International Journal of Engineering Research-Online
A Peer Reviewed International Journal
Articles available online http://www.ijoer.in

Vol.3., Issue.2, 2015

RESEARCH ARTICLE

ISSN: 2321-7758

SECURE DATA AGGREGATION IN WIRELESS SENSOR NETWORKS WITH CLOCK-SKEW

V.SOUNDHARYA, K.VIJAYALAKSHMI

[1]Dhanalakshmi Srinivasan Engineering College, Perambalur

[2]Assistant professor, Dhanalakshmi Srinivasan Engineering College, Perambalur

Article Received: 30/03/2015 | Article Revised on:10/04/2015 | Article Accepted on:14/04/2015

ABSTRACT

A wireless sensor network mostly relies on multi-hop transmissions to deliver a data packet along a sequence of nodes to Base Station. It is of essential importance to measure the forwarding quality of multi-hop paths and such information shall be utilized in designing efficient routing strategies. In a large WSN, in-network data aggregation (i.e., combining partial results at intermediate nodes during message routing) significantly reduces the amount of communication overhead and energy consumption. The research community proposed a loss-resilient aggregation framework called synopsis diffusion, which uses duplicate insensitive algorithms on top of multipath routing schemes to accurately compute aggregates (e.g., predicate count or sum). However, this aggregation framework does not address the problem of false sub-aggregate values contributed by compromised nodes. This attack may cause large errors in the aggregate computed at the base station, which is the root node in the aggregation hierarchy. The attack-resilient computation algorithm is used to enable the base station to securely compute predicate count or sum even in the presence of such an attack. Our algorithm computes the true aggregate by filtering out the contributions of compromised nodes in the aggregation. In Future to select efficient multi path routing in WSN using Ant Colony Optimization algorithm and also implement for Clock Skew between the Sensor node communication.

Key Words— Data aggregation, hierarchical aggregation, in-network aggregation, sensor network security, synopsis diffusion, attack resilient.

I. INTRODUCTION

With advance in technology, sensor networks composed of small and cost effective sensing devices equipped with wireless radio transceiver for environment monitoring have to monitor the environment[1]-[2].These sensor nodes can monitor the environment by collecting information from their surroundings, and work cooperatively to send the data to a base station, or sink, for analysis.

The main goal of data aggregation algorithms is to gather and aggregate data in an energy efficient manner so that network lifetime is enhanced. Wireless sensor networks (WSN) offer an increasingly attractive method of data gathering in distributed system architectures and dynamic access via wireless connectivity.

In the TAG system [13], users connect to the sensor network using a workstation or base station directly connected to a sensor designated as the sink. Aggregate queries over the sensor data are formulated using a simple SQL-like language, then distributed across the network.

Aggregate results are sent back to the workstation over a spanning tree, with each sensor combining its own data with results received from its children. If there are no failures, this in-network aggregation technique is both effective and energy-efficient for distributive and algebraic aggregates [4] such as MIN, MAX, COUNT and AVG. However, this technique is much less effective in sensor network scenarios with moderate node and link failure rates.

However, aggregating along a tree is very susceptible to node and transmission failures, which are common in sensor networks. [11]-[13]. Because each of these failures loses an entire sub tree of readings, a large fraction of the readings are typically unaccounted for in a spanning tree based system.

Synopsis diffusion, a general framework for combining multi-path routing schemes with clever algorithms used to avoid double-counting. By decoupling aggregation from message routing and allows the level of redundancy in message routing (as a trade-off with energy consumption) to be adapted to sensor network conditions. As a result, highly accurate and

reliable answers can be obtained using roughly the same energy consumption as with tree-based schemes.

Unfortunately, none of the above algorithms or systems include any provisions for security; as a result, they are vulnerable to many attacks that can be launched by unauthorized or compromised nodes. A compromised node might attempt to thwart the aggregation process by launching several attacks, such as eavesdropping, jamming, message dropping, message fabrication, and so on. This paper focuses on a subclass of these attacks in which the adversary aims to cause the BS to derive an incorrect aggregate.

By relaying a false sub-aggregate to the parent node, a compromised node may contribute a large amount of error to the aggregate. As an example, during the Sum computation algorithm [7], [8], a compromised node *X* can inject an arbitrary amount of error in the final estimate of Sum by falsifying *X*'s own sub-aggregate. We refer to this attack as the *falsified sub-aggregate attack*.

In this paper, we design an algorithm to securely compute aggregates, such as Count and Sum despite the falsified sub aggregate attack. In particular, our algorithm which we call the *attack-resilient computation algorithm* consists of two phases. The main idea is as follows: (i) In the first phase, the BS derives a preliminary estimate of the aggregate based on minimal authentication information received from the nodes. (ii) In the second phase, the BS demands more authentication information from only a subset of nodes while this subset is determined by the estimate of the first phase. At the end of the second phase, the BS can (locally) filter out the false contributions of the compromised nodes from the aggregate.

The key observation which we exploit to minimize the communication overhead is that to verify the correctness of the final synopsis (representing the aggregate of the whole network) the BS does not need to receive authentication messages from all of the nodes.

II. AGGREGATION PROTOCOLS

TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks(2002)

Tiny AGgregation (TAG) service for aggregation in low-power, distributed, wireless environments. TAG allows users to express simple, declarative queries and have them distributed and executed efficiently in networks of low-power, wireless sensors. Tiny AGgregation (TAG), a generic aggregation service for ad hoc networks of TinyOS motes. There are two essential attributes of this service. First, it provides a simple, declarative interface for data collection and aggregation, inspired by selection and aggregation facilities in database query languages. Second, it intelligently distributes and executes aggregation queries in the sensor network in a time and power-efficient manner, and is sensitive to the resource constraints and lossy communication properties of wireless sensor networks. TAG processes aggregates in the network by computing over the data as it flows through the sensors, discarding irrelevant data and combining relevant readings into more compact records when possible.

In general, queries in TAG have the form:

```
SELECT {agg(expr),attrs} from sensors
WHERE {selPreds}
GROUP BY {attrs}
HAVING {havingPreds}
EPOCH DURATION i
```

TAG consists of two phases: a *distribution* phase, in which aggregate queries are pushed down into the network, and a *collection* phase, where the aggregate values are continually routed up from children to parents. Recall that our query semantics partition time into epochs of duration, and that we must produce a single aggregate value (when not grouping) that combines the readings of all devices in the network during that epoch.

Proposed system have two techniques to improve performance and accuracy of our sensor network

i) Taking Advantage of A Shared Channel

This system have largely ignored the fact that motes communicate over a shared radio channel. The fact that every message is effectively broadcast to all other nodes within range enables a number of optimizations that can significantly reduce the number of messages transmitted and increase the accuracy of aggregates in the face of transmission failures.

ii) Hypothesis Testing

For certain classes of aggregates, if a node is presented with a guess as to the proper value of an aggregate, it can decide locally whether contributing its reading and the readings of its children will affect the value of the aggregate.

Demerits:

However, aggregating along a tree is very susceptible to node and transmission failures, which are common in sensor networks. Because each of these failures loses an entire sub tree of readings, a large fraction of the readings are typically unaccounted for in a spanning tree based system. This introduces significant error in the query answer [6]. Efforts to reduce losses by retransmitting packets waste significant energy and de-lay query responses.

Synopsis Diffusion for Robust Aggregation in Sensor Networks (2004)

Synopsis diffusion, a general framework for achieving significantly more accurate and reliable answers by combining energy-efficient multi-path routing schemes with techniques that is used to avoid double-counting. Synopsis diffusion avoids double-counting through the use of order- and duplicate-insensitive (ODI) synopses that compactly summarize intermediate results during in-network aggregation. Synopsis diffusion achieves its decoupling of aggregation and routing through the use of order- and duplicate-insensitive (ODI) synopses.

Synopsis diffusion performs in-network aggregation. The partial result at a node is represented as a synopsis [3, 10], a small digest (e.g., histogram, bit-vectors, sample, etc.) of the data.

The aggregate computation is defined by three functions on the synopses: i)Synopsis Generation: A synopsis generation function SG(_) takes a sensor reading (including its meta-data) and generates a synopsis representing that data. _ ii)Synopsis Fusion: A synopsis fusion function SF(_; _) takes two synopses and generates a new synopsis. iii)Synopsis Evaluation: A synopsis evaluation function SE(_) translates a synopsis into the final answer.

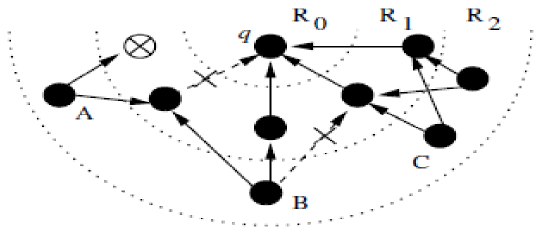

Figure 2.1: Synopsis diffusion over the Rings topology. Crossed arrows and circles represent failed links and nodes.

A synopsis diffusion algorithm consists of two phases: a distribution phase in which the aggregate query is flooded through the network and an aggregation topology is constructed, and an aggregation phase where the aggregate values are continually routed toward the querying node. Within the aggregation phase, each node periodically uses the function SG() to convert sensor data to a local synopsis and the function SF() to merge two synopses to create a new local synopsis. For example, whenever a node receives a synopsis from a neighbour, it may update its local synopsis by applying SF() to its current local synopsis and the received synopsis. Finally, the querying node uses the function SE() to translate its local synopsis to the final answer. The continuous query defines the desired period between successive answers, as well as the overall duration of the query [20, 27]. One-time queries can also be sup-ported as a special, simplified case.

Duplicate - Sensitive Aggregates

With synopsis diffusion, aggregation can be done over arbitrary message routing topologies. The main challenge of a synopsis diffusion algorithm is to support duplicate-sensitive aggregates correctly for all possible multi-path propagation schemes. To achieve this, it require the target aggregate function (e.g., Count) to be mapped to a set of order- and duplicate-insensitive (ODI) synopsis generation and fusion functions.

Demerits:

The possibility of node compromise introduces more challenges because most of the existing in-network aggregation algorithms have no provisions for security. A compromised node might attempt to thwart the aggregation process by launching several attacks, such as eavesdropping, jamming, message dropping, message fabrication, and so on.

Attack Resilient Hierarchical Data Aggregation in Sensor Networks (2006)

Previous aggregation frameworks have been designed without security in mind. Given the lack of hardware support for tamper-resistance and the unattended nature of sensor nodes, sensor networks are highly vulnerable to node compromises. This algorithm shows that even if a few compromised nodes contribute false sub-aggregate values, this result in large errors in the aggregate computed at the root of the hierarchy. Here presented modifications to the aggregation algorithms that guard against such attacks this approach is scalable and efficient.

In our approach, nodes execute the synopsis diffusion aggregation algorithm as specified in [3, 14]. However, a subset of the nodes include along with their synopses a message authentication code (MAC) that can be used by the sink to verify the validity of their contribution to the aggregate function. The key observations behind the design of our approach are that

- In order to derive the correct estimate from the final synopsis (say *S*) computed at the sink, it need only to figure out the correct lowest order bit (say *r*) in *S* that is 0.
- The number of nodes contributing a 1 to bit *j* decreases exponentially as it move from the lowest order bit (*j* = 1) to higher order bits of the synopsis.

The operation of the protocol is similar to that of the protocol used in the basic approach with some minor differences as follows. The query message broadcast to the network includes the window size *w* in addition to the other parameters. As in the original synopsis diffusion algorithm [3, 14], here assume that the time is synchronized among BS and the sensor nodes. Each node computes the start and end time of the current epoch, based on the window *w*. Further, although the MACs generated by nodes are sent to the BS over the course of multiple epochs, the fused synopsis computed by each node is forwarded to its parent in the first epoch. Thus, the BS can compute the aggregate at the end of the first epoch itself, although this aggregate may be erroneous in the presence of compromised nodes.

Demerits:

An attack-resilient aggregation algorithm for the synopsis diffusion framework, but the current attack-resilient algorithm proposed is not efficient.

IV. THE PROBLEM STATEMENT

Now present the assumptions, discuss the threat model, and formally state the problem that address in this paper.

A.ASSUMPTIONS

Now assume that sensor nodes are similar to the current generation of sensor nodes, e.g., MicaZ or Telos motes, in their computational and communication capabilities and power resources, while BS is a laptop class device supplied with long- lasting power .Here assume that BS cannot be compromised and it uses a protocol such as μTesla to authenticate its broadcast messages to the network nodes. We also assume each node shares a pair-wise key with BS. Let the key of the node with ID X be denoted as K_X. To authenticate a message to BS, a node X sends a MAC (Message Authentication Code) generated using the key K_X. We further assume that each pair of neighbouring nodes has a pairwise key to authenticate its mutual communication.

B.THREAT MODEL

The synopsis diffusion framework on its own does not include any provisions for security. To stop unauthorized nodes from interfering in (or eavesdropping on) communications from unauthorized nodes in the rest of this paper. However, cryptographic mechanisms cannot prevent attacks launched by compromised nodes because the adversary can obtain cryptographic keys from the compromised nodes. Compromised nodes might attempt to thwart the aggregate computation process in multiple ways. A compromised node *C* which happens to be an in-network data aggregator may leak (to the adversary) the sensor readings (and sub-aggregates) which *C* receives from *C*'s child nodes. Several researchers already proposed privacy-preserving aggregation algorithms, and we do not consider this problem in the rest of this paper. Below we discuss other potential problems and identify the scope of this paper.

1. Falsifying the local value: A compromised node *C* can falsify its own sensor reading with the goal of influencing the aggregate value. There are three cases.

Case (i): If the local value of a honest node can be *any* value (i.e. not bounded by the domain of application), then a compromised node can pretend to sense *any* value. In this case, there is no way to detect the falsified local value attack. We leave Case (i) out of the scope of this paper. **Case (ii)**: If the local value of a honest node is bounded, and a compromised node falsifies the local value within the bound, there is no solution for detecting such an attack as in Case (i). We only observe that in Case (ii), the impact of this attack is limited as explained in the Appendix.

Case (iii): The local value of a honest node is bounded, and a compromised node falsifies the local value outside the bound. Our proposed algorithm does detect and guard against Case (iii) attack scenario as discussed in Section V-D.

2. Falsifying the sub-aggregate: A compromised node *C* can falsify the sub-aggregate which *C* is supposed to compute based on the messages received from *C*'s child nodes. It is challenging to guard against this attack, and addressing this challenge is the main focus of this paper.

We assume that if a node is compromised, all the infor- mation it holds will be compromised. We conservatively consider that all malicious nodes can collude or can be under the control of a single attacker. We use a Byzantine fault model, where the adversary can inject any message through the compromised nodes. Compromised nodes may behave in arbitrarily malicious ways, which means that the *sub-aggregate* of a compromised node can be arbitrarily generated. However, we assume that the attacker does not launch *DoS* attacks, e.g., the multi-hop flooding attacks [11] with the goal of making the whole system *unavailable.*

III IMPLEMENTATION

Proposed work goal is to enable BS to obtain the 'true' estimate of the aggregate (which BS would compute if there were no compromised nodes) even in the presence of the attack. More formally, goal (a) is to detect if ˆ*B*, the synopsis received at BS is the same as the 'true' final synopsis *B*, and goal (b) is to compute *B* from ˆ*B* and other received information. Without loss of generality, in the rest of the paper we limit our discussion to the context of Sum aggregate (if not otherwise specified).

Let us remind the reader that a compromised node *X* launches the falsified sub-aggregate attack by inserting one or more *false '1's* in its fused synopsis. An obvious solution to guard against this attack is as follows. BS broadcasts an aggregation query message which includes a random value, *Seed*, associated to the current query. In the subsequent aggregation phase, along with the fused synopsis ˆ *BX*, each node *X* also sends a MAC to BS authenticating its sensed value *vX* . Node *X* uses *Seed* and its own ID to compute its MAC. As a result, BS is able to detect and filter out any false '1' bits inserted in the final synopsis *B*.

Considering the above observation, we design an attack resilient protocol having two phases as follows:

• In phase one, we run the simple protocol described above.

That is, each node *X* forwards one randomly selected MAC for each '1' bit in ˆ *BX*. At the end of this phase, BS verifies the received MACs. The '1's in ˆ*B* for which no valid MACs have been received by BS are reset to '0'. Let *B* represent the final synopsis at BS after the above filtering process is performed. Analyzing .*B* , we make an estimate, ˆ *r* of the expected prefix length, *r* of *B*. We will show later that ˆ *r* is a lower bound of *r*.

• In phase two, nodes which contribute to bit ˆ *r* or to the bits to the right of bit ˆ *r* send a MAC to BS. In this phase, no random selection technique is employed in forwarding MACs—each node forwards all of the received MACs toward BS. The main challenge is how to get a good estimate, ˆ *r* in the first phase. We will show that in the presence of *t* compromised nodes, the deviation can be kept within O(log2 *t*). In this case, the number of MACs transmitted (per synopsis) in the second phase will be O(*t*), i.e. proportional to the number of compromised nodes, *t*.

Performance Analysis:

The communication overhead of phase one does not depend on the number of compromised nodes. The worst case per node communication burden is to forward *l* MACs, where *l* is the maximum number of ‘1’s in the synopsis. From property 1 of Sum synopsis, *l* is approximately log2 *S*, *S* being the Sum. That means the communication overhead per node is *O*(log2 *S*).

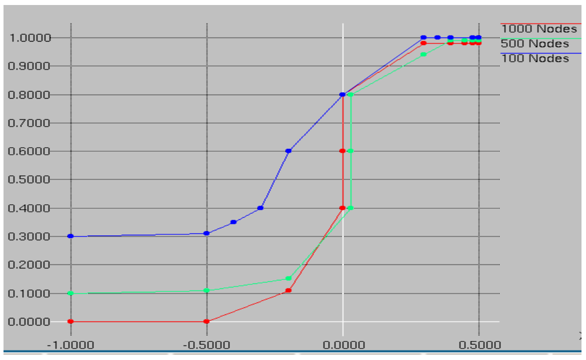

Fig 3.1 Performance

IV COMPARISION WITH PREVIOUS PROTOCOL

Protocols	Latency	Communication overhead	No of keys stored in each node	Dos resilient
Our protocol	2 epochs	*max(O(m log S), O(mt))*	O(1)	No
Attack resilient protocol[8]	*O(*log *χ/w__)*	*O(m* · 2w--)	O(1)	No
SDAP protocol[9]	2 epochs	O(N)worst case	O(1)	No
Tree sampling	O(log N)	*O(1/ δ 2 log 1/δ)*	O(log S)	Yes

[11]	epochs			

IV CONCLUSION

The security issues of in-network aggregation algorithms to compute aggregates such as predicate Count and Sum. In particular, we showed the falsified sub-aggregate attack launched by a few compromised nodes can inject arbitrary amount of error in the base station’s estimate of the aggregate. We presented an attack-resilient computation algorithm which would guarantee the successful computation of the aggregate even in the presence of the attack.

V FUTURE SCOPE

In future, to implement the concept of Efficient Ring - based hierarchical aggregation algorithms, to avoid the compromised node corruption in aggregation hierarchy during secure data communication.

VI REFERENCES

[1] M. Liu, N. Patwari, and A. Terzis, “Scanning the issue,” *Proc. IEEE*, vol. 98, no. 11, pp. 1804–1807, Apr. 2010.

[2] P. Corke, T. Wark, R. Jurdak, W. Hu, P. Valencia, and D. Moore, “Environmental wireless sensor networks,” *Proc. IEEE*, vol. 98, no. 11, pp. 1903–1917, Nov. 2010.

[3] S. Madden, M. J. Franklin, J. Hellerstein, and W. Hong, “TAG: A tiny aggregation service for ad hoc sensor networks,” in *Proc. 5th USENIX Symp. Operating Syst. Des. Implement.*, 2002, pp. 1–3.

[4] J. Zhao, R. Govindan, and D. Estrin, “Computing aggregates for monitoring sensor networks,” in *Proc. 2nd Int. Workshop Sensor Netw. Protocols Appl.*, 2003, pp. 139–158.

[5] J. Considine, F. Li, G. Kollios, and J. Byers, “Approximate aggregationtechniques for sensor databases,” in *Proc. IEEE 20th Int. Conf. DataEng. (ICDE)*, 2004, pp. 449–460.

[6] S. Roy, S. Setia, and S. Jajodia, “Attack-resilient hierarchical data aggregation in sensor networks,” in *Proc. ACM Workshop Security Sensor Adhoc Netw. (SASN)*, 2006, pp. 71–82.

[7] Y. Yang, X. Wang, S. Zhu, and G. Cao, “SDAP: A secure hop-byhop data

aggregation protocol for sensor networks," in *Proc. ACM MOBIHOC*, 2006, pp. 356–367.

[8] M. Garofalakis, J. M. Hellerstein, and P. Maniatis, "Proof sketches: Verifiable in-network aggregation," in *Proc. 23rd Int. Conf. Data Eng. (ICDE)*, 2007, pp. 996–1005.

[9] H. Yu, "Secure and highly-available aggregation queries in large-scale sensor networks via set sampling," in *Proc. Int. Conf. Inf. Process. Sensor Netw.*, 2009, pp. 1–12.

[10] S. Roy, M. Conti, S. Setia, and S. Jajodia, "Secure data aggregation in wireless sensor networks," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 1040–1052, Jun. 2012.

[11] S. Nath, P. B. Gibbons, S. Seshan, and Z. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," in *Proc. 2nd Int. Conf. Embedded Netw. Sensor Syst. (SenSys)*, 2004, pp. 250–262.

[12] M. B. Greenwald and S. Khanna, "Power-conservative computation of order-statistics over sensor networks," in *Proc. 23th SIGMOD Principles Database Syst. (PODS)*, 2004, pp. 1–11.

[13] P. Flajolet and G. N. Martin, "Probabilistic counting algorithms for data base applications," *J. Comput. Syst. Sci.*, vol. 31, no. 2, pp. 182–209, 1985.