Data Survivability and Authenticity in Unattended WSN

Laltu Sardar

Data Survivability and Authenticity in Unattended WSN

Dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Technology
in
Computer Science

by

Laltu Sardar
[Roll No: CS-1420]

under the guidance of

Dr. Sushmita Ruj
Assistant Professor
Cryptology and Security Research Unit

Indian Statistical Institute
Kolkata-700108, India

July 2016

To my family and my supervisor

CERTIFICATE

This is to certify that the dissertation entitled **"Data Survivability and Authenticity in Unattended WSN"** submitted by **Laltu Sardar** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bonafide record of work carried out by him under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Dr. Sushmita Ruj
Assistant Professor,
Cryptology and Security Research Unit,
Indian Statistical Institute,
Kolkata-700108, INDIA.

Acknowledgments

I would like to show my highest gratitude to my advisor, *Sushmita Ruj*, Cryptology and Security Research Unit, Indian Statistical Institute, Kolkata, for his guidance and continuous support and encouragement. She has literally taught me how to do good research, and motivated me with great insights and innovative ideas.

My deepest thanks to all the teachers of Indian Statistical Institute, for their valuable suggestions and discussions which added an important dimension to my research work.

Finally, I am very much thankful to my parents and family for their everlasting supports.

Last but not the least, I would like to thank all of my friends for their help and support. I thank all those, whom I have missed out from the above list.

Laltu Sardar
Indian Statistical Institute
Kolkata - 700108 , India.

Abstract

An Unattended Wireless Sensor Network (UWSN) is a type of sensor network where a trusted sink visits each node periodically to collect the data. Due to offline nature of this network, every node has to secure the sensed data until next visit of the sink i.e. until data is being sent to the sink.

We will consider UWSNs operating in hostile environment where the goal of an attacker is to prevent targeted data from ever reaching the sink or to send corrupted data to the sink. So, in a UWSN data confidentiality and data authenticity are required. Besides we have to take care of survivability of sensed data.

In this thesis we have presented a scheme that provides authenticity, confidentiality and survivability of sensed data in efficient manner. All these issues were not addressed together in any of the previous scheme for securing UWSNs. We analyse the security performance of scheme and show that it enjoys high survivability, authenticity and confidentiality.

Keywords: *public key cryptography, symmetric key cryptography, PRNG, MAC, secret sharing scheme.*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Wireless Sensor Network (WSN) is a group of small sensors with communication infrastructure to monitor or record conditions at some diversified region. Traditionally data sensed by the sensor nodes were collected by a sink node in WSN. The sensor nodes were connected with sink throughout their lifetime.

UWSNs are a special kind of Wireless Sensor Network. They are consist of large number of very small sensor nodes. However in case of UWSNs, sensors are deployed in hostile environments. Since in most of the cases the environment is risky, sensors are deployed easily one time from helicopters, ships etc. So, sink node can not be connected with sensors all the time.

These sensor nodes are able to monitor different types of parameters like pressure, temperature, movements of vehicles, whether condition etc. On the other hand, sensors have limited power of computation as these have light CPUs, limited storage as little memories. Since they run on battery, their energy is also limited.

After deployment other nodes collect large amount of data till they runs out of battery power. They sense data and either keep the data saved with them or send the data to other nodes in the network. In such environments it is very difficult for a receiver node to collect data simultaneously from many sensor nodes. So, there is a sink node in the network to collect the data sensed by these sensors. Sink can not be connected all the time with all sensor nodes in the network due to unsuitable environment or other security purposes.

The sink visits sensors at regular intervals. Whenever the sink visits a node the node offloads saved data to the sink. Then sink sends all the collected data to a base station for further processing.

UWSNs have applications in many places hostile to human being like impassable mountains, underwater area, wide battlefield, extensive desserts, volcanic area, deep forest etc. Besides there are many applications of UWSNs in health care, environmental monitoring, traffic control, home automation etc.

1.2 Security Concerns

There are many kind of security problems exist in UWSNs. Providing data confidentiality, integrity, authenticity with high data survivable probability are major goals. Besides these, reducing data storage cost and communication cost, reducing the effects of compromised sensors and protecting location privacy are of interest.

There are three main properties that any UWSN should have - data confidentiality, data authenticity and data survivability.

Data Confidentiality The data collected by the sensors may be very sensitive. Let us consider a battlefield. Data should be kept secret from a third party. So we have to keep the data confidential. Looking only raw data transferred in the network, an eavesdropper should not learn anything about the sensed data. These can be achieved by the adversary encrypting the data.

Data Authenticity Adversary can modify sensed data or inject fraudulent data in such a way that it is undetected by the sink. These can lead one party to take wrong decision. Thus the sink has to be sure that the data is collected by the appropriate sensor and has not been altered in the mid way. Designing a good authentication scheme for UWSNs is needed to avoid such a situation.

Data Survivability Again a data sensor may be affected by some adversary who can erase data stored in some node or can destroy the sensor. The sink should get back the data which was collected by the affected sensor after previous visit of sink and before being destroyed. Since in most cases data is sensitive and we want to be reached data to sink, survival probability of data should be as higher as possible in UWSNs.

Thus in such a environment the goal of the adversary is not only to inject fraudulent data but also read and delete the data. Bringing data confidentiality, data authenticity and data survivability schemes can give a secure rigid communication in UWSNs.

Previous works [2, 3, 9–13, 18] have focused on authentication, confidentiality and survivability. Where no previous scheme has brought these three properties in a single platform before.

1.3 Our Contribution

We present a secure scheme for protecting either of these properties addressed by survivability authenticity and confidentiality. We have brought an authentication technique in that scheme. We have used a collaborative authentication technique together with data replication which increases the survivability of data in the network. Both public and private encryption schemes have been used. This is how we have brought the above mentioned three properties together in an efficient way.

We have provided mathematical analysis as well as simulation in details for our scheme. We have compared performance of our scheme with the existing schemes with respect to communication and storage cost. Our algorithm takes $O(mh)$ communication cost per round per node and $O(mv)$ storage cost in v round per node where, m is number of replication and h is the maximum distance of the neighborhood nodes.

1.4 Organization

The rest of the thesis is organized as follows.

- In Chapter 2 we have discussed previous work related to UWSNs.
- There is a brief description of required background in the Chapter 3.
- We have elaborated our proposed scheme in Chapter 4.
- In Chapter 6, Section 6.1 contains performance of the algorithm visualized by simulation and Section 6.2 contains the performance evaluation of our proposed scheme
- Conclusion and future direction research in this field is described in Chapter 7.

Chapter 2

Related Work

Wireless Sensor Network is a very mature research area. In this rich literature a number of topics like key management [16], authentication [5,20] have been discussed. Data confidentiality in WSNs is discussed in [6]. In [4] a good implementation of secure and distributed data discovery and dissemination protocol had been proposed. Vaidehi V in [21] proposed an light-weight secure data aggregation technique CKD minimizing the power usage and maximizing the secureness of data in the wireless sensor network. But most the schemes for WSNs are not suitable for UWSNs. This is because in WSNs constant presence of sink node is considered.

Data survivability in UWSNs in presence of capable adversary, was first introduced by Pietro et al. [9]. To increase the data survival probability they discussed the use of replication. They gave both analytical and simulation results on high value data. But the approach was non-cryptographic. Later on the same authors in their paper [11] presented the same work delving into the problem much deeper and constructed more effective and efficient countermeasures to maximize data survivability in UWSNs in the presence of a powerful mobile adversary.

In 2009 they presented [12] an in-depth investigation of security problems unique to UWSNs and proposes some simple and effective countermeasures for a certain class of attacks in non-cryptographic way.

In 2009 with Soriente et al. [13] he first introduced authentication scheme for UWSNs where mobile adversary trying to replace sensed data. They discussed two techniques $CoMAC$, $ExCo$ based on sensor cooperation that achieve a higher level of security.

In the same year, Ma et al. [7] defined adversarial model suitable for UWSNs.

They discussed different security challenges in UWSNs against the *Curious, Search-and-erase* and *Search-and-replace* type of powerful mobile adversary. Ren et al. [17] proposed a distributed data storage and retrieval scheme for UWSNs. Using Homomorphic Encryption and secret sharing the scheme was proposed keeping in mind the goals of confidentiality, resilience to node compromises, reliability, and efficiency of storage and transmission.

Backward secrecy gives security against proactive adversaries. Such an adversary starts compromising before receiving any information about the target sensor and the target data collection round. Forward secrecy is resilient against reactive adversaries which compromises a sensors after identifying its target [13].

Proper data authentication scheme should be both forward and backward secure. A backward secure scheme ensures that a key compromisation should not disclose any past keys, where a forward secure scheme ensures that compromisation of a current key does not reveal any future keys. Forward secure schemes for UWSNs has been proposed in [2].

Pietro et al. [8] constructed a *Proactive co-Operative Self-Healing (POSH)* protocol that provides both forward and backward secrecy in the presence of a powerful mobile adversary. Forward and backward secrecy are also present in [3, 13].

In 2011 Dimitriou and Sabouri [3] introduced two schemes, *Pollination* and *Pollination Light*, that defend against both reactive and proactive mobile adversaries. The schemes were inspired by the *Pollination* process in nature.

Reddy et al. [14] proposed two non-cryptographic distributed schemes $DS - PADV$ and $DS - RADV$ based on replication of data that maximizes the data survivability with little overhead. $DS - PADV$ protects against proactive adversary which compromises nodes before identifying its target. $DS - RADV$ gives security against in the network reactive adversary which compromises nodes after identifying the target.

Pietro and Guarino [10] in 2013 focused on information availability and confidentiality via secret sharing in UWSN. They used secret sharing only to share sensed data. Based on simple cryptographic hashing and replication, Reddy et al. [15] introduced a data authentication scheme which provides a good defense against the mobile adversary with a little communication and memory overhead.

In 2015 Sen et al. [18] proposed a new scheme ($ADSC$) that increased data survivability by replicating and data confidentiality by encrypting. However these scheme does not have any authentication technique.

Chapter 3

Preliminaries

In this chapter we present descriptions and assumptions about the sensor network environment. We also discuss here adversarial attack model and its limitations. Notations to be used throughout rest of the thesis are summarized in the Table 3.1.

3.1 Network Environment

We define some terms that we will use throughout the thesis. We are to discuss about a homogeneous UWSN which work as follows.

- There are N small sensors $\{SN_1, SN_2, \ldots, SN_N\}$ in the network uniformly deployed in a geographical area and a sink node which collects data from sensors.
- **Rounds:** Time is divided in rounds. In each round each sensor collects a single data. For the jth sensor SN_j collects data d_j^r at round r. We assume that all the sensor nodes in the network are synchronised.
- **Independent work:** Each sensor in the network works independently of any other sensor.
- **Unattendedness:** Each sensor SN_j offload data to the sink node on or before v_j rounds.
- **Energy and storage:** Any sensor has enough energy and sufficient storage for required computation. Each sensor assume to have micro-processor in built for such computation.

Symbol	Meaning
$\mathcal{ADV}$	The mobile adversary
N	Number of sensor nodes in UWSN
n_a	Number of nodes $\mathcal{ADV}$ can compromise
n_s	Number of nodes the sink visits to collect data in a round
Enc_{sym}	Probabilistic symmetric key encryption scheme
Enc_{pub}	Probabilistic public key encryption scheme
x_j	An unique random number given by the sink to SN_j.
d_j^r	Data sensed by the sensor node j at round r
z_j^r	Digest of the d_j^r
K_j^r	Key to find digest of d_j^r
sk_j^r	Randomly generated secret key corresponding Enc_{sym}
sk_j^r	Secret symmetric key of node SN_j at round r
n	Number of key shares
t	Number of key shares to reconstruct symmetric key
m	Number of data copies
DC_j^r	Set of m nodes to send data shares
$DC_j^r[i]$	ith element of DC_j^r
KS_j^r	Set of n nodes to send key shares
$KS_j^r[i]$	ith element of KS_j^r
$v_{j'}$	Maximum Number of rounds corresponding to node SN_j'.
PK_j	Public key corresponding to node SN_j
$seed_j$	Seed given to node SN_j at the beginning of round

Table 3.1: Notations

- **Communication:** As soon as sensor node offloads the data, the complete storage will be deleted and sink will reinitialise its secret parameters. Also value of any secret parameter after sink visit is not depended on that of before sink visit.
- **Awareness:** Every sensor in the network is aware of its location and the location of its immediate neighbours.
- **Connectedness:** All sensor nodes are connected all the time with each other either directly or indirectly through another sensors. The path depends on the underlying protocol.
- **Cryptographic capabilities:** Every node has a Pseudo-Random Number Generator (PRNG) and has a hash function like SHA-256.

3.2 Adversarial Attack Model

We will now give a description about a mobile adversary $\mathcal{ADV}$.

- **Goal:** The goals of the adversary are as follows:
  - $\mathcal{ADV}$ may want to inject fraudulent data in the network remaining undetected.
  - $\mathcal{ADV}$ may want to delete all data stored in the sensor before the data is offloaded.
  - $\mathcal{ADV}$ may want to read what data exchanged between nodes and sink.
- **Compromising power:** $\mathcal{ADV}$ can compromise at most n_a sensors in any round. As soon as $\mathcal{ADV}$ compromises a sensor it can read all data stored in the memory, control all communication and delete all storage.
- **Knowledge of topology:** $\mathcal{ADV}$ has complete knowledge of the protocol and the topology of the network.
- **Independent choice of nodes:** The set of nodes compromised by $\mathcal{ADV}$ in a round may be dependent on that of on previous rounds. Any two set of nodes compromised in two different rounds may have non empty intersection.
- **Security awareness:** $\mathcal{ADV}$ is aware of all cryptographic algorithms used for protection in UWSN.

Besides these attacks $\mathcal{ADV}$ can physically corrupt the nodes. Sensors may fail due to the power depletion or due to any other natural causes resulting in loss of functionality.

3.3 Network Assumptions

In addition to the network and adversarial model we have few assumptions. These are as follows.

- $\mathcal{ADV}$ attacks the network after the initialization phase. i.e. after data being sensed and encrypted and key shares being generated.
- Sensor nodes transmit data copies or key shares with multicast communication in parallel over multiple channels. An adversary can listen at most $t-1$ channel at a time.

- In each round, number of nodes compromised by an adversary $\mathcal{ADV}$ is very less compared to number of nodes visited by the sink. i.e. $n_a \ll n_s$,
- Sink is assumed to be a trusted authority and cannot be compromised.

3.4 Secret Sharing Scheme

Secret sharing scheme is a threshold scheme in which, without enough shares of a secret reconstruction is impossible from information theoretic sense. Formally,

A (t, n) *threshold Secret sharing scheme* is a cryptographic algorithm which generates n shares of a secret S where,

- the secret S can be reconstructed from any t or more shares.
- No information can be revealed about the secret S from $t-1$ or less number of shares,.

3.4.1 Shamir's secret sharing scheme

Shamir's Secret Sharing is a such threshold secret Secret Sharing scheme. It was created by *Adi Shamir* [19] in 1979. His scheme is based on polynomial interpolation where elements are taken from finite field.

Description Let $\mathbb{Z}_p$ be a finite field of prime order p and S be a secret. We represent secret as a element from $\mathbb{Z}_p$. That is $S \in \mathbb{Z}_p$.

Share Generation To construct a (t, n) threshold scheme we take a polynomial $P(x) = \sum_{i=0}^{t-1} a_i x^i$ of degree $t-1$ where $a_0 = S$ and other coefficients are chosen randomly from $\mathbb{Z}_p$. n shares are generates as n linearly independent pairs $(x_1, P(x_1))$, $(x_2, P(x_2))$, ..., $(x_n, P(x_n))$.

Reconstruction Since the polynomial P is of degree $t-1$ with t unknown coefficients, value at t points can uniquely identify the polynomial.

An efficient approach to find the secret is to use Lagrange polynomial L. Then the secret will be

$$S = L(0) = \sum_{j=1}^{t} P(x_j) \prod_{\substack{m=1 \\ m \neq j}}^{t} \frac{x_m}{x_m - x_j} \tag{3.1}$$

where $(x_1, P(x_1))$, $(x_2, P(x_2))$, ..., $(x_t, P(x_t))$ are known.

3.5 Cryptographic Primitives

Cryptography is a way of transmitting or storing data in a particular format so that it can be read or access by only the people whom it is intended for.

Definition 3.5.1 *A cryptosystem is a five tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where*

- *$\mathcal{P}$ is called "plaitext space", the finite set of possible plaintexts.*
- *$\mathcal{C}$ is called "ciphertext space", the finite set of possible cypertexts.*
- *$\mathcal{K}$ is called the "keyspace", the finite set of all keys.*
- *$\mathcal{E} = \{E_k : k \in \mathcal{K}\}$ is the set of functions $E_k : \mathcal{P} \to \mathcal{C}$, called encryption functions.*
- *$\mathcal{D} = \{D_k : k \in \mathcal{K}\}$ is the set of functions $D_k : \mathcal{C} \to \mathcal{P}$, called decryption functions.*
- *For each $e \in \mathcal{K}$, there is $d \in \mathcal{K}$ such that $D_d(E_e(p)) = p$ for all $p \in \mathcal{P}$*

3.5.1 Symmetric key cryptosystem

In *symmetric key cryptosystem* same key is used for both encryption and decryption process. This cryptosystem privately verifiable since both sender and receiver kept the key secret. Use of these type of cryptosystm make encryption process fast. *3DES, Serpent, AES (Rijndael), Blowfish, CAST5, Grasshopper and RC4* are examples of some popular symmetric key cryptosystem.

3.5.2 Public key cryptosystem

Public key cryptosystem is publicly verifiable. In this cryptosystem for each encryption key $e \in \mathcal{E}$, there is a decryption key $d \in \mathcal{D}$ where d is kept in public and e in private. *RSA, ElGamal, Paillier* are few examples of such public key cryptosystem.

3.5.3 Message Authentication Code

Message Authentication Code (MAC) is a small piece of information derived from a message which helps the receiver to verify authenticity of the sender. It is also called "keyed hash function". Formally,

A *MAC* consists of three probabilistic polynomial time algorithms G S and V where,

- On input 1^n (n is the security parameter), key generation algorithm G output a key k.
- On input the key k and a message m, signing algorithm S outputs a tag t.
- Verification algorithm V tells whether a received message m' together with tag t is authenticated or not by checking $S(k, m') = t$ or not.

Few popular examples of MAC algorithm are MD5, SHA-1, SHA-256 etc.

Chapter 4

Proposed Scheme

In this paper we introduced a scheme that ensures authenticity, confidentiality and survivability of data in UWSNs. Our scheme has two sets of algorithms- one for sensor nodes and other for sink node. Sensor nodes has three algorithms-

1. SensingNode: which senses the data
2. SendingNode: which sends data copies and key shares to the neighboring nodes
3. StoreNode: which stores the data received from other nodes and update the key used for authentication scheme

Sink node has two algorithms-

1. SinkNodeDataCollection: in which sink collects data from sensor nodes and verifies integrity.
2. SinkNodeDataAuthentication: in which sink verifies data authenticity.

At the beginning of each round r as soon as a node SN_j senses data d_j^r it encrypt the data with a random secret sk_j^r with symmetric encryption and find a authentication tag z_j^r with. Then the secret key sk_j^r is broken into shares and sending node sent them into the neighbouring node to make adversary difficult to get the key back. Few encrypted data are also replicated and replication are being sent by sending nodes to the neighbouring node.

Store nodes store the key shares and data replicas and keep tags to be sure that stores information is not being altered. At the same period of time it also update the MAC key used to find tag of the sensed data.

The sink visits nodes after few rounds and collect the stored informations from them. Thereafter since sink has all the seeds it can regenerate all situations and authenticate the received data.

4.1 Algorithms for Sensor Nodes

Before discussing the algorithm we have few assumptions.

1. We assume the existence of a secure communication channel through which sensor node communicates with sink node.
2. There exists a one-way collision-resistant function $F(\circ)$ (for example SHA-256).
3. All data will be reached its destination from source node. No transmitted data will be missed.

Whenever sink visits a node SN_j, it collects data stored in the sensor node and gives $seed_j$, PK_j, K_j^1 and x_j back to the node. $seed_j$ is a seed to find the set of neighbouring nodes to distribute data copies and key shares. PK_j is a public key corresponding to Enc_{pub} that will be used to store data in the sensor node. The secret key corresponding to that will be kept by sink itself.

K_j^1 is the key for MAC for the first round. For rest of the rounds it will be calculated in collaboration with other nodes in the network. x_j is to identify the node SN_j. This can be a random number but unique in the network.

4.1.1 Algorithm for Sensing Nodes

At the beginning of round r, as soon as sensor SN_j senses data d_j^r, data digest z_j^r with key K_j^r and cipher text c_j^r with sk_j^r is calculated. Then the sensed data is deleted permanently. sk_j^r is randomly generated secret key for the symmetric encryption function Enc_{sym}. So after this step sensed data can be recovered only if sk_j^r is known.

Algorithm 1 $SensingNode()$, operated by SN_j in round r

Input: K_j^r,
Output: z_j^r, c_j^r, SK_j^r
1: Select sk_j^r
2: Sense d_j^r
3: $z_j^r \leftarrow HMAC(d_j^r, K_j^r)$,
4: $c_j^r \leftarrow Enc_{sym}(d_j^r, sk_j^r)$
5: Delete d_j^r
6: Construct key shares $(sk_{j,1}^r, sk_{j,2}^r, \ldots, sk_{j,n}^r)$ of sk_j^r
7: $SK_j^r \leftarrow \{sk_{j,1}^r, sk_{j,2}^r, \ldots, sk_{j,n}^r\}$
8: Delete sk_j^r
9: Store z_j^r temporarily
10: **return**

Instead of keeping sk_j^r, its shares are generated by using Shamir's (t, n) secret sharing scheme. Then sk_j^r is also deleted from memory. At the end of the algorithm z_j^r is stored temporarily i.e. without any encryption (by PK_j).

4.1.2 Algorithm for the Sending Nodes

After generating the key shares, sending nodes send them to the neighbouring nodes. Each node SN_j has the knowledge of its set of neighboring nodes N_j. N_j can be either the set of all nodes in the network or its subset. N_j is fixed for a node and known to the sink.

Algorithm 2 $SendingNode()$, operated by SN_j in round r

Input: x_j, c_j^r, SK_j^r
Output: Null
1: $DC_j^r \leftarrow SelectNodesDC(N_j, m, r, j)$
2: $KS_j^r \leftarrow SelectNodesKS(N_j, n, r, j)$
3: **for** $i \leftarrow 1$ to m **do**
4: Send $(c_j^r || x_j)$ to $DC_j^r[i]$
5: **end for**
6: **for** $i \leftarrow 1$ to n **do**
7: Send $(sk_{j,i}^r || x_j)$ to $KS_j^r[i]$
8: **end for**
9: Delete c_j^r, KS_j^r, DC_j^r and SK_j^r
10: **return**

Each node SN_j in round r selects two set of nodes DC_j^r and KS_j^r. $SelectNodesDC$ and $SelectNodesKS$ are algorithms which takes inputs N_j j, and r together with m and n respectively. They respectively returns m and n number of nodes from N_j. The selection is deterministic and based on the result of a PRNG initialized by the seed $seed_j$ given by the sink.

DC_j^r and KS_j^r are to send data copies and key shares respectively. Instead of sending directly, x_j is concatenated. x_j is a unique random number given to SN_j to identify the node. Since data copies and key shares will be stored in other nodes, sink need to know source of the data for authentication process. After distributing, all encrypted data copies and key shares, DC_j^r and KS_j^r will be deleted permanently.

4.1.3 Algorithm for the Store Nodes

All nodes send data replica and keys shares. At a given point of time a node $SN_{j'}$ can get either a key share $(sk_{j,i}^r||x_j)$ or replica $(c_j^r||x_j)$. If destination of received data is any other node in the network then it is forwarded according to the routing protocol, otherwise is stored in the node.

Data replicas and key shares are stored in two different arrays $SKA_{j'}^r$ and $CA_{j'}^r$ respectively. $XK_{j'}^r$ and $XC_{j'}^r$ are arrays to store the identities of the sender nodes (given by the sink) from which key shares and data replicas respectively are received. To preserve integrity of the received data we keep MAC with $z_{j'}^r$ for the data in both arrays $SKA_{j'}^r$ and $CA_{j'}^r$.

Algorithm 3 *StoreNode*(), operated by $SN_{j'}$ in round r

Input: $K^r_{j'}$, $PK_{j'}$, $z^r_{j'}$
Output: $K^{r+1}_{j'}$

Initialisation : $XK^r_{j'}$, $SKA^r_{j'}$, $XC^r_{j'}$, $CA^r_{j'} \leftarrow []$, $p, q \leftarrow 0$
while round is not over **do**
  Receive data copy $(c^r_j||x_j)$ **or** key share $(sk^r_{j,i}||x_j)$
  if destination is not other node **then**
    if receives key Share **then**
      $p \leftarrow p + 1$
      $XK^r_{j'}[p] \leftarrow x_j$
      $SKA^r_{j'}[p] \leftarrow sk^r_{j,i}$
    else
      $q \leftarrow q + 1$
      $XC^r_{j'}[q] \leftarrow x_j$
      $CA^r_{j'}[q] \leftarrow c^r_j$
    end if
  else
    Forward received data according to routing protocol
  end if
end while
Sort $XK^r_{j'}$ and change $SKA^r_{j'}$ accordingly
Sort $XC^r_{j'}$ and change $C^r_{j'}A$ accordingly
$HSK^r_{j'} \leftarrow MAC(z^r_{j'}||SKA^r_{j'}[1]||SKA^r_{j'}[2]|| \ldots ||SKA^r_{j'}[p])$
$HC^r_{j'} \leftarrow MAC(z^r_{j'}||CA^r_{j'}[1]||CA^r_{j'}[2]|| \ldots ||CA^r_{j'}[q])$
$K^{r+1}_{j'} \leftarrow F(K^r_{j'}||XK^r_{j'}[1]|| \ldots ||XK^r_{j'}[p]||XC^r_{j'}[1]|| \ldots ||XC^r_{j'}[q])$
Store $CA^r_{j'}$, $HC^r_{j'}$ $SKA^r_{j'}$, $HSK^r_{j'}$, $z^r_{j'}$
Delete $K^r_{j'}$
return $K^{r+1}_{j'}$

Since sequence of receiving data is not fixed, after the end of receiving, array of identities of the sources XK^r_j and XC^r_j are sorted and corresponding $SKA^r_{j'}$ and $CA^r_{j'}$ are arranged accordingly. At the end next round key K^{r+1}_j is updated. A one-way collision-resistant function $F(\circ)$ is used to compute the next rounds key. Also $CA^r_{j'}$, $HC^r_{j'}$ $SKA^r_{j'}$, $HSK^r_{j'}$, $z^r_{j'}$ are stored in encrypted form. Enc_{pub} with $PK_{j'}$ is used for encryption.

4.2 Algorithms for the Sink node

In every round sink visits n_s number of nodes in the network. The algorithm of sink node is divided in two parts - *data collection* and *data authentication.*

4.2.1 Data Collection by Sink Node

Sink collects all encrypted data stored in the nodes for the sink. The public key encryption Enc_{pub} is used to encrypted that data. Sink gives the public key for encryption to the nodes and stores corresponding private keys. Here we assume that sink can not be compromised. As soon as it collects data from a node sink can decrypt it. For some data d, *get* d means sink get d back after collecting its encrypted from $\overline{d}$ and decrypt it as d with appropriate private key.

Algorithm 4 *SinkNodeDataCollection*()

Input: $\mathcal{N}_t$
Output: $\mathcal{N}'_t$, $\{z_t^r, SKA_j^r, CA_j^r\}$s for all $SN_j \in \mathcal{N}_t$
Initialisation: $\mathcal{N}'_t \leftarrow \Phi$, $p, q \leftarrow 0$
for All node $SN_j \in \mathcal{N}_t$ **do**
  for $r \leftarrow 1$ to v_j **do**
    get CA_j^r, HC_j^r, SKA_j^r, HSK_j^r, z_j^r
    $M_1 \leftarrow MAC(z_j^r||SKA_j^r[1]||SKA_j^r[2]||\ldots||SKA_j^r[p])$
    if $HSK_j^r \neq M_1$ **then**
      Add SN_j to $\mathcal{N}'_t$
      break
    end if
    $M_2 \leftarrow MAC(z_j^r||CA_j^r[1]||CA_j^r[2]||\ldots||CA_j^r[q])$
    if $HC_j^r \neq M_2$ **then**
      Add SN_j to $\mathcal{N}'_t$
      break
    end if
  end for
  Re-initialises all required fields
end for
return

Let $\mathcal{N}_t$ be the set of of nodes that sink visits at some point of time t. Sink gets all CA_j^r, HC_j^r, SKA_j^r, HSK_j^r, z_j^r 's for all nodes $SN_j \in \mathcal{N}_t$. At the time of collection it

verifies integrity of the arrays CA_j^r and SKA_j^r by calculating MAC. Since sink can decrypt z_j^r it can verify MAC. If verification fails it adds the corresponding node to $\mathcal{N}'_t$ and stops collection from that node. At the end sink re-initialises all require fields for all nodes.

4.2.2 Data Authentication by Sink Node

After collecting all data from nodes in the network sink recovers data sensed by sensor nodes and authenticates them. At that point sink has sensed data encrypted with symmetric key.

Algorithm 5 *SinkNodeDataAuthentication*()

Input: All K_j^r's, $\mathcal{N}'_t$, $\{z_t^r, SKA_j^r, CA_j^r\}$s for all $SN_j \in \mathcal{N}_t$
Output: d_j^r's

1: **for** each node j **do**
2: **for** $r \leftarrow 1$ to v_j **do**
3: **if** #(key shares from nodes $\notin \mathcal{N}'_t$) $< t$ **then**
4: **continue** for next r
5: **else**
6: take any t key shares
7: reconstruct sk_j^r
8: **end if**
9: **if** no c_j exists in nodes $\notin \mathcal{N}'_t$ **then**
10: **continue** for next r
11: **else**
12: take c_j from a node $\in \mathcal{N}'_t$
13: **end if**
14: calculate $d_j^r \leftarrow Dec_{sym}(c_j, sk_j^r)$
15: calculate $\hat{z}_j^r \leftarrow MAC(d_j^r, K_j^r)$
16: **if** $\hat{z}_j^r = z_j^r$ **then**
17: accept data d_j^r for the node SN_j and round r
18: **else**
19: reject d_j^r
20: **break**
21: **end if**
22: **end for**
23: **end for**
24: **return**

For each round r and each sensor SN_j Sink collects takes t key shares from uncompromised nodes and reconstructs key sk_j^r. So it can get the data d_j^r back. As sink knows the seed $seed_j$ for each node SN_j it can calculate sets of destination nodes as well as set of received nodes for every round. Hence it can calculate round key K_r^j for each node j.

We assume sink has all such round keys calculated in advance. At the time of calculation the key K_j^r sink also sort the both arrays of receiving nodes. Thus sink can authenticate by checking if z_j^r equals to $MAC(d_j^r, K_j^r)$ or not.

Chapter 5

Mathematical Analysis

In this chapter, we give a brief analysis of our proposed scheme. We analyse *security, storage and communication costs*. Besides we calculate *data survivable probability*. At the end we compare performance of our scheme with the existing schemes. We have verified the outcomes of the analysis in Section 6.1.

5.1 Security Analysis

In our scheme we have considered data authenticity, confidentiality and survivability. So adversary wins if

1. If $\mathcal{ADV}$ reads the sensed data.
2. If $\mathcal{ADV}$ changes data and remains undetected.
3. If $\mathcal{ADV}$ deletes all data sensed by a node.

In first case we will assume adversary can not harm during execution of any algorithm. Since the data is encrypted just after sensing, adversary cannot read the sensed data except if it gets the key sk_j^r. To get sk_j^r it needs at least t key shares.

A sensor sends data to other nodes in the network in multicast protocol. If we assume that adversary can eavesdrop on at most $t-1$ different channels simultaneously then, adversary will not get any data after public key encryption is done and stored.

In second case, let adversary $\mathcal{ADV}$ wants to compromise a node in round $\bar{r}$. So it has to eavesdrop on channel and calculate $sk_j^{\bar{r}}$. Also it has to replace all the authentication tags generated using $c_j^{\bar{r}}$ and $sk_j^{\bar{r}}$. To generate that tags adversary needs z_j^r of the corresponding nodes. To change z_j^r adversary need K_j^r and d_j^r. But adversary is not able to get K_j^r.

Thus the only way to compromise the network is to delete the data sensed by a node. We have discussed below about the *Data Survivable Probability*.

5.2 Data Survivable Probability

Let sink can collect n_s and adversary can compromise $n_a(\ll n_s)$ nodes in a round. The network contains N nodes.

So before $\bar{r}$ round sink can cover $(\bar{r}-1)n_s$ and adversary can compromise $(\bar{r}-1)n_a$ nodes.

Suppose adversary compromises node containing $C_j^{\bar{r}}$ at round $\bar{r}$. The probability that $C_j^{\bar{r}}$ is not covered by the sink on round $\bar{r}$ is given by

$$Pr_1 = \left(\frac{N-(\bar{r}-1)n_s}{N}\right)\left(\frac{n_a}{N-(\bar{r}-1)n_a}\right) \tag{5.1}$$

Probability that a replica is safe till the sink covers the entire network in v round

$$Pr_2 = \prod_{q=0}^{v-1}\left(1-\left(\frac{N-n_sq}{N}\right)\left(\frac{n_a}{N-n_aq}\right)\right) \tag{5.2}$$

Probability that a node is compromised at the end of v round

$$Pr_3 = 1 - Pr_2 \tag{5.3}$$

Probability that k nodes are compromised is

$$Pr_4 = Pr_3^k \tag{5.4}$$

To get back the data we need at least t shares. In that case adversary can compromise 0 to at most $n-t$ number of nodes where shares resides. Probability that at least t replica is safe ie. any number form 0 to $n-t$ of nodes are compromised is

$$Pr_5 = 1 - \sum_{k=0}^{n-t} Pr_3^k \tag{5.5}$$

Thus the probability that at least t key shares are safe is given by

$$Pr_t = 1 - \sum_{k=0}^{n-t} \left[1 - \prod_{q=0}^{v-1} \left(1 - \left(\frac{N - n_s q}{N} \right) \left(\frac{n_a}{N - n_a q} \right) \right) \right]^k \tag{5.6}$$

Similarly Probability that at least 1 data replica is safe is given by

$$Pr_d = 1 - \sum_{k=0}^{m-1} \left[1 - \prod_{q=0}^{v-1} \left(1 - \left(\frac{N - n_s q}{N} \right) \left(\frac{n_a}{N - n_a q} \right) \right) \right]^k \tag{5.7}$$

Since the key shares and data shares are distributed randomly the resulting probability of data survivability in the network is given by

$$Pr_s = Pr_t . Pr_d \tag{5.8}$$

v can be taken as the maximum number of rounds for a node. If we consider for a particular node j we can take it as $v = v_j$.

5.3 Storage and communication Costs

Communication

In each round, each sensor sharing m copies of data and n shares of symmetric secret key. Since in general key size is much smaller than data size we can take size of $2n$ key shares $\approx$ size of a data. Thus every node in the network sends $O(m)$ data. Since there are N nodes in the network, $O(mN)$ amount of data are being sent in the network in each round.

In algorithm 2, we select DC_j^r and KS_j^r from set of nodes N_j in the network. Let us consider $N_j = N$. Again each node passes through several nodes to reach its destination. Since there are N nodes in the network and the network is homogeneously distributed in the network, we can say that to reach its destination a piece of data visits $O(\sqrt{N})$ nodes. Taking care of all data and key shares, we have an overall communication cost of $O(mN\sqrt{N})$

If $N_j \neq N$ then N_j is a proper subset of N. For a larger network, in our algorithms, we keep nodes in N_j which are within some fixed distance h. In that case communication cost will be reduced to $O(mNh)$.

Storage

Sensor nodes in the network collect key and data copies and key shares. There are at most v rounds before sink visit.

In a round each sensor generates $O(m+1)$ amount of data and an authentication tag of the sensed data. This data is sent to neighbouring nodes randomly. On an average a node needs $O(v(m+2))$ storage. It may happen that a particular sensor node gets more than that in a particular round. So we can give a bound on storage needed.

If we consider some node SN_j, then only from the nodes which are in a fixed distance h can send data to it. Let $N_b(\gg m)$ maximum number of neighbouring nodes.

In all v round there will be approx $N_b(m+1)v$ data generated. Each data has probability $\frac{m}{N_b}$ to reach SN_j. Probability that exactly i data will reach to SN_j is

$$\left(\frac{m}{N_b}\right)^i \tag{5.9}$$

Probability that less than l data will reach to SN_j is

$$\sum_{i=1}^{l}\left(\frac{m}{N_b}\right)^i \tag{5.10}$$

Probability that more than l data will reach to SN_j is

$$P_l = 1 - \sum_{i=1}^{l}\left(\frac{m}{N_b}\right)^i \tag{5.11}$$

If $l \approx O(cv(m+2))$, for some constant c, then probability that more than $O(cv(m+2))$ data will reach to SN_j is

$$P_l \approx 1 - \sum_{i=1}^{O(cv(m+2))}\left(\frac{m}{N_b}\right)^i \approx 1 \tag{5.12}$$

when $\sum_{i=1}^{O(cv(m+2))}\left(\frac{m}{N_b}\right)^i \approx 0$. We can choose c such that the probability is approximated to zero.

In case when the storage is limited, we can set a value $c > 1$. When the storage will be full we just forward the data to other random node depending on $seed_j$ or another seed $seed_{j'}$.

Chapter 6

Analysis and Discussion

6.1 Simulation

In this chapter we have evaluated effectiveness of our scheme. The important parameters in our algorithm are N, m, n, t, n_s and n_a. The data survivability in the network depends on all the parameters. We also used (t, n) Shamir's threshold scheme to provide confidentiality. The objective of the simulation is to find optimal value for the variables so that survivability is optimised with communication and storage cost. Simulation result in the section helps to determine optimal values.

6.1.1 Simulation Environment

In our simulation we have taken 1000 nodes i.e. $N = 1000$. We considered the nodes are in the network are homogeneously distributed. We have use python language for simulation. We ran each experiment 50 times to keep statistical robustness. Instead of taking different neighbourhood set we have chosen the set of all nodes as neighbourhood set. i.e. $N_j = N$, $\forall j$. We assume that sink can visit $N_s = 50$ nodes in a round to collect data from sensor nodes and initialise them. Here we have assume that if a node is compromised in some round r then the data stored in that nodes up to round r will be rejected. Thereafter sink initialises the node and collect the farther data stored in that node if that node is not compromised again.

6.1.2 Simulation Results

In the Figure 6.1.2 we have shown data survivability with respect to n_a, the number of compromised nodes by adversary per round. n_s is the number of nodes sink visits in every round and initializes them. In our simulation we have assume that $n_a < n_s$. So we varied n_a from 0 to n_s. We have simulate this for three different values 3, 4 and 5 of n, where n is the number of key shares. That is we have compare the performance for $(3, 5)$, $(3, 4)$ and $(3, 3)$ secret sharing scheme. The performance is checked for each of the cases where n, the number of data copies is 2, 3 and 4.

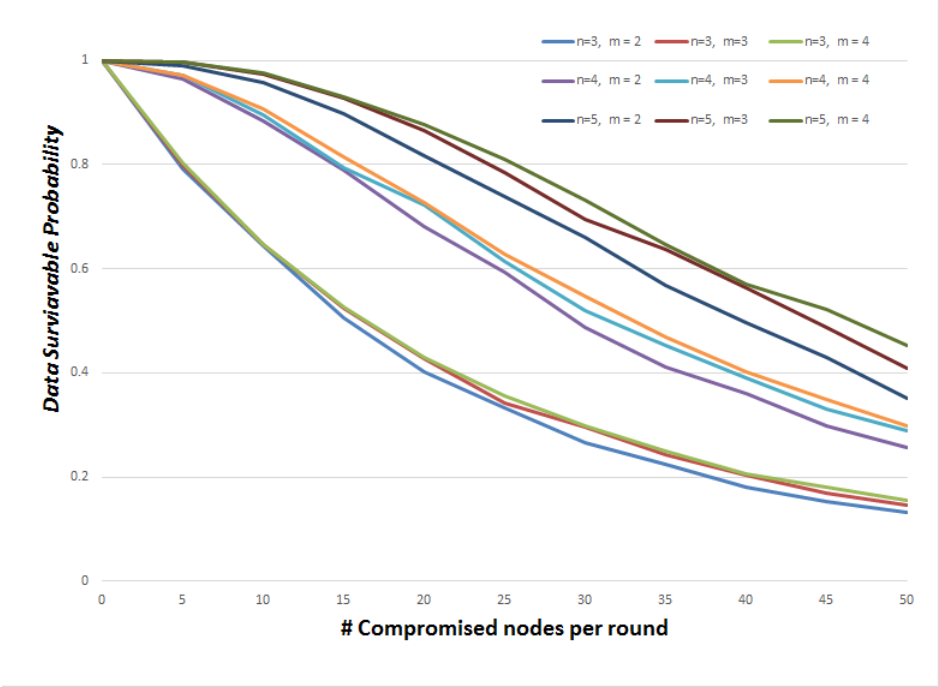

Figure 6.1: Survivable probability for different values of n_a

It is obvious that incrementing data copies can increment data survivability. But keeping in mind that copying data and sending it to the network increment communication cost. The figure 6.1.2 is showing that for $n = 5$ it performs best. We we can choose $(3, 5)$ secret sharing for better performance. Since our assumption permits an adversary not to compromise more that n_s nodes, we make our simulation accordingly. We have taken $n_s = 50$. We varied n_s from 0 to 50.

Difference in survivable probability for different values of n are easily visible. For $n = 5$ it performs best. Keeping $n = 5$ fixed we can see that for $m = 2$ network performance is not better than both when $n = 4$ or $n - 5$. Again performance of the

same for $n = 4$ and $n - 5$ are closer. Since communication cost matter we can say that $n = 4$ is enough. Thus we can take $m = 3$, $n = 5$ and $t = 3$ as the best value.

In Figure 6.1.2 we have shown survivable probability of different schemes. From

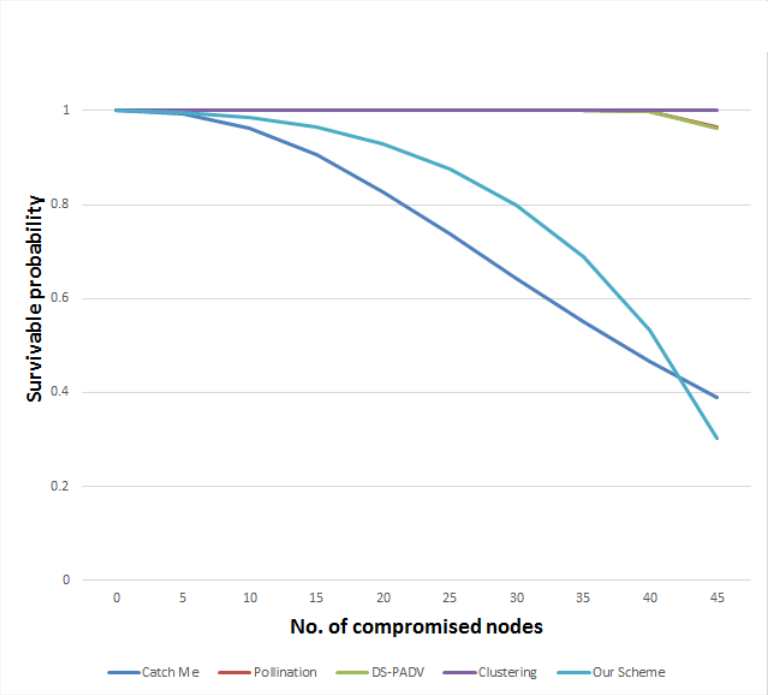

Figure 6.2: Survivable probability of different schemes of n_a

the both two figures it can be seen that the simulation result of our scheme gives almost same survivable probability.

6.2 Performance Evaluation

In this chapter we have given performance evaluation of our scheme in with respect to communication and storage cost. Comparison of the complexity with other schemes is also shown. The comparison is done theoretically.

6.2.1 Complexity Comparison

In our scheme we have deduced that we need $O(cv)$ storage where, maximum value of $c = O(max_j\{|N_j|\})$. Though it looks like quite large but actually it needs $c = O(m)$ memory on average. In general replicating of data and keeping them stored increases storage requited for a sensor. Schemes like *Pollination* [3], *Clustering* [15], $DS - RADV$ [14] etc. have used replication technique. In the table 6.1 we have compared our scheme with existing schemes with respect to storage and communication cost.

Table 6.1: Complexity Comparison (per node v rounds)

Scheme	Communication (per node per round)	Storage (per node v rounds)
Pollination [3]	$O(\sqrt{N})$	$O(v)$
$DS - PADV$ [14]	$O(h)$	$O(vh)$
$DS - RADV$ [14]	$O(mh)$	$O(vmh)$
Clustering [15]	$O(m\sqrt{N}/l)$	$O(vmh/l)$
$CoMAC$ [13]	$O(m\sqrt{N})$	$O(vt)$
$ExCo$ [13]	$O(m\sqrt{N})$	$O(vt)$
$ADSC$ [18]	$O(mh)$	$O(mv)$
Our Scheme	$O(mh)$	$O(cv)$

where $h = \#$ hops, $N = \#$ nodes, $v = \max \#$ rounds, $m = \#$ replica
$t = \#$ co-authenticator, $l = \#$ nodes in a cluster

From the table it can be seen that our scheme better than schemes like $DS - RADV$ [14] with respect to storage cost and like $CoMAC$ [13], $ExCo$ [13] etc. with respect to communication cost.

6.2.2 Problems Addressed in Different Schemes

In the table 6.2, issues related to existing schemes has been represented. The table is made for three features data confidentiality, authenticity and high Survivability. If a scheme has some feature, we use "✓" corresponding to that, where "✗" is used while the scheme does not have the corresponding feature.

Table 6.2: Problems Addressed in Different Schemes

Schemes	Confidentiality	Authenticity	High Survivability
Pollination [3]	✗	✓	✓
$DS-PADV$ [14]	✗	✓	✓
$DS-RADV$ [14]	✗	✓	✓
Clustering [15]	✗	✓	✗
$CoMAC$ [13]	✗	✓	✗
$ExCo$ [13]	✗	✓	✗
$ADSC$ [18]	✗	✓	✓
Our Scheme	✓	✓	✓

Each of the schemes which are mentioned in the table, also a lot which aren't, have some above mentioned features but not all. This is what we have tried to contribute in this paper. We have brought the three features in a single platform.

Chapter 7

Conclusion and Future Work

In this thesis, based on simple cryptographic hash functions, symmetric and public key encryptions, we have proposed a scheme where data authenticity and confidentiality are preserved with high data survivable probability. This scheme is inspired from Collaborative authentication techniques of *Pietro* [13] and data survivable scheme of [18]. In presence of authenticity and confidentiality, our scheme maximizes the survivable probability. The proposed scheme is also analysed and simulated based on compromisation power of adversary. Communication and storage costs are also compared with the existing scheme. However one can reduce the communication cost keeping the survivable probability unchanged.

Bibliography

[1] R. Di Pietro, S. Guarino, N. V. Verde, and J. Domingo-Ferrer. Review: Security in wireless ad-hoc networks - a survey. *Comput. Commun.*, 51:1–20, September 2014.

[2] Roberto Di Pietro, Luigi V. Mancini, Claudio Soriente, Angelo Spognardi, and Gene Tsudik. Playing hide-and-seek with a focused mobile adversary in unattended wireless sensor networks. *Ad Hoc Netw.*, 7(8):1463–1475, November 2009.

[3] T. Dimitriou and A. Sabouri. Pollination: A data authentication scheme for unattended wireless sensor networks. In *2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 409–416, Nov 2011.

[4] S. Ghormare and V. Sahare. Implementation of data confidentiality for providing high security in wireless sensor network. In *Innovations in Information, Embedded and Communication Systems (ICIIECS), 2015 International Conference on*, pages 1–5, March 2015.

[5] J. Kim, J. Baek, and T. Shon. An efficient and scalable re-authentication protocol over wireless sensor network. *IEEE Transactions on Consumer Electronics*, 57(2):516–522, May 2011.

[6] J. Luo, P. Papadimitratos, and J. P. Hubaux. Gossicrypt: Wireless sensor network data confidentiality against parasitic adversaries. In *2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 441–450, June 2008.

[7] D. Ma, C. Soriente, and G. Tsudik. New adversary and new threats: security in unattended sensor networks. *IEEE Network*, 23(2):43–48, March 2009.

[8] R. D. Pietro, D. Ma, C. Soriente, and G. Tsudik. Posh: Proactive co-operative self-healing in unattended wireless sensor networks. In *Reliable Distributed Systems, 2008. SRDS '08. IEEE Symposium on*, pages 185–194, Oct 2008.

[9] R. D. Pietro, L. V. Mancini, C. Soriente, A. Spognardi, and G. Tsudik. Catch me (if you can): Data survival in unattended sensor networks. In *Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on*, pages 185–194, March 2008.

[10] R. Di Pietro and S. Guarino. Data confidentiality and availability via secret sharing and node mobility in uwsn. In *INFOCOM, 2013 Proceedings IEEE*, pages 205–209, April 2013.

[11] Roberto Di Pietro, Luigi V. Mancini, Claudio Soriente, Angelo Spognardi, and Gene Tsudik. Maximizing data survival in unattended wireless sensor networks against a focused mobile adversary. *IACR Cryptology ePrint Archive*, 2008:293, 2008.

[12] Roberto Di Pietro, Luigi V. Mancini, Claudio Soriente, Angelo Spognardi, and Gene Tsudik. Data security in unattended wireless sensor networks. *IEEE Transactions on Computers*, 58(11):1500–1511, 2009.

[13] Roberto Di Pietro, Claudio Soriente, Angelo Spognardi, and Gene Tsudik. Collaborative authentication in unattended wsns. In *WISEC*, 2009.

[14] S. K. V. L. Reddy, S. Ruj, and A. Nayak. Distributed data survivability schemes in mobile unattended wireless sensor networks. In *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 979–984, Dec 2012.

[15] S. K. V. L. Reddy, S. Ruj, and A. Nayak. Data authentication scheme for unattended wireless sensor networks against a mobile adversary. In *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1836–1841, April 2013.

[16] A. S. Reegan and E. Baburaj. Key management schemes in wireless sensor networks: A survey. In *Circuits, Power and Computing Technologies (ICCPCT), 2013 International Conference on*, pages 813–820, March 2013.

[17] Y. Ren, V. Oleshchuk, and F. Y. Li. A distributed data storage and retrieval scheme in unattended wsns using homomorphic encryption and secret sharing. In *2009 2nd IFIP Wireless Days (WD)*, pages 1–6, Dec 2009.

[18] A. Sen, S. Ghosh, A. Basak, H. P. Puria, and S. Ruj. Achieving data survivability and confidentiality in unattended wirelesxs sensor networks. In *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, pages 239–246, March 2015.

[19] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

[20] P. Singaravelu and S. Verma. Rainbow signature for authentication in wireless sensor network. In *Wireless Communication and Sensor Networks (WCSN), 2010 Sixth International Conference on*, pages 1–5, Dec 2010.

[21] V. Vaidehi, R. Kayalvizhi, and N. C. Sekar. Secure data aggregation in wireless sensor networks. In *Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on*, pages 2179–2184, March 2015.

[22] Ren Yi. *Security Mechanisms in Unattended Wireless Sensor Networks*. Universitetet i Agder, Grimstad, Norway, December 2011.