International Journal of Science and Research (IJSR)
ISSN: 2319-7064
ResearchGate Impact Factor (2018): 0.28 | SJIF (2018): 7.426

Distributed Data Provenance (DDP) to Secure the Sensor Data Stream

Mohammad Amanul Islam

Xidian University, Xi'an, 710071, China

Abstract: ***Introducing provenance has already been established as a very substantial approach of securing the sensor data that records the history (e.g., creation, ownership, significance) and operations performed on the data or its travel path. The data provenance is also crucial for ensuring data accountability and privacy as preserving the quality of data stored in numerous systems. Hence, keeping the provenance information requires addressing several challenges including storage overhead, high throughput, and secure transmission, etc. In this paper, we propose architecture of distributed data provenance that avoids the problem of data degradation because of adding a large size of data provenance. Moreover, it enables the transparency of data accountability, helps to enhance the privacy and availability of data provenance. In this scheme, the provenance signifies the creation history of data in the hash form of data, and it has been distributed as chained information with the data into the given data payload of the data packet. Experiments demonstrate the efficiency, scalability on the provenance encoding capacity of the proposed scheme, and the reliability of this scheme has been illustrated with low transmission overhead for the payload storage application of a packet in the wireless sensor environment.***

Keywords: bits, data, provenance, payload, wireless sensor network.

1. Introduction

The significance of data provenance for streaming data is very crucial nowadays in order to confirm high trustworthy data [15]. Data trustworthiness can be evaluated by the summarized history of the ownership, and the actions performed on the data. Thus, the assurance of data trustworthiness and prioritizing the management of secure provenance are now becoming an active area of research. Provenance for affirming the trustworthiness of sensor data [10], location data [3], and multi-hop network [22] have already been studied recently and have marked the key contribution for adding the provenance with streaming data. The importance of provenance for streaming data is also highlighted in the Research and Development Challenges for National Cyber Security report [23], which recommends research initiatives for efficient and secure implementation of provenance for real-time systems. Simmhan et al. [17] concluded that depending on the particular application either provenance is propagated along with data, or in a separate channel within a fixed size of its data type.

In past research, homomorphic encryption techniques [24], [11] have used as an application of cryptography [26] to en-route data verification that allows direct aggregation of encrypted data. Another scheme by R. Hasan et al. [6] proposed on both encryption and incremental chained signature to protect sensory data. Though the use of encryption, digital signature or MAC provides the data security, it consumes significant bandwidth and impacts efficiency and scalability. Moreover, the secret message that hides as provenance in using such mechanisms might be common along with other parameters tagged with many packets in a data stream.

Regarding data provenance, past research mainly focused on modeling an efficient channel (e.g., inter-packet delay [18], data field [19], chaining group of data [7], [20], etc.) to hide information, insert data provenance as inline metadata [9]; analyze capacity of data provenance on the channel with or without tempering the data; assure a secure and efficient transmission of data provenance [18], very few approaches have reported on the provenance capacity. In particular, acquaint with provenance to secure the wireless sensor data imposes a set of challenges:

- Provenance must not increase storage cost, should be efficiently managed so as the assigned transport channel is utilized well;
- The security (confidentiality, authenticity, integrity) for both data and provenance or any secret message should be ensure;
- Adding provenance to data must adapt fast processing and low computational cost, and network parameters (e.g., delay, packet receiving rate) must not introduce significant overhead on processing.

In this paper, to target the aforementioned challenges, we design a distributed data provenance encoding framework to secure data stream, or DDP for short.The proposed framework introduces multiple bit-shift windows, i.e., unique segments of the total length of a data field, to query the minimum required bit-width for each sensed value of a particular data set. Thus, a variable length between the data has been observed in a particular data set. In this methodology, the minimum required bits refer to the bits that belong to the sensor value out of the predefined size of its data type. Assume the binary of a sensor data $X = 29$ in 8 bits space is 00011101. Our scheme accounts only 11101 as the minimum required bits in binary. Thus, the accumulated number of minimum required bits for X is 5 bits out of 8 bits.

Thereafter, the proposed method employs a bit shift operation on each data with its corresponding bit-shift value that has been initiated based on the actual-bit width of data following several bit-shift windows. Thus, a certain length of vacated space is found on the observed data at the end of the shifting process. The proposed scheme parts the provenance in several lengths of chunks and utilizes the vacated space to encode the data provenance with the transmitted sensor data. A data source follows the above norms to encode the variable length of provenance chunk along with a variable length of data.

However, the proposed scheme also confines the shifting process so as the shifted bits do not exceed the limit that set over the length of each segmented data filed of a given data payload. Thus, the original data does not lose any bits while it is being shifted so as the employed shifted code of data as a provenance carrier suffers no data degradation. Furthermore, the security scheme is able to detect error on access violation to any data field caused by attacks. In summary, the contributions of this paper include:

- Introducing a distributed provenance encoding approach with the variable length of the data stream;
- Ranging the encoding capacity without adding additional storage overhead;
- Designing security properties with composing the data arrival time, data specific key and another protocol parameter e.g., bit-shift value, leading to the security of data and provenance;
- Designing an efficient provenance recovery procedure with an optimum complexity;
- Evaluating the performance of network parameters using the synthetic data that reflects the bit-shift characteristics and demonstrating the effectiveness and efficiency of the proposed network.

The rest of the paper is organized as follows: Sec. II focuses on the related works. Sec. III introduces the system model and preliminaries. Sec. IV explains the proposed framework, different stages of provenance encoding, retrieval process. Sec. V evaluates the performance and analyzes the security and complexity of our framework. Finally, Sec. VI and VII present the discussion and conclusions part of the paper respectively.

2. Related works

In terms of wireless sensor network security, encoding data provenance to ensure the data trustworthiness have already been explored as a well-known technology. Additionally, previous research also resolved the difficulties of adding a large quantity of data provenance in a self-describing binary format. Insert provenance into different channels has also been investigated extensively. Therefore, it is not an overhead to encode the provenance in today's world of increasingly dynamic sensor environments.

A fragile encoding algorithm has proposed by Guo et al. [5], to verify the integrity of streaming data. This scheme divides the data into groups; secret messages are chained across the group and encode the secret data into the LSB of data. Data deletion can be correctly detected due to the chaining characteristic.

Another fragile encoding method with a dynamic group size of data and chaining approach has proposed by Wang et al. [20]. In this scheme, the data first converted in character, and then discover the blank character as encoding scope. Afterward the data is divided into groups dynamically. Finally, generate the hash of each group data as their own fragile secret data. Though, the above schemes intelligent to resolve the data forgery and the integrity problems but it affects the data accuracy and encoding capacity.

In [19] Xingming Sun *et al.*, we found some work relevant to our scheme, but the characteristics of data and data fields haven't examined extensively in this scheme.In this work, fixed redundant bits are initially figured out among the data fields to insert the secret message. This scheme was proposed to verify the data integrity based on the accuracy of received secret data from the data field. This method is convenient in terms of the computational cost of its security scheme. It doesn't adopt any additional secret keys other than group XOR of hashed data, where each data is hashed with its arrival time. Much proficient research work regarding active timing based data hiding [7], [21] have already been studied to protect the network flow. Another significant protocol, inter-packet delay (IPD) based provenance encoding proposed by Sultana et al., [18]. In this scheme, each provenance bit inserted into consecutive delay between the packets in a particular flow. Thus, assuming a particular traffic flow with 250 packets, a data source can encode, $250 - 1 = 249$ bits as data provenance, as the first packet doesn't meet any delay. This scheme has also a limitation on encoding a large size of provenance by a single node. The difference between the active timing and IPD based methods is, active timing scheme may encode a single secret message over the IPDs in a particular flow, whereas IPD method allows multiple nodes to encode provenance over the same set of IPD.

Our proposed scheme significantly differs from the above approaches in several aspects: (i) it computes the shifted code of data by exploring the length of a data, data field, the bit-shift value according to bit-shift windowing, (ii) a data provenance break down into several chunks, and each chunk encoded over the shifted code of the sensor data. An idea of imposing additional information as an encrypted signature during the data acquisition was proposed by J. Feng *et al.* [4]. In such a case, secret data is selected to identify tradeoffs between the accuracy and the strength of proof of the authorship. On the contrary, in our scheme, SN authorizes each pre-shared data specific keys with the bit-shift value and last modification period correspond to each data. Furthermore, the modification timestamp and the bit-shift value play a key role in the key synchronization process on querying bit-shift value to confirm a successful recovery of data and data provenance.

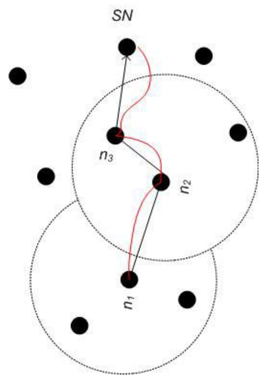

Figure 1: A GPS enabled sensor network for the data traffic

3. System Model and Preliminaries

A. Network Model

The proposed framework considers a typical deployment of wireless sensor networks that is enabled with position based routing protocol [14], [8] and comprised of numerous sensor nodes, and routing algorithm finds the paths from a traffic source to a traffic destination through a series of intermediate forwarding nodes. Each node is stationary after deployment, and able to determine its own geographical position information using a Global Position System (GPS) device.

Routing paths may change due to node failure, resource optimization, position changes, etc. Fig. (1) shows the topology scenario of a GPS enabled sensor network [8] and a route for the data traffics. The data originated from the node $n_i$towards the sink node (SN). To forward the data packets, the routing protocol exploits the communication between geographic position and connectivity in a wireless network by using the position of nodes.

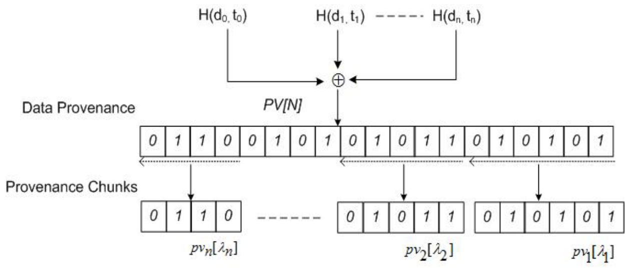

Figure 2: Provenance Structure.

The modeled network consists of an SN to receive the data and associated with outside network infrastructures such as the Internet. The SN broadcasts security information to the sender nodes using a secure mechanism (e.g., μTESLA [16]) to confirm the incoming packets are not forged by any adversary, and the SN itself is not compromised. Thus, a reliable, and in order message delivery is assured. In the context of proposed framework, each sender node shares a set of security components including secret keys, bit-shift values, and original provenance, with the SN. These security components are functional to a particular data that stored into the equal number of data fields of a data packet.

B. Data Model

The proposed method utilizes a particular set of data fields dF of a data package $pk_i \in PK$to transport a particular data set DStowards the SN. Here, $df_i \in dF$ and PK are the data field and a set of data packages respectively. To be specific each data packet contains reading values read by sensors, and the sensor values are stored into the data fields of a packet during transmission. A data package pk_i can be expressed as $pk_i = pk_i(df_1||df_2|| \ldots ||df_n)$, where each data field df_i has L bits storage space, similar the size of the data type of sensor value.

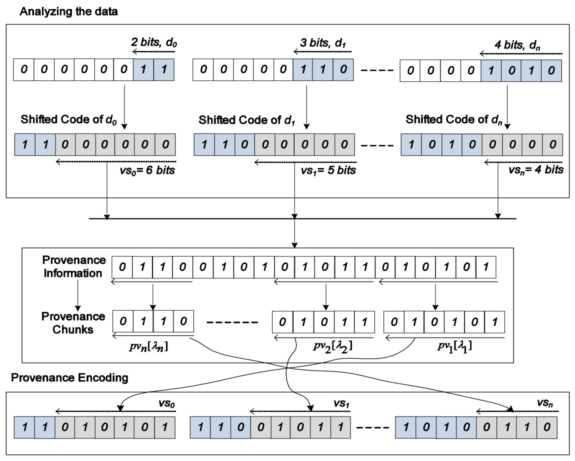

Figure 3: Provenance Encoding Structure.

Definition 1: A data packet that originated by a sensor node in the WSN contains n number of data fields, $dF = \{df_1, df_2, \ldots, df_n\}$ to store an equal number of data $D = \{d_1, d_2, \ldots, d_n\}$ temporarily as a data package according to the pre-decided order. Each d_i indicates one sensor sample in decimal format and its actual bit-width is compared against the total length of df_i.

C. Data Provenance

The idea of provenance exhibits the information that describes a source node or the nodes (i.e., access points, router) and operation on data are being transmitted [10]. However, the provenance can be characterized in many ways based on the particular application domain [13].

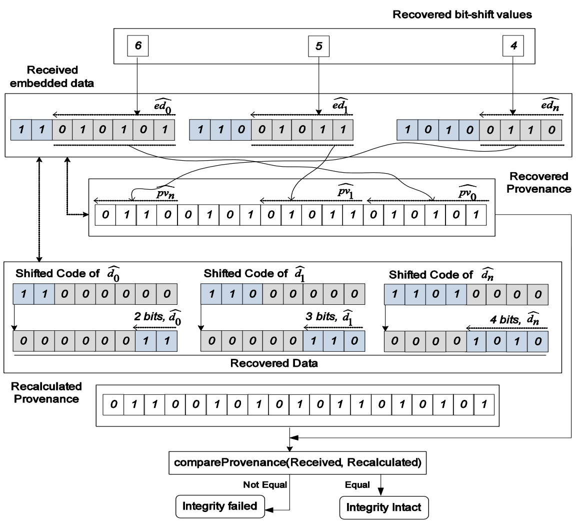

Figure 4: Provenance Decoding Structure.

In our scheme, data provenance appeared as hashed information which is composed of each hashed form of each sensor value and its certain number of modification periods if they occur for a particular data set. Thus the data provenance ensures the creation history of sensor data. The structure of data provenance charted in this scheme have shown in the Fig. (2). The generated data provenance initially parted into multiple groups (in bits) as a provenance chunk, then each chunk is selected for encoding with the formatted (shifted code) sensor values according to the functions of the DDP. However, the size of each provenance chunk depends on the length of vacated space (freed bits) remained after the shifting process with a selected bit-shift value. The encoding process performed within the data fields of a data packet Provenance to encode may formally be defined as:

Definition 2: A data provenance $PV = \{pv_1||pv_2|| \dots ||pv_n\}$ is selected to be encoded with each sensory data that obtain following properties: (i) each pv_i is a variable length of provenance chunk (a group of bits) of a hashed provenance message. (ii) For any particular stream, $\forall\, d_i \in D$ must be checked for encoding a pv_i and storing the encoded data into the df_i of a data packet; (iii) SN is a designated recipient of the data provenance;

The proposed scheme introduces a framework to query the length of each input data stream using several bit-shift windows, which are mapped to a distinct bit-shift value. Thus, a set of variable length of binary sequences as a set of provenance chunks is encoded with a variable length of numerous sensor data.

D. Provenance Encoding

Encoding data provenance technique, provides a hiding capability of identification code (e.g., visible or invisible secret message) that is encoded with the data, and remains exist within the data after any decryption process. Two key components of encoding secret message are encoder, and decoder. The encoder that involves in encoding a secret message (e.g., provenance) within the digital content. The decoder extracts the data provenance from the received digital content. During this process, an encoder may use a single or multiple secret keys to protect the encoded data, which may share with a decoder in a secure channel to decode received information. Fig. (3) and Fig. (4) illustrates the provenance encoding and decoding structure respectively that has followed by the proposed DDP framework. From the perspective of our proposed framework, two complementary process of data provenance encoding are as follows:

(1) Data encoding function FE that works to encode a provenance $pv_i[j] \in PV$ utilizes the d_i' as a secret data carrier, and possibly, a bit-shift value $\lambda_i \in \lambda S$ to generate the encoded data $ed_i = FE(d_i', \lambda_i, pv_i[j])$, where d_i' is a shifted form of d_i. A data receiver SN later examines a set of received information $\widehat{ED} = \{\widehat{ed}_1, \widehat{ed}_2, \dots, \widehat{ed}_n\}$ per pk_i to determine the encoded PV.

(2) Data decoding function FD that takes a received data $\widehat{ed_\iota} \in \widehat{ED}$ as an input to determine the existence of $pv_i[j]$ using an appropriate bit-shift value, $a\lambda_i$. The function returns the $pv_i[j] = FD(\widehat{ed_\iota}, a\lambda_i)$.

In this framework, each bit-shift value is recovered from each $\widehat{ed_\iota}$ specific $k_i \in KS$ before the decoding process and verify its correctness. Here, each k_i is used for three purpose. First, to encrypt a particular $\widehat{ed_\iota}$ by a source node, and second, to decrypt each received $\mathcal{E}(\widehat{ed_\iota})$ that extracted from the data fields of a data packet, and finally, to recover λ_i as a $a\lambda_i$ that applied on $\widehat{ed_\iota}$. Thus, the computational cost of this scheme is little high.

4. The Proposed Framework

a) Overview

In this work, the proposed scheme shows a distributed approach of encoding data provenance, where a single provenance is distributed in the distinct length of several chunks and encoded with a variable length of sensor data in a particular data stream. Encoded information is carried inside the data fields of a data packet. The proposed DDP framework has the following functional properties: manipulate the sensor data stream, selection of bit-shift value, encode the provenance and ensure security for data and provenance. Hence, the DDP scheme verifies the minimum required bits b_a (the necessary size in bits) of each d_i using several bit-shift windows (i.e., $b_l \le b_a \le b_u$), and finds which bit-shift window it fits to.

Bit-shift windows are ranged within the total bit length (i.e. L bits) of a data field, where each bit-shift window is specified with a lower limit and an upper limit as b_l and b_u respectively. Before, encoding a provenance, each data must shift its bits with a selected λ_i from a set of bit-shift values λS, and find the equal length of freed bits γ_i. After the shifting process, the new shifted code d_i' is used to encode the data provenance length up γ_i bits.

However, the DDP uses a data specific k_i to create an encrypted $\mathcal{E}(ed_i)$ before transmission. Though the SN can decrypt the received $\mathcal{E}(ED)$ using a set of pre-shared K, the SN also uses the k_i to determine the source applied λ_i as $a\lambda_i$ based on a key synchronization procedure at the end of the decryption process. Thereafter, the recovered $a\lambda_i$ is used to decode the provenance from the decrypted received data.

However, the tailed procedure with the DDP framework to encode and decode the provenance can be described in the following steps: (i) initialize the framework with multiple distinct bit-shift windows, (ii) execute the shifting process according to bit-shift windowing, (iii) generate the data provenance and its encoding process, (iv) prepared security initiatives before transmission. (v) decrypt and decode the data provenance.

b) Initialize the framework with multiple distinct bit-shift windows

Initially, a node n_i constructs the payload structure of its packet with a certain number of data fields of a primitive data type. Due to common data type, each df_i has the same bit width (i.e. L bits) as storage space for the data. The sum of storage size of all the data fields is the given payload of a data packet as, $payload(pk_i) = \sum_{i=1}^{n} df_i(L)$. The DDP uses each df_i to transport the sensor data d_i. Primarily, to construct a ed_i, Each d_i is queried through a set of bit-shift windows ΔS to find a bit-shift value to construct a d_i', where each $\Delta_i \in \Delta S$ is a particular partition of the total length ($L\ bits$) of a df_i.

$$\Delta S = \{\Delta_1[l_i, u_i], \Delta_2[l_{i+1}, u_{i+1}], \dots \Delta_{|\Delta S|}[l_{|\Delta S|}, u_{|\Delta S|}]\}$$

where, the lower limit l for any Δ_{i+1} is, $l_{i+x} = u_{i+x} + 1$.

Remark: Each bit-shift window Δ_i is a particular range of typical bit width (e.g., 0 to 7 bits, 8 to 15 bits) within the entire bit width (i.e., 32 bits) of a data field. Thus, several distinct unsigned ranges (e.g., 0 to 255, 256 to 65535) are formed according to the typical bit width.

While a d_i find its appropriate Δ_j, it can discover an appropriate λ_i, which is utilized to shift its bits and open up γ_i length of freed bits in a vacated space vs_i, where aprovenance chunk is encoded. However, a node n_i may generate bit-shift windows offline with the knowledge of the length of each data field of a data packet.

c) Execute the shifting process according to bit-shift windowing

At this stage, each d_i specific selected $\lambda_i \in \lambda S$ is applied to the data with a left shift operation in this scheme. Note that, the partitioned bit-shift windows per df_i arethe same for any d_i, thus the selected λ_i for any d_{i+1} might be the same or distinct depending on its actual bit width. Here, $\lambda_i = 0$ if $d_i \notin \Delta_i$ that indicates a non-encoded data, otherwise $0 < \lambda_i \le pv_i^{MAX}$ if $d_i \in \Delta_i$, where $\lambda_i = L - b_a = \gamma_i = pv_i^{MAX}$ is a maximum limit (in bits) of a provenance chunk to encode with a formatted d_i. Shifted bits are restricted to be set within the length of the data field. The allocated λ_i in each Δ_j constructed by subtracting the size of bit-shift window Δ_j^{ws} (bit width of a range) from the total length of df_i. Note that, limiting the shifted binary digits within the length of a data field restricts the bits to get shifted off the end while a selected λ_i is being applied. Thus, the maximum and minimum bit width of a λ_i in a particular Δ_j are determined by solving the equation number (1) and (2), respectively for a d_i as follows:

$$\lambda_i^{MAX} = \left(L - \Delta_j^{ws}\right) = L, if \;\; 2^0 \le d_i < 2^1 \qquad (1)$$

$$\lambda_i^{MIN} = \left(L - \sum_{j=0}^{M} \Delta_j^{ws}\right)$$

$$= L - (L-1), if \;\; 2^{L-2} \le d_i < 2^{L-1} \qquad (2)$$

where, $\Delta_j^{ws} = f(\Delta_j) = (u - l) + 1$, the length of L is ranged in between $[0, (L-1)]$ and M denotes the number of bit-shift window.Each d_i performs a sequential query through several bit-shift windows to select a λ_i according to its actual bit width.Hence, the simplified formation of d_i using the elected λ_i can be defined as, $d_i' = d_i \ll \lambda_i$ or $d_i' = d_i^{\lambda} \parallel \gamma_i$, where d_i^{λ} indicates the status as the bit-shift value is applied on respective d_i. However, the data source originates a set of formatted shifted code $D' = \{d_1', d_2', \dots, d_n'\}$ for a particular data set following the above procedure. Thus, a set of vacated space for n data items is denoted as $\{vs_1, vs_2, \dots, vs_n\}, (0 \le i \le n)$, where n implied the number of data fieldsand $vs_i = \gamma_i$ indicates the length of freed bits in the vacated space of i^{th} data field, and the total size of vacated space $N = \sum_{i=1}^{n} \gamma_i$ that uses for encoding provenance in distributed approach.

d) Provenance generation and its encoding process

In this step, the DDP scheme follows the below formal steps to construct the data provenance and continue its encoding process:

Algorithm 1. Provenance Encoding

Input: n number of data according to data fields quantity, data fields length L bits.
Output: the encrypted encoded data set $\mathcal{E}(ED)$.
Start

```
1.  for $i$ = 1 to |$n$| do
2.    for $j$ = 1 to |$T$| do
3.      $hd_{ij} \leftarrow Hash(d_i, t_{ij})$
4.    end for
5.  end for
6.  $PV \leftarrow groupXOR(hd_{11} \oplus hd_2 \oplus \dots \oplus hd_{|n|\times|T|})$
7.  for $i$ = 0 to $(L - \lambda[k]) - 1$ do
8.      $tmp1 \leftarrow d_1[i]$
9.      $d_1[i + \lambda[k]] \leftarrow tmp1$
10.     $d_1[i] \leftarrow 0$
11. end for
12. for $i$ = 0 to $\lambda[k] - 1$ do
13.     $d_1[i] \leftarrow PV[i]$
14.     $p \leftarrow p + 1$
15. end for
16. for $i$ = 0 to $(L - \lambda[k+1]) - 1$ do
17.     $tmp1 \leftarrow d_2[i]$
18.     $d_2[i + \lambda[k+1]] \leftarrow tmp1$
19.     $d_2[i] \leftarrow 0$
20. end for
21. for $i$ = 0 to $\lambda[k+1] - 1$ do
22.     $d_2[i] \leftarrow PV[p]$
23.     $p \leftarrow p + 1$
24. end for
25. …
26. for $i$ = 0 to $(L - \lambda[k + (n-1)]) - 1$ do
27.     $tmp1 \leftarrow d_n[i]$
28.     $d_n[i + \lambda[k + (n-1)]] \leftarrow tmp1$
29.     $d_n[i] \leftarrow 0$
30. end for
31. for $i$ = 0 to $\lambda[k + (n-1)] - 1$ do
32.     $d_n[i] \leftarrow PV[p]$
33.     $p \leftarrow p + 1$
34. end for
35. Encrypt $(ED)$
36. Send $(\mathcal{E}(ED))$
```

Stop

Rule1: A data source n_i initiates n number of sensor data $\{d_1, d_2, \dots, d_n\}$ for transmission and a certain number of data modification periods $T = \{t_1, t_2, \dots, t_{|T|}\}$, where the last modification time $l_{t_i} = t_{|T|}$ for each $d_i \in D$.Hence, each vector value hd_{ij} of an $n \times |T|$ matrixrepresents a hash message based on a one-ways hash function [1], which is composed of each data and the corresponding timestamp of each modification.

$$hd_{ij} = H(d_i, t_j)$$

where t_j is the modification period of i^{th} data d_i. Then, a provenance data PV is calculated as

$$PV = groupXOR(hd_{11} \oplus hd_{12} \oplus \dots \oplus hd_{|n|\times|T|})$$

where, the creation history for i^{th} data d_i can be denoted as,

$$PV_{d_i} = groupXOR(hd_{i1} \oplus hd_{i2} \oplus \dots \oplus hd_{i|T|})$$

Here, $\oplus$ denotes the XOR operation [25].

Rule 2: A provenance message PV should parted in several chunks as $\{pv_1[\lambda_1], pv_2[\lambda_2], \dots, pv_n[\lambda_n]\}$, where the size of each pv_i is determined by the selected λ_i. Thus, the choice of bits for pv_i signifies the selection of j of $PV[j]$, where the j

Algorithm 2. Provenance Decoding

Input: A data set ED with n encoded data items according to n number of data fields, and the size of each data field L bits.
Output: the result of data integrity based on the provenance comparison.

```
Start
  1.  $p \leftarrow a\lambda[m] - 1$
  2.  for $i = a\lambda[m] - 1$ to 0 do
  3.      $\widehat{PV}[p] \leftarrow \widehat{ed_1}[i]$
  4.      $\widehat{ed_1}[i] \leftarrow 0$
  5.      $p \leftarrow p - 1$
  6.  for $i = a\lambda[m]$ to $L$ do
  7.      $tmp0 \leftarrow \widehat{ed_1}[i]$
  8.      $\widehat{d_1}[i - a\lambda[m]] \leftarrow tmp0$
  9.      $\widehat{d_1}[i] \leftarrow 0$
  10. end for
  11. $p \leftarrow (a\lambda[m] + a\lambda[m+1]) - 1$
  12. for $i = a\lambda[m+1] - 1$ to 0 do
  13.     $\widehat{PV}[p] \leftarrow \widehat{ed_2}[i]$
  14.     $\widehat{ed_2}[i] \leftarrow 0$
  15.     $p \leftarrow p - 1$
  16. end for
  17. for $i = a\lambda[m+1]$ to $L$ do
  18.     $tmp0 \leftarrow \widehat{ed_2}[i]$
  19.     $\widehat{d_2}[i - a\lambda[m+1]] \leftarrow tmp0$
  20.     $\widehat{d_2}[i] \leftarrow 0$
  21. end for
  22.   ...
  23. $p \leftarrow (a\lambda[m] + a\lambda[m+1] + \cdots + a\lambda[m+(n-1)]) - 1$
  24. for $i = a\lambda[m+(n-1)] - 1$ to 0 do
  25.     $\widehat{PV}[p] \leftarrow \widehat{ed_n}[i]$
  26.     $\widehat{ed_n}[i] \leftarrow 0$
  27.     $p \leftarrow p - 1$
  28. end for
  29. for $i = a\lambda[m+(n-1)]$ to $L$ do
  30.     $tmp0 \leftarrow \widehat{ed_n}[i]$
  31.     $\widehat{d_n}[i - a\lambda[m+(n-1)]] \leftarrow tmp0$
  32.     $\widehat{d_n}[i] \leftarrow 0$
  33. end for
  34. for $i$ = 1 to $|n|$ do
  35.   for $j$ = 1 to $|T|$ do
  36.     $hd_{ij} \leftarrow Hash(\widehat{d_\iota}, \widehat{t_{\iota J}})$
  37.   end for
  38. end for
  39. $\widehat{\widehat{PV}} \leftarrow groupXOR(hd_{11} \oplus hd_2 \oplus \ldots \oplus hd_{|n|\times|T|})$
  40. if $\widehat{PV}$ is not equal to $\widehat{PV}$ then
  41.     Return (Integrity Failed)
  42. Else
  43.     Return (Integrity Intact)
Stop
```

value for $pv_{i+1}[j]$ will start after the j value used by pv_i last. Thus, each pv_i is added to γ_i space of shifted code of data d_i'.

Rule3: Encoding pv_i to the data set includes the following steps:

(1) Generates the provenance information and its distribution as a provenance chunk according to rule 1 and rule 2;

(2) Encode a pv_i into the γ_i freed bits of each shifted code data, and denotes a simplified encoded data as $ed_i = d_i'|pv_i$.

Though the size of pv_i has not been pre-determined in provenance encoding algorithm, a data source can analyze the sensed data to encode a certain length of pv_i with it in offline based on the predetermined size of a data field. **Algorithm 1** describes a simplified provenance encoding approach. In this method, provenance information encoded in a binary form. In order to simplify the description of the algorithm, p uses as a position counter; end of its designated loop it signifies the position index of bits of a provenance chunk pv_{i+1} to insert. However, at the end of the encoding process an aggregate pool of encoded data can be stated as $ED = \{ed_1, ed_2, \ldots, ed_n\}$.

e) Prepared security initiatives before transmission

In this section, the security approach that involved in our proposed methodology has described explicitly. Before initiating the transmission, each encoded data ed_i is encrypted based on the XOR encryption method [25].The DDP framework uses a message transformation function enc to encrypt each ed_i with each formatted d_i specific k_i, and thus an encrypted encoded message $\mathcal{E}(ed_i)$ is computed as,

$$\mathcal{E}(ed_i) = enc(ed_i; k_i)$$

where k_i is a secret key for each ed_i. The DDP framework also follows the one-way hash function [1] to generate each data specific k_i as,

$$k_i = H(i, l_{t_i}, \lambda_i)$$

where, l_{t_i}, λ_i, i are the last modification timestamp, selected bit-shift value and the position index of i^{th} formatted d_i respectively, and thus a set of secret keys $\{k_1, k_2, \ldots, k_n\}$ is created for n data items. Encryption operation performed in a manner, where the index of k_i and ed_j should be the same as, $i == j$.

Furthermore, the SN will regenerate k_i as k_i^{new} with a reciprocal relation between the pre-shared $\widehat{k_\iota}$ and the knowledge of $\widehat{\lambda_\iota}$, which is a key synchronization policy to determine whether it does match or not. If it does match, the involved λ_i will be regarded as the exact applied bit-shift value that uses for further recovery of encoded provenance chunk and original form of data .

Table 1: Relationship between bit-shift values and bit-shift windows.

bit-sift windows	range of values	coverage bit-width	allocated bit-shift values
Δ_1	2^0 to 2^7	8	24
Δ_2	2^7 +1 to 2^{15}	16	16
Δ_3	2^{15} +1 to 2^{23}	24	8
Δ_4	2^{23} +1 to 2^{24}	25	7
Δ_5	2^{24} +1 to 2^{25}	26	6
Δ_6	2^{25} +1 to 2^{26}	27	5
Δ_7	2^{26} +1 to 2^{27}	28	4

f) Decrypt and decode the encoded provenance

In this section, the SN extracts the received ciphered data set $\mathcal{E}(\widehat{ED}) = \{\mathcal{E}(\widehat{ed_1}), \mathcal{E}(\widehat{ed_2}), \ldots, \mathcal{E}(\widehat{ed_n})\}$ from all the data fields of a data packet. Then, performs a decryption procedure with a set of pre-shared data specific keys $\{\widehat{k_1}, \widehat{k_2}, \ldots, \widehat{k_n}\}$. Here, the decrypt function dec uses each $\widehat{k_\iota}$ as an input for each $\mathcal{E}(\widehat{ed_\iota})$ to recover a decrypted form of $\mathcal{D}(\widehat{ed_\iota})$ as,

$$\mathcal{D}(\widehat{ed_\iota}) = dec(\mathcal{E}(\widehat{ed_\iota}); \widehat{k_\iota})$$

Thus, the SN originates a set of encoded messages $\widehat{ED} = \{\widehat{ed}_1, \widehat{ed}_2, \dots, \widehat{ed}_n\}$ at the end of the extraction and decryption process for decoding the data provenance.

In our proposed methodology, SN uses the applied bit-shift values as a recovery mechanism that corresponds to each encoded data. Finally, the SN verifies the integrity of data and provenance by comparing the retrieved provenance and pre-shared original provenance. Thus, the DDP formalizes the decoding process in two separate procedures: bit-shift value recovery and provenance recovery.

Bit-shift value recovery: Since the sender n_i doesn't share the bit-shift value λ_i that used to encodepv_i, the SN performs recovery of applied bit-shift value $a\lambda_i$ from each well-ordered pre-shared $\widehat{k_\iota}$ that has the information about $a\lambda_i == \lambda_i$. Such a recovery process is performed based on a key synchronization procedure. In this procedure, $\widehat{k_\iota}$ is regenerated as k_i^{new} that involves the position index value of all the ordered $\{\widehat{ed}_1, \widehat{ed}_2, \dots, \widehat{ed}_n\}$ and a set of pre-shared knowledge about the bit-shift values $\{\widehat{\lambda}_1, \widehat{\lambda}_2, \dots, \widehat{\lambda}_{|L|}\}$. Here, the procedure of generating k_i^{new} is the same as it was followed by the n_i, where, k_i^{new} is composed of $\widehat{l_{t_\iota}}$, i, $\widehat{\lambda_\iota}$.

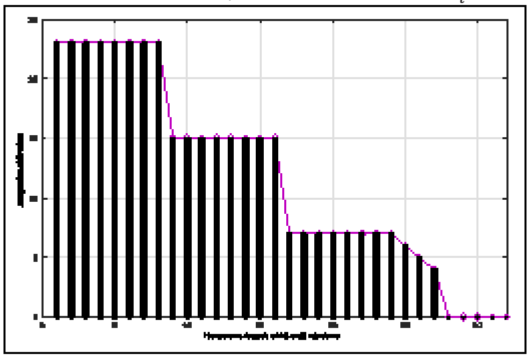

Figure 5: Allocated bit-shift against the bit-width of numerous bit-shift windows.

For any particular $\widehat{ed_\iota}$, if its index value i and the pre-shared arrival time $\widehat{l_{t_\iota}}$ match to an appropriate $\widehat{\lambda_\iota}$ then a correct k_i^{new}will be synchronized to $\widehat{k_\iota}$as, $k_i^{new} == \widehat{k_\iota}$, and return the matched λ_i as $a\lambda_i$ .Note that, the vacated bits γ_i were considered for encoding apv_iafter applying aλ_i. Here, the value of $a\lambda_i$could be 0that means none of the bits related to any pv_i areinserted with that particular sensed data. Such a condition might exist if$d_i \notin \Delta_j$occurred during encoding.

Provenance Recovery: At the end of the bit-shift value recovery process, the SN originates a set of applied bit-shift values $\{a\lambda_1, a\lambda_2, \dots, a\lambda_n\}$ that corresponds to $\{\widehat{ed}_1, \widehat{ed}_2, \dots, \widehat{ed}_n\}$. Thereafter, each $a\lambda_i$ is applied to each corresponding element of $\widehat{ED}$ according to **Algorithm 2**. The program calculates the received data provenance$\widehat{PV}$, where each binary digits belongs to$\widehat{pv_\iota}$is extracted from$\widehat{ed_\iota}[L]$. Here, the binary digits of$\widehat{pv_\iota}$accrued from $\widehat{ed_\iota}[a\lambda_i - 1]$ down to$\widehat{ed_\iota}[0]$, where the position of each bit into $\widehat{PV}[N]$have been tracked by a position counter p. However, all the bits between $[\widehat{ed_\iota}[a\lambda_i], \widehat{ed_\iota}[L-1]]$ take $a\lambda_i$ bits right shift to repossess the received data as$\widehat{d_\iota}$. SN recalculates the data provenance as$\widehat{\widehat{PV}}$according to same rule followed by the data source and compare $\widehat{\widehat{PV}}$ and $\widehat{PV}$ to determine whether the integrity of encoded provenance and the transmitted data is intact or not.

5. Performance Evaluation and security Analysis

a) Experimental result

To assess the performance of this scheme, we implement the experimental simulation with OPNET 14.5, where the simulated network has scaled with randomly deployed 800 nodes over 1000 m * 1000 m monitoring area, and the communication radius is 70.0m. In this sensor environment, each data packet of a node has structured with 8 init32 type data fields as 256 bits pay load length. The generated experimental results have analyzed using MATLAB R2016a.

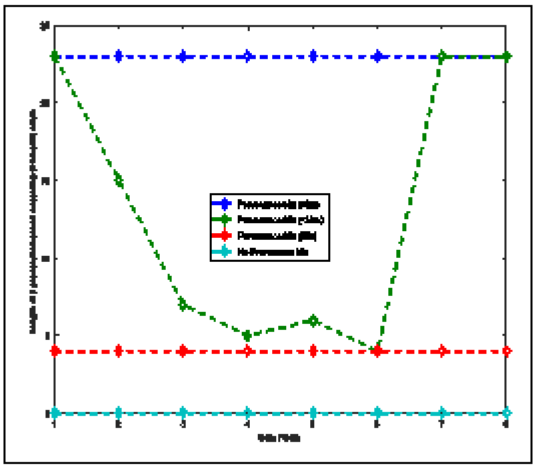

Figure 6: The length of provenance chunk measured between the data fields of a 256 bits data payload.

The proposed scheme verifies the sensed data with different bit-shift windows to determine its variable length. The bit-shift windows are the several partitioned bit widths of the total length of a data field and each partitioned segment is mapped to a unique bit-shift value. Table (1) reports a brief relationship between the mapped bit-shift values and the bit-shift windows that followed in this experiment. Here, the coverage bit-width for any bit-shift window has been observed from 8 bits to 28 bits and defines the bit-shift value corresponds to it. It describes that a 28 bits sensor value could encode atleast 4 bits provenance chunk with it. However, the bit-width (in bits) of a bit-shift window based on init32 type data field might be ranged between[1,32]. Fig. (5) illustrates the plot on allocated bit-shift values against the size (bit-width) of numerous partitioned segments as bit-shift windows. However, a sensed data can encode a possible maximum and minimum length of provenance chunk if it does satisfy the condition defined in (1) and (2) respectively.

Table 2: Measured performance of the network Parameters.

routing algorithms	average delay	maximum delay	packet reception rate(%)
GPSR	97 ms	100 ms	98.2759
GPSR-E	110 ms	139 ms	99.4709

Delay time unit denoted as, ms = millisecond.

From Fig. (6), it can be demonstrated that a data packet may transport a distinct length of provenance chunk through the 8 data fields of a data packet, where the given payload was 256 bits. Since the data has been analyzed based on the length of data in bits against the size of a data field, any arbitrary length of provenance chunk less than or equal to its selected bit-shift value can encode with the sensed data.

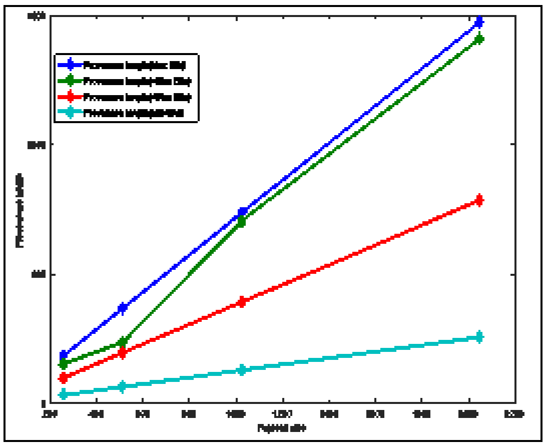

Figure 7: Several length of provenance encoding capacity over distinct length of payload size.

Fig. (7), shows the scalability and the adaptability of the provenance encoding. The illustration indicates that provenance encoding capacity could be enlarged by increasing the payload size of a data packet. The efficiency of this scheme has examined by performing a network simulation, where the sensor nodes used two different positions based routing protocol: (i) general GPSR [8] (ii) improved GPSR based on energy (GPSR-E) gradient [12] to discover the traffic route towards the SN. During the simulation, the transmission efficiency due to the provenance encoding process has been examined in two different scenarios. In the first scenario, a particular set of sensor data considered 4 bits provenance chunk to encode without applying analyzing the length of each data according to bit-shift windowing. In the second scenario, using the DDP scheme each variable length of sensor data has examined to encode a variable length of provenance chunk. Each scenario has trialed several times with each position based routing protocol during the simulation.

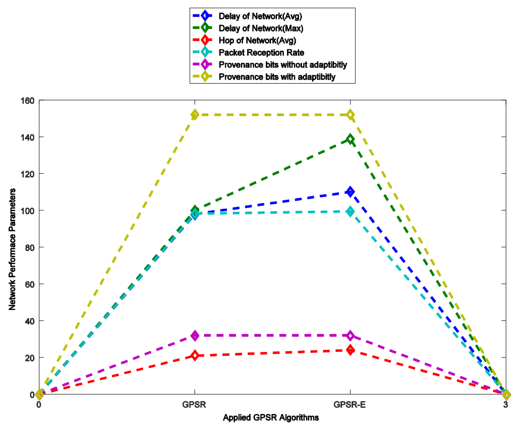

Figure 8: Measured transmission efficiency with two positioned based routing protocol.

According to Fig. (8), it may illustrate that the encoding capacity with adaptability has a five times greater increase compared to without adaptable scheme. However, the number of traversed network hops differs a little between GPSR and GPSR-E. Table (2) reports the network parameters performance overhead. Though there has been a rise in terms of navigated nodes and transmission delays in GPSR-E, the packet reception rate has been greater compared to GPSR.

b) Complexity

Complexity in encoding step: A source node uses $n$number of data fields of a data packet to transmit data provenance that corresponds to n number of sensor data. Thus, the computational cost for encoding a provenance chunk to each d_i that belongs to each df_i is $O(n * \log n)$.

Complexity in decoding step: The SN performs the decoding process for $n$number of received encoded data per packet that obtain after the extraction and decryption process. Hence, considering two major decoding steps: (i) recover the $a\lambda_i$, (ii) apply the recovered $a\lambda_i$ on the corresponding $\widehat{ed_\iota}$, the provenance decoding complexity is $O\big(n(n+1)+n(n+1)\big) = O(2n^2+2n) \cong O(n^2)$.

c) Security Analysis

An adversary may inspect data packets in a particular data flow by utilizing many packet sniffing tools. Thus, the data of interest may capture, copy and formally send it to the destination. Such adversary activities are reasonable to get informed about the data carried through the data fields of a packet. We analyzed the security properties of our scheme on avoiding the data provenance to disrupt in different ways. However, the following assertion illustrates the DDP security functions can avoid such detection and privacy threat.

Claim 1: It is not possible to detect the presence of encoded provenance without knowing about the provenance by observing the data field characteristics (i.e. the length of a data field) of a data packet in a data stream. Even if the attacker is aware of data provenance, it is difficult to break the trustworthiness of encoded data by retrieving the provenance information.

Justification: As basic security primitive, the data confidentiality is achieved through encryption of each encoded data. Each encoded data in this scheme is encrypted with a unique key k_i that authenticates with the last modification time of data t_i, selected $\lambda_i$along with other unique parameters, thus it is not sufficient for an adversary trying recovery with only the key besides knowing the df_i characteristics of a packet.

Claim 2: An authorized parties (e.g., the SN) only get access to the data and provenance that received, and perform necessary integrity check for the provenance.

Justification: In this scheme, a particular timestamp related to each data and its position index in a particular data set, are the dependencies that are required in generating a data specifick_i by a data source. Thus, the SN also needs to resolve the above dependencies to perform a successful key synchronization process. If the key synchronization validates correctly the $\widehat{k_\iota}$andk_i^{new}, indicates the claimed data originator is authenticated and an appropriate $a\lambda_i$ is returned for decoding the related $\mathcal{D}(\widehat{ed_\iota})$. Thus, the SN also can be assured that received keys are originated from the claimed data source.

Claim 3: (Freshness) SN can skip the detection if an adversary replicates the provenance between the data fields. However, an adversary may change the order of data that stored in eachdf_i, in a way that may cause of receiving an unordered sequence of data at the SN. Regardless of such changes, the SN can identify such malicious activity. Later, by applying some supplement mechanism such attacks can be prevented and data can be recovered as a fresh format.

Justification: If the $\widehat{ed_\iota}$ in a particular df_i is changed, the synchronization of $\widehat{k_\iota}$ and k_i^{new} could be failed or an incorrect $a\lambda_i$ could return upon the index of respective $\mathcal{E}(\widehat{ed_\iota})$. Thus, the SN determines the pv_i encoded to a particular $\mathcal{D}(\widehat{ed_\iota})$ is not correct, or the position of $\mathcal{E}(\widehat{ed_\iota})$among the data fields is changed by an adversary. Hence, the desired freshness of each df_i specific d_i and pv_i can be assured with the correct order of received data and key synchronization process.

Claim 4: (Unforgeability) An adversary cannot claim that a valid provenance is modified since the provenance encoded data that persist in a data field of a data packet belongs to another data field of the same packet.

Justification: Assume that a malicious routing node $n_m$wants to insert the provenance pv_a of a particular df_i into another $df_i$of a pk_i. To perform such an attack, n_m has to follow the timing characteristic of packets flow. However, before creating any alteration on data, n_m has to be certified as an authenticated packet recipient. Even if, the adversary authenticated itself, figure out the storage quota for a pv_i in a df_i is quite challenging. Being unaware of k_i and exact $a\lambda_i$, it is rather difficult to decrypt and signify the designated pv_i from $\mathcal{E}(\widehat{ed_\iota})$. However, the following operations: decryption, bit-shift value recovery, provenance chunk detection, and its modification by analyzing the data stream eventually increase the delay between the data packets in mid-transmission. Moreover, revealing the secret keys $\{k_1, k_2, \dots, k_n\}$ is not sufficient to retrieve the appropriate $a\lambda_i$ for each $\widehat{ed_\iota}$without knowing the k_i generation procedure. Hence, retrieving $a\lambda_i$ with key synchronization technique also introduce an additional cost. Too much delay that doesn't match the adjustable timing for regular data packets that transmitted towards the SN, will certainly be detected at the SN. However, the data integrity check at the SN may detect and prevent such provenance forgery attack using the data specific secret key authentication and timing irregularities as well.

6. Discussion

Cost Analysis: The encoding capacity according to our proposed method has been analyzed in contrast to the scheme of adding secret data into redundant bits that have been observed between the data fields [19]. This scheme was proposed to ensure the integrity of transmitted sensor data. In this methodology, the considered length of each data field is of 2 bytes (16 bits). The length of sensor data is fixed and assumed to be 12 bits that are less than the total length of each df_i. Hence, the scheme identifies a fixed length of redundant bits $r_i = 4$ bits in each of n data fields. This space usually used to add blank character or encode other information. Thus, total length of secret information (e.g., provenance) can be calculated as, $length(PV) = n * r_i = \sum_{i=1}^{n} r_i$, where r_i is fixed. In this method, the provenance capacity is constant.

Table 3: Cost comparison between DDP and Fixed length Approach.

Data Properties	The DDP Scheme	Fixed Redundant Length Approach [5]
Size of data field	16 bits	16 bits
Allocated bits for data	7 bits	12 bits
Abused space	0 bits	5 bits
Remark	able to encode large provenance chunk	encoding capacity is limited

In this case, increasing provenance capacity depends on either increasing the numbers of data fields or increasing the length of the data field. In contrast, our protocol doesn't limit the space within thedf_i for any of the sensor d_i; d_i uses a λ_i value to shift its bits according to bit-shift windowing; create $\gamma_i$vacated bits over the shifted code of d_i to add the variable length ofpv_i. Thus,the maximum length of data provenance can be calculated between n number of data fields as, $length(PV) = N = \sum_{i=1}^{n} \gamma_i$, where γ_i is not fixed due to selecting distinct λ_i. Thus, the DDP method maximizes the utilization of each data field.

Table. (3) evaluates the DDP and the fixed redundant bits schemes in terms of storage dimension with 2 bytes data field for the 7 bits sensor data. It shows the fixed redundant bits approach abuse 5 bits due to limit the space in 12 bits for the data in each df_i.On the contrary, the DDP scheme attains 9 bits free space (without abusing any bits) for the provenance chunk into the data filed due to its bit-shift windowing characteristics for the same size of data. Moreover, If the length of a data would $(12 + 1)$ bits, the DDP scheme can still attain minimum 3 bits for the provenance chunk due to its adaptability. The computational cost of our proposed method on computing the group XOR operation relatively equivalent to the method built on fixed redundant bits. Furthermore, the DDP scheme computes the coverage bit width query, bit-shift value selection. Meanwhile, our scheme also performs the secret key generation and data encryption computation. Though our scheme performs several operations before entering into the transmission channel, the introduced delay by each source node doesn't exceed the usual magnitude as network jitter of sensor network. However, conferring the data in Table (2), improving the routing algorithm may increase the transmission efficiency besides saving energy in a resource-constrained sensor environment. Our proposed method can also be signified as a reversible technique of data hiding because of distinguishing both the secret provenance and original form of data [2]. To apply our scheme in sensor environments, a secure mechanism is required to distribute knowledge about the bit-shift values, the data timestamps and secret keys that correspond to each data to the designated recipient.

7. Conclusions

In this paper, our proposed framework tried to identify and solve a novel problem of encoding provenance in the distributed form to ensure secure transmission of the sensor data stream. Experiment outcomes have been conducted with regard to the data transmission efficiency, performance of the network parameters besides the encoding capacity, which confirms the scalability and adaptability of this scheme. In the future, we will examine the efficiency of this scheme against several adversary attacks and control the error if the encryption failed besides optimizing the computational cost.

References

[1] Jizhi, W., Shujiang, X., Min, T., and Yinglong, W., "The analysis for a chaos-based one-way hash algorithm", International Conference on Electrial and Control Engineering, (2010), pp. 4790-4793.
[2] An, L., Gao, X., Li, X., Tao, D., Deng, C., Li, J., et al.: Robust reversible watermarkingvia clustering and enhanced pixel-wise masking. IEEE Trans. Image Processing 21(8), 3598-3611 (2012).
[3] Dai, C., Lin, D., Bertino, E., Kantarcioglu, M.: An approach to evaluate data trustworthiness based on data provenance. In: Workshop on Secure Data Management, pp. 82-98. Springer (2008).
[4] Fang, J., Potkonjak, M.: Real-time watermarking techniques for sensor networks. In: Security and Watermarking of Multimedia Contents V, vol. 5020, pp. 391-403. International Society for Optics and Photonics (2003).
[5] Guo, H., Li, Y., Jajodia, S.: Chaining watermarks for detecting malicious modifications to streaming data. Information Sciences 177(1), 281-298 (2007).
[6] Hasan, R., Sion, R., Winslett, M.: The case of the fake picasso: Preventing history forgery with secure provenance. In: FAST, vol. 9, pp. 1-14 (2009)
[7] Houmansadr, A., Kiyavash, N., Borisov, N.: Multi-flow attack resistant watermarks for network flows. In: International Conference on Acoustics, Speech and Signal Processing, pp. 1497-1500. IEEE (2009).
[8] Karp, B., Kung, H.T.: Gpsr: Greedy perimeter stateless routing for wireless networks. In: Proceedings of the 6th annual international conference on Mobile computing and networking, pp. 243-254. ACM (2000).
[9] Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., Njilla, L.: Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In: Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid '17, pp. 468{477. IEEE Press, Piscataway, NJ, USA (2017). DOI 10.1109/CCGRID.2017.8.
[10] Lim, H.S., Moon, Y.S., Bertino, E.: Provenance-based trustworthiness assessment in sensor networks. In: Proceedings of the Seventh International Workshop on Data Management for Sensor Networks, pp. 2-7. ACM (2010).
[11] Liu, B., Chen, L., Zhu, X., Qiu, W.: Encrypted data indexing for the secure outsourcing of spectral clustering. Knowledge and Information Systems pp. 1-22 (2018).
[12] Liu, Z., Feng, X., Zhang, J., Li, T., Wang, Y.: An improved gpsr algorithm based on energy gradient and apit grid. Journal of Sensors 2016 (2016).
[13] Mattoso, M., Glavic, B.: Provenance and annotation of data and processes. In: 6th International Provenance and Annotation Workshop (IPAW), vol. 9672, pp. 280-292. Springer (2016).
[14] Mauve, M., Widmer, J., Hartenstein, H.: A survey on position-based routing in mobile adhoc networks. IEEE network 15(6), 30-39 (2001).
[15] Pérez, B., Rubio, J., Sáenz-Adán, C.: A systematic review of provenance systems. Knowledge and Information Systems pp. 1-49 (2018).
[16] Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V., Culler, D.E.: Spins: Security protocols for sensor networks. Wireless networks 8(5), 521-534 (2002).
[17] Simmhan, Y.L., Plale, B., Gannon, D.: A survey of data provenance in e-science. SIGMOD Rec. 34(3), 31-36 (2005). DOI 10.1145/1084805.1084812.
[18] Sultana, S., Shehab, M., Bertino, E.: Secure provenance transmission for streaming data. IEEE Transactions On Knowledge And Data Engineering 25(8), 1890-1903 (2013).
[19] Sun, X., Su, J., Wang, B., Liu, Q.: Digital watermarking method for data integrity protection in wireless sensor networks. International Journal of Security and Its Applications 7(4), 407-416 (2013).
[20] Wang, B., Sun, X., Ruan, Z., Ren, H.: Multi-mark: multiple watermarking method for privacy data protection in wireless sensor networks. Information Technology Journal 10(4), 833-840 (2011).
[21] Wang, X., Chen, S., Jajodia, S.: Network flow watermarking attack on low-latency anonymous communication systems. In: IEEE Symposium on Security and Privacy, pp. 116-130. IEEE (2007).
[22] Wang, X., Govindan, K., Mohapatra, P.: Provenance-based information trustworthiness evaluation in multi-hop networks. In: Globecom, pp. 1-5 (2010).
[23] Wynbourne, M.N., Austin, M.F., Palmer, C.C.: National Cyber Security Research and Development Challenges Related to Economics, Physical Infrastructure and Human Behavior: An Industry, Academic and Government Perspective. Institute for Information Infrastructure Protection (2009).
[24] Yi, X., Paulet, R., Bertino, E.: Homomorphic encryption and applications, vol. 3. Springer (2014).
[25] Protopopescu, Vladimir A and Santoro, Robert T and Tolliver, Johnny S, ``Fast and secure encryption-decryption method based on chaotic dynamics," U.S. Patent 5479513A, Dec. 26, 1995.

Author Profile

Mohammad Amanul Islam (Aman) is pursuing his Ph.D. degree in Computer Science and Technology from the Xidian University, Xi'an, China. His research interests include computer and network security, applied cryptography. He is also interested to the application of secure computation technology, and data mining techniques to enhance network security and data privacy.