A SECURE COMMUNICATION PROTOCOL FOR WIRELESS BIOMEDICAL SENSOR NETWORKS

by

Krishna Kumar Venkatasubramanian

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

ARIZONA STATE UNIVERSITY

August 2004

A SECURE COMMUNICATION PROTOCOL FOR WIRELESS BIOMEDICAL SENSOR NETWORKS

by

Krishna Kumar Venkatasubramanian

has been approved

July 2004

APPROVED:

_____, Chair

_____

_____

Supervisory Committee

ACCEPTED:

_____

Department Chair

_____

Dean, Division of Graduate Studies

ABSTRACT

Biomedical sensors, also called *biosensors*, are the next generation of health monitoring technology which will provide automated, accurate and real-time health monitoring capabilities for a patient. With a rapidly aging population, there will be a severe shortage of nursing staff in the coming decade, and biosensors have the potential to provide efficient and automated health-care for all.

Individual biosensors are extremely small in size and therefore have very limited processing capabilities, memory and battery power. Each biosensor therefore communicates the data it senses to an external sink with higher capability, called the *base-station*. The base-station processes the data sent by the biosensors and takes necessary action based on the processed data, like administration of a drug or calling a doctor. As the biosensors communicate sensitive health data to the base-station, it is essential to secure this data transfer in order to protect the patient's privacy. At the same time, energy consumption overhead due to communication within the network must be reduced.

One way to reduce the energy consumption in a biosensor network due to communication is by forming energy-efficient topologies. One such topology is a cluster where biosensors are organized in groups. Each cluster has a leader (called cluster head) which collects data from its cluster and relays it to the base-station. The cluster head is rotated regularly so that all nodes are elected cluster head thereby reducing the energy consumption per biosensor. Traditionally such clusters are formed in a distributed manner based on the signal strength of a solicitation beacon broadcasted by the elected cluster heads. As these protocols were not built with security in mind, they are susceptible to attacks such as HELLO Flood attack, Sinkhole attack and Selective Forwarding attacks, where a malicious external node can get complete control of the sensor network-base-station communication.

In this thesis two protocols are presented for preventing this security flaw when the sensors are organizing themselves into clusters in a biosensor network. The first is a centralized protocol, where the base-station is asked to form the cluster within the biosensor network. The second protocol is a distributed cluster formation protocol, which modifies traditional cluster formation protocol presented above by including a novel authentication scheme called *biometric authentication*. Extensions to these protocols have been proposed to distributed keys for securing actual data communication. The analysis of the protocols suggested that centralized cluster formation protocol is energy-intensive compared to the distributed protocol, but it provides better security guarantees. We have also done an actual implementation of the proposed protocols on a test-bed built using Mica2 sensor motes.

To

My Parents

ACKNOWLEDGMENTS

I would like to thank Dr. Sandeep Gupta for his constant encouragement and support without which this work would not have been possible. He has been a constant source of motivation and support. I am also thankful to my committee members Dr. Rida Bazzi and Dr. Hessam Sarjoughian for sparing their valuable time for being a part of my committee and providing useful feedback. Many thanks to everyone in the IMPACT lab for helping me at various stages of this thesis work. Last but not the least, I would like to thank my parents who have been a constant source of support and inspiration to me. Thank you all. This work was partially funded by NSF grant ANI-0086020.

TABLE OF CONTENTS

Page

Page

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Advances in the field of micro-electronics and low powered devices have paved the way for development of multi-functional smart-sensors. These sensors are extremely small in size, consist of data sensing, processing and short distance communication capabilities. The small size of the sensors makes it possible to deploy them in remote and hostile environments with relative ease. The communication ability of the sensors, allows us to organize them into networks. A typical sensor network consists of large number of sensor nodes that are densely deployed in the environment that needs to be monitored. The essential difference between a sensor network and other types of networks, like Internet, is that node-to-node communication is not of importance. The whole sensor network acts as a data source which can sends its sensed data to an external recipient. The communication within the network is used to facilitate the transmission of data to this recipient efficiently.

Sensor networks have many applications in many fields, from military to medicine to inventory control. Some of the uses of sensor networks include:

- Emergency response systems: This informs immediately of traffic accidents on free-ways.
- Medical monitoring of patients: Sensors in human body monitoring blood pressure,

blood sugar etc.

- Inventory management for a warehouse: Sensors inform the warehouse manager if certain important item stock is depleted, without physical inventorying.
- Battlefield information: Sensors sprayed over an area in enemy territory can inform us about troop movements, formations etc.
- Environmental Monitoring: Sensors can be deployed on coast lines for monitoring oil spills or in forests to monitor specific endangered animals.

As each sensor node is limited in size, their communication and computation capabilities are very limited. A sensor node can therefore do only a limited processing on the data it senses. In order to extensively process the data collected within the sensor network and take appropriate action on it (like administering drug or summoning paramedics to a crash site), the sensor nodes transmit their data to a collector entity located outside the sensor network. This collector entity is called a *base-station* or a sink. The base station thus acts as the interface to the outside world for the sensor network and is a powerful entity. A network can have many base-stations depending upon the size of the network and coverage of each base-station.

1. Biomedical Sensors

As mentioned above, sensor networks have many varied applications. One of the important domains where sensors can be used is patient care. A network of sensors can be deployed on or in the human body to monitor specific body parameters, such as blood glucose level, heart rate, presence or growth of cancerous cells, presence of specific antibodies

in the blood an so on. Such health monitoring sensors are called biomedical sensors or biosensors. Biosensors are also used in prosthetics, for example, using sensors to stimulate the retina to make the blind see [35], and drug delivery, where a doctor can control release a specific amount of drug into the body using an implanted sensor. Such network of biosensors on the body have the potential to revolutionize health care. With an aging world population, in the near future, there will be a severe shortage of health care professionals to take care of the old [1]. The development and adoption of biomedical sensors can automate the health monitoring process and provide good to care to all. Much work is being done to develop biosensors which are both implantable and external and used for biometrics like glucose, and delivery chemicals and drugs like insulin. Some examples are - external biosensors *I-Stat "lab on a chip"* developed for point of care diagnostics to be used in emergency rooms [4], implantable microprocessors like those manufactured by VeriChip [7], implantable biosensors for drug delivery being developed by MicroChips [5].

Some of the important characteristics of biosensors are summarized below:

- **Limited Size:** As the biosensors have to be carried by the patient, they have to be of very small size, in order to be non-intrusive.
- **Limited Energy Source:** As a biosensor is very small in size, the size of battery that can be placed within them is also limited.
- **Limited Memory, Processing and Communication Capability:** Due to the small sensor size, the memory available is limited and the processing capability is also constrained. Also as the energy source available is limited, communication capability is constrained as well.
- **Need for Continuous Operation:** As biosensors measure the health of a patient,

it is an essential need for sensors to be able to operate continuously over a long period of time, despite its limited energy source.

- **Robustness and Fault Tolerance:** As biosensors are placed inside the human body, replacing them is not easy, therefore a biosensor should be robust, work for many years without faults or errors.

2. Security in Biosensor Networks

Security is an essential need for biosensor networks. A typical biosensor collects sensitive medical information from a patient's body or provides services (drug delivery, prosthetics) to the patient. The health information collected from the patient's body is considered to be the property of the patient and needs to be protected as a legal requirement [9]. Therefore any biosensor network which senses various body parameters needs to ensure that the data is not leaked or provided to unauthorized entities, either during the actual sensing or communication process. Security is also necessary when biosensors are used to provide services like prosthetics and drug delivery. If unchecked, a malicious entity can masquerading as the controlling base-station, fool the biosensors to perform un-necessary and potentially dangerous tasks (like unwanted drug administration).

Providing security however is not an easy task, especially for biosensors. There are two main reasons for this, one, the biosensors use the wireless medium to communicate. As the wireless communication is broadcast in nature, it is easy for a malicious node to listen to the communication taking place and also to manipulate it by introducing bogus messages or modifying legitimate messages or simply jamming the communication. The second reason for difficulty in providing security is the limited capability of individual biosensors. Most of

the cryptographic algorithms used have been designed for wired networks with desktop machines which have huge computation power. A biosensor, as seen above, lacks many of the computation and communication capabilities of a traditional computer. Therefore implementing traditional cryptographic primitives used in wired networks directly on biosensors is not a feasible task.

Traditionally security has always been treated as an add-on rather than as an essential component of a system. The Internet is the best example of this scenario. Adding security as an after thought leads to flawed protocols which lead to easy compromise of the system. It is therefore essential that security issues for biosensor networks are addressed now, when the field is still in early stages of development.

We conclude this section, by listing out some of the problems that can be faced by an un-secure biosensor network.

- Passive monitoring of the communication in a biosensor network can give sensitive medical information to an unauthorized entity, who can use it for personal advantage.
- If security is not implemented in biosensor networks, a malicious node can easily prevent a legitimate warning generated by the health monitoring system from reaching appropriate authorities. This can cause severe harm to the patient's health.
- A malicious node can also do the reverse and generate false alarms for a patient. This can result in unwanted actions being taken by the system, like drug administration at a higher dosage, causing harm to the patient.
- As mentioned above, biosensor nodes need to be able to operate continuously for a long period of time. A malicious node can take advantage of the fact that biosensor have limited energy source and can send them bogus packets. A biosensor will try to

receive all these packets and act upon them leading to wastage of energy and possible network partition.

- An implanted biosensor dissipates heat into its surrounding tissue when operational. Another side effect of "overworking" the sensors will be the problem of tissue heating. Heating of body tissue to abnormal temperatures for prolonged periods of time can cause severe health problems (like leukemia). A malicious node, can therefore send bogus command messages to a biosensor forcing it to work (receive packets and process it) un-necessarily causing potentially, excessive heating of the nearby tissue.

3. Asymmetric, Symmetric Cryptography and Biosensors

As mentioned above, it is essential to maintain security in a biosensor network. Cryptography is used to maintain security for a communication between two entities. Cryptographic systems used can be divided into two types: Symmetric and Asymmetric Cryptosystems [34]. In symmetric cryptosystems, the communicating entities first agree upon a shared secret, called a key K. This key has to be both secret and authenticated. Subsequently, when the sender has to send any data, she first encrypts the data using an encryption function E and the key K to obtain a cipher text C which is sent to the receiver. $C = E_K(Data)$. The receiver uses a decryption function D and the same key to retrieve the data back. $Data = D_K(C)$. This ensures security, for data authentication, a message authentication code (which is a keyed hash function) is appended to the message being sent out. At the receiver, the message authentication code is computed again and compared with the one received with the message. If they match then the message is authenticated.

Though efficient, symmetric key cryptography has an inherent problem of key management-

key distribution and rekeying. Before any communication starts, the sender and the receiver have to obtain a shared secret key for encryption and authentication. Also, the same shared key cannot be used infinitely, the sender and the receiver have to, from time-to-time, change the shared key to prevent a malicious entity from figuring out the key by observing the encrypted traffic exchanged.

To reduce this overhead associated with symmetric key cryptography, asymmetric key cryptography was proposed [34]. Here we have two keys, one is a public key K_{pub} and another is a private key K_{prv}. The public key is known to every one, while the private key is a secret. A sender encrypts data using the receivers public key, and transmits it. $C = E_{K_{pub}}(Data)$, where C is the ciphertext obtained and E is the encryption function. The receiver then uses its private key and decrypts the data. $Data = D_{K_{prv}}(C)$, where D is the decryption function. The property of a public-private key pair is that one cannot be derived from the other. The problem however with asymmetric key cryptography is that, it is extremely computation intensive. Therefore asymmetric key cryptosystems are generally used for distributing symmetric key, while secure communication was done using symmetric cryptography.

From a biosensor point of view, symmetric cryptography is more efficient. Asymmetric cryptography involves heavy exponentiation and factorization of large prime numbers and are very expensive to implement in biosensors which have very limited resources. We therefore cannot even use them for facilitating symmetric key exchange. Therefore, we will use symmetric cryptography for biosensors.

It should be mentioned here, that the algorithms for various cryptographic primitives, like encryption and one-way hash functions, rely on the computational in-feasibility of guessing the input parameters (plaintext or key) from the output of the algorithms (ci-

phertext and the hash value) [34].

4. Thesis Overview

Communication protocols considered in this thesis, can be logically split into have two phases- *initialization phase or setup phase* and *data communication phase.* We define the initialization phase as organizing biosensors into a specific topology for energy-efficient communication [1]. Most biosensors will be smaller that sensors used in other applications like inventory management and therefore any direct communication with the base-station, which is located at a average distance of about 1m from any given biosensor, can be prohibitive. For this reason, we need dynamic topology management in our biosensor networks. Even if we use a fixed topology, some of the biosensors will consume more energy that others in communicating and will need to be charged soon causing tissue heating problems. In this data communication phase, the base-station sends out queries and the biosensors respond to them by transmitting the data collected from sensing their environment. In this thesis, we primarily look at security issues in the biosensor network **initialization phase**. The security issues of the data communication phase are a simple extension to those of the initialization phase. We touch upon how data communication phase is secured after we describe the protocols for securing the initialization phase.

In this thesis we primarily present two protocols (centralized and distributed) for securing the topology formation process in a biosensor network. We then extend the two protocols to include distribution of communication keys for secure data communication (after the initialization phase). We are addressing the security issues in a specific topology

[1] Initialization also consists of distributing control information to biosensors like communication schedule, but that can be done along with topology formation without any additional overhead

called *cluster* in this thesis. The two solutions proposed (and their extensions) are analyzed and their comparisons are presented along with the results of their implementation.

The thesis is organized as follows, Chapter 2 presents the preliminaries for the research. Chapter 3 presents the related work in this area. Chapter 4 presents our problem statement, followed by the description of our protocols in Chapter 5. Chapter 6, 7 and 8 present the mathematical analysis of the protocols, security analysis and the implementation results respectively. Chapter 9 finally presents the conclusion and future work.

CHAPTER 2

PRELIMINARIES

In this chapter we present some preliminary information that will help in better understanding of the research presented here.

1. Topology in Sensor Networks

A typical sensor in any sensor network (including biosensor network) is extremely small in size and severely energy constrained. Energy-efficiency is therefore one of the primary needs for any sensor network. It has been empirically shown that for a sensor node, the communication costs can be prohibitive [17]. In a sensor network, each sensor node senses its environment and sends the raw data collected to the base-station for processing. One of the ways of sending this data would be direct transmission to the base-station. This however is not a very energy-efficient way of communicating the data from the network to the base-station and the communication costs incurred are very high [23]. One of the ways of minimizing this communication cost is by organizing nodes into energy-efficient topologies, so as to average out the communication cost over the entire network. One such topology is a *cluster*.

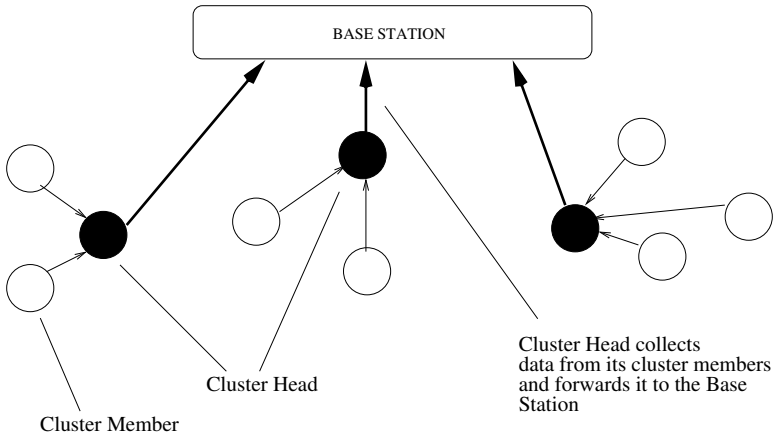

Figure 1. **Cluster Topology**

1.1. Cluster Topology. A cluster is basically a group of co-located sensor nodes which have formed a group or cluster to minimize their communication energy costs. A typical cluster is characterized by a leader node called cluster head. The cluster head is the sensor which is responsible for collecting data from its cluster members, aggregating it and forwarding it to the base-station. Therefore, in this topology the cluster heads are the nodes which perform the long distance communication to the base-station and hence consume more energy than the cluster member which perform only a short distance communication. In order to prevent the cluster head node from dying due to lack of energy, the cluster head node is periodically rotated. In the long run therefore the energy consumption due to communication is averaged out or spread over the entire network. The Figure 1 shows a typical cluster topology. In this thesis, we restrict ourselves to cluster formation mechanism in a biosensor network.

2. System Model and Assumptions

For the purpose of this work, we define a biosensor network as a network of wireless heterogeneous biomedical sensor nodes either worn outside or implanted inside the human

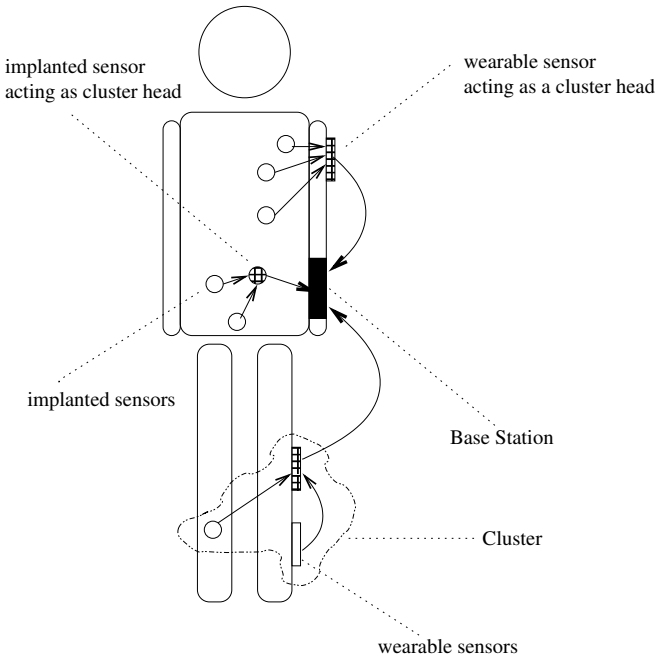

Figure 2. **System Model**

body, to monitor different body parameters. All biosensor communication is wireless in nature, as having wired communications between implanted and wearable sensors will be impractical. We assume the existence of protocols which maintain communication reliability for our model [31]. Figure 2 shows our system model, which contains both implanted and wearable sensors. We assume that after the initialization phase, the sensors use a query based communication model, where sensors sense their environment and reply only when the base-station queries them.

Our system model has three main types of entities: the base-station, the biosensors and the adversaries. We first describe the assumptions for the former two and then move on to describe the different types of adversaries.

Base Station:

- There is only one base-station controlling all the biosensors.

- The base-station performs the necessary key management for the entire biosensor network.
- The base-station is the gateway of the network to the outside world.
- It is assumed to have sufficient capability (power and computational capability) and means to communicate with external entities.
- The base station's signal covers the whole body. Therefore all biosensors are in the range of the base-station.
- The base-station is assumed to know the topology of the biosensor network.

Biosensors:

- Biosensors can be of two types, cluster heads and sensor nodes.
- The biosensors are heterogeneous in nature. Each biosensor senses one specific type of stimuli and cannot be re-programmed.
- As biosensors need to be able to operate continuously for long period of time, the existence of a mechanism to charge them, like charging using IR induction [35], is assumed.
- As the biosensors will organize themselves into a cluster topology, the requisite cluster heads are assumed to have been decided before the cluster formation starts. Also as the cluster heads have to be rotated over a period of time, we assume the existence of a new cluster head selection algorithm [23].
- The biosensors which will be the cluster heads know their roles, and the base-station is assumed to know their identities as well.

In the sequel, we make a clear distinction between the biosensors which are cluster heads and those which are not. The biosensors which are not acting as cluster heads are referred to as sensor nodes or cluster members. The whole network will be referred to as biosensor network.

We now define the assumptions about the capabilities of the adversary who will attack our system model. By defining our adversary, we can predict the attacks that are possible to mount on our system and what are the attacks that our system does not prevent. For our system model we define two types of adversaries: passive and active. We list their capabilities below:

- **Passive Adversary**: We define a passive adversary, as a malicious entity that has the capability to listen to all the communication taking place within our system. Such an adversary is assumed to be equipped with at least a processing power of a laptop or PC and have very powerful transmitter and a very sensitive receivers to capture weak signals originating from implanted sensors. In [27], they are called laptop-class adversaries. A passive adversary can therefore insert new messages, alter messages, replay old messages, spoof acknowledgments, spoof identity, and send messages at a higher signal strength in an attempt to gain control of the network. Passive adversaries do not have physical access to the network.

- **Active Adversary**: We define active adversaries as those which can physically compromise or replace (with a malicious sensor) biosensors within a network. To compromise means to be able to physically change the workings of a biosensor to make it behave maliciously. We assume that once a biosensor is compromised any secrets (like keys) stored within it are also lost to the adversary. In our system model, biosensors

are both implanted and wearable. We assume that active adversaries can only compromise or replace a wearable biosensors. Also such a wearable biosensors can only be attacked if it is currently not being worn by the target. The assumptions about active adversaries prevents us from having a network wide secret key for encryption and authentication purposes. Also an active adversary is assumed to compromise the biosensors to use them for gaining health information and not to mount a denial of service (DoS) attack on the network by jamming or causing collisions. Therefore such DoS preventive schemes are not addressed.

From now on, the terms adversary and a malicious entity(node) are used interchangeably.

3. Trust Assumptions

In this section we first define the trust assumptions for our system. Trust between two entities is defined as the belief of one entity on the other to work correctly. We make the following trust assumptions:

- The base-station is trusted by all the nodes in the biosensor network.
- The base-station is assumed to never be compromised and execute its protocol exactly.
- Biosensors (cluster head and sensor node) in the network do not trust each other because wearable biosensors can be compromised. Therefore we refrain from having a network wide cryptographic key.
- The wireless communication channel, it is not trusted. Therefore for any communication to take place within this system model proper data and source authentication

mechanisms are necessary.

4. Security Requirements

We now describe some of the security requirements that our protocols need to meet:

- **Confidentiality**: The communication between the source and destination node needs to be a secret. An eavesdropper should not be able to extract the contents of a confidential message.
- **Authenticity**: This ensures that any receiver node is sure that it is communicating with a legitimate sender and that the data it receives is from the sender it is listening to.
- **Integrity**: The messages communicated during the course of our protocol should not be tamper-able by a malicious entity without being detected.

CHAPTER 3

PROBLEM STATEMENT

As mentioned above, for securing biosensor communication, we need to secure the initialization phase as well as the data communication phase. The securing of the former ensures that no malicious entity has access to the network, from then on securing data communication is a simple extension. Secure topology formation is an essential task of the initialization phase. In this thesis we assume that biosensors organize themselves into a cluster topology.

Cluster formation is nothing but establishing a relationship between the cluster heads and sensor nodes in order to facilitate energy-efficient communication. By joining a cluster, a sensor node establishes a relationship between itself and the cluster head of the cluster to which it was assigned. Cluster formation protocols facilitate the establishment of this relationship. We begin this chapter by first describing the traditional cluster formation protocol, the security flaws associated with it and a overview of what needs to be done to correct the flaws.

1. Traditional Cluster Formation

Traditionally, clusters are formed, in a distributed manner, around a set of elected cluster heads. Let W be the set of all biosensors in a network. Let M be the set of elected cluster heads in the network, we define N to be a set of all biosensors, such that $N = W - M$. We assume the presence of a broadcasting schedule which all the biosensors know and their clocks have perfect synchronization. The cluster formation process takes place in three steps:

1. **Broadcast Solicitation:** Each cluster head $p \in M$, broadcasts a solicitation beacon which contains its id and other control information. There are a maximum of $|M|$ solicitation beacons broadcasted in this step.

2. **Choosing Cluster Head:** Each of the $|M|$ solicitation beacons is received by i sensor nodes, where $0 \leq i \leq |N|$. Each sensor node $q \in N$, which has received at least one solicitation, then chooses its cluster head $j \in M$, such that, out of all the beacons received by q, $S = s_1, s_2, ...$, where s_k is the signal strength of the solicitation beacon from cluster head k, $max(S) = s_j$

3. **Transmitting Reply:** Each sensor node $q \in N$, then transmits a reply message to the chosen cluster head, thereby joining its cluster.

An inherent problem with this non-secure protocol is that:

- In this protocol, sensor nodes assumes that only the elected cluster heads broadcast the solicitation beacon and that each elected cluster head is trustworthy. Therefore when it chooses a cluster head as the one whose solicitation beacon signal strength was the highest, it does not know if it is joining a cluster whose cluster head is malicious.

- Similarly, a cluster head also assumes that the reply it received, in response to its solicitation beacon, was from a sensor node which is totally trustworthy.

2. List of Possible Attacks

The problems associated with the non-secure traditional cluster formation protocol makes it prone to many security flaws. We have listed some of the major attacks that are possible on the cluster formation protocol which help a malicious entity infiltrate the biosensor network and give complete control of the network to them.

- **HELLO Flood Attack :** This attack was introduced in [27]. Generally speaking, in many routing protocols, nodes have to broadcast a HELLO packet to announce themselves to their neighbors. A malicious node with the capability can broadcast such HELLO packets at a large enough transmission power to convince every node in the network that it is the closest neighbor and packets can be forwarded to it.

  A similar scenario can happen here, where an adversary, posing as a cluster head, can broadcast a solicitation beacon at such a high transmission power, that all the sensor nodes in the network will choose it as their cluster head. Being the cluster head for all the sensor nodes in the network, it will receive all the data that needs to be sent to the base-station, thereby having complete control over the network. [1]

- **Sinkhole Attack :** In a sinkhole attack [27][36], the adversary's goal is to lure as much traffic generated within the network as possible toward itself, thus creating a sinkhole (black-hole) with the adversary in the center of it. The idea is to make the

[1] HELLO Flood attack is in essence identity spoofing of a cluster head to infiltrate the network. A malicious entity can also spoof the identity of a sensor node and infiltrate the network by just replying to the solicitation beacon of a regular cluster head.

adversary especially attractive to the surrounding nodes with respect to the routing algorithm.

In our case, the traditional cluster formation scheme is prone to HELLO Flood attack as presented above. This attack makes the adversary the cluster head for a large portion if not all the network. This in turn makes the adversary a sinkhole, which can control all the data passing from the biosensor network toward the base-station as all the data has to pass through it.

- **Sybil Attack :** A separate class of attacks that are possible to mount of traditional cluster formation protocol is the Sybil attack. In this attack a malicious node presents multiple identities to other nodes in the network [27].

  In our case, Sybil attacks can be used to facilitate HELLO Flood attack and sinkhole attacks in special case of traditional cluster formation protocol. It is mentioned before that the cluster head has to be rotated over a period of time to average energy over the entire network. In many cases, sensor nodes are programmed not to choose the same biosensor as its cluster head for two consecutive rounds. In such cases, a malicious entity mounting a Sybil attack can change its identity and again mount a HELLO Flood attack by broadcasting a stronger solicitation beacon.

- **Physical Compromise :** The attacks mentioned above can be mounted by a passive adversary without coming in physical contact with the network, by using a sophisticated hardware. Another way of taking control of the network during the traditional cluster formation protocol is by node compromise.

  If an active adversary is able to get hold of a elected cluster head from the network before the cluster formation process starts, it can then force it to broadcast solicitation

beacons at a higher signal strength (possibly by updating the hardware) or replacing it a node of higher capability to mount a HELLO Flood attack leading to sinkhole and selective forwarding attacks. If a cluster head is not available, an active adversary can compromise a sensor node and become part of the network.

The attacks listed above were used to infiltrate the network during the initialization phase. A communication system without security can lead to further attacks during the data communication phase, some of these attacks build on the attacks mounted in the initialization phase while others can be mounted independently. The possible attacks in the data communication phase when biosensors reply to queries of base-station are:

- **Selective Forwarding Attack :** Multihop networks are formed based on the assumption that participating nodes will forward the data received by them. In selective forwarding attack a participating node can refuse to forward data received by them, drop them or tamper with the data before forwarding. A sinkhole is a perfect place for this attack to be mounted as most if not all nodes within a network forward their data to the sinkhole [27].

  Thus once a malicious node forms a cluster head for all the sensor nodes in the network, either by node compromise, a Sybil attack or HELLO flood attack, during the data communication phase, it can collect data from all the sensor nodes in the network and then forward data from only a few sensor nodes to the base-station, provide wrong aggregation values, modify or drop the data sent by the sensor nodes. Such a sinkhole can reliably forward data from certain sensor nodes and limit suspicion of its wrong doing. It can be seen that the HELLO Flood attacks creates an environment for sinkhole attack to be mounted, which itself mounts a selective forwarding attack.

- **Altered or Spoofed Data :** If a malicious entity infiltrates the network in the initialization phase as a sensor node (either by responding to a solicitation beacon (passive adversary) or by physically compromising a sensor node (active adversary), during the data communication phase, it can forward bogus messages to the base-station. As the malicious node is receiving data transmitted by all the other legitimate sensors, due to lack of data confidentiality, the data sent out by the malicious node can be tuned in such that the base-station is forced to make a wrong decisions.

- **Identity Spoofing :** A malicious entity can pose as a sensor node (or cluster head) and send bogus messages to the base-station during the data communication phase, without doing anything during the initialization phase. All it has to do is to reply to a query by pretending to a legitimate biosensor. With no security primitives in place, this is as easy as changing the node id in the packet. A malicious entity can even spoof the identity of a base-station and transmit frivolous queries, in response to which the biosensors will reply and expend their valuable battery resources. The unnecessary activity in a biosensor can also cause heating of nearby tissue for a sustained period of time.

3. Design Goals

Malicious entities are able to mount such attacks on the traditional cluster formation protocol because of the lack of authentication between the cluster heads and sensor nodes. If we are able to remove the security flaws in the cluster formation protocol, we can prevent the attacks during the data communication phase. We therefore need to meet the following **goals** for securing the cluster formation of network communication:

1. For each message received by the base-station, it should be able to ensure that message originated from a biosensor $q \in N$ and forwarded by a $p \in M$.

2. For message received by the cluster head or sensor node, both should be able to ensure that message originated from a biosensor $q \in N$ or base-station in case of a cluster head and from a biosensor $p \in M$ or base-station in the case of a sensor node respectively.

3. For each sensor node $q \in N$, choose a cluster whose cluster head p, where $p \in M$, is the closest cluster head to the node q.

4. For each sensor node $q \in N$, choose a cluster whose cluster head p, where $p \in M$, is the closest cluster head to the node q and the chosen cluster head p is not compromised or replaced by a malicious biosensor.

Here N is the set of sensor nodes and M is the set of legitimate cluster heads.

4. Constraints in Securing the Protocol

Security adds a sizable overhead to any system. For a biosensor network, with each biosensor having limited capability it can be prohibitive in terms of communication costs, memory usage or processing need. In developing protocols for securing cluster formation we have tried to minimize the overhead by making it work under certain constraints. These constraints are:

- No pre-deployment of network-wide key. As individual biosensors can be compromised it is not secure.

- Minimize the number of pre-deployed keys. As our effective problem here is authentication, one way of achieving it would be to have all biosensors pre-deploy keys so as to share keys with all the nodes within the network. As the number of biosensors are large in a biosensor network, such an idea is not practical.

- As the base-station is the controlling entity for the biosensor network, individual biosensors will share keys with it, we have to minimize the number of keys stored at the base station and find easy ways to retrieve the appropriate key to perform cryptographic functions.

- We have to minimize the key distribution overhead for ensuring secure cluster formation as biosensors have limited capabilities.

It is to be noted that, cluster topology is formed essentially to average out the communication cost over the entire biosensor network. Energy-efficiency is thus the primary aim of cluster formation. Securing this process however adds a overhead to it and its energy-efficiency is reduced. Therefore there exists a trade-off between security and cluster topology formation, the higher the security guarantee needed the more expensive the communication during topology formation will become.

CHAPTER 4

RELATED WORK

In this thesis, we are trying to solve the problem of secure cluster formation in a biosensor network. Specifically we are trying to prevent the HELLO Flood attack, sinkhole attack, selective forwarding attacks, Sybil attack, and adverse effects of node compromise [27] during the cluster formation process. Each of these attacks which results in a malicious node getting complete control of the sensor network by forming a sinkhole. The problem occurs due to the distributed nature of the cluster formation process without any authentication. We therefore looked into work done in the area of secure communication, key management and cluster formation in the course of this thesis. We present some of the work available in the literature which is related to our work and from which we use ideas to solve our problem.

Security is an essential need for biosensor networks, but little attention has been given to security in biosensor network communication. In [19], which was the preliminary work in this area, we developed a scheme that uses biometrics from the body to facilitate secure data communication without any previous distribution of a secret between the communicating sensors. In this scheme, the data is first encrypted using an arbitrary key, say k. This key is then xor-ed with a biometric value, $BioKey$, which is measured by the sensors, to generate a key commitment. The key commitment is called γ, where $\gamma = BioKey \bigoplus k$. The

biometric value can be different body parameters like heart-rate, body temperature, specific hormone level in the blood etc. The encrypted data and key commitment are then sent to the receiver, which uses its measured biometric value, decommits the key ($k = BioKey \bigoplus \gamma$) and decrypts the data. It is possible that the BioKey at the receiver is not exactly the same as the sender. In such cases, we use an error correction algorithm, like majority encoding to correct the receiver's $BioKey$ and correctly decommit k. This idea is based on the concept of fuzzy commitment scheme presented in [25]. It incorporates error correction codes in order to protect or encrypt data. There are two phases in this scheme namely the *commit phase* and *decommit phase*. In the commit phase the entity to be protected (say) c is committed with x as proof using $F_{com}()$:

$$F_{com}(c, x) \quad = \quad (h(c) \parallel \delta)$$

where $\delta = x \oplus c$ ($\oplus$ is the bitwise XOR operation) and $h()$ is a hash function. The receiver receives $h(c) \parallel \delta$ from the sender. Now the receiver decommits c using $F_{dec}(h(c) \parallel \delta, x')$ as follows. It computes $c' = f(x' \oplus \delta)$, where x' is variant version of proof x available to the receiver and f is an error correction function. Now the receiver checks if $h(c') = h(c)$. If they are equal then the receiver will go ahead and use c' in place of c. [25].

Example: This is a simple example to explain the above scheme. Consider an error correcting code with code set $C = \{00000, 11111\}^2$, and f is a majority decoding function which decodes five bits at a time. Thus error to the extent of two bits can be corrected. Now choose $c = \{00000 \quad 11111\}$ from C. Suppose the proof for committing c, x=$\{01010\ 10101\}$. Therefore $\delta = \{01010 \quad 01010\}$ and $F_{com}(c, x) = (\alpha, \delta) = (h(00000 \quad 11111), 01010 \quad 01010)$. Suppose the receiver has the commit proof corrupted in 2 bits i.e $x' = \{11010 \quad 11101\}$. Now the decommit operation computes $f(x' \oplus \delta) = f(10000 \quad 10111) = \{00000 \quad 11111\} = \alpha$. Hence the decommit operation is successful [25].

Our scheme of using biometrics for exchanging encrypted data and using error correction to correct biometrics at the receiver end, though useful has another inherent problem associated with it. The problem is that the biometric value is inherently non-random and therefore brute force decommitment of the key is easy for any adversary.

In contrast to biosensors, generic sensor networks have been explored in some depth over the last couple of years. Lot of work has been done in securing sensor network communication and many innovative security protocols have been proposed in the literature. We present some of the work which is more relevant to this thesis. One of the first papers in the area of secure sensor network communication is the SPINS protocols by Perrig et al. [32]. SPINS protocol has two main parts: a secure node-to-node communication protocol and an authenticated broadcast protocol using key-chains. It assumed that each pair of communicating nodes shared a master key which was used for communication. The work however did not address the issues of secure topology formation or multicasting. Any secure communication protocol requires that keys be first distributed among the communicating entities. One of the important works in key-distribution for sensor networks in presented in [20], where a large number of keys are pre-deployed in each sensor node. Nodes which want to communicate, can do so only if they both have a shared pre-deployed key. They then present a probabilistic view as to how many keys need to be pre-deployed for nodes to communicate with each other. This though a novel idea requires large memory resources to store the pre-deployed keys. We therefore need a scheme where the number of keys, if pre-deployed, are minimum. Other works which are similar to this including randomized key assignment [33] and random subset key assignment [29] have been proposed but do not cater to the needs of the problem we are trying to solve here.

As sensor networks have limited capabilities, security attacks that can be mounted

on them are numerous. Two of the important works which re-count the security attacks possible on sensor networks are [27] and [36]. The former talks about secure routing in sensor networks and presents many attacks that are possible in ensuring secure routing of data through a sensor network. It contends that protocols developed till now have been designed for energy efficiency and not with security in mind and this makes them vulnerable to many attacks. It also introduces the HELLO Flood attack that is possible to mount on cluster formation protocols leading to sinkholes. Other attacks like wormholes, Sybil, selective forwarding have also been mentioned. The second paper describes the denial of service (DoS) attacks that are possible in a sensor network at all layers of the protocol stack. Many of the attacks like sinkhole formation, jamming are common in both these papers.

A lot of work has been done on cluster topology in the realm of wireless networks. In [23] [16] [15] [13] [18] [11] [14] [10] [30] [28] a variety of cluster formation schemes are presented for different types of wireless networks. In all these instances the stress is on the development of algorithms for selecting the cluster head. Once the cluster head is selected, the clusters are formed, distributively, by signal strength of the cluster head solicitation beacon, without proper authentication leading to the problem of sinkhole formation as mentioned in the last chapter. In [21], a cluster formation scheme is presented that does not utilize cluster heads at all, but here we do not consider such cases. It is important to mention that in [22], a centralized cluster formation protocol has been described, which uses the base-station to decide the cluster for each node. It however does not do so securely and is prone to sinkhole formation.

As far as we can say, this is the first work in the area of secure cluster formation in the domain of biosensor networks.

CHAPTER 5

PROPOSED SOLUTION

In this section we present our solution to secure cluster formation problem in the context of a biosensor network. In order to facilitate secure cluster formation, we assume that each biosensor shares a secret key with the base-station before the cluster formation process starts. We therefore first describe the key pre-deployment scheme, followed by the notion of using biometrics for authentication before going onto the description of the protocols.

1. Key Pre-deployment

Sinkhole formation occurs mainly because of the lack of authentication between the cluster head and the sensor nodes as they do not have a shared secret between them. Ideally, to avoid this problem a cluster head would have to share keys with all the sensor nodes, to authenticate themselves (because it does not know to begin with which sensor node will join its cluster). For a large network this can have very high memory requirements. Therefore, in our protocol we want to minimize the number of keys stored per node.

We assume that each biosensor shares a unique pair-wise secret key, NSK, with the base-station. If one node is compromised, then only its key is lost, while the other keys

are still secure. The NSK is pre-deployed into each node at the time of manufacturing, which is assumed to be a secure process. The NSK is assumed to be generated using a cryptographically secure random number generator. The base-station does not store each NSK, but derives them, on the fly given, a cryptographic one-way hash function H, a secret value *secret*, the base-station's id $BSID$ and the concerned node's id. For a node with id $N1$, the secret key will be: $NSK_{N1} = H(secret, N1, BSID)$. A similar scheme is mentioned in [17]. The advantages of using this scheme is that it decreases the amount of memory used at the base-station in storing individual secret keys for each node in the network. Also, if new sensors need to be added to the network the base-station can derive the NSK of a new node on the fly and need not be updated.

The biosensors and the base-station, do not use their NSK directly, they derive keys from it, which will actually be used to secure the transaction during the cluster formation. We call the NSK as the *base key* and the keys derived from it as *derived keys*. We use the derived keys to prevent the loss of the base key even if the derived keys is guessed by an adversary which is monitoring the traffic (by re-hashing the base key).We have 4 derived keys (2 encryption and 2 Message Authentication Code (MAC) keys), one for each direction of communication as the names suggest in the table. Using two MAC and encryption keys removes the need for putting the node ids in the MAC (which is considered necessary to prevent replay attacks [12]). We do not use the same keys for MAC and encryption in-order to prevent potential interactions between the primitives that might introduce a weakness [32]. The Table 1, shows the computation of derived keys from the base key.(H is a one-way hash function, (BS is the base-station, N is the node(cluster head or sensor node):

Table 1. Computation of Derived (Encryption and MAC) Keys from Base Key

Encryption Keys	MAC Keys
$K_{N-BS} = H(NSK_N, 1)$	$K'_{N-BS} = H(NSK_N, 2)$
$K_{BS-N} = H(NSK_N, 3)$	$K'_{BS-N} = H(NSK_N, 4)$

2. Biometrics and Authentication

Biosensors, being constrained in capabilities, can use only symmetric key cryptography for securing their communication. In order to facilitate this, there needs to be a shared secret key between the communicating biosensors. As the biosensors are placed in and around the human body, they can use their environment to securely distribute keys to other biosensors. Here, the environment is the human body and the technique used for secure key distribution is using *biometrics*.

Traditionally, the term *biometrics* has been used in two contexts:

- Technology used to identify an individual based on analysis of physiological and behavioral traits like fingerprints, face recognition, gait recognition and iris scanning.
- Field of development of statistical and analytical model for data analysis in biological sciences.

In our case, we use the term biometrics in a third context, the use of human body parameters for generating cryptographic material. Any time-varying, measurable human body parameter can be considered (heart rate). We refer to these parameters as biometrics and use them to facilitate secure communication for biosensors. As the biometric values are similar through out the body, and are difficult to measure without physical contact with the body, we have a shared secret available to the biosensors from the environment. Even though biometrics can be used in many different ways, we use them for authentication

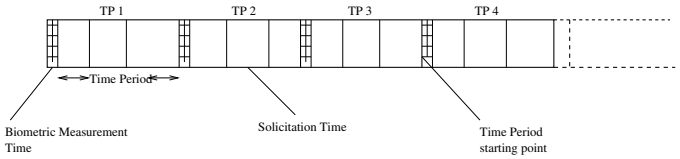

Figure 3. **The Biometric Measurement Schedule**

purposes. We now present a general overview of how biometric authentication works. In the next section when we describe our protocol, the details of how biometric authentication is actually used is presented.

Biometric authentication is a process, where human body parameters (biometric) are used to authenticate data exchanged between two sensors which do not share any secret. This is a completely on-line process and it removes the need for any previous key distribution overhead. We assume the following about the biometric authentication:

- There is a specific pre-defined biometric (like blood glucose level or heart rate) that all biosensors can measure [1].
- There is a well known biometric measurement schedule that all the biosensors follow.
- The clocks on all the biosensors are synchronized.
- The biometric used cannot be measured without at least a physical contact with the patient's body.

A message is generally authenticated by appending a message authentication code (MAC) to it, the computation of which is done by a key shared between the sender and the receiver. Biometric authentication scheme essentially derives this shared MAC key on the fly without any pre-established keys. Biometric authentication works as follows: First,

[1] Already work is being done in developing multipurpose implantable biosensors which can sense and analyze different types of data. Already sensors have been developed which non-invasively monitor glucose levels along with other parameters like blood oxygen level [3]

the sender and receiver measure a specific, biometric value called $BioKey_s$ and $BioKey_r$ respectively, at a given instance of time based on an established schedule. The sender then generates the data it wants to send and computes a *certificate* for it. A certificate is defined as $Cert[data] = \mathbf{MAC}(RandKey, data), \gamma$, where $\gamma = RandKey \bigoplus BioKey_s$, $\bigoplus$ is bitwise-xor function, $RandKey$ is the certificate key generated on the fly using a pseudo-random number generator. The certificate is nothing but a MAC which uses a random key generated by the sender. In order to communicate this key to the receiver, we *commit* (xor) it using the biometric value measured at the sender and append it along with the message: $Sender \rightarrow Receiver : data, Cert[data]$. At the receiving end, the receiver would already have measured the biometric value $BioKey_r$ at the same time as the sender. To authenticate the received data, the receiver first *de-commits* (xor) the MAC key generated by the sender by doing $RandKey' = \gamma \bigoplus BioKey_r$. It then uses the $RandKey'$ to compute the certificate and compare it with the one received with the certificate. If the values (computed and received) match, then the message is authenticated.

It is important to note that the value of $BioKey_s$ and $BioKey_r$ may not be exactly the same, because the value of biometric varies based upon the location of measurement [19]. Hence at the receiver end, the value of de-committed $RandKey'$ may not equal the committed $RandKey$ at the sender. This problem was presented in [25] and they used an error correction function (using majority decoding), as mentioned in the related work, to correct the problem, and we assume as much here.

Therefore the receiver on receiving a message from the sender, decommits the certificate key ($RandKey'$) from γ using its version of the biometric value $BioKey_r$. It then uses an error correction function f on $RandKey'$ to get the sender's certificate key $RandKey$-$f(RandKey') = RandKey$ which can then be used to compute and validate the certificate

correctly.

Biometrics have another important property that their values are not inherently random. The lack of randomness means that the values of biometrics can be easily guessed by brute-force. To prevent brute-force attacks on biometric values, which will result in loss of the certificate key, we divide the biometric measurement schedule into time-periods see Figure 3. Each time-period has two parts, the Biometric Measurement Time (BMT) and Solicitation Time (ST). There can be one or more STs in a time-period, but only one BMT. Each sender node is assigned to a ST in a specific time-period when it will send its data. Figure 3 shows as an example, nodes 1,3,4 are assigned to ST 1-3 in time-period1 (TP1). These three will measure their biometrics values in the BMT of TP1. The receiver which will receive their messages from nodes 1,3,4 also measure their biometric values in BMT of TP1. At the end of the BMT, the sender and receiver nodes sleep till their allotted ST within the time-period, when they send and receive data respectively. If a receiver is to get data from multiple senders, then it is awake for all the ST periods when it is supposed to receive data from the sender. Optimizations can be done, to minimize the start-up costs here, just in case a receiver node is scheduled in alternate STs within a time-period. But if the solicitations being sent are long enough, this may not be needed. The length of each time-periods is a design issue and depends upon the adversary's capability to brute force the biometric value and also on the precision of biometric measurement at each biosensor. The length of the time-period should be less than time taken for any adversary to brute-force break the biometric at a given precision. Other schemes such as combining multiple biometrics and using multiple readings of the same biometric can be used along with biometric measurement schedule to improve chances of detection against a brute-force attacker. The notations used in the description of the protocols is given in tables below.

The Table 2 pertains to the centralized cluster formation protocol only, Table 3 to the distributed cluster formation only and the Table 4 has notations which are common to both protocols .

Table 2. Notations Specific to the Centralized Cluster Formation Protocol

Specific Centralized Protocol Variables	**Description**
$BSReplyMsg$	Identity of a message received at sensor node from BS
$NodeMsg$	Identity of a message received at CH from sensor node
$CHForwdMsg$	Identity of a message received at BS from CH
K_{CH-N}	Is the key which will be shared between sensor node and its CH
K_{Random}	Stores a random key used for the biometric certificate
$chosenCH[X]$	Is an array of structures (used for storing the CH for a node) which has the following elements: $Chid$, $Nonce$,Enc_K_{CH-N}, $Counter$. The length of the array is same as maximum number of sensor nodes in the network, for node 1, data is written in chosenCH[1] and so on.
$BSReplyCounter$	Counter, random number used for encryption in BS - N message

Table 3. Notations Specific to the Distributed Cluster Formation Protocol

Specific Distributed Protocol Variables	**Description**
$NodeReplyMsg$	Reply message received at CH from a node which joins its cluster
$CHSolicitMsg$	Solicitation message sent by the CH received at a sensor node
K_{temp}	Is the random key which will be used to compute the message authentication code of a sensor nodes reply
$K_{DistRandom}$	Stores a random key used to derive the keys (encryption and message authentication code key) in the CH solicitation
$K_{encrypt}$	Derived encryption key in the CH solicitation
K_{mac}	Derived message authentication code key in the CH solicitation
$distchosenCH$	Is a structure, which stores for a node, the id of its CH, value varies as more solicitations are received from other CHs. Its elements are: $Chid$,s$Nonce$, Enc_K_{temp}, $Enc\_K_{encrypt}$,$Counter$
$delta$	Variable stores the committed $K_{encrypt}$

The next section goes into the details of the protocols.

3. Centralized Secure Cluster Formation Protocol

We first describe a centralized secure cluster formation protocol, which uses the base-station to decide the clusters in the network. It consist of three phases:

Table 4. Common Notations the Cluster Formation Protocols

Variables, Functions and Events used in both Protocols	**Description**
BS, CH, N	Base station, cluster head and sensor node respectively
$SignalStrength$	Variable stores signal strength value as measured by the CH
$Ready\_to\_Receive$	Is the flag, which when set, keeps the node in a receiving mode
$BioKey$	Biometric used in the authentication
$Timer, Timer1$	Are timer variables whose value is set by a function
$ID, Chid$	Identification number of a sensor node and CH respectively
$Nonce$	Random value used for maintaining freshness of a transaction
$Cert$	Biometric Certificate
$Msg.X$	If Msg is the received message, then Msg.X is the variable
$Gamma$	Variable stores the committed random key of the certificate
$Cert\_MAC$	Variable stores the message authentication code part of the certificate
Enc_X	Is the encrypted Variable X
$computedMAC, computedCert$	Are the message authentication code and certificate values computed at a node to compare it with the value received with the message for authentication
$send(dest, data)$	Is the function to send $data$ to the node $dest$, if $*$ used then it is broadcast
$generateRandom()$	generates random number
f	Error Correction function used to correct the value of the decommited secret using biometric value on the receiver side.
$HMAC(K, P\|Q)$	The message authentication code (MAC) function, where K is the key, P and Q are variables used in the computation
$E_{<K,Cntr>}(data)$	The encryption function. It is done in a output feedback mode (OFB), where key K first encrypts the counter $cntr$, the output of which is xored with $data$, to get the encrypted value. This scheme is useful because with only an encryption function both encryption and description can be done.
$get\_ST\_TimerValue()$	Is the value to which timer is set after BMT, for broadcasting. In other words sets the ST value for the node
$Timer.Fired$	Is the event of timer being fired event. Action is taken when this event occurs like calling a function

Phase 1: First, each sensor node broadcasts a solicitation beacon with its id, a nonce (to maintain freshness) and a MAC. The MAC is to authenticate this message to the base-station and uses the appropriate derived key, namely K'_{N-BS}. We also append a certificate along with this message which will authenticate the message to the cluster head to prevent an adversary from sending bogus messages to relay to the base-station, thereby wasting energy at the cluster head. Appropriate biometric schedule are created where each sensor node is assigned a ST slot in a time-period. All sensor nodes are initially sleeping. When

the BMT of the assigned time-period of a node starts, it measures the biometric. At the end of the BMT the node again goes back to sleep till its allotted ST starts. The cluster heads being the receivers measure biometric in each time-period and keep their transceiver on for all time-periods. In summary:

$\mathbf{Node_k} \rightarrow \mathbf{ALL}$:$NodeID_k, Nonce_{Node_k}, HMAC(K'_{N_k-BS}, Nonce_{Node_k}), Cert[NodeID_k, Nonce_{Node_k}]$

Here : k is the index of all the nodes that are sensor nodes (non-cluster heads). It should be seen that the certificate data item is the sensor node ID and the nonce. We marry the certificate to the message being sent to prevent a malicious node from attaching this certificate to its own bogus solicitation message. Also as the nonce varies with time and so does the biometric key, the certificate cannot be replayed at a later time.

Phase 2: The cluster heads which receive a solicitation sent in phase 1 by the sensor nodes, above a signal strength threshold, will validate the certificate (in the solicitation) using their version of the biometric value and necessary error correction function. If the certificate is valid, they then relay this message to the base-station appending their id, the signal strength (SS) at which they received the solicitation, an encrypted random number K_{CH-N} and a MAC to authenticate these to the base-station. This MAC is the one in addition to the MAC appended by the sensor node in its solicitation. As the sensor node MAC cannot be authenticated by the cluster head (as it lacks the requisite key), it is left alone. This random number will act as a shared secret between the cluster head and sensor node if it joins its cluster, thereby eliminating the need for weak biometrics in the future. The MAC attached by the cluster head (uses the key K'_{CH-BS}) will authenticate the cluster head to the base-station, thus ensuring that only a legitimate cluster head has relayed the

solicitation beacon. In summary:

$\mathbf{CH_p \rightarrow BS}$:$NodeID_k, Nonce_{Node_k}, HMAC(K'_{N_k-BS}, Nonce_{Node_k}), Chid_p, SS, E_{<K_{CH_p-BS}, Counter>}$ $(K_{CH_p-N}), Counter, HMAC(K'_{CH_p-BS}, MSG|SS|Counter)$

Here : p is the index of all the cluster heads that receive sensor node solicitation. $MSG = E_{<K_{CH_p-BS}-Counter>}(K_{CH_p-N})$.

Phase 3: The base-station will potentially receive each sensor node's solicitation beacon from multiple cluster heads. For each of the solicitation received, it will verify the two MACs - the first MAC is attached by the sensor node which broadcasted the solicitation beacon, the second MAC is appended by the cluster head which forwarded the beacon to the base-station. Once all the copies of a solicitation for a given sensor node are received and verified, the base-station checks the SS value for each copy of the solicitation received for a sensor node and picks the cluster head, whose SS value is the largest as the cluster head for the sensor node. It then sends the identity of the chosen cluster head, and transmits encrypted - the random number K_{CH-N} sent by the chosen cluster head when it was forwarding the sensor node's solicitation. A MAC is also appended to the message using the key K'_{BS-N}. The base-station thus assigns a sensor node to the cluster of the closest cluster head, assuming there is no interference or multipath propagation. In summary:

$\mathbf{BS \rightarrow Node_k}$:$Chid_z, E_{<K_{BS-N_k}, Cntr>}(K_{CH_z-N}), Cntr, HMAC(K'_{BS-N_k}, Chid_z|Counter|$ $Nonce_{Node_k}|MSG)$

Here: k is index of all the sensor node, $MSG = E_{<K_{BS-N_k}, Cntr>}(K_{CH_z-N})$, $Cntr$ is the counter value used for encryption and z is the index of chosen cluster head. The base-

station need not tell the cluster head about its cluster members now. When the sensor nodes communicate with the cluster-head, they will append a MAC computed using the key K_{temp}. Only the cluster head which assigned the key can authenticate the message and therefore come to know which node belongs to its cluster.

We can see that a malicious entity cannot pretend to be legitimate cluster head or a sensor node, to the base-station because it will not have the requisite keys to compute a valid MAC while relaying the solicitation beacons or while generating a beacon to join a cluster. Similarly, a cluster head is sure about the identity of the sensor node which whose solicitation beacon it receives by using biometric authentication and as the base-station uses the shared key in phase of the protocol to the sensor nodes, base-station as a sender is also authenticated. Therefore by asking the base-station and using appropriate authentication, legitimate sensor nodes are assigned into clusters of legitimate cluster heads, thereby meeting the goals 1,3 and 4 as stated in Section 4 of Chapter 3. We will describe how goal 2 is met in Chapter 7 Sections 4 and 5. Refer to algorithms 1, 2, 3 for the algorithms of sensor node, cluster head and the base-station respectively for the centralized secure cluster formation protocol.

4. Distributed Secure Cluster Formation Protocol

We now present the distributed secure cluster formation protocol, where sensor nodes decide which cluster they want to join based on the signal strength of the cluster head solicitation beacon received. It consist of two phases:

Algorithm 1: The sensor node protocol: Centralized Cluster Formation

$Ready\_to\_Receive = FALSE$
if (BMT of SCHEDULED time-period) **then**
  $BioKey = getBiometric()$
  $Timer = get\_ST\_TimerValue()$
end if
if $(Timer.Fired == TRUE)$ **then**
  $ID = getID()$
  $Nonce = generateRandom()$
  $K_{Random} = generateRandom()$
  $MAC = HMAC(K'_{N-BS}(ID|Nonce)$
  $Cert\_MAC = HMAC(K_{Random}(ID|Nonce)$
  $Gamma = BioKey \bigoplus K_{Random}$
  $Cert = Cert\_MAC, Gamma$
  $sendData(*, ID|Nonce|MAC|Cert)$
  $Ready\_to\_Receive = TRUE$
end if
if $(Ready\_to\_Receive == TRUE)$ **then**
  if $(BSReplyMsg.Received == TRUE)$ **then**
    $Chid = BSReplyMsg.CHID$
    $Enc\_K_{CH-N} = BSReplyMsg.Enc\_K_{CH-N}$
    $BSReplyCounter = BSReplyMsg.BSReplyCounter$
    $computedMAC = HMAC(K'_{BS-N}(Chid|Nonce|Enc\_K_{CH-N}|BSReplyCounter)$
    if $(computedMAC == BSReplyMsg.MAC)$ **then**
      $Enc\_BSReplyCounter = E_{K_{BS-N}}(BSReplyCounter)$
      $K_{CH-N} = Enc_BSReplyCounter \bigoplus Enc\_K_{CH-N}$
      Store K_{CH-N} for future use in sensor node - cluster head communication
    else
      Discard the packet
    end if
  end if
end if

Algorithm 2: The cluster head protocol: Centralized Cluster Formation

$Ready\_to\_Receive = TRUE$
if (BMT of ANY time-period) **then**
  $BioKey' = getBiometric()$
end if
if $(Ready\_to\_Receive == TRUE)$ **then**
  if $(NodeMsg.Received == TRUE)$ **then**
    $ID = NodeMsg.ID$
    $Nonce = NodeMsg.Nonce()$
    $MAC = NodeMsg.MAC()$
    $K'_{Random} = Biokey' \bigoplus NodeMsg.Cert.Gamma$
    $K_{Random} = f(K'_{Random})$
    $computedCert = HMAC(K_{Random}(ID|Nonce)$
    if $(computedCert == NodeMsg.Cert.MAC)$ **then**
      $Timer1 = getRandom()$
    else
      Discard the packet
    end if
  end if
end if
if $(Timer1.Fired == TRUE)$ **then**
  $K_{CH-N} = generateRandom()$
  $Counter = generateRandom()$
  $Enc\_Counter = E_{<K_{CH-BS}>}(Counter)$
  $Enc\_K_{CH-N} = Enc_Counter \bigoplus K_{CH-N}$
  $SignalStrength = get\_SignalStrength(Msg)()$
  $Chid = getID()$
  $MAC1 = HMAC(K'_{CH-BS}(Chid|Enc\_K_{CH-N}|SignalStrength|Counter)$
  $sendData(BS, ID|Nonce|MAC|Chid|Enc\_K_{CH-N}|SignalStrength|Counter|MAC1)$
end if

Algorithm 3: The base-station protocol: Centralized Cluster Formation

$Ready\_to\_Receive = TRUE$
if $(Ready\_to\_Receive == TRUE)$ **then**
  if $(CHForwdMsg.Received == TRUE)$ **then**
    $ID = CHForwdMsg.ID$
    $Nonce = CHForwdMsg.Nonce()$
    $Counter = CHForwdMsg.Counter()$
    $SignalStrength = CHForwdMsg.SignalStrength()$
    $Enc\_K_{CH-N} = CHForwdMsg.Enc\_K_{CH-N}()$
    $Chid = CHForwdMsg.Chid$
    $computedMAC = HMAC(K'_{N-BS}(ID|Nonce)$
    $computedMAC1 = HMAC(K'_{CH-BS}(Chid|Enc\_key|SignalStrength|Counter)$
    if $(computedMAC == CHForwdMsg.MAC)$ **then**
      if $(computedMAC1 == CHForwdMsg.MAC1)$ **then**
        if $(chosenCH[ID].signalstr < SignalStrength)$ **then**
          $chosenCH[ID].Chid = Chid$
          $chosenCH[ID].Nonce = Nonce$
          $chosenCH[ID].Enc\_K_{CH-N} = Enc\_K_{CH-N}$
          $chosenCH[ID].Counter = Counter$
          $Timer = getRandom()$
        else
          Do not update structure chosenCH, current CH is farther than stored CH
        end if
      end if
    else
      Discard the packet
    end if
  end if
end if
if $(Timer.Fired == TRUE)$ **then**
  for $(index = 0; index < MAX\_NODEX; index++)$ **do**
    $Enc\_Counter = E_{<K_{CH-BS}>}(chosenCH[index].Counter)$
    $K_{CH-N} = Enc\_Counter \bigoplus chosenCH[index].Enc\_K_{CH-N}$
    $BSReplyCounter = getRandom()$
    $Enc\_BSReplyCounter = E_{<K_{BS-N}>}(BSReplyCounter)$
    $Enc\_K_{CH-N} = Enc\_BSReplyCounter \bigoplus K_{CH-N}$
    $MAC = HMAC(K'_{BS-N}(chosenCH[index].Chid|chosenCH[index].Nonce|$
      $Enc\_K_{CH-N}|BSReplyCounter)$
    $sendData(ID, Chid|Enc\_K_{CH-N}|BSReplyCounter|MAC)$
  end for
end if

Phase 1: Here, the cluster heads first broadcast their solicitation beacons with their id, an encrypted temporary key (called K_{temp} which will act as a shared secret and be used to authenticate the reply) and a nonce along with the biometric certificate. (Again a biometric measurement schedule is created as in the centralized protocol, only this time the senders are the cluster heads and the receivers are sensor nodes.) To encrypt the temporary key, we generate a random key $K_{DistRandom}$ and hash it to generate two keys out of it, K_{mac} and $K_{encrypt}$. $K_{mac} = H(K_{DistRandom}, 1)$ and $K_{encrypt} = H(K_{DistRandom}, 2)$. K_{mac} is used in generating the certificate, while $K_{encrypt}$ is used for encrypting K_{temp}. In summary:

$\mathbf{CH_p} \rightarrow \mathbf{All}: Chid_p, E_{<K_{encrypt}, Counter>}(K_{temp}), Nonce_{CH_p}, Counter, \delta, Cert[Chid, Nonce_{CH_p}, Counter, MSG]$

Here: p is the index of all CHs, The biometric certificate $Cert$ is defined : $MAC(K_{mac}, Chid|Nonce_{CH_p}|Counter|MSG)$,$\gamma$, where $\gamma = BioKey \bigoplus K_{mac}$ and $MSG = E_{<K_{encrypt}, Counter>}(K_{temp})$. The key encrypting K_{temp} also needs to be conveyed to the receiver, and is therefore hidden using the BioKey and stored as δ, $\delta = BioKey \bigoplus K_{encrypt}$

Phase 2: On receiving the solicitation beacons, the sensor nodes first verify the certificate (using their biometric value and error correction function) by first decommiting K_{mac} using its version of the biometric. Once the certificate is verified, the $K_{encrypt}$ is decommited from δ using the measured biometric as presented in section 2 of this chapter. The sensor node then decrypts K_{temp}. The node also measures the signal strength of each solicitation it receives. Once all solicitations have been received, it sends a reply message to the cluster head whose received beacon signal strength value was the largest. To authenticate this message a MAC is computed which uses the key K_{temp}. The cluster head on receiving the

reply, stores the identity of its cluster members. In summary:

$\mathbf{Node_s} \rightarrow \mathbf{CH_d}$:$NodeID_s, HMAC(K_{temp}, NodeID_s|Nonce_{CH_d}|Chid_d|Counter)$

Here: s is the index of all sensor nodes, where d is the index of chosen cluster head.

The use of biometrics has ensured that sensor node knows it is receiving solicitation from a legitimate cluster head while the use of K_{temp} in the reply convinces the cluster head about the identity of the sensor node, thus meeting goal 2 and 3. Goal 1 does not apply here and the fulfillment of goal 2 are based on certain conditions which is explained in Chapter 7 Sections 4 and 5. Refer to algorithms 4, 5 respectively for the description of cluster head, sensor node algorithms for the distributed secure cluster formation protocol.

5. Extension to Secure Cluster Formation Protocols for Data Communication

So far we proposed two protocols for secure cluster formation. These are initialization protocols, which will securely setup the communication topology of the biosensor network. These protocols setup clusters within a biosensor network and also distribute secrets (keys) between the cluster heads and cluster members in-order to authenticate messages exchanged during the data communication phase (K_{CH-N} in centralized protocol and K_{temp} in the distributed protocol). These combined with pair-wise shared secret key between the base-station and biosensors can ensure secure biosensor to base-station communication.

In any sensor network, it is efficient to query a group of sensor nodes rather than individual sensors [24]. Therefore, if the base-station wants to securely send a query.[2] to the

[2]queries have to be sent securely, otherwise a passive adversary, based on the query, can try to make a educated guess about the response

Algorithm 4: The cluster head protocol: Distributed Cluster Formation

$Ready\_to\_Receive = FALSE$
if (BMT of SCHEDULED time-period) **then**
  $BioKey = getBiometric()$
  $Timer = get\_ST\_TimerValue()$
end if
if $(Timer.Fired == TRUE)$ **then**
  $Chid = getID()$
  $Nonce = generateRandom()$
  $K_{temp} = generateRandom()$
  $K_{DistRandom} = generateRandom()$
  $K_{encrypt} = HMAC(K_{DistRandom}, 1)$
  $K_{mac} = HMAC(K_{DistRandom}, 2)$
  $Counter = generateRandom()$
  $Enc\_Counter = E_{K_{encrypt}}(Counter)$
  $Enc\_K_{temp} = Enc\_Counter \bigoplus K_{temp}$
  $delta = BioKey \bigoplus K_{encrypt}$
  $Gamma = BioKey \bigoplus K_{mac}$
  $Cert\_MAC = HMAC(K_{mac}, Chid|Nonce|Counter|Enc\_key)$
  $Cert = Cert\_MAC|Gamma$
  $sendData(*, Chid|Nonce|Counter|Enc\_key|delta|Cert)$
  $Ready\_to\_Receive = TRUE$
end if
if $(Ready\_to\_Receive == TRUE)$ **then**
  if $(NodeReplyMsg.Received == TRUE)$ **then**
    $NodeID = NodeReplyMsg.ID$
    $computedMAC = HMAC(K_{temp}, NodeID|Nonce|Chid|Counter)$
    if $(computedMAC == NodeReplyMsg.MAC)$ **then**
      Designate NodeID as a node in cluster
    else
      Discard the packet
    end if
  end if
end if

Algorithm 5: The sensor node protocol: Distributed Cluster Formation

$Ready\_to\_Receive = TRUE$
if (BMT of ANY time-period) **then**
  $BioKey' = getBiometric()$
end if
if $(Ready\_to\_Receive == TRUE)$ **then**
  if $(CHSolicitMsg.Received == TRUE)$ **then**
    $Chid = CHSolicitMsg.Chid$
    $Nonce = CHSolicitMsg.Nonce$
    $Counter = CHSolicitMsg.Counter$
    $SignalStrength = CHSolicitMsg.SignalStrength$
    $delta = CHSolicitMsg.delta$
    $Enc\_K_{temp} = CHSolicitMsg.Enc\_K_{temp}$
    $K'_{mac} = Biokey' \bigoplus CHSolicitMsg.Cert.Gamma$
    $K_{mac} = f(K'_{mac})$
    $K'_{encrypt} = Biokey' \bigoplus delta$
    $K_{encrypt} = f(K'_{encrypt})$
    $computedCert = HMAC(K_{mac}(Chid|Nonce|Counter|Enc_key)$
    if $(computedCert == CHForwdMsg.Cert.MAC)$ **then**
      if $(distchosenCH.signalstr < SignalStrength)$ **then**
        $distchosenCH.Chid = Chid$
        $distchosenCH.Nonce = Nonce$
        $distchosenCH.Enc\_K_{temp} = Enc\_K_{temp}$
        $distchosenCH.Counter = Counter$
        $distchosenCH.K_{encrypt} = K_{encrypt}$
        $Timer = getRandom()$
      else
        Do not update structure chosenCH, current CH is farther than stored CH
      end if
    else
      Discard the packet
    end if
  end if
end if
if $(Timer.Fired == TRUE)$ **then**
  $Key = distchosenCH.K_{encrypt}$
  $Enc\_CHSolicitCounter = E_{Key}(distchosenCH.Counter)$
  $K_{temp} = Enc_CHSolicitCounter \bigoplus distchosenCH.Enc\_K_{temp}$
  $ID = getID()$
  $MAC = HMAC(K_{temp}, ID|distchosenCH.Nonce|distchosenCH.Chid|$
    $distchosenCH.Counter)$
  $sendMsg(distchosenCH.Chid, ID|MAC)$
end if

biosensors, it would have to unicast it, because it shares a unique key with each biosensor. This is an expensive task and therefore we need to be able to securely multicast queries to biosensors. Secure multicasting can be achieved by distributing keys to biosensors such that each multicast group has a unique key. In the case of biosensors, we can group them according to specific attributes to assign them unique keys. One such attribute could be the biosensor type. Therefore all temperature measuring biosensors will have the same key. We call this key *Attribute Based Key* (ABK). ABK can then be used to derive encryption and MAC keys just with NSK.

Formally: We want to distribute encryption keys among sensors, such that, all nodes which have an attribute $a \in A$, where A is the set of all attributes a sensor can posses, get a corresponding attribute based key K_a. Here for every $i, j \in A, K_i \neq K_j$.

Therefore A base-station can now broadcast an encrypted query using ABK_{BS-N} and compute a MAC using ABK'_{N-BS}. The biosensors can reply by using the keys ABK_{N-BS} and ABK'_{N-BS} for encryption and MAC respectively. The sensor nodes in the biosensor network can use the key they share with the cluster heads to authenticate their reply.

5.1. Attribute Based Keying. Key distribution is a difficult task to manage. One way of distributing these keys would be to securely pre-deploy them. But as we want to minimize such assumptions, we will not consider that option. The other way is to ask the base-station to distribute keys to the nodes. The base-station - which is trustworthy, has higher computation power and knows all about the network, is the ideal entity for distributing keys. The process of key distribution however adds a overhead to the network. The idea is to minimize this overhead as much as possible because of the

limited communication capability available to each sensor. One of the ways of minimizing the overhead due to attribute based keying is by incorporating the key distribution process in the cluster formation process itself.

Centralized cluster formation is ideal for distributing such attribute based keys. Only a small extension is needed. In phase 1, the *sensor nodes also broadcast their attribute information, $a_i \in A$, i is the index of the node* along with their id and a nonce. MACs are calculated accordingly. The cluster head node forwards the message as usual along with the signal strength and an encrypted shared secret K_{CH-N}. The base-station then selects the cluster head for each sensor node and sends it encrypted - its attribute based key (K_{a_i}) based on the value of a_i and K_{CH-N}. Thus each node has a been given a attribute based key which can be used to communicate data and queries securely. Here as the cluster heads are not assigned keys during the cluster formation process, they have to first get their keys separately by directly contacting the base-station. We call this scheme the *Extended Centralized Cluster Formation Protocol.*

The distributed cluster formation process however is more expensive. As the base-station is not involved in the cluster formation process, we cannot directly involve attribute based keying in the cluster formation process. Therefore the cluster heads and the sensor nodes have to individually ask the base station and get their attribute based keys. All such messages can be authenticated using the shared keys and is done outside the cluster formation process. We call this scheme the *Extended Distributed Cluster Formation Protocol.*

The Figure 4 shows a network where attribute based keying has been implemented. It shows the individual key and the attribute based keys. The tuple $< Kx, Ky >$ where x = 1 to N and y = {*,',$...} means that a sensor node has a pre-deployed secret key Kx and

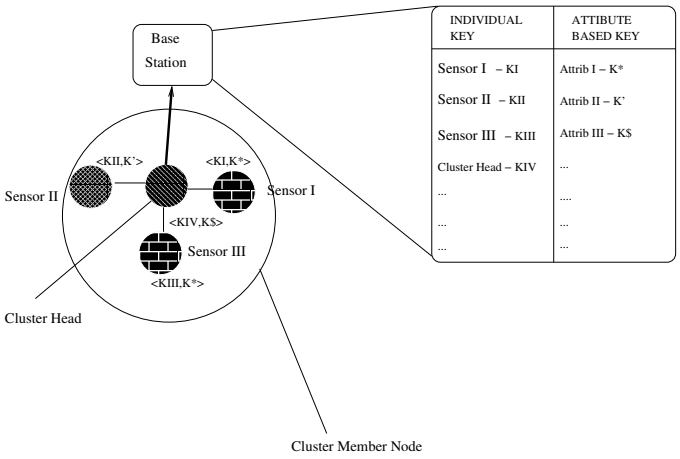

Figure 4. **The Keys used in Biosensor Network**

an attribute based key Ky. The base station maintains a secret key (pre-deployed) for each sensor node and one key for each sensor attribute that it wants to query or receive data from. Nodes I and III have the same attributes and therefore the same attributed based key K^*.

5.2. Rekeying of Biosensors. The protocol presented above, describes an efficient means to form secure cluster and key distribution within a biosensor network. For our system model, we believe that only the attribute based keys needs to be changed over time, because they are used to encrypt data sensed by the biosensors, which is highly repetitive and has limited range. The other keys used, have very short exposure time and are either temporary (biometric keys, certificate keys) or are used to encrypt random key data (derived keys).

It is however important to note, that any n bit keys require some finite time to break. We can leverage this time to efficiently rekey the sensor nodes. Suppose time required to break a given attribute based key is T_{break}.The attribute based key has to be changed before time T_{break}, therefore whenever the base-station distributes an attribute based key to a set

of sensor nodes, it starts a timer for that particular attribute based key. The timer can be set to a value T_{rekey}, such that $T_{rekey} + t_0 < T_{break}$. Here t_0 is the time taken to perform the actual rekeying of the sensor nodes which use the attribute based key. The choice of T_{rekey} should be such that, it minimizes the number of rekeys. The choice of T_{rekey} depends upon the capability of the adversary. Changing the attribute based key entails doing $New\_ABK = H(ABK)$. Clock synchronization information can be sent periodically by the base-station, using a authenticated broadcast scheme as in [32].

The key shared between the sensor nodes and their cluster heads can be changed during cluster head rotation, when a new cluster is formed around a new cluster head.

CHAPTER 6

PERFORMANCE ANALYSIS

In this chapter, we analyze the energy requirements for the cluster formation protocols including the extensions we proposed. We want to be able to quantify the overhead security primitives add to the cluster formation process. We assume a n node network, out of these m nodes are chosen as cluster head. We assume that for centralized cluster formation protocol, $p\%$ (p is a value between 0-1) of cluster heads receive the each sensor node's solicitation beacon. For distributed cluster formation protocol $q\%$ (q is a value between 0-1) of sensor nodes receive each cluster head's solicitation beacon. We assume energy consumption is minuscule for computation. We use the first order radio model given in [23] to calculate the energy consumption of the network.

1. Energy Consumption for Secure Centralized Protocol

To compute the energy consumption for the centralized protocol, we calculate the amount of energy consumed in communication (transmission and reception) by the whole network. We proceed systematically and calculate the energy consumption for each phase of the protocol and finally sum it to get the total energy consumed by the sensor network.

Here the symbol E_{tx} is energy consumed by the sensor circuitry in data transmission, E_{rx} is energy consumed by circuitry in data reception, E_{cr} is energy consumed in amplification of signal before transmission.

In this protocol, first, each of $n-m$ (non cluster head) sensor nodes broadcast a solicitation beacon, each of which is received by $p\%$ of cluster heads. The energy consumed in beacon transmission within the network for a message of size $k1$ will be:

$$E_{Broadcast-Solicit} = (n-m)[E_{tx}k1 + E_{cr}k1s^2]$$

Here s is the maximum transmission range during solicitation. As each of the $n-m$ beacons are received by $p\%$ of cluster heads, the total energy consumed by the cluster heads in receiving the beacons is:

$$E_{Receive-Solicit} = pm(n-m)E_{rx}k1$$

Each cluster head then forwards all the solicitations it receives to the base-station (at an average distance r), in the process consuming energy:

$$E_{Forward-Solicit} = pm(n-m)[E_{tx}k2 + E_{cr}k2r^2]$$

Above $k2$ is the packet size. The base-station on receiving the all the solicitations chooses the cluster head for each node and informs it individually. The energy spent in the reception of this message (which of size $k3$) at the sensor nodes is:

$$E_{Receive-Reply} = (n-m)E_{rx}k3$$

The total energy consumed by the centralized secure cluster formation protocol is:

$E_{centralized} = E_{Broadcast-Solicit} + E_{Receive-Solicit} + E_{Forward-Solicit} + E_{Receive-Reply}$

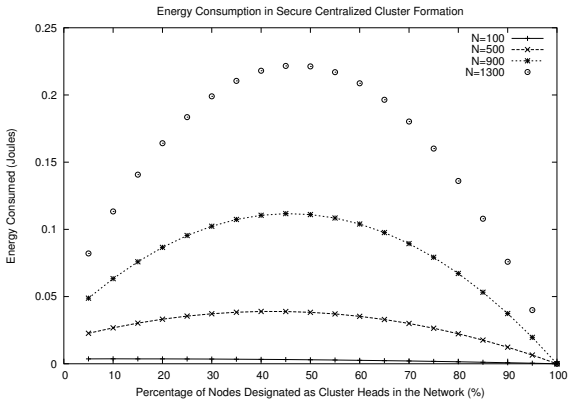

Figure 5. **Energy Consumption in Secure Centralized Cluster Formation Protocols**

2. Energy Consumption for Secure Distributed Protocol

Now we compute the energy consumed by the network in the distributed secure cluster formation protocol. The technique used is the same as the centralized protocol, the summation of energy consumed by the network at each phase of the protocol.

Here, the cluster heads broadcast a solicitation beacon, each of which is received by $q\%$ of sensor nodes. The energy consumed in the beacon transmission of the packet of size $k4$ by m cluster heads will be:

$$E_{Broadcast-Solicit} = m[E_{tx}k4 + E_{cr}k4s^2]$$

When the m cluster heads transmit, each solicitation message is received by $q\%$ of sensor nodes, consuming energy equivalent to:

$$E_{Receive-Solicit} = q(n-m)mE_{rx}k4$$

The sensor nodes on receiving the beacons, decide their cluster head and intimate it (with a packet of size $k5$) consuming the following amount of energy in transmission:

$$E_{Reply} = (n-m)[E_{tx}k5 + E_{cr}k5d^2]$$

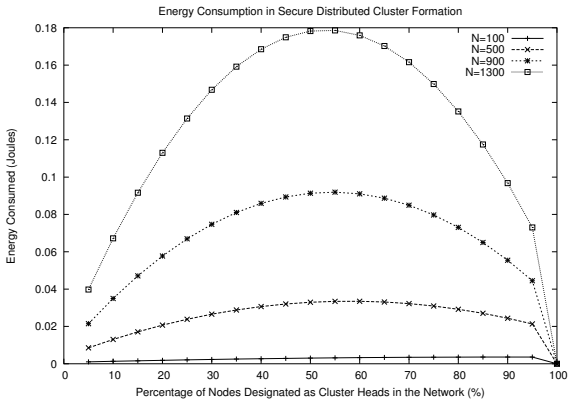

Figure 6. **Energy Consumption in Secure Distributed Cluster Formation Protocols**

Above d is the average inter-node distance. The cluster heads will receive this reply message, the total energy consumed in receiving these $n - m$ reply messages at the cluster heads will be:

$$E_{Receive-Reply} = (n - m)E_{rx}k5$$

The total energy consumed by the distributed secure cluster formation protocol is:

$$E_{distributed} = E_{Broadcast-Solicit} + E_{Receive-Solicit} + E_{Reply} + E_{Receive-Reply}$$

3. Energy Consumption in Centralized and Distributed Cluster Formation Protocol with Extension

If attribute based keying is introduced during the cluster formation process, the energy consumption will vary slightly from the above mentioned equations. For the extended centralized cluster formation protocol, the cluster head nodes will first get their attribute based key consuming, in transmission (with packet size $k6$) and reception (with packet size $k7$):

$$E_{CH-Key-Setup} = m[E_{tx}k6 + E_{cr}k6s^2] + mE_{rx}k7$$

This step will be followed by the cluster formation, which is exactly like the centralized cluster formation protocol, only the packet size is different to accommodate the node attribute data. The energy consumed in this step will be:

$$E_{Cluster-Formation} =$$

$$(n-m)[E_{tx}k8 + E_{cr}k8s^2] + pm(n-m)E_{rx}k8 + pm(n-m)[E_{tx}k9 + E_{cr}k9r^2] + (n-m)E_{rx}k10$$

Therefore total energy consumed by the extended centralized secure cluster formation protocol is:

$$E_{ExtendedCentralized} = E_{CH-Key-Setup} + E_{Cluster-Formation}$$

Similarly, for extended distributed cluster formation protocol, all the cluster heads will first get their attribute based keys just like the centralized version, followed by the cluster formation process consuming same energy as $E_{distributed}$ and then cluster member nodes will contact the base-station through their cluster head to get their attribute based keys, consuming a total energy of:

$$E_{Node-Key-Setup} = (n-m)[E_{tx}(k11 + k13) + E_{cr}(k11d^2 + k13r^2) + E_{rx}(k12 + k14)]$$

Therefore total energy consumed by the extended distributed secure cluster formation protocol is:

$$E_{ExtendedDistributed} = E_{CH-Key-Setup} + E_{distributed} + E_{Node-Key-Setup}$$

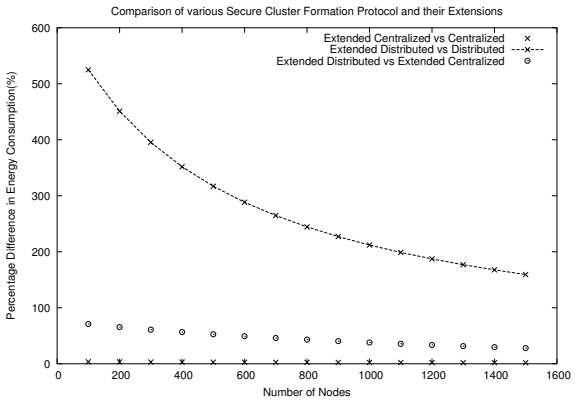

Figure 7. **Comparison of the Secure Cluster Formation Protocols and their Extensions**

Table 5. Parameters Used in Analysis

$NodeID = 8bits$	$Nonce = Counters = 128bits$
$KeySize = 128bits$	$SignalStrength = 16bits$
$E_{cx} = 100pJ/bit/m^2$ [23]	$Etx = Erx = 50nJ/bit$ [23]
$n = 100to1500$	$m = 5\%ofn$
$p = 1\%$	$q = 1\%$
$s = 0.5m, d = 0.1m, r = 0.75m$	$MAC = 64bits$

4. Comparison

We wrote simple C programs to compare the various types of cluster formation protocols described in the previous chapter. The table 5, shows the value of the parameters used in our programs. The results we obtained were as follows:

1. We first compared the centralized, distributed cluster formation protocols (without extension) with the non-secure cluster formation protocol (also distributed in nature). We chose the number of cluster heads to be 5% of total nodes and the size of the network was varied from 100-1500 nodes. We computed the percentage variations in the three cases and found, 1) the secure centralized protocol is energy-intensive compared to the non-secure

protocol - the reasons being the large number of communications to the base-station and the presence of cryptographic primitives. 2) The secure distributed protocol consumed more energy than the non-secure protocol because of the overhead due to cryptographic primitives, 3)Finally, we compared the two secure cluster formation protocols and we found the centralized cluster formation protocol was more expensive than the distributed one due to the large number of long distance communications to the base-station. The Figure 8 shows the graph we plotted.

2. We then ventured to see the behavior of the centralized and distributed cluster formation protocols. We chose four values of network sizes 100,500,900 and 1300 respectively. For each network size, we varied the percentage of cluster heads in the network from 5% to 100%. For both the protocols the energy consumed first increased as the number of cluster heads increased and then it started decreasing as more nodes became cluster heads and lesser nodes needed to be organized into clusters. For 100% both resulted in zero energy consumption because, all the biosensors are cluster-heads and no clustering is needed. This is however only a theoretical result. The Figures 5 and 6 show the behavior of the two protocols.

3. We now compared the performance of the centralized protocol with its extended version, the distributed protocol with its extended version and both the extended versions. The percentage of cluster heads was chosen to be 5% and the number of nodes varied from 100-1500. We computed a percentage variation between in the three cases here and found 1) The extended centralized protocol has a higher overhead than its plain version, because of the key distribution overhead. But the overhead is minimal. 2) The extended distributed

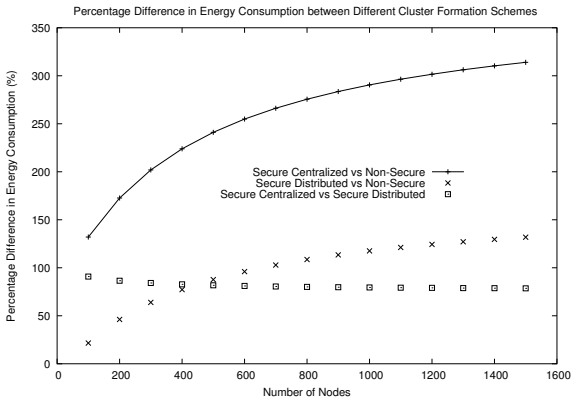

Figure 8. **Comparison of Secure and Non-Secure Cluster Formation Protocols**

protocol has a much higher overhead than its plain version because of the substantial cost in getting the attribute based keys. The process is not integrated with the cluster formation as in the centralized case. 3) Finally we compared the two extensions and found the extended distributed was more expensive than the extended centralized protocol, because after cluster formation base-station needs to be contacted to get the keys (which is like a centralized protocol followed by a distributed one). The result is shown in the Figure 7.

CHAPTER 7

SECURITY ANALYSIS

In this chapter we analyze how our protocols mitigate the attacks that are possible in a non-secure cluster topology based biosensor networks and also some new attacks that can be mounted on the secure protocol.

1. Passive Adversary Spoofing the Identity of a Cluster Head

Vulnerability The non-secure cluster formation protocols using cluster head solicitation beacon strength to form clusters are susceptible to HELLO Flood attack, as pointed out in [27]. As the cluster head and the sensor nodes do not share any secret in traditional cluster formation protocol, any passive adversary can send strong solicitation beacon to attract all the sensor nodes to its cluster and thus mount a sinkhole attack and perform selective forwarding during data communication phase.

Prevention For a centralized protocol, HELLO Flood attack followed by sinkhole attack can only be possible if an adversary is able to forward the solicitation from the sensor node to the base-station with a highest signal strength value appended to it. But as the adversary has to append a MAC computed using K'_{CH-BS}, which it does not have, it cannot authenticate itself to the base-station. The base-station will simply drop such messages,

thereby preventing HELLO Flood attack and the attacks that can be mounted as a result.

For a distributed protocol, the biometric authentication is used to prevent an adversary from posing as a cluster head. As the adversary is assumed not to be in contact with the body, it cannot measure the biometric directly and also because of the time-period based scheduling, the biometric value changes before they are compromised using brute-force. An adversary therefore cannot generate the correct biometric certificate and all the attacks above fail because the sensor node will simply reject any solicitation beacon without proper certificate, thereby preventing HELLO Flood and other resultant attacks.

In either case Sybil attack, leading to HELLO Flood and other attacks are not possible because, if an adversary does not have the secret key or biometrics in the centralized and distributed case respectively and it can never authenticate itself to become part of the network.

2. Passive Adversary Spoofing the Identity of a Sensor Node or Base-Station

Vulnerability In the non-secure traditional cluster formation protocol, a passive adversary can spoof the identity of a sensor node and infiltrate the network. Such an adversary can then send altered and spoofed data to the base-station. It can also send bogus messages to cluster head to relay to deplete cluster head's energy and cause probable tissue heating. Similarly, a passive adversary can spoof the identity of a base-station and try to send bogus queries and keys or create inefficient clusters, in case of centralized protocol.

Prevention For the centralized protocol, this passive adversary has to have the shared key with the base-station to join a cluster. As a passive adversary does not have the requisite keys, the MAC in the solicitation beacons will not match. The base-station will simply drop such packets. If a passive adversary spoofs the identity of the base-station

again it has to attach a MAC to its messages, computed using the key the base-station shares with each biosensor. As an adversary does not have the necessary keys, spoofing the identity of the base-station is not possible.

For the distributed protocol, as the passive adversary cannot measure the biometric, it cannot derive K_{temp}, and therefore cannot authenticate its reply. For the distributed case, spoofing base-station identity is not possible for the same reason as the centralized protocol.

3. Cryptographically Weak Biometric

Vulnerability The biometric values measured are inherently weak. If the biometric based authentication is used in secure cluster formation, then for the centralized protocol breaking the biometric value can lead to energy wastage at the cluster head, by being asked to forward bogus packets by an adversary. In case of the distributed protocol however, sinkhole formation cannot be avoided if the biometric key is broken as an adversary can broadcast a solicitation beacon with valid certificate.

Prevention To prevent the breaking of biometric value, we limit its validity to the same time period where it is measured in. The length of the time-period depends upon the capability of the adversary and the randomness of the biometric measured.

4. Physical Compromise of Cluster Head

Vulnerability Another vulnerability that can arise with our system model is that of biosensor compromise by an active adversary. For the centralized protocol, a compromised cluster head can then be forced to report higher signal strength values (while relaying the solicitation to the base-station) and try to get all the sensor nodes in the network into its

cluster. For the distributed protocol, such cluster head's compromise can cause it to send beacon at a higher power than normal and try to attract as many nodes in the network as possible.

Prevention Node compromise is a big problem the effects of which may not be completely prevented. In the centralized protocol, as the base-station decides the cluster formation, it has complete control over the topology of the sensor network. To prevent a large number of biosensors in the network making one biosensor as their cluster head, the base-station can have policies in-built which control the maximum number of nodes that can be part of a cluster. If a cluster head is trying to get more than its share of nodes in its cluster, the base-station can tag such a cluster head and try to monitor its traffic to deduce its compromise. If such a compromise is detected, the node can be easily excluded from future clusters.

For a distributed protocol, node compromise can be catastrophic and its effects cannot be eliminated without an in-built intruder monitoring protocol. Therefore even if the biometrics are strong for distributed protocol, cluster head compromise can lead to HELLO Flood attack, sinkhole attack and selective forwarding attack.

5. Physical Compromise of Sensor Node

Vulnerability If an active adversary is able to compromise a sensor node, then it can become part of the biosensor network because it can generate correct MACs. Such a compromised sensor node can send its data to the adversary using out-of-band channels and also send incorrect data to the base-station during the data communication phase.

Prevention There is no clear prevention from this attack in either centralized or distributed protocol. An active adversary can become part of it. The base-station can

single out such a sensor node if its data varies by a large percentage from the data received from other biosensors replying along with it to a query. As all responses are encrypted the compromised sensor node has to make a guess about the value it wants to send to the base-station to misguide it, which can be a difficult task.

Such a sensor node, once identified can be prevented from being part of the network in the centralized protocol from the next round of cluster head selection. In case of the distributed protocol data from the sensor node can be ignored by the base-station. It can even instruct other biosensors not to forward data from the compromised sensor node.

6. Physical Compromise During Distribution of Attribute Based Keys

Vulnerability In the protocol extension proposed, we distribute attribute based keys for efficient secure querying of biosensors during data communication phase. A vulnerability that can occur is that if a sensor node is compromised within the network before the cluster formation begins, during the cluster formation process it can first obtain its attribute based key and then send another solicitation message with different attribute information (same node id) and try to get as many attribute based keys as possible. The queries addressed to all the nodes which have the compromised attribute based keys can be read and so can the data sent by the sensors in reply to such queries.

Prevention In both the centralized and distributed cluster formation protocol extensions, the base-station decides on the attribute based keys for biosensors. By simply preventing the base-station from replying to a node's solicitation twice, an adversary can be prevented from getting attribute based key for another attribute value than that of itself. As even though a biosensor can send different attribute information in different messages it cannot change its identity value because the MAC will not match (because of the on the fly

generation of pair-wise keys from the identity of the biosensor which sent the solicitation). Therefore multiple attempts of a compromised biosensor to get other attribute based keys will fail.

7. Physical Compromise During Data Communication Phase

Vulnerability During the data communication phase, if the cluster head or sensor node is compromised, then it can send bogus data to the base-station and also communicate with the adversary using an out of band channel.

Prevention An intruder detection algorithm can be used for detecting any difference in communication patterns to find such nodes. We assume the intruder monitoring protocol reports any abnormalities to the base-station. If such biosensors are identified, then for the centralized protocol, the base-station can exclude them from the network during the next cluster rotation phase. For the distributed case, base-station can maintain a list of compromised biosensors which can be provided to all the un-compromised biosensors to ensure that the compromised ones are excluded from future cluster formation.

8. Other Attacks

Vulnerability Other standard attacks that are possible in a wireless communication are possible here in a biosensor network communication. Some of the common attacks are: replay attack, altered message and acknowledgment spoofing. Also cryptographic attacks can be mounted like chosen-plaintext attack, know-plaintext attack and ciphertext-only attack.

Prevention Both the protocols prevent these attacks while forming clusters. Replay attacks are prevented using a nonce with each message. The only condition is that the

value of the nonce has to vary with each message never repeated again. The nonce provides the requisite freshness of a transaction. Data alteration and acknowledgment spoofing are prevented by using a MAC with each message. The MAC is computed with all the data that has been exchanged so far in the course of the current transaction and the data being sent with current phase of the transaction. This totally maintains the freshness of data and prevents any kind of spoofing. During the initialization phase of the protocol, the encrypted data is mostly large random values like keys, therefore mounting any cryptanalytic attack is not feasible.

A word about energy-efficiency and security relationship is necessary here. It can be seen from the last two sections that security and energy-efficiency share a inversely proportional relationship. Security itself adds a overhead to the cluster formation process and the higher the security needed the more the energy consumption. For example the centralized protocol is the most secure but it consumes the most amount of energy (we do not consider the extension here). If we are able to increase the randomness of the biometric keys sufficiently then we can have a secure and energy-efficient cluster formation protocol.

CHAPTER 8

IMPLEMENTATION

We have done actual implementation of the protocols developed for secure cluster formation in sensor networks.

1. Mica2 Motes and TinyOS

The implementation was done using Berkeley Mica2 sensor motes, designed at University of California, Berkeley and manufactured at Crossbow Inc [2]. Each mote has a 8MHz Atmel ATMega128L micro-controller with 128KBytes of programmable flash memory and a 4KBytes of RAM. Each mote is equipped with multi-channel radio transceiver capable of working at different frequencies and is powered by 2 AA batteries. The architecture of the mote is shown in figure 9.

Table 6. Parameters used in Implementation

Parameters	**Size**	Data Type Used in Implementation
Node id	8 bits	Unsigned 8-bit integer
Node Attribute	8 bits	Unsigned 8-bit integer
Cryptographic Keys	128 bits	Unsigned 8-bit integer array of size 16
Nonces and Counters	128 bits	Unsigned 8-bit integer array of size 16
Message Authentication Code	64 bits	Unsigned 8-bit integer array of size 8
Biometric Keys	128 bits	Unsigned 8-bit integer array of size 16
Signal Strength	16 bits	Unsigned 16-bit integer

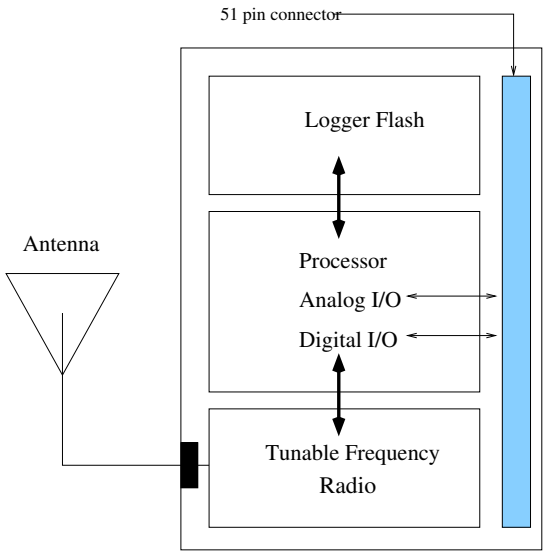

Figure 9. **A Mica2 Mote Architecture Block Diagram**

The micro-controller runs a sensor operating system called TinyOS [6], which is loaded in its flash memory. TinyOS is a open-source operating system designed for wireless low-powered sensors networks. It has a component based architecture which enables rapid development and implementation of applications for sensors. The component library is vast and contains network protocols, sensor drivers, security primitives, data acquisition tools and many other features. The TinyOS is designed to support concurrency intensive operations using the minimal hard-ware support available with the sensors.

NesC is a C like language in which the TinyOS components, libraries and applications are written. NesC is designed to develop structured, component based applications for embedded sensor networks. Each application developed for sensors is written in NesC and is in form of a component. Between multiple components there is a by-directional interface. Each component uses a interface which is the only point of access of components. The interface defines a set of functions that the interface provider must implement and a set of events that the interface user must implement. One or more such inter-linked components

(both user defined and in-built) form an executable which will run on the motes.

As ours is a security protocol, we need security primitives like encryption and MAC for implementing our protocol. TinyOS comes with a link layer security solution in-built. It is called TinySec [26] and provides us with security components which can be integrated into our application for ensuring security. In TinyOS version 1.1.0, which was used for this implementation, TinySec comes three built in encryption algorithms, they are SkipJack, RC5 and Identity Cipher. It also provides components for CBC-MAC (Cyclic Block Chaining-Message Authentication Code) which can be attached to a message to maintain authenticity. Each of the three encryption algorithms implementation support a block size which is 64 bits long. As the CBC-MAC size was same as the block size in their implementation, the size of the MACs used in our implementation is only 64 bits.

2. Implementation

The protocol implementations have four types of entities: **the base-station, the cluster head, the sensor node and the listener**. The centralized cluster formation protocol was implemented with one mote acting as the base-station, three cluster head motes, four sensor node motes and one listener. The base station mote was assigned the id 8, the cluster head motes' ids were 5,6 and 7, the sensor nodes motes' id was between 1-4. The listener node was not assigned any identification number. (The distributed cluster formation protocols do not need the base-station and which was therefore not included in the implementation of the two distributed schemes). The listener application captures the message packets communicated during the cluster formation process and sends them to the desktop. The listener mote was programmed with the TOSBase application which comes with the TinyOS package. The mote programmed with TOSBase kept its receiver on all

the time and listened to any communication going on within its receive range. Whenever it (TOSBase) receives a message on the wireless channel, it sends it to the desktop through a serial port. At the desktop, a special application called Serial Forwarder monitors the serial port, collects any data it receives through the serial port and sends it to another Java application. which displays the content of the message on the monitor.

When the motes communicate, they use a TinyOS message packet. This message packet has a Active Message (AM) component which allows us to store various application specific data in it. All the data exchanged during the various phases of the protocols were written and read from the AM component of the TinyOS packet. The size of the AM component is variable but by default is fixed at 29 bytes. We increased AM size to 100 bytes to fit in the data that was to be exchanged during the cluster formation process.

As this is a security protocol, many of the elements of the various message exchanged during the cluster formation process were cryptographic in nature. For ensuring data confidentiality, integrity and authenticity we needed encryption components and MAC components. As mentioned before TinySec comes with these security components which were wired in our application and used with ease. For encryption we used the SkipJack algorithm. For MAC, TinySec provides an implementation of CBC-MAC, the components of which were wired to our application.

Another important aspect of our protocol is the biometric measurement and the schedule to do so. In this implementation we did not actually measure any biometrics, but used pre-programmed biometric values. The biometric was assumed to be blood glucose level, whose values varied between 60-140mg/dl [19]. Each mote while being programmed was assigned a biometric value based on the time-period when it would transmit its solici-

tation [1]

For the centralized cluster formation protocol (both with and without attribute based keying), we logically assigned motes 1 and 2 to one time-period called $TP0$, both these motes were programmed with identical biometric values $BioKey_C0$. Motes 3 and 4 were assigned to $TP1$ and were programmed with the same biometric value $BioKey_C1$. The motes 4-6, which are the cluster heads, were programmed with both biometric values- $BioKey_C0$ and $BioKey_C1$. Timers were set appropriately such that motes 1,2,3,4 broadcasted their solicitation in the appropriate order at an interval of 10 seconds. The cluster head motes used the $BioKey_C0$ for authenticating the solicitation from motes 1 and 2 and used $BioKey_C1$ for authenticating motes 3 and 4 solicitation, thereby logically achieving a change in the time-period. The cluster head motes forward the data to the base-station in order of their identification numbers. Therefore mote 4 will forward the solicitation it receives to the base-station followed by node 5 and 6 in that order.

For both the distributed cluster formation cases, the roles are reversed, the cluster head motes- 4 and 5 were assigned to $TP0$, with a biometric value $BioKey_D0$ and mote 6 to $TP1$ pre-programmed with $BioKey_D1$. Motes, 1,2,3,4, used appropriate biometric values depending upon which mote (cluster head) was transmitting. The sensor motes reply to the solicitation in the order of their identification numbers.

In both the cases, the inter-solicitation interval was 10 seconds, resulting in an effective time-period length to be 20 seconds. The parameters used in our implementation are given the table 6.

[1] We have assumed in this implementation, that the biometrics are random in nature and cannot be guessed before the end of a time-period

3. Attacks Simulated

In order to prove the merit of the secure cluster formation protocol, we implemented attacks on our protocol implementation. We wanted to show that the main attacks against which we started out are prevented, namely - HELLO Flood attack, Sinkhole attack and Selective Forwarding attack. It should be seen that if HELLO Flood attack is mounted then sinkhole attack and selective forwarding attack are consequences of those and cannot be avoided. Therefore the main aim of the implementation was to avoid HELLO Flood attack.

Hello Flood Attack

Goal of the Experiment : The goal of simulating this attack is to show that HELLO Flood attack is not successfully mounted and a sensor node does not choose a malicious entity as its cluster head.

Setup : In order to implement this attack, for the centralized protocol, we first setup our test-bed with 3 motes acting as cluster heads, 2 motes as sensor nodes, one mote base station and one as the malicious entity. The malicious entity was programmed to forward the solicitations it receives with a power level value of 5. For Mica2 motes the closer a node is lower the power value. Other legitimate cluster heads were receiving solicitations at values varying from 25-45. The malicious entity was thus trying to mount a HELLO Flood attack.

For the distributed protocol, the base station was not involved and the malicious entity was located closer to the sensor nodes than the legitimate cluster heads. Thus the sensors nodes were receiving the beacon from the malicious entity at a higher signal strength (lower mica2 signal strength value) than others. The malicious entity's signal strength received

from between 0-10, while the legitimate cluster head's signal strength value was between 35-70. Again a HELLO Flood attack was being mounted this time in the distributive scenario. The malicious entity was using a random value biometric value for its certificate.

Result : In both cases Hello Flood attack was prevented, for the centralized protocol, the malicious entity which was sending it bogus messages (with elevated signal strength values), was rejected by the base-station as the MAC appended by the malicious entity did not match. To increase the signal strength value the malicious entity was placed closer to the sensor nodes than legitimate cluster heads. For the distributed case, as the malicious entity did not have the appropriate biometric value and the certificate was not authenticated at the sensor node and therefore the solicitation was rejected.

Identity Spoofing

Goal of the Experiment : The goal of simulating this attack is to show that identity spoofing does not let a malicious entity disrupt the activities of the network.

Setup : The attack setup was similar to the previous attack, only this time the malicious entity was trying to spoof the identity of a sensor node and trying to join the network.

Result : For the centralized protocol, a malicious entity was trying to masquerade as sensor nodes sending messages to a specific chosen cluster head in the distributed protocol failed because the malicious entity does not know the key K_{temp} sent by the cluster head during the broadcast of the solicitation.

We did not implement any physical compromise attacks because they would need a intrusion detection protocol which was out of the scope of this thesis. Also synchronization is essential for communication in sensors and was extremely difficult to achieve. The lack of synchronization was causing the disruption of legitimate communication within the network.

This illustrates the ease of mounting the denial of service attacks (DoS) due to jamming on sensor networks which we do not take into consideration here.

We show 6 scenarios which were used for testing the implementation of the two basic secure cluster formation protocols and their extensions in the appendices to this thesis. The 6 cases are as follows:

- Appendix 1 shows the first scenario shows the output of a successful run of centralized secure cluster formation protocol. It uses 4 sensor nodes, 3 cluster heads and 1 base-station.

- Appendix 2 shows the second scenario shows the output of a successfully run of distributed secure cluster formation protocol. It uses 4 sensor nodes, 3 cluster heads and 1 base-station.

- Appendix 3 shows the third scenario shows the output of a successfully run of centralized secure cluster formation protocol with two malicious entities (node 3 as sensor node and node 6 as cluster head). It uses 2 legitimate sensor nodes, 2 legitimate cluster heads, 1 malicious sensor node, 1 malicious cluster head and 1 base-station.

- Appendix 4 shows the fourth scenario shows the output of a successfully run of distributed secure cluster formation protocol. Here we have 1 malicious entity (node 7) which sends in a stronger solicitation signal to form sinkholes. It uses 2 legitimate sensor nodes, 2 legitimate cluster heads, 1 malicious cluster head and 1 base-station.

- Appendix 5 shows the fifth scenario shows the output of a successful run of centralized secure cluster formation protocol with extensions. It has 2 sensor nodes, 3 cluster heads and 1 base-station.

- Appendix 6 shows the first scenario shows the output of a successful run of distributed secure cluster formation protocol. It has 2 sensor nodes, 3 cluster heads and 1 base-station.

CHAPTER 9

CONCLUSIONS AND FUTURE WORK

Security in biomedical sensors is an important topic to address in detail. In this thesis we have described two secure cluster topology formation protocols, for biosensor networks. The first was the centralized protocol which asked the base-station to decide on the cluster of a sensor node. The distributed protocol followed the traditional cluster formation approach which, in which the cluster-heads and sensor nodes distributively decide on the clusters within the network.

We used the novel scheme of biometric authentication to authenticate communication between the cluster head and sensor nodes within the network. For the centralized scheme, biometric authentication prevented bogus solicitation packets from being forwarded towards the base-station by the cluster head, thereby saving its energy. For the distributed scheme biometric authentication prevented a malicious node from becoming a sinkhole by mounting a HELLO Flood attack.

We also proposed extensions to these basic protocols to include data key distribution. These data communication keys will be used for secure communication of queries and reply data between the base-station and the sensor network. These keys are distributed based on specific attributes of a node and therefore allow secure attribute centric communication of queries and responses, which is ideal for sensor networks. Both centralized and distributed

secure cluster formation protocols were extended to allow the distribution of attribute based keys to the nodes. For centralized protocol this could be done in the same phase as cluster formation, but for the distributed protocol, keys had to be distributed separately.

When comparing the protocols in terms of energy consumed, we found that the distributed protocol without key distribution consumed the least amount of energy, followed by the centralized without key distribution, centralized protocol with key distribution and finally distributed protocol with key distribution in that order.

In terms of security, the distributed protocols were less secure than the centralized ones because they depend on biometrics for their entire security. As biometrics are weaker, security can be compromised by a motivated adversary. The centralized protocols are more secure as they ask the base station to decide on the clusters within the network with which all nodes share a secret pre-deployed key.

A note on the possible future work. In this thesis work, it has been assumed that biometrics are random enough for committing certificate keys. In reality, the biometrics may not be random enough and we need schemes which can increase the randomness to make the process more secure. Another problem with biometrics is that the biometric values do not vary much over time during cluster formation in the distributed protocol. If this happens an adversary gets multiple time-periods to break a biometric value. In such cases the protocol provides little help. In order to prevent this scenario, we need schemes to randomize the biometric value in each time-period by a limited extent. But such scenarios will also require sophisticated error correction schemes at the receiver. This is however a topic is for future work. In order to improve the implementation of the protocols, we plan to use devices like EKG monitors [8] to collect biometrics in real-time and use them in authentication and secure communication.

REFERENCES

[1] Aging stats website. http://www.agingstats.gov/.

[2] Crossbow inc. website. http://www.xbow.com/.

[3] Hitachi inc. website. http://www.hitachi.com/.

[4] I-stat.com website. http://www.istat.com/.

[5] Microchips website. http://www.mchips.com/.

[6] Tinyos website. http://www.tinyos.net/.

[7] Verichip website. http://www.4verichip.com/index.htm.

[8] Vernier technologies website. http://www.vernier.com.

[9] Summary of hipaa (health insurance probability act). May 2003. US Department of Health and Human Service.

[10] A.D. Amis and R. Prakash. Load balancing clusters in wireless ad hoc networks. pages 25–35, March 2000. In Proceedings of ASSNET Conference.

[11] A.D. Amis, R. Prakash, T.H.P Vuong, and D.T. Huynh. Max-min d-cluster formation in wireless ad hoc networks. volume 1, pages 32–41, March 2000. In Proceedings of IEEE Infocom Conference.

[12] T. Aura. Strategies against replay attack. pages 59–68, June 1997. In Proceedings of 10th IEEE Computer Security Foundation Workshop.

[13] D.J Baker and A. Ephremides. The architectural organization of a mobile radio via a distributed algorithm. *IEEE Transactions on Communications*, 29(11):1694–1701, 1981.

[14] S. Bandyopadhyay and E.J. Coyle. An energy efficient hierarchical clustering algorithm for wireless sensor networks. volume 3, pages 1713– 1723, April 2003. In Proceedings IEEE Infocom 2003.

[15] Stephano Basagni. Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks. volume 2, pages 889–893, September 1999. In Proceedings of the Vehicular Technology Conference.

[16] Stephano Basagni. Distributed clustering for ad-hoc networks. pages 310–315, June 1999. In Proceedings of the Symposium on Parallel Architectures Algorithms and Networks.

[17] David W. Carman, Peter S. Kruus, and Brian J. Matt. Constraints and approaches for distributed sensor network security. NAI Labs Technical Report #00-010.

[18] M Chatterjee, S.K Das, and D. Turgut. Wca: A weighted clustering algorithm for mobile ad hoc networks. *Journal of Cluster Computing, Special Issue on Mobile Ad hoc Networking*, 5:193–204, April 2002.

[19] S. Cherukuri, K. Venkatasubramanian, and S.K.S. Gupta. Biosec: A biometric based approach for securing communication in wireless networks of biosensors implanted in

the human body. pages 432 – 439, October 2003. In Proceedings of the ICPP, Wireless Security and Privacy Workshop (WiSPr 03).

[20] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. pages 41 – 47, November 2002. Proceedings of the 9th ACM conference on Computer and Communications Security.

[21] M. Gerla and J.T.C Tsai. Multicluster, mobile, multimedia radio networks. *Wireless Networks*, 1(3):255–265, 1995.

[22] W.R. Heinzelmann. Application specific protocol architectures for wireless networks, 2000. PhD Thesis.

[23] W.R. Heinzelmann, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocols for wireless microsensor networks. pages 8020–8029, January 2000. In Proceedings of Hawaii International Conference on System Sciences.

[24] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. pages 56–67, 2000. In Proceedings of the 6th Annual International Conference on Mobile Computing and Networking.

[25] Ari Juels and Martin Wattenberg. A fuzzy commitment scheme. pages 28–36, November 1999.

[26] Chris Karlof, Naveen Sastry, and David Wagner. Tinysec: Link layer encryption for tiny devices: Userguide. January 2004.

[27] Chris Karloff and David Wagner. Secure routing in wireless sensor neworks: Attacks

and countermeasures. *Elsevier's AdHoc Networks Journal, Special Issue on Sensor Network Applications and Protocols*, 1(2-3):293–315, September 2003.

[28] C.R. Lin and M. Gerla. Adaptive clustering for mobile wireless networks. *Journal on Selected Areas in Communication*, 15(7):1265–1275, September 1997.

[29] Donggang Liu and Peng Ning. Establishing pairwise keys in distributed sensor networks. pages 52–61, October 2003. In Proceedings of the 10th ACM Conference on Computer and Communications Security.

[30] A.B. McDonald and T.Zanti. A mobility based framework for adaptive clustering in wireless ad hoc networks. *IEEE Journal on Selected Areas in Commmunications*, 17(8):1466–1487, August 1999.

[31] Seung-Jong Park and Raghupathy Sivakumar. Poster: Sink-to-sensor reliability in sensor networks. pages 27–28, June 2003. Symposium on Mobile Ad-hoc Networking and Computing.

[32] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tygar. Spins: Security protocol for sensor networks. pages 189–199, July 2001. In Proceedings of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking.

[33] Roberto Di Pietro, Luigi V. Mancini, and Alessandro Mei. Random key assignment for secure wireless sensor networks. pages 62 – 71, October 2003. In Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03).

[34] Bruce Schneier. *Applied Cryptography.* John Wiley and Sons, 2 edition, 1996.

[35] L. Schwiebert, Sandeep K. Gupta, and Jennifer Weinmann. Research challenges in wireless networks of biomedical sensors. pages 151–165, July 2001. In Proceedings

of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking.

[36] Anthony D. Wood and John A. Stankovic. Denial of service in sensor networks. *IEEE Computer*, 35(10):48–56, 2002.

APPENDIX A

CENTRALIZED SECURE CLUSTER FORMATION PROTOCOL

SENSOR NODE 4 Broadcasting Solicitation

Sensor Node ID 4

Gamma is

53 79 32 98 122 212 61 194 10 92 158 37 134 111 247 88

Nonce is

22 50 242 179 38 21 20 92 39 126 254 71 104 217 120 114

Sensor Node 4 MAC is

152 125 153 221 93 115 109 113

Assigned Cert is

206 173 209 81 48 77 163 83

CH 5 Forwarding SENSOR NODE 4's Solicitation to BS

Sensor Node ID 4

Counter is

244 249 76 87 30 3 200 47 248 130 240 99 86 248 183 247

Nonce is

22 50 242 179 38 21 20 92 39 126 254 71 104 217 120 114

Sensor Node 4 MAC is

152 125 153 221 93 115 109 113

CHID is 5

SS is 30

KCH-N in Encrypted Form is

93 73 25 247 28 37 140 131 135 108 126 39 11 133 102 15

Cluster Head 5 MAC is

248 95 193 82 32 38 63 174

CH 6 Forwarding Node 4's Solicitation to BS

Sensor Node ID 4

Counter is

218 18 97 104 66 206 141 162 233 47 189 212 220 122 187 168

Nonce is

169 65 139 102 142 194 185 175 242 177 157 214 85 106 22 143

Sensor Node 4 MAC is

153 251 188 200 16 152 214 214

CHID is 6

SS is 28

KCH-N in Encrypted Form is

193 5 135 116 116 20 239 201 241 162 0 13 58 54 56 195

Cluster Head 6 MAC is

14 246 114 252 114 29 21 55

CH 7 Forwarding Node 4's Solicitation to BS

Sensor Node ID 4

Counter is

45 47 144 27 63 46 12 3 196 239 104 40 178 118 132 147

Nonce is

169 65 139 102 142 194 185 175 242 177 157 214 85 106 22 143

Sensor Node 4 MAC is

153 251 188 200 16 152 214 214

CHID is 7

SS is 23

KCH-N in Encrypted Form is

74 150 89 232 106 206 204 233 86 54 91 134 201 3 114 214

Cluster Head 7 MAC is

179 169 155 119 206 188 249 132

****

SENSOR NODE 1 Broadcasting Solicitation

Sensor Node ID 1

Gamma is

236 56 250 56 31 11 168 42 146 249 88 150 75 56 4 139

Nonce is

171 53 108 165 209 189 10 35 71 187 35 117 25 216 32 210

Sensor Node 1 MAC is

204 131 198 199 205 184 104 40

Assigned Cert is

233 216 23 154 150 228 202 225

CH 5 Forwarding SENSOR NODE 1's Solicitation to BS

Sensor Node ID 1

Counter is

217 172 248 2 115 222 12 9 237 231 19 189 97 114 68 207

Nonce is

171 53 108 165 209 189 10 35 71 187 35 117 25 216 32 210

Sensor Node 1 MAC is

204 131 198 199 205 184 104 40

CHID is 5

SS is 26

KCH-N in Encrypted Form is

137 202 30 123 36 212 94 211 54 58 170 253 180 222 243 30

Cluster Head 5 MAC is

122 134 105 150 244 130 67 111

CH 6 Forwarding Node 1's Solicitation to BS

Sensor Node ID 1

Counter is

53 133 140 141 140 160 78 103 47 54 7 210 158 138 214 173

Nonce is

171 53 108 165 209 189 10 35 71 187 35 117 25 216 32 210

Sensor Node 1 MAC is

204 131 198 199 205 184 104 40

CHID is 6

SS is 28

KCH-N in Encrypted Form is

66 226 220 67 129 204 159 118 113 231 228 245 22 141 250 95

Cluster Head 6 MAC is

228 2 38 153 212 41 56 107

CH 7 Forwarding Node 1's Solicitation to BS

Sensor Node ID 1

Counter is

74 115 74 25 179 64 204 81 58 193 197 132 16 188 51 198

Nonce is

171 53 108 165 209 189 10 35 71 187 35 117 25 216 32 210

Sensor Node 1 MAC is

204 131 198 199 205 184 104 40

CHID is 7

SS is 29

KCH-N in Encrypted Form is

83 245 122 129 85 114 251 151 114 146 91 24 130 162 172 26

Cluster Head 7 MAC is

244 199 251 5 85 239 22 5

****

SENSOR NODE 2 Broadcasting Solicitation

Sensor Node ID 2

Gamma is

88 51 152 129 10 62 106 90 157 17 42 29 247 66 58 217

Nonce is

179 238 234 206 166 44 222 107 78 217 78 107 227 202 145 248

Sensor Node MAC is

92 238 173 205 10 157 200 42

```
Assigned Cert is
246  251  0  29  40  86  166  78
-----------------------------------------------------------------
CH 5 Forwarding SENSOR NODE 2's Solicitation to BS
Sensor Node ID  2
Counter is
30  113  181  66  119  133  41  127  113  187  80  129  222  44  117  211
Nonce is
179  238  234  206  166  44  222  107  78  217  78  107  227  202  145  248
Sensor Node 2 MAC is
92  238  173  205  10  157  200  42
CHID is   5
SS is   38
KCH-N  in Encrypted Form is
106  155  106  137  216  204  145  203  39  192  9  169  20  145  117  9
Cluster Head 5 MAC is
162  245  236  198  123  133  170  162
-----------------------------------------------------------------
CH 6 Forwarding SENSOR NODE 2's Solicitation to BS
Sensor Node ID  2
Counter is
15  141  98  228  158  114  253  73  153  121  153  37  161  168  84  93
Nonce is
179  238  234  206  166  44  222  107  78  217  78  107  227  202  145  248
```

```
Sensor Node 2 MAC is
92  238  173  205  10  157  200  42
CHID is   6
SS is   30
KCH-N  in Encrypted Form is
224  37  97  182  96  107  114  5  200  29  104  110  172  184  60  137
Cluster Head 6 MAC is
151  102  26  99  197  67  17  231
-----------------------------------------------------------------
CH 7 Forwarding SENSOR NODE 2's Solicitation to BS
Sensor Node ID  2
Counter is
105  111  149  201  184  62  194  223  23  135  44  83  100  191  234  74
Nonce is
179  238  234  206  166  44  222  107  78  217  78  107  227  202  145  248
Sensor Node 2 MAC is
92  238  173  205  10  157  200  42
CHID is   7
SS is   32
KCH-N  in Encrypted Form is
79  236  150  88  75  253  146  229  38  140  220  106  148  248  165  58
Cluster Head 7 MAC is
192  162  164  43  39  121  42  84
*****************************************************************
```

SENSOR NODE 3 Broadcasting Solicitation

Sensor Node ID 3

Gamma is

202 34 142 75 190 112 213 146 76 64 181 230 83 164 166 232

Nonce is

169 65 139 102 142 194 185 175 242 177 157 214 85 106 22 143

Sensor Node 3 MAC is

153 251 188 200 16 152 214 214

Assigned Cert is

139 138 59 91 137 23 112 168

CH 5 Forwarding SENSOR NODE 3's Solicitation to BS

Sensor Node ID 3

Counter is

218 192 147 32 166 222 2 205 249 37 207 92 77 210 123 27

Nonce is

169 65 139 102 142 194 185 175 242 177 157 214 85 106 22 143

Sensor Node 3 MAC is

153 251 188 200 16 152 214 214

CHID is 5

SS is 34

KCH-N in Encrypted Form is

193 36 184 134 147 114 139 188 17 243 114 129 152 196 141 225

Cluster Head 5 MAC is

171 115 34 196 116 82 219 145

CH 6 Forwarding Node 3's Solicitation to BS

Sensor Node ID 3

Counter is

116 143 127 161 15 95 101 93 251 118 93 227 243 10 26 116

Nonce is

169 65 139 102 142 194 185 175 242 177 157 214 85 106 22 143

Sensor Node 3 MAC is

153 251 188 200 16 152 214 214

CHID is 6

SS is 20

KCH-N in Encrypted Form is

88 251 96 45 212 51 28 248 36 181 227 3 166 229 172 122

Cluster Head 6 MAC is

20 150 183 46 114 168 106 14

CH 7 Forwarding Node 3's Solicitation to BS

Sensor Node ID 3

Counter is

142 122 237 177 10 64 46 95 148 171 195 149 207 33 245 65

Nonce is

169 65 139 102 142 194 185 175 242 177 157 214 85 106 22 143

Sensor Node 3 MAC is

153 251 188 200 16 152 214 214

CHID is 7

SS is 25

KCH-N in Encrypted Form is

37 162 242 138 230 143 253 98 104 127 191 11 231 13 170 26

Cluster Head 7 MAC is

4 185 190 247 10 172 109 82

****

BS replying to SENSOR NODE 1

Cluster Head assigned to Sensor Node 1 is 5

Counter is

52 17 82 40 41 43 47 46 32 57 11 111 174 37 51 31

Nonce is

171 53 108 165 209 189 10 35 71 187 35 117 25 216 32 210

KCH-N in Encrypted with KBS-N is

a 0 ? # " + 7 S I { {) F ? & p

MAC sent is

58 179 195 200 209 94 142 195

SENSOR NODE 1 receives the following reply from BS (after decryption)

Cluster Head assigned to Sensor Node 1 is 5

Key received is

x c d f p o l k m k l a x c d f

MAC computed is

```
58  179  195  200  209  94  142  195
**********************************************************************
BS replying to SENSOR NODE 2

Cluster Head assigned to Sensor Node 2 is  6
Counter is
229  179  31  78  224  185  11  102  176  20  92  204  236  161  59  6
Nonce is
179  238  234  206  166  44  222  107  78  217  78  107  227  202  145  248
KCH-N  in Encrypted with KBS-N is
?  +  s    v  +  ?    +    K  ?  ?  -    ?
MAC sent is
1  246  144  198  113  3  150  107
----------------------------------------------------------------------
SENSOR NODE 2 receives the following reply from BS (after decryption)
Cluster Head assigned to Sensor Node 2 is  6
Key received is
f  g  s  h  d  l  f  j  k  l  z  o  l  f  g  s
MAC computed is
1  246  144  198  113  3  150  107
**********************************************************************
BS replying to SENSOR NODE 3
Cluster Head assigned to Sensor Node 3 is  6
Counter is
```

31 219 143 130 119 200 91 8 176 26 66 229 246 27 152 71

Nonce is

169 65 139 102 142 194 185 175 242 177 157 214 85 106 22 143

KCH-N in Encrypted with KBS-N is

^ * h a @ 1 ;) . U ? & ~ - +

MAC sent is

152 41 132 248 153 88 60 4

SENSOR NODE 3 receives the following reply from BS (after decryption)

Cluster Head assigned to Sensor Node 3 is 6

Key received is

f g s h d l f j k l z o l f g s

MAC computed is

152 41 132 248 153 88 60 4

****

BS replying to SENSOR NODE 4

Cluster Head assigned to Sensor Node 4 is 7

Counter is

40 127 218 123 196 76 67 201 222 73 26 251 15 11 103 96

Nonce is

22 50 242 179 38 21 20 92 39 126 254 71 104 217 120 114

KCH-N in Encrypted with KBS-N is

! h] 2 * = % ? { ; > n * < $

MAC sent is

```
7  240  183  83  21  217  239  41

-----------------------------------------------------------------

SENSOR NODE 4 receives the following reply from BS (after decryption)

Cluster Head assigned to Sensor Node 4 is  7

Key received is

a  d  r  g  n  h  d  q  k  l  p  o  b  h  m  j  w

MAC computed is

7  240  183  83  21  217  239  41
```

APPENDIX B

DISTRIBUTED SECURE CLUSTER FORMATION PROTOCOL

CH 5 Broadcasting Solicitation

CHID is 5

Encrypted Key Ktemp is

q - E o = t ‘ i 7 S 0 l

Cert is

96 91 81 225 29 106 236 155

Delta is

117 154 160 148 13 184 126 28 238 135 180 224 162 48 92 68

Nonce is

126 187 106 85 7 98 67 146 123 176 24 97 188 102 211 194

Gamma is

128 232 153 189 63 111 55 165 118 181 128 152 65 253 224 242

Counter is

159 161 192 131 177 39 12 45 232 180 202 179 239 134 148 21

CH 6 Broadcasting Solicitation

CHID is 6

Encrypted Key Ktemp is

l ^ # ~ ^ z (- m _ ’ > ; + A

Cert is

91 253 94 128 66 135 99 118

Delta is

2 33 155 149 13 205 11 248 182 130 24 27 88 110 77 23

Nonce is

95 130 40 161 110 164 170 222 157 199 53 210 198 177 20 52

Gamma is

254 8 134 173 77 228 123 239 42 203 253 239 238 38 224 94

Counter is

22 45 233 248 69 111 61 116 192 105 130 204 83 235 48 108

CH 7 Broadcasting Solicitation

CHID is 7

Encrypted Key Ktemp is

_ q x & s * I V c { o } | ‘

Cert is

37 216 211 108 253 221 199 249

Delta is

205 225 200 70 41 153 251 52 58 211 15 156 29 232 57 32

Nonce is

88 97 30 202 20 159 42 190 82 103 49 204 242 91 200 245

Gamma is

243 251 249 97 119 102 165 139 59 152 226 164 210 237 171 163

Counter is

32 3 59 222 14 234 170 53 186 225 233 27 241 139 92 174

****

Reply Sent out by SENSOR NODE 1

Sensor Node ID 1

MAC is

```
35  78  228  194  130  224  171  134

Chosen CHID is 5

Decrypted Key Ktemp (Just for Information) is

x  c  d  f  p  o  l  k  m  k  l  a  x  c  d  f

Signal Strength Values as received (Just for Information) by the SENSOR NODE 1 is

SS from 5  56

SS from 6  70

SS from 7  83

**********************************************************************

Reply Sent out by SENSOR NODE 2

Sensor Node ID 2

MAC is

131  42  194  246  200  115  119  150

Chosen CHID is 5

Decrypted Key Ktemp (Just for Information) is

x  c  d  f  p  o  l  k  m  k  l  a  x  c  d  f

Signal Strength Values as received (Just for Information) by the SENSOR NODE 2 is

SS from 5  34

SS from 6  90

SS from 7  54

**********************************************************************

Reply Sent out by SENSOR NODE 3

Sensor Node ID 3

MAC is
```

```
117  62  173  150  1  150  229  165

Chosen CHID is 7

Decrypted Key Ktemp (Just for Information) is

a  d  r  g  n  h  d  q  k  l  p  o  b  h  m  j  w

Signal Strength Values as received (Just for Information) by the SENSOR NODE 3 is

SS from 5  76

SS from 6  60

SS from 7  49

**********************************************************************

Reply Sent out by SENSOR NODE 4

Sensor Node ID 4

MAC is

83  54  92  114  177  183  59  245

Chosen CHID is 6

Decrypted Key Ktemp (Just for Information) is

f  g  s  h  d  l  f  j  k  l  z  o  l  f  g  s

Signal Strength Values as received (Just for Information) by the SENSOR NODE 4 is

SS from 5  71

SS from 6  69

SS from 7  83
```

APPENDIX C

CENTRALIZED SECURE CLUSTER FORMATION PROTOCOL WITH MALICIOUS ENTITY

```
SENSOR NODE 3 (Malicious Entity) Broadcasting Solicitation

Sensor Node ID  3

Gamma is

133  227  154  33  232  84  66  24  132  16  207  223  239  247  195  123

Nonce is

144  16  29  14  33  127  195  187  75  162  121  207  163  114  217  134

Sensor Node MAC is

119  42  94  70  251  175  147  176

Assigned Certificate is

244  187  111  209  99  6  247  117

**********************************************************************

SENSOR NODE 2 (Legitimate Sensor Node) Broadcasting Solicitation

Sensor Node ID  2

Gamma is

133  227  154  33  232  84  66  24  132  16  207  223  239  247  195  118

Nonce is

97  82  61  234  77  31  163  219  38  220  40  192  16  185  231  94

Sensor Node MAC is

10  70  16  174  250  228  60  37

Assigned Cert is

41  150  203  71  56  235  101  214

----------------------------------------------------------------------

CH 4 (Legitimate Cluster Head) Forwarding SENSOR NODE 2's Solicitation to BS

Sensor Node ID  2
```

```
Counter is
178  206  54  207  52  203  60  219  9  181  192  42  247  68  34  231
Nonce is
97  82  61  234  77  31  163  219  38  220  40  192  16  185  231  94
Sensor Node 2 MAC is
10  70  16  174  250  228  60  37
CHID is   4
SS is   38
KCH-N  in Encrypted Form is
145  51  96  145  44  70  42  214  237  122  20  219  166  197  104  176
Cluster Head 4 MAC is
180  22  197  45  131  5  138  101
-----------------------------------------------------------------------
CH 5 (Legitimate Cluster Head) Forwarding SENSOR NODE 2's Solicitation to BS
Sensor Node ID  2
Counter is
211  140  59  85  128  38  98  234  247  205  185  88  147  12  59  92
Nonce is
97  82  61  234  77  31  163  219  38  220  40  192  16  185  231  94
Sensor Node 2 MAC is
10  70  16  174  250  228  60  37
CHID is   5
SS is   30
KCH-N  in Encrypted Form is
```

21 90 246 79 76 78 136 151 135 17 164 192 209 10 171 92

Cluster Head 5 MAC is

153 215 154 118 228 152 39 224

CH 6 (Malicious Entity) Forwarding SENSOR NODE 2's Solicitation to BS

Sensor Node ID 2

Counter is

197 110 56 148 197 110 49 147 207 122 16 196 101 46 177 147

Nonce is

97 82 61 234 77 31 163 219 38 220 40 192 16 185 231 94

Sensor Node 2 MAC is

10 70 16 174 250 228 60 37

CHID is 6

SS is 5

KCH-N in Encrypted Form is

150 253 234 242 142 125 97 53 28 75 215 75 131 112 40 203

Cluster Head 6 MAC is

120 5 243 151 198 142 244 112

****

SENSOR NODE 1 Broadcasting Solicitation

Sensor Node ID 1

Gamma is

129 182 155 33 232 84 66 24 132 16 207 223 239 247 204 123

Nonce is

```
79  136  26  43  64  159  33  93  165  85  188  103  209  189  101  213
Sensor Node 1 MAC is
101  147  89  219  232  45  148  191
Assigned Cert is
15  163  243  252  98  236  96  37
-----------------------------------------------------------------
CH 4 (Legitimate Cluster Head) Forwarding SENSOR NODE 1's Solicitation to BS
Sensor Node ID  1
Counter is
101  96  99  108  123  73  56  218  30  150  143  177  197  45  240  74
Nonce is
79  136  26  43  64  159  33  93  165  85  188  103  209  189  101  213
Sensor Node 1 MAC is
101  147  89  219  232  45  148  191
CHID is   4
SS is   26
KCH-N  in Encrypted Form is
145  51  96  145  44  70  42  214  237  122  20  219  166  197  104  176
Cluster Head 4 MAC is
208  143  159  239  141  43  209  158
-----------------------------------------------------------------
CH 5 (Legitimate Cluster Head) Forwarding Node 1's Solicitation to BS
Sensor Node ID  1
Counter is
```

63 84 139 60 78 178 71 164 107 252 219 149 9 56 86 143

Nonce is

79 136 26 43 64 159 33 93 165 85 188 103 209 189 101 213

Sensor Node 1 MAC is

101 147 89 219 232 45 148 191

CHID is 5

SS is 28

KCH-N in Encrypted Form is

21 90 246 79 76 78 136 151 135 17 164 192 209 10 171 92

Cluster Head 5 MAC is

173 112 33 175 1 97 185 250

CH 6 (Malicious Entity) Forwarding Node 1's Solicitation to BS

Sensor Node ID 1

Counter is

122 16 196 101 43 191 154 217 163 162 169 131 226 41 131 239

Nonce is

79 136 26 43 64 159 33 93 165 85 188 103 209 189 101 213

Sensor Node 1 MAC is

101 147 89 219 232 45 148 191

CHID is 6

SS is 5

KCH-N in Encrypted Form is

150 253 234 242 142 125 97 53 28 75 215 75 131 112 40 203

```
Cluster Head 6 MAC is

50  181  81  180  129  101  142  69

**********************************************************************

BS replying to SENSOR NODE 1

Cluster Head assigned to Sensor Node 1 is  4

Counter is

52  17  82  40  41  43  47  46  32  57  11  111  174  37  51  31

Nonce is

79  136  26  43  64  159  33  93  165  85  188  103  209  189  101  213

KCH-N  in Encrypted with KBS-N is

a  0  ?  #  "  +  7  S  I  {  {  )  F  ?  &  p

MAC sent is

58  179  195  200  209  94  142  195

----------------------------------------------------------------------

Node 1 receives the following reply from BS (after decryption)

Cluster Head assigned to Sensor Node 1 is   4

Key received is

x  c  d  f  p  o  l  k  m  k  l  a  x  c  d  f

MAC sent is

58  179  195  200  209  94  142  195

**********************************************************************

BS replying to SENSOR NODE 2

Cluster Head assigned to Sensor Node 2 is  5

Counter is
```

```
229  179  31  78  224  185  11  102  176  20  92  204  236  161  59  6

Nonce is

97  82  61  234  77  31  163  219  38  220  40  192  16  185  231  94

KCH-N  in Encrypted with KBS-N is

?  +  s    v  +  ?    +    K  ?  ?  -    ?

MAC computed is

1  246  144  198  113  3  150  107

-------------------------------------------------------------------

Node 2 receives the following reply from BS (after decryption)

Cluster Head assigned to Sensor Node 2 is  5

Key received is

f  g  s  h  d  l  f  j  k  l  z  o  l  f  g  s

MAC computed is

1  246  144  198  113  3  150  107
```

APPENDIX D

DISTRIBUTED SECURE CLUSTER FORMATION PROTOCOL WITH MALICIOUS ENTITY

CH 5 (Legitimate Cluster Head) Broadcasting Solicitation

CHID is 5

Encrypted Key Ktemp is

+ @ ? [? I - ? + s x ? e D +

Cert is

133 134 111 221 64 139 251 148

Delta is

11 170 215 100 119 170 141 248 69 107 210 79 66 7 64 108

Nonce is

178 207 60 181 247 231 96 123 218 143 45 55 238 11 7 96

Gamma is

90 202 57 78 58 73 34 109 30 249 87 84 151 39 162 120

Counter is

206 52 219 192 68 97 99 73 30 177 240 196 127 188 161 106

CH 6 (Legitimate Cluster Head) Broadcasting Solicitation

CHID is 6

Encrypted Key Ktemp is

y N T ? ? v | ? m - - (d ? ? :

Cert is

119 157 130 115 219 131 243 240

Delta is

11 170 215 100 119 170 141 248 69 107 210 79 66 7 79 111

Nonce is

```
211  85  98  205  147  92  84  78  164  219  56  52  148  91  50  121

Gamma is

90  202  57  78  58  73  34  109  30  249  87  84  151  39  173  123

Counter is

140  128  234  185  12  142  139  178  107  149  86  126  11  156  71  216

---------------------------------------------------------------------

CH 7 (Malicious Entity) Broadcasting Solicitation

CHID is 7

Encrypted Key Ktemp is

n  ?  ^    %  Y  g  ?  +  7  +  n  C  ,  (  )

Cert is

192  119  185  13  125  191  35  129

Delta is

11  170  215  100  119  170  141  248  69  107  210  79  66  7  79  111

Nonce is

197  148  49  122  101  147  16  43  217  169  41  55  39  170  34  190

Gamma is

90  202  57  78  58  73  34  109  30  249  87  84  151  39  173  123

Counter is

110  197  147  16  46  207  196  191  163  131  131  135  167  185  160  152

*********************************************************************

Reply Sent out by SENSOR NODE 1

Sensor Node ID 1

MAC is
```

```
109  192  212  149  9  207  208  215

Chosen CHID is 5

Decrypted Key Ktemp (Just for Information) is

x  c  d  f  p  o  l  k  m  k  l  a  x  c  d  f

Signal Strength Values as received by the SENSOR NODE 1 is

SS from 5  43

SS from 6  65

SS from 7  6

**********************************************************************

Reply Sent out by SENSOR NODE 2

Sensor Node ID 2

MAC is

38  97  10  145  43  198  69  226

Chosen CHID is 6

Decrypted Key Ktemp (Just for Information) is

f  g  s  h  d  l  f  j  k  l  z  o  l  f  g  s

Signal Strength Values as received by the SENSOR NODE 2 is

SS from 5  36

SS from 6  34

SS from 7  6
```

APPENDIX E

CENTRALIZED SECURE CLUSTER FORMATION PROTOCOL WITH EXTENSIONS

```
CH 5 Broadcasting Solicitation

CHID  5

CHID Attribute 101

Nonce is

233  165  143  61  102  208  58  100  154  52  129  153  222  145  243  52

CH 5 MAC is

181  93  216  238  192  181  55  247

---------------------------------------------------------------------

BS replying to CH 5 with its ABK

Counter is

162  24  132  172  61  66  137  8  105  26  135  146  162  17  76  122

Nonce is

233  165  143  61  102  208  58  100  154  52  129  153  222  145  243  52

ABK  Encrypted with KBS-CH is

  S    ‘ ‘    5    /  #  1  %  :  ’  \  |

MAC sent is

81  37  44  2  116  199  73  193

---------------------------------------------------------------------

CH 5 receives the following reply from BS (after decryption)

ABK received is

z  a  o  f  r  k  w  l  d  p  o  d  h  n  v  m

MAC computed is

81  37  44  2  116  199  73  193

*********************************************************************
```

```
CH 6 Broadcasting Solicitation

CHID  6

CHID Attribute 102

Nonce is

26  189  66  122  58  141  100  48  96  165  105  133  156  78  95  8

CH 6 MAC is

57  129  130  201  6  97  66  117

---------------------------------------------------------------------

BS replying to CH 6 with its ABK

Counter is

21  120  197  156  15  136  57  212  245  242  229  173  35  124  106  57

Nonce is

26  189  66  122  58  141  100  48  96  165  105  133  156  78  95  8

ABK  Encrypted with KBS-CH is

?  e  Z  :  |  n  *  y  J  &  ?    Y  A  _  1

MAC sent is

79  116  214  185  19  138  203  123

---------------------------------------------------------------------

CH 6 receives the following reply from BS (after decryption)

ABK received is

q  k  g  h  o  p  l  m  u  f  n  y  o  p  l  z

MAC computed is

79  116  214  185  19  138  203  123

*********************************************************************
```

```
CH 7 Broadcasting Solicitation

CHID  7

CHID Attribute 102

Nonce is

132  202  149  38  168  216  197  79  251  251  19  254  230  222  46  143

CH 7 MAC is

199  245  157  123  27  14  244  86

---------------------------------------------------------------------

BS replying to CH 7

Counter is

224  132  1  199  61  146  134  253  208  185  98  109  170  80  132  85

Nonce is

132  202  149  38  168  216  197  79  251  251  19  254  230  222  46  143

ABK  Encrypted with KBS-CH is

c  q  !  @  7  -  m  @  A  O  +  &  =  P  >

MAC sent is

56  236  14  185  219  38  3  31

---------------------------------------------------------------------

CH 7 receives the following reply from BS (after decryption)

ABK received is

q  k  g  h  o  p  l  m  u  f  n  y  o  p  l  z

MAC computed is

56  236  14  185  219  38  3  31

*********************************************************************
```

SENSOR NODE 1 Broadcasting Solicitation

Sensor Node ID 1

Sensor Node Attribute 101

Gamma is

100 185 145 238 200 77 233 81 234 198 165 69 178 252 36 228

Nonce is

230 126 3 31 130 126 179 98 214 50 88 152 160 143 245 60

Sensor Node 1 MAC is

124 34 10 210 17 102 16 94

Assigned Cert is

132 42 107 152 171 70 205 42

CH 5 Forwarding SENSOR NODE 1's Solicitation to BS

Sensor Node ID 1

Counter is

242 139 74 148 15 123 56 189 228 53 107 9 249 202 187 222

Nonce is

230 126 3 31 130 126 179 98 214 50 88 152 160 143 245 60

Sensor Node 1 MAC is

124 34 10 210 17 102 16 94

CHID is 5

SS is 29

KCH-N in Encrypted Form is

26 82 23 143 143 204 52 247 157 244 248 82 247 89 163 128

Cluster Head 5 MAC is

9 194 75 27 117 163 16 96

CH 6 Forwarding Node 1's Solicitation to BS

Sensor Node ID 1

Counter is

51 189 71 246 63 242 169 94 149 15 245 144 147 182 133 17

Nonce is

230 126 3 31 130 126 179 98 214 50 88 152 160 143 245 60

Sensor Node 1 MAC is

124 34 10 210 17 102 16 94

CHID is 6

SS is 48

KCH-N in Encrypted Form is

2 208 95 245 194 149 228 5 185 39 41 163 131 149 217 72

Cluster Head 6 MAC is

184 185 58 201 63 72 117 25

CH 7 Forwarding Node 1's Solicitation to BS

Sensor Node ID 1

Counter is

49 59 113 119 221 97 7 80 3 152 30 16 203 168 41 156

Nonce is

230 126 3 31 130 126 179 98 214 50 88 152 160 143 245 60

```
Sensor Node 1 MAC is

124  34  10  210  17  102  16  94

CHID is   7

SS is   35

KCH-N  in Encrypted Form is

199  50  182  208  110  64  199  110  118  120  119  134  42  8  248  36

Cluster Head 7 MAC is

108  225  143  200  218  119  90  140

**********************************************************************

SENSOR NODE 2 Broadcasting Solicitation

Sensor Node ID  2

Sensor Node Attribute 103

Gamma is

208  223  20  236  84  215  159  57  10  30  218  66  52  108  140  21

Nonce is

5  219  32  9  98  157  230  249  59  54  235  178  48  7  134  229

Sensor Node MAC is

234  107  54  137  24  167  32  141

Assigned Cert is

156  24  200  221  213  84  103  20

----------------------------------------------------------------------

CH 5 Forwarding SENSOR NODE 2's Solicitation to BS

Sensor Node ID  2

Counter is
```

209 69 218 189 189 13 254 251 187 116 151 10 225 91 244 189

Nonce is

5 219 32 9 98 157 230 249 59 54 235 178 48 7 134 229

Sensor Node 2 MAC is

234 107 54 137 24 167 32 141

CHID is 5

SS is 38

KCH-N in Encrypted Form is

90 129 62 216 111 66 180 185 243 96 35 85 35 47 40 44

Cluster Head 5 MAC is

39 214 205 66 3 119 3 13

CH 6 Forwarding SENSOR NODE 2's Solicitation to BS

Sensor Node ID 2

Counter is

164 17 144 146 210 193 208 52 78 129 20 247 214 124 66 243

Nonce is

5 219 32 9 98 157 230 249 59 54 235 178 48 7 134 229

Sensor Node 2 MAC is

234 107 54 137 24 167 32 141

CHID is 6

SS is 30

KCH-N in Encrypted Form is

47 102 57 158 181 177 216 48 16 17 115 199 73 50 231 145

```
Cluster Head 6 MAC is
229  184  191  44  84  164  56  71
-----------------------------------------------------------------
CH 7 Forwarding SENSOR NODE 2's Solicitation to BS
Sensor Node ID  2
Counter is
28  214  152  179  103  244  35  140  95  22  130  165  155  131  64  193
Nonce is
5  219  32  9  98  157  230  249  59  54  235  178  48  7  134  229
Sensor Node 2 MAC is
234  107  54  137  24  167  32  141
CHID is   7
SS is   32
KCH-N  in Encrypted Form is
200  131  12  16  183  129  76  230  188  124  113  75  234  225  164  165
Cluster Head 7 MAC is
12  219  228  71  183  224  110  228
*****************************************************************
BS replying to SENSOR NODE 1
Cluster Head assigned to Sensor Node 1 is  5
Counter is
173  127  181  253  140  30  230  123  123  12  77  29  174  44  95  168
Nonce is
230  126  3  31  130  126  179  98  214  50  88  152  160  143  245  60
```

KCH-N in Encrypted with KBS-N is

a 0 ? # " + 7 S I { {) F ? & p

ABK in Encrypted with KBS-N is

I) * c D ^ 0 Y V O ! C ! @ #

MAC sent is

97 119 210 1 15 75 12 169

SENSOR NODE 1 receives the following reply from BS (after decryption)

Cluster Head assigned to Sensor Node 1 is 5

KCH-N received is

x c d f p o l k m k l a x c d f

ABK received is

z a o f r k w l d p o d h n v m

MAC computed is

97 119 210 1 15 75 12 169

****

BS replying to SENSOR NODE 2

Cluster Head assigned to Sensor Node 2 is 6

Counter is

103 150 156 118 222 214 122 139 42 205 217 229 112 192 252 225

Nonce is

5 219 32 9 98 157 230 249 59 54 235 178 48 7 134 229

KCH-N in Encrypted with KBS-N is

? + s v + ? + < K > . - ?

ABK in Encrypted with KBS-N is

@ c < 1 % P c ^ x y - | V \ { >

MAC sent is

244 78 100 160 184 65 237 131

SENSOR NODE 2 receives the following reply from BS (after decryption)

Cluster Head assigned to Sensor Node 2 is 6

KCH-N received is

f g s h d l f j k l z o l f g s

ABK received is

h j q h l l o i b x z g b d s q

MAC computed is

244 78 100 160 184 65 237 131

APPENDIX F

DISTRIBUTED SECURE CLUSTER FORMATION PROTOCOL WITH EXTENSIONS

```
CH 5 Broadcasting Solicitation

CHID  5

CHID Attribute 101

Nonce is

203  144  207  183  148  62  15  112  101  60  117  209  230  164  179  231

CH 5 MAC is

251  77  199  120  135  44  252  152

---------------------------------------------------------------------

BS replying to CH 5

Counter is

162  24  132  172  61  66  137  8  105  26  135  146  162  17  76  122

Nonce is

203  144  207  183  148  62  15  112  101  60  117  209  230  164  179  231

ABK  Encrypted with KBS-CH is

O  <  a  X  j  1  ^  <  ,  p  0  u  J  *  )  V

MAC sent is

219  120  108  242  29  86  94  167

---------------------------------------------------------------------

CH 5 receives the following reply from BS (after decryption)

ABK received is

z  a  o  f  r  k  w  l  d  p  o  d  h  n  v  m

MAC computed is

219  120  108  242  29  86  94  167

*********************************************************************
```

CH 6 Broadcasting Solicitation

CHID 6

CHID Attribute 102

Nonce is

7 197 142 114 140 79 185 254 45 51 48 210 174 20 78 194

CH 6 MAC is

57 129 130 201 6 97 66 117

BS replying to CH 6

Counter is

74 83 189 33 159 232 50 160 163 38 189 142 135 118 144 93

Nonce is

7 197 142 114 140 79 185 254 45 51 48 210 174 20 78 194

ABK Encrypted with KBS-CH is

@ C] ~ ‘ 3 R 9 % % N

MAC sent is

209 48 141 240 45 39 168 169

CH 6 receives the following reply from BS (after decryption)

ABK received is

q k g h o p l m u f n y o p l z

MAC computed is

209 48 141 240 45 39 168 169

****

```
CH 7 Broadcasting Solicitation

CHID  7

CHID Attribute 102

Nonce is

38  151  129  156  33  230  214  57  82  64  170  229  164  70  81  169

CH 5 MAC is

222  49  174  26  163  153  151  123

---------------------------------------------------------------------

BS replying to CH 7

Counter is

52  164  62  3  16  78  205  30  124  151  246  88  133  152  121  162

Nonce is

38  151  129  156  33  230  214  57  82  64  170  229  164  70  81  169

ABK  Encrypted with KBS-CH is

!  V  G  }  ‘  #  @  )  3     (  4  "  <  O  Z

MAC sent is

215  19  54  151  14  203  219  197

---------------------------------------------------------------------

CH 7 receives the following reply from BS (after decryption)

ABK received is

q  k  g  h  o  p  l  m  u  f  n  y  o  p  l  z

MAC computed is

215  19  54  151  14  203  219  197

*********************************************************************
```

CH 5 Broadcasting Solicitation

CHID is 5

Encrypted Key Ktemp is

_] a $ $ (h Y % # ; | ^ / w ?

Cert is

183 245 19 88 70 4 64 46

Delta is

250 167 11 192 245 25 4 241 215 39 240 53 10 37 11 82

Nonce is

201 154 73 91 213 211 137 9 189 105 58 222 117 224 125 104

Gamma is

151 89 78 198 85 24 57 92 56 194 203 46 9 119 233 87

Counter is

228 134 101 29 229 120 9 169 228 161 107 158 152 75 237 174

CH 6 Broadcasting Solicitation

CHID is 6

Encrypted Key Ktemp is

[3 Z 9 ‘ { k Z ^ e 0 % l m #

Cert is

148 18 81 162 127 83 55 251

Delta is

173 122 37 19 9 173 204 244 210 217 138 188 120 189 32 165

Nonce is

28 237 5 34 163 14 127 253 13 164 181 83 252 181 249 11

Gamma is

105 51 123 81 192 145 189 153 74 127 140 180 58 12 192 8

Counter is

73 99 83 53 17 18 16 89 125 134 11 108 254 141 233 89

CH 7 Broadcasting Solicitation

CHID is 7

Encrypted Key Ktemp is

‘ 2 5 " ^ l % ^ @ # ^ : < # r

Cert is

193 40 12 44 133 159 156 134

Delta is

75 79 98 121 73 130 23 140 156 154 212 113 210 138 241 248

Nonce is

62 154 57 204 53 111 57 92 20 188 167 76 108 14 51 126

Gamma is

195 224 124 80 209 128 153 15 58 53 226 247 11 159 107 116

Counter is

148 52 43 95 41 190 231 23 65 188 245 76 199 161 25 169

****

Reply Sent out by SENSOR NODE 1

Sensor Node ID 1

MAC is

```
156  33  14  140  110  158  133  47

Chosen CHID is 6

Decrypted Key Ktemp (Just for Information) is

f  g  s  h  d  l  f  j  k  l  z  o  l  f  g  s

Signal Strength Values as received (Just for Information) by the SENSOR NODE 1 is

SS from 5  56

SS from 6  34

SS from 7  83

**********************************************************************

Reply Sent out by SENSOR NODE 2

Sensor Node ID 2

MAC is

94  55  58  1  180  254  141  52

Chosen CHID is 5

Decrypted Key Ktemp (Just for Information) is

x  c  d  f  p  o  l  k  m  k  l  a  x  c  d  f

Signal Strength Values as received (Just for Information) by the SENSOR NODE 2 is

SS from 5  29

SS from 6  81

SS from 7  44

**********************************************************************

SENSOR NODE 1 Broadcasting Solicitation

Sensor Node ID  1

CHID Attribute 101
```

```
Nonce is
14  217  2  141  20  74  220  247  72  45  225  191  94  155  205  22
Sensor Node 1 MAC is
113  57  244  220  53  123  199  96
-----------------------------------------------------------------
CH 6 Forwarding SENSOR NODE 1's Solicitation to BS
Sensor Node ID  1
Nonce is
14  217  2  141  20  74  220  247  72  45  225  191  94  155  205  22
Sensor Node 1 MAC is
113  57  244  220  53  123  199  96
CHID is   6
Cluster Head 6 MAC is
191  213  1  196  85  149  200  129
*****************************************************************
SENSOR NODE 2 Broadcasting Solicitation
Sensor Node ID  2
CHID Attribute 103
Nonce is
108  229  97  182  79  190  63  248  181  196  39  197  66  183  88  221
Sensor Node 2 MAC is
138  198  65  28  48  189  40  103
-----------------------------------------------------------------
CH 5 Forwarding SENSOR NODE 1's Solicitation to BS
```

```
Sensor Node ID  2

Nonce is

108  229  97  182  79  190  63  248  181  196  39  197  66  183  88  221

Sensor Node 1 MAC is

138  198  65  28  48  189  40  103

CHID is   5

Cluster Head 6 MAC is

80  168  236  225  226  41  31  215  170  95  162  142  206  165  15  170

**********************************************************************

BS replying to SENSOR NODE 1

Counter is

20  139  75  120  241  34  31  244  128  25  225  220  152  200  111  193

Nonce is

14  217  2  141  20  74  220  247  72  45  225  191  94  155  205  22

ABK  in Encrypted with KBS-N is

X  H  b    w  e  #  ‘  1  4  _  -  ;  (  ?  )

MAC sent is

126  95  157  96  188  43  45  192

----------------------------------------------------------------------

SENSOR NODE 1 receives the following reply from BS (after decryption)

ABK received is

z  a  o  f  r  k  w  l  d  p  o  d  h  n  v  m

MAC computed is

126  95  157  96  188  43  45  192
```

```
**********************************************************************
BS replying to SENSOR NODE 2
Counter is
110  25  2  157  57  229  214  125  24  82  111  141  135  107  66  144
Nonce is
108  229  97  182  79  190  63  248  181  196  39  197  66  183  88  221
ABK  in Encrypted with KBS-N is
X  H  b     ?   &  7    0    5  /  ;  (  |  )
MAC sent is
38  91  253  193  73  161  44  11
----------------------------------------------------------------------
SENSOR NODE 2 receives the following reply from BS (after decryption)
ABK received is
h  j  q  h  l  l  o  i  b  x  z  g  b  d  s  q
MAC computed is
38  91  253  193  73  161  44  11
```