating on essentially a single application. For example, collectively programming all nodes before deployment could put thousands of nodes within communication range, making suppression of protocol messages essential. Third, complete reliability is required since every byte must be correctly received by all nodes that need reprogramming, even in the presence of high loss rates and evolving link qualities common to WSNs. Fourth, propagation must be a continuous effort to ensure that all nodes receive the newest code since network membership is not static: nodes come and go due to temporary disconnections, failure, and network repopulation. The dissemination process should require a minimal amount of time, reducing any service interruptions to a deployed application and the debugging and testing cycle. Data dissemination in wireless networks, naive retransmission of broadcasts can lead to the broadcast storm problem, where redundancy, contention, and collisions impair performance and reliability [6]. The authors discuss the need to have a controlled retransmission scheme and propose several schemes, such as probabilistic and location-based methods. The experiments conducted by Ganesan et al. identify several interesting effects at the link-layer, notably highly irregular packet reception contours, the likeliness of asymmetric links, and the complex propagation dynamics of simple protocols [7]. Scalable Reliable Multicast (SRM) is a reliable multicast mechanism built for wired networks [8], using communication suppression techniques to minimize network congestion and request implosion at the server. The epidemic property is important since WSNs experience high loss rates, asymmetric connectivity, and transient links due to node failures and repopulation.

They presented an incremental network programming mechanism which sends the new version of program by transmitting the difference of the two program images. Unlike previous approaches, they generated the program code difference by comparing the program code in block level without any prior knowledge of the program code structure. This gave a general solution that can be applied to any hardware platform. They used the Rsync algorithm [10] to generate the difference. The Rsync algorithm finds the shared code blocks between the two program images and allows us to distribute just the key changes of the program. Originally, the Rsync algorithm was made for computationally powerful machines exchanging the update of binary files over a low-bandwidth communication link. They tuned the Rsync algorithm for wireless sensor network programming. Deepak et al. proposed Selective Encryption (SEEN) method to maintain confidentiality levels of big sensing data streams with data integrity. In SEEN, a Data Stream Manager (DSM) independently maintains an intrusion detection and shared key management as two major components. Our method has been designed based on a symmetric key block cipher and multiple shared keys use for encryption. By employing the cryptographic function with selective encryption, the DSM efficiently rekeys without retransmissions. The rekeying process never disrupts ongoing data streams and encryption/decryption. SEEN supports the source node authentication and shared key recovery without incurring additional overhead. SEEN provides a significant improvement in processing time, buffer requirement and prevents data confidentiality and integrity from malicious attackers. SEEN method is to improve the efficiency of symmetric-key encryption. In addition planning to introduce the access control model over big sensing data streams, which will give access to the end user or query processor based on the data levels.

III. PROPOSED SOLUTION

A. Data Stream Classification

Data stream classification has been a widely studied research problem in recent years. The dynamic and evolving nature of data streams requires efficient and effective techniques that are significantly different from static data classification techniques. Two of the most challenging and well studied characteristics of data streams are its infinite length and concept drift. Data stream classification poses many challenges to the data mining community. Four major challenges, namely, infinite length, concept-drift, concept-evolution, and feature-evolution. Since a data stream is theoretically infinite in length, it is impractical to store and use all the historical data for training. Also a data stream is a fast and continuous phenomenon, it is assumed to have infinite length. The most obvious alternative is an incremental learning technique. Concept drift is a common phenomenon in data streams, which occurs as a result of changes in the underlying concepts. Feature evolution is a frequently occurring process in many streams, such as text streams, in which new features appear as the stream progresses. Most existing data stream classification techniques address only the first two challenges, and ignore the latter two. In addition, concept-drift occurs in the stream when the underlying concepts of the stream change over time. There are two other significant characteristics of data streams, namely, concept-evolution and feature evolution, that are ignored by most of the existing techniques. Concept evolution occurs when new classes evolve in the data. For example, consider the problem of intrusion detection in a network traffic stream. If we consider each type of attack as a class label, then concept-evolution occurs when a completely new kind of attack occurs in the traffic. The problem of concept-evolution is addressed in only a very limited way by the currently available data stream classification techniques. We investigate this problem in this paper, and propose improved solutions. Our current work also addresses the feature-evolution problem in data streams, such as text streams, where new features (words) emerge and old features fade away. We aim at four major contributions in enhanced class detection for data streams. First, we propose a flexible decision boundary for outlier detection by allowing a slack space outside the decision boundary. This space is controlled by a threshold, and the threshold is adapted continuously to reduce the risk of false alarms and missed novel classes. Second, we apply a probabilistic approach to detect novel class instances using the discrete Gini Coefficient. With this approach, we are able to distinguish different causes for the appearance of the outliers, namely, noise, concept drift, or concept-evolution. We derive an analytical threshold for the Gini Coefficient that identifies the case where a novel class appears in the stream. We empirically show the effectiveness of this approach. Third, we apply a graph-based approach to detect the appearance of more than one novel classes simultaneously, and separate the instances of one novel class from the others. Finally, our proposed approach addresses the feature-evolution problem on top of the enhancements discussed above. To the best of our knowledge, this is the first work that proposes these advanced techniques for novel class detection and classification in data streams and addresses feature evolution. We apply our technique on a number of data streams including Medical messages, and also image classification is included to complement our work. The novel class detection process consists of three steps. First, a decision boundary is built during training. Second, test points falling outside the decision boundary are declared as outliers. Finally, the outliers are analyzed to see if there is enough cohesion among themselves (i.e., among the outliers) and separation from the existing class instances. But none of the existing system address the feature-evolution problem.

B. Outlier Detection

When the data arrived is more and the classes formed out of them increases the problem is termed as infinite length problem. This is to be avoided. Each incoming instance in the data stream is first examined by an outlier detection module to check whether it is an outlier. If it is not an outlier, then it is classified as an existing class using majority voting among the classifiers in the ensemble. If it is an outlier, it is temporarily stored in a buffer. When there are more new classes formed, then the classes with less content are discarded so that the number of classes is maintained within a given limit and this avoids the infinite problem.

C. Concept drift identification

The words and the category to which it belongs are added in the ‘category’ table. A client application is developed in which the text content is sent to the server application which updates the incoming message. The words are extracted and the words fell in the given category are identified and counted. If there are more words in the category and the word count reduced in the successive incoming messages, then the concept is found to be reduced and when the number of words reduced to zero, the concept is said to be drifted. The number of observation time count is set so that when the number of word count is zero for that given number of time, then the concept is said to be drifted.

D. Enhanced Class Detection

During the concept evolution phase, the novel class detection module is invoked. If a novel class is found, the instances of the novel class are tagged accordingly. Otherwise, the instances in the buffer are considered as an existing class and classified normally using the ensemble of models. The words occurred frequently but not matched with any of the category available, and then the word is considered to be fallen in new class.

Algorithm Used

Adjust-Threshold(x, OUTTH)

Input: x, OutTh

Which are most recent labeled instance and OutTh is current outlier threshold

Process:

i. Populate the class labels
ii. Check the incoming X data is matched with any of the class
iii. If not fallen in any of the class, then new class is said to be occurred. OutTh is increased with a slack variable .

Output: OUTTH

New outlier OutTh threshold.

E. Hash Chain Model

key agreement protocols, all communication entities are in volved to determine session keys. The most commonly used key agreement protocol is Diffie-Hellman (DH) key agreement protocol. In DH protocol, the session key is determined by exchanging public keys of two communication entities. Since the public key itself does not provide any authentication, a digital signature can be attached to the public key to provide authentication. However, DH public key distribution algorithm can only provide session key for two entities; not for a group more than two members.There are three problems associated with this approach. First, signature verification incurs a computation overhead, which is a concern given the limited power resources of sensor nodes. Second, memory usage required to include the signature is significantly higher.

Third, these approaches require the number of data packets to be known in advance of the network programming process, since they construct a hash chain which starts from the last data packet and works back to the first. This requirement limits the expansion of these approaches to other broadcast applications where the number of broadcast packets is not known in advance.The proposed system is an authentication scheme to secure multihop network programming with multiple oneway hash chains. Instead of the expensive asymmetric cryptographic primitives used in much prior work, the scheme employs only symmetric cryptographic primitives, in a circular geographic node deployment model. The proposed system discussed the possible attacks an adversary could mount on the scheme and provided simple and effective counter measures against them. Finally, it provided a comprehensive performance evaluation of the scheme in terms of end-to-end latency and power consumption, which it believes is the first power consumption evaluation of a security scheme for network programming protocols. The proposed system is group key transfer protocol using secret sharing scheme. In this system, each user needs to register at KGC to subscribe the group key transfer service and to establish a secret with KGC. Thus, a secure channel is needed initially to share this secret with each user. Later, KGC can transport the group key and interact with all group members in a broadcast channel. The confidentiality of group key distribution is information theoretically secure; that is, the security of this transfer of group key to each group member does not depend on any computational assumption. The authentication of the group key is achieved by broadcasting a single authentication message to all group members. The proposed system has following advantages.

1. Every user needs to register at a trusted KGC initially and pre-share a secret with KGC.
2. KGC broadcasts group key information to all group members at once.
3. The confidentiality of our group key distribution is information theoretically secure.
4. We provide group key authentication.
5. Security analysis for possible attacks is included. through the use of multiple one-way hash chains, multihop network programming protocol is secured.
6. Lower in computational, power consumption, and communication costs yet still able to secure multihop propagation of program images.
7. A digital signature is not required to bootstrap the network programming process.
8. Multiple distinct one-way key chains are distributed to the nodes based on their hop distances to base station for packet authentication. This obviates the need for digital signature, resulting in efficient use of the limited computational power of each sensor node.
9. No time synchronization among sensor nodes is required, and consequently, the packets of a program update can be verified immediately as they are received. The scheme is extensible to other broadcast application in WSNs.

1. Initialization of KGC.

The KGC randomly chooses two safe primes p and q (i.e., primes such that p' = (p-1)/2 and q' = (q-1)/2 are also primes) and compute n =pq. n is made publicly known.

2. User Registration.

Each user is required to register at KGC for subscribing the key distribution service. The KGC keeps tracking all registered users and removing any unsubscribed users. During registration, KGC shares a secret, (x_i, y_i), with each user U_i, where $x_i, y_i \mathbb{Z}_n^*$

3. Group key generation and distribution.

Upon receiving a group key generation request from any user, KGC needs to randomly selects a group key and access all shared secrets with group members. KGC needs to distribute this group key to all group members in a secure and authenticated manner. All communication between KGC and group members are in a broadcast channel. For example, we assume that a group consists of t members, {U1, U2, . . . ,Ut),

and shared secrets are (x_i, y_i), for i = 1, . . . , t. The key generation and distribution process contains five steps.

Step 1. The initiator sends a key generation request to KGC with a list of group members as {U1, U2, . . . ,Utg}.

Step 2. KGC broadcasts the list of all participating members, {U1, U2, . . . ,Ut}, as a response.

Step 3. Each participating group member needs to send a random challenge, Ri belongs to$\mathbb{Z}_n^*$, to KGC.

Step 4. KGC randomly selects a group key, k, and generates an interpolated polynomial f(x) with degree t to pass through (t + 1) Ri), for i =1, . . . , t. KGC also computes tpoints, (0, k) and (xi, yi additional points, Pi, for i = 1, . . . , t, on f(x) and Auth = h(k,U1, . . . , Ut, R2, . . .,Rt, P1, . . . , Pt), where h is a one-way hash function. All computations on f(x) are over $\mathbb{Z}_n^*$. KGC broadcasts (Auth, Pi), for i = 1, . . . , t, to all group members. All computations are performed in $\mathbb{Z}_n^*$

Step 5. For each group member, Ui, knowing the shared Ri), and t additional public points, Pi, for i = 1, . . . ,secret, (xi, yi t, on f(x), is able to compute the polynomial f(x) and recover the group key k = f(0). Then, Ui computes h(k, U1, . . . , Ut;R1, . . ., Rt, P1, . . . , Pt) and checks whether this hash value is identical to Auth. If these two values are identical, Ui authenticates the group key is sent from KGC.

F. Multi Hop Hash Chain Model

The scheme has a hierarchical organization of program images. The scheme works at the packet level and consists of two phases: first, initialization and key predistribution, and second, packet preprocessing and verification. The notations used in this study are provided in Table I.The following modules are present in the study.

1. Phase 1: Initialization and Key Predistribution
2. Phase 2: Packet Preprocessing and Verification
   - Packet Preprocessing
   - Packet Verification
   - Justification of Key Update Segment and Packet Authentication Segment

Phase 1: Initialization and Key Predistribution

The scheme employs multiple one-way hash chains to secure the Deluge protocol. Hash chains are based on a function H with the property that its computation is easy, whereas its inverse H^{-1} is extremely difficult to compute. A hash chain with length L is generated by applying H to an initial element repeatedly for L times (i.e., $0 \le i \le L$, $1 \le j \le S$ $K_{i,j}=H(K_{i+1,j})$, as shown in Fig. 1). The last value after H has been applied L times is called the committed value of the hash chain (see Fig. 1).

Before the sensor nodes are deployed, the base station constructs S hash chains. It generates S distinct random seed numbers and computes a one-way hash chain withlength of L + 1 starting from each seed, as shown in Fig. 1 (the (L - i + 1)th output of hash function derived from jth random seed number is denoted as $K_{i,j}$).

Sensor nodes are divided into S groups according to their hop distance to the base station (see Fig. 2). The committed value of the ith key chain ($K_{0,i}$ in Fig. 1), corresponding to the hop distance, is predistributed to nodes in the ith hop group before they are deployed. A secure method suitable for key distribution before node. 1. In the study, the words 'hash chain' and 'key chain' are used interchangeably.

TABLE I
The Notations and Meanings

Notation	Meaning
H(B)	Hash of B
A||B	Message A concatenated with message B
K_a	Shared key between node a and base station
Ek_a (M)	Encryption of message M with symmetric key K_a
Dk_a(M)	Decryption of M with symmetric key K_a
A:	The operation after : occurs in node A
A→ B:P	Node A unicasts a packet P to node B
A→ *:P	Node A broadcasts a packet P
M_A=> M_B	Message$M_B$is copied to message M_A
R	The transmission range of sensor nodes and base station
ϒA	The distance between node A and the base station
ϒAB	The distance between node A and B
ϒD_i	The radius of deployment circle for the i^{th} group
S	The number of hopes in which a secure program image will propagate in our scheme
N	The number of nodes in WSN
L	The maximum number of packets that the base station will need to broadcast

deployment is the Message-In-Bottle (MIB) protocol [24].[2] We adopt this approach to the distribution of the commitment values because

- Predistribution will not incur the overhead of a Diffie-Hellman key exchange protocol [8], and
- Key agreement between the base station and all sensor nodes would requireDiffie-Hellman exchanges for each node if the Diffie-Hellman approach is adopted.

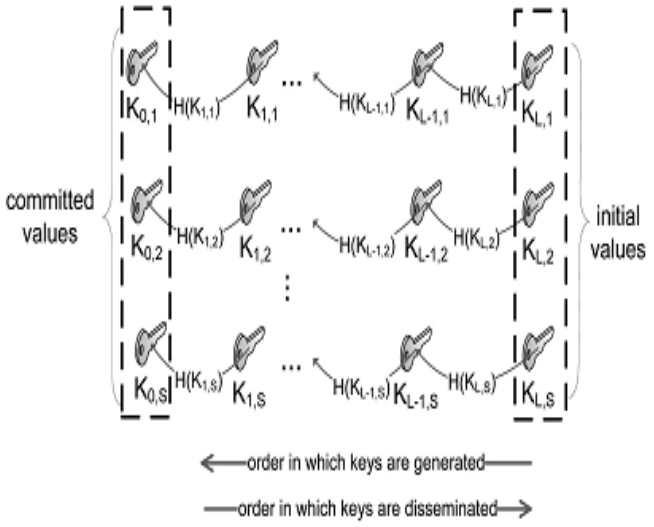


Figure 1. Establishment of multiple one-way key chains.

Phase 2: Packet Preprocessing and Verification

After the node deployment, when the base station has a program update to be broadcast, the program image is divided into fixed-size packets (e.g., the program image is divided into P_0,P_1,….P_{L-1} in Fig. 4). The base station preprocesses these packets so that the sensor nodes can verify the packets accordingly.

Packet Preprocessing

First, it describes the packet preprocessing of the very first packet of program image (P_0). The committed value of the first key chain ($K_{0,1}$ in Fig. 1) is used to encrypt the next key element in the order of key dissemination ($K_{1,1}$ in Fig. 1). The encrypted result ($EK_{0,1}(K_{1,1})$) is the key update segment for the first hop group. Then, $K_{1,1}$ is concatenated with P_0 and the result is hashed, yielding the packet authentication segment ($H(P_0||K_{1,1})$).The key update segments and packet authentication segments for the successive hop groups are generated in the same way using their corresponding key chains. Finally, all these segments are concatenated with P_0 as shown in Fig. 3, giving the first packet to be transmitted. The way in which the key update and packet authentication segments are concatenated with the data packet is used in a countermeasure against tunnel attack, further discussed. The above packet preprocessing procedure is repeated for successive packets in the image to be broadcast. The packet preprocessing result ($Pi^{'}$) for the i^{th} ($\square i \leq 0 \leq L$) packet (P_i) is propagated to S hops is

$$Pi^{'} = P_i \,||\, T_{i,S} \,||\, U_{i,S} \,||\, ... \,||\, T_{i,1} \,||\, U_{i,1},$$

where

$$T_{i,j} = H(P_i||\ K_{i+1,j}),$$

And

$$U_{i,j} = EK_{i,j}\ (K_{i+1,j}).$$

The complete mechanism of packet preprocessing is illustrated in Fig. 4, while the complete verification of i^{th} packet in node A (A ϵ Gj) is shown in Algorithm 1.

Algorithm.

The $i^{th}(1 \leq i \leq L)$ packet(P_i) verification in node(A) where A is inGj($1 \leq j \leq S$)

1: base station: $Key\_Section\_ij <= EK_{i-1,j}(Ki,j)$
2: base station: $P_{i\_Ki,j} <= H(P_i||K_{i,j})$
3: base station: $Pi <= P_i||P_{i\_Ki;j} \,||\, Key\_Section\_ij$
4: base station $\rightarrow$* : $Pi^{'}$
5: Pi' has been propagated by j - 1 nodes
6: A: receive Pi' $\square$A ϵGj group
7: if $H(DK_{i,j}(Key\_Section\_ij)) = K_{i,j}$ then
8: A : $K_{i+1,j}$ ($DK_{i,j}$ (Key_Section_ij)
9: A: remove Key_Section_ij from Pi'
10: if $H(P_i||KA) = P_{i_Ki,j}$then
11: A: remove $P_{i_Ki,j}$from Pi'
12: A: accept P_i
13: A: advertise P
14: else
15: A: drop Pi'
16: end if
17: else
18: A: drop Pi'
19: endif
20: if P_i is requested then
21: A $\rightarrow$* : $Pi^{'}$
22:endif

Packet Verification

In this section, the packet verification for the first data packet destined to the first hop group will be described. The verification of subsequent packets in the other hop groups uses the same procedure with keys corresponding to those that were used in the packet preprocessing.

After the preprocessing of a packet and the respective concatenation (i.e., $P_0' = P_0||T_{0,1}||U_{0,1}$), P_0' is transmitted to nodes in the first hop group. After retrieving the correct group information from P_0' (i.e., the sensor node parses the right fields $T_{0,1}$ and $U_{0,1}$), the sensor nodes verify the key update segment ($U_{0,1}$) and packet authentication ($T_{0,1}$) segments as follows:

Key update segment

The sensor node decrypts $U_{0,1}$ with $K_{0,1}$, i.e., $DK_{0,1}$ ($U_{0,1}$). The sensor node checks if $H(DK_{0,1},U_{0,1}))$ isthesameas $K_{0,1}$ (line 7 in Algorithm). If it is, the authenticity of $U_{0,1}$ is

ensured and $K_{0,1}$ is replaced by $K_{1,1}$ for the next packet verification (line 8 in Algorithm). Otherwise, this packet is discarded (line 17 in Algorithm).

Packet authentication segment

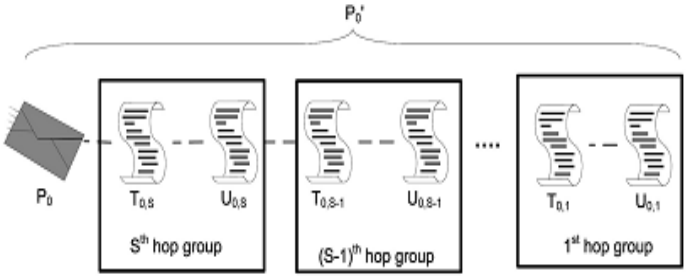
After the sensor node retrieves the authenticated $K_{1,1}$ from $U_{0,1}$, it performs $H(P_0||K_{1,1})$ to see if it matches $T_{0,1}$ (line 10 in Algorithm). If it does, the integrity of packet is assured (line 12 in algorithm). Otherwise, this packet is discarded (line 14 in Algorithm).

Justification of Key Update Segment and Packet Authentication Segment

Key update segment

This segment is needed for three reasons. First, the key in jth group can be dynamically updated by packets (line 8 in Algorithm) without time synchronization between base station and sensor nodes. With the key updated, compromised nodes cannot launch a replay attack (lines 17-19 in Algorithm).

Second, since different hop groups of nodes employ different one-way key chains, attacks on compromised nodes from different groups can be prevented (especially a man-in-the-



middle attack). Third, the encrypted part contains the next element of the respective key in the dissemination order, so the updated key can be verified as a consequence of the one-way property of the hash chain (line 7 in Algorithm).Packet authentication segment . Integrity of the packet content is ensured with this segment. Had Pi been altered, the verification of this segment (line 10 in Algorithm) would fail (lines 14-16 in Algorithm). This segment serves as a "signature" from the base station. However, due to the absence of non repudiation in symmetric cryptography, this signature could be forged by a compromised node. However, the difficulty of forging this signature is made great enough by selecting appropriate design parameters (e.g., size of this segment).

First design a simplified mechanism to determine the number of neighboring nodes for any given node. Within time Tv, the given node crosses through an area and meets a number of neighbors N. Since mobile nodes are assumed uniformly distributed in the network, we may approximate N by

$$N = (\pi r^2 + 2rvT_v)p,$$

Where r denotes the transmission range of nodes, *v* is the velocity, and *p* is the density of nodes in the network. Based on the obtained number of neighboring nodes N, we can on firm the value of threshold K.

IV. RESULTS

The following Table II describes experimental result for existing system secure transmission node analysis. The table contains number of time slot interval and given time interval to calculate average numbers of send transmission node details are shown.

Table II

SEEN -Secure Transmission

S.no	Number of Time slot (m)	Ratio of secure transmission node
1	10	0.43
2	20	0.52
3	40	0.61
4	60	0.69
5	80	0.74
6	100	0.80
7	120	0.86
8	140	0.90
9	150	0.93
10	160	0.97

The following Figure 1 describes experimental result for existing system secure transmission node analysis. The table contains number of time slot interval and given time interval to calculate average numbers of send transmission node details are shown.

The following Table 2 describes experimental result for proposed system secure transmission node analysis. The table contains number of time slot interval and given time interval to calculate average numbers of send transmission node details are shown

FIGURE 2

SEEN Secure Transmission

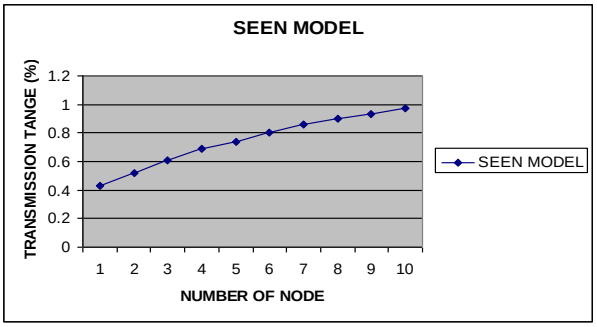


TABLE III

HASH Secure Transmission

S.no	Number of Time slot (m)	Ratio of secure transmission node
1	10	0.48
2	20	0.57
3	40	0.66
4	60	0.72
5	80	0.77
6	100	0.83
7	120	0.89
8	140	0.92
9	150	0.95
10	160	0.98

The following Figure 3 describes experimental result for proposed system secure transmission node analysis. The table contains number of time slot interval and given time interval to calculate average numbers of send transmission node details are shown.

The following Figure 3 describes experimental result for differences existing system (SEEN) and proposed system (HASH) Secure transmission communication node analysis. The table contains number of time slot interval and given time interval to calculate average numbers of send secure communication transmission node details are shown.

FIGURE3

Hash Chain Secure Transmission

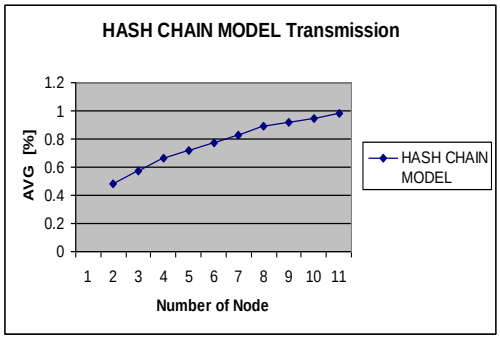


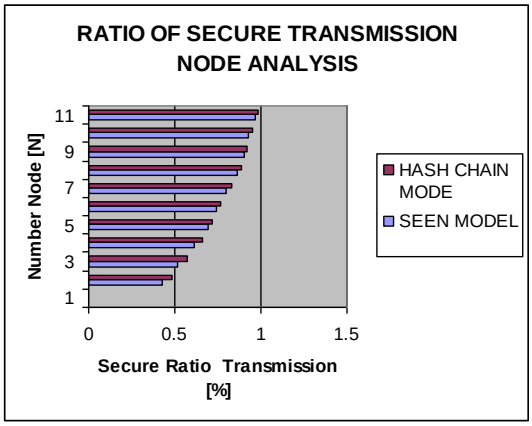
TABLE IV

Comparisons for SEEN and HASH Secure Transmission

S.no	Number of Time slot (m)	Ratio of secure transmission node	
		SEEN	HASH
1	10	0.43	0.48
2	20	0.52	0.57
3	40	0.61	0.66
4	60	0.69	0.72
5	80	0.74	0.77
6	100	0.80	0.83
7	120	0.86	0.89
8	140	0.90	0.92
9	150	0.93	0.95
10	160	0.97	0.98

FIGURE 4

Secure Transmission node analysis



V. CONCLUSION

An authentication scheme is proposed to secure multihop network programming with multiple one-way hash chains. Instead of the expensive asymmetric cryptographic primitives used in much prior work, the scheme employs only symmetric +cryptographic primitives, in a circular geographic node deployment model. Possible attacks an adversary could mount on the scheme is discussed and provided simple and effective counter measures against them. Finally, it provides a comprehensive performance evaluation of the scheme in terms of end-to-end latency and power consumption, which is said to be believed, is the first power consumption evaluation of a security scheme for network programming protocols. The system eliminates the difficulties in the existing system. It is developed in a user-friendly manner. The system is very fast and any transaction can be viewed or retaken at any level. Error messages are given at each level of input of individual stages. This software is very particular in securing the multi-hop network programming.

VI. REFERENCES

[1] Arasu, et al. "STREAM: the stanford stream data manager (demonstration description)." In ACM

SIGMOD international conference on Management of data, pp. 665-665, ACM, 2003.
[2] H-S. Lim, Y-S. Moon and E. Bertino, "Provenance-based trustworthiness assessment in sensor networks." In SeventhInternational Workshop on Data Management for Sensor Networks, pp. 2-7. ACM, 2010.
[3] S. Sultana, G. Ghinita, E. Bertino and M. Shehab, "A lightweight secure provenance scheme for wireless sensor networks." In 18th International Conference on Parallel and Distributed Systems (ICPADS), pp. 101-108, 2012.
[4] R. A. Shaikh, S. Lee, M. AU Khan and Y. J. Song, "LSec: lightweight security protocol for distributed wireless sensor network." In IFIP International Conference on Personal Wireless Communications, pp. 367-377. Springer BerlinHeidelberg, 2006.
[5] G. Selimis et al., "A lightweight security scheme for wirelessbody area networks: design, energy evaluation and proposed microprocessor design." Journal of medical systems, vol. 35, no. 5, pp. 1289-1298, 2011.
[6] G. Selimis, at al. "Evaluation of 90 nm 6 T-SRAM as Physical Unclonable Function for Secure Key Generation in Wireless Sensor Nodes", in IEEE ISCAS Brazil, pp. 567-570, 2011.
[7] M. Roesch,"Snort: Lightweight Intrusion Detection for Networks." LISA, vol. 99, no. 1, pp. 229-238. 1999.
[8] N. Tsikoudis, A. Papadogiannakis and E. P. Markatos, "LEoNIDS: a Low-latency and Energy-efficient Network-level Intrusion Detection System." IEEE Transactions on Emerging Topics in Computing, vol. 4, no. 1, pp.142-155, 2016.
[9] W. Lee and S. J. Stolfo, "Data Mining Approaches for IntrusionDetection." In Usenix security. 1998.
[10] Y. Xie, D. Feng, Z. Tan and J. Zhou, "Unifying intrusion detection and forensic analysis via provenance awareness." Future Generation Computer Systems, vol. 61, pp.26- 36, 2016.

First S. Savitha is an Assistant Professor in the Department of Computer Science and Engineering, K.S.R. College of Engineering (Autonomous), India. She received her Master of Engineering degree in Computer Science and Engineering in 2012 from Anna University, Chennai, India. She is a Research scholar in Anna University, Chennai. She has published more than 15 papers in referred journals and conference proceedings. Her research interest includes Data Mining, Big Data, Cloud computing, Databases and Artificial Intelligence. She is a professional member of ISTE.

Second K. *Logeswaran* is an Assistant Professor in the Department of Information Technology, Kongu Engineering College (Autonomous), India. He received his Master of Engineering degree in Computer Science and Engineering in 2011 from Anna University, Chennai, India. He has published more than 15 papers in referred journals and conference proceedings. His research interest includes Wireless Sensor Networks, Big Data, Network Security and Cloud computing. He is a professional member of CSI.

Third N. Abinaya is a final year student in the Department of Computer Science and Engineering, K.S.R. College of Engineering (Autonomous), India. Currently she is doing her final year project in security issues in cloud.

Fourth R. Dhinakaran is a final year student in the Department of Computer Science and Engineering, K.S.R. College of Engineering (Autonomous), India. Currently he is doing his final year project in security issues in cloud.

Fifth S. Mahalakshmi is a final year student in the Department of Computer Science and Engineering, K.S.R. College of Engineering (Autonomous), India. Currently she is doing her final year project in security issues in cloud. She has also presented papers on security issues in cloud in various events.

Sixth B. Dharmesh is a final year student in the Department of Computer Science and Engineering, K.S.R. College of Engineering (Autonomous), India. Currently he is doing his final year project in security issues in cloud.