International Journal of Engineering & Technology, *7 (3.27) (2018) 285-289*

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Safe Diffusion of Provenance in Wireless Sensor Networks Using in-Packet Bloom Filter Provenance Encoding Technique

Sujesh P Lal[1]*, P.M. Joe Prathap[2]

[1]*Research Scholar, Department of Computer Science and Engineering, Sathyabama University, Chennai, India.*
[2]*Professor, Department of Information Technology, RMD Engineering College, Chennai, India.*
** Corresponding author E-mail:sujeshlal@fisat.ac.in*

Abstract

Wireless Sensor Networks (WSN) comprises of tiny wireless sensor nodes for continuous observation of physical or environmental conditions. Sensor networks are increasingly deployed in decision-making infrastructures. They are widely used for battlefield monitoring systems and Supervisory Control and Data Acquisition (SCADA) systems. Making decision makers aware of the trust worthiness of the collected data is crucial. WSNs are used by Business Applications. These applications depend on trustworthy sensor data to control business processes. It is important to ensure the trustworthiness of the data generated from sensor nodes so that effective decisions can be made. Making decision makers aware of the trustworthiness of the collected data is crucial. WSNs are used by Business Applications. These applications depend on trustworthy sensor data to control business processes.

We have proposed a different approach for provenance diffusion for WSN using Bloom filters. The major security attributes of the scheme are freshness, confidentiality and integrity. Experimental characteristics and results evaluating the scheme output the efficiency of the provenance encoding and its transmission.

Keywords: *Bloom filter, encoding, provenance, WSN.*

1. Introduction

Advances in hardware and network technologies enable the development of large-scale sensor networks. Sensor networks are deployed everywhere. Wireless Sensor Networks (WSNs) has the ability to control and monitor different physical environments. Erroneous or non-trustworthy sensor data are due to Intentional misbehaviour and unintentional errors. Unintentional errors of the sensor data are caused by mal-function of the hardware malposition of the node or exhausted batteries. Intentional misbehaviour is caused by attackers, exploiting security vulnerabilities of WSNs. Security for WSN has often to be balanced for saving energy and the limited resources (memory, CPU) that are available. WSN node are often easily accessible and rarely tamper-resistant. Hijacking of nodes and extraction of cryptographic material is easy and gives the attacker the possibility to add malicious nodes or inject bogus data into the network.

The trustworthiness [19] of the collected data and making decision makers aware of the trustworthiness of these data become crucial. A possible approach to this problem is to associate each data item with a trust score. The trust score [2,4] provides an indication about the trustworthiness of the data item and can be used for data comparison or ranking. If a data item has the highest trust score in a data set, then we can say that the data item is the most trustworthy compared with the other data items in the set.

A multi-hop wireless sensor network consists of a number of sensor nodes and a base station. The node in a WSN has three roles, as a data source, a data forwarder and data aggregator [4,15]. The sensed information is processed by the sensor node and the data is transmitted to the base station. Packets are sent to the base station through sensor nodes when a data source acquires data. The data forwarder sends the received packets to the base station. The aggregator joins more than one packets into a larger packet and sends the newly created packet to the base station [5,6]. When more than one packets are aggregated the energy requirement for transmitting the aggregated packet is lower than the energy requirement when the packets are independently transmitted.

2. System Model

In a multi-hop wireless sensor network, all nodes transmit data on the basis of a local clock, and they do not have access to global timing information. We assume that nodes that transmit data to other node or to one or more sink nodes may fail, i.e. packet transmitted is not delivered or received at the intended destination node. So it is assumed to be a lossy channel. Each sink node is the root of a multi-hop tree that consist of 1:n parent-child relations. All the sensor nodes have the dual functionality, like generating data and forwarding received as well as generated data to other nodes. A finite FIFO send queue is used by each node. A packet is added to the queue on generating or receiving data and transmitting the contents of its send queue to its parent node.

The base station[6] is the central control authority of the routing tree, which does not have any resource constraint. Sensor nodes monitor their environment and the generated data is periodically communicated to the base station or the designated cluster head, if any. An event is monitored by a number of sensors. On a particular time period, independent monitoring is required at sink nodes, if any, or the base station.

3. Provenance Model

A multi-hop WSN consist of a number of nodes are usually modeled as an acyclic directed graph G(N,E), where N is the set of nodes and E is the set of edges, defined as:

N={n_i | n_i is a network node with an identifier *i*}, a set of nodes.
E={e_{ij} | e_{ij} is an edge connecting network nodes n_i and n_j}, a set of directed edges between nodes.

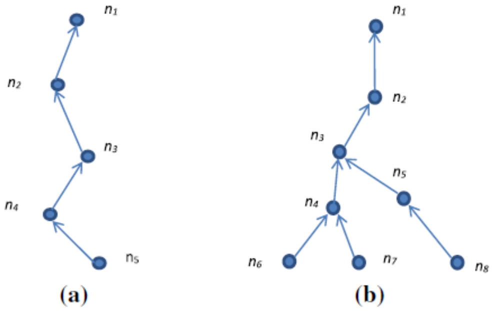

Fig. 1: a) Simple provenance; b) Aggregated provenance

In the fig.1 Provenance Graphs with nodes numbered n1, n2, etc and BS is the Base stations are represented.

4. Provenance Encoding Using Bloom Filter

A Bloom filter is a bit array on *m* bits, all set to zero. Here we represent the Bloom filter as a data structure for probabilistic representation of a set of data items $D=\{d_1, d_2, ..., d_n\}$ using an array of *m* bits with *k* different hash functions $h_1, h_2, ..., h_k$. The data item d_i is associated to the output of hash function h_i, which is mapped uniformly to the range [0, m-1] and is represented as an index pointing to a bit in an *n*-bit array. Bloom filter is represented as $\{b_0, b_1..., b_{m-1}\}$ with an array initial value, for all b_i, set to 0.

By using Bloom Filter[6,9] approach false positive matches are possible on the other hand false negative matches are not possible. False positive error is the incorrect rejection of a true null hypothesis. This type of errors leads one to conclude that a supposed effect or relationship exists when in fact it does not. Examples of false positive errors include a sensor node shows a data to have a value when in fact the data does not have the value or which represents a *null* value. A false negative error is the failure to reject a false null hypothesis. Examples of false negative errors would be a sensor node shows a *null* value when in fact the data does have a value.

Constructing Bloom Filters

Consider a set $D=\{d_1, d_2, ..., d_n\}$ of *n* elements. Bloom filters describe membership information of D using a bit vector V of length *m*. For this, *k* hash functions, $h_1, h_2, ..., h_k$ with $h_i : X \to \{1..m\}$, are used as described below:

The following procedure builds an *m* bits Bloom filter, corresponding to a set D and using $h_1, h_2, ..., h_k$ hash functions:

```
Function BloomFilter(set D, hash_func, int m){
return BF;
BF = allocate m bits initialized to 0;
for each $a_i$ in set D{
for each hash_func $h_j${
BF[$h_j(a_i)$] = 1;
}}
return BF;
}
```

Therefore, if a_i is member of a set D, in the resulting Bloom filter V all bits obtained corresponding to the hashed values of d_i are set to 1. Testing for membership of an element *e* is equivalent to testing that all corresponding bits of V are set:

```
Function MemberCheck (e, filter, hash_func) {
return 1 or 0;
for each hash_fun $h_j${
 if BF[$h_j$(elm)] != 1 return 0;
}
 return 1;
}
```

Filters, BF, can be built incrementally as new elements are added to a set the corresponding positions are computed through the hash functions and bits are set in the filter. Moreover, the filter expressing the reunion of two sets is simply computed as the bit-wise OR applied over the two corresponding Bloom filters.

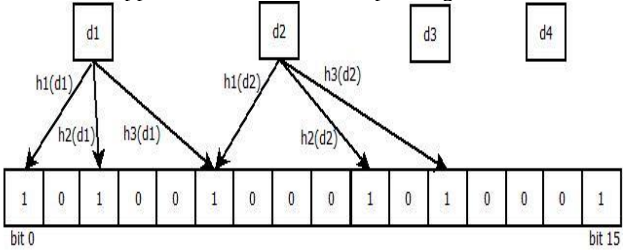

Fig. 2: A sample bloom filter with k=3, n=4, and m=16

Initially the array values are set to 0s, each item in the d_i is being hashed *k* times and the bits corresponding to the values are then set to 1 in the bit array.

One illustrious feature of Bloom filters is that there is a clear balance between the size of the filter and the rate of false positives. Observe that after inserting *n* keys into a filter of size *m* using *k* hash functions, the probability that a particular bit is still 0 is:

$$p_0 = \left(1 - \frac{1}{m}\right)^{kn} \approx 1 - e^{-\frac{kn}{m}} \qquad (1)$$

(Here we assume perfect hash functions that spread the elements of D evenly throughout the space {1..*m*}. In practice, good results have been achieved using MD5 and other hash functions). Hence, the probability of a false positive (the probability that all *k* bits have been previously set) is:

$$p_{err} = (1 - p_0)^k = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{-\frac{kn}{m}}\right)^k \qquad (2)$$

In Formula-2 p_{err} is minimized for $k = \frac{m}{n} \ln 2$ hash functions.

In practice however, only a small number of hash functions are used. The reason is that the computational overhead of each hash additional function is constant while the incremental benefit of adding a new hash function decreases after a certain threshold.

Formula-2 is the base formula for engineering Bloom filters. It allows, for example, computing minimal memory requirements (filter size) and number of hash functions given the maximum acceptable false positives rate and number of elements in the set.

$$\frac{m}{n} = \frac{-k}{\ln\left(1 - e^{\frac{\ln p_{err}}{k}}\right)} \text{(bits per entry)} \qquad (3)$$

The main design commutations are the number of hash functions used (driving the computational overhead) and the size of the filter and the collision (error) rate[12]. Formula-2 is the main formula to tune parameters according to application requirements.

For inserting an element d_i belongs to D into a bloom filter, d_i is hashed with all the k hash functions generating the output $h_1(d_i)$(1<=i<=k). To examine the membership of an item d' from D, the bits at indices $h_i(d')$(1<=i<=k) is checked, if found any 0 values, then d' is not belongs to D, else all the bits are set to 1 with a high probability of being d' belongs to D. If a hash collision is occurred false positive method is used to make the membership verification [6] and marks the indices $h_i(d')$ to 1. Consider k number of hash functions, m as the bloom filter size (16bit in Fig. 1) and maximum number of elements in D is P [16]. The false positive probability is equal to that of getting 1 in all the k array positions computed by the hash functions while checking the membership of an element that was not inserted in the bloom filter.

Each node in the packet path encodes its ID into an array through the bloom filter and then adds the array to the passing by packet. All elements in the array are set to 0 before the source of data node ID is encoded [21]. On receiving the packet at destination, the base station tests all the nodes in the wireless sensor network to get the nodes in the packet path.

5. Provenance Collection and Encoding

A distributed algorithm is used to encode provenance in an in-packet bloom filter [16], and a centralized algorithm is used by the base station to decode the provenance [10]. A unique sequence identification number is attached to the forwarded packet along with data value and an in-packet bloom filter which holds the provenance also transmitting the provenance graph vertices over an in-packet bloom filter.

For a data packet, provenance encoding is the generation of vertices in the provenance graph and inserting them into the bloom filter. Each vertex originates at a node and represent s the provenance record of the host node. A *vertex id* uniquely identifies a vertex. The *vertex id* is generated for each packet based on the sequence number (*sqc)* of the packet and the secret key (K_i) of the host node. A cryptographic function is used to produce this *id* in a secure way. For a given data packet, the *vertex id* of a vertex of the node n_i is computed[13] as:

$vertex\_id_i=GENvertex\_id(n_i,sqc)=SB_{Ki}(sqc)$

where SB is a secure block cipher such as DES, AES, etc.

A bloom filter is created whenever a source node generates a data packet, and initialized it to all 0's, and is referred to as ibf_0. According to the above equation the source node generates a vertex and inserts the *vertex_id* into it and transmits the bloom filter as a part of the packet. On receiving the packet, the intermediate node n_j performs data aggregation as well as provenance aggregation [4] [18]. If n_j receives data from one child n_{j-1}, then n_j aggregates the partial provenance contained in the packet with own provenance record. Now the in-packet bloom filter (ibf_{j-1}) belongs to the received packet has a partial provenance (the sub-path provenance graph from source to destination, n_{j-1}. If the node n_j has more than one child, the node generates an aggregated provenance from the partial provenance received from its child nodes and from its own provenance record.

Initially, n_j calculates a bloom filter ibf_{j-1} by performing bitwise OR operation on the in-packet bloom filters received from received from its children. ibf_{j-1} constitutes a partial but aggregated provenance from all the child nodes. In both the case, the conclusive aggregated provenance is generated by encoding the provenance record of n_j into ibf_{j-1}, and the node nj creates a vertex using the Eq.(1) , inserts t he *vertex_id* into ibf_{j-1} which is then cited as ibf_j.

Provenance decoding and verification

The base station conducts the verification process [13] to check the integrity of the transmitted provenance when a data packet is received at the base station. First the base station initializes a bloom filter with all zeros, and the bloom filter is updated by adding the *vertex_id* of each node in the path *P* and inserting this *id* into the bloom filter [15], which interprets the encoded provenance. The base station now compares the bloom filter to the received in-packet bloom filter *ibf*. If the comparison fails, it indicates the change in the data packet transmission path of a bloom filter modification attack.

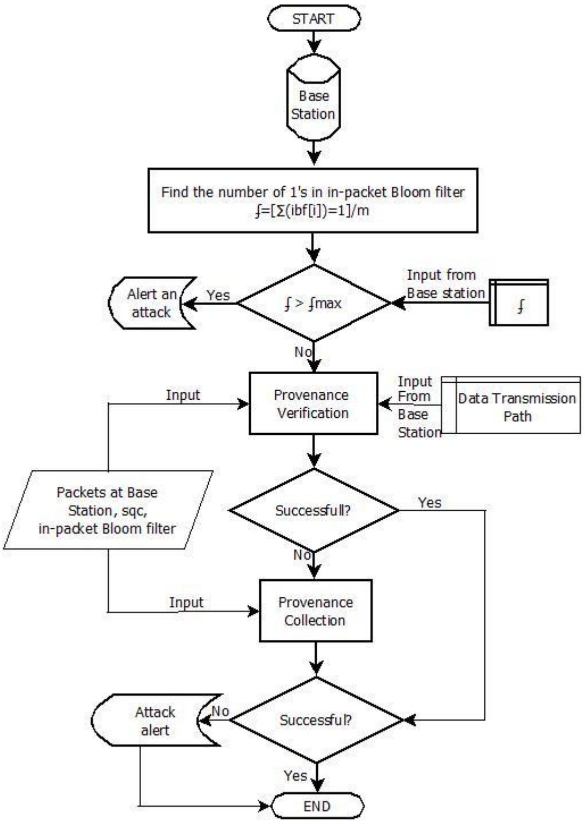

Fig. 3: Provenance workflow at the base station

After the successful provenance verification, if it is found that there is a natural change in the data flow path, we can determine the path correctly, otherwise an error has occurred probably in the form of an attack. One possible attack is to make all the provenance bits to 1, signifies the presence of every provenance nodes. The threshold introduced in the Fig. 3 Provenance workflow graph, ʃ>ʃ$_{max}$; where ʃ is the density metric, indicates the number of 1's in the provenance information in the in-packet bloom filter. If the density metric is below or equal to the threshold, ʃ>ʃ$_{max}$, the provenance is considered to be valid. As the bloom filter has *m* elements and *k* hash functions, the upper bound for the number of 1's in a bloom filter is set as *n*ʃ$_{max}$. Considering all these bounds, for an attacker, there is a very little chance to flip some bits to modify a legitimate node. Attacker [10] has to identify the *k* bit positions to the corresponding node, which is different for each packet, which makes an attacker unsuccessful in this attack. Random modification of some bits is detected at the verification phase and makes it unsuccessful during the provenance collection phase.

6. Experimental Analysis and Results

In this section, we present our experimental performance analysis. We first describe the experimental environment, and then present the experimental analysis and results.

Experimental Environment

We have done simulation using OMNeT++ for wireless sensor network with a minimum of 2 nodes and a maximum of 100 nodes. Simulation model integrated a simple and robust technique to handle topological changes, since encoded provenance is compared with the calculated provenance at each intermediate node, we can reduce the overhead of carrying all the provenance information to the base station. So the decoding errors are minimized and speed up the provenance transmission.

The objective of our experiments is to analysis the competence and effectiveness of our approach for the encoding and decoding of data provenance. To evaluate the efficiency, we measure the elapsed time for processing a data item with our cyclic framework in the context of a large scale sensor network and a large number of data items. To evaluate the effectiveness, we simulate an insertion of incorrect data items into the network and shown that trust scores rapidly reflect this situation. For simplicity, we model our sensor network as an *N*-ary complete whose maximum number of inputs that the output of a network node can feed (fanout) and depth are *N* and *d*, respectively. We changed the values of *N* and *d* to control the size of sensor networks for assessing the scalability of our trust framework.

Experimental Results

The study shows the balance between trustworthiness and provenance similarity of data. If the dissimilarity increases the transmission overhead is also increases. To demonstrate the performance of the probabilistic provenance transmission in a dynamic and asymmetric network we have performed OMNet++ simulation. And the result show that the probabilistic provenance transmission with the Bloom filter encoding scheme can utilize up to 40-60% less energy and converge with very less about 40% fewer packets than the traditional approach [], which increases the network life as the energy consumption is less.

The decoding process for probabilistic Bloom filter provenance method requires a priori knowledge of order of nodes. Since topological changes are usual in WSNs, we need to keep node order information up-to-date so that encoding methods can correctly decode provenance. To study link change, we have done a simple simulation experiment. Fig. 4 shows a snapshot of link changes for two thousand packets transmitted from a source to the base station in a highly asymmetric 10X10 grid network. The link change of a packet denotes the number of next hop changes on the way from the source node to the base station with respect to the path followed by the preceding packet. The goal of our approach is to observe changes and transmit provenance of nodes that are part of the changed links with order information among them.

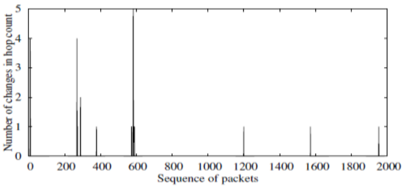

Fig. 4: Topological changes of packets in a grid network

Provenance Decoding Error: The provenance decoding process retrieves the provenance from the ipBF (in-packet Bloom-Filter. To measure the accuracy and efficiency of our provenance scheme, we calculate decoding error in both the verification and collection phases. The verification fails when there is a data flow path change or upon a Bloom-Filter modification attack. Provenance verification failure rate (VFR) measures the ratio of packets for which verification fails. [20]

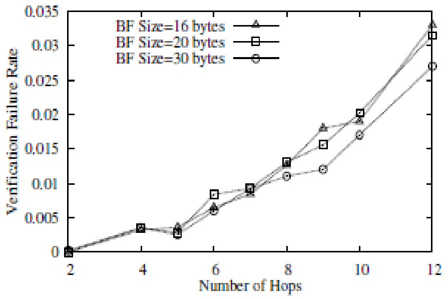

Fig. 5: VFR for paths of 2 to 12 hops with various bloom filter sizes

For each path length, the verification failure rate is averaged over 1000 distinct paths. The results show that the provenance verification process fails only for a very small fraction of packets. For most of the packets, verification process is sufficient to retrieve the provenance. The expensive provenance collection process is executed only for a very few packets when verification fails. Here verification failure rate increases linearly with the increase in path length. Also verification failure rate is not remarkably effected by Bloom-filter size, proving that even small Bloom-filter sizes provide good security.

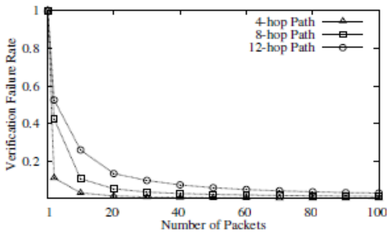

Fig. 6: The variation of provenance VFR over time

The above figure shows the variation of provenance verification failure rate (VFR) over time as the number of packet transmissions increases. As the network gets stable with time, the data paths do not change often and hence the VFR approaches 0.

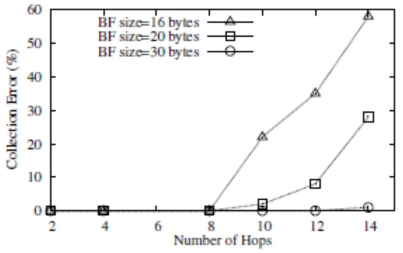

Fig. 7(a): Provenance collection analysis with number of hops

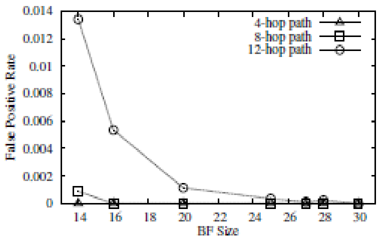

Fig. 7(b): Provenance collection analysis with BF size

Figures 7(a) and 7(b) shows the percentage of provenance collection error for different number of hops and the corresponding false positive rates, respectively. [21]

Fig. 8(a) shows the false positive rate as a function of the number of hash functions used. The size of the Bloom filter is 32 bits per entry (m/n=32). In this case using 22 hash functions minimizes the false positive rate. Note however that adding a hash function does not significantly decrease the error rate when more than 10 hashes are already used.

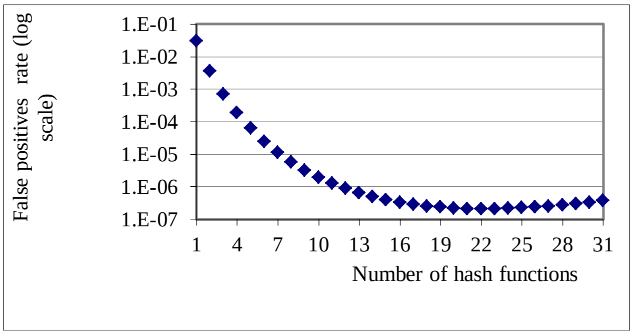

Fig. 8(a): Representation of false positive rate

Size of Bloom filter (bits/entry) as a function of the error rate desired. Different lines represent different numbers of hash keys used. Note that, for the error rates considered, using 32 keys does not bring significant benefits over using only 8 keys, which is shown in Fig. 8(b)

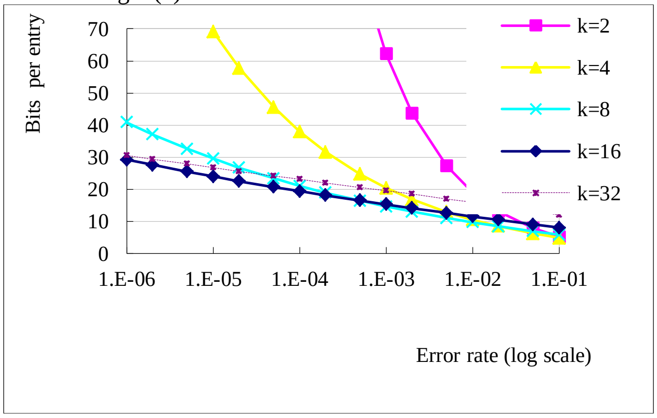

Fig. 8(b): Size of BF as a function of the error rate

7. Related Work

Provenance for a packet document only its aggregation and forwarding information [12]. The size of the provenance increases when there is an increase in the transmission hops and network nodes in a wireless sensor network. Even if there are a number of encoding schemes are available, when considering very large scale wireless sensor networks, these encoding methods suffer from lack of something to reach a particular standard, such as, (a) the broadcast flooding [9] can drain the power on each node, (b) compression techniques fail to accommodate provenance information in a packet as the size of the provenance exceeds the capacity of the packet.

8. Conclusion

We address the issues of transmitting provenance for wireless sensor networks by discussing the Bloom filter method of provenance transmission scheme. We have proposed a novel approach for provenance transmission for WSN using bloom filters. The major security attributes of the schemes are freshness, confidentiality and integrity. Experimental characteristics and results evaluating the scheme output the efficiency of the provenance encoding and transmission. In future work we focus on ranking the information trustworthiness of provenance data in wireless sensor network, also study how good a security framework can observe and respond to different attacks.

References

[1] Hyo-Sang L, Moon YS & Bertino E, “Provenance based trustworthiness assessment in sensor networks”, *Proceedings of the seventh international workshop on data management for sensor networks*, (2010), pp.2-7.

[2] Ramachandran A, Bhandankar K, Tariq M & Feamster N, “Packets With Provenance”, *Technical Report Gt-Cs-08-02, Georgia Tech*, (2008).

[3] Wang C, Hussain SR & Bertino E, “Dictionary based secure provenance compression for wireless sensor networks”, *IEEE Trans Parallel Distributed Systems*, Vol.27, No.2,(2016), pp.405– 418.

[4] Sultana S, Ghinita G, Bertino E & Shehab M, “A Light weight Secure Provenance Scheme for detecting provenance forgery and packet drop attacks in Wireless Sensor Networks”, *IEEE Trans Dependable Secure Computing*, Vol.12, No.3, (2015), pp.256-269.

[5] Zhou W, Zhou X, Yang F & Li X, “Contact-based trace back in wireless sensor networks.”, *International conference on wireless communications, networking and mobile computing*, (2007), pp. 2487-2490.

[6] Sujesh Lal P & Joeprathap PM, “A novel approach for Provenance Transmission in Wireless Sensor Networks”, *Journal of Advanced Research in Dynamical & Control Systems*, Vol.10, No.3, (2018), pp.321-326.

[7] Xue K, Ma C, Hong P & Ding R, “A temporal credential based Mutual Authentication and Key agreement Scheme for Wireless Sensor Networks”, *International Journal of Network and Computer Applications*, Vol.36, No.1,(2013), pp.316-323.

[8] Salmin S, Mohamed S & Elisa B, “Secure provenance transmission for streaming data”, *IEEE Trans Knowledge Data Eng*, Vol.25, No.8, (2013), pp. 1890-1903.

[9] Jiang Q, Ma J, Lu X & Tian Y, “An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks”, *Peer-to-peer Networking and Applications*, (2015), pp.1-12.

[10] Lal SP & Joe Prathap PM, “Security Issues in Wireless Sensor Networks–An Overview”, *International Journal of Computer Science and Information Technology*, Vol.6, (2015), pp.920-924.

[11] Althobaiti O, Al-Rodhaan M & Al-Dhelaan A, “An efficient biometric authentication protocol for wireless sensor networks”, *International Journal of Distributed Sensor Networks*, Vol.9, No.5,(2013).

[12] Menon VG, Joe Prathap PM & Vijay A, “Eliminating Redundant Relaying of Data Packets for Efficient Opportunistic Routing in Dynamic Wireless Ad Hoc Networks”, *Asian Journal of Information Technology*, Vol.12, No.17, (2016).

[13] Das AK, “A Secure and Efficient Biometric-based User Authentication Scheme for Wireless Sensor Networks using Smartcard and fuzzy extractor”, *International journal of Communication Systems*, (2015).

[14] Lal SP & Viswakarma HR, “QoS Based Bandwidth Allocation for Networks”, *International Journal of Computer Science and Information Technology*, Vol.2, No.2, (2009), pp.111-119.

[15] Choi Y, Lee Y & Won D, “Security improvements on biometric based authentication schemes for WSN using fuzzy extraction”, *Int J. Distri. Sensor Netw.*, Vol.116, (2016).

[16] Rothenberg C & Macapuna C, “In-packet bloom filters: Design and networking applications”, *Computer Networking*, Vol.55, No.6, (2011), pp.1364-1378.

[17] Hussain S, Wang C, Sultana S & Bertino E, “Secure data provenance compression using arithmetic coding in wireless sensor networks”, *IEEE International, Performance computing and communications conference (IPCCC)*, (2014), pp.1-10.

[18] Alam SI & Fahmy S, “A practical approach for provenance transmission in wireless sensor networks”, *AdHoc Networks*, (2014), pp.28-45.

[19] Sultana S, Ghinita G, Bertino E & Shihab M, “A Light weight Secure Provenance Scheme for Wireless Sensor Networks”, *IEEE International Conference on Parallel and Distributed Systems*, (2012), pp.101-108.

[20] Wang C, Zhenh W & Bertino E, “Provenance for wireless sensor networks: A Survey”, *Open Access–Springerlink.com*, (2016).

[21] Z Yesembayeva (2018). Determination of the pedagogical conditions for forming the readiness of future primary school teachers, Opción, Año 33. 475-499

[22] G Mussabekova, S Chakanova, A Boranbayeva, A Utebayeva, K Kazybaeva, K Alshynbaev (2018). Structural conceptual model of forming readiness for innovative activity of future teachers in general education school. Opción, Año 33. 217-240