UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

A FRAMEWORK FOR EVALUATING COUNTERMEASURES AGAINST SYBIL ATTACKS IN WIRELESS SENSOR NETWORKS

by

Servapalan Govender

Submitted in partial fulfilment of the requirements for the degree

Master of Engineering (Computer Engineering)

in the

Faculty of Engineering, the Built Environment and Information Technology

UNIVERSITY OF PRETORIA

August 2010

Acknowledgements

I would like to fervently thank my mum, Vijialutchmi, for the kind, encouraging words during the entire period of my masters. There were times when I wanted to give up but she always said “Just complete it and when it’s over you can relax”. If it weren’t for her, I know I would not have gone this far.

I give thanks to my dad, Marimuthu, for reminding me to concentrate on the task at hand and to stop eulogising its completion. I must admit that without these gentle promptings and coaxing, I’d probably be still searching for a topic. My dad has an uncanny ability for showing the problem as well as giving the solution in a simple comment.

To my younger brother, Avinesan, I give my heartfelt thanks for being there so I could bounce ideas off. His valuable comments and suggestions always proved to be worthwhile. His knowledge on research methodology principles was invaluable to my masters.

To my elder brother, Kuvarshan, I am grateful to him for reviewing some of my drafts and giving insightful comments. This assisted greatly with the structure of my thesis. Also, he always reminded me to maintain a balanced life so that I kept my sanity during the course of my masters.

To my fiancé, Deshnee, I’m eternally grateful for staying up late with me so I could complete this thesis. I know it wasn’t easy when I was stressed and not good company to be around but she stayed at my side. I also thank her for tolerating my frustration and sacrificing “our” time so that I could focus on studying. She also helped with proof reading of my thesis.

I must thank Prof GP Hancke for the opportunity to do my masters. His knowledge on WSNs and his sincere appreciation for the subject gave me the inspiration to complete this thesis. His encouraging words and his eagerness for the completion of my studies is also greatly appreciated.

To my supervisor, Dr GP Hancke, I’m grateful for being so understanding. His vast knowledge and exposure in the field was so apparent in his significant input to my study. He added so much of value to my thesis that was initially so unrefined. He put in so much of time and effort to steer my masters in the right direction that it was nearly impossible for me to have gone astray.

Finally, I must give thanks to God, Sri Swami Sivananda and Pujye Swamiji for giving me the opportunity to attempt my masters. For without their strength, their courage, their grace and eternal bliss, none of this would have even been possible.

SUMMARY

A FRAMEWORK FOR EVALUATING COUNTERMEASURES AGAINST SYBIL ATTACKS IN WIRELESS SENSOR NETWORKS

Author: Servapalan Govender

Supervisor: Dr. G.P. Hancke

Department: Electrical, Electronic and Computer Engineering

University: University of Pretoria

Degree: Master of Engineering (Computer Engineering)

Although Wireless Sensor Networks (WSNs) have found a niche in numerous applications, they are constrained by numerous factors. One of these important factors is security in WSNs.

There are various types of security attacks that WSNs are susceptible to. The focus of this study is centred on Sybil attacks, a denial of service attack. In this type of attack, rogue nodes impersonate valid nodes by falsely claiming to possess authentic identities, thereby rendering numerous core WSN operations ineffective. The diverse nature of existing solutions poses a difficult problem for system engineers wanting to employ a best fit countermeasure.

This problem is the largely unanswered question posed to all system engineers and developers alike whose goal is to design/develop a secure WSN. Resolving this dilemma proves to be quite a fascinating task, since there are numerous factors to consider and more especially one cannot assume that every application is affected by all identified factors.

A framework methodology presented in this study addresses the abovementioned challenges by evaluating countermeasure effectiveness based on theoretical and practical security factors. Furthermore, a process is outlined to determine the application's

engineering requirements and the framework also suggests what security components the system engineer ought to incorporate into the application, depending on the application's risk profile. The framework then numerically aligns these considerations, ensuring an accurate and fairly unbiased best fit countermeasure selection. Although the framework concentrates on Sybil countermeasures, the methodology can be applied to other classes of countermeasures since it answers the question of how to objectively study and compare security mechanisms that are both diverse and intended for different application environments.

The report documents the design and development of a comparative framework that can be used to evaluate countermeasures against Sybil attacks in wireless sensor networks based on various criteria that will be discussed in detail. This report looks briefly at the aims and description of the research. Following this, a literature survey on the body of knowledge concerning WSN security and a discussion on the proposed methodology of a specific design approach are given. Assumptions and a short list of factors that were considered are then described. Metrics, the taxonomy for WSN countermeasures, the framework and a formal model are developed. Risk analysis and the best fit methodology are also discussed. Finally, the results and recommendations are shown for the research, after which the document is concluded.

Keywords: WSN, Sybil attacks, countermeasure evaluation framework, risk profiling and analysis, best fit analysis, countermeasure taxonomy, security

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

OPSOMMING

’N RAAMWERK OM TEENMAATREËLS VIR SYBIL-AANVALLE IN DRAADLOSE SENSORNETWERK TE EVALUEER

Outeur: Servapalan Govender

Studieleier: Dr. G.P. Hancke

Departement: Elektriese, Elektroniese & Rekenaaringenieurswese

Universiteit: Universiteit van Pretoria

Graad: Magister in Ingenieurswese (Rekenaar Ingenieurswese)

Hoewel Draadlose Sensornetwerke (DSN) ’n nismark gevind het in ’n wye verskeidenheid toepassings, word hulle nogtans aan bande gelê deur baie faktore. Een van hierdie belangrike faktore is sekuriteit in DSN.

DSN is onderhewig aan verskillende tipes sekuriteitsaanvalle. Die fokus van hierdie studie is Sybil-aanvalle, waardeur diens geweier word. In hierdie tipe aanvalle stel ongeldige nodes geldige nodes voor deur valslik voor te gee om egte identiteite te hê en sodoende talryke DSN-kernbedrywighede oneffektief te maak. Die uiteenlopende aard van oplossings stel ’n moeilike problem aan sisteemingenieurs wat die mees geskikte teenmaatreël wil aanwend.

Hierdie problem is die hoofsaaklik onbeantwoorde vraag wat gestel word aan sowel alle sisteemingenieurs as ontwikkelaars wat ten doel het om ’n veilige DSN te ontwerp of ontwikkel. Die oplossing van hierdie dilemma het geblyk om ’n heel boeiende taak te wees, aangesien daar talryke faktore is wat in ag geneem moet word en veral omdat ’n mens nie kan aanvaar dat al die geïdentifiseerde faktore ’n invloed op elke aanwending het nie.

’n Raamwerkmetodologie wat in hierdie studie aangebied word, gee aandag aan bogenoemde uitdagings deur die effektiwiteit van teenmaatreëls te evalueer aan die hand van teoretiese en praktiese sekuriteitsfaktore. Daarbenewens word ’n proses beskryf om die ingenieursvereistes van die aanwending te bepaal en die raamwerk stel ook die sekuriteitskomponente voor wat die ingenieur by die aanwending behoort in te sluit, afhangende van die risikoprofiel. Die raamwerk rig dan hierdie oorwegings numeries, wat ’n akkurate en grootliks onbevooroordeelde bes moontlike keuse van teenmaatreëls verseker. Alhoewel die raamwerk op Sybil-teenmaatreëls konsentreer, kan die metodologie aangewend word vir ander teenmaatreëls in ander kategorieë, aangesien dit die vraag beantwoord oor hoe om ’n objektiewe studie en vergelyking te doen van uiteenlopende sekuriteitsmeganismes wat vir verskillende aanwendingsomgewings bedoel is.

Die verslag dokumenteer die ontwerp en ontwikkeling van ’n vergelykende raamwerk wat gebruik kan word om teenmaatreëls teen Sybil-aanvalle in DSN te evalueer op die grondslag van verskeie kriteria wat in detail bespreek sal word. Hierdie verslag oorweeg kortliks die doelwitte en beskrywing an die navorsing. Daarna volg ’n literatuuroorsig oor die totale beskikbare kennis oor DSN-sekuriteit en ’n bespreking van die voorgestelde metodologie van ’n spesifieke ontwerpbenadering. Aannames en ’n kort lys faktore wat oorweeg is, word vervolgens beskryf. Maatstawwe, die metodologie van DSN teenmaatreëls, die raamwerk en ’n formele model word ontwikkel. Risiko-analise en die mees geskikte metodologie word ook bespreek. Ten slotte word die bevindinge en aanbevelings van die navorsing aangetoon.

Sleutelwoorde: WSN, Sybil-aanvalle, teenaanvalle-evaluasieraamwerk, risikoprofilering en-ontleding, bestepassing-ontleding, teenaanvalleklassifikasie, sekuriteit

LIST OF ABBREVIATIONS

AFECA	Adaptive Fidelity Energy-Conserving Algorithm
AOA	Angle of Arrival
BS	Base Station
CA	Certificate Authority
CAE	Common Administrative Entity
CEC	Cluster-based Energy Conservation
CIA	Central Identification Authority
DOS	Denial of Service
GEAR	Geographic and Energy Aware Routing
GPS	Global Positioning System
GPSR	Greedy Perimeter Stateless Routing
IDS	Intrusion Detection System
KMS	Key Management Schemes
LOS	Line-of-Sight
MLE	Maximum Likelihood Estimation
NIST	National Institute of Standards and Technology
NLOS	Non-Line-of-Sight
PKC	Public Key Cryptography
PKI	Public Key Infrastructure
PKS	Public Key Scheme
POC	Proof of Concept

PPM	Parts per Million
RF	Radio Frequency
RKP	Random Key Pre-distribution
RRT	Radio Resource Testing
RT	Resource Testing
TBS	Trust Based Stations
TDOA	Time Difference of Arrival
TIA	Trusted Identification Authority
TOA	Time of Arrival
TTP	Trusted Third Party
WSN	Wireless Sensor Network

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

Owing to the recent significant technological advances in microcontroller design and the semiconductor fabrication processes, inexpensive electronic sensor devices have become the ideal choice for wireless sensor nodes, i.e. the most basic building blocks for wireless sensor networks (WSNs). However, these devices suffer from memory, computational power and energy limitations. Also, owing to the usually unmanned and sometimes hostile environments in which these nodes are deployed, measures have to be taken to ensure that the system discriminates between true data and false data and responds accordingly.

General or traditional network security principles either fail to an extent, if not completely, when applied to WSNs owing to the nature of the deployment and physical limitations of the node itself. Currently, no guaranteed security solution exists, but some advances have been made in this field of study.

WSNs are susceptible to various types of attacks, e.g. wormhole attacks, sinkhole attacks, Sybil attacks, physical node tampering etc. This research concentrates on Sybil attacks, where illegitimate nodes falsely lay claim to genuine identities, thereby disrupting several fundamental WSN operations. Many solutions have been proposed to counter Sybil attacks and each proposal has demonstrated or claimed various degrees of functionality and success. Moreover, each countermeasure has different requirements and constraints for the WSN environment it is used in. This diverse nature of solutions poses a difficult problem for system engineers wanting to employ a best fit countermeasure, since measuring the performance of these countermeasures is not a small order.

The largely unanswered question posed to all system engineers and developers alike, whose goal is to design/develop a secure WSN, has not thus far been addressed, viz. how is it possible to uniformly measure the effectiveness of security for the countermeasure and its suitability to fit a specific application’s needs when there are so many variables to consider? These variables can loosely be classified as theoretical security constraints or practical security constraints. In the engineering world, where the theory and practice of security do not necessarily coincide, it is crucial that WSN designs consider both theoretical and practical issues, thus creating the best of breed security for applications.

When attempting to address this challenge, there are various factors to consider and more especially one cannot assume that every application is affected by all identified factors. In

fact, two identical applications could place very different emphasis on a specific constraint or requirement. This effectively eliminates the possibility of creating a “one size fits all” solution and begs for a dynamic comparison of both countermeasures and application requirements. Broadly speaking, one has to match the applications goals and applications requirements with functionality offered by the abundance of available countermeasures. This part of the design process is no small undertaking, usually consumes a considerable amount of time and can consequently have a severe impact on rapid application designs.

The aim of this study is to achieve a framework to evaluate Sybil countermeasures consistently to facilitate a best fit selection. A consequence of the developed framework is that one now has a reference for comparing new countermeasures to old ones just by evaluating factors in the new countermeasure rather than judging if there are already mechanisms available in the existing countermeasures. Furthermore, countermeasure designers can improve on existing countermeasures by considering what the framework indicated was lacking in existing solutions. This would in essence allow superior countermeasures to be developed more rapidly and past mistakes not to be repeated.

CHAPTER 2 RESEARCH PROBLEM AND OBJECTIVES

This chapter elaborates on the problem definition of this research study i.e. the dilemma faced by non-security experts in selecting optimum countermeasures for their applications. Thereafter, it presents solution objectives in the form tasks that would facilitate creating an evaluative framework for countermeasures.

Security techniques used in traditional networks cannot be directly substituted for WSN security. As mentioned previously, numerous techniques have been developed to address threats in WSNs. However, software developers and designers alike are faced with the dilemma of managing security without having a pre-defined framework/guideline for choosing one technique over another. This, together with the lack of practical implementation requirements for these countermeasures, poses quite a challenge and increases the time to realise rapid solutions.

The purpose of this research was to attempt to create a point-of-reference scheme/framework for WSNs that could be used to evaluate countermeasures against Sybil security breaches. The idea was to develop and introduce a grading system that would rate the specific defence according to certain rules and criteria. Furthermore, depending on the objectives of the network designer, the significance of these different factors would have to be adjusted accordingly to cater for specific needs. Typical criteria would be factors such as code size, data size, processing time and network overhead.

It should be noted that the proof of concept (POC) tool is an attempt to achieve two objectives:

- To create a grading system for countermeasures against Sybil attacks (countermeasure evaluation)
- To assist an application developer to choose a suitable countermeasure against Sybil attacks (risk analysis and best fit computation).

It is the intention of the author to use the proof of concept tool as a type of library/repository to hold more accurate simulated characteristics of current and future Sybil countermeasures in the hope that it will become as comprehensive as possible. This being said, most, if not all solutions were characterised based on the functional value of

their success and where possible the claimed performance statistics by the respective researchers were used in the tool without verification.

It is also hoped that, eventually, this tool may lead to better insight into the creation of more optimal countermeasures against Sybil attacks.

CHAPTER 3 LITERATURE REVIEW

This chapter presents a backdrop to familiarise the reader with enough background information regarding WSN operations. Issues like WSN topologies, routing protocols, WSN security, the Sybil attack and countermeasures against them are covered in an appropriate degree of detail.

3.1 OVERVIEW OF WIRELESS SENSOR NETWORKS

Sensor nodes are usually dispersed in a sensor field [1] as shown in Figure 3.1.

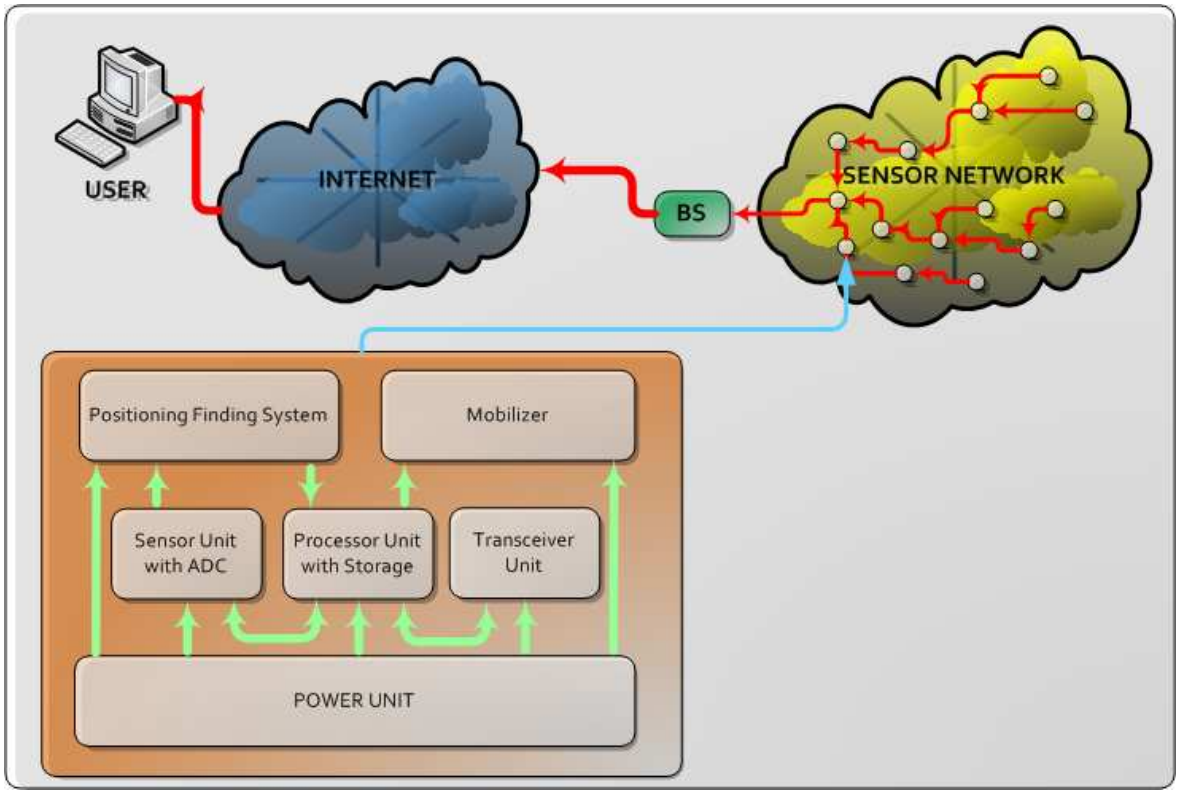

Figure 3.1: Sensor Network Architecture [2]

Each node collects sensed data and, using some form of multi-hop protocol, routes the data back to the sink or base station (BS) and eventually to the front end application. It is quite possible that the network may have many sinks and/or BSs. These nodes consist of essentially four basic elements, i.e. a power unit, a sensing unit, a processing unit and a transceiver unit. Arguably, the most important element (in terms of limitations) is the power unit. While most traditional networks focus on quality of service (QOS), WSN developers and WSN researchers alike tend to concentrate more effort on optimal power consumption [1]. Most of these solutions revolve around the following techniques: Duty cycling, data-driven reduction schemes, energy-efficient data acquisition and mobility-

based techniques [3]. Security solutions for traditional networks are not feasible owing to their large power and computational requirements. However, as critical as it is to conserve power, it is sometimes required that data transmissions be communicated securely. The key to durable and secure WSN designs is to find the optimal precarious balance between power consumption and the required robust security solution.

3.2 WSN TOPOLOGIES

WSN applications are arranged in either homogeneous or heterogeneous topologies.

In the homogeneous arrangement, all member nodes send their data to the BS either via single-hop or multi-hop techniques. Furthermore, from the standpoint of all member nodes, the BS is the only leader or head of the system. Having this type of arrangement has its advantages and disadvantages, but it is better suited to smaller networks.

In heterogeneous layouts (sometimes known as cluster-based networks), the entire network is divided into smaller networks or clusters, either automatically or through human intervention. In each cluster, the cluster head (CH) is responsible for allocating bandwidth to all its member nodes, collecting sensed data, managing data sent by the BS and sending data to the BS [4]. Typically, one would find one CH associated with each cluster, although this is not a requirement, as one could assign more CHs for redundancy. It is obvious that every CH can communicate with all nodes in its cluster but it should be mentioned that it is sometimes a pre-requisite that this communication is in the form of a single-hop message, i.e. every node is within communication range of the CH. Likewise, it should be noted that some applications require that the BS be able to talk directly with every CH,; however, if cryptographic techniques are employed the network is not limited with regard to scalability. In other words, if cryptographic techniques are not employed, the transmission range of the BS would have to cover all CHs, having the consequence that new CHs being introduced will have to be within this communication range. For a given number of nodes in a network, cluster-based topologies have more intimate management of their member nodes in view of the number of hops being decreased when communicating to border nodes.

In both topologies (i.e. heterogeneous and homogenous topologies), three types of communication methods are available, namely unicast, multicast and broadcast. This document attaches the following meaning to these terms:

- Unicast – If a node wants to send a message to just one other node and no-one else.
- Multicast – If a node wants to send a message to a selected few nodes and no-one else.
- Broadcast – If a node wants to send a message to every node in the network.

3.3 ROUTING PROTOCOLS

There are various routing protocols that WSN nodes use to communicate. Existing protocols are usually designed and optimised for the limited computational, storage and energy capabilities of nodes, in addition to the specific characteristics of the application, but are rarely optimised for security requirements and considerations [5]. Routing protocols can be classified on a high level as being either an active or passive routing protocol, depending on how they respond to forwarding messages and maintenance of their routing tables. Below is a list of some common routing protocols that can be adversely affected by WSN security threats. The number of protocols affected by the Sybil attack (explained later) should be noted.

Table 3.1: Routing Protocols and Associated Attacks [5]

Routing Protocol	Possible Associated Attacks
TinyOS Beaconing	Bogus routing information, selective forwarding, sinkholes, Sybil Attack, wormholes, HELLO floods
Directed Diffusion or its Multipath variants	Bogus routing information, selective forwarding, sinkholes, Sybil Attack, wormholes, HELLO floods
Geographic Routing (GPSR, GEAR)	Bogus routing information, selective forwarding, Sybil Attack
Minimum Cost Forwarding	Bogus routing information, selective forwarding, sinkholes, wormholes, HELLO floods
Clustering based Protocols (LEACH, TEEN, PEGASIS)	Selective forwarding, HELLO floods
Rumour Routing	Bogus routing information, selective forwarding, sinkholes, Sybil Attack, wormholes
Energy Conserving Topology Maintenance	Bogus routing information, Sybil Attack,

(SPAN, GAF, CEC, AFECA)	HELLO floods

3.4 WSN APPLICATION TYPES

WSN application types can usually be categorised into the following classifications [1, 2, 6, and 7]:

- Data Aggregation
- Data Dissemination
- Fault-tolerant schemes such as
- Redundancy Mechanisms
- Distributed Storage
- Dispersity
- Multipath Routing and
- Network Topology Maintenance
- Voting
- Fair Resource Allocation
- Misbehaviour Detection
- Group-based Decisions
- Security/Sensing/Detection System
- Routing

Note that WSN applications quite often consist of a composite of the above functions.

3.5 SECURITY ISSUES CONCERNING WSNS

The conventional security considerations for WSN design are confidentiality, integrity, authentication and availability. These goals were consolidated from [2, 6, 8, 9, and 10].

These objectives are similar to traditional networks but there are also further WSN-specific considerations that are taken into account. These are authorisation of nodes, non-repudiation of source nodes, the freshness of data, forward secrecy and backward secrecy of data [2].

Apart from these goals, the designer/developer has to know what threats are imminent. A combined list of attacks taken from [2, 6, 8, 9, 11 and 12] is given below:

- Sinkhole attacks
- Attacks on routing protocols (e.g. “HELLO” flood attacks, spoofed/altered routing and selective forwarding)
- Sybil attacks
- Wormholes
- Physical tampering (e.g. node capture and radio jamming)
- Denial of service (DOS) (e.g. collisions).

Several solutions have been proposed to counter security breaches, e.g. key management protocols, secure routing protocols, link-layer encryption, spread spectrum techniques and intruder detection.

3.6 THE SYBIL ATTACK

In Sybil attacks, a type of DOS attack, a rogue node or entity dishonestly mimics multiple identities to represent several other nodes in a WSN. This type of attack was first discussed in P2P networks [13], in which traditional digital signatures are a viable solution. However, this solution, because of large computing overheads, is unsuitable for the WSN environment.

There are various configurations or ways in which Sybil attacks can be launched [7]. These are direct communication attacks, indirect communication attacks, simultaneous attacks, non-simultaneous attacks, fabricated identities and stolen identities. To illustrate the classifications above, suppose there is a network with valid nodes A, B, C and D initially *(see Figure 3.2)*. Node D is then compromised by an attacker and allows other rogue nodes to be introduced into the system i.e. nodes E, F and G. It should be noted that these extra nodes are logical entities and may or may not be physical entities.

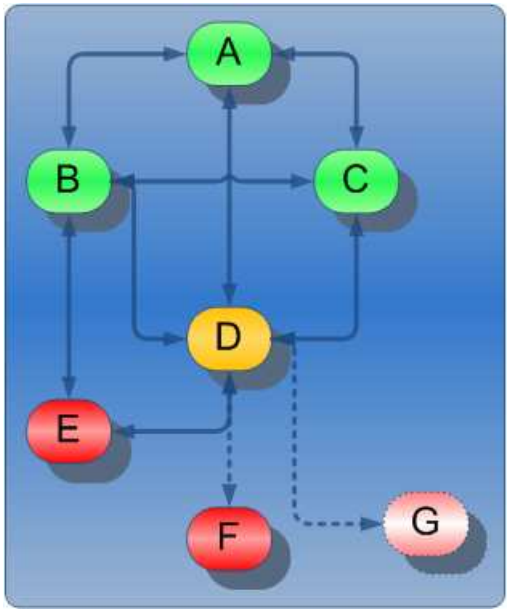

Figure 3.2: Types of Sybil Attacks

For direct attacks, the valid network can communicate directly with the rogue nodes e.g. node B communicates with node E. The other option is for node D to claim that it is in contact with node F and act as a gateway or forwarding node for communication between the valid nodes and F. In reality, nodes D and F would be the same entity.

Unlike simultaneous attacks where all Sybil identities are presented to the network, non-simultaneous attacks occur when just one or a portion of Sybil identities attack the system [7]. Non-simultaneous attacks can also be launched if a Sybil identity seems to leave the network and re-appear later on. Suppose node G *(see Figure 3.2)* is mapped to an actual physical node and this device cycles through various identities. If the system sees the nodes leaving and new nodes entering the network, this would be a type of non-simultaneous attack, since not all the identities are used at the same time. However, if the system only sees new nodes being presented and is unaware of the identities not being used, then this would be a simultaneous attack.

The issue of fabricated or stolen identities is fairly self-explanatory. This dimension of attacks refers to whether a valid node had this identity prior to the attack (e.g. the case of node D) or if the rogue nodes introduce themselves with new identities (e.g. nodes E, F and G).

It should be noted that an attack doesn't necessarily have to use just one type of the configuration presented above, but can be a hybrid of these. In fact, the more diverse the attack, the harder it would be to prevent it.

Sybil attacks can have dire effects on the performance and reliable operation of the network. The performance of mechanisms such as routing protocols (especially multi-path routing and dynamic routing protocols), voting, incentive schemes, distributed storage and other security mechanisms cannot be guaranteed [13]. To illustrate this, suppose there is a vote in Figure 3.2 to see which node should be the new CH and the rule is that every node can only nominate one other node for this role. For simplicity's sake, it is assumed that all Sybil nodes are mapped to node D (i.e. there is only one hardware device). Compared to the number of valid entities, the number of Sybil entities is small. However, votes are based on identities and no matter which node the valid nodes nominate, their combined decision can easily be outweighed by the Sybil identities. Similarly, other core WSN functions can be disrupted by a Sybil attack.

3.7 TECHNIQUES USED TO COUNTER SYBIL ATTACKS

Various techniques that were proposed by researchers yielded varying documented degrees of success [14]. Below is a list and description of methods that were surveyed. Note that it is not implied by any means that this list is mutually exclusive. It would be more accurate to note that some of the techniques described below are either children or parents of others e.g. "Localization Methods" have "Angle of Arrival" and "Received Signal Strength Indicator" as a child.

3.7.1 Resource Testing

[7, 15, 16] propose resource testing (RT) as a means of direct validation. The goal is to determine if identities are indeed independent. The assumption is that an entity is limited in some resource and that it is possible to test the boundaries of this resource in order to establish the number of identities mapped to the number of entities. In other words, the claim to independence should be backed up with sole access to certain resources. If there are no Sybil nodes, then there should be one-to-one mapping of identities to entities. [7] goes on to note that computation and storage are unsuitable for WSNs since to the attacker may possess a physical device that is orders of magnitude less constrained in these areas, i.e. an attacker does not need to use a 'standard' node. [15] proposed a method to test the

communication resources on devices by sending out a broadcast requesting all identities to respond and only accepting ones that responded in a given time interval. This was done in the interest of simultaneity. Although not specifically applicable to WSNs, [17] presents another form of resource testing. Here, the limited availability of edges between nodes is tested in social networks.

3.7.2 Radio Resource Testing

Radio resource testing (RRT) is a special type of RT that bases its method on the assumption that every independent device has only one radio and furthermore is only capable of simultaneously transmitting and receiving on one channel [7]. A case in point would be when a non-Sybil node wants to verify that none of its neighbours are Sybil entities; it would assign each identity/neighbour a channel to send a message on and request that all nodes send a message at some time. This node would then randomly choose a channel to listen on and if nothing is received, it would imply that the neighbour assigned to that channel is a Sybil identity. [7] goes on to prove formally that after several rounds of this, the probability of detecting a Sybil node is quite high. It should be stated, however, that the author does not demonstrate a protocol that inherits this behaviour or propose one that could, although a fair amount of statistical analysis has been done on the performance of this method, showing its success.

3.7.3 Trusted Certification

[15, 16] propose trusted certification methods. These sets of techniques effectively require that each node should have some certificate (some unique information that cannot be forged without being noticed) issued by a certificate authority (CA). It is the responsibility of the CA to ensure that each certificate is unique and that each node has only one certificate. [15] actually proves formally that without the presence of the CA, it is impossible to prevent Sybil attacks completely. This is in fact true; however, [15] presents no formal method for establishing this uniqueness. Real world applications have seen this method only employed through manual or human intervention [16].

3.7.3.1 Symmetric Encryption

Symmetric encryption is a technology where the same key enables encryption and decryption of a message. Usually the decryption algorithm is the reverse operation to the encryption one [18]. Symmetric encryption techniques are faced with the challenge of key

distribution i.e. how the two communicating nodes originally get their key. A case in point is if X and Y share a key and X wants to communicate with Z, then another key has to be introduced. In general, the number of keys that is needed is $n(n-1)/2$ keys.

3.7.3.2 Asymmetric Encryption

Asymmetric encryption or public key cryptography (PKC) is a system where the public key is used for encryption and the private key is used for decryption. Unlike symmetric encryption techniques where there are two entities that share a key (so any message encrypted with the key could come from either), asymmetric systems (through digital signatures) allow authentication to be achieved quite readily. Furthermore, asymmetric encryption allows for easier distribution of keys. By nature of the public key approach, one can broadcast the public key, since only the corresponding private key can decrypt any public key encrypted message [18]. In WSNs, the limited storage does not permit the use of traditional public key schemes. This is primarily due to keys being much longer for asymmetric systems than their symmetric counterparts to achieve similar levels of security.

3.7.3.2.1 Elliptic Curve Cryptography

ECC, an asymmetric encryption scheme developed in 1985, bases its approach on the mathematics of elliptic curves. Performance statistics show that ECC can obtain the same security level as RSA while using a smaller key, thus making it suitable in WSNs [19, 20]. This is achieved by using exponential algorithms to solve the elliptic curve discrete logarithm problem as opposed to RSA, where small runtime algorithms solve the large number factorisation.

3.7.3.3 Hash Functions

Hash functions are mathematical and/or logical functions that are applied to large, varying sizes of data to convert them into usually smaller but always into constant sizes of data. Hash functions enable data integrity rather than secrecy since cryptographic hashes are irreversible (i.e. one-way functions) [18]. Hash functions are akin to sealing data so that in the event that seal is broken, the recipient can know if the message has been tampered with [18, 21]. In the world of cryptography, this can be achieved by applying the hash function to the message and sending the concatenation of the message and its corresponding hash. When the message is received, the recipient should test its integrity by applying the hash

function to the message and if the result is different from the hash received, then the data cannot be deemed to be true.

3.7.4 Trusted Devices

This method is an extension of trusted certification. The primary goal here is to bind an identity to an entity cryptographically, i.e. a trusted hardware platform. This effectively means that one hardware device can only have one identity. While this approach looks promising, very few real-world applications exist. Also, it is claimed that there is no effective approach to avert one entity from acquiring multiple hardware devices other than manual intervention [22].

3.7.5 Key Management Schemes

A fundamental aspect of all cryptographic techniques is the process of creating, distributing, verifying and in general, managing keys or unique information that confirms identity. Hence key management schemes (KMS) are of great importance to security solution designers.

[19] lists and describes various key management schemes that could be used in WSNs:

- Single network-wide key,
- Pairwise key establishment,
- Trusted base station,
- Public key schemes,
- Key pre-distribution schemes,
- Dynamic key management, and
- Hierarchical key managements

According to [19], the single network-wide key scheme is the simplest of the key management techniques, something that ought to be investigated before venturing out to other more sophisticated versions of key management schemes. However, the single

network-wide key scheme fails to provide the basic requirements for WSNs if one takes into account that if a single node is compromised, the entire network is compromised.

Pairwise key establishment solved this issue by having (n-1) keys stored in each node, where n is the number of nodes in the network [19]. This, however, introduced the challenge of storage of keys; a challenge that cannot be solved with the limited capacity of WSN nodes, especially in large WSNs.

To limit the number of keys stored in each node, a trusted base station (TBS), a central identification authority (CIA) or a trust identification authority (TIA) is used. These generally more powerful and more secure nodes issue session keys that can be used for node–to-node authentication. However, in WSNs these nodes become the target for attacks. Scalability with this type of solution also poses challenges [19].

3.7.5.1 Key Pre-distribution

As mentioned earlier, key pre-distribution (KPD) is quite a challenge for key management schemes. Usually, in KPD, keys are stored in each node before deployment by some CIA/TIA/TBS. After deployment, sensor nodes go through a key-discovery phase through which they establish pairwise keys with neighbours [19]. However, it must be noted that KPD schemes don't generally guarantee that two nodes will always find a pairwise key with each other.

3.7.5.1.1 Random Key Pre-distribution

The first simple scheme [23] assumes that sensor nodes are too limited to use PKS. In random key pre-distribution (RKP), before deployment, each node is pre-loaded with a random set of K keys from a larger pool of keys, S. After deployment, each sensor exchanges it stored keys with neighbours and they are secure-linked if they find a matching pair [19, 24]. This limits the number of keys that need to be stored in each node, but the probability of finding the pairwise key with another node decreases as the size of K does. However, even if two nodes cannot find a matching pair, they can still set up a pairwise path-key if they manage to find a third node that shares a key with both of them [24].

3.7.5.1.2 q-Composite Random Key-Pre-distribution

An improvement on the RKP, q-Composite RKP, requires the number of matches that need to be found between nodes in the bootstrapping stage to be q, where $q \geq 2$. This method ensures greater network resilience against attacks from compromised nodes when compared to its predecessor [24]. It should be noted that as the number of matches above q that are established between valid nodes increases, it becomes harder for attackers to launch an attack against the system successfully. This is because an attacker would have to compromise more keys before being able to pretend to possess any one of these identities, e.g. suppose node A and node B share five keys but q=3; if an attacker wants to pretend to be A, he/she would have to compromise five keys instead of only three to be able to communicate with B.

3.7.5.1.3 Random pairwise key scheme

Something lacking in both RKP and q-composite RKP is node-to-node authentication. The advantage of having node-to-node authentication is that when a node is compromised, other nodes in the vicinity can take action to remedy this instead of the BS sending out a broadcast instruction on how to proceed. This reduces overhead significantly and ensures that each node uses a unique identity [19, 25]. In this method, during the initialisation phase, each node is randomly paired with a few selected nodes and their pairwise key as well as the identity of the selected node is stored. During the key discovery phase, each node broadcasts its identity to see if there are neighbours that have previously been paired. Cryptographic handshakes can then take place. Note that the primary difference here is that pairwise matches are done prior to deployment. No keys are therefore broadcast, as nodes only need to transmit their identities.

3.7.6 Localization Techniques

Localization is a process of calculating a node's position either relative to other reference nodes/landmarks or its global position. In WSNs, these reference nodes are called anchors primarily because their position (whether it is relative or global) is known to the network/application. Usually, because of the specialised feature requirement of these nodes, they are either base stations or cluster heads. For the purposes of terminology, the nodes that need to be localised are called unknown nodes. The fundamental approaches to localization exploit geometric relationships between anchor nodes and unknown nodes to

calculate the localization result. Usually, there are two elementary methods of localization algorithms. These are the three-point localization method and angle localization method. Note that the algorithm assumes that the data, i.e. the distance(s) or angle(s), are available. The first algorithm requires the distances between the unknown node and at least three anchor nodes, provided the anchor nodes are not in one line and that the anchor nodes are not effectively in one place. The second method necessitates the realisation of the angles between the unknown node and two anchor nodes and the distance of two anchor nodes [26, 27].

Using this information, these algorithms then use Euclidian geometry and trilateration, multi-lateration, triangulation or maximum likelihood estimation (MLE) methods to localise the unknown nodes' position [26]. It should be noted that some countermeasures require a minimum of only two anchor nodes for localization. This is a simplification of the above-mentioned methods (i.e. three-point or angle localization methods). Received signal strength indicator, angle of arrival (AOA), time of arrival (TOA) and time difference of arrival (TDOA) are examples of localization techniques.

By knowing the accurate position of an unknown node, the system can associate the entity (the actual node) with its claimed identity using an indirect method, i.e. associating the entity with a known position, which in turn maps to an identity that the entity first claims. The node can be detected as a Sybil node for one of two reasons:

- If the unknown node presents an identity that is known by the system but the location is different to what has already been mapped, this would effectively be indicative of a malicious node trying to steal an existing identity. Note that it is assumed that this is a static WSN (i.e. nodes are not moving).

- If an unknown node presents an identity that correlates to a position that is known to be mapped to another entity, then this entity is a Sybil node and both the previous identity as well as this new identity should be in question, as being transmitted by the same node.

Localization can be computed in one of two ways:

- Centralised localization - all information crucial for computation is aggregated to a central node, usually a more powerful node such as a BS or a CH, where the localization algorithm is executed. The advantage of doing this is global

programming, more memory and computation power to perform the calculation, which translates to a substantially more accurate localization result. However, like most central schema used in WSNs, this means a considerable communication load for the nodes which are close to the centralised computing node. Consequently, these nodes run out of energy more easily, which will then create an avalanche effect in disconnecting the network and the entire localization service would thereafter be unavailable for the whole network [28].

- Distributed localization - is a localization method which relies on an information interchange between different nodes. According to [28], the four most conventional distributed localization algorithms are
    - Robust position
    - Bounding box
    - Euclidean and
    - DV-Hop.

3.7.6.1 Received Signal Strength Indicator

Received signal strength indicator (RSSI) is the technique that uses the spatial correlation between the signal strength and physical location [29] to detect the presence of Sybil attacks. This method measures the received power associated with incoming message(s) from a node and relates this to a unique location and subsequently a unique ID. Usually, there are several nodes measuring this value in order to triangulate this position. In the event that another message is received having the same location but a claim to a different ID, the system will assume that this entity is a Sybil node. Usually a minimum of three nodes are required but there are claims that four nodes are required to locate a node using RSSI [30] effectively. The choice of the frequency of RSSI calculations (i.e. on which transactions does the system need to verify a node's identity claim) varies for different solutions based on the researcher's goal. Sometimes, this value is calculated only if a new node sends a message or if the system is in bootstrap, while other solutions run this calculation on every message.

3.7.6.1.1 Jakes Channel Path Loss

To cater for path loss and fading influences, typical characteristics of the wireless channel as indicated by [4, 22, 31], one should not just use the received power of the incoming message, but use Jakes's channel path loss model to calculate RSSI more accurately. William Jakes found that Rayleigh fading, a process described by the summation of complex sinusoidal signals, quite accurately models the path loss characteristics in a wireless medium [4, 31]. In summary, these equations can be manipulated to calculate RSSI:

$$RSSI_r = \frac{RSSI_t x G_{channel}}{d^{\alpha}} = \frac{RSSI_t x \|H\|^2}{d^{\alpha}} \tag{3.1}$$

where $RSSI_t = 10 \log P_{rec}$, α is the distance power descending ramp, H is the impulse response of the channel model and G is the effective gain of the transmitter's and receiver's antenna. Note the strong relation to distance.

So, to calculate the received signal ratio effectively (consequently factoring out the environmental factors), one computes:

$$\frac{RSSI_i}{RSSI_j} = \frac{\frac{RSSI_t x \|H\|^2}{d_i^{\alpha}}}{\frac{RSSI_t x \|H\|^2}{d_j^{\alpha}}} = \left(\frac{d_j}{d_i}\right)^{\alpha} . \tag{3.2}$$

From here, one can use four or more detectors to check if an entity is indeed a Sybil node.

3.7.6.2 Angle of Arrival

AOA is defined as the angle between the propagation direction of an incident wave and a pre-defined reference direction (usually referred to as the orientation angle). Often, AOA is represented as the number of degrees clockwise measured from north. The AOA is absolute when the orientation is due north, otherwise it is relative [32].

The assumption is that every node in the network has a primary axis, which is the reference angle measurement of all incoming messages. Heading is the angle between the pre-defined absolute (usually north) and the primary axis. Furthermore, all nodes have the

ability to sense the direction of an incoming message either by using an antenna array or several ultrasound receivers [32, 33].

3.7.6.3 Time of Arrival

TOA is the measured time at which the incoming message first arrives at a receiving node. The measurement is the time of transmission plus a propagation-induced time delay [34]. The assumption is that the time delay, $T_{i,j}$, between transmission from node *i* and reception at node *j*, is the distance between them divided by the propagation velocity v_p [34]. Generally, a free-space wave propagation model is used for the models. In free space, $v_p \approx 3\times10^8 m/s$. A critical success factor for TOA is the receiver's ability to estimate the arrival time of the line-of-sight (LOS) incoming message accurately.

3.7.6.4 Time Difference of Arrival

The TDOA method is an improvement of the TOA method. In TOA, there is a requirement to know actual timestamps when messages are sent and when they arrive at the anchor node. Hence there is a crucial necessity for strict time synchronisation of the whole network. The TDOA method uses the time difference of signal propagation between anchor nodes (or beacons) and unknown nodes, but not propagation time itself. In doing so, TDOA techniques reduce the requirement for stringent time synchronisation for the WSN. Essentially, there are two fundamental methods of achieving the TDOA in WSNs [26]:

- The measurement of the TOA from an unknown node to two different anchor nodes must be known and thereafter the time difference can be calculated.
- Alternatively, the distance is calculated by forcing the unknown node to resend its message and thereafter by using trilateration, triangulation or MLE, it is possible to calculate the position of the unknown node.

3.7.7 Clock Skew

In WSNs, every sensor node has its own clock, and the resonant frequency of the quartz crystal in every device is slightly different. Clock skew is defined as the rate of deviation of a device clock from the true time. The frequency of a device's clock actually depends on its environment, such as the temperature and humidity, as well as the type of crystal [35].

The difference between two clocks is called the offset. Generally, the offset between two clocks gradually increases over time, and the relative speed of the offset is called the skew error. However, the frequency of the clocks may vary slightly over time because of aging or varying environmental conditions such as temperature; this fluctuation is called drift. In fact, clock skews of the same node have experimentally demonstrated the same characteristics as what is observed in general networks [35].

Clock skew varies a little from device to device and can be quantified with the unit “parts per million” (p.p.m.). This identity of nodes based on different clock skew can be used for fingerprinting nodes in WSN [35].

3.7.8 Hybrid Solutions

These are solutions that take two or more of the above solutions to present a defence against Sybil attacks. Provision for these composites is not made exclusively available by this study and consequently neither the framework nor the taxonomy has addressed this issue sufficiently. Suffice it to say that its building blocks are presented as individual countermeasures.

CHAPTER 4 RESEARCH DESIGN AND METHODOLOGY

Chapter 4 documents the structure that the research undertook. Moreover, it describes the authors thought process in solving the problem at hand. Presented is the chronological order of steps that were followed by the research i.e. the processes from examining and document solutions that were in existence, categorising them into a structured taxonomy, identifying metrics for comparison purposes, realising and implementing the frameworks (i.e. evaluation, risk analysis and best fit frameworks) and finally consolidating these results.

The research was partitioned into several phases to facilitate the achievability of the end result i.e. developing the evaluation framework and creating a best fit methodology.

4.1 ARCHITECTURAL RESEARCH

The first stage involved researching different existing techniques that either eliminate/counter Sybil attacks or have the potential to do so. The focus was on the solutions that were developed recently (within the last two years) but also considered any prominent countermeasures that were proposed earlier. A fair amount of literature not pertaining specifically to WSNs but more partial to traditional networks was also studied.

4.2 TAXONOMY DEVELOPMENT

After researching various solutions, these countermeasures were partitioned into groups depending on their fundamental operations and their aims. Instead of classifying the actual countermeasure implementation, the countermeasure type was focused on and ordered into a type of hierarchy. This was effectively based on an inheritance structure that some countermeasure classes possessed.

4.3 METRICS AND MEASUREMENT IDENTIFICATION

In this phase, a consolidated view of security was realised based on different solutions that were chosen in phase 1. Factors that affect the operation and performance of the WSN, as well as the functionality of the solution itself, were realised. Furthermore, since Sybil countermeasures are a subgroup of a bigger WSN security solution, very definite trade-offs were present in most, if not all, countermeasure designs. These considerations were also made a part of the metrics that were used to develop the evaluation framework. This

included functional and non-functional criteria, e.g. success of solution, speed, computational overhead, battery consumption, code size and data size, as well as security primitive achievability. Also, every class of countermeasures exhibited properties or functionality that was specific to its type; these were also included as metrics. Essentially, this phase's deliverable was a list of metrics that aimed to encompass all possible properties of the countermeasures that were discovered in phase 1.

4.4 EVALUATION FRAMEWORK DEFINITION

The metrics were then divided into various logical groups. Broadly speaking, this classification was overheads, security primitives, performance and functionality. Every countermeasure class had characteristics that resided in the overheads, security primitives and performance sections, but only possessed various degrees of association in the functionality section. For this reason, functionality was further broken down into general and specific subsections, i.e. general functionality and specific functionality. This logical separation also seemed appropriate when dealing with the overheads classification to create a distinction between general overheads found in traditional networks and specific overheads only inherent to WSNs.

Each of the metrics was then translated into questions to be posed to the user in order to rate how well a particular countermeasure was implemented. The now classified metrics, more accurately, the metrics from the specific functionality subsection, were associated to each countermeasure class defined in the taxonomy to form the eventual evaluation framework.

4.5 IMPLEMENTATION

The countermeasure evaluation framework, effectively a structured set of questions based on countermeasure properties, was implemented in a POC tool (which was developed as part of the master's study). Actual countermeasure instantiations of the countermeasure classes were then evaluated by the framework implementation to effectively create a library of rated countermeasures. This exercise served three purposes:

- To verify that the framework can effectively be used to rate real countermeasures
- To improve the framework after scrutinising the results from the evaluation

- To create a reference library from which the best fit analysis can be implemented.

4.6 RISK ANALYSIS FRAMEWORK DEFINITION

In order to facilitate best fit analysis, the application and the countermeasures needed to be evaluated and matched. The evaluation framework developed had been created to rate countermeasures so it was only the former (i.e. the application evaluation) that required some intervention. Using the rule that security should only be strong enough to cover the risk of the application, a risk analysis framework was created to achieve application rating. Similar to the metrics and measurement identification exercise performed above, various factors that affected the risk profile of the application with regard to Sybil attacks were acknowledged and translated into structured questions. This, together with a classification of WSN risk profiles, formed the risk analysis framework.

4.7 BEST FIT COMPUTATION METHODOLOGY

To resolve the existing challenge that designers are faced with in choosing countermeasures against Sybil attacks using consistent rules, the best fit methodology was developed. The methodology was structured in two components.

This first part of the methodology consists of a structured composition of various properties in the form of questions that the user is prompted with to facilitate gaining insight into the application's security requirements. These properties were classified under the already existing headings, i.e. overheads, security primitives, performance and functionality.

The second part of the methodology maps each question to metrics that were used to evaluate the effectiveness and efficiency of countermeasures in the evaluation framework. These mappings were not simply a one-to-one mapping and furthermore had both positive and negative relationships with metrics in the evaluation framework. The set of questions posed to the user is dependent on the risk profile of the application. A best fit matrix was used to achieve the mapping and after scoring each answer (for the question in the first part of the methodology), each countermeasure is rated for its suitability to the applications requirements.

4.8 FRAMEWORKS CONSOLIDATION

This stage allowed the framework definitions realised in Phase 4 and Phase 6 (i.e. the countermeasure evaluation framework and the risk analysis framework) to be refined and to assign more accurate weightings for the different criteria based on WSN design goals. Important features identified during implementation were added to the framework. Also, the risk analysis and best fit computation were included in the POC tool.

4.9 RESULTS AND CONCLUSIONS

Finally, the results from the frameworks were documented and presented. Valuable deductions about countermeasure selections were also offered. Recommendations that can be used to improve on existing or new countermeasures against Sybil attacks were also identified.

CHAPTER 5 ASSUMPTIONS AND CONSIDERATIONS

This chapter outlines the factors and conditions (i.e. in terms of WSN operations) that were considered effectively defining inclusions and exclusions. This was done in order to facilitate the reader to either validate or reproduce any findings achieved by this research.

Upon deliberation on the high diversity of applications and types of countermeasures available, there were some assumptions that had to be made for the research and development that were undertaken. Furthermore, only certain aspects of WSN operation were taken into consideration. An account of these assumptions and considerations are listed below:

It is assumed that:

- An entity is an actual hardware device. This does not change if the system is under a Sybil attack.
- An identity is an abstract representation that persists across multiple communication events [15]. It is merely a name/number/reference bound to an entity.
- Under normal operation, there is one-to-one mapping of identities to entities.
- In the event of a Sybil attack, the number of entities (E) is less than the number of identities (I). However, the definition used for this research was defined somewhat more accurately. If $I \geq E + X$, then the system is under attack, where X is a number (usually a percentage of the total number of nodes in the network) defined by the user. This effectively allows the user to determine the number of allowable compromised nodes.
- Detection rate is defined as

$$DR = \frac{(\text{number of Sybil nodes detected - number of nodes incorrectly detected})}{\text{Actual number of Sybil nodes}} \text{x}100\% \tag{5.1}$$

  on condition that this number cannot be more than 100%. Detection rate is sometimes referred to as completeness.

- False negative is defined as

$$FN = \frac{(\text{number of non-Sybil nodes incorrectly detected as Sybil nodes})}{(\text{number of Sybil nodes detected})} \text{x}100\% \quad . \tag{5.2}$$

- False positive is defined as

$$FP = \frac{(\text{number of Sybil nodes incorrectly detected as non-Sybil nodes})}{(\text{number of Sybil nodes detected})} x100\% \quad . \tag{5.3}$$

- False positives and false negatives are sometimes collectively referred to as detection accuracy.

- All cryptographic laws hold true and basic assumptions about the performance of security algorithms and mechanisms are assumed to be valid, e.g. an attacker cannot determine a key or create a chosen hash result through brute force.

- Attackers have an unlimited amount of resources at their disposal in terms of storage and computation. However, with regard to communication, a single hardware device can only transmit or receive data on one channel at a time.

- The base station and/or cluster head is a more powerful node in terms of storage, transmission/reception capabilities and energy.

- The base station and/or cluster head is considerably more secure, from a physical viewpoint.

- A sensor node may be able to perform a limited number of public key cryptographic operations e.g. ECC. However, a node cannot sustain too many of these cryptographic operations as a result of the intensive computation and energy consumption.

- A Sybil attack can occur during bootstrapping and/or during normal operation (i.e. after bootstrapping).

- No countermeasure is 100% secure.

- Any countermeasure can be improved upon.

- Security breaches are inevitable irrespective of what countermeasure or security features are employed.

- Security must continuously be monitored and managed and cannot be a once-off implementation. Effectively this requires countermeasures to be adaptive.

The scenarios that were taken into consideration were that:

  - Nodes in a WSN are stationary.

- Nodes can be captured and their contents read by an attacker. Whether this content is interpreted depends on the data being encrypted as well as the effectiveness of the encryption.

- WSNs can have homogenous or heterogeneous topologies.

- Note that WSNs with just one base station and no cluster heads are regarded as a homogenous topology.

CHAPTER 6 A FORMAL MODEL

Chapter 6 formally describes the models that were utilised as building blocks to develop the final framework. Also illustrated is the manner in which these artefacts interact with each other. 3 models were defined i.e. a WSN model, a security model and an attack model.

To gain better insight into the issues that could affect the WSN applications and its operational processes, countermeasure operations and the attacker's aims and methods, various formal models were developed. Accomplishing this also assisted in defining the boundaries between operations, security and threats. Furthermore, the integration of these models allowed the study to close gaps and possible vulnerabilities that countermeasures ought to have addressed. Figure 6.1 shows the integration of the models and is indicative of how the system operates.

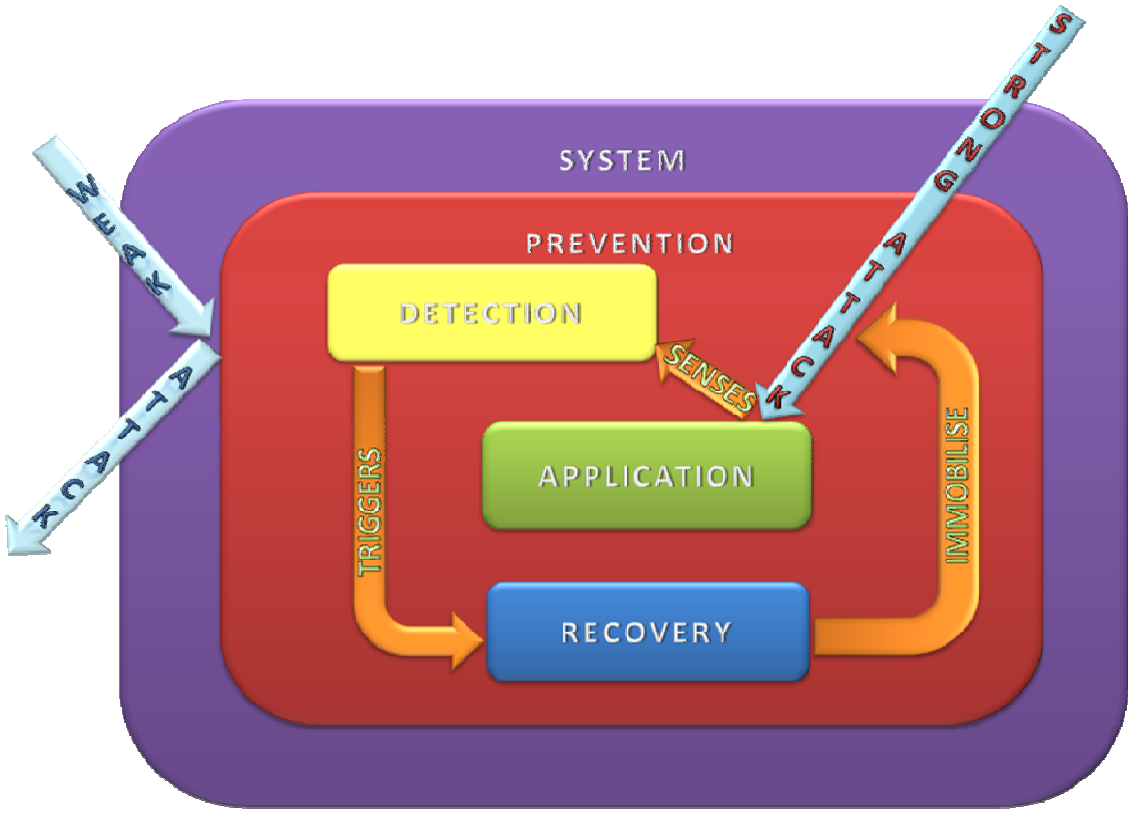

Figure 6.1: System Model

6.1 WSN MODEL

The WSN model consists of the hardware model and the system model. The hardware model is adapted from [15] and the system model is adapted from [1].

The hardware model consists of a communication cloud (i.e. a wireless medium), entities (nodes) and identities that can be either true or false. These entities communicate via messages (i.e. a string of bits that is transmitted over the wireless medium).

The system model defines the way software is loaded onto the nodes, is deployed into the target area and begins its application operations. It should be noted that the application is referred to as just the routines that run on the nodes, including routing protocols but excluding security. The system, as can be seen in Figure 6.1, encompasses both the application and the security (i.e. the security model) but excludes attacks (i.e. the attack model).

Regarding the application, it is assumed that the software is preloaded onto the nodes before deployment. After being deployed into the target area, nodes find their neighbours in a manner dependent on their routing protocol. Thereafter, the nodes begin operating as intended i.e. sensing their environmental conditions and reporting back to either their cluster head or the base station.

6.2 SECURITY MODEL

The security model is defined as having three different and mutually exclusive components. These are Prevention, Detection and Recovery. This model has been adapted from traditional network theory where there are preventative methods and intrusion detection systems (IDS) [18].

Prevention is the mechanism that protects the system from being attacked by Sybil attacks. It is in operation irrespective of whether there is a breach but more importantly is unaware of an attack that breaches the protection barrier it offers. Prevention methods are, crudely speaking, attack blocking methods. Against Sybil attacks, these mechanisms attempt to prevent entities from successfully presenting false identities to the system.

Detection is the mechanism that checks the validity of nodes by monitoring behaviour as well as the messages sent by the respective nodes. In the case of Sybil attack detection, the

mechanism monitors if an entity (i.e. a node) presents only one valid identity throughout its lifetime (direct Sybil attacks) or checks if an entity pretends to be in contact with identities that are in reality false (indirect Sybil attacks). Once a direct or indirect attack is perceived, the detection mechanism triggers off a recovery mechanism if available.

Recovery is the mechanism that sustains network connectivity and application operations during and after an attack. Pertaining to Sybil attack scenarios, recovery mechanisms can tell all true member nodes to ignore the identified false nodes and find other nodes to support the application. Recovery mechanisms are an add-on to detection techniques and cannot exist on their own.

6.3 ATTACK MODEL

According to [7], there are three orthogonal factors that influence the manner in which Sybil attacks are launched. These are direct communication attacks, indirect communication attacks, simultaneous attacks, non-simultaneous attacks, fabricated identities and stolen identities (*See Figure 6.2*).

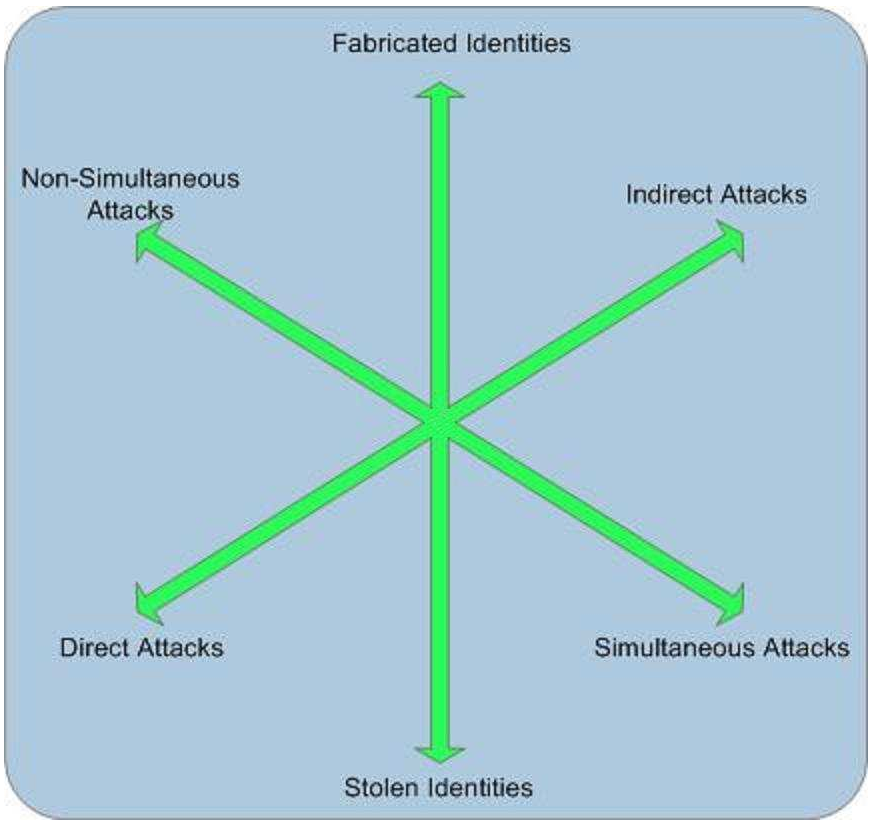

Figure 6.2: Orthogonal Dimensions for Sybil Attacks

The study has included a fourth orthogonal dimension, time of attack. This effectively relates to whether the attack happens during or after bootstrapping. The attack model is defined as the following:

The attacker can launch either external or internal Sybil attacks. The attacker may even launch a combination of these types of attacks. For internal attacks, the adversary has physically compromised one or more valid nodes in the network and understood its contents. For external attacks, the adversary may or may not control actual nodes in the network, however, no true nodes are controlled i.e. the attacker has placed his/her own malicious nodes into the deployment area of the original network.

The attacker may attempt to snoop for confidential (or crucial) information, insert bogus messages or re-transmit previously overheard messages to allow Sybil attacks to be launched. The attacker can compromise a few nodes and use them collectively to launch Sybil attacks against the rest of the network. The attacker can allow the compromised nodes to operate normally so as to seem genuine before launching an attack. However, the compromised nodes can also be commanded not to cooperate with others and exploit weaknesses of various protocols.

The adversary has the ability to use computationally powerful devices as nodes that, for all intensive purposes of the model, have unlimited energy supplies when compared to normal WSN nodes e.g. laptops, handheld devices. Sybil attacks will have characteristics for the properties of the four dimensions mentioned earlier.

CHAPTER 7 METRICS FOR COMPARISON

The structure, together with the details of the comparative factors, is documented in this chapter. The details for the Specific Functionality are covered in Chapter 9. This separation is primarily due to the framework evaluating these metrics for every countermeasure but would have different criteria to consider for the Specific Functionality depending on the position of a countermeasure in the taxonomy.

The factors that were used to rate each countermeasure were grouped into categories i.e. General Overhead, Specific Overheads, Security Primitives, Performance, General Functionality and Specific Functionality. Within these categories various questions had to be answered. In general, the first five categories remain constant for any countermeasure but the specific functionality scores are based on factors that are unique to the type of countermeasure. Below is a table of metrics that were used. Note that the test environment should contain the same number of nodes for this evaluation or at least the same constraints in its topology class i.e. the homogeneous or heterogeneous topology class.

Table 7.1: High-level Categories of the Framework

Framework Categories	Descriptions
General Overheads	This category considers how lightweight the countermeasure is. These constraints must also pertain to traditional networks.
Specific Overheads	This section considers the other overheads that are specific to WSNs that are introduced by the countermeasure.
Security Primitives	Authentication, Integrity, Availability, Data Confidentiality – Here, these factors are evaluated based on whether this is achieved and if so, how well.
Performance	These are typically the non-functional specifications of the countermeasure.
General Functionality	This is the functionality that is common to all countermeasures.
Specific Functionality	This functionality is unique to the type of countermeasure (will be

	covered in a later chapter)

It should be noted here, that this table is effectively the high-level detail of the framework. More detail for the Specific Functionality "block" will be covered in a later chapter.

7.1 GENERAL OVERHEADS

These metrics are factors that are negatively influenced in traditional networks/applications when security is introduced. It seems obvious but it is worthwhile to note that any application would have a finite value for any of these factors even in the absence of security. Herein lies the method to abstract the values that a countermeasure would introduce. Usually, to obtain accurate results, one would simulate an application without any security, measure these factors, then introduce a countermeasure and once again measure the same factors. The difference would yield the results that the framework requires.

- Latency – The extra delay that the countermeasure introduces, measured in milliseconds (ms). This value is measured for just one node or averaged for a few nodes.

- Energy Cost – The extra battery power that is consumed, measured in milli-Joules (mJ).

- Throughput – An important concern for WSNs is the throughput. Essentially, this is a measure of how much of the communication needs to be transmitted and received with the introduction of a countermeasure. This can be measured by monitoring a sink node and noting the number of incoming messages when the sink starts becoming a bottleneck (i.e. starts to drop messages because its buffer is full). This is measured in bits per second (bps).

- Code Size – The actual lines of code introduced by the countermeasure is a significant factor, considering that most WSN nodes are manufactured with usually no more than 2 Megabytes of code space. This value is measured in kilobytes (kB)

- Data Size – In a traditional device, this measure would translate into the amount of random access memory that a solution requires. Often, there is only around 512kB

of data space of which a significant portion is used up by the operating system of the node. This value is measure in bytes (B)

7.2 SPECIFIC OVERHEADS

Requirements that Sybil countermeasures can possibly demand in order to be successful are listed here. As a general rule, the more stringent the requirements, the less appealing this countermeasure would be and this would be evident from the rating that this countermeasure would score in this section. These are the questions that the countermeasure is required to answer:

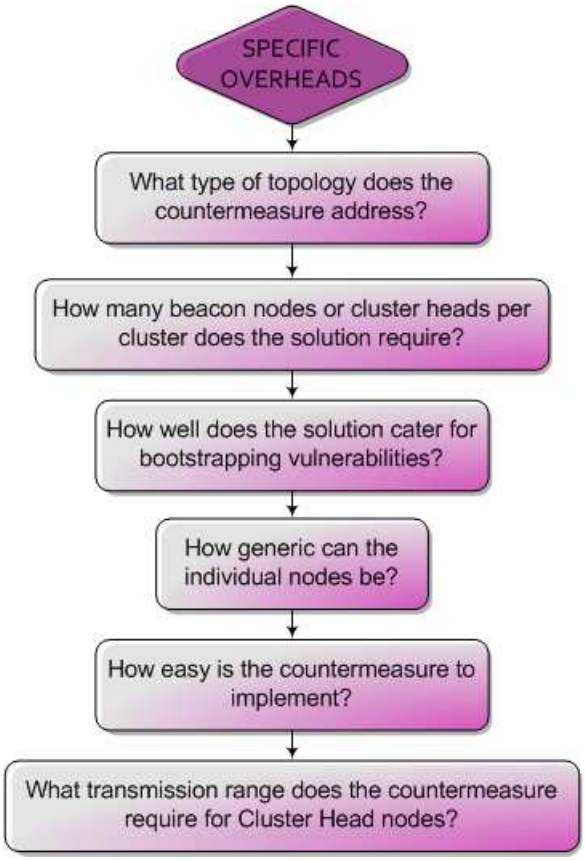

Figure 7.1: Specific Overheads Metrics

7.3 SECURITY PRIMITIVES

These are the traditional security primitives that countermeasures ought to address, namely authentication, authorisation, data confidentiality, network availability and data integrity.

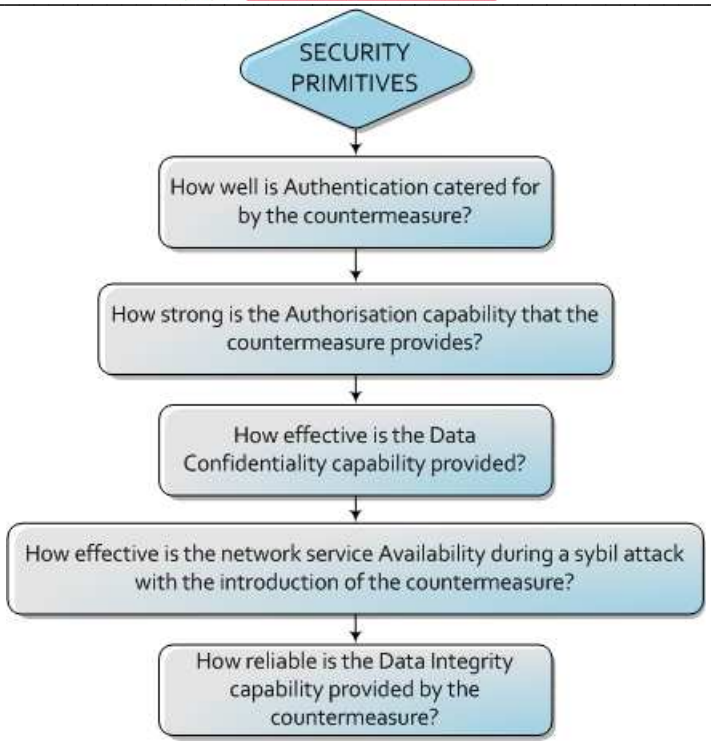

Figure 7.2: Security Primitives Metrics

7.4 PERFORMANCE

Speed difference of application, i.e. before vs. after implementing a countermeasure, has been taken into consideration at the overheads section. Here the typical non-functional specifications are rated. Factors such as reliability, scalability, flexibility, compatibility and resilience to attacks are evaluated here.

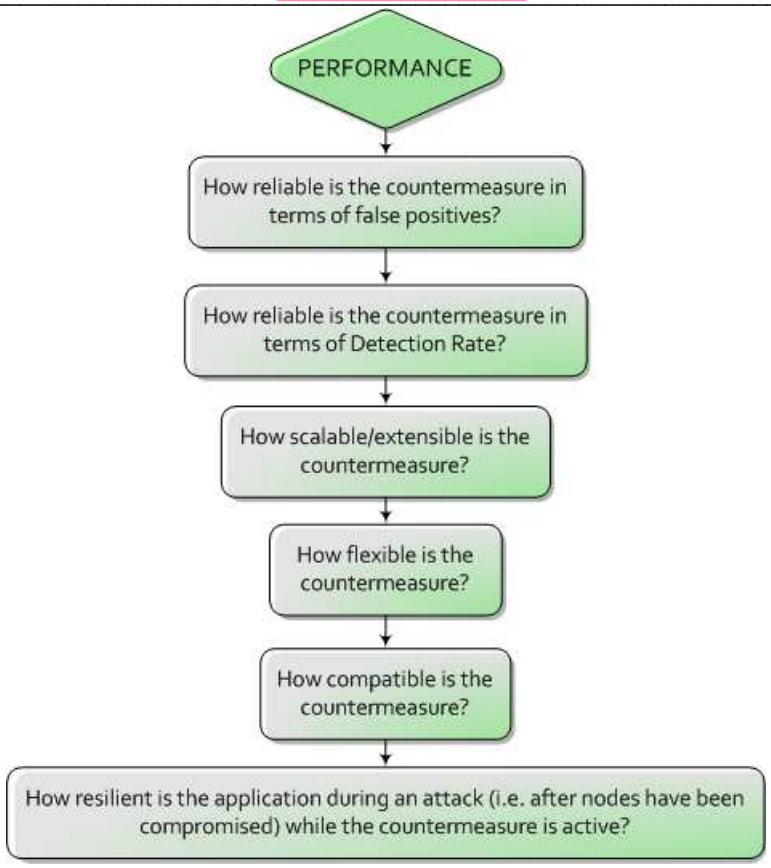

Figure 7.3: Performance Metrics

7.5 GENERAL FUNCTIONALITY

The framework evaluates how the countermeasure addresses Sybil attack characteristics that were earlier defined in the attack model, i.e. the four orthogonal dimensions of the Sybil attack. Factors such as logging of attacks, graceful degradation and fail-secure techniques are also evaluated here since all countermeasures should cater for these.

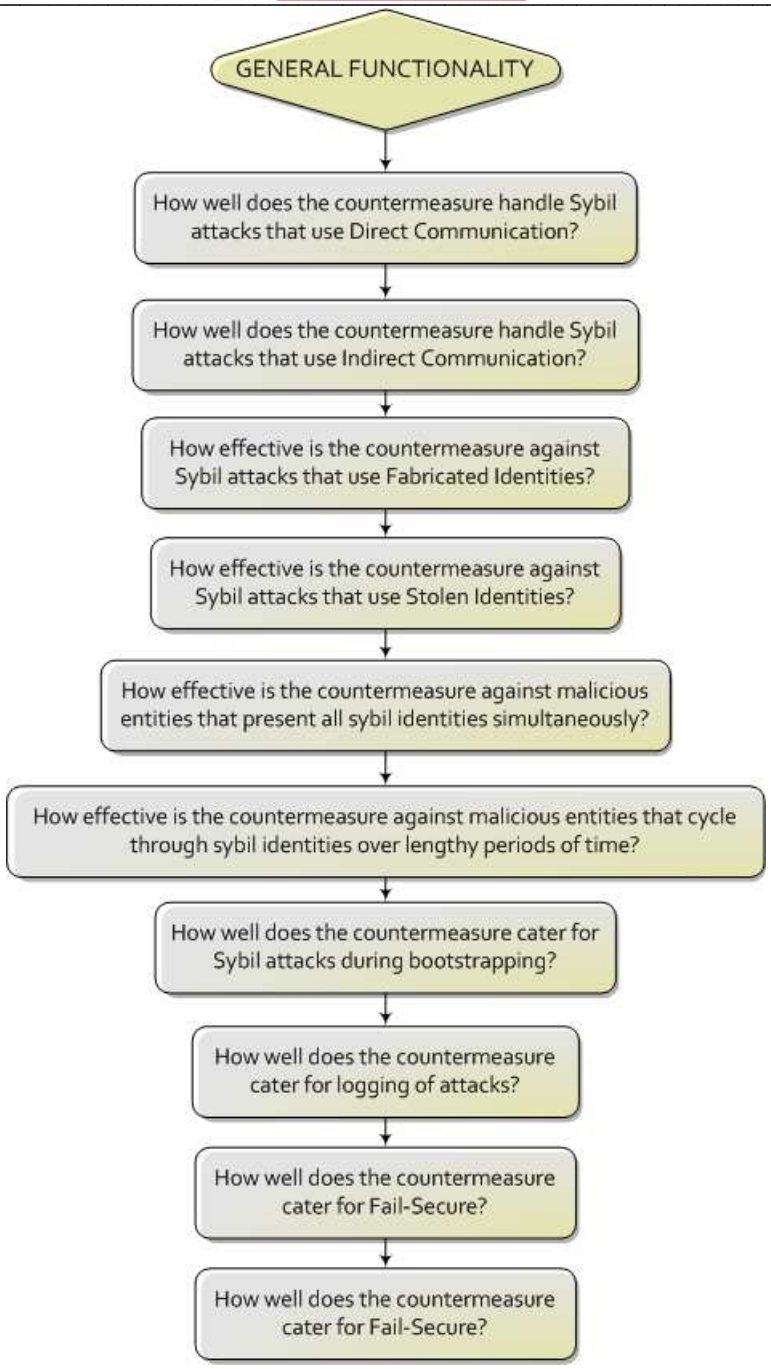

Figure 7.4: General Functionality Metrics

7.6 SPECIFIC FUNCTIONALITY

These metrics are unique to the type of countermeasure and will be covered in a later chapter.

CHAPTER 8 THE TAXONOMY FOR SYBIL COUNTERMEASURES

This chapter gives insight into the crux of the research. In effect, the entire framework can be abstracted into the taxonomy presented in this chapter. A high level explanation of the hierarchical structure is offered which also includes some ground rules on taxonomy extensibility.

To facilitate defining the group(s) of questions that can be used to rate countermeasures and somewhat more importantly to group countermeasures that can be queried by the same questions with the intention of grading, the taxonomy for Sybil countermeasures was developed. It is not implied by any means that this is the total set of countermeasures that exists nor that the type of classification used is perfect but more importantly, it was the intention of the author to achieve a reference model on which the framework could be based.

8.1 A BRIEF EXPLANATION OF THE TAXONOMY

A fairly comprehensive collection of countermeasures were classified according to what function they perform in a typical WSN i.e. Prevention – the preventative measures that are used to make it harder for Sybil attacks (denoted by red blocks), Detection – the techniques employed to check for Sybil nodes/attacks (depicted in yellow blocks) and Recovery – the actions taken by the countermeasure/system in the event that a Sybil attack is detected (shown in blue).

The green blocks are those methods that have been identified but have not been classified because insufficient literature was available at the time of the study to develop metrics for these types of countermeasures. However, these countermeasure types were placed to show that:

- These methods exist and are not known to be part of the existing taxonomy
- These methods have been considered.

The structure of the taxonomy is clearly hierarchical and the intention was to be able to trace a single, unique path to the uppermost level (i.e. the purple block). The implication of

this is the elimination of ambiguity in upward and downward traces from pure child nodes to the highest level block and vice versa.

The taxonomy is designed such that actual countermeasures may only be instantiations of pure children i.e. blocks that don't have arrows leaving them; in other words they are not a parent of another block. For instance, one would not find an instantiation of symmetric cryptography but would find an instantiation for current encryption algorithms, e.g. AES. If there is a countermeasure that cannot be placed in one of the children leaf nodes of the taxonomy, it would imply that the taxonomy needs to be extended.

In the event that the taxonomy needs to be extended, it should obey the design rules of the existing taxonomy for reasons of reusability. These rules are:

- Instantiations can only be of leaf nodes.
- New leaf nodes may not be repeated under any circumstance.
- New leaf nodes should fall into only one category i.e. prevention, detection or recovery. Hybrids have not been catered for by the framework for this study and it is recommended that either the predominant part of the countermeasure be classified or the countermeasure be split into various components and be evaluated.
- Placing a new leaf node must make sense with regard to all its parents right to the uppermost level (i.e. the purple block)

One should also take note that the complete list of questions for any instantiation can be found if one traces a path from the leaf node (i.e. the instantiation block or the respective pure child) up towards the final parent (i.e. the purple block).

Lastly, comparisons of countermeasures may only be done at the level of a common parent or above. For instance, a "digital signature" instantiation cannot be compared with a "phased-out encryption algorithm" instantiation on the level they exist in the taxonomy; instead comparisons can be made at a "cryptographic technique" level. This comparison would effectively rate how well these instantiations perform as cryptographic techniques under Sybil attacks. The framework rates all countermeasures at the highest level block, i.e. how well this countermeasure performs on an overall basis.

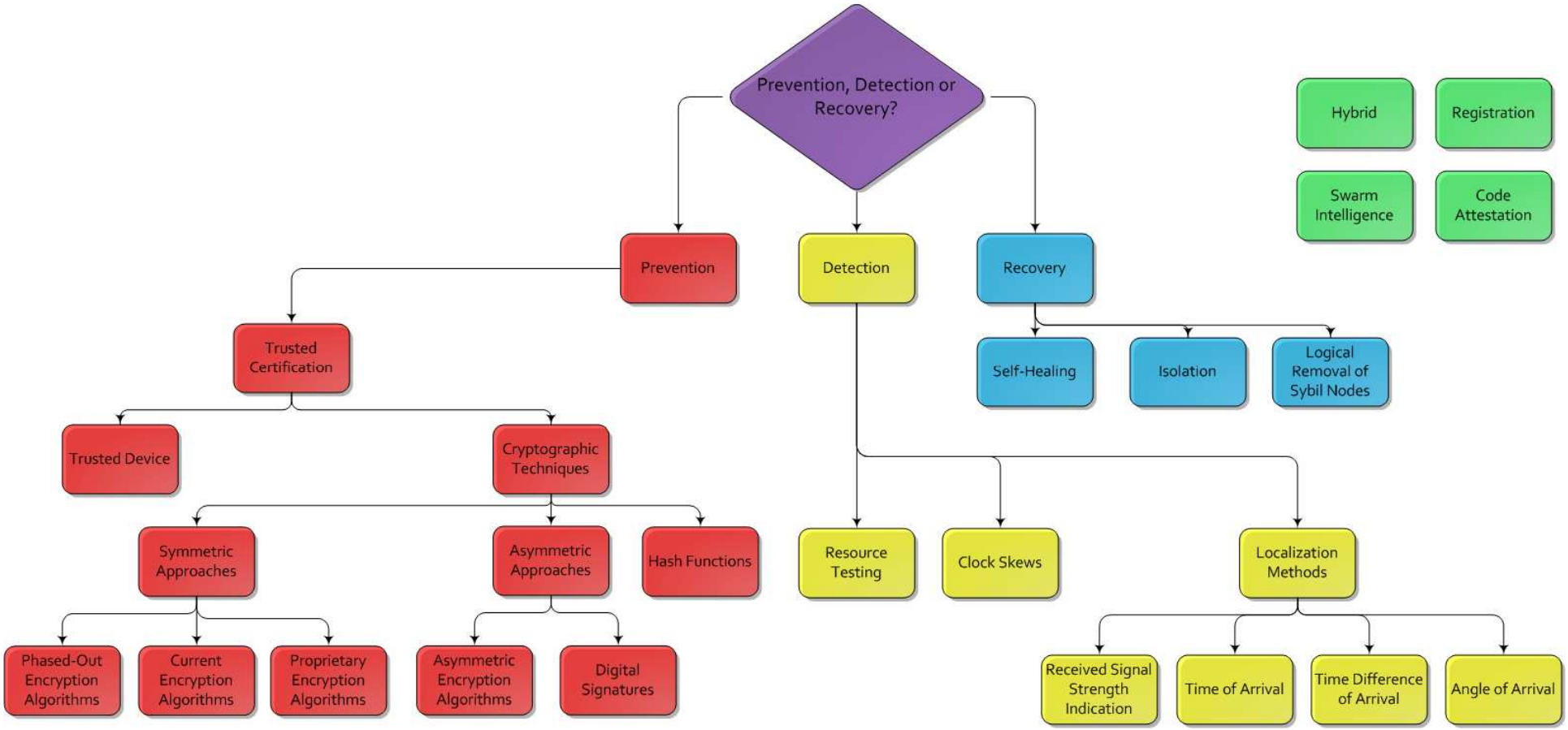

Figure 8.1: The Taxonomy of Sybil Countermeasures

CHAPTER 9 THE FRAMEWORK

This chapter contains the details of the Specific Functionality for countermeasures. It presents information in the same order as described by the taxonomy i.e. Prevention, Detection and Recovery. In essence, the details under each block of the taxonomy are elaborated on here.

As mentioned previously, the General Overhead, Specific Overhead, Security Primitives, Performance and General Functionality components are present for all types of countermeasures and this forms the initial part of the framework. To complete the framework, each countermeasure, depending on its relative position on the taxonomy, has specific factors that need to be queried. These criteria are covered in Specific Functionality.

Again, it should be noted that the complete set of factors that would be queried for a particular countermeasure can be traced on the taxonomy e.g. the complete set of questions that an Elliptic Curve Cryptography countermeasure would be asked would include the criteria for Prevention, Trusted Certification, Cryptographic Techniques, Asymmetric Approaches and Elliptic Curve Cryptography.

The framework is the consolidation of all these criteria as well the metrics defined earlier. The framework referred to in this chapter is the countermeasure evaluation framework and should not be confused with the risk analysis framework, described in a later chapter.

The individual blocks of specific functionality and their explanations thereof are given below. The framework is effectively a set of questions targeted at a specific countermeasure type, depending on its location on the taxonomy. Where possible, some solutions that are options to be built into instantiations of the countermeasure type are also given in the following discussions.

It should be noted that some questions relate to the implementation of the countermeasure classes that are contained in the framework while others relate to the countermeasure type.

9.1 PREVENTION

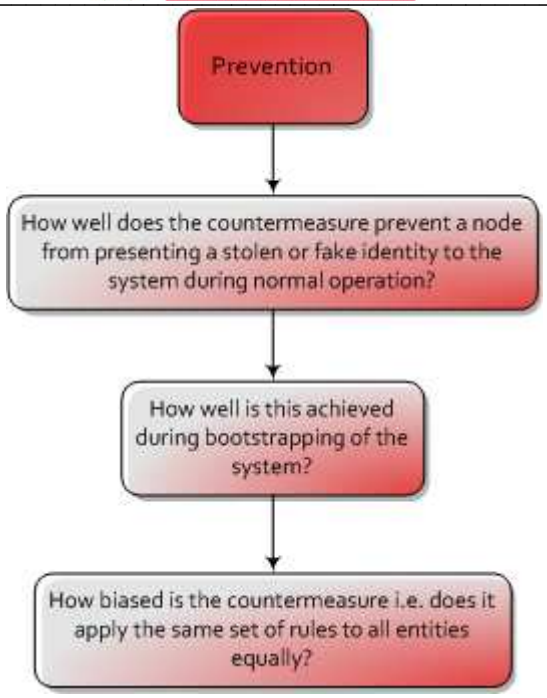

Figure 9.1: Prevention Specific Functionality

As explained earlier, prevention is the measure(s) that is employed to make it difficult for attackers to succeed when attempting to launch a Sybil attack. The system is interested in how well the countermeasure defends it against the possibility of an entity presenting wrong identities. It should be noted that if a preventative countermeasure had a 100% success rate, there would be no need for detection or recovery techniques.

The countermeasure should operate both during bootstrapping and normal operation. As stated earlier, bootstrapping is the process where nodes find their neighbours and "get accustomed" to their surroundings. This would be quite an opportunistic time for a malicious attacker to exploit the vulnerability of innocent nodes.

Furthermore, the technique should not favour a selection of nodes. Effectively, the suspicion factor should remain constant for all types of communication i.e. either the countermeasure either suspects that all nodes could present Sybil identities for a part of the communication or that all nodes are honest for another part of the communication. A case in point would be that when any node wants to communicate with a CH, the countermeasure should verify its authenticity irrespective of rank but after a cryptographic handshake has been completed successfully, the countermeasure may relax its scrutiny of this node for the remainder of that cryptographic session.

9.2 TRUSTED CERTIFICATION

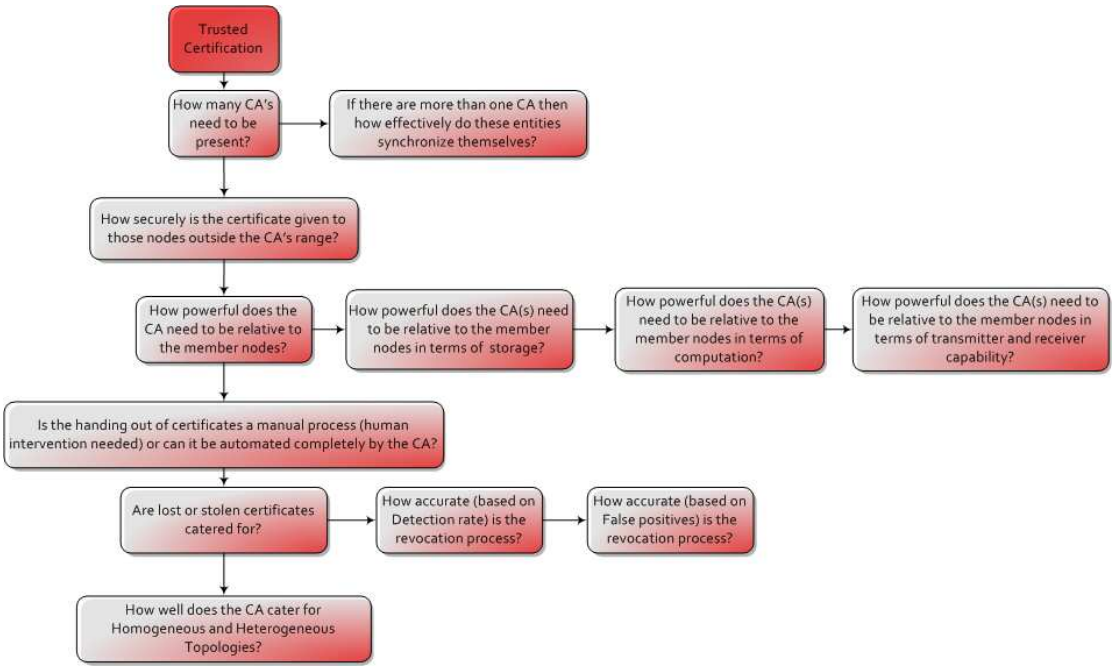

Figure 9.2: Trusted Certification Specific Functionality

For Trusted Certification to be successful, it is obvious that every node would need to be certified by a CA. Although, it is quite possible that the countermeasure requires more than one CA for the following reasons:

- To cater for redundancy
- To assist in larger networks
- To fulfil the requirement that certificates be handed out directly by a CA and not by another node
- To optimise cluster based networks

As much as these are positives, from the standpoint of the requirements the framework views more CAs as a negative. The rationale being that the choice of having more than one CA should be a network design consideration and not a stipulation of a countermeasure. As such, the developer or designer should not be limited by this constraint.

However, one is then faced with the challenge of synchronising all CAs. If not addressed properly, valid nodes that have been certified or registered on one CA may not be able to

communicate with nodes belonging to another CA's jurisdiction. This would be detrimental to network connectivity and indirectly scalability of the network.

CAs can grant nodes certificates within their transmission range or outside their range. This is achieved by using either single-hop or multi-hop techniques. When using multi-hop techniques, the CA is effectively delegating some of its responsibility to the intermediate nodes in the hope that all nodes relay the exact message to the intended recipient. If this matter is not handled securely and verifiably, the countermeasure leaves the system vulnerable to a range of attacks. One of these attacks can even be a Sybil attack, implying that the countermeasure introduces vulnerability that it was intended to eliminate.

CAs are generally more powerful in terms of communication, computation and storage compared to member nodes. Once again though, the requirement that a very powerful device is needed for the successful implementation of a particular countermeasure is a disadvantage rather than an advantage.

One of two methods can be employed for the complete creation, distribution, updating and revocation of certificates (i.e. identities, secret information or a combination thereof) that are required by nodes to communicate securely. This can either be completely automated or some components of certificate management may require human intervention. The framework rates the need for human intervention lower than a fully automated countermeasure.

Certificates, even when dynamically created using timeout techniques, can be stolen by malicious intervention or simply be lost when a node leaves the network due to its battery being depleted or it moves to another cluster that is out of range. If this possibility is not considered by a countermeasure, unnecessary storage for more certificates or worse, the continuation of communication with a Sybil node could be a very real possibility. The need for effective certificate revocation must be satisfied here. Moreover, the revocation should be accurate in terms of false negatives and false positives.

As noted earlier, multiple CAs can be introduced to assist in cluster based topologies. However, irrespective of the number of CAs, the countermeasure should still optimise its operations to take advantage of the pros of the topology that it is being utilised in.

9.3 TRUSTED DEVICE

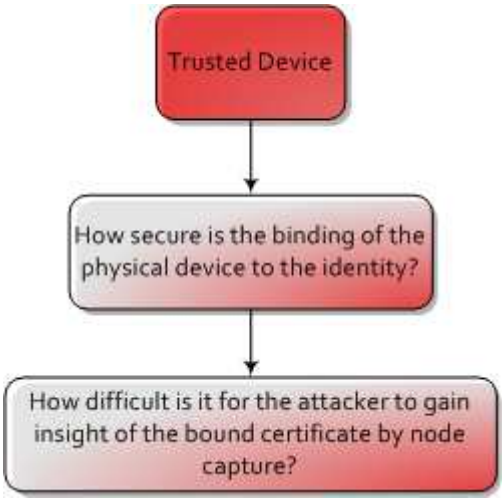

Figure 9.3: Trusted Device Specific Functionality

The binding of secret information to a specific device has to be cryptographically secure, especially if done via a wireless medium during normal operation (i.e. after bootstrapping). Also, the countermeasure (through its CA) should ensure at all times that there is one-to-one mapping of bounded secrets to devices.

In the event of an attacker being able to tamper physically with the node and read its contents, the binding methods that were used should be strong enough to guarantee that the secret information is kept secret.

9.4 CRYPTOGRAPHIC TECHNIQUES

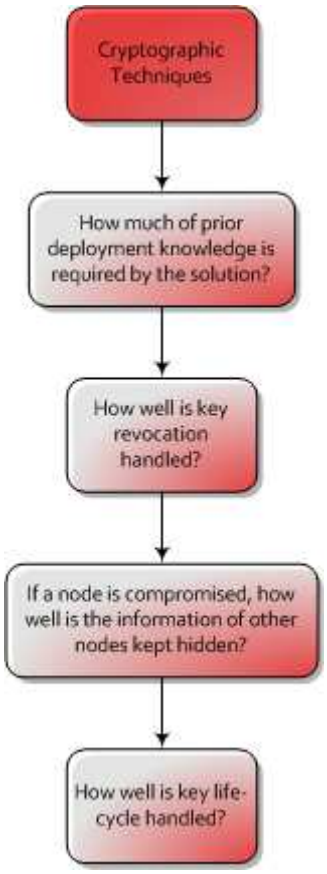

Figure 9.4: Cryptographic Techniques Specific Functionality

In order for cryptography to be successful, it is essential that operations regarding keys are of a high standard. Some of these operations are:

- Key creation
- Key management (for the purposes of this document, key management includes key distribution and key revocation but excludes key creation).

For key distribution, some cryptographic approaches use pre-distribution techniques and to this end, some countermeasures optimise their pre-distribution based on specific deployment knowledge. Despite the benefits that can be derived from employing this technique, the framework sees the stipulation of requiring deployment knowledge in a poor light. It must be said, though, that if a countermeasure can draw on the benefits of deployment knowledge when it exists, the framework would not penalise this behaviour.

Much as trusted certification is able to handle certificate revocation, it is a requirement from any cryptographic technique to be able to handle key revocation effectively and accurately. Severe vulnerabilities can be exposed and exploited if key revocation is poorly handled.

Cryptographic techniques in WSNs are less impregnable than in traditional networks for two simple reasons; firstly they operate almost completely without any human intervention, thus making it a necessity that secret information must be stored on the node itself (whereas in traditional networks, this information could be an input from a user) and secondly they are quite often deployed in hostile environments where their secret information can be disclosed if a node is compromised and its memory read. Well-designed cryptographic countermeasures will provide assurance that if a node or key is compromised, the security of the entire network is not breached. At the very least, information on other nodes should not be divulged to the compromised node, although a superior method would be to disallow communication with this node.

To further curb the possibility that keys that have been stolen or lost are not known to the system (i.e. the detection techniques have missed), keys should have a finite lifetime. In addition, the life-cycle of keys should be managed well and be suitable for the application's needs. The implication is that the lifetimes of keys should be a variable that can be optimised.

9.5 SYMMETRIC APPROACHES

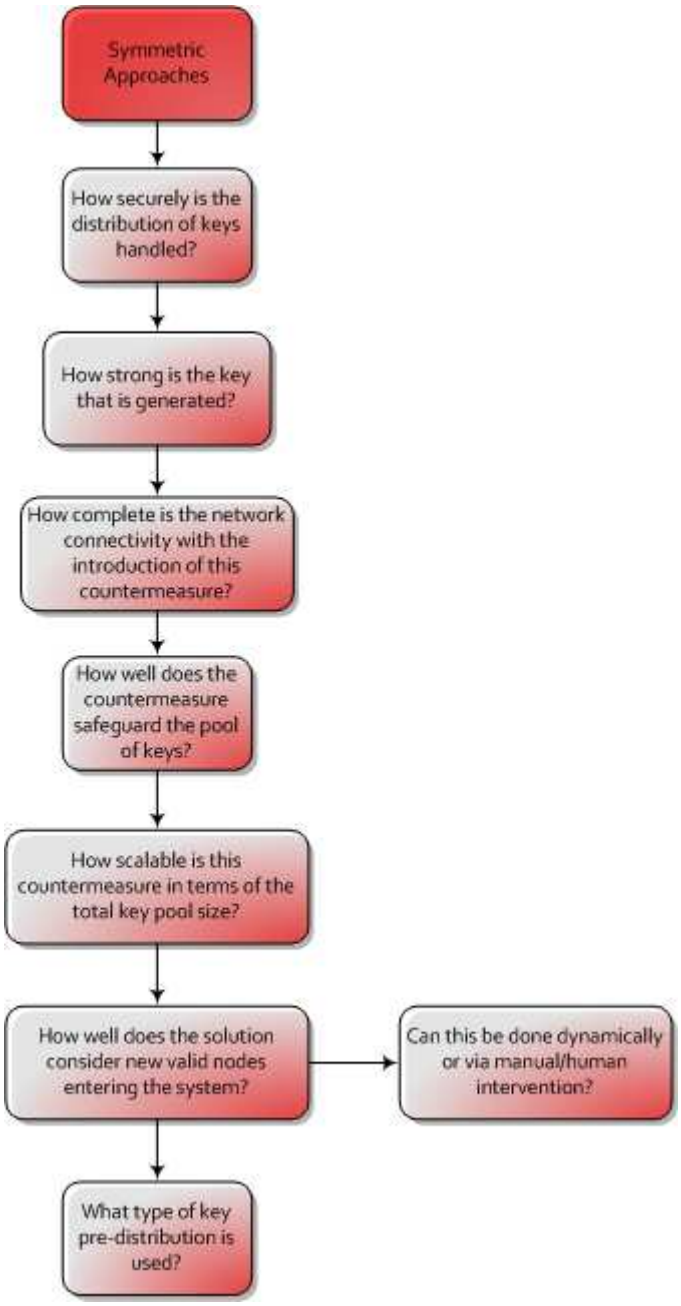

Figure 9.5: Symmetric Approaches Specific Functionality

Shared key procedures are quite often more focused on the distribution of keys rather than the symmetric encryption algorithm applied. To this end, it is quite difficult to find papers that are focused on symmetric algorithms in WSNs. Taking into account that symmetric algorithms have only secret keys, which need to be shared between nodes, it is of the utmost importance that this information be very securely communicated when necessary. Failing to do so would render the system cryptographically insecure.

Although not much focus is placed on the actual algorithm used in symmetric cryptography for WSNs, the framework still uses the strength of the generated key as a metric to rate countermeasures. The stronger the key, the more secure the countermeasure is. The trade-off is that it would take longer to encrypt and decrypt messages with stronger keys and more storage would be required for longer keys. A fine balance between the strength of keys and hardware capabilities according to the application requirements will have to be reached. In this block, however, the framework is more concerned with keys being short and the trade-off is handled in the overheads section.

When using shared key technology, there are many instances of nodes being able to communicate with each other, if and only if they share the same key either with each other or with a common third node. This is in spite of the fact that they could very well be within transmission range of each other. This unwanted side-effect introduces a problem of limiting the network connectivity and is in direct violation of a primary goal of distributed networks.

The idea behind symmetric approaches is that keys are distributed to nodes that want to communicate with each other. The CA has a finite number of keys at its disposal and allocates one or more keys to these nodes with the intention that they should have a shared secret between them. It is absolutely vital that this pool of keys be safeguarded well, otherwise the entire security of the application can be affected, especially when the keys that have been compromised are unknown.

Since the pool of keys is a finite set, having keys that are too short can pose a problem with regard to scalability. A case in point would be that if a CA generates a certain length of keys, the size of the network would be limited to the number of combinations that this particular length of key can have. On the other hand, having very long keys will place a strain on the individual nodes. Once again, a balance according to the application requirements would have to be attained.

In many symmetric schemes, shared keys are distributed to nodes prior to deployment or even during bootstrapping. Quite often, addition of new valid nodes to the system is a reality for various reasons. For the sake of scalability, provision for this scenario has to be adequately dealt with. Furthermore, it is preferable that the countermeasure caters for this without the need for human intervention, irrespective of whether the application can handle automatic additions or not.

As mentioned earlier, symmetric cryptographic schemes require keys to be distributed to nodes before they can communicate securely. Various options have been analysed. The sections below rate these pre-distribution schemes. Note that these functional blocks start off with a red diamond rather than a red rectangle. The intention is to distinguish these sections as an extension of the "Symmetric Approaches" block as opposed to being a child of symmetric approaches on the taxonomy. To this end, one would not find these blocks represented on the taxonomy.

9.5.1 Single Network-wide Key

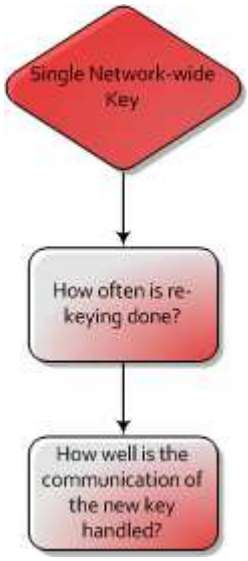

Figure 9.6: Single Network-wide Key Specific Functionality

Considering the downfall that a single network-wide key scheme has (i.e. if a node is compromised, the entire network is completely vulnerable to attacks), the countermeasure can be rendered somewhat more secure if the key is valid for only a certain period.

When a new key is generated by the CA, all nodes would then require this key to communicate. Communication of this new key must be done in a very secure manner and must only be sent to valid nodes to prevent man-in-the-middle attacks. It should be noted that successful man-in-the-middle attacks can enable malicious entities to launch Sybil attacks.

9.5.2 Pairwise Key Establishment Schemes

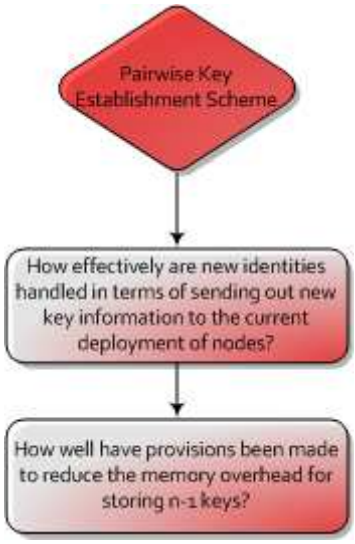

Figure 9.7: Pairwise Key Establishment Scheme Specific Functionality

In pairwise key establishment schemes, whenever a new node enters the network, its unique identity will have to be circulated among the existing infrastructure of nodes. Much like the single network-wide scheme, this dissemination must be done securely in order to prevent man-in-the-middle attacks and subsequently Sybil attacks. In addition, if not carried out effectively (i.e. if all nodes are not presented with the new identity), network connectivity would suffer.

The pairwise key establishment scheme requires that n-1 keys be stored in each node's memory. It is quite possible that network sizes for WSNs could reach far into the thousands if not tens of thousands. These large network sizes imply huge memory allocation for keys, something not physically possible with memory-limited WSN nodes. This scheme may work better in a clustered environment or with small homogeneous networks, but the developer should still consider modifying the scheme to adapt to hardware constraints of WSN nodes.

9.5.3 Trusted Base Station

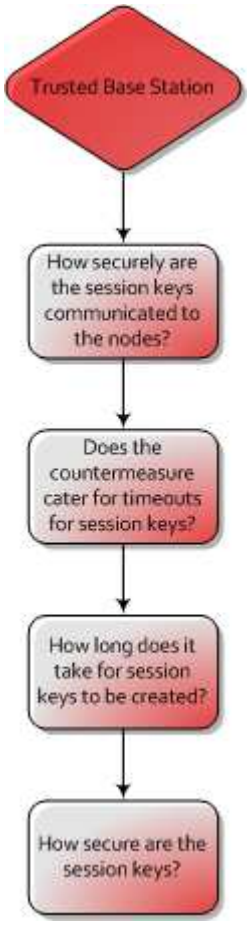

Figure 9.8: Trusted Base Station Specific Functionality

Session keys that are generated and supplied to nodes wanting to communicate should be dealt with securely so as to prevent possible malicious nodes within transmission range from launching attacks with this information. Not unlike the pairwise key establishment scheme, in the absence of secure dissemination of keys, man-in-the-middle and Sybil attacks are a very real possibility.

Because this is a technique where session keys are used, it is imperative that time optimisation concerning key validity be calculated, since it is not always enough to allow two or more nodes to communicate indefinitely using a particular session key; hence the requirement for having timeouts on session keys. Having too short legitimate times for session keys reduces the possibility of attacks but effectively introduces unnecessary overhead since the CA(s) need to circulate keys often, besides the balance that ought to be achieved regarding timeout values.

As much as the system requires a secure foundation on which to operate the application, restricting the system unnecessarily is impractical. To this end, if CAs take too long to generate session keys (considering that unlike pre-distribution, a pool of keys does not exist), it is hoped that a more secure key is being generated; however, this must not exceed the associated risk of the application.

For the very same reason, the level of security should not be too low so as to make it a trivial task for a malicious individual to gain insight into communicated messages during that session. Optimisation on a case-by-case basis is required to address this balance between the level of security and the time it takes to generate session keys.

9.5.4 Random Key Pre-Distribution

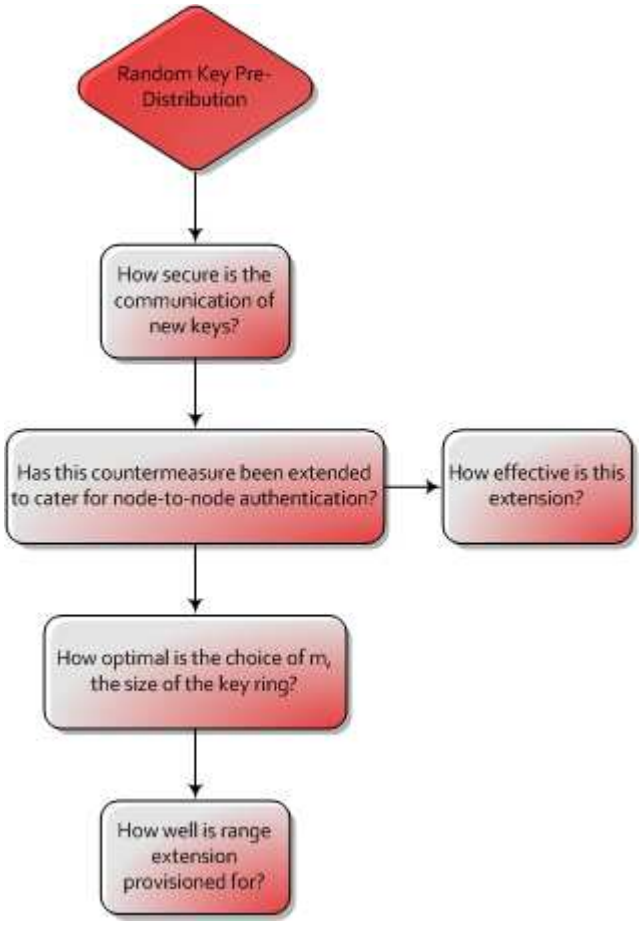

Figure 9.9: Random Key Pre-Distribution

The basic random key pre-distribution (RKP) scheme preloads nodes with keys and only considers the current pool of keys. In other words, no new keys can be introduced into the system by a new node without having some old keys pre-loaded onto it as well. This may not be problematic when new nodes enter the system, but could have dire consequences if

a particular key has been compromised, since just revoking this key would reduce network connectivity. A workaround would imply that re-keying has to be done. However, this new key has to be securely communicated or else this would amount to it possibly being compromised as well.

RKP in its most basic form is unsuccessful in carrying out node-to-node authentication. This is primarily due to the non-zero probability of two nodes possessing the same key ring after the random selections from the key pool have been made. This effectively prevents nodes from being cryptographically unique. The framework is open to the possibility that extensions to the basic scheme are achievable and entrusts it to the developer/designer to assess the effectiveness of this extension.

RKP, in the initialisation phase, loads m random keys into each member node's memory. Depending on the density of nodes and the size of the network, an optimisation for the value of m, the size of the key ring, would need to be calculated to ensure fair balance between security and network connectivity.

The basic RKP does not guarantee that every node will be within transmission range of a node with which it shares a key. Range extension is a technique that allows nodes either to amplify their transmission range or request other nodes to forward their messages to the intended recipients. Once again, if not done securely or without suspecting that intermediate nodes could be Sybil nodes, this could lead to valuable information being disclosed.

9.5.5 q-Composite RKP

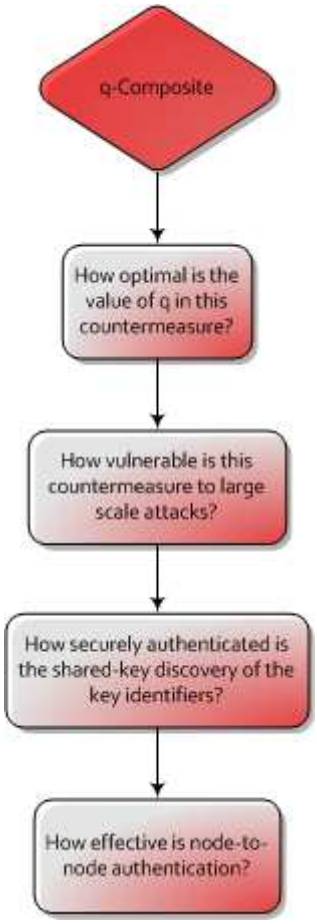

Figure 9.10: q-Composite RKP Specific Functionality

The choice of the value of q is critical in the success of q-composite schemes. An improper selection can result in an undesirable imbalance with regard to network connectivity and level of security for a given network density. More accurately stated, for a given q (i.e. the minimum number of keys that two nodes wanting to communicate must share), there is an optimal value of S (the size of the key pool) that must be chosen. Fundamentally, the trade-off is that the bigger the value of S for a given q, the lower the probability that these two nodes will be able to communicate but having S too small would imply that it is that much easier for an attacker to discover a major fraction of the key pool by capturing just a few nodes.

Although q-composite schemes are somewhat more secure than the basic RKP technique, they are still vulnerable to attacks that compromise a significant number of nodes. Effectively, this type of attack aims to acquire a considerable portion of key pool by collating the disparate information about secret keys on individual nodes. There is a

definite probability that if enough nodes are compromised, the entire key pool can be determined, thus rendering the countermeasure ineffective.

During the shared key discovery process, nodes will determine if they are able to communicate with each other by looking at the minimum number of keys they share. If this value is greater than or equal to q, then successful communication can occur. The challenge, however, is that nodes cannot explicitly trust that if a node claims to have the same portion of the key ring, this is in fact true. Some method will have to be employed in order to test this validity without divulging any secret information. There are various ways to accomplish this, e.g. one-time challenges, hashing, etc. but rather than place restrictions, the framework asks only that the implementation be rated for effectiveness.

In view of the fact that q-composite is essentially an upgraded view of RKP, there are still some shortfalls that have been inherited from its predecessor. Node-to-node authentication remains a dilemma in q-composite schemes for the same reason that RKP is affected i.e. there is a real possibility that two nodes could have exactly the same keys loaded in the key pre-distribution stage, thus eliminating any guarantees of node uniqueness.

9.5.6 Random Pairwise Scheme

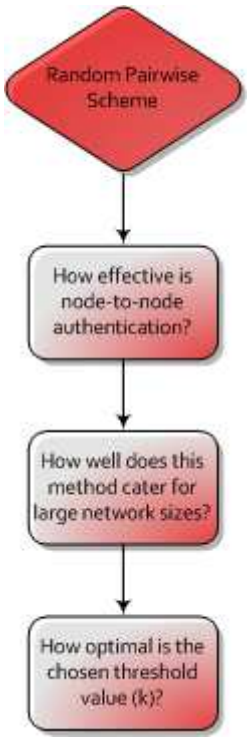

Figure 9.11: Random Pairwise Scheme Specific Functionality

The random pairwise scheme is an improvement on the basic pairwise key establishment scheme where node-to-node authentication is achieved but without demanding the large storage space of n-1 key spaces. Implementations of this scheme could, however, have varying degrees of success in achieving high effectiveness with regard to this form of authentication. More accurately stated, node-to-node authentication would be more worthwhile in situations where the presence of BSs or CHs is not substantial (e.g. member nodes are unable to communicate directly with the BS/CH). The onus is left to the designer/developer to decide if member node-to-node authentication is required and whether the implementation effectively delegates this responsibility away from the BS/CH to the member nodes.

Bearing in mind that this countermeasure pairs off nodes prior to deployment, scalability of the network with regard to post-deployment node additions is an unresolved issue. This is especially true for large networks. Provision can be made to future-proof the implementation if growth factors are considered during initial sizing of the network, but quite often the evolution of the network is unpredictable.

The size of the key ring (k) is imperative in determining the secure network connectivity, as well as the upper bound on the network size. It is then obvious that the value of k should be given careful consideration when implementing the random pairwise scheme.

9.6 PHASED-OUT ENCRYPTION ALGORITHMS

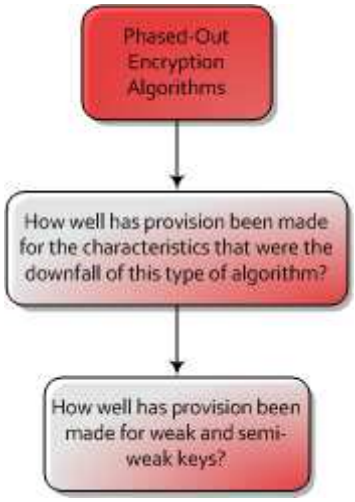

Figure 9.12: Phased-Out Encryption Algorithms Specific Functionality

Essentially, this classification of algorithms has been phased out by the National Institute of Standards and Technology (NIST) for a specific reason. Usually, this reason is that the

algorithm isn't cryptographically secure any longer. To illustrate this, Data Encryption Standard (DES) will be used as an example.

If the developer/designer is still comfortable using a phased-out algorithm (which is not unreasonable), he/she must take into account the reason(s) that justified the phasing out of this algorithm as insecure. Moreover, provision ought to be made for addressing these shortcomings if it is deemed possible that this vulnerability could exist in the intended environment. It must be noted that some of these algorithms, like DES, may be too resource- hungry for current WSN nodes to implement practically; however, the framework rates make provision for these countermeasures as possible Sybil countermeasures for future-proofing the taxonomy. At the rate that WSN nodes are growing with regard to hardware resources, in the near future nodes may be able to handle stronger computations.

Sometimes, with phased-out algorithms, there are weak keys or pairs of semi-weak keys where encryption with a key would function no differently when compared to decryption with another key (a case in point being DES). Caution should be exercised that these keys are avoided in the implementation.

9.7 CURRENT ENCRYPTION ALGORITHMS

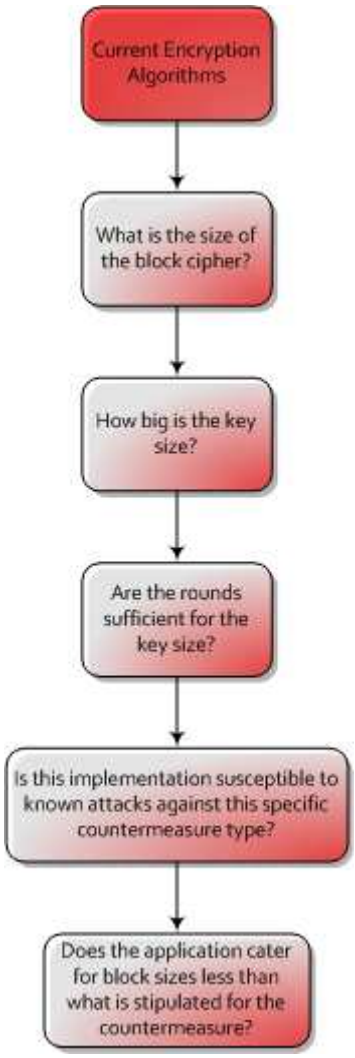

Figure 9.13: Current Encryption Algorithms Specific Functionality

This classification is the group of symmetric algorithms that are NIST-approved and have not officially been proved to be insecure. Here again, the framework does not rate the countermeasure; instead it attempts to ensure that strict adherence of the implementation to the standard is maintained. Advanced encryption standard (AES) is going to be used to illustrate the properties of this part of the framework. Again, it should be noted that although the known current algorithms may be too computationally strenuous for current WSN nodes to implement practically, the framework rates these techniques as possible countermeasures with the intention of future-proofing the taxonomy.

Typically, algorithms specify the size of the block cipher to be encrypted. Failure to comply with the standard when implementing this countermeasure can yield unwanted results and more specifically cannot guarantee the documented level of security. For

example, AES uses the Rijndael algorithm [36], which can operate on block sizes of multiples of 32. AES, however, stipulates that the cipher block must be 128 bits only.

The standard for some symmetric encryption algorithms exists in various versions. AES exists as AES-128, AES-192 and AES-256 and can use key sizes of 128 bits, 192 bits or 256 bits respectively; the weakest security being offered by the 128bit option, while the 256 bit version exhibits the strongest level of security. The framework is cognisant of the fact that the bigger the key, the more resources a node would use to encrypt and decrypt messages, However, this trade-off would be handled in the general overhead section and in this case would reward the security strength of the countermeasure.

The rounds or repetitions that ciphers ought to be iterated through are generally specified with the NIST publication. AES is stipulated as a number of repetitions or rounds of Rjindael transformations that alter the plaintext into ciphertext. The official standard for these algorithms prescribes the number of rounds of transformations, which is normally different for each key size. Deviation from this instruction cannot guarantee the documented performance.

Even though these algorithms are existing standards, there is no implicit assurance that there are no possible attacks against this algorithm. A case in point is that side channel attacks are vulnerabilities that are exploited on system implementations of AES instead of on the cipher itself. The designer needs to take precautions that he/she does not open the system to types of attacks that are known to be vulnerabilities, as this could effectively leak information on keys, thereby allowing other attacks, including Sybil attacks, to be launched.

As noted above, the ciphers operate on specific data block sizes and considering that WSNs may have smaller data blocks to transmit or encode, provision for this ought to be made. Although padding may be used, the designer must take heed that this may yield lower strengths in security.

9.8 PROPRIETARY ENCRYPTION ALGORITHMS

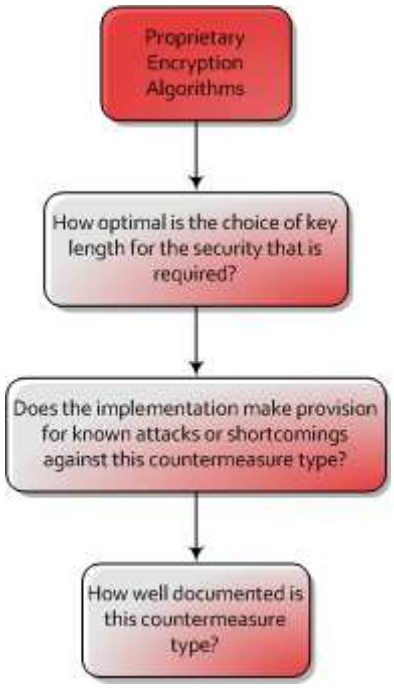

Figure 9.14: Proprietary Encryption Algorithms Specific Functionality

Proprietary algorithms are algorithms that have not officially been evaluated by a standards body, such as NIST, and any claims to performance, security levels or success levels can either be accepted by implementers or would need to be verified. To illustrate functionality, the TinySec security feature will be used. TinySec is a link-layer security feature based on SKIPJACK, an 80-bit symmetric cipher for TinyOS [37]. TinyOS is an operating system for WSN nodes.

Crudely speaking, the length of the key is proportional to the level of security that can be leveraged from a countermeasure, assuming certain other parameters remain constant. There is a fine balance between the level of security and the performance of the network and it remains the responsibility of the implementer to choose a suitable size for key length, bearing in mind that longer keys typically imply increased resource requirements.

Although not official, common proprietary algorithms have their known attacks or shortcomings made public. The designer/developer would need to cater for this if there is a finite but significant possibility that this vulnerability could be exploited. An illustration is the weakness of TinySec. TinySec is dependent on a single key and therefore cannot securely execute re-keying. Some form of extension would need to be introduced into the implementation to eliminate further vulnerabilities.

To facilitate implementers to achieve desired security levels, the algorithm's performance and known shortcomings should be well documented. Furthermore, the algorithm itself should be detailed enough in its documentation to allow developers to implement it consistently.

9.9 ASYMMETRIC APPROACHES

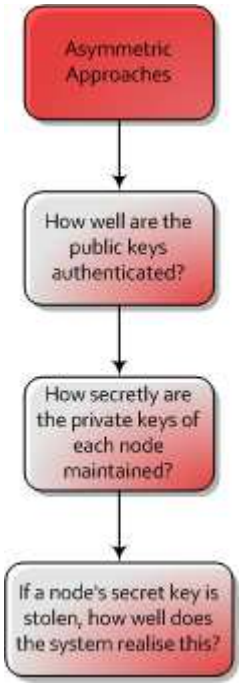

Figure 9.15: Asymmetric Approaches Specific Functionality

Unlike some symmetric approaches where there is a different key that is associated with communication between a pair of nodes, asymmetric cryptography insists that any node wanting to send a secure communication to a particular node (e.g. node A) will have to use the public key of this node (i.e. the public key of node A). However, if this node (node A) has been compromised without the system being aware of the breach, then all nodes will still communicate with the malicious/faulty node. More especially, no node can be absolutely sure that a public key really does belong to a particular entity without this node verifying this ownership; hence the requirement for the authentication of public keys.

Loss of private keys, whether intentional or accidental, implies that encrypted messages, irrespective of how critical they are, cannot be decrypted. Furthermore, if private keys are indeed stolen and go undetected, then Sybil attacks can be launched quite easily against the system. It is important that strong measures be implemented in securing private keys.

In the event that a node's private key is stolen or lost, the system should be able to detect this anomaly, revoke this key and communicate this securely to the network. Note that this is unlike the "detection" component in the taxonomy, as this method does not detect a Sybil attack but rather the realisation of a possible breakdown in the efficacy of the "prevention" method (i.e. the asymmetric technique).

9.10 ASYMMETRIC ENCRYPTION ALGORITHMS

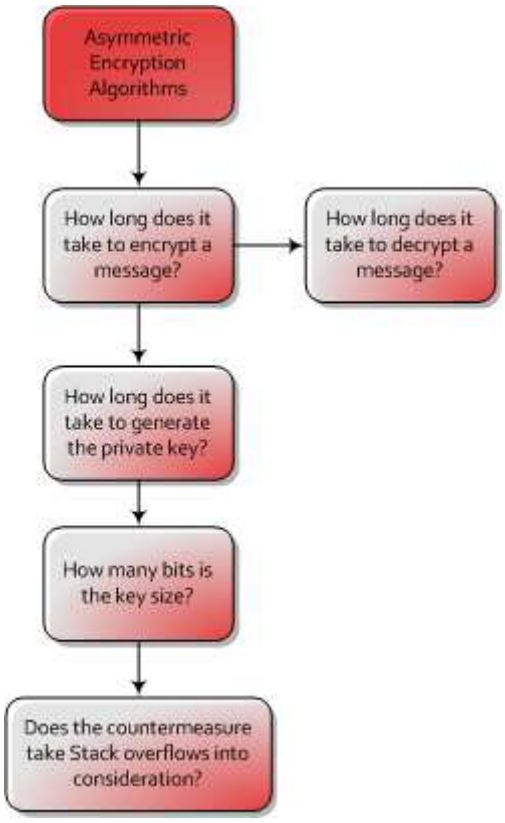

Figure 9.16: Asymmetric Encryption Algorithms Specific Functionality

The framework does categorise the total speed of the countermeasure in the general overheads section; however, the individual operations that asymmetric cryptographic algorithms use have to be rated. The three types of operations are: how long it takes to generate a private key, how long it takes to encrypt messages and how long it takes to decrypt messages.

Similar to symmetric encryption, these processes are significantly dependent on the key size. Once again, depending on the security requirements, the length of the key would need to be optimised. The "specific functionality" component of the framework rewards the use

of stronger keys but the overhead issues related to required resources are catered for in the overhead section.

Implementations of asymmetric cryptography on WSN nodes can run into stack overflow problems owing to the smaller sizes of memory addresses. Because of the volatile nature of this issue, the designer/developer cannot afford to overlook this possibility. An example to illustrate this point is the case of ECC; implementations of EccM 2.0, to some extent, overcame stack overflow problems that were present in its predecessor, EccM 1.0 [38].

9.11 DIGITAL SIGNATURES

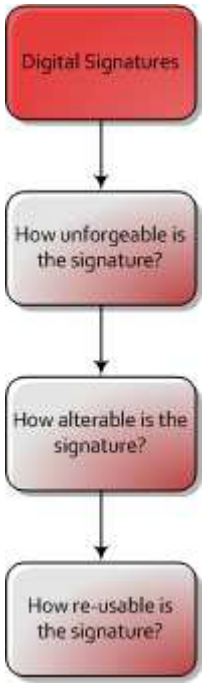

Figure 9.17: Digital Signatures Specific Functionality

Asymmetric cryptographic techniques are perfectly positioned to enable digital signatures [18]. As much as this method has proven to be computationally exhaustive for current WSN nodes, the technique is made a part of the framework in the interest of future-proofing the taxonomy and framework alike. [18] also describes some properties that digital signatures ought to abide by.

The signature should not be forgeable. Much akin to real world paper-based signatures, the authenticity of an electronic signature is directly related to the difficulty of forging it.

It should be cryptographically impossible for any digital signature to be altered or more accurately, if any alteration is made it should not go undetected and/or be accepted as an honest signature.

With the use of digital signatures, legitimate messages should be disallowed to be offered again to any node. Any deviation from this can leave the system susceptible to replay attacks. Specifically related to Sybil attacks, in a trust-based system, if malicious nodes exploit this vulnerability of being allowed to launch replay attacks, they can then use previous signed messages to force the application to trust false identities as being honest entities. The developer must ensure that digitally signed messages, once presented, must somehow be flagged as expired, or that each signed message is different by using appropriate nonces to ensure freshness.

9.12 HASH FUNCTIONS

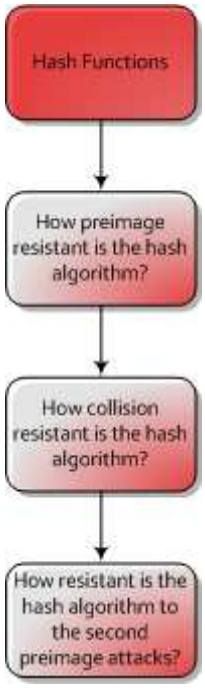

Figure 9.18: Hash Functions Specific Functionality

Hash functions can be utilised in both symmetric and asymmetric cryptography. These operations are affected primarily by three types of challenges [21].

Preimage vulnerabilities occur when the hash algorithm has the property that if a hash is known, it is not cryptographically infeasible for another message to be transformed into the same value when operated upon with the hash function [21].

A hash function is said to be collision-resistant if given two different messages, it is computationally infeasible to realise a hash function which, when applied to these messages, will yield the same result [21].

Second preimage attacks exploit the vulnerability that given a message; it isn't hard to find a second message which, when the hash function is applied to it, will yield the same result as when applied to the first message.

It should be noted that these vulnerabilities appear to be similar or even the same at a cursory glance, but are quite different. Due consideration ought to be given to understanding the differences, but it is even more important that the developer/designer should carefully make provision for these possibilities existing in the selected hash technique being employed.

9.13 DETECTION

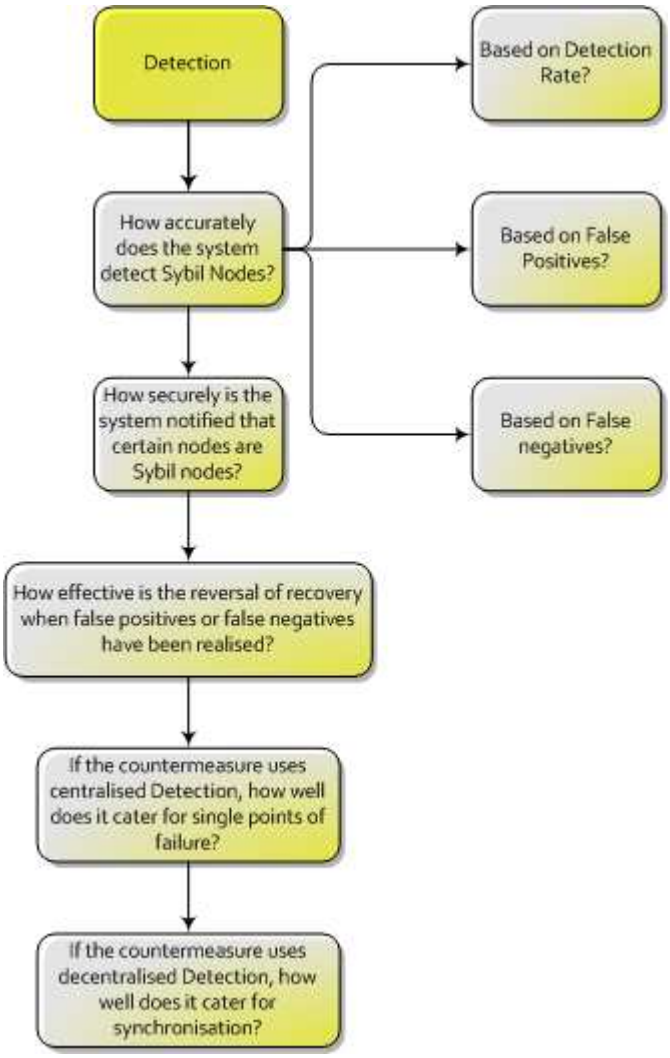

Figure 9.19: Detection Specific Functionality

Detection is the technique(s) that is employed to check if the system has Sybil nodes or is being attacked by Sybil nodes. Noting that detection methods, by themselves, cannot protect the system from attacks or resolve any issues with regard to a Sybil attack, one would typically find this type of defence being implemented with some recovery technique.

The framework rewards countermeasures in terms of how accurately Sybil nodes/attacks are detected. Accuracy is based on three different criteria, namely detection rate, false positives and false negatives. These definitions have been defined in an earlier chapter.

Once Sybil nodes have been accurately detected, the system must be made aware of these malicious entities and their respective identities. Moreover, this communication should

preferably not be sent to any malicious nodes, allowing them to take recourse. It is hoped that the malicious attacker will continue attempting to attack the system using the now conspicuous identity, eventually stopping because of vain attempts or depleted resources.

It is quite unlikely that any countermeasure would be 100% accurate; in reality, most countermeasures would have a significantly finite element of inaccuracy. For this reason, it makes sense that when the system does become aware of its mistake(s), the countermeasure rescinds its judgement against wrongfully accused nodes.

Calculations for detection can be done either centrally or can employ distributed detection techniques. Central detection usually involves all relevant detection data being forwarded to the CH or BS and the result then computed. The advantage is that these nodes are typically more powerful than member nodes; however, this also implies single points of failure.

Decentralised detection methods involve delegating the responsibility of detection calculations to member nodes, which relieves the pressure on the CH or BS, depending on the topology class. However, there is now the issue of synchronisation to ensure consistency in the classification of Sybil and non-Sybil nodes.

9.14 RESOURCE TESTING

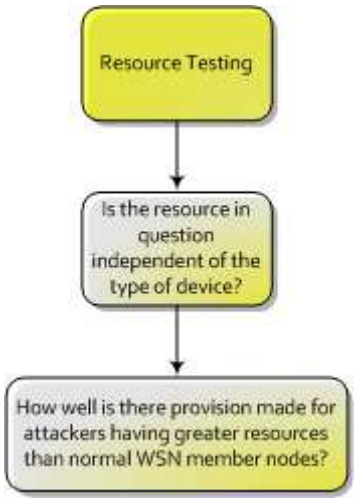

Figure 9.20: Resource Testing Specific Functionality

There are various resources that can be tested to verify the one-to-one mapping of identities to entities; however, as previously mentioned, computation and storage are not suitable tests to use for WSN resource testing. It should be stated that hardware resource

verification are not the only tests that can be performed. It is quite possible to use certificate information to achieve resource testing if there is high confidence that a physical node, irrespective of its class (i.e. laptop class or mote class) can only possess one certificate.

If it is deemed possible that this type of user may have access to either more resources or more than one resource. Provision ought to be made to influence this. This is especially true if the strength/amount of resource in question varies with the type of hardware device. For instance, laptop class attackers indeed have more hardware resources (i.e. computation, storage and possibly the number of radios that can be connected to it) when compared to common WSN nodes, but there are certain techniques that can be combined with resource testing that can weed out these malicious nodes.

9.15 CLOCK SKEWS

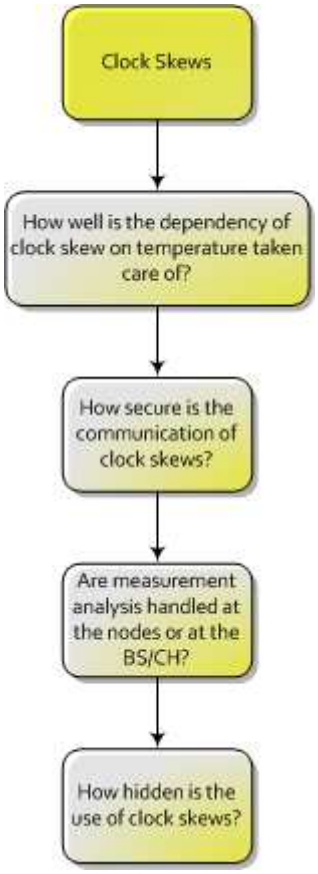

Figure 9.21: Clock Skews Specific Functionality

Clock skews are directly related to the clock signals given off by the timing circuit in WSN nodes. This is typically either done by an RC circuit or a quartz crystal. Because of the

reliance of this measurement on hardware, it does not come as a surprise that these values are dependent on temperature. Association or “fingerprinting” nodes using clock skews can yield inaccurate results if variation in temperature is not a variable in the clock skew equation. Thermal compensation techniques seem a viable solution to counter this effect.

When nodes are communicating their clock skew values, it would be worthwhile to cloak this information from any other node except the intended recipient. This would effectively prevent a Sybil attacker from duplicating known correct values and laying claim to these skews.

In order for an attacker to gain insight into a clock skew, he/she would need both the timestamps for the end nodes and the sink nodes [35]. If the countermeasure implementation of a clock skews method to counter Sybil attacks requires the analysis of the measurements done at a member node, instead of at a cluster head or a base station, there is a greater chance of divulging timestamp information, as there would be more links to secure. Furthermore, the path links could prove to be unpredictable. In the case of just the CH or BS doing the analysis, the developer/designer would just need to secure these links, with timestamps only being sent to the CH/BS and not the other way around.

Clock skews need to be measured over a covert channel in WSNs. Member nodes should not even be aware that timestamp measurements are being analysed for clock skew calculations. It may even be worthwhile if normal routing uses timestamps, thereby making it difficult for attackers to know whether clock skews are being used as the active countermeasure. Furthermore, member nodes must not be able to distinguish between packets that are used for clock skew measurements and packets that are used for other time-dependent applications.

9.16 LOCALIZATION

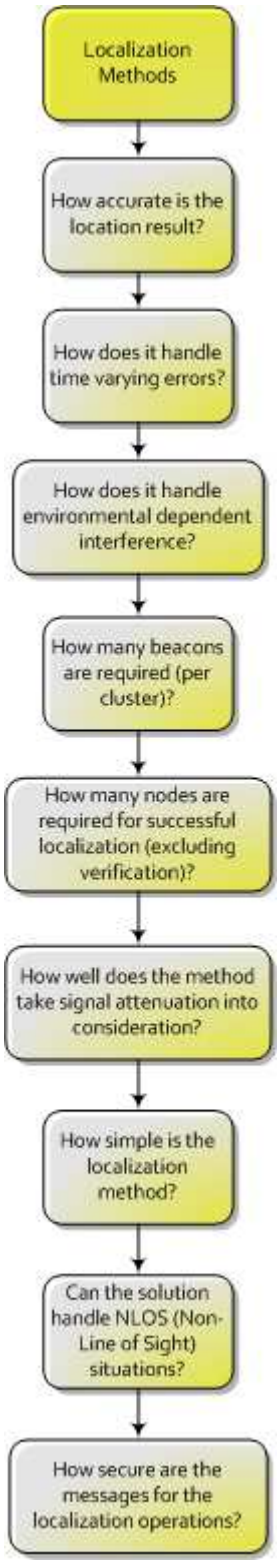

Figure 9.22: Localization Specific Functionality

For localization to be effective in countering Sybil attacks, the accuracy of geographical position is crucial. It is sufficient for the system to have at least enough detail in the

localization result to determine if two nodes are right next to each other, but not to confuse them as one entity.

Localization is quite sensitive to time-varying errors, since wireless sensor networks primarily utilise radio frequency (RF) technology to communicate. Time-varying errors stem from additive noise and RF interference such as harmonics and can be considerably reduced if a few measurements are used and averaged to get a final result instead of just assuming that one result is correct. If not considered, localization can yield unpredictable results.

Environmental factors affecting localization are generally due to obstacles and can cause signals to bounce or be absorbed, depending on the specifics of the environment. Because of the diverse possibilities or unwanted side-effects brought about by the surroundings, environmentally dependent factors are dealt with by considering them random variables. However, there are certain methods that can be employed to reduce these effects.

Localization generally requires a node to calculate its position in relation to another node that knows its global position. Usually, because of the special hardware required (e.g. GPS), these nodes are beacons or cluster heads. Some localization methods go further in requiring more than one beacon to be used for position calculations. The framework rewards the implementation of more than one beacon being required with the premise that the possibility of a rogue beacon is being considered or in the absence of a compromised beacon, that redundancy is being attempted with this instance of the countermeasure in question.

Besides using a beacon(s) or a landmark(s) to retrieve global positions, nodes themselves have to use this information to calculate the location of another node. A minimum requirement is that localization needs two nodes to calculate the position of a third node. Here again, the framework encourages the use of more nodes, the focus being on achieving better accuracy.

RF signals are prone to attenuation due to energy losses (i.e. absorption, reflection, refraction and dispersion) to the environment. The greater the distance between two communicating nodes, the greater the signal attenuation they experience. Suitable signal-propagation models need to be incorporated in order to alleviate the countermeasure from these phenomena.

Localization calculations can vary in degrees of complication with varying degrees of success. Complicated calculations in order to realise accurate localization are viewed in a poor light by the framework purely from a platform and re-usability point of view.

Although WSN nodes do not have to be in direct view of each other, when using non-line-of-sight (NLOS) deployments, the localization implementation needs to take this into account. Furthermore, it is not unusual for the application owner not to know the status of the environment prior to deployment with regard to line-of-sight (LOS). It is good practice to plan for both scenarios.

When attempting to localise a node, two or more nodes need to collaborate and verify their findings in order to reach a conclusion with regard to the localization result. Ironically, in doing this, the system can be duped into believing a false reading if the nodes that are verifiers are actually other Sybil nodes. It helps if the communication required to achieve a localization result is done securely so as to prevent malicious nodes from eavesdropping.

9.17 RSSI

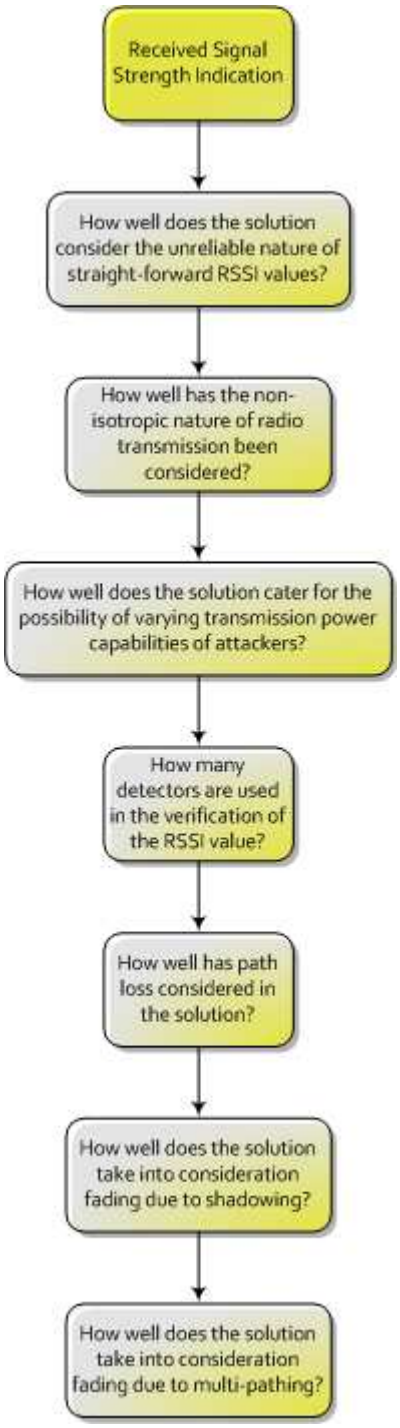

Figure 9.23: Received Signal Strength Indication Specific Functionality

The accuracy of RSSI is quite dependent on the inherent nature of RF signals. Any implementation of RSSI should avoid using only the straightforward received signal strength, as this would be far too erroneous to be useful. The countermeasure could have a

calibration phase to determine the values of variables that are solely dependent on environmental effects that remain constant.

An isotropic antenna is a hypothetical, lossless antenna that has equal radiation intensity in all directions [39]. RF antennas usually exhibit very non-isotropic characteristics. From a localization point of view, this effectively translates to non-uniform signal strength properties of a transmitting source node being experienced and this effect is especially dependent on the orientation of the antenna. It must be noted that it is quite possible to get quasi-isotropic antenna (such as a small loop antenna or a short dipole antenna); however, not all devices are expected to use this hardware.

In order to dupe RSSI techniques, attackers may vary their devices' transmitting power so as to make it appear that they are further or closer than previous messages that were transmitted. If this is successful, the detection method will inadvertently allow the attacker to present multiple Sybil identities. Due consideration needs to be exercised with these types of attacks.

A possible counter to the above-mentioned challenge would be to use at least two time-synchronised localizing nodes to verify the transmitting node. For this reason, the framework recognises the use of more than one node to verify the RSSI amplitude. Note that this is different from getting the initial localization result.

RF Path loss is the decrease of power density when an RF signal travels through free space [39]. It is influenced by various factors such as reflection, refraction, dispersion, diffusion and absorption. The countermeasure should use signal propagation and path loss models to enhance the localization result effectively.

Fading is a variation of path loss attenuation and varies with time. It is dependent on geographical position, transmission modulation frequency and environmental effects. Unlike path loss, calibration cannot solve this issue and some adaptive means of factoring in this effect needs to be introduced. Shadow fading is the fading that results when the transmitted signal is obstructed by objects that affect wave propagation [40].

Multipath interference occurs when the transmitted signal arrives at the receiver from different directions, with different path lengths, attenuation and delays [39]. Variations of constructive and destructive super-positioning occur and the worst case scenario is the possibility of the transmitted signal disappearing completely. Multipath-induced fading

results when frequency-selective fading is experienced. Once again, due emphasis should be placed on eliminating this effect on the localization result. A possibility is to use either the first signal arriving or the signal with the highest amplitude and to calculate the position accordingly.

9.18 TIME OF ARRIVAL

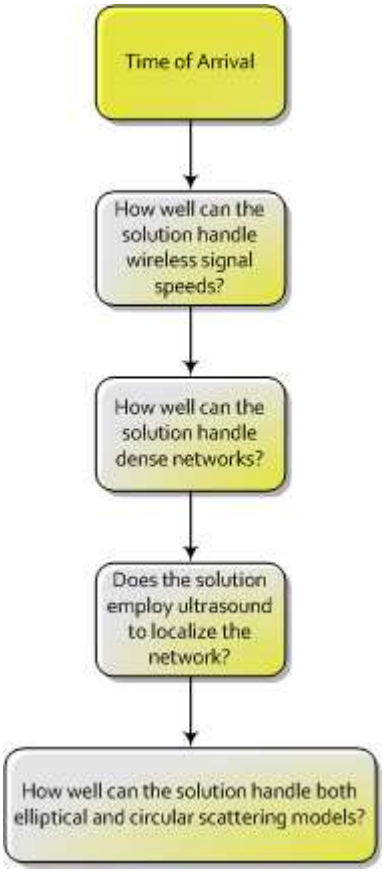

Figure 9.24: Time of Arrival Specific Functionality

It is obvious that TOA techniques are heavily dependent on high temporal resolutions, especially when working with wireless signal frequencies that are typically around the 2.4 GHz range if not higher. Considering that most WSN nodes have 8-bit processors, the handling of stacks and addresses can become rather cumbersome. According to [34], the basis for the success of TOA techniques is the receiver's ability to approximate the arrival time of the LOS signal accurately.

Unlike AOA and RSSI, TOA implementations can be less sensitive to inter-device distances and if this opportunity is seized, localization in dense networks can be less of a challenge when compared to some other localization counterparts. The framework rewards

taking advantage of these opportunities, e.g. keeping the timestamp of the original source node in the message even if it is being forwarded.

The speed of RF signals is about 10^6 times faster than the speed of sound and on these grounds nodes may have acoustic transmitters that use TOA for localization. The framework rewards the use of this technology only so that better accuracy can be achieved through reduction in sensitivity to speed. However, the requirement of ultrasonic hardware is punished in the overheads section, as it requires additional hardware.

The existence of LOS from the transmitting node to the localization nodes is critical to the success of TOA methods. However, objects that are not directly obstructing LOS pose the problem of scattering the signal. Two types of scattering models are generally used with RF signals: circular and elliptical. Provision for both types is encouraged by the framework.

9.19 TIME DIFFERENCE OF ARRIVAL

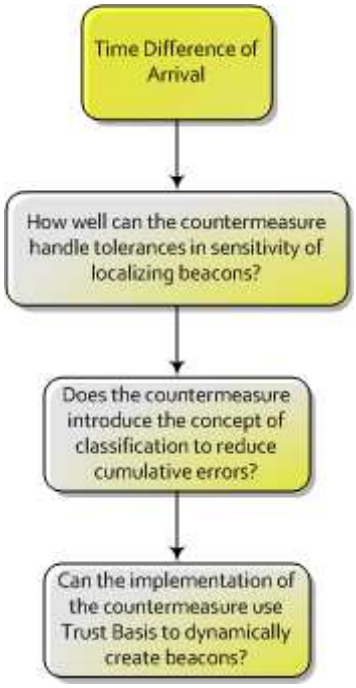

Figure 9.25: Time Difference of Arrival Specific Functionality

TDOA is less dependent on time synchronisation of the network than its predecessor, TOA. However, when calculating the TDOA ratios, beacon nodes must exhibit the same sensitivity when sensing the arrival of the source node's message, else there would be errors in the result. Usually this would result in a single intersection point not being

established. The implementation should consider having certain tolerances built into its localization method.

In TDOA, beacon nodes can be classified depending on how they achieve their own global position. If it were originally an anchor node, i.e. either through manual placement or by using hardware (such as a GPS) then this beacon would be considered a high anchor node. If it had been made an anchor after being localised by other anchor nodes, then this would be a lower level anchor. By using this classification, the countermeasure could factor this variable into calculations, thus reducing cumulative errors for later localization results.

Because of the speciality of beacon or anchor nodes, the framework encourages the creation of other secondary anchor nodes. These secondary anchors help to reduce the cost of too many beacons as well as effectively increasing the density of beacon nodes in a given area. However, appropriate caution needs to be exercised when delegating responsibility for localization in order to prevent malicious attackers from becoming a beacon.

9.20 ANGLE OF ARRIVAL

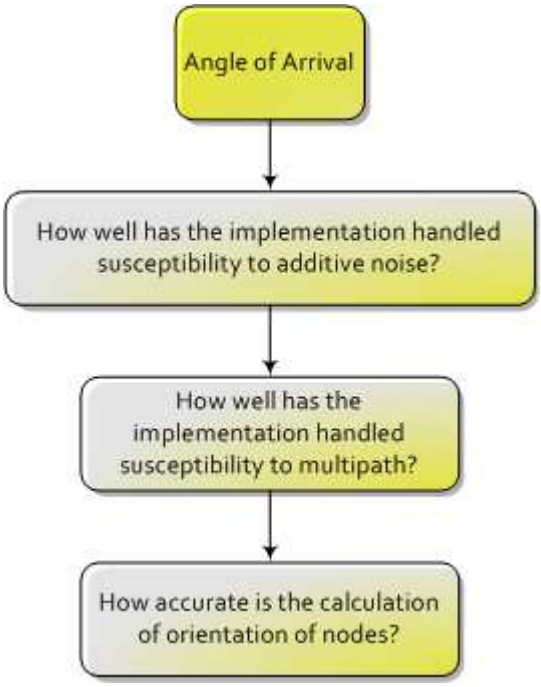

Figure 9.26: Angle of Arrival Specific Functionality

AOA measurements are susceptible to noise and to achieve a reasonable result, the signal should be applied to pre-filters and amplify the spectral components that do not exude

excessive noise. Thereafter, cross-correlation techniques can be carried out [34]. Suitable models that can be used to reduce static noise should also be introduced.

AOA, very similar to RSSI, is quite negatively affected by multipath-induced fading. For AOA, early-arriving multipath signals appear rapidly after the LOS signal, thus affecting the location of the peak from the LOS signal [34]. This in turn introduces random errors in the cross-correlation computations.

AOA is quite dependent on the orientation of the node. Instead of using distance, localization is effectively calculated on relative angles. To achieve accurate localization, both the orientation and the orientation calculations need to have high resolutions. [34] claims that RF AOA exhibits a deviation around 3^{o} in orientation realisation.

9.21 RECOVERY

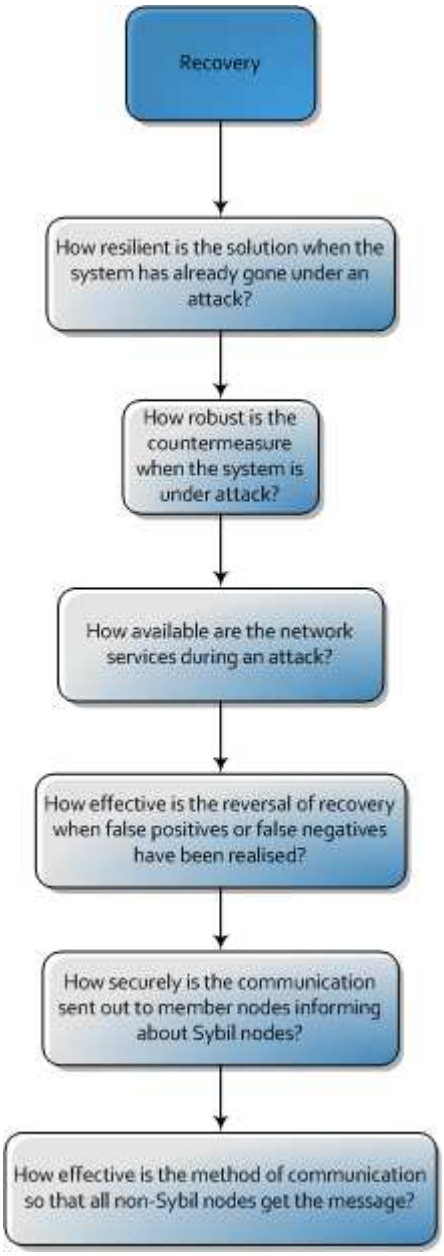

Figure 9.27: Recovery Specific Functionality

Recovery is the action taken by the countermeasure/system in the event that a Sybil attack is detected. This can be evasive, offensive or defensive in nature. Any recovery technique should ensure that the system is able to reject further spread of the Sybil attack after infection. After an attack, the Sybil nodes may not be actively destroying the application's operational processes but may very well be listening in on normal messages. This essentially suggests that true nodes should not trust any messages coming from untrustworthy entities but at the same time should not kill the operation of the application

entirely. A possibility is for the network to increase its vigilance and security until the threat has been inoculated, e.g. apply re-keying or increase the strength of the keys.

During the attack, however, the system should be enabled to be fast in responding to a declared threat/attack. This obviously depends significantly on the speed of the detection mechanism used. Recovery mechanisms are inherently dependent on detection schemes; however, the converse is not necessarily true. It should be noted here that fast detection performance can kick-start a recovery process even before the attack has run its full course. However, even in the presence of an efficient detection implementation, a lethargic recovery method can prove insufficient to stem the rate of infection. The recovery technique should make the system agile enough to prevent the entire WSN or intended portion of the WSN from being compromised. This may even call for the application to behave unpredictably so as to thwart the Sybil attack, e.g. run sub-routines differently or address only fully secured nodes for communication.

As stated previously, a Sybil attack is a type of DOS attack and as such, if the recovery technique disables too large a portion of the WSN, effectively rendering it inoperable, the attacker would have achieved his/her goal. Only in extremely rare cases, where either the data are too confidential or too many nodes have been compromised, would it be acceptable to shut down the entire application. Network connectivity should be maintained to support continued application operation.

In the event that a node or a group of nodes have been falsely judged as being Sybil nodes, the countermeasure should have the ability to reinstate them securely into the network. Nevertheless, developers should be cognisant of the double-edged sword that this functionality presents. In a trust-based scheme, a Sybil node that has managed to gain the trust of true nodes may be able to restore other malicious entities into the system using this feature. As stated in the framework, this feature should be used to rectify both false positives and false negatives.

When the detection schema reveals certain nodes as being Sybil entities, it is the task of the recovery component to declare this to the rest of the WSN. This must be communicated in a secure fashion to prevent malicious nodes from realising they have been noticed. The advantage of doing this is that these Sybil nodes may continue transmitting on effectively deaf true nodes and either deplete their (i.e. the Sybil nodes') energy source (in the case of a normal WSN node attacker) or grow weary (in the case of a laptop-class attacker).

Besides the need for security in the communication of this important message, there should be some form of feedback to ensure that all true nodes get the message. For practical purposes, this calls for a handshaking process between true nodes and the designated "loudspeaker" node. The developer should take note that the nodes deemed true that this message is being delivered to could in reality be Sybil nodes that have not yet been identified. Furthermore, it is arguably unlikely that if a Sybil node gets this message it would confirm reception of the message. So, it is quite possible to detect further Sybil nodes by making it an obligation on the part of receiving nodes to provide feedback. As much as this handshaking process seems to prove to be twofold, caution should be taken not to label nodes that do not provide feedback as guaranteed Sybil nodes as, they could just not have received the communication.

9.22 SELF-HEALING

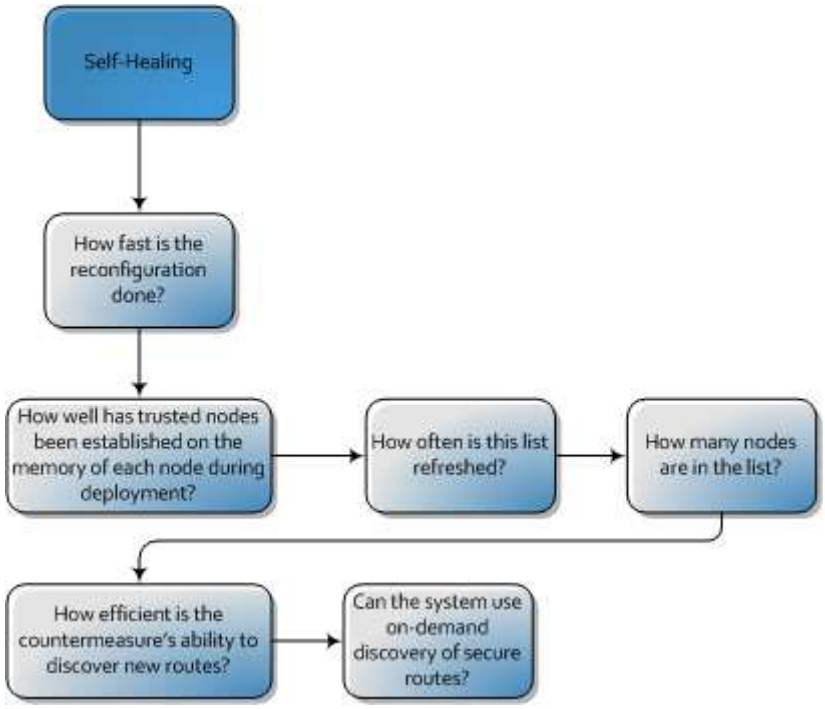

Figure 9.28: Self-Healing Specific Functionality

Self-healing is the ability of the network to re-arrange itself logically in order to sustain continued operation. Similar to the importance of speed in detection countermeasures, the efficiency of self-healing techniques in reconfiguring the network links for communication is of paramount importance. It is important to note, irrespective of how obvious it may seem, that true nodes closest to a Sybil node run the highest risk of being compromised. Analogous to traditional wired networks where actual communication paths are changed

for sustainability, WSN self-healing (being a wireless medium) should involve nodes that are within close proximity of Sybil nodes to communicate securely to known secure nodes and to discard any messages coming from its malicious neighbour.

In this regard, it makes sense for every true member node to maintain a list of definite trusted nodes securely within its transmission range. There are various practical options in accomplishing this:

- One may have these trusted nodes pre-loaded into the memory of member nodes and deploy them within transmission range of the nodes on this list.
- In the bootstrapping phase, one ought to ensure that every member node discovers its trusted nodes.
- One must ensure the use of a trust-based scheme that allows nodes to gain other node's trust dependent on a decent length of history of participating with this node.

There are various pros and cons to the above options and the developer should apply due diligence when choosing a suitable technique. The regularity with which this list is updated and the method by which it is updated securely are other factors to consider. Also, depending on the available memory, the number of trusted nodes that ought to be on this list must be optimised.

Notwithstanding the existence of the above-mentioned feature, nodes within a single hop of a known Sybil node should be able to discover new routes dynamically but securely to maintain network connectivity and to sustain the longevity of the application. This technique, however, is closely related to capabilities of routing protocols and is outside the scope of this study. Suffice it to say that this approach, if accomplished in a secure manner, can help to curb further Sybil attacks.

9.23 ISOLATION

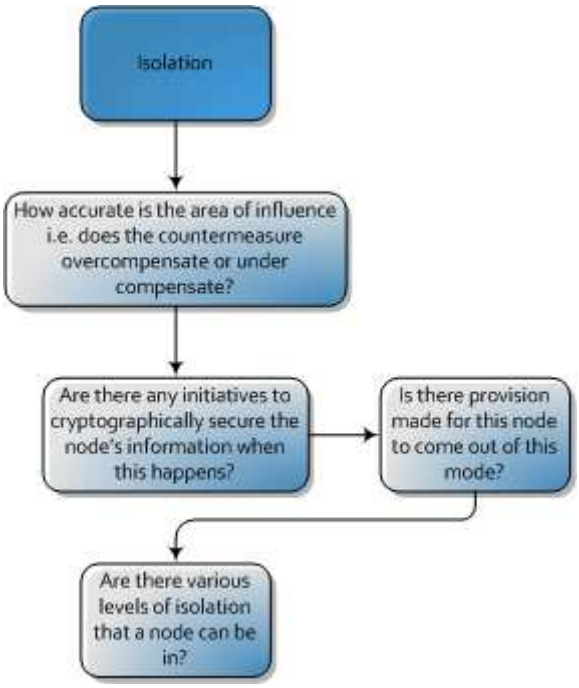

Figure 9.29: Isolation Specific Functionality

This method allows nodes, a cluster or a dynamic area of the network around detected Sybil nodes to be rendered inoperable, thus isolating the attack. The countermeasure should be accurate when isolating a specific section of the network. As a result of there being a delicate balance between losing network connectivity and the risk of excluding a Sybil node from isolation, the implementation should avoid both under-compensating and/or overcompensating.

It is highly likely (depending on the size of the isolated area) that true nodes are going to be subject to isolation and effectively not be in contact with the rest of the WSN. During this time, these non-Sybil nodes should conserve their precious battery-powered energy supply and effectively go into sleep mode. However, before doing so, it may be useful to cryptographically encrypt their data locally so in the event of it being physically compromised it would be that much more challenging for the attacker to gain information stored on the node's memory.

However, it is advisable that provision be made for the node to "come out of hiding" when the environment is deemed safe. At the very least, the node in sleep mode must be able to listen for messages declaring a safe environment as well as the "declaring node" to be able to communicate this message securely to sleeping, isolated nodes.

A desirable option that may further seek to improve network connectivity is not to limit the options of isolated true nodes only to go to sleep but, depending on the risk profile of the application, as well as the severity of the attack, to be able to operate on a sub-optimal level, e.g. only be able to forward encrypted messages.

9.24 LOGICAL REMOVAL OF SYBIL NODES

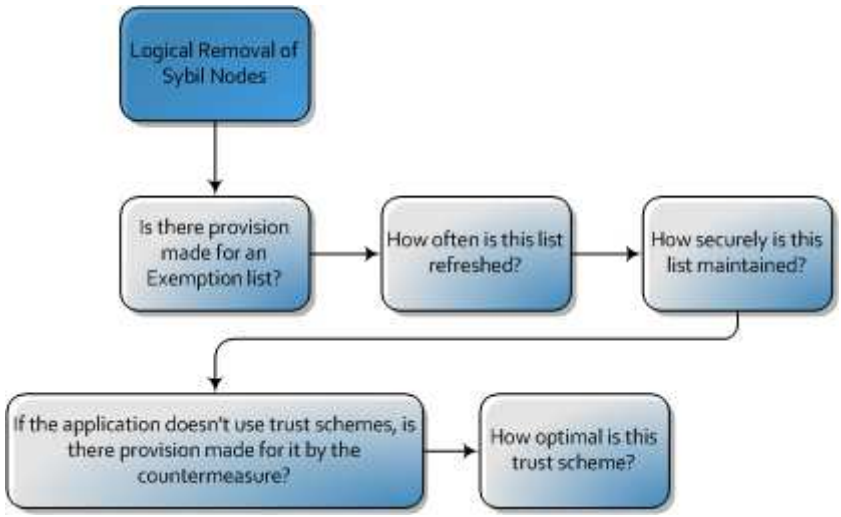

Figure 9.30: Logical Removal of Sybil Nodes Specific Functionality

This practice is employed when the Sybil nodes are effectively exposed and thereafter ignored by the rest of the system. As much as this seems to be a very noncommittal and lazy approach to recovery, it does not draw on too much of already limited resources, thus making it favourable for applications that require basic recovery. Regardless of this, the basic measure can be expanded, depending on the requirements of the developer. An exemption list (i.e. a list of entities that should not be logically removed from communication irrespective of the detection mechanism declaring this node going rogue) can be distributed or pre-distributed to member nodes. Due caution must be taken in the event that a node on the exemption list does indeed become compromised.

If this list is a dynamic register of nodes, then care must be taken to keep it up to date optimally and regularly. Moreover, this list should not be maintained trivially, as it is likely to be a target for malicious entities. Options to store this list securely may be to keep it on a CH or BS and only allow member nodes to verify against it before ignoring a suspected or declared Sybil node.

In the event that this list is not present, effective trust-based schemes may prove to assist in populating it. If the system does not employ one, it may be desirable for the recovery mechanism to make provision for the introduction of an optimal trust-based mechanism.

CHAPTER 10 RISK ANALYSIS

Chapter 10 describes the risk profiles (i.e. low risk, medium risk, high risk and the ultimate risk profiles). Furthermore, each risk profile is rationalised so as to further clarify these categories. The Risk Analysis framework is then introduced and a detailed discussion on the contents thereof is presented.

As stated earlier, there is a requirement to assist an application developer to choose a suitable countermeasure against Sybil attacks. A key point to remember when selecting any security option is that the cost of security should not outweigh the risk posed to the application. It is primarily for this reason that each application has to go through a risk analysis process to evaluate what would be the most appropriate countermeasure.

10.1 RISK CLASSIFICATIONS OF WSN APPLICATIONS

Due to the diversity of application types and the varying overlap of security requirements, the risk analysis framework classifies WSNs based on risk profile. Using this approach, the designer/developer is given the choice to decide how strong a countermeasure he/she wants to employ in the system. Four risk profile classifications are defined based on the importance of the data, the criticality of the application and the operational functionalities of the application.

Effectively, these categories would allow the POC tool to gather requirements. Each category will define how stringent a countermeasure needs to be to satisfy the requirement of the user. It should be noted that examples used here by no means imply that if an application does fit the description of the example, the risk profile classification would comply with the developer's/designer's requirements. The onus still rests on the developer/designer to decide on the risk profile that he/she deems suitable, as well as the measures that he/she wants to employ. The examples and the measures to be taken are only recommendations and are the minimum requirements.

10.1.1 Low Risk

The WSN is not used for any critical application and the loss of data, either accidental or through acts of maliciousness, doesn't pose a problem.

The system designer/developer may opt for either no Security Solution or just Detection so he/she may know when to trust the integrity of the data or when to view the data/results with suspicion. An example could be if the WSN is a training network to simulate an application that requires real world data but the designer is not concerned with data going missing.

10.1.2 Medium Risk

The WSN isn't used for any critical application but the loss of data is frowned upon. Furthermore, the choice of the routing protocol is not affected by Sybil attacks.

This scenario emphasises the need for Prevention so that the application would be protected from attacks being launched against the system. However, in the event that there is a breach the developer may want Detection so that, just as in the previous scenario, he/she would be able to know when not to rely on the quality of the data.

10.1.3 High Risk

The WSN application is critical and the loss of data is undesirable. However, the operational functionalities (e.g. selection for the routing protocol) are unaffected by Sybil attacks.

Typically, this system would require Prevention, Detection and Recovery. The reasons for this are as follows:

- The application needs to be able to prevent Sybil attacks from being launched against it in the first place and to filter weak Sybil attacks from even affecting it.
- In the likelihood that a strong/intelligent Sybil attack breaches the Prevention boundaries, the system must be able to detect its presence and inform the application.
- Moreover, the system must then take defensive, offensive or evasive action against the malicious nodes. This would be to ensure continued operation of the application in view of its criticality.

10.1.4 Ultimate Risk

This is the highest possible risk category of WSN applications where data are absolutely critical and the application demands extremely high availability as far as WSNs are concerned. Loss of data is more critical than the application being operational, i.e. the system must fail securely if it ever does. It is expected that there would be very few real world applications of this nature but, for example, one could imagine a battlefield scenario where nodes are effectively monitoring the distribution of soldiers, targets or supplies. Instead of being provided with incorrect data, a soldier making decisions would rather be informed that the system is not in operation as a result of a successful attack against it.

Besides having prevention, detection and recovery mechanisms in place, these circumstances call for specific mechanisms to be implemented. The study has defined a few rules that would be a step up from the “high risk” profile:

- The countermeasure should be able to prevent an attacker from reading the contents of a physically compromised node.
- The detection techniques should not only trigger recovery mechanisms when a Sybil attack is realised, but also inform the preventative measures to adapt so that they would not miss this type of Sybil attack in future.
- Recovery methods should first isolate the area around the Sybil attack, then use smaller granularity to remove only the Sybil nodes logically and eventually use self-healing techniques, if possible, to improve network connectivity for nodes that are in the vicinity of the attack. This stems from isolation recovery techniques being the most stringent and self-healing to be the least stringent, i.e. an implication of requiring the WSN to fail securely.

10.2 APPLICATION RISK ANALYSIS

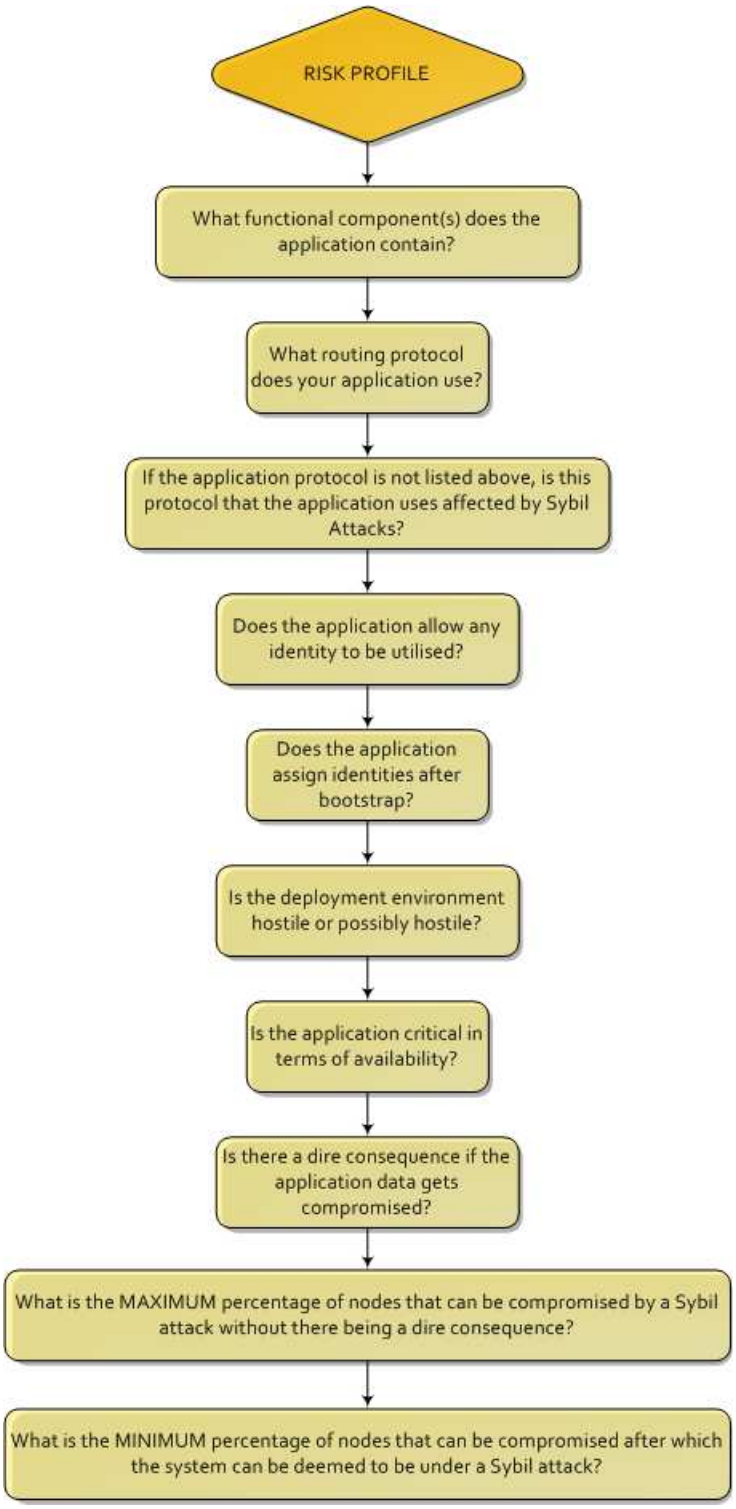

Figure 10.1: Risk Profiling Framework

In order to achieve the risk profiling of the application, a risk analysis framework was developed. By completing the questions in this framework, the designer would assist in effecting the best fit computation. Figure 10.1 contains the individual questions for the

framework and the discussion that follows presents descriptions and where possible, expected answers. As stated earlier, the higher the risk profile, the more stringent the requirements for security ought to be. The general rule obeyed by this risk analysis framework is that if a certain property of the application increases the opportunity of Sybil attacks being launched against the system, then this property would increase the risk profile of the application irrespective of whether it has positive connotations associated with it in terms of functionality.

There are known application functionality types that are susceptible to Sybil attacks. These can broadly be listed as distributed storage, data aggregation, data dissemination, voting, fair resource allocation, misbehaviour detection, trust-based schemes and fault-tolerant schemes [2, 6 and 7]. Often, applications may utilise a composite of these functionalities in their normal operation. In doing so, however, the application is effectively much more vulnerable to attack. This risk analysis framework allows multiple options to be selected and increases the risk that is scored accordingly.

As stated earlier, there are some routing protocols that are prone to Sybil attacks because of protocol design and operation considerations. These protocols are TinyOS Beaconing; Directed Diffusion or its multipath variants; Geographic Routing (i.e. GPSR, GEAR); Rumour Routing and Energy Conserving Topology Maintenance protocols (i.e. SPAN, GAF, CEC, AFECA) [5]. The framework also makes provision for routing protocols that are not listed here but could be vulnerable against Sybil attacks. However, this relies on the designer being cognisant of this inherent weakness.

A simple but effective design consideration for applications is to use identity domain optimisation: simply put, reducing the number of possible identities that can be used to only those that are currently required. If this range is left completely open, there is a greater risk of Sybil attacks being launched.

It is quite common for networks to increase in size after the initial deployment. However, keeping this option available effectively allows malicious attackers to introduce new identities owing to the expectation of already deployed member nodes. It is not always acceptable to disallow new nodes from entering the network. However, this does increase the risk profile of the system.

Some WSNs are deployed in hostile environments where the possibility of being attacked is quite high. Actually, if the designer is not quite confident about knowing whether the deployment area is friendly; he/she should expect and design for a hostile atmosphere.

Depending on the reason for the application, it may require high availability or the system could withstand nodes that are unable to transmit/receive data. Sybil attacks, being DOS attacks, when launched successfully can hinder or even stop the network from performing its usual operations. If high availability is a necessity then the risk for the system is significantly increased.

The confidentiality of data that are passed between the nodes of the network is another consideration for rating how risky a system should be deemed to be.

The risk analysis framework allows the designer to enter a range of values for the two considerations, i.e. the maximum nodes that are allowed to be compromised by a Sybil attack, as well as the minimum number of nodes to be compromised before the system declares a Sybil attack. It is not always necessary for low or medium risk systems to declare a Sybil attack when only a few nodes are compromised. A very important constraint to remember is that the minimum number of nodes that is allowed to be compromised for an attack declaration cannot be more than the maximum number of nodes allowed to be compromised.

After completing the risk analysis via this framework, the application is rated based on its risk-related designs and the score then determines the risk profile of the application. The actual ranges for classifying what risk score range corresponds to which risk profile can be predetermined by the designer. For example, one can use the criteria in Table 13.1.

Table 10.1: An example of a Risk Analysis Classification

Risk Profile	**Risk Analysis Score Range**
Low Risk	Score <= 50%
Medium Risk	50 < Score <= 70
High Risk	70 < Score <= 95
Ultimate Risk	Score > 95%

CHAPTER 11 APPLICATION BEST FIT COMPUTATION

This chapter gives insight into the process of taking requirements from the application designer and assisting him/her to select the most suitable countermeasure i.e. Best Fit Selection. A questionnaire-like table is answered by the designer and the answers provided are mapped to evaluated countermeasures thereby making it possible to optimally select a countermeasure.

To assist the designer/developer of an application to optimally select a countermeasure against Sybil attacks, the criteria for application requirements are mapped to countermeasure functionality and countermeasure specifications, taking into consideration the risk profile. It must be noted here that even though a countermeasure may be evaluated and deemed to possess high scores for functionality by the countermeasure evaluation framework, this countermeasure may not be best suited for a specific application's requirements; hence the necessity for best fit computation. The application requirements are obtained by the application designer answering a structured questionnaire that is dependent on the risk profile (see Table 11.1).

Table 11.1: Best Fit Computation Questionnaire

GENERAL
How many kilobytes can each node have for a Sybil countermeasure with regards to Code Space?
How many bytes can each node have for a Sybil countermeasure with regards to Data Size?
How much of latency (milliseconds) can be tolerated for security? Note that this is the increased latency that the entire network will see.
Does the application use a homogenous topology?
Is the density of the deployment of nodes high?
Does the application require the security to be scaled down if energy is low on nodes?
Does the application require high standards for security primitives? (only answer yes, if absolutely necessary)
Does the application require high performance? (only answer yes, if absolutely necessary)
Does the application need an event log for security breaches?
Does the application routing mechanism use multi-hop techniques?
PREVENTION

Is there a likelihood of being attacked by a Sybil attack during bootstrapping?
Does the application have a cluster head or base station that can act as a central trusted authority?
Does the system have more than one CA?
Are the nodes capable of handling cryptographic binding?
Is re-keying important?
Is network connectivity more important than the strength of the keys?
Is it acceptable to have human intervention for the operation of the WSN?
Is the BS/CH/CA secure (both physically and cryptographically)?
Does the application require secure communication between member nodes?
Is it possible for malicious attackers to read the contents of a compromised node?
DETECTION
Is it feasible for all the nodes to be equipped with range-based sensing capabilities?
Is it feasible for all the CH nodes to be equipped with range-based sensing capabilities?
Does the routing protocol use time stamps of individual node communications?
Is it likely that two nodes are right next to each other after deployment?
Is it highly possible that LOS between nodes would be obstructed?
Are all the member nodes of the same hardware type?
Is the temperature of the deployment environment either extreme or rapidly fluctuating?
Is there a high likelihood of Radio Interference in the WSN environment?
Does this application already use localization in its calculations?
Is this scheme accurate enough to be used to determine if a node's identity is true or false?
RECOVERY
Is key revocation important?
Is the availability of network services vitally important after/during a Sybil attack?
Is it feasible not to reverse incorrect classifications of nodes as Sybil nodes?
Should there be a threshold reached before wrong classifications are rectified?
Is a highly accurate area designation needed when isolating the network around the Sybil attack?
Does the system have isolation domains?
Does the application make provision for holding a list of trusted nodes in the member nodes memory?
Can the routing protocol handle route discovery?
Does the solution use node classes for BS/CHs?

Does the application hold exemption lists?

Once the risk profile has been determined using the risk analysis framework, the designer would be required to complete the necessary components of the above questionnaire according to the risk classification. For example, if the risk profile for the application was deemed to be a medium risk application, then according to the risk classification, this application requires prevention and detection countermeasure mechanisms. These components (i.e. prevention and detection sections of the questionnaire) as well as the general component (this section would be required by all applications irrespective of its risk profile) would be presented to the designer to complete.

There is a one-to-many mapping of the questions in the best fit computation questionnaire to the criteria in the countermeasure evaluation framework. Moreover, the mapping has either a positive or negative relationship in order to score the degree of suitability of the countermeasure to the application requirements. To facilitate simplicity for this mapping, a best fit matrix is used as the approach. For example, under the prevention section of the questionnaire the designer is asked, “Is it acceptable to have human intervention for the operation of the WSN?” This relates to two criteria in the evaluation framework, i.e. human intervention in both “trusted certification” and “symmetric approaches”. Since both of these criteria in the framework have the same type of connotation (i.e. human intervention is inferior to automation) for the best fit questionnaire, they are regarded as having positive relationships. The score for this question is then the algebraic sum of the scores for the respective properties (in this case, the positive sum of the scores of these two questions from the countermeasure evaluation framework).

Table 11.2: Possible answer for questionnaire

Question from Questionnaire	Answer
Is it acceptable to have human intervention for the operation of the WSN?	Yes

Table 11.3: Hypothetical score for criteria in evaluation framework

Specific Functionality	Question	Answer	Score		
			Actual	Max	Min
Trusted Certification	Is the handing out of certificates a manual process (human intervention needed) or can it be automated completely by the CA?	Automatic	2	2	1
Symmetric Approaches	Can new valid nodes entering the system be done dynamically or via manual/human intervention?	Human	1	2	1

From the tables above (i.e. Table 11.2 and Table 11.3, where hypothetical values are provided), the score for this question (i.e. the question in the best fit questionnaire) would then be:

$$Score = \frac{(\text{Actual Score 1}) + (\text{Actual Score 2})}{(\text{Max Sum})} = \frac{(2)+(1)}{(4)} = 0.75 \quad \textbf{(11.1)}$$

Each countermeasure in its category (i.e. prevention, detection and recovery techniques) is similarly rated according to every question in the questionnaire. By choosing the highest scored result from the best fit computation exercise, the designer would be able to choose the most suitable countermeasure in its category (i.e. the most suitable from all the countermeasures that have been rated by the evaluation framework).

CHAPTER 12 THE PROOF OF CONCEPT TOOL

Chapter 12 documents details about the POC tool that was developed to illustrate the validity of the frameworks. The tool was not intended to be a finished product and as such did not have fully developed possible answers. However, the results of evaluated countermeasures are presented in Chapter 13. Discussions on weightings, confidence indicators (techniques used in the tool) and user bias are offered in a subsection of this chapter.

To improve the framework as well as prove that the developed frameworks can be implemented a POC tool was developed based on the frameworks and taxonomy that were discussed in the previous chapters. The tool made provision for all six sections of the evaluation framework i.e. general overheads, specific overheads, performance, security primitives, general functionality and specific functionality. The tool also maintained countermeasure evaluations to facilitate the library of rated countermeasures that was mentioned earlier.

A user would be allowed to evaluate a countermeasure and add its ratings to the library thus enabling this countermeasure to be compared to other Sybil countermeasures. The tool graphically compares evaluated countermeasures based on various criteria; examples of these will be presented in a later chapter.

Risk profiling is also done based on the risk analysis framework but the user has the option to either upgrade or downgrade this recommendation if he/she deems it necessary.

After confirming the user's risk profile, he/she is prompted for criteria that would enable the tool to assist with best fitting of countermeasures. Based on these requirements, for each mechanism (i.e. prevention, detection, or recovery techniques depending on the risk profile) the user would then be presented with a prioritised list of countermeasures. These countermeasures would be taken from the library of rated countermeasures.

12.1 POSSIBLE ANSWERS AND SCORING

It was not within the scope of this study but to make the POC tool usable, answers to the questions in the frameworks had to be developed. To ensure data quality, only validated lists were used as possible answers. In most cases, the possible answers were "low", "medium" or "high".

Also, scores in relation to the question were allocated to these answers to allow the users' answers to be rated. For instance, if "low" was favourable for a particular question, it would have been allocated 3 points whereas "high" for the same question would have been granted 1 point.

To cater for the user not knowing the answer to a particular question for the countermeasure evaluation, the option of "unknown" was also offered, the implication being that this question was effectively not taken into consideration when calculating the score. However, this does have an impact on the quality of the rating.

12.2 WEIGHTINGS AND CONFIDENCE INDICATORS

The POC tool also considers the scenario where the user places different priorities on the four main sections of the countermeasure evaluation framework, i.e. overheads, performance, security primitives and functionality. This essentially allows the user to skew the results depending on what he/she is looking for. To produce a fair comparison, however, this skew is applied to all countermeasures.

As stated earlier, the user has the option to select "unknown" for questions that he/she does not know the answer to. This question, although ignored for the rating, is used in the calculation of confidence of rating. The confidence indicator is a measure of how true the reflection of the score is to the countermeasure. There is a linear relationship between the number of questions answered successfully (i.e. the answer is not "unknown") and the confidence indicator.

Furthermore, the user has the option of selecting if the answers spring from an actual simulation or were gauged from documented (or untested) results. The POC uses the terminology of simulated results being quantitative whereas untested results are qualitative. For qualitative analysis the confidence indicator is capped at 70% as opposed to quantitative analysis where the confidence indicator is not restricted (i.e. 100%). The implication is that should a user answer all the questions without using the "unknown" option, but produce all answers through untested results, the confidence indicator would be 70%. This would allow another person to know that the score for this countermeasure is not a perfectly true reflection of how good (or poor) this countermeasure really is.

12.3 A NOTE ON USER BIAS

As a final note on the POC, it is obvious that a user rating a new countermeasure may not have the same bias that previous analysers had. This puts new countermeasures at either an advantage or disadvantage, but will always compromise the quality of the comparison of countermeasures. This in turn has an effect on the best fit computation. This bias is especially evident in questions where answers are not numerical in nature.

It should be mentioned that the bias of the user filling in scores was limited as far as possible by the use of actual numbers/percentage for metrics that countermeasures were rated against. However, where this was not possible, it is the responsibility of the user of the framework to ensure objectivity when rating countermeasures.

CHAPTER 13 RESULTS AND RECOMMENDATIONS

This chapter presents results graphically and based on the countermeasures that were evaluated (*see Table 13.1*) by the POC tool, some deductions are realised. Also, recommendations for future countermeasures are offered by noting shortfalls of existing solutions.

13.1 RESULTS

Various countermeasures were evaluated using the POC tool. Table 14.1 shows the names and the respective types of countermeasures that were rated and compared. Due to the lack of recovery techniques for WSNs available in current literature, no recovery countermeasures were evaluated.

Table 13.1: Countermeasures evaluated by the Framework

Number	Name	Type of Countermeasure
1	CRSD [10]	Received Signal Strength Indication
2	TinyECC [41]	Asymmetric Encryption Algorithms
3	Cooperative Target Location Algorithm [42]	Time Difference of Arrival
4	Angle of Arrival Localization for WSNs [38]	Angle of Arrival
5	Toward Clock Skew Based Wireless Sensor Node Services [31]	Clock Skews
6	Detection using Ranging [43]	Received Signal Strength Indication
7	Attack Detection in Hierarchical Sensor Network [44]	Current Encryption Algorithms
8	Ad Hoc Positioning [39]	Angle of Arrival
9	Key Management for Clusters [45]	Asymmetric Encryption Algorithms
10	Defending against Sybil Attacks [46]	Hash Functions

Figures 13.1, 13.2, 13.3 and 13.4 show the normalised scores for the four categories (i.e. overheads, performance, security primitives and functionality) for the countermeasures listed in Table 13.1.

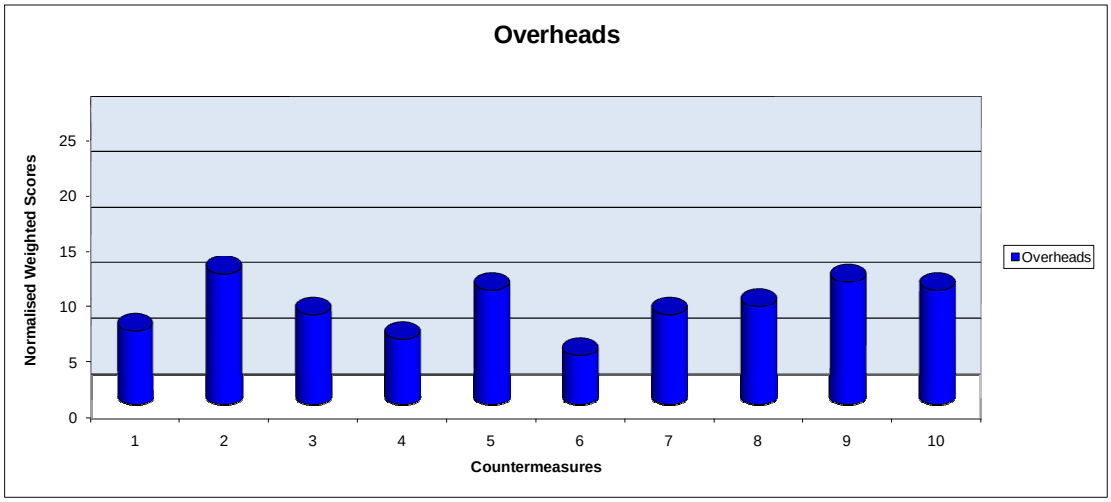

Figure 13.1: Overheads for Rated Countermeasures

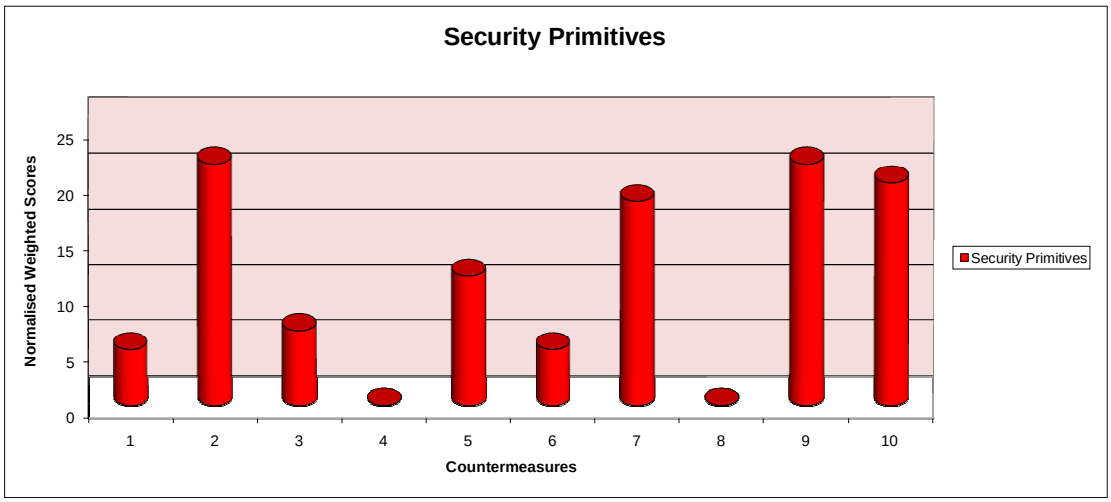

Figure 13.2: Security Primitives for Rated Countermeasures

Figure 13.3: Performance for Rated Countermeasures

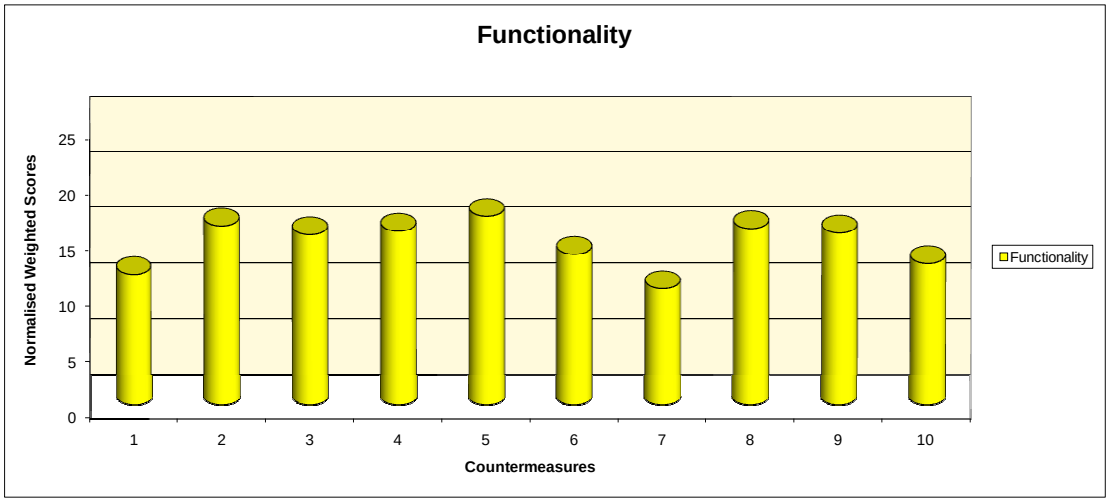

Figure 13.4: Functionality for Rated Countermeasures

Figure 13.5 and 13.6 show the overall scores for rated countermeasures. With the exception of clock skews, cryptographic techniques seem to be outperforming the other methods quite significantly.

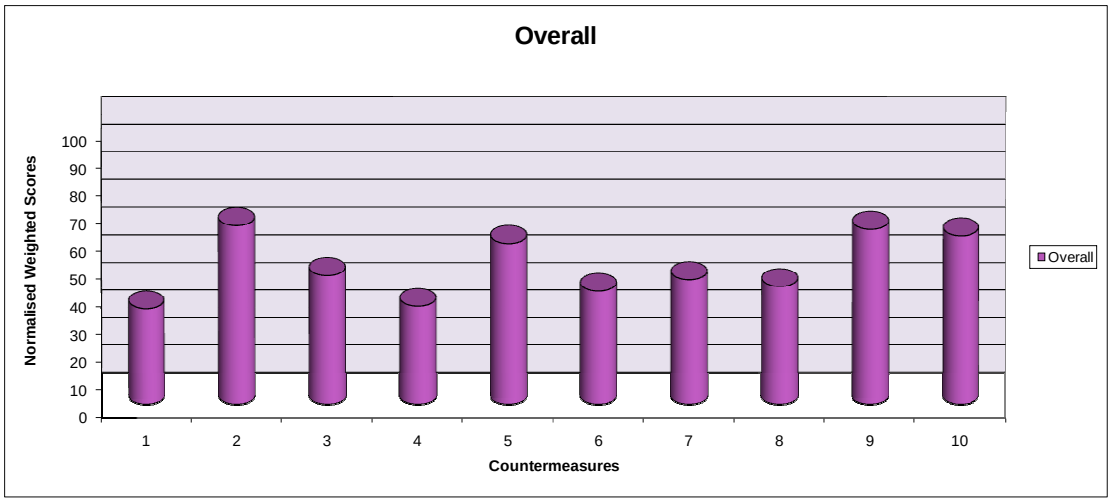

Figure 13.5: Overall Ratings for Rated Countermeasures

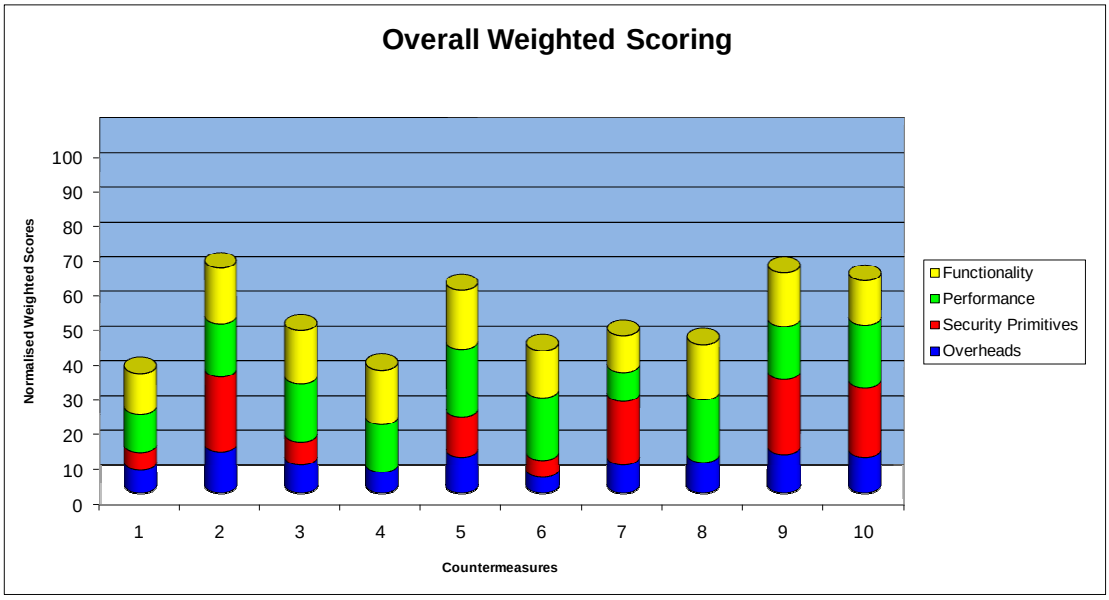

Figure 13.6: Overall Weighted Scores for Rated Countermeasures

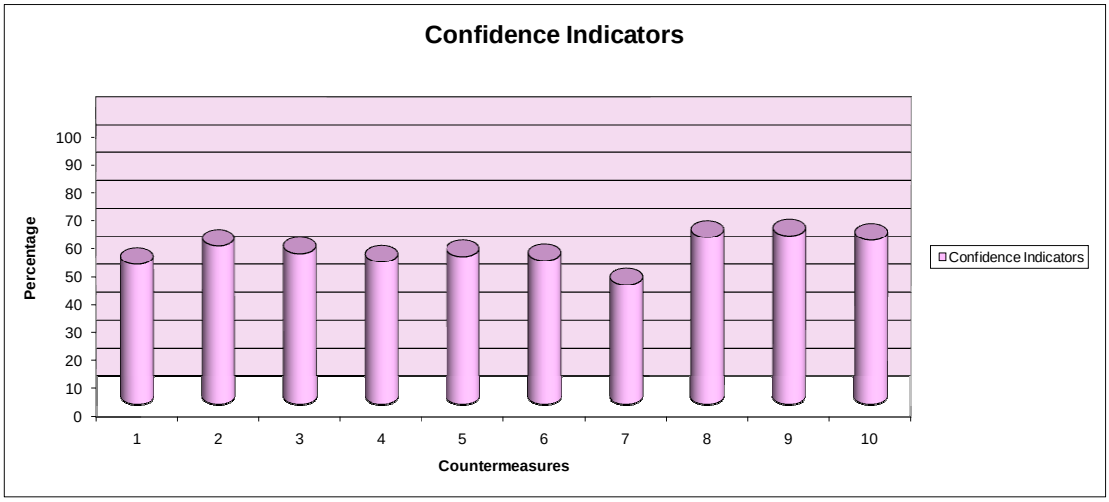

Figure 13.7: Confidence Indicators for Rated Countermeasures

As can be seen from Figure 13.7, no countermeasure rating has very high confidence indications. The highest confidence indicator was for countermeasure 9, namely 60.34%. Since no countermeasure was simulated and all results were taken from unofficial documented results, the evaluations for all the countermeasures were considered qualitative analysis.

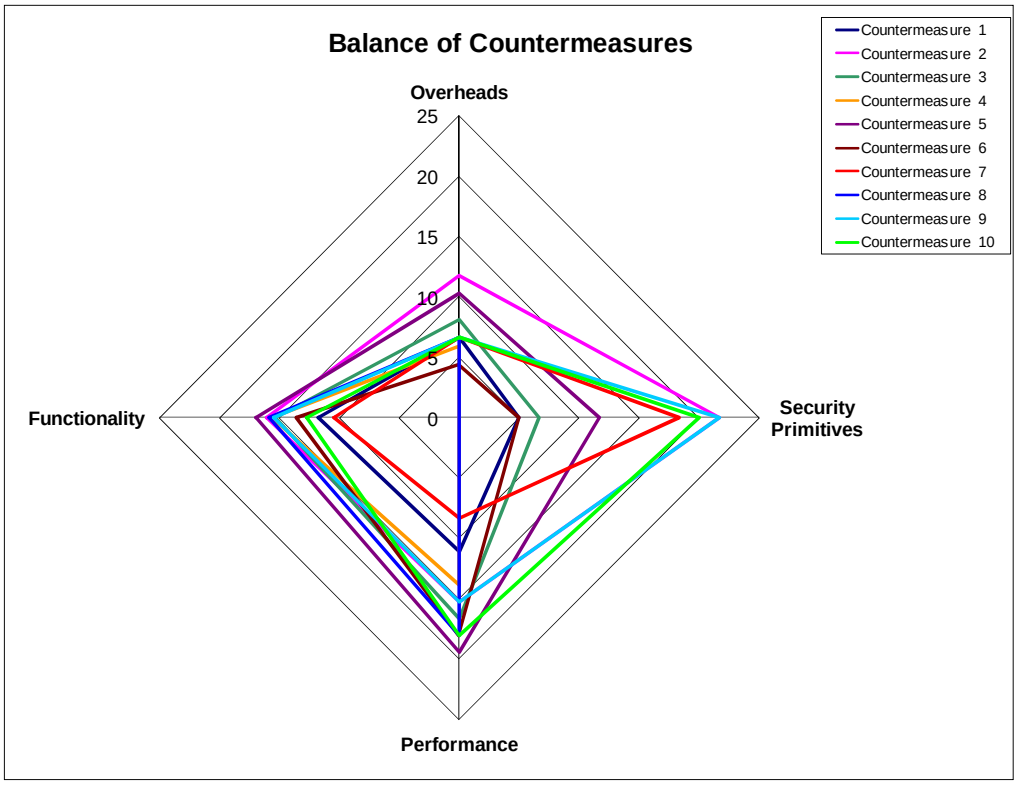

Figure 13.8: Balance of Rated Countermeasures

Figure 13.8 shows how each rated countermeasure fares in the four categories of the evaluation framework. Countermeasures 2, 7, 9 and 10 are glaringly strong in the "security primitives" category which is realistic since they are cryptographic techniques. However, just as glaring was that countermeasure 4 and 8 (both AOA techniques) had scored zero in this category.

Since this study is purely qualitative, there was low confidence in the overheads category since most documented results only published some overhead cost analysis.

For the other two categories (i.e. functionality and performance), there were no obvious discrepancies in the comparison of countermeasures.

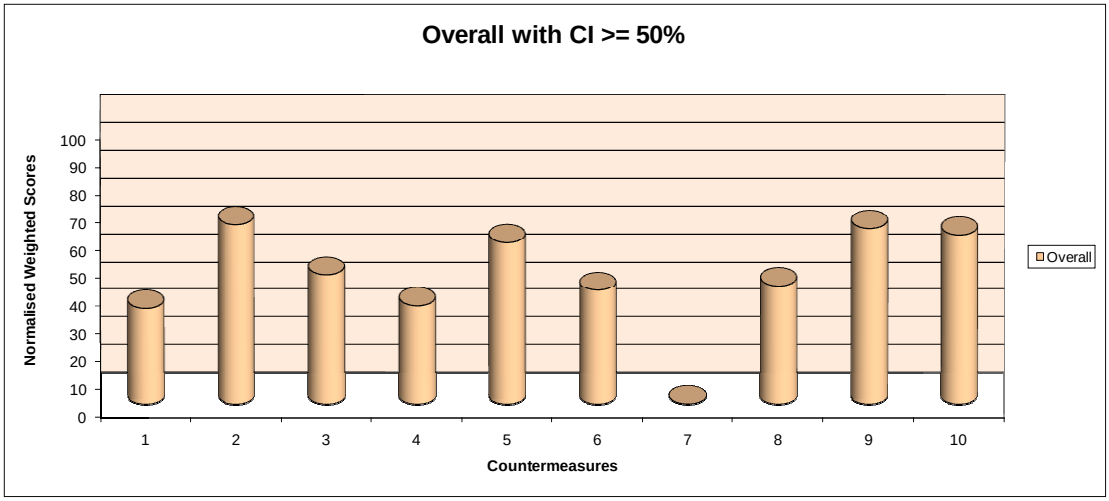

Figure 13.9: Minimum Confidence Indicator

Figure 13.9 shows the countermeasures that have a minimum confidence indicator of at least 50%. This would allow a designer to view only countermeasures with ratings that can be trusted above a specific value (in this case 50%). The tool can be customised to allow for values other than 50%.

13.2 RECOMMENDATIONS

From the results above, some deductions can be made about existing Sybil countermeasures and the way forward. It must be noted that any observation was made on results with at most a 60% confidence indication and must therefore be re-affirmed with simulated results.

As stated earlier, security primitives seem to exist strongly only in cryptographic techniques, which are prevention initiatives, whereas AOA techniques seem to fare dismally in this category. This is to be expected, since cryptography advances are not only undertaken for WSNs but for traditional networks as well. Furthermore, AOA seems to have low overheads and it makes sense for this type of technique to reinforce itself in its weaker categories. This can be said in general for all detection schemes for WSNs.

Regarding detection mechanisms, clock skews seem to be a very attractive technique.

For all countermeasures, it would be worthwhile if researchers and developers published better documented results for the overhead cost analysis.

CHAPTER 14 CONTRIBUTION AND FUTURE WORK

Chapter 14 outlines what this research has accomplished and its significance to the body of knowledge. Also suggestions for improvements to the framework are offered.

14.1 CONTRIBUTION TO THE BODY OF KNOWLEDGE

Without having a reference or methodology to compare Sybil countermeasures, every system engineer and developer would have to create this to enable a best “fit for purpose” countermeasure selection. This will prove to be quite a time-consuming task and moreover may not be consistent or extensive enough to yield accurate results. WSNs are already a relatively new field of engineering and can benefit greatly if application design and development times are reduced.

Thus far, there has been no consistent method to evaluate countermeasures against Sybil attacks in WSNs. This study, to the best of the author’s knowledge, has developed the first taxonomy for Sybil countermeasure classes and designed an extensive evaluative framework. This unique framework allows system engineers to compare countermeasures accurately in a short time, allowing them to focus their efforts elsewhere in the application development and design.

A risk analysis framework was also developed to enable a designer to select the most appropriate countermeasure by means of the best fit computation. Depending on the application’s requirements, based on some critical factors, the application can be classified into a risk profile. This distinctive characteristic allows the framework to suggest what countermeasure components the application designer ought to consider as compulsory, thus making it possible to do best fit analysis.

A POC tool was developed as proof that the framework can be implemented with realistic results. A spin-off was that a library containing previously rated countermeasures was created. This is extremely useful in the interest of reducing application development time, since system engineers would just have to look at these results to compare existing countermeasures (assuming they have been rated by the tool).

Based on the rated countermeasures’ performance, some valuable deductions about countermeasure techniques were made that can be used for future countermeasure designs. This is yet another by-product of creating the POC tool. Now countermeasure designers

can use this framework as a reference to identify critical areas of improvement for future countermeasure designs. Furthermore, this tool makes design collaboration among researchers much easier.

This study focused on Sybil attacks for the framework design. However, it can quite easily be applied to other attack types, since the principles are similar if not the same. In addition, many metrics are identified in this framework that would have to remain for other classes of attacks, so it makes sense for this framework just to be extended if need be.

14.2 FUTURE WORK

There is scope for improvement regarding the frameworks as well as the implementation of the framework. Some possible improvements are listed here:

The framework does not make explicit provision for hybrid countermeasures although these solutions are described in existing literature. Two solutions have, however, been proposed, i.e. either that the predominant part of the countermeasure be classified or that the countermeasure be split into various components and evaluated. Other methods may also be looked into.

The risk profiles were defined for the entire application, though in reality there may be cases where only certain transactions or messages are of one profile whereas the rest of the operations could be different. For instance, when distributing keys, an application could be deemed to be high risk but otherwise could be a medium or low risk application. There is some worth in investigating risk profile combinations for different types of data or messages within an application.

As stated earlier, developing answers for the framework questions was not within the scope of this study but answers were to some extent developed to make the POC tool usable. More work is required on this to eliminate the bias of the user rating new countermeasures. The study did propose that most if not all answer ranges be numerical or at least mutually exclusive to create objectivity in answers.

When rating countermeasures accurately, the need for simulated studies was already pointed out. However, to make the comparison fair, a standard set of network variables should be developed. Values for factors such as node range, number of nodes in the

network, routing protocol, interference patterns, number of base stations or cluster heads and node hardware specifications should be included as part of the standardised variables.

Lastly, the three main classifications of countermeasure classes were defined as prevention, detection and recovery and the framework rated how well a countermeasure performed in one of these spaces. Furthermore, the best fit analysis allowed the designer to select the best countermeasure in each class. However, a factor that was not considered was the integration between the three. For instance, hypothetically speaking, it could very well be that the top countermeasures in each class would yield sub-standard results when working within an application if compared to the second best countermeasures. To create a superior best fit selection, integration ought to be investigated and combined with the developed frameworks.

CHAPTER 15 CONCLUSION

Most attacks in WSNs are caused by incorrect data being distributed across the network. Traditional network security mechanisms cannot be used to counter the threats that are eminent in WSNs because of the physical constraints of these nodes.

In Sybil attacks, a type of DOS attack, a rogue node illegitimately impersonates multiple identities to represent several other nodes in a WSN, thereby degrading the performance and reliability of the WSN. Various countermeasures against Sybil attacks have been proposed by researchers and developers; however, application designers do not have a methodology to evaluate consistently which solution to implement to protect their specific application.

In this report, a framework for evaluating Sybil attack countermeasures has been proposed. It was the intention of this research to enable software developers/designers to choose a specific countermeasure against Sybil attacks based on what their goals are.

A detailed approach enabling the realisation of this benchmarking framework has been discussed. The evaluation framework based on a developed taxonomy of countermeasure classes, the risk profiling of applications as well as the best fit methodology were discussed. Results from a POC tool were also presented and some deductions and recommendations for future countermeasure solutions were given.

CHAPTER 16 REFERENCES

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramanium, E. Cayiri, "*Wireless Sensor Networks: A Survey*", Elsevier – Computer Networks, 2002, pp 393 – 422

[2] Yong Wang, Garhan Attebury and Byrav Ramamurthy, "*A survey of Security Issues in Wireless Sensor Networks*", IEEE Communications Surveys, 2nd Quarter 2006, Volume 8, No. 2, pp 2 – 23

[3] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, Andrea Passarella, "*Energy conversation in wireless sensor networks: a survey*", Elsevier – Computer Networks, 2009, pp 537 – 568

[4] Jingtao Wang, Geng Yang, Yuan Sun, Shengshou, "*Sybil Attack Detection Based on RSSI for Wireless Sensor Network*", IEEE, 2007, pp2684 – 2687

[5] Chris Karlof and David Wagner, "*Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures*", Ad Hoc Networks, 2003, pp 293 – 315

[6] Yun Zhou and Yuguang Fang, "*Securing Wireless Sensor Networks: A survey*", IEEE Communications, 3rd Quarter 2008, Volume 10, No. 3, pp 6 – 28

[7] James Newsome, Elaine Shi, Dawn Song, Adrian Perrig, "*The Sybil attack in Sensor Networks: Analysis & Defenses* ", IPSN, 2004, pp 259 – 268

[8] Tanveer Zia and Albert Zomaya, "*Security Issues in Wireless Sensor Networks*", School of Information Technologies – University of Sydney, International Conference on Systems and Networks Communication (ICSNC'06), 2006, pp 40 – 44

[9] Holger Karl and Andreas Willig, *Protocols and Architectures for Wireless Sensor Networks*, John Wiley & Sons, 2005

[10] Kui Ren, Wenjing Lou and Yanchou Zhang, "*LEDS: Providing Location Aware end-to-end Data Security in Wireless Sensor Networks*", IEEE, 2008, pp 585 – 598

[11] Xiaojiang Du and Hsiao-Hwa Chen, "*Security in Wireless Sensor Networks*", IEEE, 2008, pp 60 – 66

[12] Dr Sami S, Al-Wakeel and Saad A Al Swaleim, *"PRSA: A Path Redundancy Based Securiyt Algorithm for Wireless Sensor Networks"*, IEEE, 2007, pp 4159 – 4163

[13] Shaohe Lv, Xiaodong Wang, Xin Zhao and Xingming Zhou, *"Detecting the Sybil Attack Cooperatively in Wireless Sensor Networks"*, IEEE, 2008, pp 442 - 446

[14] Brian Neil Levine, Clay Shields, N Boris Margolin, "*A survey of solutions to the Sybil Attack*", Tech Report, University of Massachusetts, 2006

[15] John R Douceur, "*The Sybil attack*", IPTPS, 2002, pp 1 – 6

[16] Diogo Miguel da Costa, Castro Monica Oliveira, "*Thwarting the Sybil attack in Wireless Ad Hoc Networks*", Instituto Superior Technico, 2009, pp 1 – 28

[17] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, Abraham D. Flaxman, "*SybilGuard: Defending Against Sybil Attacks via Social Networks*", IEEE, 2008, pp 576 – 589

[18] Charles P Pfleeger, Shari Lawrence Pfleeger, "*Security in Computing*", 3rd Edition, 2003

[19] Yang Xia, Venkata Krishna Rayi, Bo Sun, Xiaojiang Du, Fei Hu, Michael Galloway, "*A survey of key management schemes in wireless sensor networks*", Computer Communications, 2007, pp 2314 – 2341

[20] P Szczechowiak, L Oliveira, M Scott, M Collier, R Dahab, "*Nanoecc: Testing the limits of elliptic cureve cryptography in sensor networks*", EWSN08 - Springer, 2008

[21] Erhan Kartaltepe, "*Properties of Secure Hash Functions*", Denim Group, 2010, [Online] http://www.denimgroup.com/know_artic_secure_hash_functions.html

[22] Murat Demirbas, Youngwhan Song, "*An RSSI-based Scheme for Sybil Attack Detection in Wireless Sensor Networks*", IEEE, 2006, pp 1 – 5

[23] L Eschenauer, VG Gligor, "*A key-management scheme for distributed sensor networks*", ACM, 2002, pp 41 – 47

[24] Jing Chao, Ren Xiuli, "*A novel random key algorithm in Wireless Sensor Networks*", IEEE, 2008, pp 687 – 691

[25] H Chanm A Perrig, D Song, "*Random Key Predistribution Schemes for Sensor Networks*", IEEE, 2003, pp 197 – 213

[26] Jun Xiao, Lirong Ren, Jindong Tan, "*Research of TDOA Based Self-localization Approach in Wireless Sensor Network*", IEEE, 2006, pp 2035 – 2040

[27] Zenon Chaczko, Ryszard Klempous, Jan Nikodem, Michal Nikodem, "*Methods of Sensors Localization in Wireless Sensor Networks*", IEEE, 2007, pp 1 – 8

[28] Jun Xiao, Hong Chen, Shi Zhang, "*Research of Range-based Synergetic Localization Algorithm in Wireless Sensor Networks*", IEEE, 2008, pp 2040 – 2044

[29] Jie Yang, Yingying Chen, "*Detecting Sybil attacks in Wireless and Sensor Networks using Cluster Analysis*", IEEE, 2008, pp 834 – 839

[30] Z Sheng, L Li, L Yanbin, Y Richard, "*Privacy-Preserving Location based Services for Mobile Users in Wireless networks*", Technical report YALEU/DCS/TR, 2004, pp 2684 – 2687

[31] L Yunxin and H Xiaojing, "*The simulation of independent Rayleigh Faders*", IEEE, 2002, pp 1503 – 1514

[32] Rong Peng and Mihail L. Sichitiu, "*Angle of Arrival Localization for Wireless Sensor Networks*"

[33] Gragos Niculescu and Badri Nath, "*Ad Hoc Positioning System (APS) Using AOA*", IEEE, 2003

[34] Neal Patwari, Joshua N. Ash, Spyros Kyperountas, et al, "*Cooperative Localization in Wireless Sensor Networks*", IEEE, 2005, pp 54 - 69

[35] Md. Borhan Uddin, Claude Castelluccia, "*Toward Clock Skew Based Wireless Sensor Node Services*", WICON, 2010

[36] Federal Information Processing Standards Publication 197, "*ADVANCED ENCRYPTION STANDARD (AES)*", November 26, 2001, [Online] http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[37] David J. Malan, Matt Welsh and Michael D. Smith, "*A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography*", Harvard University, IEEE, 2004

[38] N. Gura, A. Patel, A. Wander, H. Eberle, S.C. Shantz, "*Comparing elliptic curve cryptography and RSA on 8-bit CPUs",* Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems, Boston, Massachusetts, August 2004.

[39] Breeze Wireless Communications, "*Radio Signal Propagation*", online at http://www.breezecom.com, 2010

[40] Neal Patwari and Piyush Agrawal, "*Effects of Correlated Shadowing: Connectivity, Localization, and RF Tomography*", International Conference on Information Processing in Sensor Networks, 2008, pp 82 – 93

[41] An Liu & Peng Ning, "*TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks*", IEEE

[42] Pengfei Peng, Hao Luo, Zhong Liu and Xiongwei Ren, "*A Cooperative Target Location Algorithm Based on Time Difference of Arrival in Wireless Senor Networks*", IEEE, 2009

[43] Ren Xiu - li and Yang Wei, "*Method of Detecting the Sybil attack based on ranging in WSNs*", IEEE, 2009

[44] Jian Yin and Sanjay Kumar Madria, "*Sybil Attack Detection in a Hierarchical Sensor Network*", IEEE, 2008

[45] Reza Azarderakhsh, Arash Reyhani-Masoleh and Zine-Eddine Abid, "*A Key Management Scheme for Cluster Based Wireless Sensor Networks*", IEEE, 2008

[46] Qinghua Zhang, Pan Wang, Douglas S Reeves and Peng Ning, "*Defending against Sybil Attacks in Sensor Networks*", IEEE, 2005

CHAPTER 17 ADDENDUMS

17.1 A FULL EXAMPLE FROM THE POC TOOL

To illustrate how results were obtained for Chapter 13, a worked example is provided here. The countermeasure being evaluated is [45].

Table 17.1: Countermeasure Description for Worked Example

COUNTERMEASURE DESCRIPTION	
Questions	**Answers**
Name of Countermeasure	Key Management for Clusters [45]
Author(s) of Countermeasure	Reza Azarderakhsh, Arash Reyhani-Masoleh and Zine-Eddine Abid
Source/Reference of the Countermeasure	"A Key Management Scheme for Cluster Based Wireless Sensor Networks", IEEE, 2008
Confidence of Answers	Qualitative Analysis
Type of Countermeasure	Elliptic Curve Cryptography
COUNTERMEASURE METRICS	
Confidence Indicator (MAX=70%)	**60.34%**
Current Score is	**119**
Maximum Score is	**182**

As can be seen from Table 17.1, the countermeasure is an instantiation of the Elliptic Curve Cryptography type. Table 17.2 provides insight to the actual answers that were provided to questions that the framework posed to the evaluator.

Table 17.2: Countermeasure Answers for Worked Example

CODE	QUESTIONS	ANSWERS	SCORES	MAX
1	How much of latency is introduced by this countermeasure?	Unknown		5
2	How many extra Joules of energy does the application require due to the countermeasure (i.e. per transaction)?	Unknown		5
3	How much does countermeasure impede the throughput of the system measured at a sink node just before the BS?	Between 8 kbps and 12 kbps	4	5
4	How much of code space (program memory) does the countermeasure require?	Between 384. kB and 51.2 kB	3	5

5	How much of data space (RAM) does the countermeasure require?	Between 1228.8 B and 1638.4 B	3	5
6	What type of topology does the countermeasure address?	Both	2	2
7	How many beacon nodes or cluster heads per cluster does the solution require? (If the topology is non-cluster based, then 1 for the BS)	One	4	4
8	How well does the solution cater for bootstrapping vulnerabilities?	Medium	2	3
9	How generic can the individual nodes be?	Generic nodes	3	3
10	How easy is the countermeasure to implement?	Medium	2	3
11	What transmission range does the countermeasure require for Cluster Head nodes? (i.e. single hop to all nodes or multi hop to all nodes?)	Multi Hop	2	2
12	How well is Authentication catered for by the countermeasure?	Strong	3	3
13	How strong is the Authorisation capability that the countermeasure provides?	Strong	3	3
14	How effective is the Data Confidentiality capability provided?	Medium	2	3
15	How effective is the network service Availability during a sybil attack with the introduction of the countermeasure?	Medium	2	3
16	How reliable is the Data Integrity capability provided by the countermeasure?	Strong	3	3
17	How reliable is the countermeasure in terms of false positives?	Medium	2	3
18	How reliable is the countermeasure in terms of Detection Rate?	Medium	2	3
19	How scalable/extensible is the countermeasure?	Medium	2	3
20	How flexible is the countermeasure?	Low	1	3
21	How compatible is the countermeasure?	Medium	2	3
22	How resilient is the application during an attack (i.e. after nodes have been compromised) while the countermeasure is active?	Medium	2	3
23	How well does the countermeasure handle Sybil attacks that use Direct Communication?	Average	2	3
24	How well does the countermeasure handle Sybil attacks that use Indirect Communication?	Average	2	3
25	How effective is the countermeasure against Sybil attacks that use Fabricated Identities? (Outsider Attack)	Excellent	3	3
26	How effective is the countermeasure against Sybil attacks that use Stolen Identities? (Insider Attack)	Poor	1	3
27	How effective is the countermeasure against malicious entities that present all sybil identities simultaneously?	Average	2	3
28	How effective is the countermeasure against malicious entities that cycle through sybil identities over lengthy periods of time?	Average	2	3

29	How well does the countermeasure cater for Sybil attacks during bootstrapping?	Poor	1	3
30	How well does the countermeasure cater for logging of attacks?	Poor	1	3
31	How well does the countermeasure cater for Fail-Secure?	Poor	1	3
32	How well does the countermeasure cater for graceful degradation?	Poor	1	3
33	How well does the countermeasure prevent a node from presenting a stolen or fake identity to the system during normal operation?	Excellent	3	3
34	How well is this achieved during bootstrapping of the system?	Average	2	3
35	How biased is the countermeasure i.e. does it apply the same set of rules to all entities equally?	Low	3	3
36	How many CA's need to be present?	One	2	2
37	If there are more than one CA then how effectively do these entities synchronize themselves?	Excellent	3	3
38	How securely is the certificate given to those nodes outside the CA's range?	Excellent	3	3
39	How powerful does the CA(s) need to be relative to the member nodes in terms of storage?	Medium	2	3
40	How powerful does the CA(s) need to be relative to the member nodes in terms of computation?	Medium	2	3
41	How powerful does the CA(s) need to be relative to the member nodes in terms of transmitter and receiver capabilty?	Medium	2	3
42	Are the handing out of certificates a manual process (human intervention needed) or can it be automated completely by the CA?	Automatic	2	2
43	Are lost or stolen certificates catered for?	Unknown		2
44	How accurate (based on Detection rate) is the revocation process?	Unknown		3
45	How accurate (based on False positives) is the revocation process?	Unknown		3
46	How well does the CA(s) cater for Homogeneous and Heterogeneous Topolgies?	Excellent	3	3
49	How much of prior deployment knowledge is required by the solution?	Low	3	3
50	How well is key revocation handled?	Unknown		3
51	If a node is compromised, how well is the information of other nodes kept hidden?	Excellent	3	3
52	How well is key life-cycle handled?	Unknown		3
94	How well are the public keys authenticated?	Excellent	3	3
95	How secretly are the private keys of each node maintained?	Excellent	3	3
96	If a node's secret key is stolen, how well does the system realise this?	Poor	1	3
97	How long does it take to encrypt a message?	Less than 5 seconds	5	5
98	What version of ECC does the solution employ?	EccM 2.0	2	2

100	How many bits is the key size?	Between 160 bits and 240 bits	2	3
101	How long does it take to generate the private key?	Less than 30 seconds	5	5
102	Does the countermeasure take Stack overflows into consideration?	Unknown		2

Table 17.3: Countermeasure Weightings for Worked Example

Number	Category	Weighting
1	**Overheads**	**25**
1.1	General Overheads	50
1.2	Specific Overheads	50
2	**Security Primitives**	**25**
3	**Performance**	**25**
4	**Functionality**	**25**
4.1	General Functionality (All countermeasures against Sybil attacks should cater for this)	50
4.2	Specific Functionality (i.e. Specific to how this class of countermeasures fares against sybil attacks)	50
	TOTAL	**100**

Based on the weightings in Table 17.3, this countermeasure scores an overall of 63.407%, as can be seen from Table 17.4.

Table 17.4: Countermeasure Score for Worked Example

COUNTERMEASURE CATEGORIES		COUNTERMEASURES (Numbers according to List Sheet)	
		Number	9
		Confidence Indicator %	60.34
		Confidence Type	Qualitative Analysis
		Countermeasure Type	Elliptic Curve Cryptography
1	**Overheads**		11.029
1.1	General Overheads		0
1.2	Specific Overheads		44.118
2	**Security Primitives**		21.667
3	**Performance**		15.278
4	**Functionality**		15.433
4.1	General (All countermeasures against Sybil attacks should cater for this)		26.667
4.2	Specific (i.e. Specific to how this class of countermeasures fares against sybil attacks)		35.065
TOTAL (MAX = 100)			63.407
TOTAL (MAX = 100) with CI >= 50%			63.407
TOTAL (MAX = 100) with CI >= 70%			
TOTAL (MAX = 100) with CI >= 90%			

17.2 LIST OF FIGURES

17.3 LIST OF TABLES

17.4 LIST OF EQUATIONS

$$RSSI_r = \frac{RSSI_t x G_{channel}}{d^{\alpha}} = \frac{RSSI_t x \|H\|^2}{d^{\alpha}} \quad \ldots\ldots\ldots\ldots 19$$

$$\frac{RSSI_i}{RSSI_j} = \frac{\frac{RSSI_t x \|H\|^2}{d_i^{\alpha}}}{\frac{RSSI_t x \|H\|^2}{d_j^{\alpha}}} = \left(\frac{d_j}{d_i}\right)^{\alpha} \quad \ldots\ldots\ldots\ldots 19$$

$$DR = \frac{(\text{number of Sybil nodes detected - number of nodes incorrectly detected})}{\text{Actual number of Sybil nodes}} \text{x}100\% \quad \ldots\ldots\ldots 26$$

$$FN = \frac{(\text{number of non-Sybil nodes incorrectly detected as Sybil nodes})}{(\text{number of Sybil nodes detected})} \text{x}100\% \quad \ldots\ldots\ldots\ldots 27$$

$$FP = \frac{(\text{number of Sybil nodes incorrectly detected as non-Sybil nodes})}{(\text{number of Sybil nodes detected})} x100\% \quad \ldots\ldots\ldots\ldots 27$$

$$Score = \frac{(\text{Actual Score 1}) + (\text{Actual Score 2})}{(\text{Max Sum})} = \frac{(2)+(1)}{(4)} = 0.75 \quad \ldots\ldots\ldots\ldots 97$$