International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 05 | May-2016 www.irjet.net p-ISSN: 2395-0072

DETECTING DERIVATION FAKE AND PACKET DROP ATTACKS IN WIRELESS SENSOR NETWORKS USING RSA

[*1] Ms. Dhanalakshmi M., [*2] Ms. Anitha Auxilia S.,

[*1]M.Phil Research Scholar, PG & Research Department of Computer Science & Information Technology Arcot Sri Mahalakshmi Women's College, Vellore, Tamil Nadu, India.

[*2]Assistant Professor, PG & Research Department of Computer Science & Information Technology Arcot Sri Mahalakshmi Women's College, Vellore, Tamil Nadu, India.

Abstract: - *Large-scale sensor networks are deployed in numerous application domains, and the data they collect are used in decision-making for critical infrastructures. Data are streamed from multiple sources through intermediate processing nodes that aggregate information. A malicious adversary may introduce additional nodes in the network or compromise existing ones. Therefore, assuring high data trustworthiness is crucial for correct decision-making. Data provenance represents a key factor in evaluating the trustworthiness of sensor data. Provenance management for sensor networks introduces several challenging requirements, such as low energy and bandwidth consumption, efficient storage and secure transmission. In this paper, we propose a novel lightweight scheme to securely transmit provenance for sensor data. The proposed technique relies on in packet Bloom filters to encode provenance. We introduce efficient mechanisms for provenance verification and reconstruction at the base station. In addition, we extend the secure provenance scheme with functionality to detect packet drop attacks staged by malicious data forwarding nodes. We evaluate the proposed technique both analytically and empirically, and the results prove the effectiveness and efficiency of the lightweight secure provenance scheme in detecting packet fake and loss attacks.*

Key Words: *Bloom Filter, Wireless Sensor Network, detection packet drop attacks, Sensor Data, malicious attacks.*

I. INTRODUCTION

In a wireless sensor network, data are produced at a large number of sensor node sources and processed in network at intermediate hops network on their way to a Base Station that performs decision [1] - making. The diversity of data sources create the need to assure the trustworthiness of data such as only trustworthy information is considered in the decision process. Sensor nodes monitor the environment, detect events of interest, produce data and collaborate in forwarding the data towards a sink, which could be a gateway, base station, storage node, or querying user[2]. A sensor network is often deployed in an unattended and hostile environment to perform the monitoring and data collection tasks. When it is deployed in such an environment, it lacks physical protection and is subject to node compromise. After compromising one or multiple sensor nodes, an adversary may launch various attacks to disrupt the in - network communication. In a multi - hops sensor network and data provenance allows the BS to trace the source and forwarding path of an individual data packet [2]. Provenance must be recorded for each packet, but important challenges arise due to the tight storage[6], energy and bandwidth constraint of sensor nodes. Therefore, it is necessary to devise a light - weight provenance solution with low overhead. Hence it's necessary to address security requirements like confidentiality, integrity and freshness of provenance. Our important goal is to design a provenance encoding and decoding method that satisfies security and performance need [4][5]. To deal with packet droppers, a broadly adopted countermeasure is multi - path forwarding in which each packet is forwarded along multiple redundant paths and hence packet dropping in some but not all of these paths can be tolerated [7] . This scheme introduces high extra communication overhead

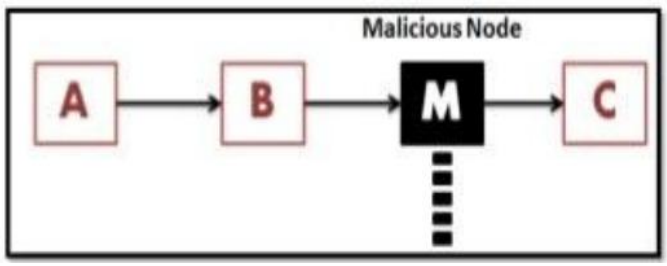


Fig1.1 : Packet Drop Attacks

Wireless sensor network has several limitations each node has limited battery, limited bandwidth to communicate, limited processing power and limited memory. Existing system to detect the provenance forgery attack considered such limitations of the WSN therefore it is efficient system for this task. Several WSN routing protocols are simple and are vulnerable to attacks from those works on routing in ad hoc networks.

Most threats against WSNs fall into one of the following groups: (i) Spoofed, altered, or replayed routing information, (ii) Selective forwarding, (iii) Sinkhole attacks, (iv) Sybil attacks, (v) Wormholes, (vi) HELLO flood attacks, (vii) Acknowledgment spoofing[3][6].

This is a distributed mechanism in order to encode provenance at the nodes and it will work as centralized algorithm to decode it at the BS. The technical core of this survey is the notion of (iBF). In this packet consists of a unique sequence number, data value, and an iBF which contains the provenance. The focus of this scheme is a securely transmitting provenance with the data to the BS. In this aggregation framework, securing the data values is an important factor,[4] The secure provenance technique can be used to obtain a complete solution that provides security for data, provenance and data-provenance binding. The three Security Objectives in sensor networks is a confidentiality, Integrity and freshness.

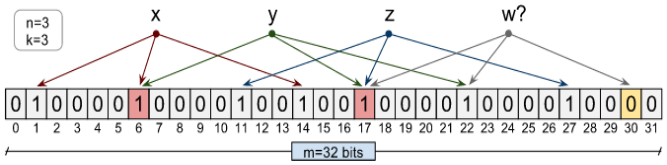


Fig 1.2: Overview of the Bloom filters probabilistic data structure.

We can be sure that the element was not inserted (no false negative property). If all the k bits are set to 1, we have a probabilistic argument to believe that the element was actually inserted. The case of collisions to bits set by other elements causing a non-inserted element to return 'true' is referred to as a false positive. In the example of Fig. 1, a false positive for w would be returned if all three hashes would map to 1s[7]. For the sake of generality, we refer simply to elements as the objects carried in the iBF. Depending on the application, elements may take different forms such as interface names, IP addresses, certificates, and so on. False positives manifest themselves with different harmful effects such as bandwidth waste, security risks, computational overhead, etc. Thus, a system design goal is keeping false positives to a minimum.

1.1 Detecting Packet Drop Attacks

Provenance encoding could be used for a packet acknowledgement. By using this sensor can transmit more meta-data. For an any individual data packet, the provenance record generated by a node will now consist of the node ID and an acknowledgement in the form of a sequence number of the lastly seen (processed/forwarded) packet belonging to that data flow. If the intermediate packet could be drop by the attacker means some nodes on the path do not receive that packet. Hence, during the next round of packet transmission the mismatch between the acknowledgements should be generated from different nodes on the path[5]. This factor could be to detect the packet drop attack and to localize the malicious node.

II. Related work

2.1 LEGITIMATE PACKET DROPPING

Packet dropping can be experienced in wireless ad hoc networks where no compromised nodes are present. This packet loss is mainly associated with the following events;

I. Network Congestion

Network congestion in wireless ad hoc networks is something unavoidable. These networks are mainly scalable due to in and out movements of nodes. As a result, congestion is more likely to happen which can lead to loss of packets.

II. Channel Conditions

In wireless networking the channel condition cannot be neglected since it changes drastically. Free path loss, interference, presence of noise on the channel and fading of the transmitted wireless signals are among the channel [5] conditions that can lead to packet loss or bit errors in the transmitted signal. In the presence of these factors, some packets can get dropped.

III. Resource Constraints

Nodes in wireless ad hoc networks have limited energy resource. Intermediate nodes in these networks may behave selfishly and fail to forward the received packets in order to conserve their limited resources battery power. These packets in turn get dropped[9].

2.2 MALICIOUS PACKET DROPPING

Mostly, the first step in launching a packet dropping attack is for a malicious node to get involved during route

formation. This is better done by exploiting the vulnerabilities of the underlying well known routing protocols used in wireless ad hoc networks which are designed basing on the assumption of trustworthiness between nodes in a network[1][5]. Once in the route, the malicious node can do anything including maliciously dropping packets.

This Packet dropping at a malicious intermediate node can lead to suspension of communication or generation of wrong information between the source and destination which is an undesirable situation. For better understanding, following are the illustrations of malicious packet dropping scenarios in wireless ad hoc networks under the two commonly used routing protocols AODV (Ad hoc On Demand Distance Vector) and OLSR (Optimized Link State Routing).

2.3 DETECTING PACKET DROP ATTACKS MECHANISM

In a wireless ad hoc network, nodes communicate with each other via wireless links either directly or relying on other nodes as routers. The nodes in the network not only act as hosts but also as routers that route data to/from other nodes in network. An adversary may misbehave by agreeing to forward packets and then failing to do so. Once being included in a route, the adversary starts dropping packets. That means it stop forwarding the packet to the next node[7]. The malicious node can exploit its knowledge about the protocol to perform an insider attack. It can analyze the importance of the transmitting packet and can selectively drop those packets. Thus it can completely control the performance of the network. If the attacker continuously dropping packets, it can be detect and mitigate easily. Because even if the malicious node is unknown, one can use the randomized multi - path routing algorithms to circumvent the black holes generated by the attack. If the malicious nodes get identified, the node can be deleted from the routing table of network[4]. The detection of selective packet dropping is highly difficult. Sometimes the dropping of packets may not be intentional. It can be occurred as a result of channel errors. So the detection mechanism should be capable of differentiating the malicious packet dropping and the dropping due to link errors. The algorithm introduced here provides an efficient mechanism to detect the selective packet dropping[1],[2]. It improves the detection accuracy by calculating the correlation between lost packets with the help of Auto Correlation Function of the bitmaps at each node in the route

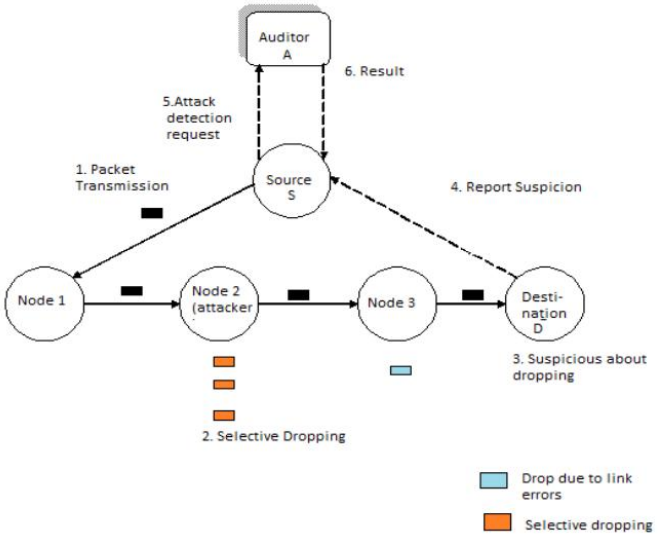


Fig 1.3 System Model

1. Data Packet Representation

To enable packet loss detection, a packet header must securely propagate the packet sequence number generated by the data source in the previous round. In addition, as in the basic scheme, the packet must be marked with a unique sequence number to facilitate per-packet provenance generation and verification. Thus, in the extended provenance scheme, any jth data packet contains (i) the unique packet sequence number (seq[j]), (ii) the previous packet sequence number (pSeq), (iii) a data value, and (iv) provenance.

2. Provenance Encoding

Depicts the extended provenance encoding process. The provenance record of a node includes (i) the nodeID, and (ii) an acknowledgement of the lastly observed packet in the flow. The acknowledgement can be generated in various ways to serve this purpose.

3. Provenance Decoding at the BS

Not only the intermediate nodes, but also the BS stores and updates the latest packet sequence number for each dataflow. Upon receiving a packet, the BS retrieves the preceding packet sequence (pSeq) transmitted by the source Node from the packet header, fetches the last packet sequence for the flow from its local storage (pSeqb), and utilizes these two sequences in the process of provenance verification and collection.

III. PREVIOUS IMPLEMENTATIONS

We study a multi-hop wireless sensor network, containing of a number of sensor nodes and a base station that gathers data as of the network. The network is modelled as a graph G (N, L), where N = {ni|, 1 ≤ i ≤ |N|} is the set of nodes, and L is the fixed of links, containing an element li,j for each pair of nodes ni and nj that are communicating directly with each other. Each node reports its neighboring node information to the BS after deployment. The BS assigns each node a single identifier nodeID and a symmetric cryptographic key Ki. In addition, a set of hash functions H = {h1, h2, ...,hk} are broadcast to the nodes aimed at use during provenance embedding.

3.1 DATA MODEL

We adopt a multiple-round process of data collection. Each sensor makes data periodically, and individual values are combined near the BS using any existing hierarchical like tree-based dissemination scheme. A data path of D hops is represented as < nl, n1, n2, ..., nD >, where nlis a leaf node representing the data source, and node ni is i hops away from nl.

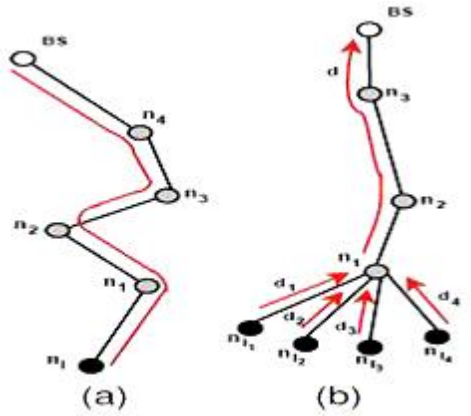


Figure 1.4 : Provenace Graph

Each non-leaf node in the path collections the received data and provenance with its own locally-generated data and provenance. The data packet contains (i) a exclusive packet sequence number, (ii) a data charge, and (iii) provenance. The sequence number is involved to the packet by the facts source, and all nodes use the same sequence number for a given round.

Definition for Provenance: Given a data packet d, the provenance pd is a directed acyclic graph G(V,E) satisfying the following properties: pd is a sub graph of the sensor network G(N,L); for vi, vj ∈V , vi is a child of vj if and only if HOST (vi) = ni participated in the distributed calculation of d and/or forwarded the data to HOST (vj) =nj ;for a set U = {vi} ⊂V and vj ∈V ,U is a set of children of vj if and only if HOST (vj) collects processed/forwarded data from each HOST(vi ∈U) to generate the aggregated result.

IV. SYSTEM IMPLEMETNATION

In proposed system using key exchanging, cryptography, and signature technique are used. So easily detect the suspicious data. In verify module detect the suspicious data and provenance data. Receiving packet data suspicious data means placed in suspicious box. Suppose data will be provenance data means placed in provenance box.

Cryptography

Cryptography is the practice and study of techniques for secure communication in the presence of third parties more generally, it is about constructing and analyzing protocols that overcome the influence of adversaries and which are related to various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.

4.1 KEY GENERATION ALGORITHM

RSA involves a public key and a private key. The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted using the private key. The keys for the RSA algorithm are generated the following way:

1.Choose two distinct prime numbers p and q.

For security purposes, the integer's *p* and *q* should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.

2. Compute $n = pq$.

n is used as the modulus for both the public and private keys

3. Compute $\varphi(n) = (p - 1)(q - 1)$, where φ is Euler's totient function.

4. Choose an integer e such that $1 < e < \varphi(n)$ and greatest common divisor of $(e, \varphi(n)) = 1$; *i.e., e and* $\varphi(n)$ are co prime

e is released as the public key exponent.

e having a short bit-length and small Hamming weight results in more efficient encryption - most commonly 0x10001 = 65,537. However, small values of e (such as 3) have been shown to be less secure in some settings.[4]

5. Determine d as:

i.e., d is the multiplicative inverse of e mod $\varphi(n)$.

- This is more clearly stated as solve for d given $(de) = 1 \bmod \varphi(n)$
- This is often computed using the extended Euclidean algorithm.
- *d* is kept as the private key exponent.

By construction, $d*e = 1 \bmod \varphi(n)$. The public key consists of the modulus n and the public (or encryption) exponent e. The private key consists of the modulus n and the private (or decryption) exponent d which must be kept secret. *(p, q, and* $\varphi(n)$ must also be kept secret because they can be used to calculate d.)

- An alternative, used by PKCS#1, is to choose d matching $de \equiv 1 \bmod \lambda$ with $\lambda = \text{lcm}(p-1, q-1)$, where lcm is the least common multiple. Using λ instead of φ(n) allows more choices for d. λ can also be defined using the Carmichael function, λ(n).
- The ANSI X9.31 standard prescribes, IEEE 1363 describes, and PKCS#1 allows, that p and q match additional requirements: be strong primes, and be different enough that Fermat factorization fails.

4.1.1 Encryption

Alice transmits her public key to Bob and keeps the private key secret. Bob then wishes to send message M to Alice. He first turns M into an integer m, such that by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext corresponding to This can be done quickly using the method of exponentiation by squaring. Bob then transmits to Alice.

Note that at least nine values of m could yield a ciphertext c equal to m,[5] but this is very unlikely to occur in practice.

4.1.2 Decryption

Alice can recover from by using her private key exponent via computing. Given the can recover the original message M by reversing the padding scheme.

4.1.3 Sign Module

In sign module following process are preformed. 1. Key generation, 2.encryption, 3.key exchanging 4.signature 5.send to verify module

4.1.4 Provenance Verification

In verify modules following process are preformed. 1. Key generation, 2.decryption, 3.key exchanging 4.send to receiver module

4.1.5 Provenance Collection:

In receiver module receive a packet data suspicious means place in suspicious box suppose data correct data means placed in province box.

4.1.6 Data-provenance

Setup: the data producer sets up its signing key k and data consumer sets up its verification key k_0 in a secure fashion that prevents malware from accessing the secret keys. Sign *(D, k)*: the data producer signs its data D with a secret key k, and outputs D along with its proof sig. Verify *(sig, D,* k_0*):* the data consumer uses key k_0 to verify the signature sig of received data D to ensure its origin, and rejects the data if the verification fails.

N_i*:* the number of members in a level-i cluster,
L_i *:* the sum of distances between the members of a level-I cluster and their level-i CH,
H_i : the number of hops from a member to its CH in a typical level-i cluster,
CH_i : the total number of level-i CHs,
The Energy and Clustering based on the Location and based on distance information.

$$E[D_i \mid N = n] = \int_A \sqrt{x_i^2 + y_i^2}\left(\frac{1}{4a^2}\right) dA = 0.765a\,.$$

Since there are on an average np CHs and the location of any CH is independent of the locations of other CHs, the total length of the segments from all these CHs to the processing center is $0.765npa$.

$$E[C_1 \mid N=n] = \frac{E[L_v \mid N=n]}{r}.$$

Define *C*2 to be the total energy spent by all the sensors communicating 1 unit of data to their respective cluster heads. Because, there are *np* cells, the expected value of *C*2 conditioned on *N*, is given by

$$E[C2 \mid N=n] = npE[C1 \mid N=n].$$

The sum of distance of level-(i-1) CHs from a level-i CH, i = 2,3,..., h in a typical level-i cluster or the sum of distance of sensors from a level-1 CH is given by

$$E[L_i \mid N=n] = \frac{(1-p_i)\lambda \prod_{j=1}^{i-1} p_j}{2\left(\lambda \prod_{j=1}^{i} p_j\right)^{3/2}}.$$

E[C] is minimized by a value of p that is a solution of

$$Cp^{3/2} - p - 1 = 0$$

The above equation has three roots, two of which are imaginary. The second derivative of the above unction ispositive for the only real root of (9) and hence it minimizes th eenergy spent.

Define:

r_n *to be the ID of the root selected by node n*
d_n *to be the shortest-path distance from* r_n *to node n*
$g_n = (_n, r_n, d_n)$ *to be the message sent by node n*
pn *to be the ID of the parent selected by node n*
*trecv,*n *to be the time node n received the message from its parent*

Initialize: g_n *to* $(n, n, 0)$ $\forall$ $n \in N$
p_n *to* n $\forall$ $n \in N$
$t_{recv,n}$ *to* 0 $\forall$ $n \in N$

Get Span (node ID n, time t, timeframe T)

1. *if n is not an event source,*
2. *return*
3. *else {single-hop broadcast gn and start a timer P*
   that expires every T_{sec}
4. *while true,*
5. *if timer P expires and (rn = n or t >* $t_{recv,n+T}$*),*
6. *set* g_n *to* $(n, n, 0)$
7. *set* p_n *to* n
8. *set* $t_{recv,n}$ *to* t
9. *single-hop broadcast gn*
10. *if receiving a message gi from node i,*
11. *if* $r_i <$ rn, *or* $(ri = rn$ *and* $di+1 < d_n)$, *or* $(r_i = r_n, d_{i+1} = d_n$, *and* $i \_ p_n)$,
12. *set gn to* $(n, r^i, di+1)$
13. *set* p_n *to* i
14. *set* $t_{recv,n}$ *to* t
15. *single-hop broadcast gn and restart timer P}*

EVALUATION RESULT

In this section, three experiments were designed to evaluate the performance of the improved clustering algorithm. The simulating program was developed by our team using NS2. In the following experiments, Most of studies only consider that wireless sensor networks are equipped with only Omni-directional antennas, which can cause high collisions. It is shown that the per node throughput in such networks is decreased with the increased number of nodes. Thus, the transmission with multiple short - range hops is preferred to reduce the interference. However, other studies show that the transmission delay increases with the increased number of hops. Found that using directional antennas not only can increase the throughput capacity but also can decrease the delay by reducing the number of hops.

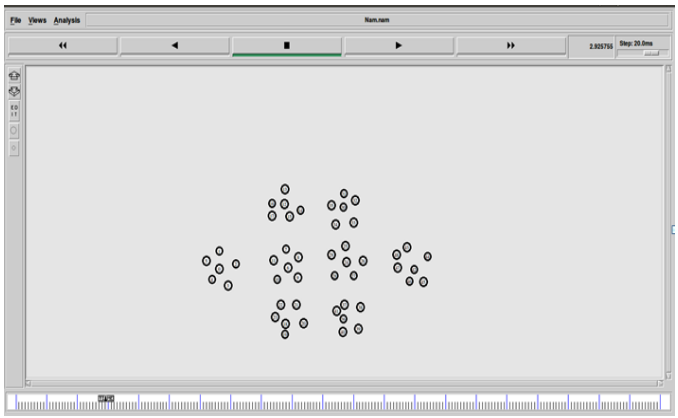
Cygwin is free software that provides a Unix-like environment and software tool set to users of any modern x86 32-bit and 64-bit versions of MS-Windows (XP with SP3/Server 20xx/Vista/7/8) and (using older versions of Cygwin) some obsolete versions (NT/2000/XP without SP3) as well. Cygwin consists of a Unix system call emulation library, cygwin1.dll. With Cygwin installed, users have access to many standard UNIX utilities. They can be used from one of the provided shells such as bash or from the Windows Command Prompt.

Contains both merits and limitations when people use it to simulate WSNs. To the merits, firstly as a non-specific network simulator, NS-2 can support a considerable range of protocols in all layers. For example, the ad-hoc and WSN specific protocols are provided by NS-2. Secondly, the open source model saves the cost of simulation, and online documents allow the users easily to modify and improve the codes. However, this simulator has some limitations. Firstly, people who want to use this simulator need to familiar with writing scripting language and modeling technique; the Tool Command Language is somewhat difficult to understand and write. Secondly, sometimes using NS-2 is more complex and time-consuming than other simulators to model a desired job. Thirdly, NS-2 provides a poor graphical support, no Graphical User Interface (GUI) the users have to directly face to text commands of the electronic devices. Fourthly, due to the continuing changing the code base, the result may not be consistent, or contains bugs.

Parameter used in the Simulation

Parameter	Value
Simulation time	300
Number of nodes	46
Traffic model	CBR
Node Placement	Uniform
Performance parameter	Energy consumption, End to End delay
Routing protocol	AODV
No Coding	TCL
Node ID	1
Signal Strength	70%
Mobility	Clustering Head
Connection	

Table 1.2: Simulation Parameters

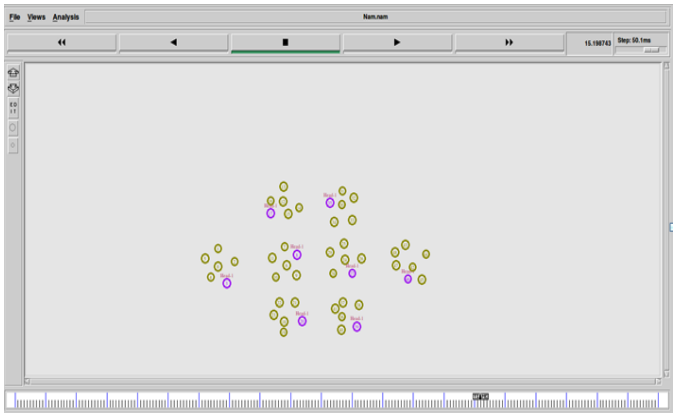


Red- Clustering; black- cluster Head; black – No of Node; green- Energy

Fig 1.4: Proposed Node

For the experimental results presented in this section, a network of sensors randomly distributed in a 1000m * 1000m field is considered. The number of sensors in the network, i.e. the network size, is kept at 100, 200, 300 and so on. Each sensor has an initial energy of 1.Joule and the base station is located at (250, 330). Each sensor generates packets of size 1000 bits. The energy model for the sensors is based on the first order radio model.

Number of Sensors (n)	Density (d)	Probability (p_{opt})	Maximum Number of Hops (k)
500	5	0.1012	5
1000	10	0.0792	4
1500	15	0.0688	3
2000	20	0.0622	3
2500	25	0.0576	3
3000	30	0.0541	3



Red- Clustering; Blue - cluster Head; black – No of Node; Green- Energy

Fig 1.5: Clustering Head

The manner in which tree leaders are selected in each level of the hierarchy is examined. Obviously an aggregation tree, for every round of data gathering is defined. For comparison, the MST tree to perform data gathering, with and without aggregation is implemented. In the case of no aggregation, sensors use the similar chain-based hierarchy to transmit their packets to the base station.

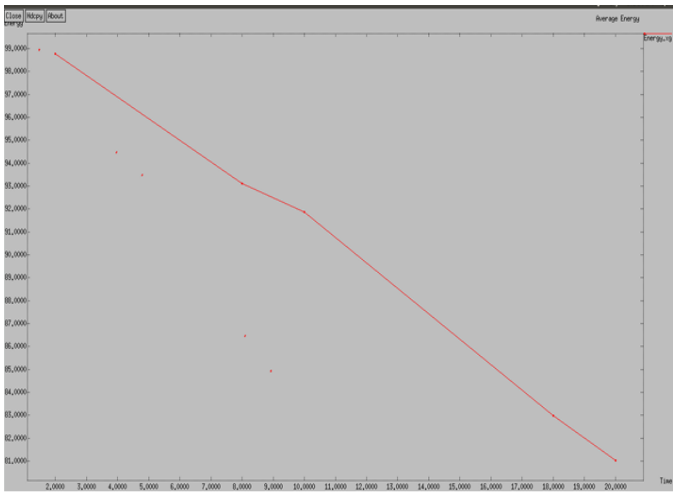


Fig 1.6: Average Energy

To evaluate the performance of the proposed EECH in terms of the delay for accessing the nodes, the network is simulated for various range of mobility. When the mobility of the nodes in the network increases in m/s, the data redundancy has also been removed and the time taken to process the data is decreased.

CONCLUSION

We addressed the problem of securely transmitting provenance for sensor networks, and proposed a light-weight provenance encoding and decoding scheme based on Bloom filters. The scheme ensures confidentiality, integrity and freshness of provenance. We extended the scheme to incorporate data-provenance binding, and to include packet sequence information that supports detection of packet loss attacks. Experimental and analytical evaluation results show that the proposed scheme is effective, light-weight and scalable. In future work, we plan to implement a real system prototype of our secure provenance scheme, and to improve the accuracy of packet loss detection, especially in the case of multiple consecutive malicious sensor nodes.

REFERENCES:

[1] H. Lim, Y. Moon, and E. Bertino, *"Provenance-Based Trustworthiness Assessment in Sensor Networks,"* Proc. Seventh Int'l Workshop Data Management for Sensor Networks, pp. 2-7, 2010.

[2] I. Foster, J. Vockler, M. Wilde, and Y. Zhao, *"Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation,"* Proc. Conf. Scientific and Statistical Database Management, pp. 37-46, 2002.

[3] K. Muniswamy-Reddy, D. Holland, U. Braun, and M. Seltzer, *"Provenance-Aware Storage systems,"* Proc. USENIX Ann. Technical Conf., pp. 4-4, 2006.

[4] Y. Simmhan, B. Plale, and D. Gannon, *"A Survey of Data Provenance in E-Science,"* ACMSIGMODRecord, vol. 34, pp. 31-36, 2005.

[5] R. Hasan, R. Sion, and M. Winslett, *"The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance,"* Proc. Seventh Conf. File and Storage Technologies (FAST), pp. 1-14, 2009.

[6] S. Madden, J. Franklin, J. Hellerstein, and W. Hong, *"TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks,"* ACM SIGOPS Operating Systems Rev., vol. 36, no. SI, pp. 131-146, Dec. 2002.

[7] K. Dasgupta, K. Kalpakis, and P. Namjoshi, *"An Efficient Clustering Based Heuristic for Data Gathering and Aggregation in Sensor Networks,"* Proc. Wireless Comm. and Networking Conf., pp. 1948- 1953, 2003.

[8] S. Sultana, E. Bertino, and M. Shehab, *"A Provenance Based Mechanism to Identify Malicious Packet Dropping Adversaries in Sensor Networks,"* Proc. Int'l Conf. Distributed Computing Systems (ICDCS) Workshops, pp. 332-338, 2011.

[9] L. Fan, P. Cao, J. Almeida, and A.Z. Broder, *"Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol,"* IEEE/ACM Trans. Networking, vol. 8, no. 3, pp. 281-293, June 2000.

[10] A. Kirsch andM. Mitzenmacher, *"Distance-Sensitive BloomFilters,"* Proc.Workshop Algorithm Eng. and Experiments, pp. 41-50, 2006.

[11] C. Rothenberg, C.Macapuna,M.Magalhaes, F. Verdi, and A. Wiesmaier, *"In-Packet Bloom Filters: Design and Networking Applications,"* Computer Networks, vol. 55, no. 6, pp. 1364-1378, 2011.

[12] M. Garofalakis, J. Hellerstein, and P. Maniatis, *"Proof Sketches: Verifiable In-Netwok Aggregation,"* Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), pp. 84-89, 2007.