



Energy Aware Load Balancing With False Temporal Pattern Identification for Wireless Sensor Networks

Meenalochini.M and T.Abirami

Department of Information Technology, Kongu Engineering College, Perundurai, India.

ARTICLE INFO

Article history:

Received: 21 January 2015;

Received in revised form:

19 June 2015;

Accepted: 29 June 2015;

Keywords

Wireless sensor networks,
Load balancing,
Backup nodes,
Continuous data aggregation,
Sampling.

ABSTRACT

Load balancing using clustering can increase network scalability. This paper attempting to apply efficient techniques of load, energy and security such that network life can be increased with security and also its reliability. Continuous aggregation is required in sensor applications to obtain the temporal variation information of aggregates. It helps the users to understand how the environment changes over time and track real time measurements for trend analysis. Due to the high frequency and the long period of a continuous aggregation in every epoch, the false temporal variation pattern would incur a great communication cost. In this paper, we propose a clustering technique which will balance the load among the cluster by using some backup nodes. In the continuous aggregation, the attacker could manipulate a series of aggregation results through compromised nodes to fabricate false temporal variation patterns of the aggregates. We detect and verify a false temporal variations pattern by checking only a small part of aggregation results to reduce a verification cost. A sampling based approach is used to check the aggregation results and we also proposed a security mechanisms to protect the sampling process.

© 2015 Elixir All rights reserved.

Introduction

Wireless Sensor Networks (WSN) are widely used for environmental and security monitoring. Small wireless sensors have the capacity to sense, compute, store and transmit; it integrates with each other to form a network. In a WSN, the sensors monitor the immediate surroundings, and the data is transmitted to a well-equipped node called the Sink. Patterns in the data are analyzed offline, but this results in transmitting a large amount of data through the network leading to communication overhead. Protocols can reduce transmitted power in two ways. First where nodes can emit to short distances such as data sinks or cluster nodes. The cluster node can then send the data over a larger distance preserving the power of the smaller nodes. The second is by reducing the number of bits (amount of data) sent across the wireless network applications of wireless sensor networks (WSNs), the aggregations of sensed data, such as sum, average, and predicate count, are very important for the users to get summarization information about the monitored area. Instead of collecting all sensor data and computing aggregation results at the base station (BS), in-network aggregation allows sensor readings to be aggregated by intermediate nodes, which efficiently reduces the communication overhead. In this paper, we consider the security of continuous in-network aggregation in WSNs. In many WSN applications the users often need the temporal variation information in a series of aggregation results rather than an individual aggregation result. For a continuous aggregation query, a time interval, called epoch, is specified and the aggregation is evaluated in every epoch. The duration of every epoch specifies the amount of time sensor nodes wait before acquiring and transmitting each successive sample. Because of the importance of temporal variation information of aggregation results, we focus on the attack against continuous in-network aggregation that the adversaries attempt to distort the real

temporal variation pattern of the aggregate by disrupting a series of successive aggregation results.

Methods and Implementation

The proposed approach assume network with the sensor nodes having different energy levels and processing power. Some high computing nodes are deployed nearby each other. All the nodes with high initial energy level and processing power are selected. Some nodes from the set are selected as cluster head (CH) according to their location. Each CH defines its communication range in terms of power level to form cluster. Some nodes with comparable energy and processing power in the CH range are asked to go to sleep and information about those nodes is maintained with the CH. All the cluster members will send the sensed data to the CH. The CH will send the aggregated data to the Base Station directly or by using some intermediate CH.

Reconfiguration of Clusters

When the energy level of the CH will reach to the threshold value TL, the CH will activate one of the sleeping nodes and will make it CH. In the sleeping mode also the sensor node is dissipating the energy but with very less rate. All the cluster heads forms their clusters along with the backup nodes. This information about the new CH will be sent to all the cluster member and other CH also. The old CH will become the general sensor node. After completing this reconfiguration phase, some nodes, which are not the members of any of the cluster, try to find the nearest cluster member node. Again, the same criterion of finding the minimum distance cluster member is applied to find the appropriate cluster

Representative point selection

The system initially identifies the representative points and performs the aggregation. The system performs continuous secure aggregation on every node by collecting a physical quantity of humidity and temperature from the surrounding

environment and process the acquired data and transfer the collected data to a sink node or base station. Either in the on demand verification or in the periodic verification, the BS selects some points from the series of aggregation results in the time window to be verified, and checks their correctness to detect any the modification of temporal variation patterns. Considering that the attacker can manipulate only a small number of aggregation results such as extreme points to change with the temporal variation pattern, it may be ineffective to check a set of randomly selected points to detect forged patterns because the selected points may not cover these manipulated points, which causes that the attack is not detected.

To guarantee effective attack detection, the selected points should be able to capture the temporal variation pattern in the time window and these points as representative points and the epoch of a representative point as representative epoch.

Representative Point Selection (RPS) algorithm

We give the definition of representative point to formally characterize the requirement that is to capture the temporal pattern of the whole aggregation result series. Let $P = \{(e_i, Ag(e_i)) \mid 1 \leq i \leq p, e_1 = t < e_2 < \dots < e_{p-1} < e_p = t + l\}$ be a set of points in the time window $[t, t + l]$ where $Ag(e_i)$ is the aggregation result in epoch e_i . Let $F_p(x)$ be the piecewise linear function consisting of connected line segments, each of which is between point $(e_i, Ag(e_i))$ and $(e_{i+1}, Ag(e_{i+1}))$ for $1 \leq i \leq p - 1$.

The pseudocode of RPS algorithm is shown in Algorithm 1.

INPUT: $\langle \{(k, Agk) \mid t \leq k \leq t + l, p \rangle$ ($p \geq 2$)

OUTPUT: $\langle E[l, p], S \rangle$ // S is an array indexed from 1 and records $p - 2$

number of epochs of representative points except t and $t + l$

1: // Entry $E[i, j]$ records $E(t; t + i, j)$

2: // Entry $s[i, j]$ records the representative epoch preceding $t + i$

3: for $i \leftarrow 1$ to l do

4: for $j \leftarrow 2$ to p do

5: if $j = 2$ then

6: $E[i, j] \leftarrow I(t; t + i, j)$; $s[i, j] \leftarrow t$

7: else if $j \geq i + 1$ then

8: $E[i, j] \leftarrow 0$; $s[i, j] \leftarrow t + i - 1$

9: else

10: $min \leftarrow \infty$

11: for $k \leftarrow 2$ to $i - 1$ do

12: if $min > E[k, j - 1] + I(t + k, t + i)$ then

13: $min \leftarrow E[k, j - 1] + I(t + k, t + i)$

14: $s[i, j] \leftarrow t + k$

15: $E[i, j] \leftarrow min$

16: end $\leftarrow i$

17: for $i \leftarrow 1$ to p do

18: end $\leftarrow s[end; p - i + 1] - t$

19: if end > 0 then

20: $S[p - i - 1] \leftarrow end + t$

21: else

22: break

23: return $\langle E[l, p]; S \rangle$

Secure sampling

After the selection of representative points, the BS broadcasts a verification request, which includes the representative epochs, the sampling ratio and a nonce number nonce, to the WSN. Each sensor node has a unique identifier and shares a unique secret symmetric key with the base station. By pairwise key establishment schemes, each node shares a pairwise key with each of its direct neighbours and two-hop neighbours. A broadcast authentication protocol such as TESLA exists such

that any node can authenticate a message from the base station. The WSNs have a short safe bootstrapping phase right after network deployment, during which attacker cannot successfully compromise any nodes.

Verifiable Random Sampling

In the verifiable random sampling, each node decides whether it is sampled by computing a cryptographically secure pseudorandom function h . h uniformly maps the input values into the range of $[0, 1]$ where nonce are nonce numbers that are disseminated within each aggregation query and verification request, respectively. In this way, whether a node is sampled is decided by the node individual key and two nonce numbers which are known by the BS. Since each time of verification different nonce is used, the nodes are randomly selected to be sampled for every verification. Also, a malicious node cannot decide to be sampled, since the BS can verify the legitimacy of sampled node. The actual number of samples returned by this sampling approach is random. Thus, the determination of sampling ratio ρ needs to provide a probabilistic guarantee to achieve the target sample size of at least m_t .

Local sample authentication

In the local sample authentication, each sampled node, say v , broadcasts its sample R_v to its neighbours to obtain authentication from its neighbours before sending R_v to the BS. Each neighbour u verifies the validity of R_v . If the verification is successful, u sends to v the message authentication code of R_v computed by u 's key used for the authentication of v 's sample.

Local authentication key setup

To derive keys for the local sample authentication, each node u is loaded with a seedkey before deployment. The BS holds the seed keys of all nodes. During the safe bootstrapping phase, u discovers its one-hop neighbours and computes for each neighbour v . Since each authentication keys is bound with a neighbour pair, the attacker cannot use it to authenticate samples from malicious nodes except for the corresponding neighbouring node. The erasure of seed keys prevents the attacker from deriving all the authentication keys..

Performance of Aggregation Verification

To evaluate our aggregation verification scheme, we simulate a WSN of 100 nodes and the sensing readings of each node are synthesized to the sensor readings of a random node. We consider continuous average and count aggregation that counts the number of nodes whose temperature readings are greater than 25 in the time window. We simulate two attacks against the continuous average aggregation and predicate count aggregation in the time window.

In average aggregation, the attacker aims to delay the true time when the temperature increases. In the attack against count aggregation, the attacker aims to fabricate a false fluctuation of count value. Figure1 and Figure2 show both the real aggregation results and the forged one. Real aggregation have a rapid increase pattern between (20,50) for both average and count

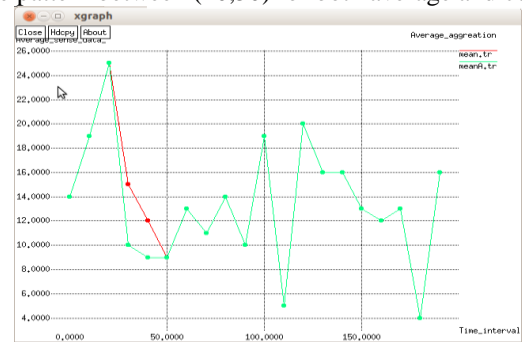


Figure 1. Continuous Average aggregation

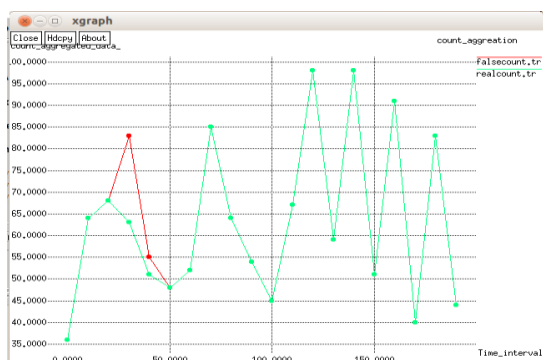


Figure 2. Continuous count aggregation under attack

Conclusion and Future Work

In this paper we provide a distinct design issue for a verification scheme to protect the authenticity of the temporal variation patterns in the aggregation results. Compared with the existing secure aggregation schemes, the proposed scheme need to check only a small portion of aggregation results in a time window. The representative point selection algorithm is proposed to detain the temporal variation pattern. By exploiting the spatial correlation among the sensor readings in close proximity, a series of security mechanisms are also proposed to protect the sampling procedure. Future work includes studying further opportunities for secure localization and addressing their threats.

References

[1] Lei S. and Li Y. (2014), 'Secure Continuous Aggregation in Wireless Sensor Networks', proceeding IEEE Transaction on Parallel and Distributed Systems, pp. 265-266.
 [2] Sajid Hussain D. and Abdul Matin W. (2013), 'Hierarchical Cluster-based Routing in Wireless Sensor Networks', proceeding on IEEE Transaction on Computing Systems (ICDCS), pp. 255-259.

[3] Ji S. and Cai Z. (2012), 'Distributed Data Collection and Its Capacity in Asynchronous Wireless Sensor Networks', IEEE INFOCOM, pp. 2113-2122.

[4] Ji S., Beyah R. and Cai Z. (2012), 'Snapshot/Continuous Data Collection Capacity for Large-Scale Probabilistic Wireless Sensor Networks', proceeding on IEEE INFOCOM, pp. 1035-1043.

[5] Cai Z., Ji S. and Bourgeois A G. (2012), 'Optimal Distributed Data Collection for Asynchronous Cognitive Radio Networks', IEEE Transaction on Distributed Computing Systems (ICDCS), pp. 245-254.

[6] Chandrakant N., Deepa Shenoy Venugopal K R (2011), 'Middleware services for security in scalable and non-scalable heterogeneous nodes of manets' International Journal of Future Generation Communication and Networking(IJFGCN), pp. 1-12

[7] Yang Y., Wang X., and Cao G. (2006), 'SDAP: A Secure Hop-By- Hop Data Aggregation Protocol for Sensor Networks', ACM Mobile Hoc, pp. 356-367.

[8] Chan H., Perrig A., and Song D. (2006), 'Secure Hierarchical In-Network Aggregation in Sensor Networks', ACM Conference Computer and Communication Security (CCS), pp. 278-287.

[9] Liu D. (2003), 'Establishing Pairwise Keys in Distributed Sensor Networks', ACM Conference on Computer and Communication Security (CCS), pp. 52-61.

[10] Przydatek B., and Perrig A. (2003), 'SIA: Secure Information Aggregation in Sensor Networks', ACM Conference on Embedded Networked Sensor Systems, pp. 255-265.

[11] Madden S. (2002), 'Tag: A Tiny Aggregation Service for Ad-Hoc Sensor Networks', Operating Systems Design and Implementation (OSDI), pp.4192-4203.