UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

DATA AGGREGATION USING HOMOMORPHIC ENCRYPTION IN WIRELESS SENSOR NETWORKS

by

Tsotsope Daniel Ramotsoela

Submitted in partial fulfilment of the requirements for the degree
Master of Engineering (Computer Engineering)

in the

Department of Electrical, Electronic and Computer Engineering
Faculty of Engineering, Built Environment and Information Technology

UNIVERSITY OF PRETORIA

October 2015

SUMMARY

DATA AGGREGATION USING HOMOMORPHIC ENCRYPTION IN WIRELESS SENSOR NETWORKS

by

Tsotsope Daniel Ramotsoela

Supervisor:	Dr. G.P. Hancke
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Computer Engineering)
Keywords:	wireless sensor networks, homomorphic encryption, network security, aggregation

Wireless sensor networks have become increasingly popular in many applications such as environment monitoring and law enforcement. Data aggregation is a method used to reduce network traffic but cannot be used together with conventional encryption schemes because it is not secure and introduces extra overhead. Homomorphic encryption is an encryption scheme that allows data processing on encrypted data as opposed to plaintext. It has the benefit that each intermediate node does not have to decrypt each packet, but the resulting cyphertext is usually much larger than the original plaintext. This could negatively affect system performance because the energy consumption of each node is directly proportional to the amount of data it transmits.

This study investigates the benefits and drawback of using homomorphic encryption in the aggregation process particularly in the context of scalable networks. It was found that conventional encryption outperforms the homomorphic encryption for smaller networks, but as the network size grows, homomorphic encryption starts outperforming conventional encryption. It was also found that the homomorphic encryption scheme does significantly reduce the performance of plaintext aggregation. This performance reduction will however be acceptable for most applications where security is a concern.

OPSOMMING

DATA-AGGREGASIE IN DRAADLOSE SENSORNETWERKE MET BEHULP VAN HOMOMORFIESE ENKRIPSIE

deur

Tsotsope Daniel Ramotsoela

Studieleier:	Dr G.P. Hancke
Departement:	Elektriese, Elektroniese en Rekenaaringenieurswese
Universiteit:	Universiteit van Pretoria
Graad:	Magister in Ingenieurswese (Rekenaar Ingenieurswese)
Sleutelwoorde:	draadlose sensornetwerke, homomorfiese enkripsie, network-sekuriteit, aggregasie

Draadlose sensornetwerke raak toenemend meer gewild vir heelwat verskillende toepassings, soos byvoorbeeld opgewingsmonitering en wetstoepassing. Data-aggregasie is ‘n metode wat gebruik word om netwerkverkeer te verminder, maar kan nie gebruik word saam met konvensionele enkripsie-skemas nie, omdat dit nie veilig is nie en oorhoofse koste verhoog. Homomorfiese enkripsie is ‘n enkripsie-skema wat dataverwerking toelaat op geënkripteerde in teenstelling met gewone-teks. Dit het die voordeel dat elke intermediêre nie nodig het om elke pakkie te dekripteer nie, maar die resulterende kodeteks is gewoonlik heelwat groter as die gewone-teks. Dit kan die stelselgedrag negatief beÏnvloed omdat die energieverbruik van elke node eweredig is aan die hoeveelheid data wat dit versend.

Hierdie studie ondersoek die voor- en nadele van homomorfiese enkripsie in die aggregasieproses, veral in die konteks van skaleerbare netwerke. Daar is gevind dat konvensionele enkripsie beter vaar as homomorfies enkripsie in kleiner netwerke. Die omgekeerde is waar vir groter netwerke. Dit is ook gevind dat homomorfiese enkripsie gewone-teks-aggregasie negatief beÏnvloed, maar dit word as aanvaarbaar beskou vir toepassings waar sekuriteit belangrik is.

LIST OF ABBREVIATIONS

WSN	Wireless Sensor Networks
NS2	Network Simulator 2
USA	United States of America
IEEE	Institute of Electrical and Electronics Engineers
CIA	Confidentiality, Integrity, Availability
DoS	Denial of Service
IT	Information Technology
MAC	Message Authentication Codes
CDAP	Concealed Data Aggregation Protocol
AES	Advanced Encryption Standard
KSA	Key-Scheduling Algorithm
PRGA	Pseudo-Random Generation Algorithm
WEP	Wired Equivalent Privacy
DES	Data Encryption Standard
NIST	National Institute of Standards and Technology
BFS	Breadth-First Search
BF	Branching Factor
SPF	Shortest Path First
R&D	Research and Development

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 PROBLEM STATEMENT

1.1.1 Context of the problem

Wireless sensor networks (WSNs) have become increasingly popular in many applications such as environment monitoring and law enforcement [1]. The networks consist of a number of cheap sensor nodes which consist of a sensor, a processor, and a power source [2]. The sensor changes depending on the specific application the sensor node is used in. The processor is usually a simple processor with low computational power. The power source is usually a limited power supply such as commercial battery.

These resource constraints mean that the efficiency of data transfer is paramount in these applications. This is because the energy consumption of the nodes is directly proportional to the amount of data transferred [2]. One of the methods that can be used to reduce traffic in the network is called data aggregation. This process involves combining data coming from different sources enroute [3].

Aggregation however becomes a problem when security is an issue in the system [4]. This is because each node would have to decrypt each packet, aggregate the data, and then encrypt the result before sending it to the next hop. Secure information aggregation in WSNs is a growing field and one of the proposed solutions to achieving this is called homomorphic encryption. Homomorphic encryption is an encryption scheme that allows data processing on encrypted data as opposed to plaintext [5].

Homomorphic encryption allows the data to be aggregated without having to decrypt each incoming packet. In homomorphic encryption schemes, for a given encryption key, each plaintext can be encrypted into a number of different cyphertexts [6]. This means the resulting cyphertext is larger than the original plaintext. As already mentioned, the energy consumption of the nodes is directly proportional to the amount of data transferred. It can thus be deduced that the larger the network traffic is, the poorer the system performance will be. So this increase in packet size depletes the power sources of the nodes faster than if encryption was not used

1.1.2 Research gap

Secure information aggregation in wireless sensor networks is a growing field. The literature survey revealed many papers which focused on the computational overhead and security of homomorphic encryption schemes. There are very few papers which actually quantify the effect of this larger packet size even though it could significantly affect the performance of WSNs. Most of the work up to now has focused on the cost to individual nodes and the schemes themselves. None have investigated the benefits or drawback in terms of scalable networks. The proposed investigation aims to fill this gap in this growing research field.

1.2 RESEARCH OBJECTIVE AND QUESTIONS

Using aggregation with conventional encryption schemes requires each node to decrypt each packet, aggregate the data, and then encrypt the result before sending it to the next hop. The problem with this is that it assumes that all the sensor nodes are trusted [7]. This means that data aggregation may not always be appropriate depending on the security requirements of the specific application. This can be combatted by using homomorphic encryption to securely aggregate the data, but this increases the sizes of the packets. The objective of this research is to quantify the effect this has on overall system performance. Taking this into consideration, the following research questions are posed:

- Does using homomorphic encryption in data aggregation significantly reduce system performance in WSNs?
- Is data-centric routing still more efficient than address-centric routing when used in this setting?

1.3 HYPOTHESIS AND APPROACH

The hypothesis for the research is formulated as follows. It is hypothesised that using homomorphic encryption in data aggregation will significantly reduce system performance. It is further hypothesised that for smaller networks, address-centric routing will be more efficient than aggregation while the opposite will be true for larger networks.

The approach to be followed in realising the goals outlined is organised as follows. A popular secure aggregation scheme will be simulated using a simulation platform called Network Simulator 2 (NS2). Baseline schemes will then be simulated on the same platform and compared to the secure aggregation schemes results. These results will then be used to find more generalized results using the Python scripting language. The results will be compared in terms of the number of bytes transmitted by intermediate nodes. A practical experiment will also be conducted on Crossbow TelosB motes. This experiment will investigate the speed and transmission times of the different algorithms. A conclusion will then be reached regarding the feasibility of secure information aggregation using homomorphic encryption in WSNs.

1.4 RESEARCH GOALS

One of goal of this research is to determine whether or not using homomorphic encryption in WSNs significantly affects system performance in the context of aggregation. This research also aims to determine whether or not secure aggregation still performs better then end-to-end encryption with no aggregation when used in this setting. The primary goal is to investigate the feasibility of using homomorphic encryption in WSNs.

1.5 RESEARCH CONTRIBUTION

This research will contribute to the body of knowledge by investigating the feasibility of using homomorphic encryption from a different perspective. While most papers focus on the security of the homomorphic encryption schemes and the overhead they cause on individual devices, this research focuses on their effects on network traffic and overall system performance. The primary reason aggregation is used in WSNs is to reduce network traffic and improve the performance of the system. If homomorphic encryption negates this, there is no advantage to using aggregation in security critical applications.

1.6 OVERVIEW OF STUDY

The rest of this study is organised as follows:

- Chapter 2 presents a comprehensive literature survey of WSNs and their security.
- Chapter 3 describes the experiments that were carried out to get the results.

- Chapter 4 presents of detailed discussion of the obtained results.
- The study is concluded in Chapter 5 and recommendations for future work are presented.

CHAPTER 2 LITERATURE STUDY

2.1 CHAPTER OBJECTIVES

This chapter aims to discuss the fundamental concepts WSN security, focussing specifically on secure information aggregation. The prominent literature in the field will be addressed in this literature study. The chapter objectives are as follows:

- Give a general overview of WSNs.
- Give a general overview of the network security.
- Give a general overview of WSN security.
- Discuss secure information aggregation in WSNs in detail.

2.2 WIRELESS SENSOR NETWORKS

Wireless sensor networks can be defined as “a network of devices, denoted as nodes, which can sense the environment and communicate the information gathered from the monitored field (e.g., an area or volume) through wireless links” [8]. They were proposed by the military of the United States of America (USA) in the 1970’s [9]. It wasn’t until the end of the 20th century that wireless sensor networks started becoming popular in applications not related to the military. This is largely attributed to the advances in the related fields, such as microelectronics and telecommunications.

These networks consist of a number of ideally cheap sensor nodes which consist of the following things [2]:

- A sensor,
- A processor,
- A power source.

The sensor changes depending on the specific application the sensor node is used in. The processor is usually a simple processor that doesn’t have a lot of computational power. The power source is usually a limited power supply such as commercial battery.

One of the key features of WSNs is minimising the power consumption in the nodes [10]. This, along with functions such as managing network protocols and interfacing the sensing and communicating units is the responsibility of the processing unit. In addition to the power constraints, another limitation of WSNs is the computational power of the processing unit [1]. The networks themselves are also limited in terms of bandwidth. This means that the efficiency of data transfer is very important in these applications.

Three of the popular network architectures that apply to WSNs will now be discussed in this paragraph and the next one [10]. The first is the star network which is also known as the single point-to-multipoint. As the name implies, there is a single base station that communicates with multiple nodes. These nodes can only send messages to the base station and not each other. This topology is clearly not robust so it isn't favoured.

The second one is the mesh network which allows nodes to transmit messages to any node within its transmission range and uses multi-hop communications for those that are not in range. A disadvantage of this is the power consumption of the nodes since the major power consumer in a sensor node is the communicating unit. The Hybrid Star-Mesh network is a combination of the previous networks. In this topology, there are designated high power nodes which allow multi-hop communication while the rest of the nodes (which have smaller power supply) do not forward any packets. This is the topology used by ZigBee networking standard.

Since most WSNs are application specific, the network requirements for the different applications are different [9]. This means that the hardware, software and communication protocols are also very different. This makes developing standards for WSNs very difficult but there has been a significant amount of work done in the field [8]. The Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 standard is a big step in the process of standardising the field of WSNs. Now that the basics of WSNs have been discussed, the focus will now shift to the basics of network security. This is to establish a reference point with which to compare the WSNs.

2.3 NETWORK SECURITY

This section looks at the fundamentals of network security in general and not specifically that of WSNs. By doing this, it becomes much easier to deduce whether or not the system is protected well enough in network security terms.

Sun Tzu said: “The art of war teaches us to rely not on the likelihood of the enemy's not coming, but on our own readiness to receive him; not on the chance of his not attacking, but rather on the fact that we have made our position unassailable.”

Sun Tzu was a Chinese military general and when he said this, he was referring to actual war and not network security. However, drawing an analogy to Sun Tzu's statement, the objective of network security can be defined as to create a system that cannot be compromised. While this might not always be practical, it should at least be improbable for this to happen.

The three key objectives of network security are confidentiality, integrity and availability [11]. Confidentiality means that only authorised parties are allowed to access data meaning that unauthorised parties shouldn’t be able to access the information. For instance, for military applications in WSNs, the collected data shouldn’t be available for everybody to see.

Integrity means that no unauthorised parties should be allowed to modify the data. Again looking at military applications in WSNs, an attacker shouldn’t be allowed to modify the data for his/her benefit. An extension of this could be that an attacker can’t masquerade as a legitimate node and feed falsified data into the system. The later refers to a form of integrity called nonrepudiation. So it has to be known with a certainty that the data comes from the specific node and has not been modified in transit [11].

When talking about availability, the implication is that the system/data should be available when required [11]. For health applications in WSNs, if a successful attacker was somehow able to take the system offline, the results could be disastrous. If a doctor is monitoring the patient’s vitals remotely and the system becomes unavailable, if something goes wrong then the patient could die.

An example will now be discussed which shows the importance of all three concepts. The example in question is a military application called Pinptr [12]. What it does is basically estimate the location of a sniper shooting at ally forces. If the enemy can see the data in the network, they will know when their position has been compromised and they will move. If they can feed false data into the system, it will produce incorrect results and the enemy location will remain unknown. The same is true if they make the system unavailable. So clearly, these three concepts can't be ignored in some WSN applications.

These three concepts combine to form what is referred to as the CIA triad. When discussing the security of WSNs, reference will always be made to these three concepts to see how they are affected.

2.4 OVERVIEW OF THE SECURITY OF WIRELESS SENSOR NETWORKS

In this section, an overview of the security concerns in WSNs will be discussed taking into consideration the core concepts of network security. While all three are important in any system, it can be easily seen that integrity and availability are more important than confidentiality in this particular system. This however does not mean that confidentiality can be brushed aside because a loss of confidentiality can also lead to devastating consequences as discussed in the previous section.

The characteristics of WSNs that make them vulnerable to attacks are [13]: (1) they are openly accessible to everyone, (2) security isn't designed into the protocols, (3) they have limited resources so the protective measures that can be implemented are limited, and (4) they are usually deployed in hostile environments. These characteristics make it very difficult to protect WSNs when compared to computer networks.

The attacks against WSNs are categorised as follows [14]:

- **External and internal attacks:** External attacks are those from nodes which are not part of the WSN. Internal attacks are from a compromised node within the WSN.

- **Passive and active attacks:** Passive attacks generally seek to compromise the confidentiality of the system without actively affecting the integrity and availability. Active attacks are the opposite of passive attacks.

- **Mote-class and laptop-class attacks:** Mote-class attacks use nodes which are similar in capability to the nodes of the WSN while laptop-class attacks use more powerful devices.

As discussed previously, one of the most important considerations in WSNs is availability. The efficiency of data transfer was noted as being very important in these applications. This was so that the power consumption of each node was kept as low as possible so that the node is active for much longer periods of time. The availability of a system can also be affected by denial of service (DoS) attacks which will be discussed next.

2.4.1 Denial of Service

DoS attacks attempt to compromise the availability of a system [15]. While confidentiality and integrity are assessed using a binary scale (i.e. they have either been maintained or compromised), availability is a bit more difficult to classify. Using the Pinptr application as an example, the system could be affected in the following ways:

- The system could be completely shut down and the location of the sniper can’t be determined at all.
- The system is extremely slow and the location of the sniper is only determined once all the ally forces have been killed.
- The system is slowed down appreciably and the sniper location is only determined once a significant number of ally forces have been killed.

- The system is noticeably slower but the location of the sniper is determined before any real damage is done.

As can be seen from the previous example, it is very difficult to classify denial of service. It is also clear that denial of service is not necessarily the result of an attack, a fault in the system could also have the same effects. So it is important to classify what is or isn’t acceptable with regards to availability. The designers have to determine a threshold, like the minimum speed the system requires to perform its task effectively. Anything falling outside the bound could then be considered denial of service and corrective measures should then be taken.

WSNs use a layered architecture and DoS attacks can occur in any of those layers [14]. This makes protecting the system against these types of attacks more challenging. Another challenge is that a loss in integrity can also lead to a loss of availability. The attacker can corrupt the data being sent in a strategic way to eventually cause the system to become unavailable.

In smart grids, for example, it has been shown that a small number of compromised meters can be used to orchestrate unobservable attacks [16]. These unobservable attacks can't be detected by bad detection algorithms and can be used to mislead the system operator into making catastrophic decisions which could shut down the entire system. Another example is when an attacker induces collisions by feeding false data into the network possibly causing an exponential back-off of some of the medium access control protocols [17].

The literature survey revealed several other papers that discussed DoS attacks. For example, in [18] they discuss sinkhole attacks and their countermeasures and in [19] they propose a detection scheme for DoS attacks. The authors in [20] propose a scheme to prevent these attacks while in [21] they discuss the different DoS attacks and countermeasures. It is clear that these attacks are quite serious and it is very difficult to defend against them. Designers should always have a way of detecting these attacks and implement the correct countermeasures if possible.

2.4.2 Privacy

The issue of privacy is one that mainly concerns confidentiality when we consider the CIA triad. As already mentioned confidentiality is not as important as the other two concepts but cannot be disregarded. The order of importance in smart grids is availability, followed by integrity and then confidentiality as opposed to normal information technology (IT) networks where the order is reversed [22]. Smart grids use sensor networks to perform their tasks so by extension the same can also be said for WSNs. This shows that in most cases normal network security policies can't be applied to WSNs as they are because the priorities are different.

Confidentiality is still a major concern and loss of confidentiality might lead to catastrophic consequences for the users. Looking at smart grids again, if an attacker can get hold of a user's electricity use patterns they can deduce when the user is at home and when they are not [23]. When the user is at home, the attacker can also further determine the activities of the user such as whether they are sleeping or watching TV. This concept can be extended to a block of flats, where an attacker can deduce the occupancy of the building and also which users are currently in their apartments and which ones are currently out [22].

The above two examples are especially relevant since some researchers ([24], [25]) propose using WSNs in the metering process to get more specific readings about which devices are using the most energy. There are schemes that go as far as suggesting the best time for users to switch on their appliances in order to reduce electricity expenses [26].

The common confidentiality attacks on WSNs are [27]:

- **Eavesdropping:** Because WSNs use an open communication channel, an attacker can monitor the communication in the network. If the packets aren't encrypted, he will easily be able to read the contents. In the Pinptr application, he will know when his position has been compromised.
- **Node tampering:** A compromised node receives packets from other nodes in the network and has the encryption key stored in it. This means that an attacker would be able to view all the content passing through that node.
- **Node replication:** A replicated node can trick other nodes into sending data to it by advertising false routes. This way the attacker can get hold of all the information in the network. He can also drop all those packets to affect the availability of the system, which can also happen in node tampering.

The main issue with trying to protect the confidentiality of WSNs is that they have very limited resources. While public key cryptography has been shown to be feasible for use in

WSNs, they are still too computationally expensive [14]. Symmetric key cryptography is more efficient but the key management schemes that use it are far from ideal. When five popular symmetric encryption schemes (RC4, RC5, IDEA, SHA-1 and MD5) were analysed, the encryption algorithms were found to be more efficient than the hashing algorithms [28]. Choosing the right encryption scheme for a particular application is no easy task and requires a lot of consideration.

2.4.3 Trust

The issue of trust falls under the concept of integrity when we consider the CIA triad. In network security, digital signatures are used as a countermeasure to integrity failure [29]. A digital signature is similar to normal signatures that people use to confirm their identities in contracts and other paperwork.

General control systems assume that only authorised parties are accessing data at an appropriate time and location and the data has not been modified, while the smart grid control system is viewed as operating in an environment of implicit trust [23]. In network security, implicit trust means that parties trust each other because they each share a relationship with some trusted third party [30]. As will be discussed later, WSNs also use this approach with the trusted third party being the base station.

To understand the role of the third party in digital signatures, consider Figure 2.1. The concept of a digital signature will be discussed in more detail later, for now this discussion will focus on a more abstract view of the trust relationship. Assume that Node A is the trusted third party and that the other nodes are the communicating parties. If B wants to send something to D, it attaches its digital signature to the message. When D receives the message, it verifies the identity of the sender by checking the signature.

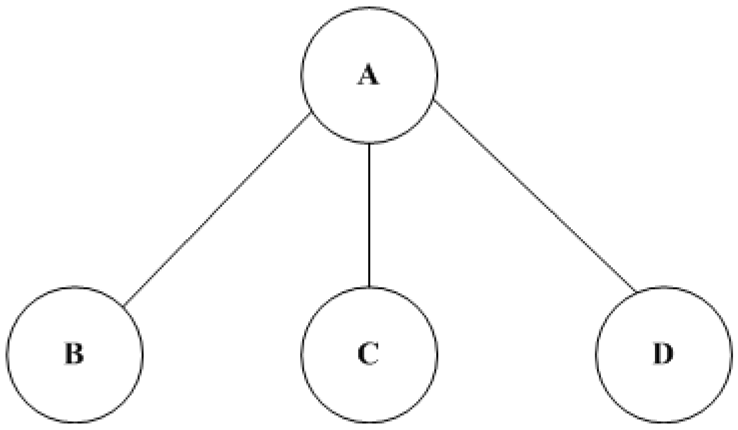

Figure 2.1. Trust relationships

The issue here is how the communicating parties obtain each other's signatures before the actually communication starts. The reason is that an attacker can send D a digital signature claiming to be B and thus D would assume that all subsequent communication with the attacker is with B. Assuming the nodes in Figure 2.1 are employees in company, they can physically give each other the signatures [29]. This however becomes impractical if the company has thousands of employees that all need to communicate with each other. This can be resolved by accompanying each signature some sort of verification from a trusted third party that the signature truly does belong to a valid sender.

Now that there is an abstract model of what a digital signature is and how a trusted third party is involved, digital signatures can be discussed in more detail. A digital signature consists of a file, a demonstration that the file has not been altered, the identity of the signer, and verification that the signature is authentic [29].

To verify that the file has not been changed in transit, a one way cryptographic hash algorithm can be used to find the message digest of the message [29]. The sender then attaches this message digest to the message and sends it to the receiver. When the message arrives at the receiver, the receiver computes the message digest of the message using the same hash algorithm as the sender, and compares it with the received message digest.

Public-key cryptography is used as a means to verify the sender's identity and ensure that the message digest has not been modified in transit [29]. The sender distributes his public key to his peers and encrypts the message digest using his private key. The message digest together with the sender's identity is then attached to the message and the receiver is able to verify that the message has not been modified and that the message is from the sender.

The only issue now is how the sender is able to distribute his public-key to his peers. This can be done by making use of a trusted third party known as a certification authority [29]. The sender obtains a certificate from the certification authority and distributes that to his peers. Looking at Figure 2.1 again, node A would be the certification authority and the other nodes would be in possession of A's public key. When B wants to distribute his public key, he sends it to A together with his identity, a message digest of this message is computed and encrypted using A's private key attached to the message. This can then be distributed to C and D and verified like before.

Before continuing with the discussion it might be important to discuss the issue accountability, sometimes referred to as non-repudiation, in more detail. As the name implies, the sender of a particular message must not be able to repudiate the message. Once the message is signed, the only possible author should be the sender whose public key can be used to check the integrity of the message [29]. It is clear from the above discussion that a digital signature has this property. The only exception is if the private key of the sender has been compromised, but in that case, the integrity of the communication as whole will have been compromised.

As already explained in the previous section, public-key cryptography is still too computationally expensive for WSN applications. Instead, symmetric cryptography is used to verify integrity. A popular scheme used in applications is called µTesla and it uses a key chain of symmetric keys instead of public-key cryptography [31]. Each key in the chain can verify a previous key using a one-way function. Each key is only valid within a certain time interval so it is important that all the nodes in the network are loosely synchronised. The key for a particular interval is only revealed once the interval has passed. The base station is usually responsible for creating the key chain and all the other nodes in the network will trust each other because they trust the base station. The scheme provides an efficient way to maintain integrity in WSNs and there have been several proposed derivatives ([32], [33]) which attempt to improve the scheme.

The common physical attacks on integrity include node compromise and replication [27]. When we consider message related attacks, attackers usually attempt to alter a message they captured in transit or replay an old unaltered message. The latter is referred to as a replay attack and may seem harmless but consider the Pinptr application again. If an attacker has old messages from a previous position calculation, they can fool the system into believing the sniper is at the previously estimated position. The danger of these attacks is that the attacker doesn't need to decrypt the messages. They can just intercept the messages and store them for later use. Since WSNs use an open communication channel, all messages in the network are freely available to the attacker. Another common attack is where the attacker sends a message falsely portraying it as one from a legitimate node in the WSN [27]. This kind of attack is called a counterfeiting attack.

Integrity attacks are also dangerous in that they can cause a loss of availability as explained previously. Replay attacks could also be used to shut down the system by flooding the network with seemingly legitimate messages [27]. By doing this, the resources of the network could become exhausted. This could result in a deterioration of system performance or a total system shutdown. So clearly, replay attacks are just as serious as other integrity attacks and the appropriate countermeasures should always be in place.

2.5 DATA AGGREGATION USING HOMOMORPHIC ENCRYPTION

2.5.1 Data Aggregation

Data aggregation involves combining data coming from different sources enroute [3]. The information is generally being sent to one or more sinks, which is basically a node that collects the information from other nodes in the network. Using data aggregation in this setting has a number of advantages such as reducing the number of transmissions and getting rid of redundancy [3]. One drawback of using aggregation is the latency caused by the processing and possible buffering of data at each node.

To illustrate the importance of aggregation, an example using smart grids will be discussed. In smart grids, aggregation involves combining small groups of residential, commercial or industrial consumers into a larger power unit [34]. We can model this system as a graph with the smart meters being the nodes and the connections being the edges. The smart meters in a particular neighbourhood communicate with a collector device (the sink) which aggregates the data it receives and sends it to a "central manager" [35]. Since each meter in the neighbourhood has to establish a connection to the collector, there will be a lot of redundant connections in the system. This is because neighbourhoods are usually quite large and each node isn't directly connected to the collector node. Figure 2.2 shows a random neighbourhood with only 10 households which illustrates the problems described above.

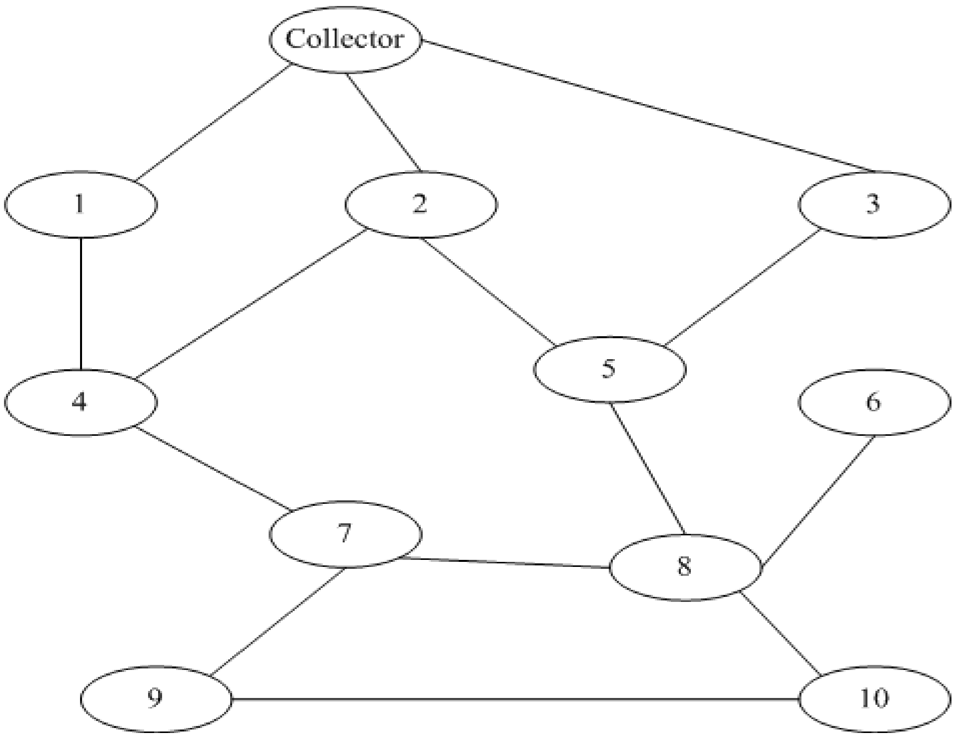

Figure 2.2. Neighbourhood with 10 households

WSNs share the same problem in that there could by thousands of nodes and not all of them will be within communicating distance with each other. In WSNs the sink is usually a base station which is assumed to have an infinite power supply unlike the other nodes in the network. The process described in the previous paragraph is an address-centric protocol [3]. Using this protocol, each node sends data independently via the shortest path to the sink node. While this protocol is efficient in computer networks such as the internet, the same cannot be said for applications such as wireless sensor networks and smart meter communication networks. This is because the mentioned networks use application specific data and the entire process can be optimised by aggregating the data.

Using the address-centric protocol, each node basically acts as a passive router to packets sent from other nodes. This means that it doesn’t perform any operations on the data before it transmits it to the next hop. Using in-network aggregation however, the data is pre-processed before it is transmitted [36]. In this way, the routing occurs along a reversed multicast tree with the sink being the root node. This process is called the data-centric protocol and the tree is referred to as the aggregation tree [3].

Data aggregation (as described) however cannot be implemented in all WSN applications. Applications such as perimeter control rely on individual sensor readings so aggregation is useless in this case [1]. In these applications an address centric approach is usually preferred over a data-centric approach. There are however two approaches to aggregating data, aggregation with size reduction and aggregation without size reduction [37]. The former (described in the previous paragraph) is where the data is combined into one before transmitting it to the next hop. Using the latter approach, each node just appends the received data to its own data and sends it to the next hop as one packet. This approach could be used in applications such as perimeter control since the individual sensor readings are maintained. It has however been found that it is more efficient to use address centric routing in those applications [38].

It is important to note at this stage that the term address-centric routing is being used loosely to explain that type of routing in WSNs. This is because sensor networks don’t use an addressing scheme like IP-addresses used in computer networks [39]. Data-centric routing also has a much broader definition, but for the purposes of this dissertation, both definitions should be considered to be as they are defined in this section.

The time when the aggregation should take place is also critical and two popular timing strategies will now be discussed [40]. In the first strategy, each node waits for a specified time interval before aggregating the data and forwarding the results to the next hop. Using the second strategy, each node aggregates the data only once it has received packets from all of its children. The timing strategy is an integral part of choosing the right protocol so choosing the right one should not be taken lightly.

The discussion so far has been limited to tree based aggregation but there are other types of aggregation in use. The different kinds of aggregation approaches are [37]:

- **Tree based:** Routing occurs along a reversed multicast tree with the sink being the root node.
- **Cluster based:** Similar to trees except that the nodes are divided into clusters.
- **Multipath based:** Instead of sending their results to one parent, each node can send its data to all its neighbours and so the data can travel along multiple paths.

The tree based approach is the most popular of the three mentioned approaches. Using this approach, the aggregation tree can be constructed in a number of different ways depending on the requirements for the particular application. In [36] for example, preserving the power of sensor networks is important so they use that as the primary factor which determines the structure of the aggregation tree. The nodes with the least remaining power are lower down in the tree (shorter waiting time) while those with more power are higher up (longer waiting time).

Some of the common schemes that are used to construct the aggregation tree are [3]:

- **Center at Nearest Source:** The node closest to the sink is chosen to do the data aggregation and all other nodes in the network send their data to that node which then performs the aggregation and forwards the result to the sink.

- **Shortest Paths Tree:** In this scheme, each node in the network sends their data to the sink using the shortest path. In the cases where the paths of different nodes overlap, they are combined to form the aggregation tree.

- **Greedy Incremental Tree:** Here the aggregation tree is built sequentially. The initial tree consists of only the sink and the shortest path to the closest node. Subsequent steps add the paths for the next closest nodes to the aggregation tree.

2.5.2 Homomorphic encryption

Homomorphic encryption is an encryption scheme that allows data processing on encrypted data instead of only on plaintext [5]. The implication of this is that each intermediate meter does not need to decrypt the data in order to perform the aggregation task. An important security feature of this scheme is that for a given encryption key, each plaintext can be encrypted into a number of different cyphertexts [6]. This means that plaintext is shorter than the resulting cyphertext and this difference in length should be chosen to be as small as possible depending on the application. Having different cyphertexts for the same plaintext makes this algorithm resistant to dictionary attacks [35].

The operations that can be performed using the homomorphic encryption scheme are multiplication and addition. A fully homomorphic encryption scheme is one that can perform both types operations on encrypted data [6]. Schemes like the one proposed in [35] can only perform one type of operation at a time and are called either additive or multiplicative schemes.

In [41], they developed a fully homomorphic encryption scheme which was a major breakthrough in the field. It is the first of its kind that has not been broken yet [6]. While this was a step in the right direction, fully homomorphic encryption schemes aren't yet efficient in practice so there are very few applications that implement them. Somewhat fully homomorphic encryption schemes, such as the one proposed in [42], are less complex than the fully homomorphic ones so they are more promising for practical applications. In these schemes, a limited number of multiplication operations are allowed while there is no limit to the number of additions allowed. These schemes might be promising but their overhead is still too high for direct implementation in practical applications [6].

Some of the most popular homomorphic encryption schemes in academia will now be discussed [43]. The ElGamal cryptosystem is multiplicatively homomorphic and is adequately secure. There have been proposed variants that make it additively homomorphic but their decryption is too computationally expensive. The Goldwasser-Micali scheme is also very inefficient when you consider the schemes expansion. The expansion is the ratio of the plaintext to cyphertext size. There have however been variants such as the Benaloh scheme which improves the expansion of the scheme. Arguably the most famous homomorphic encryption scheme is the Pallier cryptosystem. The reason this scheme and its derivatives are so popular is that not only do they provide the same level of security as the ElGamal scheme, but they are very efficient.

The schemes described so far are asymmetric encryption schemes but as already stated, public key cryptography isn't really practical in WSNs. There have been some proposed symmetric encryption schemes but most of them have been broken [43]. A generalization of the one-time pad [1] is one of the few that has not yet been broken. The discussion of the schemes used in WSNs will be left for the next section.

One of the main problems with using symmetric encryption though is that there is only one key used for both encryption and decryption. Using asymmetric encryption, there is a public key which all the nodes in the network have and a private key that only the base station has. In symmetric encryption there is one key so if one of the nodes is compromised, then the attacker is in possession of the key. This means that not only will they be able to feed false data into the network, they will also be able to decrypt all the messages in the network. Using asymmetric encryption however, only the former is possible because deducing the private key from the public key is infeasible [44].

Although asymmetric encryption algorithms are more secure than their symmetric counterparts, the authors in [2] argue that the computational overhead caused by asymmetric encryption is unacceptable in sensor networks. They say that for an attacker who wants to compromise the confidentiality of the system, it is only reasonable to break the mechanism if the cost of breaking it is lower than the value of the revealed information. Since the information exchanged by sensor networks is not usually of extremely high value, they argue that symmetric homomorphic encryption will suffice in those applications. While this may be true for most applications, this is not the case for applications such as the Pinptr. So more secure (but still efficient) symmetric encryption schemes are required if homomorphic encryption is to be implemented practically.

Looking at smart grids again, the information exchanged by smart meters is usually very sensitive, and they have a far superior computational power than sensor nodes. This means asymmetric encryption is usually the preferred choice in these applications. It has been found that the resulting overhead is small and acceptable per smart meter [35].

2.5.3 Data aggregation using homomorphic encryption

Before looking at some of the popular schemes proposed for secure information aggregation in WSNs, it might be useful to discuss why this field is starting to get a lot of attention. The discussion will be for both smart grids and WSNs because the two fields are so closely related. A very brief overview of smart metering systems is first given before considering secure aggregation. Traditional metering devices rely on tamper proof devices located at households and they are physically read by the utility provider every month [45]. Smart meter data on the other hand is remotely read over a much shorter period (e.g. every second). While this creates a much more robust system, it also leads to a number of security concerns as discussed previously. It is for this reason that a number of countries are refusing to make the transition to smart grids.

There are two main choices for smart grid metering architecture [45]:

- **Centralised:** In this scheme, the smart meters are just sensor modules that send their data to a central manager that performs all the tasks such as billing and aggregation. This central manager usually has a much higher computational power than the smart meters.
- **Distributed:** In this scheme, the load is distributed among the smart meters which jointly perform the tasks of the central manager described above.

Traditional smart meters use the centralised architecture, i.e. the aggregation occurs at the collector node. The traditional approach still uses asymmetric encryption, but because the collector does all the aggregation, homomorphic encryption is not used [35]. This means that the collector first has to decrypt all the messages from the smart meters before it can perform the aggregation. The distributed architecture is usually used in self-sufficient grids in rural areas [45].

Some schemes, such as the one proposed in [35], cannot be classified into one of the architectures described above. Since the load is distributed among the smart meters and there is still a central manager, these scheme falls somewhere in between. The centralised scheme relies too heavily on universally trusting the central manager so a more decentralised approach is required [45]. More and more researchers are starting to use the partially decentralised approach where smart meters perform some but not all of the central manager’s tasks. This approach introduces new challenges in that the trust relationships being managed now include the consumers themselves instead of just the relationship between the consumers and producers.

The literature survey revealed that many sensor network aggregation schemes, such as the one proposed in [1], are also using the partially decentralised architecture. However the authors in [7] argue that the problem with this is that most in-network aggregation schemes assume that all the sensor nodes are trusted. This is a problem because they are allowed to view the data that passes through them from other nodes (e.g. they decrypt the data in order to perform the aggregation). They argue that while this could acceptable for some applications, it might not be the case for others. They (and other researchers such as those in [1] and [2]) propose using homomorphic encryption to improve the security of the system.

In the above discussion, the problem is not limited to just trusting the aggregator. Another issue is that each node suffers significant overhead as a result of having to decrypt, aggregate, and then re-encrypting the result before transmission [7]. If the network was very large and each node was aggregating the results of many other nodes, system performance could become affected. Not only could it cause a bottleneck at the aggregator nodes, but the nodes limited supply could become depleted much faster.

Researchers in smart meter communication systems are also starting to use the partially decentralised architecture ([35], [46], and [47]). The Pallier and Castelluccia schemes are two of the popular homomorphic encryption schemes proposed for smart metering systems [45]. The traditional approach has been compared to a proposed homomorphic encryption scheme (which uses a decentralised architecture and the Pallier cryptosystem) to see how it measures up [35]:

- **Network:** The proposed scheme was found to significantly reduce network traffic.

- **Scalability:** The proposed scheme is scalable depending on the network topology. A well designed network makes the approach very scalable. The traditional approach on the other hand is scalable regardless of the network topology.

- **Bottleneck:** The proposed scheme distributes the load to the meters and has measures in place to prevent bottleneck. However, in the traditional approach all the processing happens at the collector and this could cause a major bottleneck especially as the number of meters in the neighbourhood gets large.

- **Computation:** The computational load on the collector node is reduced using the proposed scheme at the cost of extra overhead for each smart meter. The authors however found this overhead to be small an acceptable per smart meter.

Secure data aggregation methods in WSNs can be grouped into two categories, those that perform the aggregation on plaintext, on those that perform the aggregation on cyphertext [48]. The former is usually concerned with preserving the integrity of the data, while the latter focuses more on the confidentiality. There are many popular schemes in existence but only three in each category will be discussed.

2.5.3.1 Aggregation on plaintext data

The authors in [49] propose using a scheme based on µTesla which was described in section IV. In this scheme there is a key chain and keys are only valid within a certain time interval. The message from a particular node will only be verified two hops later when the authentication key is released by the base station.

In [50], they propose a witness based aggregation scheme. In this scheme, each aggregator sends the data it receives to witness nodes which also compute the aggregation. These witness nodes then compute the message authentication codes (MACs) of the aggregated result and sends this result to the base station. The base station uses these MACs from the witnesses to proves that they performed the aggregation properly.

The scheme proposed in [51] doesn’t use any cryptographic operations if all the nodes in the network are honest. If a node is suspected to be cheating, there is a weighted voting process to determine whether or not it is "guilty". If this is the case, the secure aggregation tree is restructured to exclude the possibly compromised node.

2.5.3.2 Aggregation on encrypted data

The authors in [2] propose using concealed data aggregation along a reverse multicast tree. The scheme distinguishes between aggregator nodes and sensor nodes. Sensor nodes are only responsible for collecting the sensing data and aggregator nodes are responsible for aggregating the encrypted data. Since the aggregation process is computationally expensive and the radios of the aggregator nodes are constantly on to receive data from the sensor nodes, the process depletes their power sources quite quickly. The aggregator nodes are thus elected regularly based on the remaining battery life of the batteries. The scheme uses the Domingo-Ferrer encryption algorithm.

In [52], they propose using the concealed data aggregation protocol (CDAP). This scheme uses asymmetric homomorphic encryption. Because asymmetric encryption is too expensive for normal sensor nodes to implement, more powerful aggregator nodes are used to do the aggregation. Each aggregator node shares a secret key with a group of sensor nodes which transmit the data to it using the RC5 symmetric encryption algorithm. When the aggregator node receives the data from the sensor nodes, it decrypts the packets, aggregates them, and encrypts them using the asymmetric homomorphic encryption key. The aggregator node then sends the result to the next aggregator node which aggregates the data from all its children. The process continues until the data gets to the base station, which uses its private key to decrypt the packet.

The authors in [1] propose using a variant of the one-time pad encryption technique. This is a symmetric encryption algorithm which is additively homomorphic. The main advantage of this scheme is that it has been proven to be secure and the encryption process is very efficient. This means that using this scheme is suitable for resource constrained applications such as WSN [14]. Table 2.1 shows a comparison between the discussed algorithms. The last 2 rows rate the efficiency and security in relation to each other with 1 being the best and 3 the worst between the algorithms.

Table 2.1. Comparison of secure aggregation algorithms

Name	Domingo-Ferrer	CDAP	One-time Pad
Type	Symmetric	Asymmetric	Symmetric
Operations	+, -, x	+	+
Probabilistic	yes	yes	yes
WSN Application	Limit security parameters	Use powerful aggregator nodes	As is
Efficiency	2	3	1
Security	2	1	3

CHAPTER 3 METHODS

3.1 OVERVIEW

This chapter provides details of the methods used in this dissertation to obtain the results. The scenarios that were considered are: 1) raw end-to-end data, 2) encrypted end-to-end data, 3) raw aggregate data, and 4) encrypted aggregate data. The simulations were done using Network Simulator 2 (NS2) and the Python scripting language.

The end-to-end encryption scheme used in this experiment is the RC4 encryption algorithm which is popular in WSN applications [14]. Although RC4 has been proven to be vulnerable to security attacks, many applications of small and portable devices still use it because of its speed and efficiency [53]. These applications normally use variants of the algorithm that have improved security features.

The homomorphic encryption scheme that will be used is the Domingo-Ferrer encryption algorithm [54]. It was used in the popular secure aggregation scheme that proposes using concealed data aggregation along a reverse multicast tree [2]. For the purposes of this experiment, the aggregation tree construction will be ignored and the primary focus will be the network traffic.

A practical experiment was also setup to investigate the speed and transmission times on telosB motes. The algorithms that were considered in this experiment were the Advanced Encryption Standard (AES), RC4, and the Domingo-Ferrer cryptosystem. Based on this, the speeds for different network sizes were deduced and analysed. The reason AES was not implemented in the simulations was because it also increases the packet size after encryption. The objective of the simulation was to compare a baseline end-to-end encryption scheme to homomorphic encryption and RC4 was ideal because it doesn't increase the packet size.

The rest of this chapter is organised as follows. The second section describes the RC4 encryption scheme and the third section describes the Domingo-Ferrer homomorphic encryption scheme. The fourth section describes the network topologies used on the different simulation platforms and the fifth section describes all the miscellaneous information about the simulations. The Final section describes the setup of the practical experiment.

3.2 RC4 ENCRYPTION

RC4 is a synchronous stream cipher developed by Ron Rivest which is simple and efficient [55]. The algorithm consists of 2 parts, the key-scheduling algorithm (KSA) and the pseudo-random generation algorithm (PRGA) [56]. It uses a variable length key ($\leq$ 256 bytes) to initialise a 256 byte state vector using KSA. The state vector is then used in the encryption and decryption processes by XORing the data with a pseudo-random keystream generated from the state vector using PRGA.

Making sure the state vectors are synchronised across all nodes in WSNs is difficult in non-point-to-point communication [53]. This can be done by using an initialisation vector (IV) to re-initialise the internal state vectors of the nodes using a process called re-keying. Various algorithms [53], [57] have been proposed to do this safely and efficiently.

Many applications of small and portable devices still use RC4 even though it has been shown to be vulnerable to security attacks [53]. The main reason for this is because of its speed and efficiency. Variants of the algorithm that have improved security features are usually used in these applications. Another reason it is still being used is that none of the proposed attacks are practical if a reasonable key length (e.g. 128 bits) is used [56].

The Wired Equivalent Privacy (WEP) protocol (which uses RC4) is used to provide privacy in the IEEE 802.11 wireless network standard [55]. It has been proven to be vulnerable in a number of areas. The problem however was not with the RC4 algorithm itself, but rather with the key generation process. Algorithms like the one proposed in [56] can be used to remedy this problem.

3.3 DOMINGO-FERRER ENCRYPTION

The Domingo-Ferrer encryption scheme is a homomorphic encryption scheme that is generally too computationally expensive for WSN applications. The authors in [2] however found that limiting the size of the security parameters makes this scheme feasible for practical implementation. This happens at the expense of the security of the scheme, but it was found that it still provides an appropriate level of security.

The public parameters are a large integer g which is 10^{200} or larger and a positive integer d which should be greater than two [54]. The large integer g should have many small divisors and also many integers smaller than it that can be inverted modulo g. The first limitation proposed in [2] is that d should not be greater than 4 and should include the lower bound 2. The second limitation is that g should not be greater than 2^{32}. The secret parameters are a positive integer $r \in Z_g$ (which should be chosen such that $r^{-1} mod\ g$ exists) and a positive integer g' such that $log_{g'} g$ is a secret security parameter [54]. The secret key of the scheme is thus (r, g′).

To encrypt a number $m \in Z_{g'}$, d random numbers (s_1 to s_d) should be generated such that $m = \sum_{j=1}^{d} s_j\ mod\ g'$ and $s_j \in Z_g$. The cyphertext is then found using equation (3.1 below.

$$E(m) = (s_1 r\ mod\ g, s_2 r^2\ mod\ g, \ldots, s_d r^d\ mod\ g) \tag{3.1}$$

To decrypt the cyphertext, the jth coordinate is computed by $r^{-j} mod\ g$ to retrieve $s_j\ mod\ g$. The plaintext is then found using equation (3.2 below.

$$m = D(E(m)) = \sum_{j=1}^{d} s_j\ mod\ g' \tag{3.2}$$

The addition and subtraction operations are done componentwise while the multiplication operation is done by cross multiplying the components in Z_g like polynomials. The division operation is not supported by this scheme [54].

3.4 ADVANCED ENCRYPTION STANDARD

AES was adopted by the USA government as the new federal standard intended to replace the Data Encryption Standard (DES) [58]. It was published by the National Institute of Standards and Technology (NIST) in 2001. It is a symmetric block cypher that operates on block sizes of 128 bits with a key that can be 128, 192, and 256 bits long [59]. The key size determines the number of repetitive operations (rounds) performed on each block (10, 12 and 14 respectively) and the algorithm produces a 128 bits output.

The plaintext is initially organised into a State array, which is a 4x4 array of bytes, and the key is expanded into the required number of round keys [60]. Each round consists of 4 operations, *AddRoundKey* (combine each round key with the state), *SubBytes* (replace each byte with another according to Rijndael's S-Box), *ShiftRows* (shift each row with a certain offset) and *MixColumns* (the 4 bytes in every column are combined using an invertible linear transformation). The initial round only performs the first operation and the final round only performs the first 3 operations.

The original AES algorithm is not usually implemented in WSN applications because it is computationally expensive [59]. It was however found that it is still feasible for use on these wireless devices. There have also been many proposed optimised implementations that improve the algorithms performance [60].

3.5 SIMULATIONS

In this section, the NS2 and Python simulations will be discussed. It is important to note that for this dissertation, the distance of the furthest nodes from the sink is a primary concern. The reasons for this will become evident in the chapters that follow and are only partially discussed in this chapter. The NS2 simulation is a specific implementation and the Python simulation is a generalisation.

3.5.1 NS2

3.5.1.1 Network Topology

The network topology was chosen such that a direct comparison between the data-centric and address-centric protocols can be made. It was also chosen such that the aggregation can be done without having to construct the aggregation tree. The nodes were clustered into groups of four that were all within range of one another. One of the nodes in each cluster was placed within range of a node in another cluster. This will be referred to as the boundary node and is indicated in red in Figure 3.1.

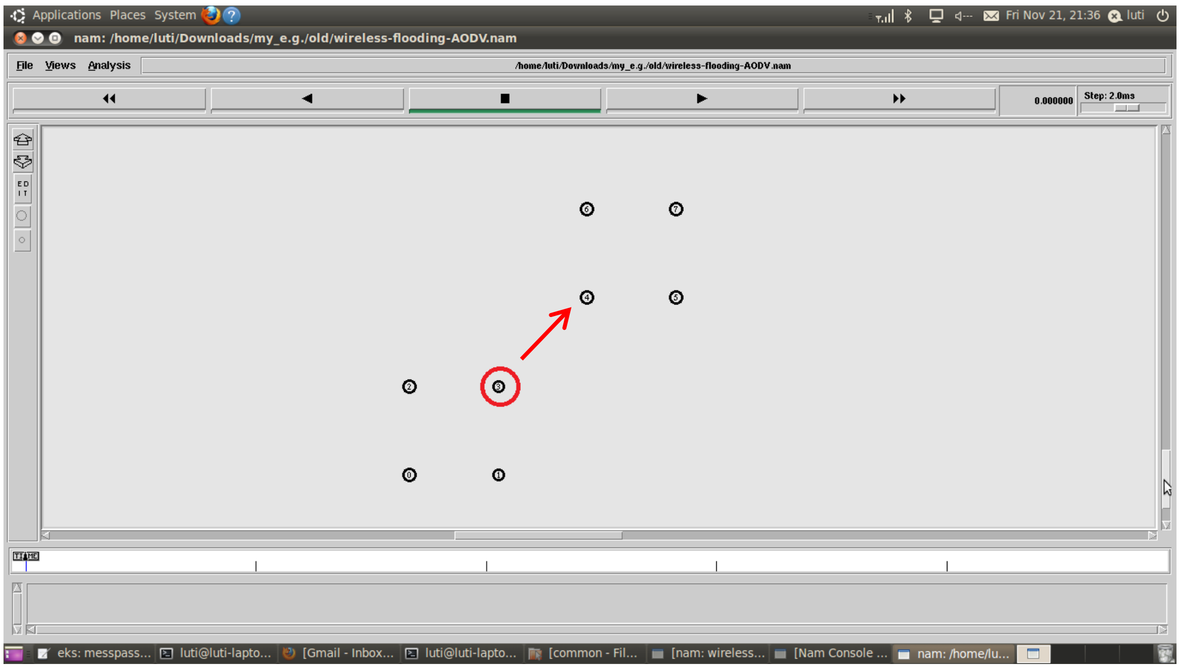
Applications Places System
Fri Nov 21, 21:36
nam: /home/luti/Downloads/my_e.g./old/wireless-flooding-AODV.nam
File Views Analysis
/home/luti/Downloads/my_e.g./old/wireless-flooding-AODV.nam
0.000000
Step: 2.0ms
eks: messpass... luti@luti-lapto... [Gmail - Inbox... luti@luti-lapto... [common - Fil... [nam: wireless... [Nam Console ... nam: /home/lu...

Figure 3.1. Two node clusters

Four network sizes were considered in this experiment: 12, 24, 36 and 48 nodes. These networks had 3, 6, 9 and 12 node clusters respectively. This network configuration was able to simulate what happens as the distance from the sink (in terms of number of hops) increases so it was not necessary to implement larger networks. Using this network configuration, it was also possible to deduce how the network would behave under different circumstances without having to change the network topology.

The network grows vertically as the number of clusters increases. The boundary node of the last cluster is taken to be the sink of the network. It is assumed that the sink does not send any packets but only receives them. For example, in the 12 node network, the boundary node of the 3rd cluster is the sink. For the 24 node network, the boundary node of the 6th cluster is the sink and that of the 3rd cluster is considered just a normal boundary node. From this point forward, the boundary node of the nth cluster will be referred to as boundary node n. The previous explanation is important because while boundary node 3 does not send any packets in the 12 node network, it does in the 24 node network. This distinction will be important in the results section and is illustrated by Figure 3.2 below. In the figure, the sink nodes are indicated in blue and the boundary nodes are indicated in red.

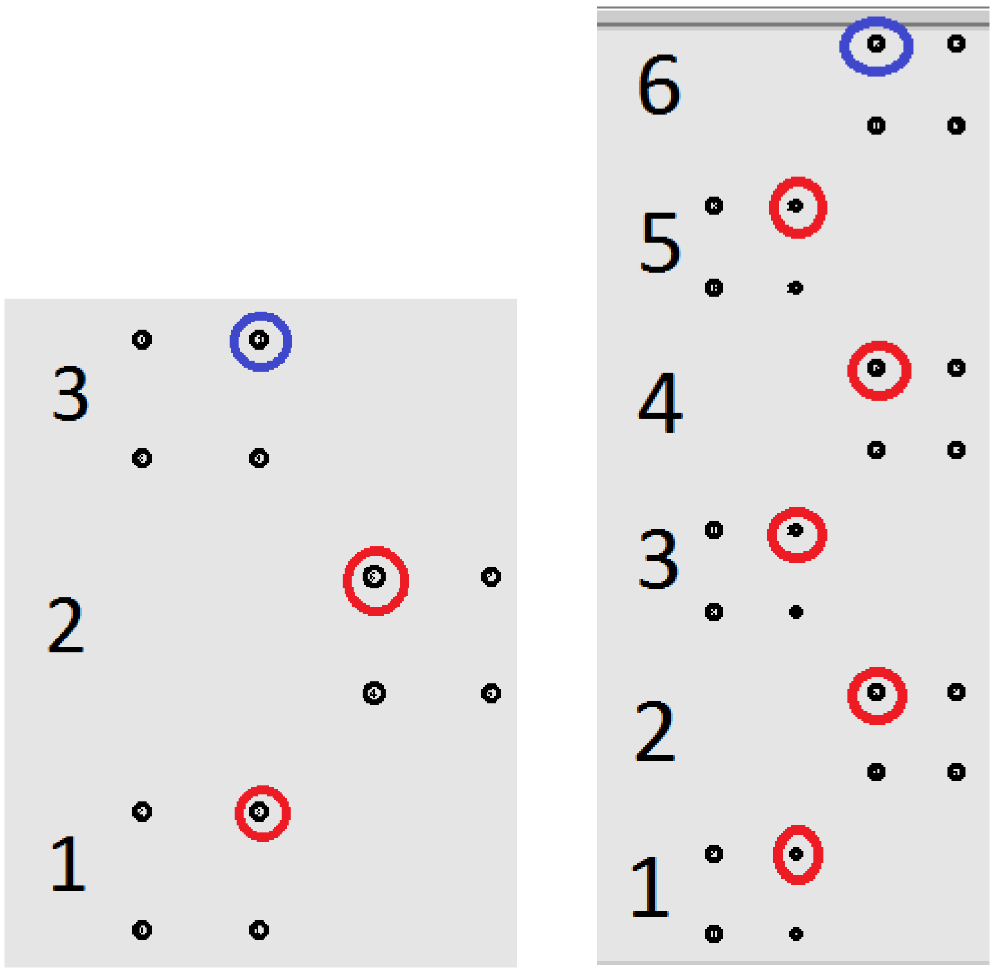

Figure 3.2. Difference between boundary and sink nodes

3.5.1.2 Experimental Setup

For address-centric routing, messages are sent to the sink via the shortest path. For data-centric routing, each node sends its packet to the boundary node in its cluster. The boundary node then aggregates the data of all its children with its own and sends the result to the in range node of the next cluster. This node then aggregates that data with its own and sends it to its own boundary node like all its siblings. The process continues until the sink receives data from all its children.

The network key of the rc4 encryption algorithm is of no significance since it does not affect the size of the packet. The security parameters of the Domingo-Ferrer encryption do affect the size of the packet so they will be mentioned here. The size of the large integer g was chosen to be 232, which is the largest it can be as explained previously. The value of g´ does not affect the packet size, but it was chosen as 216 for this experiment. The value of r, which primarily depends on g, was chosen to be 30027. For the experiment, the packet data was 3 bytes long before encryption and an overhead of 17 bytes was assumed. This is the approximate size of the IEEE 802.15.4 overhead [61].

3.5.2 Python

3.5.2.1 Network Topology

The Python simulations took the results of the NS2 simulation (which depicted the network behaviour) to find more generalised results. As mention previously, a network can be modelled as a graph, so the shortest path of each node can be deduced by looking at the minimum spanning tree (which roots at the sink) obtained using the breadth-first search (BFS) algorithm. In this way each node sends its data to the sink through all the intermediate nodes.

This is also the process followed in aggregation, except that the data is combined at each parent node. The aggregation tree and shortest paths can be found in a number of different ways depending on the specific application and topology of the network. It is for this reason that this generalisation works with a balanced tree to consider ideal situations. Before going into the specifics of balanced trees, it would be beneficial to show why working with non-ideal networks becomes problematic when comparing the two routing algorithms. Consider Figure 3.3 and Figure 3.4 which is a random network modelled as a graph and its corresponding BFS tree respectively.

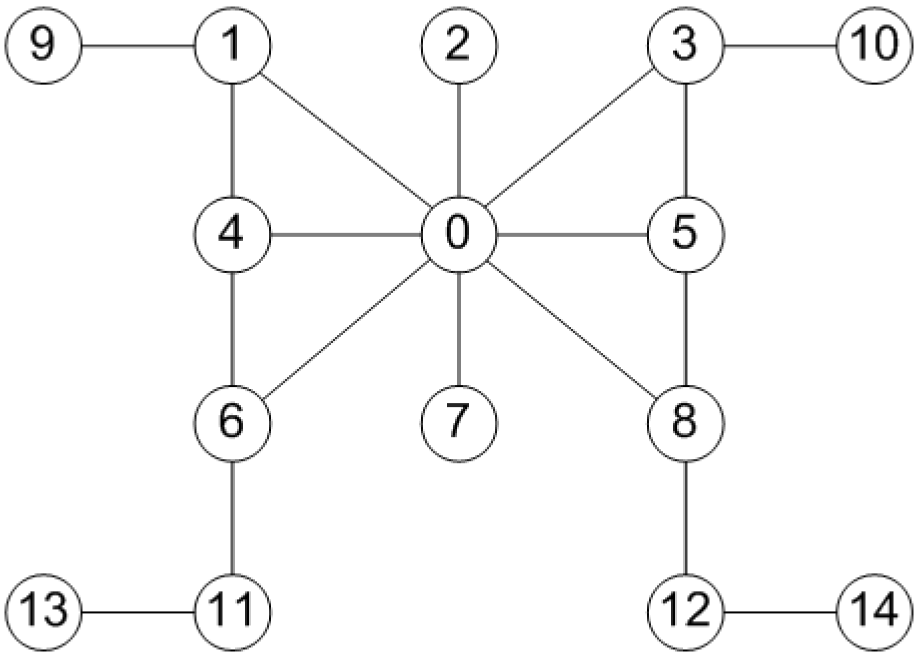

Figure 3.3. Random 15 node network modelled as graph

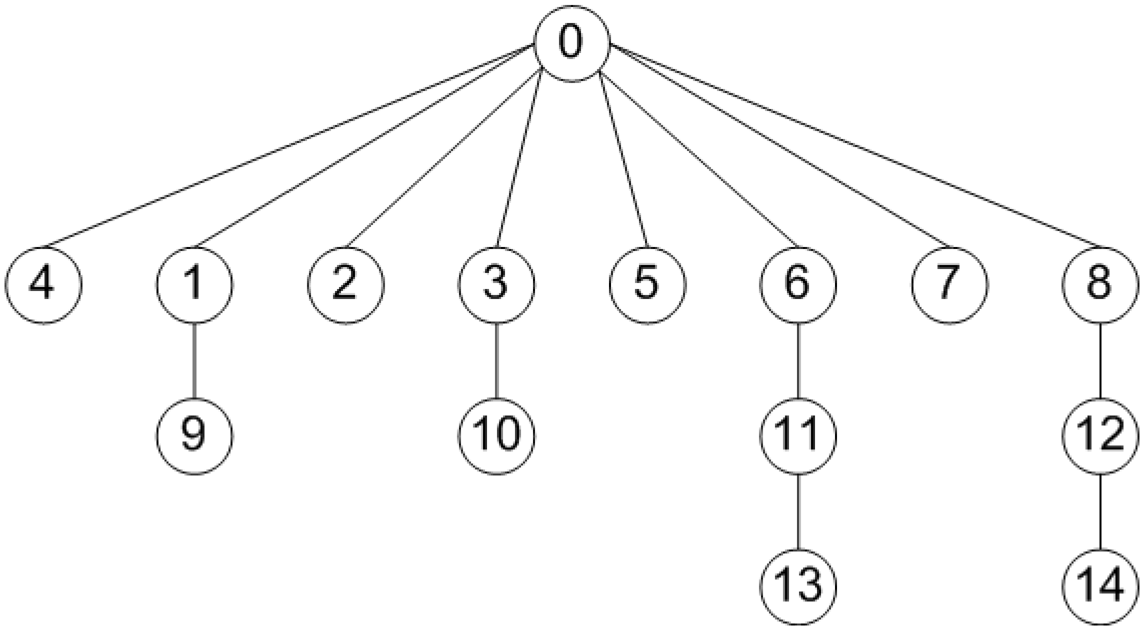

Figure 3.4. BFS tree of random graph

In the random network, most of the nodes are in range with the sink node (node 0). As a result, most of the nodes can send their packets directly to the sink. In this scenario, there is also an unfair distribution of traffic. Four nodes in range with the sink (1, 3, 6, and 8) will have more traffic going through them than other nodes. The other nodes will have the same network traffic as the leaf nodes. In addition to this, nodes 1 and 3 will have the same traffic as nodes 11 and 12 even though the former are only 1 hop away from the sink and the latter are 2 hops away.

From the previous discussion, it is clear that a direct comparison of the network traffic between the nodes is not straight forward. Grouping nodes in terms of their network traffic isn’t straight forward and can be confusing without the aid of the figure. This is because the topology plays a significant role in determining the network traffic through each node. It is impractical to provide a figure for each simulated topology if a number of different scenarios are considered. This problem is amplified considerably as the network size grows (consider 100 nodes).

Another potential issue is the network size, for example, it is still possible to have a 100 node network and for all the nodes to be no further than two hops away from the sink. The same network size can have nodes that are 20, 50 or even 90 hops away from the sink. Comparing such varying network sizes and topologies is thus not straight forward, especially since it is possible for parts of the network to be more populated than others. It is also not practical to look at each topology in isolation because of the number of the large number possible topologies for each network size.

It is thus necessary to use a generalised topology to enable a direct comparison of the network traffic between different nodes. The topology also should simplify the process of grouping nodes that have the same network traffic through them. It is for this reason that a perfectly balanced tree was chosen for this simulation. In the balanced trees used in this simulation, all the nodes were required to have the same number of children and the different branches of the tree were required to have the same height.

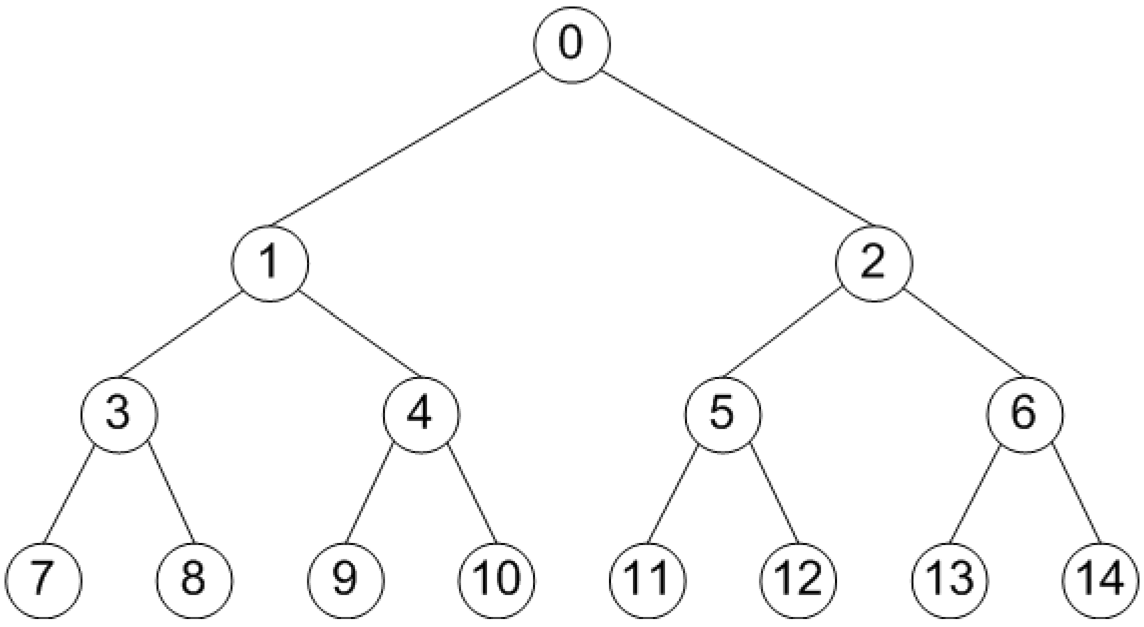

Figure 3.5. Balanced tree with 15 nodes

Figure 3.5 is a 15 node balanced tree with a branching factor (BF) of two. In this case it is easy to group nodes that have the same traffic going through them because all the nodes at that are the same distance from the sink will have the same network traffic going through them. The comparison between data-centric and address-centric routing also becomes simple because of this grouping. It is also important to note that, using this setup, it becomes possible to simulate what happens when the network size grows without having to simulate very large networks.

3.5.2.2 Experimental Setup

For this simulation, the behaviour of the network as observed from the NS2 simulation was used to find more generalised results. Four scenarios were considered in this simulation:

- A 2047 node network with a BF of 2 and a height of 10.
- A 3280 node network with a BF of 3 and a height of 8.
- A 5461 node network with a BF of 4 and a height of 6.
- A 3906 node network with a BF of 5 and a height of 5.

In these scenarios it is possible to see what happens in all the networks that have the same BF. The behaviour of networks that have a larger height can also be deduced from the results. As the branching factor increases, it can be seen what happens when the network grows but the furthest nodes are the same distance away. In this way the network behaviour can be observed without having to consider too many topologies.

3.6 PRACTICAL EXPERIMENT

3.6.1 Experimental Setup

This practical experiment was setup to investigate the encryption and transmission times on physical Wireless Sensor Nodes. The algorithms that were considered in this experiment were AES, RC4, and the Domingo-Ferrer cryptosystem. Based on the observed results, the time taken for all the packets in the network to reach the sink were deduced and analysed for different network sizes.

This experiment was conducted on Crossbow TelosB motes [62]. The mote consists of the MSP430 microcontroller and the CC2420 radio chip. It has a 16 bit processor with 10 kBytes of RAM and a 48 kBytes Program Flash Memory. It also has a 250 kbps high data rate radio and the RF transceiver is IEEE 802.15.4/ZigBee compliant.

The AES implementation used was an optimised implementation based on the Public Domain implementation of Karl Malbrain [63]. The algorithm was considered for all 3 possible key sizes (128, 192 and 256 bits). The RC4 and Domingo-Ferrer algorithms were implemented as described for the simulations.

For each algorithm, the time taken to encrypt and decrypt a packet payload of 3 bytes was observed. The respective encrypted packets were then sent to an in range node and the transmission time was observed. These results were then used to deduce the time it would take for the sink to receive all the packets in the network for varying network sizes based on the Python simulation setup.

The payload encryption can happen concurrently across all nodes. This means that this time can only be counted once for each instance of sending packets to the sink. Packets can also be sent to the sink concurrently for different branches of the tree but a node cannot simultaneously receive multiple packets. This means that the transmission times of the children of each node should be considered independently. However, nodes at the same distance from the sink can receive packets concurrently because they are in different sub-branches of the tree. It is for this reason that the results of only one node at each distance will be considered. The decryption process is only relevant at the sink because no intermediate nodes will decrypt the payloads of other nodes. The only exception is when using the conventional encryption schemes in the aggregation scenario. It is for these reasons that the cryptographic and transmission times were obtained independently of each other. It was also assumed that the sink has the same processing power as the rest of the nodes in the network.

CHAPTER 4 RESULTS

4.1 OVERVIEW

This chapter presents the results of the experiments detailed in the previous chapter. As already explained, the experiments were conducted on two simulation platforms (NS2 and the Python scripting language). The NS2 simulation focused on a specific implementation while the Python simulation depicts network behaviour for general networks. Both simulations consider four scenarios (communication with and without encryption and aggregation). There was also a practical experiment conducted on TelosB motes.

The rest of this chapter is organised as follows. The second section presents the results of the NS2 simulation and the Python simulation results are presented in the third section. The fourth section uses the results of the third section to find a general method for comparing the results for an arbitrary network size. The final section presents the results of the practical experiment.

4.2 NS2 SIMULATION

4.2.1 Address-centric Routing

Table 4.1 shows the results of the NS2 simulation for the address-centric routing. The first column (B-node) indicates the boundary node. It is followed by the plaintext, RC4 and distributed (explained later) columns respectively. Each of the results columns has the number of bytes received and transmitted by each boundary node.

It is now important to take the earlier discussion about the different boundary nodes into consideration. The row named 12 nodes marks the beginning of the 12 node network which has 2 boundary nodes and a sink node. The row named 24 nodes marks the beginning of the 24 node network and the end of the 12 node network. It has 5 boundary nodes and a sink node. It was explained earlier that it is assumed that a sink node does not send any data but a boundary node does. So when analysing the results of the 24 node network, the sink node of the 12 node network is ignored and boundary node 3 is considered instead. So the network traffic of boundary nodes 1 and 2 are identical for both networks. However the sink node in the 12 node network does not send any data, but it is a boundary node in the 24 node network which does send data. So in summary, the nodes of the 12 node network are 1, 2 and sink. In the 24 node network, the nodes are 1, 2, 3, 4, 5 and sink. In the 36 node network, the nodes are 1, 2, 3, 4, 5, 6, 7, 8 and sink. By extension, the same is done for the 48 node network.

Table 4.1. Traffic through boundary nodes in address-centric routing

	plaintext		cyphertext (rc4)		distributed	
boundary node	Receives	Transmits	Receives	Transmits	Receives	Transmits
12 nodes						
1	60	80	60	80	0	20
2	140	160	140	160	20	40
sink	220	0	220	0	220	0
24 nodes						
3	220	240	220	240	40	60
4	300	320	300	320	60	80
5	380	400	380	400	80	100
sink	460	0	460	0	460	0
36 nodes						
6	460	480	460	480	100	120
7	540	560	540	560	120	140
8	620	640	620	640	140	160
sink	700	0	700	0	700	0
48 nodes						
9	700	720	700	720	160	180
10	780	800	780	800	180	200
11	860	880	860	880	200	220
sink	940	0	940	0	940	0

The column named distributed was not the result of an NS2 simulation. By studying the network behaviour of the simulated configuration, the behaviour of a best case scenario for the topology was deduced. This was done because the simulated network used the worst case scenario where the shortest path to the sink is only one route. In the simulated network, only the boundary node of a cluster is within range with only one other node in the next cluster. This meant that all the data of a particular cluster had to pass through the boundary node to get to the next cluster.

In the best case scenario, it is assumed that all the nodes in the cluster can communicate with all the nodes in the next cluster. It is assumed that this communication happens in a distributed fashion such that all the nodes in a cluster receive and transmit the same amount of data. Considering Figure 3.2 again, the nodes in cluster 1 send their packets to different nodes in cluster 2. The nodes in cluster 2 then send the packets of cluster 1 and their own packets to different node in cluster 3 and so on. For the 12 node network, each node in cluster 2 would be able to directly communicate with the sink node. So in this case, all the packets are sent directly to the sink node. This means that the nodes in the same cluster as the sink don't receive any packets from other clusters.

4.2.2 Data-centric Routing

Table 4.2 shows the results of the NS2 simulation for the data-centric routing. The first column (B-node) indicates the boundary node. It is followed by the plaintext, and the Domingo-Ferrer encryption scheme columns respectively. The network was simulated with each of the 3 possible values for d. Each of the results columns has the number of bytes received and transmitted by each boundary node. The structure of the table is the same as for the address-centric protocol.

4.2.3 Combined Results

The energy consumption of the radio is of the same order of magnitude whether it is receiving or transmitting data [59]. This means that a more accurate measure of system performance looks at the net traffic through a node. Table 4.3 shows the combined results of all the net traffic though each boundary for the 48 node network. These results are graphed in Figure 4.1. The figure excludes the non-distributed results for address centric routing because its performance is far worse than the other scenarios. It also excludes the results of the sink which will be discussed from the table.

Table 4.2. Traffic through boundary nodes in data aggregation

	Plaintext		HE (d = 2)		HE (d = 3)		HE (d = 4)	
B-node	Rec	Trans	Rec	Trans	Rec	Trans	Rec	Trans
12 nodes								
1	60	20	111	37	141	47	171	57
2	60	20	111	37	141	47	171	57
sink	60	0	111	0	141	0	171	0
24 nodes								
3	60	21	111	37	141	47	171	57
4	61	21	111	37	141	47	171	57
5	61	21	111	37	141	47	171	57
sink	61	0	111	0	141	0	171	0
36 nodes								
6	61	21	111	37	141	47	171	57
7	61	21	111	37	141	47	171	57
8	61	21	111	37	141	47	171	57
sink	61	0	111	0	141	0	171	0
48 nodes								
9	61	21	111	37	141	47	171	57
10	61	21	111	37	141	47	171	57
11	61	21	111	37	141	47	171	57
sink	61	0	111	0	141	0	171	0

Table 4.3. Net traffic through boundary nodes

	Address		Aggregation			
B-node	RC4	Dist	Plain	HE (d = 2)	HE (d = 3)	HE (d = 4)
1	140	20	80	148	188	228
2	300	60	80	148	188	228
3	460	100	80	148	188	228
4	620	140	80	148	188	228
5	780	180	80	148	188	228
6	940	220	80	148	188	228
7	1100	260	80	148	188	228
8	1260	300	80	148	188	228
9	1420	340	80	148	188	228
10	1580	380	80	148	188	228
11	1740	420	80	148	188	228
sink	940	940	61	111	141	171

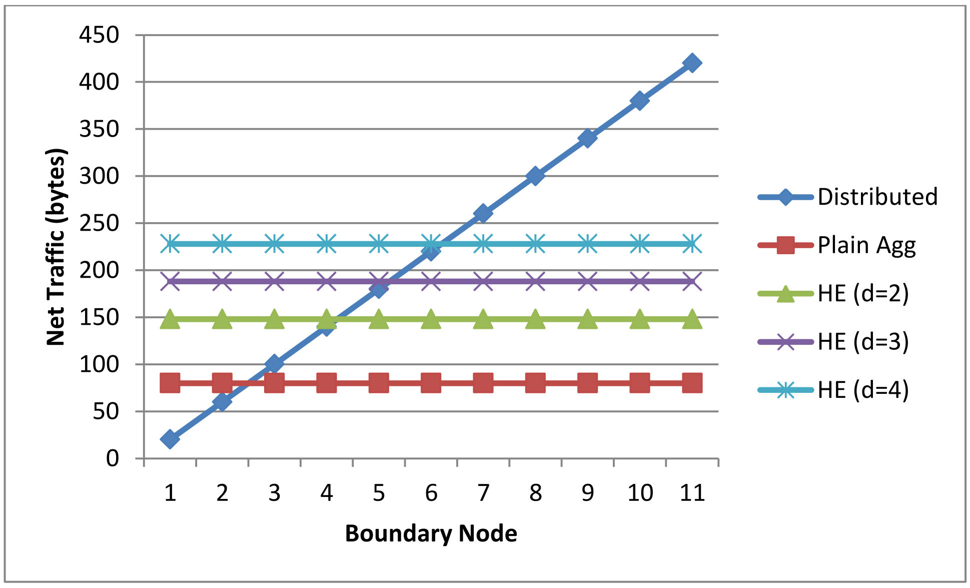

Figure 4.1. Net traffic through boundary nodes

4.3 PYTHON SIMULATION

4.3.1 Experimental Setup

As already mentioned in the previous chapter, this simulation was based on the results of the NS2 simulation. The primary consideration was the packet size of each of the different scenarios which are given in Table 4.4 below. The plaintext aggregation was assumed to be 21 bytes instead of 20 bytes because at some point during the aggregation the packet size increased. The simulation uses balanced trees like the one in Figure 3.5 which is repeated here as Figure 4.2 for convenience.

Table 4.4. Packet sizes of simulated scenarios

Scenario	Packet Size (Bytes)
Address-centric routing	20
Plaintext Aggregation	21
Domingo-Ferrer (d = 2)	37
Domingo-Ferrer (d = 3)	47
Domingo-Ferrer (d = 4)	57

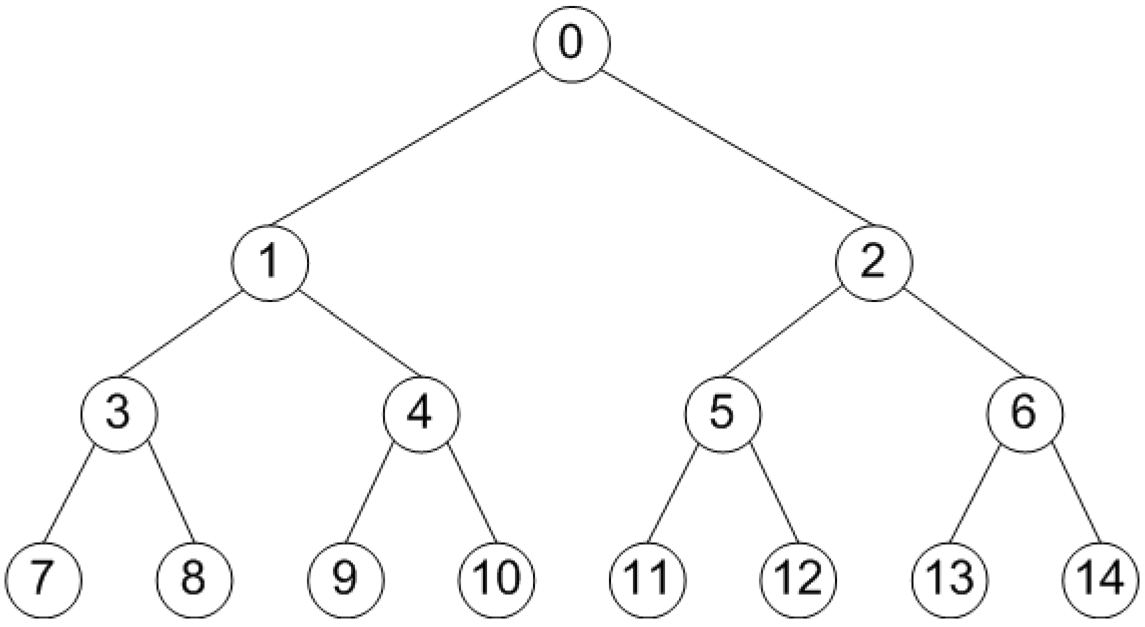

Figure 4.2. Balanced tree with 15 nodes

4.3.2 Results

The following tables show the results of the simulation as follows:

- Table 4.5 shows the results for the 2047 node network (BF = 2, height = 10).
- Table 4.6 shows the results for the 3280 node network (BF = 3, height = 8).
- Table 4.7 shows the results for the 5461 node network (BF = 4, height = 6).
- Table 4.8 shows the results for the 3906 node network (BF = 5, height = 5).

Table 4.5. Net traffic through nodes for the 2047 node network (BF = 2, height = 10)

Distance	SPF	Plain Agg	HE (d = 2)	HE (d = 3)	HE (d = 4)
0	40920	42	74	94	114
1	20460	63	111	141	171
2	10220	63	111	141	171
3	5100	63	111	141	171
4	2540	63	111	141	171
5	1260	63	111	141	171
6	620	63	111	141	171
7	300	63	111	141	171
8	140	63	111	141	171
9	60	63	111	141	171
10	20	21	37	47	57

To understand how the tables are structured consider Figure 4.2 which represents a 15 node network with a BF and height of 2 and 3 respectively. The columns labelled distance on the tables represent all the nodes that are a particular distance (in hops) from the sink. In the figure, node 0 (which is the sink) and is 0 hops away from itself, so it has a distance of 0 in the tables. Nodes 1 and 2 are 1 hop from the sink and nodes 3 – 6 are 2 hops from the sink. The nodes are represented by a distance of 1 and 2 in the tables respectively. The remaining nodes (the leaf nodes) are 3 hops from the sink so they would have a distance of 3 in the tables.

The column labelled SPF represents the shortest path first algorithm. In this scenario, the packets are sent via the shortest path to the sink as explained previously. As already mentioned, the energy consumption of the radio is of the same order of magnitude whether it is receiving or transmitting data. The SPF results only use the transmission values to account for the possibility of using energy conserving algorithms. While this simulation makes comparing the algorithms easier, it assumes there is only one path to the sink and in reality there could be many. In these cases, energy conserving algorithms can be used to distribute the load across as many nodes as possible. The Plain Agg column lists the plaintext aggregation results and the last three columns list the Domingo-Ferrer results.

In addition to the previously mentioned advantages, simulating the network in this way also scales quite well. The results of the 15 node networks can be read off the Table 4.5 because they have the same branching factor. This can be done by reading the table from the bottom up. In this way, the nodes that have a distance 10, 9 and 8 hops in the table have the same net traffic as those that have a distance of 3, 2 and 1 hops respectively in the figure. The sink in the figure would then have the same net traffic as the node with a distance of 7 hops in the table except that you have to subtract the packet size from the total. This is done because the sink doesn't send any data. This is illustrated in Table 4.9.

Table 4.6. Net traffic through nodes for the 3280 node network (BF = 3, height = 7)

Distance	SPF	Plain Agg	HE (d = 2)	HE (d = 3)	HE (d = 4)
0	65580	63	111	141	171
1	21860	84	148	188	228
2	7280	84	148	188	228
3	2420	84	148	188	228
4	800	84	148	188	228
5	260	84	148	188	228
6	80	84	148	188	228
7	20	21	37	47	57

Table 4.7. Net traffic through nodes for the 5461 node network (BF = 4, height = 6)

Distance	SPF	Plain Agg	HE (d = 2)	HE (d = 3)	HE (d = 4)
0	109200	84	148	188	228
1	27300	105	185	235	285
2	6820	105	185	235	285
3	1700	105	185	235	285
4	420	105	185	235	285
5	100	105	185	235	285
6	20	21	37	47	57

Table 4.8. Net traffic through nodes for the 3906 node network (BF = 5, height = 5)

Distance	SPF	Plain Agg	HE (d = 2)	HE (d = 3)	HE (d = 4)
0	78100	105	158	235	285
1	15620	126	222	282	342
2	3120	126	222	282	342
3	620	126	222	282	342
4	120	126	222	282	342
5	20	21	37	47	57

Table 4.9. Net traffic through nodes for the 15 node network (BF = 2, height = 3)

Distance	SPF	Plain Agg	HE (d = 2)	HE (d = 3)	HE (d = 4)
0	280	42	74	94	114
1	140	63	111	141	171
2	60	63	111	141	171
3	20	21	37	47	57

4.4 GENERAL FORMULAE

In this section, the results of the previous section are used to derive formulae that can be used to get results without having to simulate the networks. This is only possible because of the tree structure used in the Python simulations. In this way, if the BF and tree height are known, the number of nodes in the network can be calculated. Furthermore, the net traffic through nodes at each distance can be calculated using the packet size.

4.4.1 Number of Nodes

Looking at Figure 4.2 again, the number of nodes is found using:

$$1 + 2 + 4 + 8 = 15 \tag{4.1}$$

This can be used to find the general formula as follows:

$$2^0 + 2^1 + 2^2 + 2^3 = 15 \tag{4.2}$$

$$\sum_{i=0}^{3} 2^i = 15 \tag{4.3}$$

$$\frac{1 - 2^4}{1 - 2} = 15 \tag{4.4}$$

From the previous derivation, we can make a couple of observation. For a given BF, the number of nodes at a particular distance, d, can be found using:

$$num\_nodes\_at(BF, d) = BF^d \tag{4.5}$$

The number of nodes in a network with a height, h, can be found using:

$$num\_nodes(BF, h) = \frac{1 - BF^{h+1}}{1 - BF} \tag{4.6}$$

This can be verified by looking at the 3906 node network (BF = 5, height = 5):

$$num\_nodes(5,5) = \frac{1 - 5^{5+1}}{1 - 5} = \frac{-15624}{-4} = 3906 \tag{4.7}$$

Equation (4.6) can also be used to find the number of nodes in a particular branch of the tree. For example, in Figure 4.2, each branch of the sink has $\frac{1-2^3}{1-2} = 7$ nodes, which can be generalised as:

$$num\_nodes(BF, h, d) = \frac{1 - BF^{(h-d)+1}}{1 - BF} \tag{4.8}$$

4.4.2 Address-centric Routing

Looking at Figure 4.2 and Table 4.9, the net traffic through nodes at each distance can be found as a function of equation (4.8). This is because once we have the number of nodes in a particular branch, we can calculate the net traffic through the nodes at that distance. For the network in the figure, the net traffic at each node is calculated as follows:

$$traffic_{d=0} = (num\_nodes(2,3,0) - 1) \times 20 = 14 \times 20 = 280 \tag{4.9}$$

$$traffic_{d=1} = num\_nodes(2,3,1) \times 20 = 7 \times 20 = 140 \tag{4.10}$$

$$traffic_{d=2} = num\_nodes(2,3,2) \times 20 = 3 \times 20 = 60 \tag{4.11}$$

$$traffic_{d=3} = num\_nodes(2,3,3) \times 20 = 1 \times 20 = 20 \tag{4.12}$$

The reason one is subtracted from equation (4.9) is because the sink doesn't send any data. The general formula for the net traffic through nodes at a particular distance can thus be derived as follows:

$$traffic(BF,h,d) = num\_nodes(BF,h,d) \times packet\_size, \qquad d > 0 \tag{4.13}$$

$$traffic(BF,h,d) = \frac{1 - BF^{(h-d)+1}}{1 - BF} \times packet\_size, \qquad d > 0 \tag{4.14}$$

$$traffic(BF,h) = \left(\frac{1 - BF^{h+1}}{1 - BF} - 1\right) \times packet\_size, \qquad d = 0 \tag{4.15}$$

The packet size is found using the following formula:

$$packet\_size = overhead + payload \tag{4.16}$$

4.4.3 Aggregation

The formulae for the aggregation are much simpler than the previous ones. As before, we will assume a packet size that is 1 byte longer than SPF. For the plaintext aggregation, the net traffic can be calculated as follows.

$$agg\_packet\_size = packet\_size + 1 \tag{4.17}$$

$$traffic_{leaf} = agg\_packet\_size \tag{4.18}$$

$$traffic_{intermediate} = agg\_packet\_size \times (BF + 1) \tag{4.19}$$

$$traffic_{sink} = agg\_packet\_size \times BF \tag{4.20}$$

For homomorphic encryption, the packet size may vary depending on the percentage increase induced by the particular algorithm. The net traffic can thus be calculated as follows.

$$HE\_packet\_size = packet\_size + \%increase \tag{4.21}$$

$$traffic_{leaf} = HE\_packet\_size \tag{4.22}$$

$$traffic_{intermediate} = HE\_packet\_size \times (BF + 1) \tag{4.23}$$

$$traffic_{sink} = HE\_packet\_size \times BF \tag{4.24}$$

4.5 PRACTICAL EXPERIMENT

4.5.1 Operational Times

Table 4.10 shows the time taken (in microseconds) to complete particular operations for all the encryption algorithms. The column labelled Setup is the time taken to for the mote to boot. During this process, the keys are setup and the radio is switched on. The next 2 columns are the encryption and decryption times and the last column is the column is the time taken to aggregate 2 packet payloads.

As already mentioned, for Domingo-Ferrer, to encrypt a number $m \in Z_{g'}$, d random numbers (s_1 to s_d) should be generated such that $m = \sum_{j=1}^{d} s_j \, mod \; g'$ and $s_j \in Z_g$. This is done before the actual encryption process. What is shown in Table 4.10 is the latter and the encryption time including the generation of the S values is shown in Table 4.11. Because the numbers are randomly generated, the process was repeated 50 times and the statistical results of this are shown in the table.

Table 4.10. Operational times for the implemented encryption algorithms

Algorithm	Setup	Encryption	Decryption	Aggregation
RC4	17157	79	79	26
AES128	3214	1863	2101	26
AES192	3465	2213	2506	26
AES256	3638	2563	2911	26
HE (d=2)	3337	3114	3338	272
HE (d=3)	3637	4661	5031	398
HE (d=4)	3877	6226	6681	524

Table 4.11. Encryption times for the homomorphic encryption (including generation of S)

	HE (d=2)	HE (d=3)	HE (d=4)
Mean	38384202	62001188.66	95186755.94
Median	26482955	47594927.5	81142371.5
Minimum	3936516	1471814	1503551
Maximum	223293993	291446132	370633995

4.5.2 Transmission Times

Table 4.12 shows the time taken (in microseconds) to transmit an encrypted packet from one node to another for all the encryption algorithms. This value changes each time a packet is sent so the process was repeated 50 times and the statistical results of this are shown in the table.

Table 4.12. Transmission times for the implemented encryption algorithms

	RC4	AES	HE (d=2)	HE (d=3)	HE (d=4)
Mean	7289.52	7345.58	7776	8058.26	8383.16
Median	7191.5	7252	7703.5	7964	8300
Minimum	3459	3497	3947	4204	4512
Maximum	12192	12302	12708	12975	13299

4.5.3 Simulations

From the results it is important to take note of a few things. The setup only happens when a node is switched on/reset. So after the first instance of sending packets to the sink, that time should no longer be included in the simulations. As already mentioned, the encryption time is only counted once for each instance of sending packets to the sink and decryption is only applicable at the sink except when convention encryption scheme are used in the aggregation scenario. The final consideration is that the aggregation time is only relevant for the aggregation scenarios so it should be considered in the simulations.

For the address-centric results, the formulae used to calculate the traffic can be derived using the formulae in the previous section. The number of packets received and transmitted can be calculated as follows:

$$received(BF,h,d) = \frac{1 - BF^{(h-d)+1}}{1 - BF} - 1 \tag{4.25}$$

$$transmitted(BF,h,d) = \frac{1 - BF^{(h-d)+1}}{1 - BF} \tag{4.26}$$

This can then be used to calculate the time taken for nodes at a particular distance from the sink to receive and transmit data. From this point forward this value will be referred to as the net time. The net packets through a node can be calculated as follows:

$$net\_packets(BF,h,d) = received(BF,h,d) + transmitted(BF,h,d) \tag{4.27}$$

$$net\_packets(BF,h,d) = 2\left(\frac{1 - BF^{(h-d)+1}}{1 - BF}\right) - 1 \tag{4.28}$$

This can then be used to calculate the net time:

$$net\_time(BF, h, d) = net\_packets(BF, h, d) \times trans\_time \tag{4.29}$$

$$net\_time(BF, h, d) = \left[2\left(\frac{1 - BF^{(h-d)+1}}{1 - BF}\right) - 1\right] \times trans\_time \tag{4.30}$$

To verify this formula, refer back to Figure 4.2. Node 3 is 2 hops from the sink while the tree has a BF of 2 and a height of 3. Using the formula, the net time of the node for RC4 is 36447.6µs. This is because it receives 2 packets and transmits 3 and the average transmission time is 7289.52µs. Node 1, which receives 6 packets and transmits 7 has a net time of 94763.76µs.

To make it easier to calculate the total time of the system, the packets transmitted can be disregarded on the formula can focus entirely on the packets received. This results in the following formula:

$$rec\_time(BF, h, d) = received(BF, h, d) \times trans\_time \tag{4.31}$$

$$rec\_time(BF, h, d) = \left[\left(\frac{1 - BF^{(h-d)+1}}{1 - BF}\right) - 1\right] \times trans\_time \tag{4.32}$$

Using this formula it is the possible to calculate the time taken by each node to receive data from all its children. To get the total time to send packets to the sink in the system, all the results can be added together. For aggregation, the following can be used to calculate the net time and the receiving time respectively:

$$net\_time = BF \times (trans\_time + agg\_time) + trans\_time \tag{4.33}$$

$$rec\_time = BF \times (trans\_time + agg\_time) \tag{4.34}$$

When using the conventional encryption schemes in the aggregation scenario, the following can be used to calculate the net time and the receiving time respectively:

$$opp\_time = 2(dec\_time) + agg\_time + enc\_time \tag{4.35}$$

$$net\_time = BF \times (trans\_time + opp\_time) + trans\_time \tag{4.36}$$

$$rec\_time = BF \times (trans\_time + opp\_time) \tag{4.37}$$

Using these formulae, Table 4.13 to Table **4.17** were generated. The tables show the receiving times of nodes at different distances from the sink for the implemented encryption algorithms. The time at all the distances is the same for aggregation so Table 4.17 shows the results for the different BF values in one table. The height is 7 in all the tables and the BF ranges from 2 to 5 as was the case with the Python simulations. To find the total transmission time for each algorithm, add all the times for the nodes at different distances. It is also necessary to take the setup, encryption, and decryption times into consideration. This will be discussed in more detail in the next chapter.

Table 4.13. Receiving times for the implemented encryption algorithms (BF = 2, height = 7)

distance	RC4	AES	HE (d=2)	HE (d=3)	HE (d=4)
6	14579.04	14691.16	16096.00	16912.52	17814.32
5	43737.12	44073.48	16096.00	16912.52	17814.32
4	102053.28	102838.12	16096.00	16912.52	17814.32
3	218685.60	220367.40	16096.00	16912.52	17814.32
2	451950.24	455425.96	16096.00	16912.52	17814.32
1	918479.52	925543.08	16096.00	16912.52	17814.32
0	1851538.08	1865777.32	16096.00	16912.52	17814.32

Table 4.14. Receiving times for the implemented encryption algorithms (BF = 3, height = 7)

distance	RC4	AES	HE (d=2)	HE (d=3)	HE (d=4)
6	21868.56	22036.74	24144.00	25368.78	26721.48
5	87474.24	88146.96	24144.00	25368.78	26721.48
4	284291.28	286477.62	24144.00	25368.78	26721.48
3	874742.40	881469.60	24144.00	25368.78	26721.48
2	2646095.76	2666445.54	24144.00	25368.78	26721.48
1	7960155.84	8021373.36	24144.00	25368.78	26721.48
0	23902336.08	24086156.82	24144.00	25368.78	26721.48

Table 4.15. Receiving times for the implemented encryption algorithms (BF = 4, height = 7)

distance	RC4	AES	HE (d=2)	HE (d=3)	HE (d=4)
6	29158.08	29382.32	32192.00	33825.04	35628.64
5	145790.40	146911.60	32192.00	33825.04	35628.64
4	612319.68	617028.72	32192.00	33825.04	35628.64
3	2478436.80	2497497.20	32192.00	33825.04	35628.64
2	9942905.28	10019371.12	32192.00	33825.04	35628.64
1	39800779.20	40106866.80	32192.00	33825.04	35628.64
0	159232274.88	160456849.52	32192.00	33825.04	35628.64

Table 4.16. Receiving times for the implemented encryption algorithms (BF = 5, height = 7)

distance	RC4	AES	HE (d=2)	HE (d=3)	HE (d=4)
6	36447.60	36727.90	40240.00	42281.30	44535.80
5	218685.60	220367.40	40240.00	42281.30	44535.80
4	1129875.60	1138564.90	40240.00	42281.30	44535.80
3	5685825.60	5729552.40	40240.00	42281.30	44535.80
2	28465575.60	28684489.90	40240.00	42281.30	44535.80
1	142364325.60	143459177.40	40240.00	42281.30	44535.80
0	711858075.60	717332614.90	40240.00	42281.30	44535.80

Table 4.17. Receiving times for aggregation using conventional encryption (height = 7)

BF	RC4	AES128	AES192	AES256
2	15105.04	26873.16	29193.16	31513.16
3	22657.56	40309.74	43789.74	47269.74
4	30210.08	53746.32	58386.32	63026.32
5	37762.60	67182.90	72982.90	78782.90

CHAPTER 5 DISCUSSION

5.1 OVERVIEW

This chapter discusses the results of the experiments presented in the previous chapter. The rest of this chapter is organised as follows. The second section discusses the results of the NS2 simulation and the Python simulation results are discussed in the third section. The fourth section discusses the general method for comparing the results for an arbitrary network size, while the fifth section discusses the practical experiment. The final section discusses the possible practical applications of secure information aggregation.

5.2 NS2 SIMULATION

5.2.1 Address-centric Routing

From Table 4.1 it is clear that the RC4 encryption does not increase the packet size which is one of the reasons why it is so popular in WSN application. The results of the 12 cluster network for the simulated scenario have been illustrated in Figure 5.1. It can be seen that address-centric routing in this scenario has very poor results. In this case, there is only one route for the shortest path and the boundary nodes receive and transmit vast amounts of data. As the distance from the sink node increases, the boundary nodes closer to the sink will deplete their power sources very quickly.

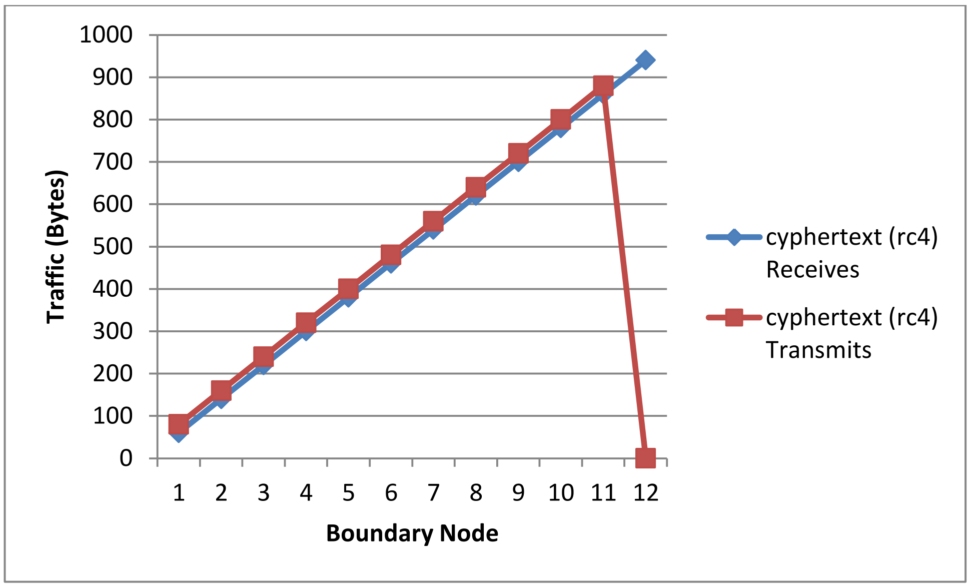

Figure 5.1. Traffic through boundary nodes for the simulated address-centric scenario

The results of the 12 cluster network for the distributed scenario have been illustrated in Figure 5.2. The figure does not include cluster 12 because the nodes in the same cluster as the sink don't receive any packets from other clusters. This is because the nodes in cluster 11 are in range with the sink and can thus send their packets directly to it. This means that the nodes in cluster 12 have the same traffic as those in cluster 1, but the sink node has the same traffic as the simulated scenario.

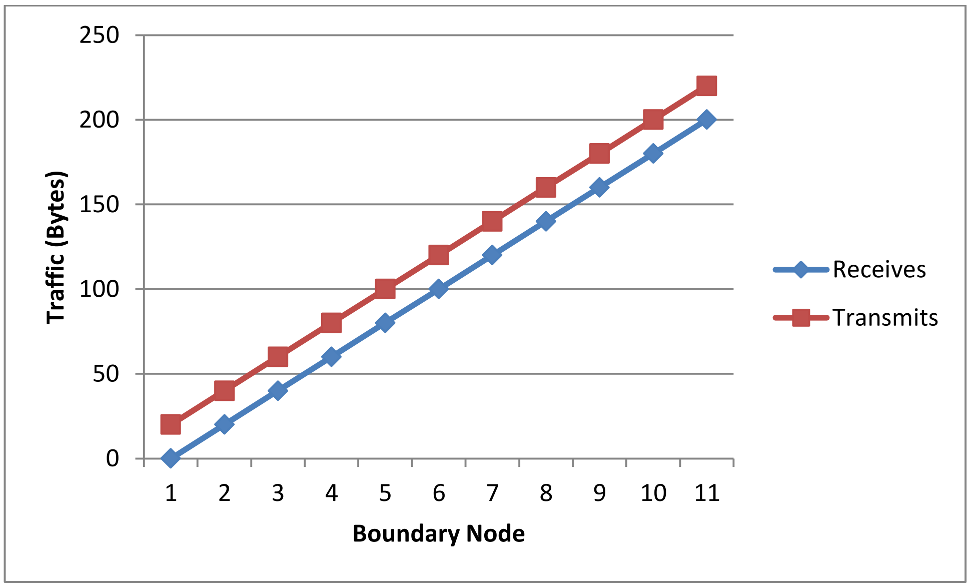

Figure 5.2. Traffic through boundary nodes for the distributed address-centric scenario

It can be seen that when the load is distributed among all the nodes in a cluster, the results are far better than the simulated results. In this case, the boundary node reduces its energy consumption by a factor of c, where c is the number of nodes in a cluster. This however coincides with an increase in energy consumption for all the other nodes in the cluster. This is because in the simulated case, non-boundary nodes don’t receive any data from the other nodes in the network. This is more acceptable though since more nodes last longer, meaning the system as a whole would function longer. In the simulated case, the boundary nodes each have a net traffic that is approximately 4 times larger the nodes in their respective clusters for the distributed case. This means that the whole system will function 4 times longer in this scenario.

It can however be seen that the nodes closer to the sink still deplete their power sources much fast than those further away. The nodes in cluster 1, for example, will last approximately 21 times longer than those in cluster 11 assuming they use the same power source. Looking at the smaller networks though, the results are much better. In the 12 node network, the furthest node from the sink is only 2 hops away and the intermediary nodes only last 3 times longer than the furthest nodes. In the 24 node network the furthest nodes are 5 hops from the sink and they last 9 times longer than those that are only 1 hop from the sink.

It is clear from the results that when using address-centric routing, the number of nodes furthest from the sink should be kept as small as possible. It is also important for the nodes to have multiple paths to the sink so as to distribute the load across as many nodes as possible. Not only does this help with network congestion, but also with the energy consumption of the nodes. This means that the routing algorithm used and the network topology both play a pivotal role in the efficiency of the system.

5.2.2 Data-centric Routing

Figure 5.3 and Figure 5.4 illustrate the results of Table 4.2. They respectively show the traffic received and transmitted in the 12 cluster network for the data-centric scenario. The sink doesn’t transmit any data so its results have been excluded from Figure 5.4. It is also important to remember that the non-boundary node don’t receive any data so they will last longer than the boundary nodes. When compared to the address-centric routing results, their traffic is the same as the simulated case but better than the distributed case. This will be discussed in more detail in the next section and this section will primarily focus on the data-centric results.

From the results, the benefits of using aggregation are immediately visible when looking at the plaintext results. Each cluster in network consumes almost the exact same amount of energy. The minor difference between boundary node 2 and 3 are due to an increase in size of the aggregate data. It increases from 3 bytes to four bytes which is a 1.25% increase of the net traffic amount so it can be considered negligible. The nodes that don't perform any aggregation tasks (i.e. the leaf nodes) will last approximately 4 times longer than the aggregating nodes. This is comparable to address-centric routing when the furthest nodes are only 2 hops away from the sink. An important thing to note is that its performance doesn't change as the distance from the sink increases. The amount of data in the network is also significantly reduced so it aids with network congestion.

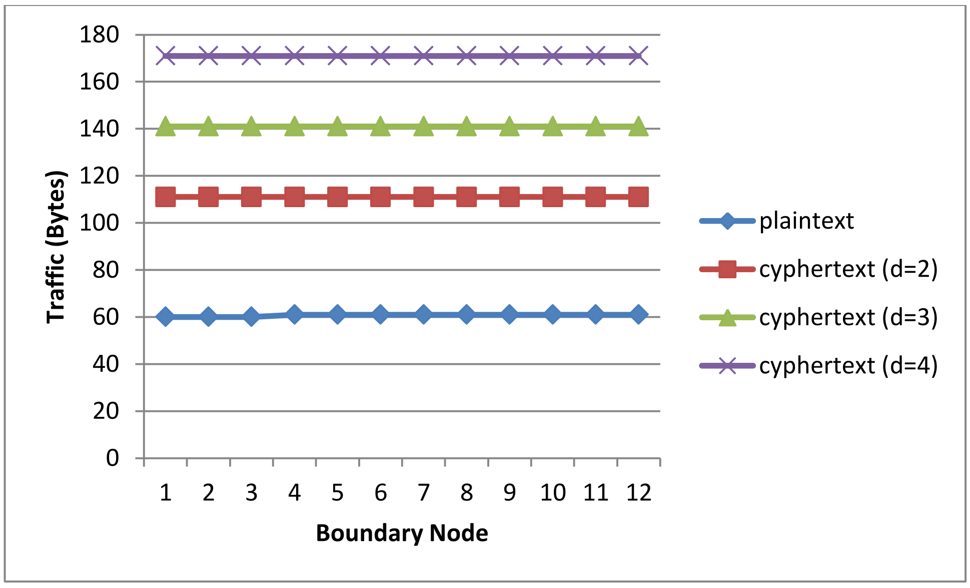

Figure 5.3. Traffic received by boundary nodes for the simulated data-centric scenario

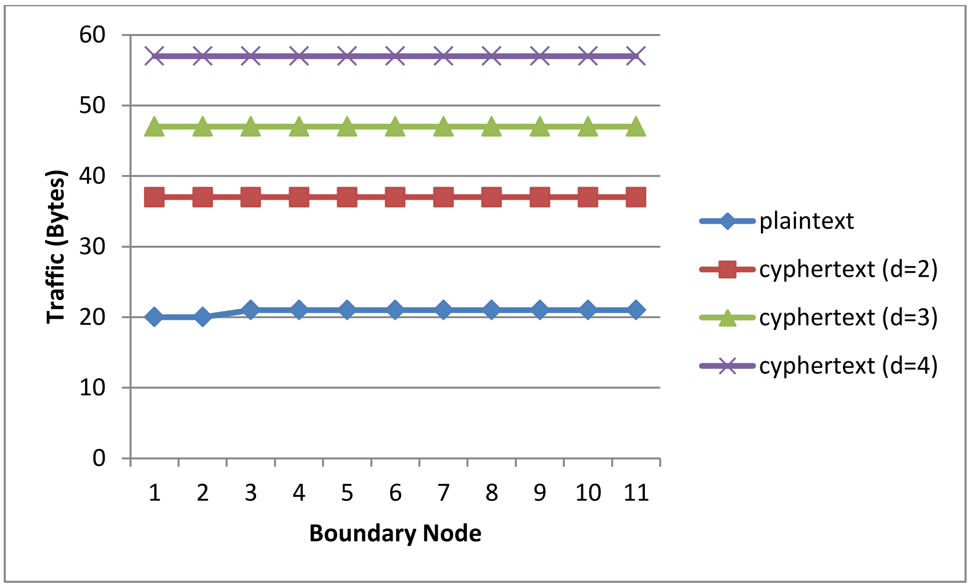

Figure 5.4. Traffic transmitted by boundary nodes for the simulated data-centric scenario

Looking at the Domingo-Ferrer results, when d = 2, the nodes have a net traffic that is approximately 1.85 times larger than if encryption was not used. The net traffic is approximately 2.35 and 2.85 larger for d=3 and d=4 respectively. So when using this scheme under the specified security parameters, the energy consumption of the nodes will, in the worst case, be reduced by a factor of 3 and the network traffic will be increased by the same amount. From the results it is clear that while this scheme does affect system performance, it will be acceptable for most applications when security is a concern.

5.2.3 Combined Results

From Table 4.3, it is clear that the performance of the simulated address-centric protocol is far worse than all the aggregation scenarios. It is for this reason that only the distributed results will be considered. This discussion will begin by discussing some of the results as they are from the table (to illustrate a few concepts) and later a more accurate comparison will be made. Looking at both Table 4.3 and Figure 4.1, the 12 node network outperforms even the plaintext aggregation. In this scenario the furthest node from the sink is only two hops away. In this case, the number of children per boundary node (for aggregation) is larger the distance (in hops) of the furthest nodes from the sink. From this it is clear that the distributed scenario is adversely affected by the distance of the furthest nodes from the sink while aggregation is affected by the number of children per boundary node. Increasing the number of nodes in each cluster would not affect its performance but it would reduce the performance of aggregation. This is shown in Table 5.1 and illustrated by Figure 5.5 for the 12 cluster network. In the table, the first column represents the number nodes in each cluster and the second column is the amount of data received by the sink in address centric routing. The last 4 columns show the data received by the boundary nodes for the plaintext aggregation and 3 homomorphic encryption scenarios.

Since the results of distributed scenario don’t change, it outperforms most of the plaintext aggregation and all of the homomorphic encryption scenarios. The only performance benefit aggregation has in this case is network congestion. The sink will still receive $(\mathrm{n}-1)\times 20$ bytes of data in address-centric routing, where n is the number of nodes in the network (including the sink). In aggregation however, the sink will only receive approximately $\mathrm{k}\times\mathrm{p}$ bytes of data, where p is the packet size and k is the number of children it has. Network congestion can also become a problem in aggregation, but it can be alleviated by limiting the number of children each aggregator node can have [35]. This also has the added benefit of reducing the energy consumption of each boundary node.

Table 5.1. Traffic received by boundary nodes as the number of cluster nodes increases

	Address-centric	Aggregation			
C-nodes	Sink	Plaintext	HE (d = 2)	HE (d = 3)	HE (d = 4)
5	1180	81	148	188	228
6	1420	101	185	235	285
7	1660	121	222	282	342
8	1900	141	259	329	399
9	2140	161	296	376	456
10	2380	181	333	423	513
11	2620	201	370	470	570
12	2860	221	407	517	627
13	3100	241	444	564	684
14	3340	261	481	611	741

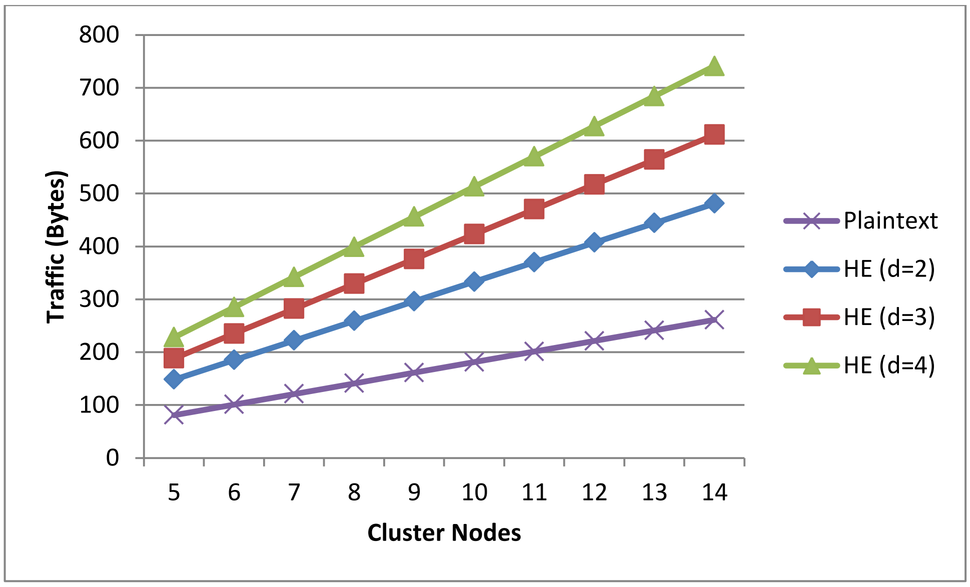

Figure 5.5. Traffic received by boundary nodes as the number of cluster nodes increases

Referring back to Table 4.3, for the 24 node network, the results of the distributed and plaintext aggregation scenarios are still comparable. Even though the nodes in cluster 5 have a net traffic that is 2.25 times larger than the aggregation nodes, the rest of the clusters have similar or better results than the aggregating nodes. In this case the furthest nodes are only five hops away from the sink. There will be benefits to using either aggregation or address-centric routing depending on the network constraints and the number of nodes in the network. So in this case it is very application specific. As the furthest nodes get further away from the sink however, the benefits of using aggregation start outweighing those of address-centric routing.

At this point it should have become apparent that comparing the results is not as straight-forward as it first appeared to be. To illustrate this point further consider Table 5.2 and Figure 5.6. The table shows the distributed results of Table 4.3 as a fraction of the other results and the figure illustrates this. When referring to this table, the plaintext aggregation while be represented by the letter P and the homomorphic encryption results will be represented by D1, D2 and D3 respectively. If the better system is the one which operates for longer (which reduces the maintenance burden), then clusters 3, 5, 6 and 7 for P, D1, D2 and D3 respectively are better than the distributed results. But these systems would only function between 14% and 25% longer than their distributed counterparts which might not be significant depending on the application. Using that as a measure requires the designers to determine what constitutes a significant enough operational time-difference.

In applications that aren't necessarily time critical, the cost will be the primary factor used to determine which system is better most of the time. This explanation will primarily be concerned with the required battery replacements for each scenario. Table 5.3 and Table 5.4 will be used to explain this concept. The former shows the amount of times each node in a particular cluster would have to be replaced in a year while the latter shows the total node replacements for all the clusters up to the current one. In both tables, it is assumed that the boundary nodes of the aggregation scenarios would have to be replaced once every month.

The 12 node network will now be used to clarify the table values. In cluster 1, the boundary nodes will have to be replaced 12 times and the rest only have to be replaced 3 times. The nodes of the distributed scenario have a net traffic that is 25% of the boundary nodes in plaintext aggregation so they will have to be replaced 3 times in a year. For the 3 homomorphic encryption scenarios, the values are 2, 1 and 1 respectively. In cluster 2, the in range node will have to be replaced 6 times in a year, while the remaining nodes are the same as cluster 1. The nodes of the distributed scenario have a net traffic that is 75% of the boundary nodes in plaintext aggregation so they will have to be replaced 9 times in a year. In the sinks cluster, there is no boundary node, so there is just the in range node and the remaining nodes which have to be replaced 6 and 3 times respectively. The nodes of the distributed scenario are the same as those in cluster 1 because they don’t receive any packets so they also have to be replaced 3 times when referring to the plaintext aggregation. So what’s important is that the values are with reference to the net traffic of the boundary nodes of the different aggregation scenarios. For the plaintext aggregation, 80 bytes constitutes a once a month replacement. For homomorphic encryption, these values are 148, 188 and 228 bytes respectively.

For the total replacements, the number of nodes in each cluster is used. For aggregation the 12 node network has 3 nodes that have to be replaced 3 times, and 1 that has to be replaced 12 times. Cluster 2 has 2 nodes that have to be replaced 3 times, 1 that has to be replaced 6 times, and 1 that has to be replaced 12 times. Cluster 3 has 2 nodes that have to be replaced 3 times and 1 that has to be replaced 6 times. It is assumed that the sink has an unlimited power supply so it doesn’t have to be replaced. The total number of battery replacements for this network is then 57. With reference to the plaintext aggregation, the distributed scenario has 4 nodes that have to be replaced 3 times in cluster 1, 4 nodes that have to be replaced 9 times in cluster 2 and 3 nodes that have to be replaced 3 times in cluster 3. The total number of battery replacements for this network is then 57.

These results can be seen in Table 5.4 and from this it can be seen that in the 12 node network, the distributed and plaintext aggregation scenarios will have the same number of battery replacements in a year. In the 16 node network however, P will have fewer node replacements. It is important to note that the nodes in cluster 4 (the sink’s cluster) have to be considered for more accurate results. This results in a total of 117 and 81 battery replacements respectively meaning that P will have 36 fewer replacements per year. When looking at D1, the 20 node network has comparable results the distributed scenario. In this case, D1 will only have 6 fewer battery replacements. In the 24 node network however, D1 will have 41 few node replacements which is more significant. D2 and D3 have better performances in the 28 and 32 node networks respectively. In these networks, they each have 30 fewer battery replacements than their corresponding distributed scenarios. This shows that as the furthest nodes get further away from the sink, aggregation becomes the superior choice.

In summary, the results of the simulated address-centric protocol were far worse than those of any of the aggregation schemes. The best case results were more comparable to the aggregation schemes, outperforming them for smaller networks. The two address-centric scenarios are however two extremes, one is extremely efficient and the other one's performance is quite awful. In reality though, one would normally get a network that is in between the two extremes. So the network topology and routing algorithms play a vital role in network performance. The results of this experiment are however able to show a general trend in the performances of the schemes. For smaller networks, where the furthest node is not too far from sink, using RC4 is generally better than Domingo-Ferrer. As the network size grows and with it, the distance between the sink and the furthest nodes, Domingo-Ferrer starts outperforming RC4. Where the performances are comparable, factors such as network topology, size and congestion should be taken into considering when choosing between the schemes. It is also important to note that the Domingo-Ferrer nodes will, in the worst case, increase energy consumption by a factor of only 3 when compared to plaintext aggregation. While this difference is not negligible, it will be acceptable for most applications where security is a concern.

Table 5.2. Distributed results as a fraction of the aggregation results

	Address-centric	Aggregation			
B-node	Distributed	Plaintext	HE (d = 2)	HE (d = 3)	HE (d = 4)
1	1	0.25	0.14	0.11	0.09
2	1	0.75	0.41	0.32	0.26
3	1	1.25	0.68	0.53	0.44
4	1	1.75	0.95	0.74	0.61
5	1	2.25	1.22	0.96	0.79
6	1	2.75	1.49	1.17	0.96
7	1	3.25	1.76	1.38	1.14
8	1	3.75	2.03	1.60	1.32
9	1	4.25	2.30	1.81	1.49
10	1	4.75	2.57	2.02	1.67
11	1	5.25	2.84	2.23	1.84

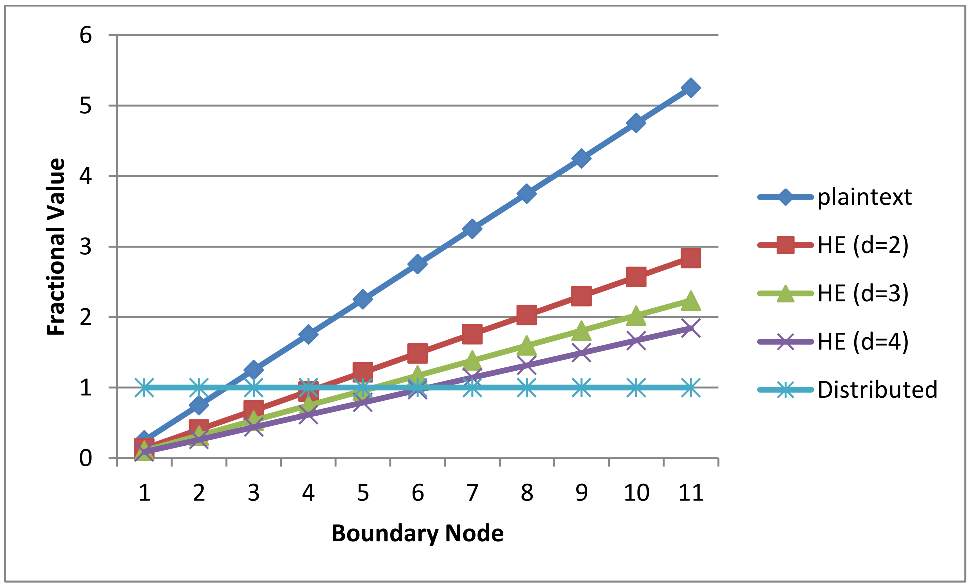

Figure 5.6. Distributed results as a fraction of the aggregation results

Table 5.3. Cluster Battery replacements in the distributed vs aggregation scenarios

	Reference	Distributed vs Aggregation			
B-node	Agg	Plaintext	HE (d = 2)	HE (d = 3)	HE (d = 4)
12 nodes					
1	3,12	3	2	1	1
2	3,6,12	9	5	4	3
sink	3,6	3	2	1	1
24 nodes					
3	3,6,12	15	8	6	5
4	3,6,12	21	11	9	7
5	3,6,12	27	15	11	9
sink	3,6	3	2	1	1
36 nodes					
6	3,6,12	33	18	14	12
7	3,6,12	39	21	17	14
8	3,6,12	45	24	19	16
sink	3,6	3	2	1	1
48 nodes					
9	3,6,12	51	28	22	18
10	3,6,12	57	31	24	20
11	3,6,12	63	34	27	22
sink	3,6	3	2	1	1

Table 5.4. Total battery replacements in the distributed vs aggregation scenarios

	Reference	Distributed vs Aggregation			
B-node	Agg	Plaintext	HE (d = 2)	HE (d = 3)	HE (d = 4)
12 nodes					
1	21	12	8	4	4
2	45	48	28	20	16
sink	57	57	34	23	19
24 nodes					
3	69	108	60	44	36
4	93	192	104	80	64
5	117	300	164	124	100
sink	129	309	170	127	103
36 nodes					
6	141	432	236	180	148
7	165	588	320	248	204
8	189	768	416	324	268
sink	201	777	422	327	271
48 nodes					
9	213	972	528	412	340
10	237	1200	652	508	420
11	261	1452	788	616	508
sink	273	1461	794	619	511

5.3 PYTHON SIMULATION

The results of the previous simulation were able to show the general network behaviour as the distance from the sink increases. These results are a more generalised version of the previous results. The results of Table 4.5 to Table 4.8 are illustrated by Figure 5.7 to Figure

5.10. The figures show the results from the furthest nodes up to the node where the SPF results are far worse than aggregation. This is because the aggregation results would otherwise not be distinguishable in the figures. These results confirm those of the previous section where as the distance from the sink increases, aggregation starts outperforming SPF.

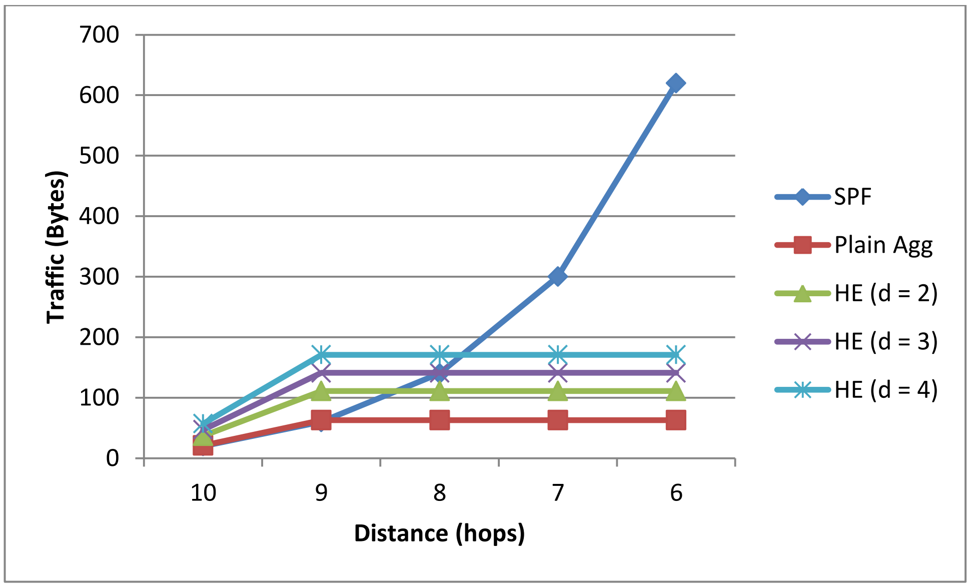

Figure 5.7. Net traffic through nodes for the 2047 node network (BF = 2, height = 10)

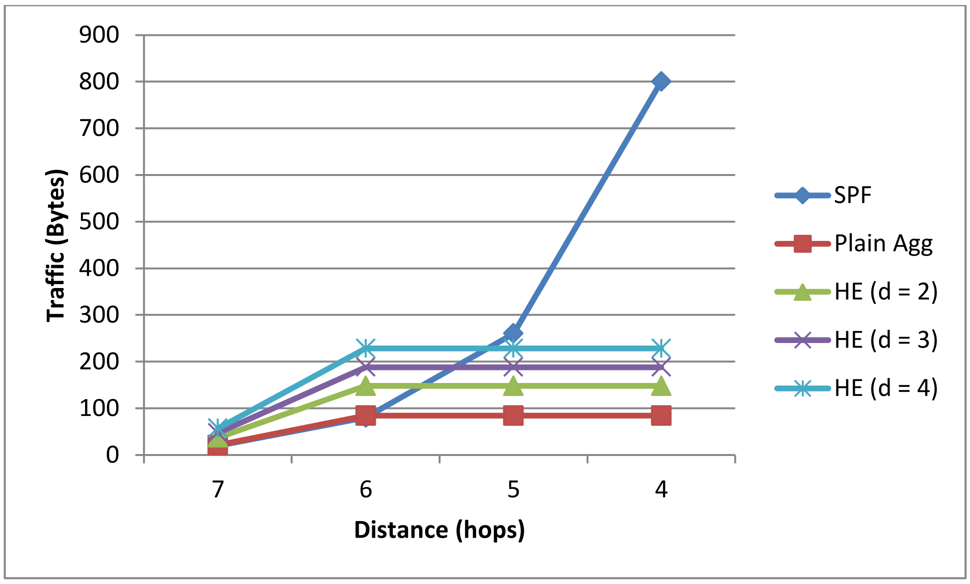

Figure 5.8. Net traffic through nodes for the 3280 node network (BF = 3, height = 7)

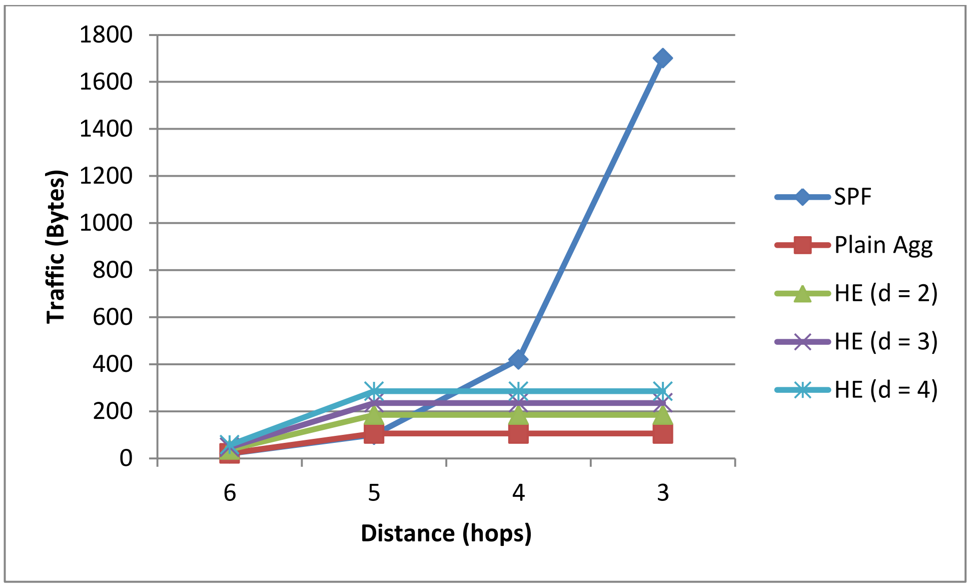

Figure 5.9. Net traffic through nodes for the 5461 node network (BF = 4, height = 6)

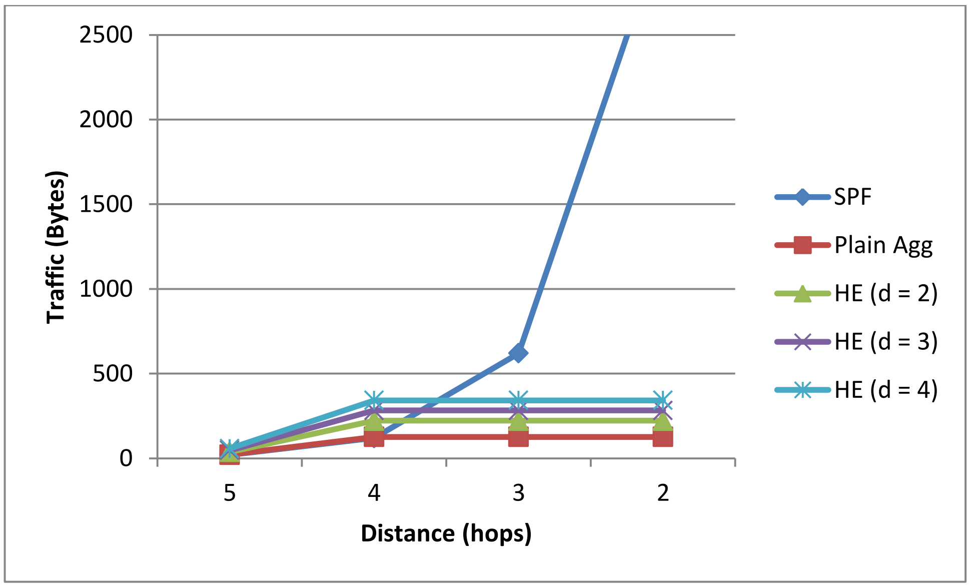

Figure 5.10. Net traffic through nodes for the 3906 node network (BF = 5, height = 5)

The previous section has already covered the comparison between aggregation and address-centric routing extensively. These results are different but nothing is to be gained from doing a similar analysis in this section because the conclusion will be the same. What is different with these results is that the SPF results aren’t distributed like the NS2 results. This means that as the number of children per node also adversely affects the performance of this scenario. To analyse this further, consider a balanced tree with a height of 5. Figure

5.11 shows the traffic through the nodes at different distances from the sink for the different BFs (excluding the sink). It can be seen that as the branching factor increases, there is an exponential increase in the traffic through the nodes as they get closer to the sink.

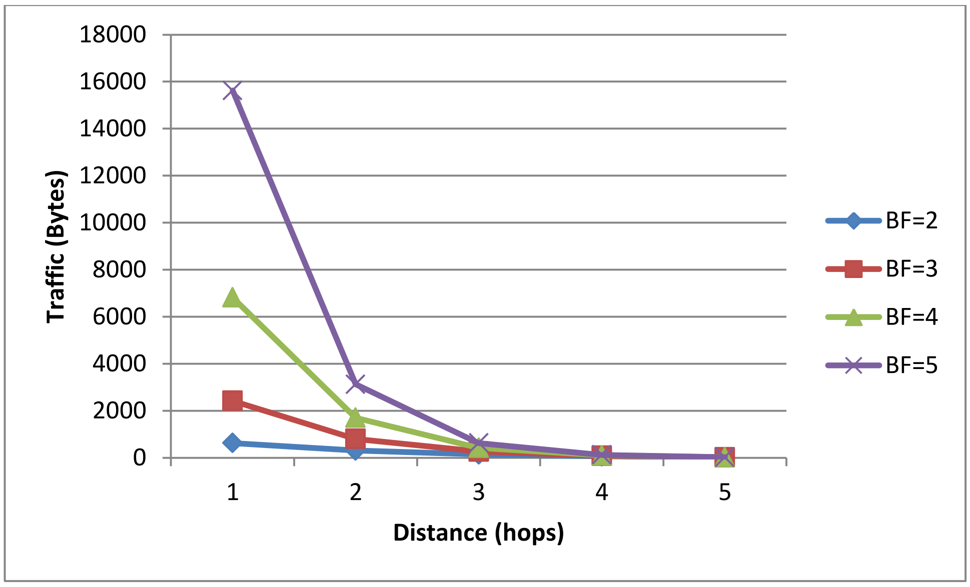

Figure 5.11. Net traffic through nodes for a balanced tree with height = 5

Figure 5.11 only illustrates the results for the SPF algorithm but aggregation results are also affected by an increasing BF. A more accurate measure of the effect thus has to consider the SPF results in relation to the aggregation results. Table 5.5 shows the SPF results as a fraction of the other results for the balanced tree with a height of 5. This table has the same format as Table 5.2. It can be seen from the table that the nodes at distance 4 and 5 aren't affected by increasing the BF while the nodes at distance 3 have comparable results. The nodes that are 2 and 1 hops from the sink have a more significant difference when comparing their results. The results of these nodes are illustrated by Figure 5.12 and Figure 5.13. The fractional difference where the BF is 5 is 5.2 times greater than when BF is 2 for a distance of 2. The values for BF = 4 and BF = 3 are 3.4 and 2 respectively. For a distance of 1 these values are 12.96, 6.6 and 2.93 respectively. This shows that when the furthest nodes aren't too far from the sink, increasing the BF doesn't affect the results by much. But as they get further, it has a significant difference on the results. It is also evident that as the percentage by which the packet size increases gets larger, the fractional difference gets smaller.

Table 5.5. SPF results as a fraction of the aggregation results

	Reference	Distributed vs Aggregation			
Distance	SPF	Plaintext	HE (d = 2)	HE (d = 3)	HE (d = 4)
BF = 2					
1	1	9.84	5.59	4.4	3.63
2	1	4.76	2.7	2.13	1.75
3	1	2.22	1.26	0.99	0.82
4	1	0.95	0.54	0.43	0.35
5	1	0.95	0.54	0.43	0.35
BF = 3					
1	1	28.81	16.35	12.87	10.61
2	1	9.52	5.41	4.26	3.51
3	1	3.1	1.76	1.38	1.14
4	1	0.95	0.54	0.43	0.35
5	1	0.95	0.54	0.43	0.35
BF = 4					
1	1	64.95	36.86	29.02	23.93
2	1	16.19	9.19	7.23	5.96
3	1	4	2.27	1.79	1.47
4	1	0.95	0.54	0.43	0.35
5	1	0.95	0.54	0.43	0.35
BF = 5					
1	1	123.97	70.36	55.39	45.67
2	1	24.76	14.05	11.06	9.12
3	1	4.92	2.79	2.2	1.81
4	1	0.95	0.54	0.43	0.35
5	1	0.95	0.54	0.43	0.35

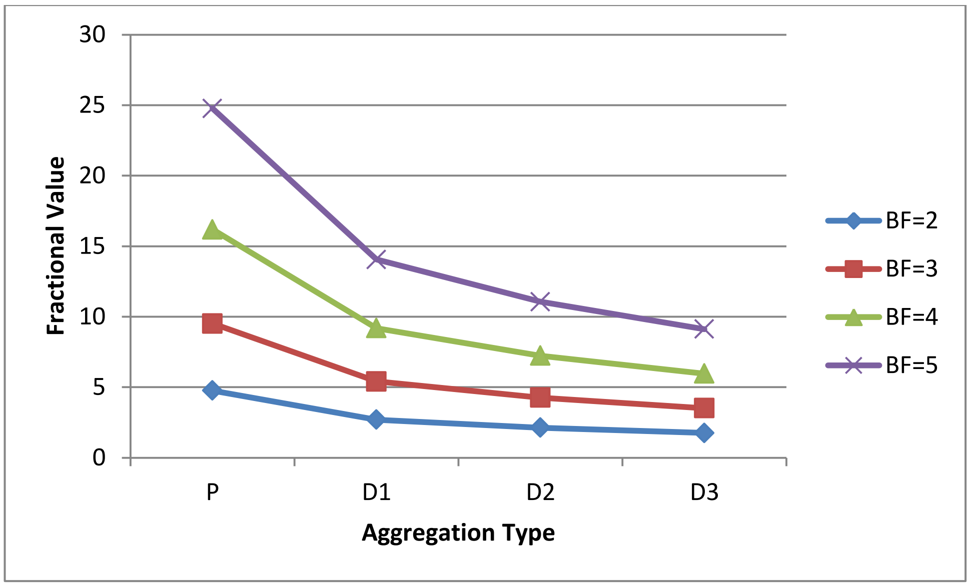

Figure 5.12. SPF results as a fraction of the aggregation results (distance=2)

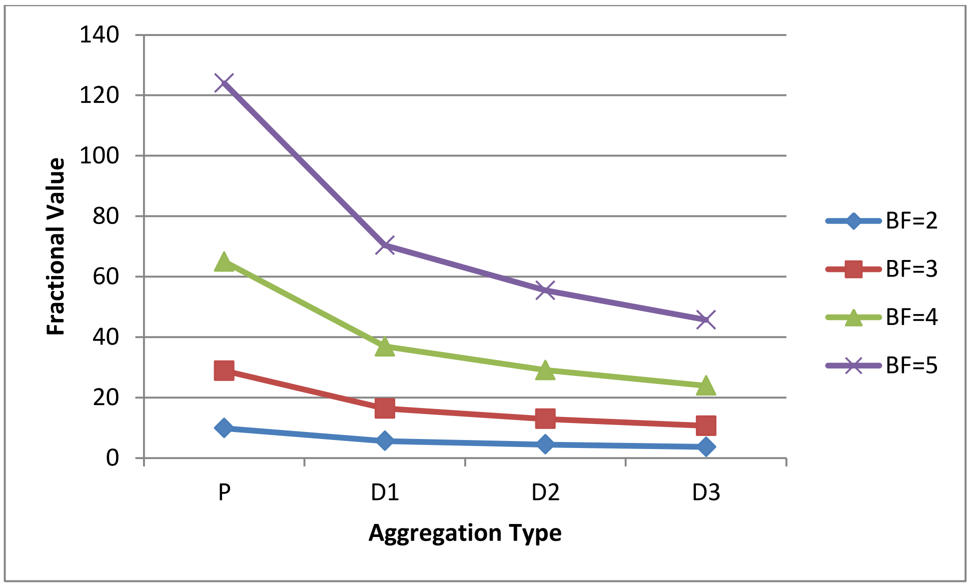

Figure 5.13. SPF results as a fraction of the aggregation results (distance=1)

What the previous discussion means for practical networks is that the height of the tree should be kept as small as possible when using SPF. This again points to the fact that aggregation is the superior choice as the network size grows. What the results of this section also show is that it is important to have an adaptive SPF algorithm. Where possible the load should be distributed as much as possible. It was seen in the previous section that distributing the load among as many nodes as possible can drastically improve the address-centric results. So again, selecting an appropriate routing algorithm for a specific topology should not be taken lightly.

5.4 GENERAL FORMULAE

The equations derived in the previous chapter make it easier to obtain the python results without having to simulate the actual networks. It also gives a lot of flexibility in terms of the actual results that one is trying to obtain. It was already discussed that the traffic through the nodes only one hop from the sink is a major problem. So with this method it is possible to see how this compares to aggregation as the height of the tree increases with very little effort. These results (ratio of SPF to aggregation) have been plotted in Figure 5.14 to Figure 5.17 as the height increase from 3 to 7 for the 4 simulated BF values. As before, it is evident that as both the height and BF increase, the SPF results deteriorate in comparison to the aggregation results. It is also possible to keep the height constant to see what happens as the BF increase as shown in Figure 5.18. The figure shows the results as the BF increases from 2 to 7. This verifies that there is an exponential increase as the BF is increased.

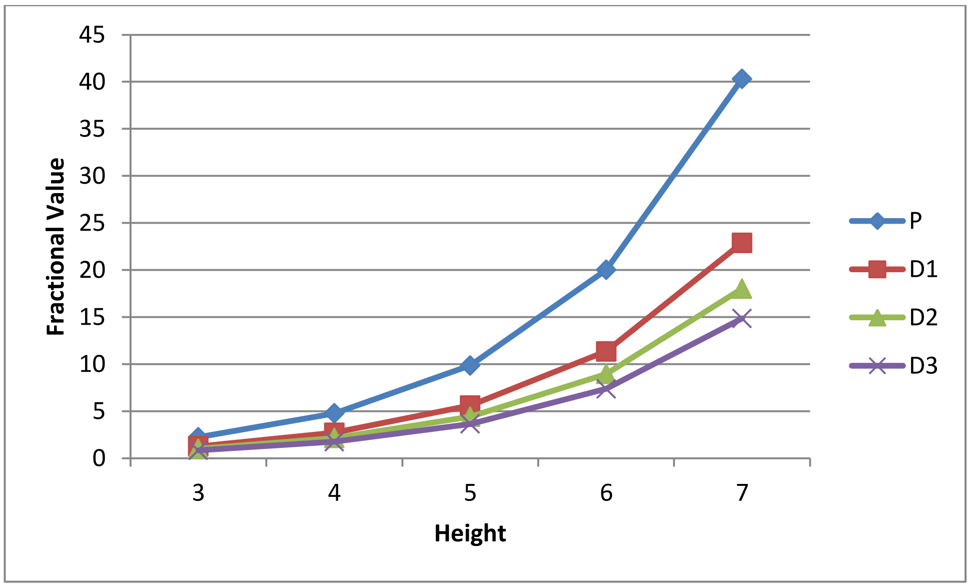

Figure 5.14. SPF results as a fraction of the aggregation results (BF = 2, distance=1)

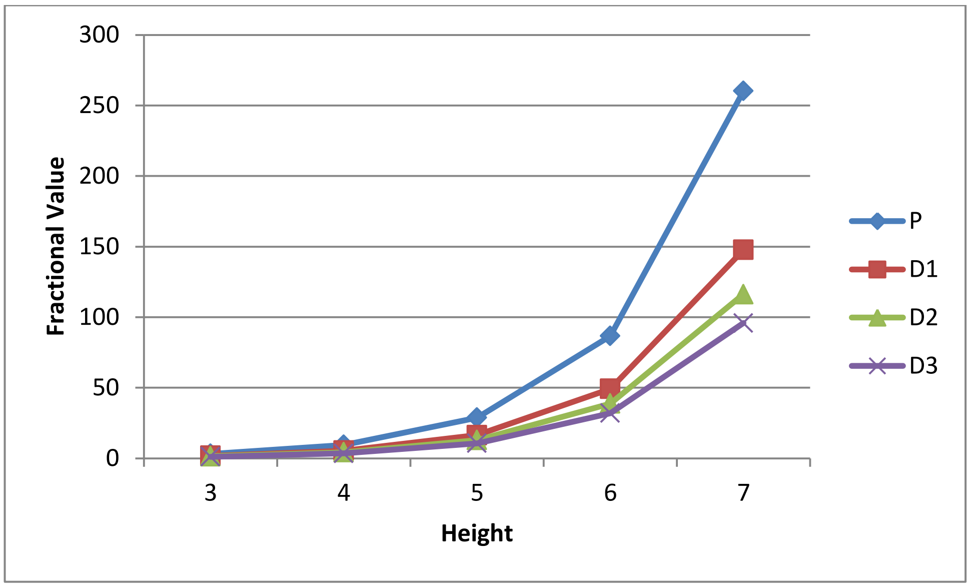

Figure 5.15. SPF results as a fraction of the aggregation results (BF = 3, distance=1)

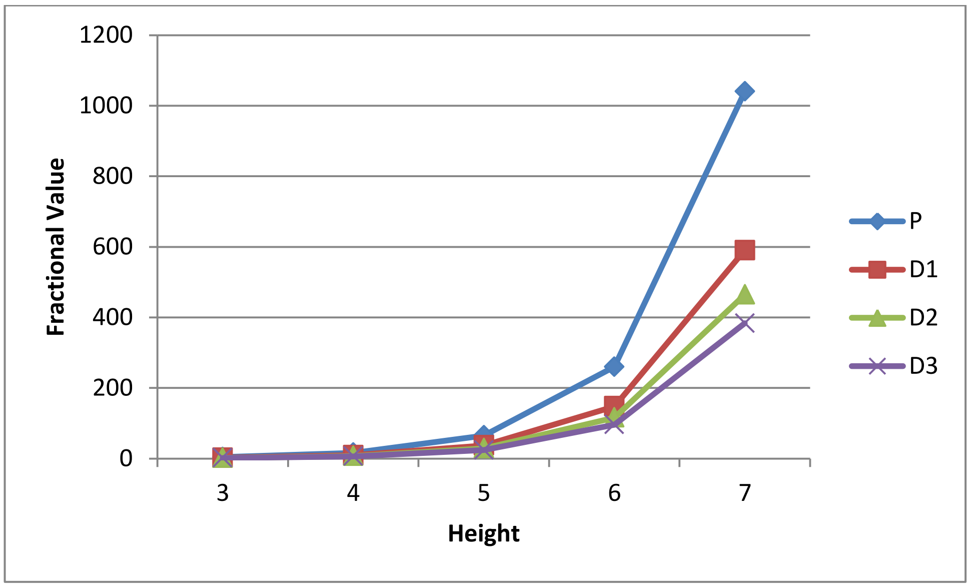

Figure 5.16. SPF results as a fraction of the aggregation results (BF = 4, distance=1)

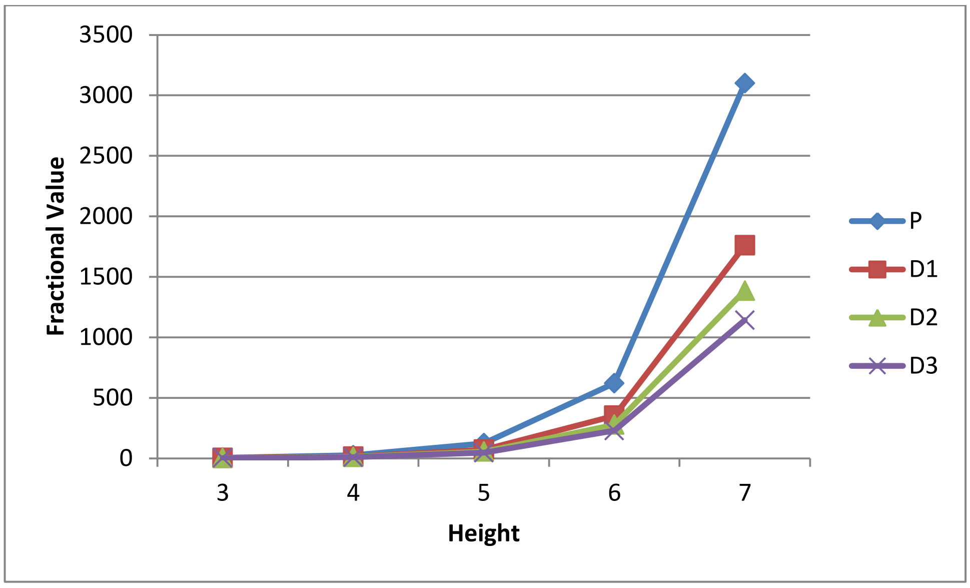

Figure 5.17. SPF results as a fraction of the aggregation results (BF = 5, distance=1)

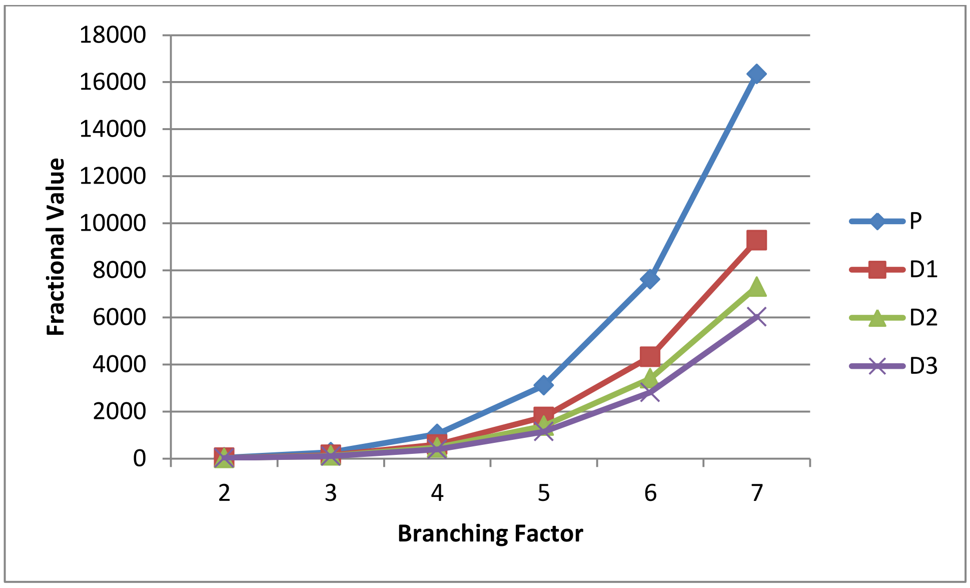

Figure 5.18. SPF results as a fraction of the aggregation results (height = 7, distance=1)

It is also possible to obtain the results for when the SPF packet size increase. In this case the homomorphic encryption results won't change so the results will still be the same except that these scenarios will start outperforming SPF for much smaller networks. These equations are also independent of the homomorphic encryption algorithm, so if the increase in packet size is known, any algorithm can be simulated. The results of trees that aren't balanced can also be obtained by finding the results of different sub-trees and then adding the results where appropriate. In this case, it is possible for the nodes that are the same distance from the sink not to have the same traffic going through them so this should be taken into consideration when doing the comparisons. So in summary, these equations provide a flexible platform and algorithm independent method of simulating varying network scenarios.

5.5 PRACTICAL EXPERIMENT

5.5.1 Operational Times

From Table 4.10 it is evident that RC4 is much faster than all the other algorithms when considering the encryption and decryption times. The setup time is however the slowest of all of them. This is generally acceptable since this happens only once every time the motes are rebooted. This time also becomes relevant in the key synchronisation process so this could adversely affect the results depending on how often this has to happen. It is also worth noting that unlike AES, this was not an optimised implementation and this could be a factor into why the setup time is so slow.

Table 5.6 shows the fractional comparisons of the encryption times for the implemented algorithms. The decryption time comparisons are similar so this discussion considers both. While the performance of RC4 is far superior, the rest of the algorithms are more comparable. There isn’t much difference between the AES algorithms with the 3 different key sizes. The 3 variations of the homomorphic encryption scheme show a more significant difference, but these values also aren’t too high. When comparing AES to Domingo-Ferrer, it can be seen that AES is faster. With a key size of 256 bits, the AES encryption is almost the same as Domingo-Ferrer when d = 2. The other key sizes aren’t much faster. When d=4, AES is between 2.4 and 3.3 times faster that Domingo-Ferrer. This difference is significant but is acceptable when considering the benefits of aggregation. When considering the results of Table 4.11, Domingo-Ferrer is much slower than all the algorithms. As was the case with the RC4 setup, this was not an optimised implementation and that could be a reason why these results are so poor.

Table 5.6. Fractional comparison of encryption times for the implemented encryption algorithms

Algorithm	RC4	AES128	AES192	AES256	HE (d=2)	HE (d=3)	HE (d=4)
RC4	1.000	0.042	0.036	0.031	0.025	0.017	0.013
AES128	23.582	1.000	0.842	0.727	0.598	0.400	0.299
AES192	28.013	1.188	1.000	0.863	0.711	0.475	0.355
AES256	32.443	1.376	1.158	1.000	0.823	0.550	0.412
HE (d=2)	39.418	1.671	1.407	1.215	1.000	0.668	0.500
HE (d=3)	59.000	2.502	2.106	1.819	1.497	1.000	0.749
HE (d=4)	78.810	3.342	2.813	2.429	1.999	1.336	1.000

5.5.2 Transmission Times

The transmission times of the algorithms varied a lot so the statistical results were obtained and tabulated in Table 4.12. This difference could be due to factors such as interference. The results show an increase in the average transmission time as the packet size grows. The differences aren't as large as one would expect, but this could be due to the fact that the experiment wasn't conducted in an ideal environment. But the results do however show the general trend that is expected as the packet size grows.

5.5.3 Simulations

Table 4.13 to Table 4.16 show the results for nodes at different distances from the sink. To get the total time for each algorithm, it is necessary to add the receiving times for each node. These results are shown in Table 5.7. It is then necessary to add the setup and encryption times to these values to get Table 5.8, which is illustrated by Figure 5.19. The encryption times used in the table for Domingo-Ferrer include the S-value generation. The sink still has to decrypt all the packets it receives, which is BF for Domingo-Ferrer and num_nodes-1 for SPF. But the results thus far already show the general pattern so it isn't necessary to tabulate these results.

Table 5.7. Total receiving times for the implemented encryption algorithms (height = 7)

BF	RC4	AES	HE (d=2)	HE (d=3)	HE (d=4)
2	3601022.88	3628716.52	112672.00	118387.64	124700.24
3	35776964.16	36052106.64	169008.00	177581.46	187050.36
4	212241664.32	213873907.28	225344.00	236775.28	249400.48
5	889758811.20	896601494.80	281680.00	295969.10	311750.60

When comparing the SPF results to each other, it can be seen that there isn’t much difference between the times of the respective algorithm, particularly as the network size grows. This is due to the fact that the transmission times were very similar and it is assumed that the encryption can happen concurrently across all nodes. Depending on how time critical the application is, the milliseconds differences could become significant. When looking at the Domingo-Ferrer results, there is a more significant difference between the different variations. In this case, a lot of time can be saved by using the smaller d values. It can also be seen that the results aren’t affected much by an increase in the network size. When comparing Domingo-Ferrer to SPF, the results conform to previous conclusion that SPF is better for smaller networks while aggregation becomes the superior choice as the network size grows.

Table 5.8. Total receiving times including the operational times (height = 7)

Algorithm	BF=2	BF=3	BF=4	BF=5
RC4	3618258.88	35794200.16	212258900.32	889776047.20
AES128	3633793.52	36057183.64	213878984.28	896606571.80
AES192	3634394.52	36057784.64	213879585.28	896607172.80
AES256	3634917.52	36058307.64	213880108.28	896607695.80
HE (d=2)	38500211.20	38556547.20	38612883.20	38669219.20
HE (d=3)	62123213.30	62182407.12	62241600.94	62300794.76
HE (d=4)	95315333.18	95377683.30	95440033.42	95502383.54

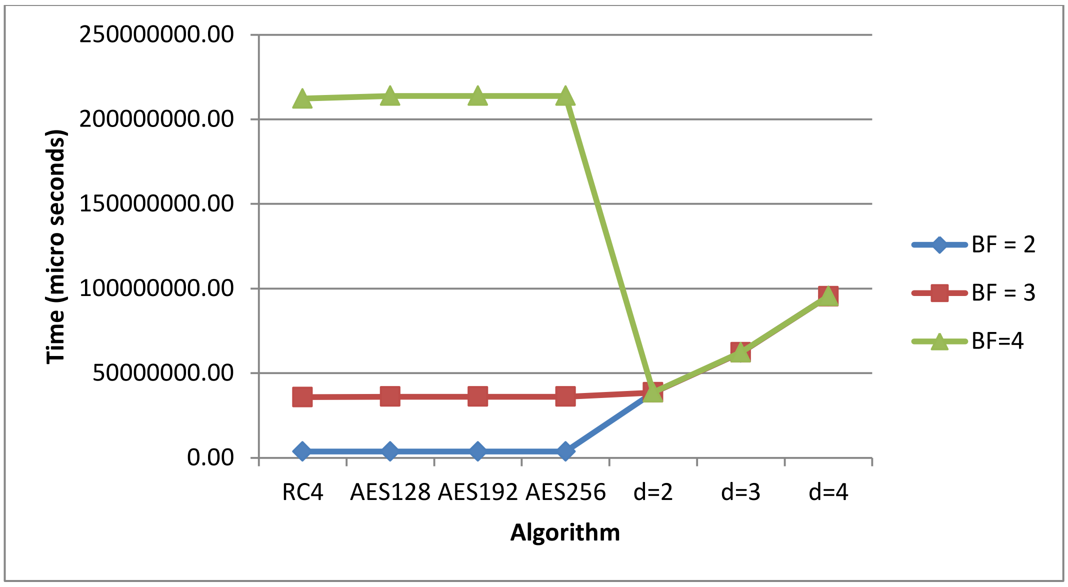

Figure 5.19. Total receiving times including the operational times (height = 7)

Table 4.17 shows the results for aggregation implemented using conventional encryption. To get the total time for each algorithm, it is necessary to multiply the value in the table by the height of the tree. These results are shown in Table 5.9. It is then necessary to add the setup and encryption times to these values to get Table 5.10, which is illustrated by Figure 5.20. There is a significant difference between the RC4 and AES algorithms. In this scenario, RC4 outperforms all of the AES algorithms convincingly even for the smaller network size. The results of the AES algorithms are similar for smaller network sizes, but as the branching factor increases, the smaller key sizes start outperforming the larger ones more considerably. It is however worth noting that the margins aren't too big so depending on the specific application, the differences could be still considered negligible.

When comparing this scenario to Domingo-Ferrer, the conventional encryption schemes all have superior results. This is due to the large S generation time of Domingo-Ferrer during the encryption process. When looking at only the total receiving times of each algorithm (i.e. Table 5.7 and Table 5.9), Domingo-Ferrer has comparable results to RC4 and outperforms the AES algorithms. This is illustrated by Figure 5.21. This means that Domingo-Ferrer is as fast as RC4 once the initial encryption has been done. This is due to the fact that this algorithm does not need to decrypt the values at each intermediate node.

When considering security, the conventional encryption schemes make the system more vulnerable as explained previously. So even though they are faster overall, the homomorphic encryption scheme is still the superior choice in the aggregation scenario.

Table 5.9. Total receiving times for aggregation using conventional encryption (height = 7)

BF	RC4	AES128	AES192	AES256
2	105735.28	188112.12	204352.12	220592.12
3	158602.92	282168.18	306528.18	330888.18
4	211470.56	376224.24	408704.24	441184.24
5	264338.20	470280.30	510880.30	551480.30

Table 5.10. Total receiving times for aggregation including the operational times (height = 7)

BF	RC4	AES128	AES192	AES256
2	122971.28	193189.12	210030.12	226793.12
3	175838.92	287245.18	312206.18	337089.18
4	228706.56	381301.24	414382.24	447385.24
5	281574.20	475357.30	516558.30	557681.30

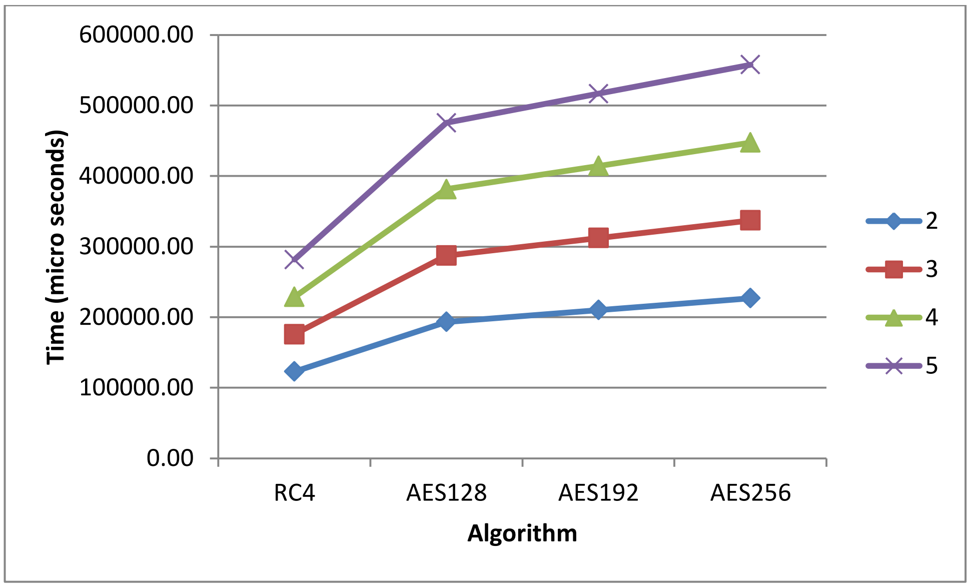

Figure 5.20. Total receiving times for aggregation including the operational times (height = 7)

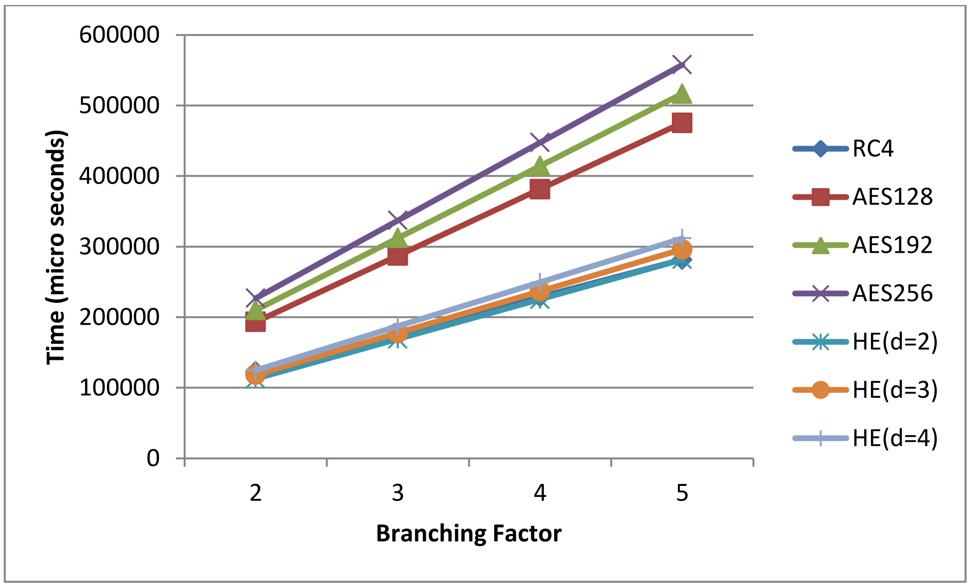

Figure 5.21. Comparison between all the aggregation scenarios for different BF values

CHAPTER 6 CONCLUSION

6.1 RESULTS

Wireless sensor networks have become increasingly popular in many applications such as environment monitoring and law enforcement. Data aggregation is a method used to reduce network traffic but cannot be used together with conventional encryption schemes because it is not secure and introduces extra overhead. Homomorphic encryption is an encryption scheme that allows data processing on encrypted data as opposed to plaintext. It has the benefit that each intermediate node does not have to decrypt each packet, but the resulting cyphertext is usually much larger than the original plaintext. This could negatively affect system performance because the energy consumption of each node is directly proportional to the amount of data it transmits.

To investigate this increase in packet size, experiments were conducted on two simulation platforms (NS2 and the Python scripting language). The NS2 simulation focused on a specific implementation while the Python simulation depicted network behavior for general networks. Both simulations considered four scenarios (communication with and without encryption and aggregation). There was also a practical experiment conducted on Crossbow TelosB motes. This experiment was setup to investigate the speed and transmission times of the different algorithms.

For the NS2 simulation it was found that for smaller networks, where the furthest node is not too far from sink, using RC4 is generally better than Domingo-Ferrer. As the network size grows and with it, the distance between the sink and the furthest nodes, Domingo-Ferrer starts outperforming RC4. Where the performances are comparable, factors such as network topology, size and congestion should be taken into considering when choosing between the schemes. . It was also the Domingo-Ferrer scheme will, in the worst case, increase energy consumption by a factor of only 3 when compared to plaintext aggregation. While this difference is not negligible, it will be acceptable for most applications where security is a concern.

The python simulations provided a convenient way to compare the different scenarios more generally. It was found that the distance of the furthest nodes from the sink should be kept as small as possible when using SPF. This again points to the fact that aggregation is the superior choice as the network size grows. What the results of these simulations also showed is that it is important to have an adaptive SPF algorithm. Where possible the load should be distributed as much as possible. Distributing the load among as many nodes as possible can drastically improve the address-centric results. So again, selecting an appropriate routing algorithm for a specific topology should not be taken lightly.

For the practical experiment, when considering the encryption/decryption times, it was found that the performance of RC4 is far superior to the other algorithms and Domingo-Ferrer was by far the slowest because of the time consuming S generation process. There isn't much difference between the AES algorithms with the 3 different key sizes. The 3 variations of the homomorphic encryption scheme showed a more significant difference, but these values also aren't too high. The decryption time of Domingo-Ferrer was comparable to that of AES.

When considering the time it would take the sink to receive all the packets in the network, there wasn't much difference between the results of the SPF algorithms, particularly as the network size grows. When looking at the Domingo-Ferrer results, there was a more significant difference between the different variations. In this case, a lot of time can be saved by using the smaller d values. It was also found that the results weren't significantly affected by an increase in the network size. When comparing Domingo-Ferrer to SPF, the results conform to previous conclusion that SPF is better for smaller networks while aggregation becomes the superior choice as the network size grows.

When considering aggregation implemented with conventional encryption, there was a significant difference between the RC4 and AES algorithms. In this scenario, RC4 outperforms all of the AES algorithms convincingly even for the smaller network size. The results of the AES algorithms are similar for smaller network sizes, but as the number of children per node increases, the smaller key sizes start outperforming the larger ones more considerably. When comparing this scenario to Domingo-Ferrer, the conventional encryption schemes all had superior results. This is due to the large S generation time of

Domingo-Ferrer during the encryption process. It was however found that Domingo-Ferrer is as fast as RC4 once the initial encryption has been done. This is due to the fact that this algorithm does not need to decrypt the packets at each intermediate node. When considering security, the conventional encryption schemes make the system more vulnerable. So even though they are faster overall, the homomorphic encryption scheme is still the superior choice in the aggregation scenario.

It can thus be concluded that homomorphic encryption schemes are feasible for use in WSN applications. Their use is however not advantageous in all scenarios, so each application should be considered on its merits.

6.2 REAL-WORLD SIGNIFICANCE

The applications considered in this dissertation assume an honest-but-curious adversary model [35]. In this scenario an attacker's main objective is to compromise the confidentiality of the system without actively affecting its operation. This means that the attacker is an eavesdropper who keeps the system functioning properly to avoid detection but seeks to uncover sensitive system information. The protection of the integrity and availability of the system were not considered in the experiments.

Secure information aggregation is particularly useful in applications that require the statistical results of a particular set of nodes. In smart grids for instance, the central manager might be interested in the average power usage of each neighbourhood and not the individual usage of each household in a neighbourhood [35]. It is also not ideal for each meter to be able to view the usage patterns of other households so secure information aggregation can be very useful in these applications.

One could also consider a large company with many departments that wants to keep their electricity bill under control so they install a WSN system to keep track of the usage patterns. Management might not be interested in the usage patterns of each division in a particular department. They might want to see which departments are using the most electricity and tell the relevant department heads to bring down their usages. A variety of values such as the total and average usage per department could be used in this system. Without going into too much detail it is easy to see why secure information aggregation would be useful in such an application. A research and development (R&D) division of a particular department might be working on a secret project and this could become evident when looking at their usage patterns. The company might not only want to keep this information from external people, but also other people within the company, even those in the same department. So in this case, secure information aggregation can be very useful. Applications such as movement detection in perimeter control can also benefit from this method [2].

As already mentioned, the operations that are supported by Domingo-Ferrer are addition, subtraction and multiplication. The division operation is not supported by this scheme. So when calculating the average, the intermediate nodes have to calculate the sum and keep track of the number of nodes in the network. If the number of nodes is known before hand and it is known that it won’t change frequently then the latter is not required. The sink would then be able to decrypt the packets and find the average. So any results that use a combination of addition, subtraction and multiplication can be obtained from the intermediate nodes and the division operation can be left to the sink when using this scheme.

6.3 FUTURE WORK

The recommended future work that can be done is looking at a more accurate model to enable network designers to directly compare aggregation to address centric routing without the need to simulate the specific network topology. In this way, knowing the number of nodes in the network and other factors such as the number of nodes in each cluster, a designer would be able to design an optimal network for a specific algorithm instead of the other way around.

REFERENCES

[1] C Castelluccia, E Mykletun, and G Tsudik, "Efficient Aggregation of encrypted data in Wireless Sensor Networks," in *Mobile and Ubiquitous Systems: Networking and Services Conference*, San Diego, CA, USA , 2005, pp. 109-117.

[2] J Girao, D Westhoff, and M Schneider, "CDA: Concealed Data Aggregation for Reverse Multicast Traffic in Wireless Sensor Networks," in *IEEE International Conference on Communications*, 2005, pp. 3044 - 3049.

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

[4] C Castelluccia, E Mykletun, and G Tsudik, "Efficient Aggregation of encrypted data in Wireless Sensor Networks," in *Mobile and Ubiquitous Systems: Networking and Services Conference*, 2005, pp. 109-117.

[5] N Saputro and K Akkaya, "Performance Evaluation of Smart Grid Data Aggregation via Homomorphic Encryption," in *IEEE Wireless Communication and Networking Conference (WCNC)*, Shanghai, 2012, pp. 2945-2950.

[6] C Aguilar-Melchor, S Fau, F Fontaine, G Gogniat, and R Sirdey, "Recent Advances in homomorphic encryption," *IEEE SIGNAL PROCESSING MAGAZINE*, vol. 30, no. 2, pp. 108-117, Mar 2013.

[7] H Chan, A Perrig, and D Song, "Secure hierarchical in-network aggregation in sensor networks," in *ACM conference on Computer and communications security*, Alaxandria, VA, 2006, pp. 278 - 287.

[8] C Buratti, A Conti, D Dardari, and R Verdone, "An Overview on Wireless Sensor Networks Technology and Evolution," *Sensors*, vol. 9, no. 9, pp. 6869-6896, Aug 2009.

[9] W Jiang, H Jin, C Yu, and C Liu, "Introduction and Overview of Wireless Sensor Networks," in *Handbook of Research on Developments and Trends in Wireless Sensor*

Networks: From Principle to Practice.: Information Science Reference, 2010, ch. 1, pp. 1-19.

[10] C Townsend, S Arms, and Inc. Microstrain, "Wireless Sensor Networks: Principles and Applications," in *Sensor Technology Handbook*.: Newnes, 2004, ch. 22, pp. 575-589.

[11] William Stalling, "Intoduction," in *Network Security Essentials*.: Peason Education Inc, 2011, pp. 15-40.

[12] D Puccinelli and M Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *IEEE Circuits and Systems Magazine*, vol. 5, no. 3, pp. 19 - 31, Sep 2005.

[13] Y Zhou, Y Fang, and Y Zhang, "Securing wireless sensor networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 3, pp. 6-28, Sep 2008.

[14] Y Wang, G Attebury, and B Ramamurthy, "A survey of security issues in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 8, no. 2, pp. 2-23, Feb 2007.

[15] C Pfleeger and S Pfleeger, "I can't get no satisfaction," in *Analyzing Computer Security*. Michigan: Peason Education International, 2011, pp. 596-657.

[16] A Giani et al., "Smart Grid Data Integrity Attacks: Characterizations and Countermeasures," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1244-1253, Apr 2013.

[17] A. Wood and J Stankovic, "Denial of service in sensor networks," *IEEE Computer*, vol. 35, no. 10, pp. 54-62, Dec 2002.

[18] H Shafiei, A Khonsari, H Derakhshi, and P Mousavi, "Detection and mitigation of sinkhole attacks in wireless," *Journal of Computer and System Sciences*, vol. 80, no. 3, pp. 644-653, May 2014.

[19] M Kim, I Doh, and K Chae, "Denial-of-service (DoS) detection through practical

entropy estimation on hierarchical sensor networks," in *IEEE Advanced Communication Technology*, Phoenix Park, 2006, pp. 1562-1566.

[20] K. Gill and S Yang, "A scheme for preventing denial of service attacks on wireless sensor networks," in *IEEE Industrial Electronics*, Porto, 2009, pp. 2603 - 2609.

[21] D Raymond and S Midkiff, "Denial-of-Service in Wireless Sensor Networks: Attacks and Defenses," *IEEE Pervasive Computing*, vol. 7, no. 1, pp. 74-81, Jan 2008.

[22] J Liu, Y Xiao, S Li, W Liang, and C. Chen, "Cyber Security and Privacy Issues in Smart Grids," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 981-997, Jan 2012.

[23] H Khurana, M Hadley, Ning L, and D Frincke, "Smart-grid security issues," *IEEE Security & Privacy*, vol. 8, no. 1, pp. 81-85, Jan 2010.

[24] A Schoofs, A Guerrieri, D Delaney, G O'Hare, and A Ruzzelli, "ANNOT: Automated Electricity Data Annotation Using Wireless Sensor Networks," in *IEEE Sensor Mesh and Ad Hoc Communications and Networks (SECON)*, Boston, MA, 2010, pp. 1-9.

[25] M Anbya, M Salehuddin, S Hadisupadmo, and E Leksono, "Wireless sensor network for single phase electricity monitoring system via Zigbee protocol," in *IEEE Control, Systems & Industrial Informatics (ICCSII)*, Bandung, 2012, pp. 261-266.

[26] M Erol-Kantarci and H Mouftah, "Wireless Sensor Networks for Cost-Efficient Residential Energy Management in the Smart Grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 2, pp. 314-325, Mar 2011.

[27] T Bin et al., "Study of Attacks and Countermeasures in Wireless Sensor Networks ," *Advances in Information Sciences and Service Sciences*, vol. 8, no. 4, pp. 311-320, May 2012.

[28] P Ganesan et al., "Analyzing and Modeling Encryption Overhead for Sensor Network Nodes," in *CM International Workshop on Wireless Sensor networks and Applications, WSNA*, San Diego, CA, 2003, pp. 151-159.

[29] C Pfleeger and S Pfleeger, "Not All as It Seems," in *Analyzing Computer Security*. Michigan: Peason Education International, 2011, pp. 520-570.

[30] W Yuan et al., "ITARS: trust-aware recommender system using implicit trust networks ," *IET Communications*, vol. 4, no. 14, pp. 1709 - 1721 , Sep 2010.

[31] A Perrig, R Szewczyk, J Tygar, V Wen, and D Culler, "SPINS: Security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521-534, Sep 2002.

[32] D Liu and P Ning, "Efficient Distribution of Key Chain Commitments for Broadcast AuthenticationDistributed Sensor Networks," in *10th Annual Network and Distributed System*, San Diego, CA, 2003, pp. 263-276.

[33] D Liu and P Ning, "Multilevel mTESLA: Broadcast Authentication for Distributed Sensor Networks," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, no. 4, pp. 800-836, Nov 2004.

[34] M Hashmi, S Hanninen, and K Maki, "Survey of smart grid concepts, architectures, and technological demonstrations worldwide," in *Innovative Smart Grid Technologies (ISGT Latin America)*, Medellin, 2011, pp. 1-7.

[35] F Li, B Luo, and P Liu, "Secure Information Aggregation for Smart Grids Using Homomorphic Encryption," in *First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Gaithersburg, MD, 2010, pp. 327-332.

[36] M Ding, X Cheng, and G Xue, "Aggregation tree construction in sensor networks ," in *IEEE 58th Vehicular Technology Conference*, 2003, pp. 2168 - 2172.

[37] E Fasolo, M Rossi, J Widmer, and M Zorzi, "In-network aggregation techniques for wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 14, no. 2, pp. 70-87, Apr 2007.

[38] S Pattem, B Krishnamachari, and R Govindan, "he impact of spatial correlation on routing with compression in wireless sensor networks," in *Information Processing in Sensor Networks*, vol. 4, Berkeley, CA, Aug 2004, pp. 28-35.

[39] K Akkaya and M Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 325-349, May 2005.

[40] I Solis and K Obraczka, "The impact of timing in data aggregation for sensor networks," in *IEEE International Conference on Communications*, Paris, France, 2004, pp. 3640-3645.

[41] C Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," in *STOC ACM Symposium on Theory of Computing*, 2009, pp. 169-178.

[42] C Aguilar-Melchor, P Gaborit, and J Herranz, "Additively homomorphic encryption with d-operand multiplications," in *CRYPTO 2010: International Cryptology Conference*, 2010, pp. 138–154.

[43] C Fontaine and F Galand, "A survey of homomorphic encryption for nonspecialists," *Eurasip Journal on Information Security*, vol. 2007, no. 1, pp. 1-10, Oct 2007.

[44] W Stalling, "Public-Key Cyptography and Message Authentication," in *Network Security Essentials*.: Pearson Education Inc, 2011, pp. 75-110.

[45] Z Erkin, J Troncoso-Pastoriza, R Lagendijk, and F Perez-Gonzalez, "Privacy-preserving data aggregation in smart metering systems: an overview," *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 75-86, Mar 2013.

[46] A Bartoli et al., "Secure Lossless Aggregation for Smart Grid M2M Networks," in *IEEE Smart Grid Communications Conference*, Gaithersburg, MD, September 2010, pp. 333 - 338.

[47] R Lu, X Liang, X Li, X Lin, and X Shen, "EPPA: An Efficient and Privacy-Preserving Aggregation Scheme for Secure Smart Grid Communications," *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, vol. 23, no. 9, pp. 1621 - 1631, Sep 2012.

[48] S Ozdemir and Y Xiao, "Secure data aggregation in wireless sensor networks: An oveview," *Computer Networks*, vol. 53, no. 12, pp. 2022-2037, Aug 2009.

[49] L Hu and D Evans, "Secure aggregation for wireless networks," in *Applications and the Internet Workshops*, Orlando, FL, 2003, pp. 384-391.

[50] W Du, J Deng, Y Han, and P Varshney, "A witness-based approach for data fusion assurance in wireless sensor networks," in *IEEE Global Telecommunications Conference*, San Francisco, CA, 2003, pp. 1435-1439.

[51] K Wu, D Dreef, B Sun, and Y Xiao, "Secure data aggregation without persistent cryptographic operations in wireless sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 100-111, Jan 2007.

[52] S Ozdemir, "Concealed Data Aggregation in Heterogeneous Sensor Networks using Privacy Homomorphism," in *IEEE International Conference on Pervasive Services*, Istanbul, Turkey, 2007, pp. 165–168.

[53] C Pu and W Chung, "Group key update method for improving RC4 stream cipher in wireless sensor networks," in *International Conference on Convergence Information Technology*, Gyongju, 2007, pp. 1366-1371.

[54] J Domingo-Ferrer, "A Provably Secure Additive and Multiplicative Privacy Homomorphism," in *Proceeding ISC '02 Proceedings of the 5th International Conference on Information Security*, London, UK, 2002, pp. 471-483.

[55] W Stallings, "Pseudorandom Number Generation and Stream Ciphers," in *Cryptography and Network Security: Principles and Practice*. New York, United States of America: Pearson Education, 2011, ch. 7, pp. 218-242.

[56] Y Qian and C Zhang, "RC4 state and its applications ," in *International Conference on Privacy, Security and Trust (PST)*, Montreal, 2011, pp. 264-269.

[57] K Prajapati and J Nyathi, "An Efficient Key Update Scheme for Wireless Sensor Networks," in *ICWN: Proceedings*, Las Vegas, 2006, pp. 8-14.

[58] W Stallings, "Advanced Encryption Standard," in *Cryptography and Network Security: Principles and Practice*. New York, United States of America: Pearson Education, 2011, ch. 5, pp. 47-191.

[59] A Banu and R Velayutham, "Secure communication in Wireless Sensor Networks using AES algorithm with delay efficient sleep scheduling," in *Emerging Trends in Computing, Communication and Nanotechnology (ICE-CCN)*, Tirunelveli, 2013, pp. 706 - 711.

[60] F Zhang, R Dojen, and T Coffey, "Comparative performance and energy consumption analysis of different AES implementations on a wireless sensor network node," *International Journal of Sensor Networks*, vol. 10, no. 4, pp. 192-201, 2011.

[61] T Burchfield, S Venkatesan, and D Weiner, "Maximizing throughput in ZigBee wireless networks through analysis, simulations and implementations," in *Proc. Int. Workshop Localized Algor. Protocols WSNs*, 2009, pp. 15-29.

[62] Crossbow. (2004) TelosB datasheet. Document Part Number: 6020-0094-01 Rev B.

[63] Sylvain Pelissier. (2009) Cryptography algorithms for TinyOS. [Online]. http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/crypto/index.html

[64] G Anastasi, M Conti, M Di Francesco, and A Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, no. 3, pp. 537–568, May 2009.