SECURE DATA AGGREGATION SCHEME FOR SENSOR NETWORKS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Kavit Shah

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Electrical and Electronics Engineering

December 2014

Purdue University

Indianapolis, Indiana

This is the dedication.

ACKNOWLEDGMENTS

This is the acknowledgments.

PREFACE

This is the preface.

TABLE OF CONTENTS

Page

LIST OF TABLES vii
LIST OF FIGURES viii
SYMBOLS ix
ABBREVIATIONS x
NOMENCLATURE xi
GLOSSARY xii
ABSTRACT xiii
1 Introduction 1
    1.1 Sensor Networks 1
    1.2 Internet Of Things 1
    1.3 Big Data 1
    1.4 Data Aggregation 2
    1.5 Cloud Computing 2
    1.6 Fog Computing 2
2 Related Work 3
    2.1 Secure Aggreagation 3
3 Problem Statment and Assumptions 4
    3.1 Assumptions 4
4 Secure Data Aggregation Scheme 5
    4.1 Network topology 5
5 Commitment Tree Generation 6
6 Verification of confirmations 7
    6.1 Network Model 7
        6.1.1 Sensor Nodes 7

Page

6.1.2 Collections of confirmations 7

6.1.3 Verification of confirmations 8

7 August 10

8 Summary 12

9 Recommendations 13

LIST OF REFERENCES 14

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

SYMBOLS

m	mass
v	velocity

ABBREVIATIONS

abbr	abbreviation
bcf	billion cubic feet
BMOC	big man on campus

NOMENCLATURE

Alanine	2-Aminopropanoic acid
Valine	2-Amino-3-methylbutanoic acid

GLOSSARY

chick	female, usually young
dude	male, usually young

ABSTRACT

Shah, Kavit Master, Purdue University, December 2014. Secure data aggregation scheme for sensor networks. Major Professor: Dr. Brian King.

This is the abstract.

1. INTRODUCTION

TIPS: USE ACTIVE VOICE USE VERBS DON'T TURN VERBS INTO NOUNS

COMMON MISTAKE: DATA ARE ; DATA IS PLURAL THAT/WHICH

Advancements in compute, storage, networks and sensors technologies have led to many new promising applications.

1.1 Sensor Networks

The sensor networks of the near future are envisioned to consist of hundreds to thousands of inexpensive wireless sensor nodes, each with some computational power and sensing capability, operating in an unattended mode. They are intended for a broad range of environmental sensing applications from vehicle tracking to habitat monitoring. Give an example and talk about energy, security constraints.

1.2 Internet Of Things

In the world of mass connectivity people need to get information all the time on an array of devices. Everything from your refrigerator to your thermostat is connected to wireless networks and joining the "internet of things". Write about bandwidth constraints.

1.3 Big Data

All the large internet companies process massive amounts of data also know as "Big Data" in real time applications. These include batch-oriented jobs such as data mining, building search indices, log collection, log analysis, real time stream processing, web search and advertisement selection on big data. To achieve high scalability, these applications distributes large input data set over many servers. Each server process its share of the data, and generates local intermediate. The set of intermediate results contained on all the servers is then aggregated to generate the final result. Often the intermediate data is large so it is divided across multiple servers which perform aggregation on a subset of the data to generate the final result. If there are N servers in the cluster, then using all N servers to perform the aggregation provides the highest parallelism. Talk about compute constraints. [**?**]

Airplanes are also a great example of "big data". In a new Boeing Co.747, almost every part of the plane is connected to the Internet, recording and sometimes sending continuous streams of data about its status. According to General Electric Co. in a single flight one of its jet engines generates half a tera bytes of data. This shows that we have too much of data and we are just getting started.

1.4 Data Aggregation

Data aggregation is an important technique used in many system architectures. The key idea is to combine the data coming from different sources eliminating the data redundancy, minimizing the number of packet transmissions thus saving energy, bandwidth and memory usage. This technique allows us to focus more on data centric approaches for networking rather than address centric approaches. [1]

1.5 Cloud Computing

1.6 Fog Computing

2. RELATED WORK

2.1 Secure Aggreagation

David Wagner in Resilient Aggregation in Sensor Networks describes various attacks on aggregation schemes and introduces stastical estimation theory to secure aggregation. It helps deciding secure aggregation function by defining function is resilient or not.

SIA: Secure Information Aggregation in Sensor Networks proposes secure aggragation scheme for single-aggregator model. It provides stastical security properties.

3. PROBLEM STATMENT AND ASSUMPTIONS

3.1 Assumptions

1. Trusted Querier 2. Leaves can report only one value at a time. 3. pre-knowledge of network topology

4. SECURE DATA AGGREGATION SCHEME

The goal of this thesis is to examine secure data aggregation schemes for various distributed systems.

Many modern world system designs are distributed in nature. The system design includes small, individual components doing their tasks precisely and lots of these components synchronize with all other components to complete the bigger task.

Many applications of sensor network are inharenlty distributed in nature. For example, scientific data collection, building health monitoring , building safety monitoring systems are distributed systems. Write an example how data aggregation happes in one particular application. [2]

The application design architecture for the internet of things is distributed as well. Write an example how data aggregation happens in one particular application. [**?**]

4.1 Network topology

Write about how all theses distributed systems can be classifieds into general tree structure.

Subsubsection heading

This is a sentence. This is a sentence.

5. COMMITMENT TREE GENERATION

Theorem 5.0.1 *Binary commitment tree is optimal for terms of verification as it requires minimum number of off-path values.*

Proof Let us say

$\log_3(n) = y$

$3^y = n$

$\log_2(3^y) = \log_2(n)$

$y * \log_2(3) = \log_2(n)$

$\log_3(n) * \log_2(3) = \log_2(n)$

$\log_3(n) = \frac{\log_2(n)}{\log_2(3)}$

$2 * \log_3(n) = [2/\log_2(3)] * log_2(n) = (1.2618) * log_2(n)$

$2 * log_3(n) > log_2(n)$

■

6. VERIFICATION OF CONFIRMATIONS

6.1 Network Model

We assume multihop network with a set $S = \{s1, ..., s_n\}$ of n sensor nodes. The network is organized in a tree topology, with the base station as the root of the tree. The trusted querier resides outside of the network & has more computation, storage capacity then the sensor nodes in the network. The base station and the querier knows total number of sensor nodes n and the network topology. All the wireless comuunication is peer-to-peer and we do not consider local wireless broadcast.

6.1.1 Sensor Nodes

We assume that each sensor node has a unique identifier s and shares a unique secret symmetric key K_s with the querier. We assume all the sensor nodes are capable of doing symmetric- key encryption and decryption. They are also capable of computing collision-resistant cryptographic hash function H.

6.1.2 Collections of confirmations

After each sensor node s has successfully performed the verification step for its leaf vertex u_s, it sends an authentication code to its parent in the aggregation tree. The authentication code for sensor node s is $\text{MAC}_{K_s}(N||ACK)$ where ACK is a special message, N is nounce and K_s is the secret key that s shares with the trusted querier. Once an internal sensor node has received authentication codes from all its children, it computes the XOR of its own authentication code with all the received a authentication codes, and forwards it to its parent. Finally, the q querier will receive an authentication code from the base station that c consists of the XOR of all the authentication codes received in the n network.

6.1.3 Verification of confirmations

Since the querier knows the secret key K_s for each sensor node s and it also knows the topology of the commitment trees in the forest, it can simulate the commitment trees in the forest. The querier simulates the commitment trees in the forest by computing the following authentication codes

$\text{MAC}_{K_1}(N||ACK) \oplus \text{ MAC}_{K2}(N||ACK) \oplus \text{ ... } \oplus \text{ MAC}_{K_n}(N||ACK)$;

$\text{MAC}_{K_1}(N||NACK) \oplus \text{ MAC}_{K2}(N||NACK) \oplus \text{ ... } \oplus \text{ MAC}_{K_n}(N||NACK)$

and creates two simulated commitment trees, one with the authentication codes of ACK messages and one with the authentication codes of NACK messages; where ACK is an acknowledgemnet message, NACK is a negative acknowledgemnet message & N is the query nounce. Then the querier merges all the commitment trees in the forest simulated with ACK messages, by taking XOR of the root of all the commitment trees in the forest and calculates a single root authentication code for the forest simulated with ACK messages. The querier does the same thing for the simulated commitment trees' forest of NACK messages. The querier stores all the simulated commitment trees and root authentication codes in the memory.

The querier receives a signle root authentication code from the base station. The querier compares the received root authentication code with its simulated root authentication code of ACK value commitment trees' forest. If those two values match it means every node in the network sent ACK message during collection of confirmation step. If those two values do not match means one or more nodes in the network sent NACK message during collection of confirmation step. In the case where root authentication codes do not match, the querier will proceed futher to find out which node or nodes in the network sent NACK message.

To find out node or nodes in the network reported NACK message, the querier will ask the base station to send authentication codes of all the commitment trees in the forest. After receiving the authentication codes from the base station, the querier will compare those authentication codes with its simulated authentication codes of ACK trees. After this comparision, whose authentication codes do not match, the querier will classify those trees as BAD TREES in the commitment forest.

The querier will store those BAD TREES. Then the querier will compare the authentication codes of those BAD TREES with its simulated authentication codes for NACK tree. If any of those authentication codes match it means all the nodes in that commitment tree sent NACK and we do not have to go further. If NACK valuse do not match then we repeat the process untill we found the node who reported NACK.

Otherwise, the querier will proceed furhter find the sensor node or nodes who sent NACK.

To find the sensor node or nodes who sent NACK the querier will ask authentication codes of all the commitment trees' root in the forest to the base station. Then the querier will compare the authentication codes of the commitment trees' root in the forest with the authentication codes of the simulated tress' root. From this comparision, the querier knows which tree or trees in the commitment forest did not report ACK. If the authentication code of the commitment tree's root does not match with the authentication code of the querier's simulted tree's root it means that one or more nodes in that commitment tree reported NACK during collection of confirmations phase.

For example, if only one tree did not report ACK then it will compare that value with the simulated tree of NACK. If those values mathces it means all the nodes in that tree reported NACK. If those two values do not match then it will repeat the process until it finds the node who reported NACK.

Simulates Receives Comparission till it finds the leave

7. AUGUST

Things discussed in meeting:

Analyzed congestion and why is it sub linear ?

In SHIA leaves verify their values with final results not with intermediate results. But in surveillance application data is compared with some base value in such network intermediate values are important.

Analyze the protocol with Digital signatures. How many signatures do we need ?

Analyze properties of commitment tree.

Definitions

A **direct data injection attack** occurs when an attacker modifies the data readings reported by the nodes under its direct control, under the constraint that only legal readings in [0, r] are reported.

An aggregation algorithm is **optimally secure** if, by tampering with the aggregation process, an adversary is unable to induce the querier to accept any aggregation result which is not already achievable by direct data injection.

For example, if A is an aggregator and it receives one reading from B. So, A needs to aggregate two values one of its own and the other is B's value. Suppose, maximum allowed value is 40. A0 = 10, B0 = 20. A1 = 30. A1 ¡= 80. If A reports any value out of that range it will get caught and any cheating within the range falls under direct data injection attack.

Congestion

As a metric for communication overhead, we consider node congestion, which is the worst case communication load on any single sensor node during the algorithm. Congestion is a commonly used metric in ad-hoc networks since it measures how quickly the heaviest-loaded nodes will exhaust their batteries [6, 12]. Since the heaviest-loaded nodes are typically the nodes which are most essential to the connectivity of the network (e.g., the nodes closest to the base station), their failure may cause the network to partition even though other sensor nodes in the network may still have high battery levels. A lower communication load on the heaviest-loaded nodes is thus desirable even if the trade-off is a larger amount of communication in the network as a whole.

For a lower bound on congestion, consider an unsecured aggregation protocol where each node sends just a single message to its parent in the aggregation tree. This is the minimum number of messages that ensures that each sensor node contributes to the aggregation result. There is $\Omega(1)$ congestion on each edge on the aggregation tree, thus resulting in $\Omega(d)$ congestion on the node(s) with highest degree d in the aggregation tree. The parameter d is dependent on the shape of the given aggregation tree and can be as large as $\Theta(n)$ for a single-aggregator topology or as small as $\Theta(1)$ for a balanced aggregation tree. Since we are taking the aggregation tree topology as an input, we have no control over d. Hence, it is often more informative to consider per-edge congestion, which can be independent of the structure of the aggregation tree.

Consider the simplest solution where we omit aggregation altogether and simply send all data values (encrypted and authenticated) directly to the base station, which then forwards it to the querier. This provides perfect data integrity, but induces O(n) congestion at the nodes and edges nearest the base station. For an algorithm to be practical, it must cause only sublinear edge congestion.

Our goal is to design an optimally secure aggregation algorithm with only sublinear edge congestion.

1. remove complement 2. variable range

8. SUMMARY

This is the summary chapter.

9. RECOMMENDATIONS

Buy low. Sell high.

LIST OF REFERENCES

LIST OF REFERENCES

[1] B. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," in *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on.* IEEE, 2002, pp. 575–578.

[2] D. Wagner, "Resilient aggregation in sensor networks," in *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks.* ACM, 2004, pp. 78–87.

[3] H. Alzaid, E. Foo, and J. G. Nieto, "Secure data aggregation in wireless sensor network: a survey," in *Proceedings of the sixth Australasian conference on Information security-Volume 81.* Australian Computer Society, Inc., 2008, pp. 93–105.