Volume 5, Issue 2, February 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Query Confidentiality in Wireless Sensor Networks

Nawghare Pushpmala S.
ZES's DCOER Pune,
Maharashtra, India

Abstract: ***Wireless sensor networks processing sensitive data are facing the risks of data manipulation, data fraud and sensor destruction or replacement. This concerns applications such as the gathering of data on environmental pollution around industrial installations, or sensor systems replacing traditional video monitoring. Large-scale deployment in practice is conditioned by solving these kinds of security problem and reducing the risks due to limited physical protection of the devices and openness of the wireless communication channel. While modern cryptography and computer security offer many ways of solving these problems. In this paper we propose SafeProtocol to improve the strength of security, for wireless transaction which is designed using combination of both symmetric and asymmetric cryptographic techniques. This protocol provides three cryptographic primitives such as integrity, confidentiality and authentication. These three primitives can be achieved with the help of DES, and HMCA Message Digest MD5. That is it uses Data Encryption Standard (DES) for encryption, HMACMD5 for authentication and integrity.***

Keywords: Confidentiality, Integrity, Sensor Networks, Authentication.

I. INTRODUCTION

Wireless sensor networks (WSNs) can be defined as network of devices denoted as nodes that can sense the environment and communication the information gathered from the monitored field through wireless links. WSN have been widely deployed for various applications, such as tracking mobile objects traversing in different geographical locations, environment sensing, building safety monitoring, earthquake prediction etc. Why there is need of security for example if we see the application of sensor networks for developing SafeHome software in that suppose we have an organization in that some confidential information is being shared and we don't want any unauthorized entry inside that home, so we can deploy a sensor network to sense the entries and give them authorization to enter inside. Security is critical aspect in two-tiered wireless sensor network. To minimize the overall energy and memory consumption in a WSN Two-tiered sensor network model has been widely adopted. In Two tired sensor network base station that is storage nodes serve as an intermediate tier between sensors and a sink for storing data and processing queries, Where storage nodes serve as an intermediate tier between sensors and a sink for storing data. However, the presence of storage nodes makes easy to attack for attackers.

Therefore, there is need to develop a system that will not allow the attackers to hack the information from both sensor collected data and sink issued queries, which typically can be modeled as range queries, and allows the sink to detect hacked storage nodes when they misbehave. For confidentiality a storage node should not allow the attacker to obtain the sensitive information that has been, and will be, stored in the node, as well as the queries that the storage node has received, and will receive. All the queries should be confidential because they may leak critical information. For righteousness, the sink needs to detect whether a query result from a storage node includes forged data items or does not include all the data that satisfy the query. To overcome the drawbacks of earlier work we proposed a protocol safeProtocol in which to maintain confidentiality proper encoding is done and correctness of query is also maintained.

II. MODELS AND PROBLEM STATEMENT

2.1 System Model

For the problem we have considered two tiered wireless sensor network as it has more advantages in saving power and memory. Two tiered sensor network consist of main three components that is sensor node, storage node and sink node.

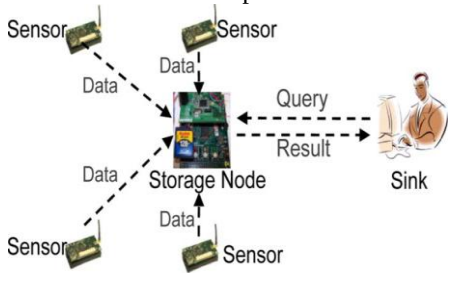

Figure 1: Architecture of two-tired sensor networks.

© 2015, IJARCSSE All Rights Reserved

Figure 2.1 shows the complete architecture of two tiered sensor network.

Sensors are hardware devices that produce a measurable response to a change in a physical condition like temperature or pressure. Sensors measure physical data of the parameter to be monitored. The continual analog signal produced by the sensors is digitized by an analog-to-digital converter and sent to controllers for further processing.

Storage nodes gather data from nearby sensors and answer queries from the sink of the network. As storage nodes store data received from sensors and serve as an important role for answering queries, they are more vulnerable to be compromised, especially in a hostile environment. A hacked storage node imposes significant threats to a sensor network. First, the attacker may obtain sensitive data that has been, or will be, stored in the storage node. Second, the compromised storage node may return forged data for a query. Third, this storage node may not include all data items that satisfy the query. The sink is the point of contact for users of the sensor network. Each time the sink receives a question from a user, it first translates the question into multiple queries and then disseminates the queries to the corresponding storage nodes, which process the queries based on their data and return the query results to the sink. The sink unifies the query results from multiple storage nodes into the final answer and sends it back to the user. Sink can detect compromised storage nodes when they misbehave.

2.2 Problem Statement

"Query Confidentiality in Wireless Sensor Networks"

The above problem statement is divided into sub problems as below:

1) Maintaining Confidentiality: Data privacy means that a storage node cannot know the actual values of sensor collected data. This ensures that an attacker cannot understand the data stored on a compromised storage node. Query privacy means that a storage node cannot know the actual value of sink issued queries. This ensures that an attacker cannot understand, or deduce useful information from, the queries that a compromised storage node receives.

2) Maintaining Righteousness: If a query result that a storage node sends to the sink includes forged data or excludes legitimate data, the query result is guaranteed to be detected by the sink as invalid. Besides these two hard requirements, a desirable solution should have low power and space consumption because these wireless devices have limited resources.

III. CONFIDENTIALITY FOR WIRELESS SENSOR NETWORKS

As WSNs are widely used, they need to overcome frequent failures to satisfy the traffic demands and unauthorized access. With aggregator nodes in WSNs First, the attacker may obtain sensitive data that has been, or will be, stored in the storage node. Second, the hacked aggregator node may return forged data for a query. Third, this storage node may not include all data items that satisfy the query. Wireless Sensor Networks need to overcome all these problems.

3.1 Query Confidentiality Preserving Method

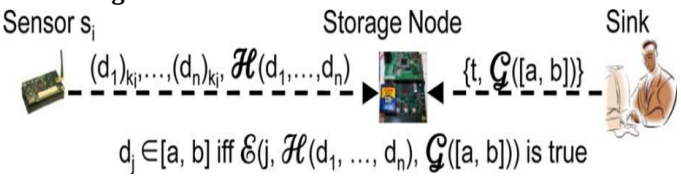

Figure 2: Idea for preserving Confidentiality

The Figure 2.shows the architecture of network topology used in experiment. This WSN is mainly subdivided into three parts Sensor node, Storage node and Sink node. To preserve privacy, it seems natural to have sensors encrypt data and the sink encrypts queries. However, the key challenge is how a storage node processes encrypted queries over encrypted data. The idea of our solution for preserving privacy is illustrated in Figure 2. We assume that each sensor Si in a network shares a secret key Ki with the sink. For the n data items d1,….,dn. that a sensor si collects in time-slot t,si. first encrypts the data items using key Ki, the results of which are represented as (d1)Ki,….,(dn) Ki. Then, Si applies a "magic" function $\mathcal{H}$ ti the n data item and obtains $\mathcal{H}$(d1,..,dn). The message that the sensor sends to its closest storage node includes both the encrypted data and the associative information $\mathcal{H}$(d1,..,dn). When the sink wants to perform query {t,[a,b]} on a storage node, the sink applies another "magic" function $\mathcal{G}$ on the range [a,b] and sends {t,$\mathcal{G}$([a,b])} to the storage node. The storage node processes the query {t,$\mathcal{G}$([a,b])} over the encrypted data (d1)Ki,….,(dn)Ki collected at time-slot t. using another "magic" function $\mathcal{E}$. The three "magic" functions $\mathcal{H}$, $\mathcal{G}$ and $\mathcal{E}$ satisfy the following three conditions. 1) A data item dj(1<=j<=n) is in range [a,b] if and only if $\mathcal{E}$(j,$\mathcal{H}$(d1..,dn),$\mathcal{G}$ [a,b]) is true. This condition allows the storage node to decide whether (dj)Ki should be included in the query result.

2) Given $\mathcal{H}$(d1,..,dn) and (dj)Ki it is computationally infeasible for the storage node to compute dj This condition guarantees data privacy.

3) Given $\mathcal{G}$([a,b]), it is computationally infeasible for the storage node to compute [a,b], This condition guarantees query privacy. We assume that sensors and storage nodes are loosely synchronized with the sink. With loose synchronization, we divide time into fixed duration intervals, and every sensor collects data once per time interval. From a starting time that all sensors and the sink agree upon, every n time intervals form a time-slot. From the same starting time, after a sensor collects data for n times, it sends a message that contains a 3-tuple (i,t,{d1…..dn}), where i is the sensor ID and t is the sequence number of the time-slot in which the n data items are collected by sensor si .

© 2015, IJARCSSE All Rights Reserved

3.2 Working of safe Protocol

SafeProtocol provides significantly better security and privacy. While prior art allows a compromised storage node to obtain a reasonable estimation on the value of sensor collected data and sink issued queries, SafeProtocol makes such estimation very difficult. Second, SafeProtocol delivers orders of magnitude better performance on both power consumption and storage space for multidimensional data, which are most common in practice as most sensors are equipped with multiple sensing modules such as temperature, humidity, pressure, etc.

To preserve privacy, each sensor *si* encrypts data items d1,…,dn using its secret key ki, denoted as (d1)ki ,...,(dn)ki. Note that, *ki* is a shared secret key with the sink. However, the key challenge is how a storage node processes encrypted queries over encrypted data without knowing their values. The idea of our solution is to convert sensor collected data and sink issued queries to prefixes, and then use prefix membership verification[1][2] to check whether a data item satisfies a range query. The sensor protocol concerns how a sensor sends its data to a storage node. Let d1,…,dn. be data items that sensor Si collects at a time-slot. Each item dj($1 \leq j \leq n$) is in the range(d0,dn+1) , where d0 and dn+1 denote the lower and upper bounds, respectively, for all possible data items that a sensor may collect. The values of and are known to both sensors and the sink. After collecting n data items, Si performs the following steps.

Sort the n data items in an ascending order. For simplicity, we assume d0<d1<d2< … dn <dn+1. If some data items have the same value, we simply represent them as one data item annotated with the number of such data items.

Convert the n+1 ranges[d0,d1],[d1,d2],[d2,d3],…,[dn,dn+1]to their corresponding prefix representation, i.e., compute S([d0,d1]), S [d1,d2], S [d2,d3],…, S [dn,dn+1])

Numericalize all prefixes. That is, compute N(S([d0,d1])), N(S [d1,d2]) , N(S [d2,d3]),…, N(S [dn,dn+1]).

Compute the keyed Hash Message Authentication Code (HMAC) of each numericalized prefix using key $\mathcal{g}$, which is known to all sensors and the sink. Examples of HMAC implementations include HMAC-MD5 [32]–[34]. An HMAC function using key $\mathcal{g}$, denoted HMAC$\mathcal{g}$, satisfies the one-wayness property (i.e., given HMAC$\mathcal{g}$(x) , it is computationally infeasible to compute x and $\mathcal{g}$) and the collision resistance property (i.e., it is computationally infeasible to find two distinct numbers x and y such that HMAC$\mathcal{g}$(x) = HMAC$\mathcal{g}$(y)). Given a set of numbers S, we use HMAC$\mathcal{g}$(S) to denote the resulting set of numbers after applying function HMAC$\mathcal{g}$ to every number in S. This step computes HMAC$\mathcal{g}$ (N(S([d0,d1]))),…, HMAC$\mathcal{g}$ (N(S([dn,dn+1]))).

Encrypt every data item with key Ki , i.e., compute (d1)Ki,..(dn)Ki.

Sensor Si sends the encrypted data along with HMAC$\mathcal{g}$ (N(S([d0,d1]))),…, HMAC$\mathcal{g}$ (N(S([dn,dn+1]))).to its closest storage node. The above steps show that the aforementioned "magic" function $\mathcal{H}$ is defined as follows:

$\mathcal{H}$(d1,….,dn)= HMAC$\mathcal{g}$ (N(S([d0,d1]))),…, HMAC$\mathcal{g}$ (N(S([dn,dn+1]))).

Due to the one-wayness and collision resistance properties of the HMAC function, given $\mathcal{H}$(d1,….,dn) and the encrypted data items (d1)Ki….(dn)Ki, the storage node cannot compute the value of any data item.

MD5 Algorithm

The **MD5** [7] message-digest algorithm is a widely used cryptographic hash function producing a 128-bit (16-byte) hash value, typically expressed in text format as a 32 digit hexadecimal number. MD5 has been utilized in a wide variety of cryptographic applications, and is also commonly used to verify data integrity.MD5 processes a variable-length message into a fixed-length output of 128 bits. The input message is broken up into chunks of 512-bit blocks (sixteen 32-bit words); the message is padded so that its length is divisible by 512. The padding works as follows: first a single bit, 1, is appended to the end of the message. This is followed by as many zeros as are required to bring the length of the message up to 64 bits fewer than a multiple of 512. The remaining bits are filled up with 64 bits representing the length of the original message, modulo 2^{64}.The main MD5 algorithm operates on a 128-bit state, divided into four 32-bit words, denoted *A*, *B*, *C*, and *D*. These are initialized to certain fixed constants. The main algorithm then uses each 512-bit message block in turn to modify the state. The processing of a message block consists of four similar stages, termed *rounds*; each round is composed of 16 similar operations based on a non-linear function *F*, modular addition, and left rotation.

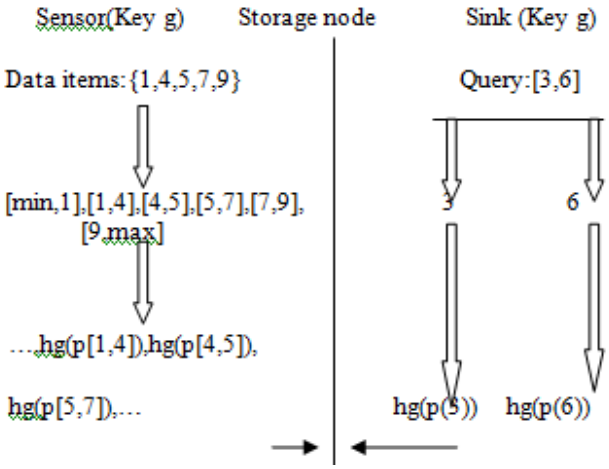

Figure 3: Privacy preserving scheme of SafeProtocol

© 2015, IJARCSSE All Rights Reserved

IV. EXPERIMENTAL DETAILS

I implemented SafeProtocol is a protocol that prevents attackers from gaining information from both sensor collected data and sink issued queries. SafeProtocol also allows a sink to detect compromised storage nodes when they misbehave. To preserve *privacy*, SafeProtocol uses a novel technique to encode both data and queries such that a storage node can correctly process encoded queries over encoded data without knowing their values. To start the communication between all nodes initially RMI registry should be their. First RMI should be initialized. After RMI Registry Sensor node is executed and started. The Sensor node shows files or data to be sent to the storage node. I have to select the data or file to be sent name of the file when I click on send button data is divided into packets and sent to the nearest storage node. If the packet is sent successfully it shows positive acknowledgement else negative acknowledgement and space consumption for that data. Once the RMI Registry done we should start storage node simultaneously Storage node consist of file details that is from which sensor node it is accepting files , It automatically generate the file Id for each file sent. If we want to see the particular file the user has to enter password to maintain confidentiality. If the password is wrong message is sent to the sink node. If the password is correct then user can access the file or data accepted by the storage node, Figure below shows the execution of storage node, that is server side and client side execution. When we start the sink node initially we have to register with the server, as we register it will provide one login id and password to the user. Using the given username and password user or sink has to login for accessing data from storage node. Once the user logged in using specified username and password he has options to see all received files by storage node from sensor node file details and search option to search or find required data. Again if we want to find particular data user has to enter registered password. At the sink node user can find the misbehavior of the node. If attacker tried wrong password to access data from storage node the sink node or user can be notified with misbehavior of the node. It shows all details as shown below.

In misbehavior details user will get the details like file name, node name, IP Address and error password. With this we can avoid gaining of files or data from unauthorized users.

4.1 Result Analysis

For graphical results JFreeChart application was used. Total 6 nodes were considered four sensor nodes, one storage node and last sink node. When sensor node send a file to storage node size of file is calculated that is space consumed by the node for that file as well as the time taken by the sensor network to send that file to storage node simultaneously calculated. We have taken time on x-axis of graph in ms and space in bytes. When sensor node one sends file or data to storage node following result will be generated.

4.1.1 Results for One Dimensional Data:

Here one dimensional data means sensor is sending single data item to storage node. In this result on x-axis time is displayed in ms and on y-axis space consumption is shown in bytes. For the file size of15000 bytes time taken is 35000ms this is for on dimensional data.

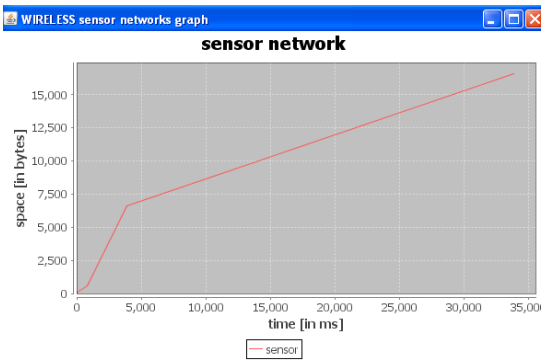

Figure 4: Graphical Result for One Dimensional Data

4.1.2 Results for Two Dimensional Data:

I measured the efficiency of Safe Protocol scheme on one- and two-dimensional data. The data set was chosen from a large real data set and it consists of the temperature, humidity, and voltage like data collected by 4 nodes.

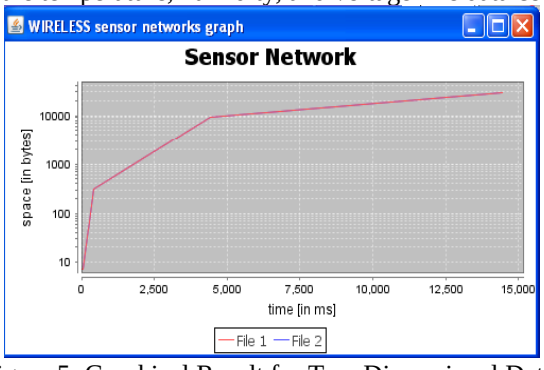

Figure 5: Graphical Result for Two Dimensional Data.

© 2015, IJARCSSE All Rights Reserved

We experimented with both SafeProtocol scheme on one-dimensional data (like temperature), two-dimensional data (like temperature and humidity). I have created one sensor node which is sending two data items to the storage node which will be accessed by sink node later. Figure below shows the execution in detail.

V. CONCLUSION

SafeProtocol an efficient protocol for handling range queries in two-tiered sensor networks in a confidentiality- and integrity- preserving fashion. SafeProtocol uses the techniques of prefix membership verification, Encryption by DES algorithm and MD5 algorithm. In terms of security, SafeProtocol significantly strengthens the security of two tiered sensor networks. Unlike prior art, SafeProtocol prevents a compromised storage node from obtaining a reasonable estimation on the actual values of sensor collected data items and sink issued queries. In terms of efficiency, our results show that SafeProtocol significantly outperforms prior art for data in terms of storage space.

REFERENCES

[1] Fei Chen and Alex X. Liu "Privacy- and Integrity-Preserving Range Queries in Sensor Networks" IEEE 2012 Transaction on Networking,Volume :PP, Issue:99
[2] F. Chen and A. X. Liu, "SafeQ: Secure and efficient query processing in sensor networks," in Proc. IEEE INFOCOM, 2010, pp. 1–9.
[3] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin,and F. Yu, "Data-centric storage in sensornets with GHT, a geographic hash table," Mobile Netw.Appl., vol. 8, no. 4, pp. 427–442, 2003
[4] P. Desnoyers, D. Ganesan, H. Li, and P. Shenoy, "Presto:A predictive storage architecture for sensor networks," in Proc. HotOS, 2005, p. 23.
[5] B. Sheng, Q. Li, and W. Mao, "Data storage placement in sensor networks," in Proc. ACM MobiHoc, 2006, pp. 344–355.
[6] B. Sheng, C. C. Tan, Q. Li, and W. Mao, "An approximation algorithm for data storage placement in sensor networks," in Proc. WASA, 2007,pp. 71–78.
[7] H. Krawczyk,M. Bellare, and R. Canetti, "HMAC: Keyed-hashing for message authentication," RFC 2104, 1997.
[8] R. Rivest, "The md5 message-digest algorithm," RFC 1321, 1992.

AUTHOR

Nawghare P. S. Received the B.E. degree in computer science and engineering and M.E. degrees in Computer Networking Engineering from BAMU Auramgabad, Maharashtra in 2008 and 2015, respectively. Currently working as a Assistant professor in ZES's DCOER, Pune, Computer Department.

© 2015, IJARCSSE All Rights Reserved