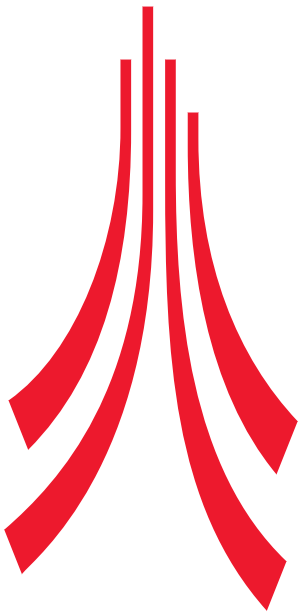


Lancaster University

# Efficient Authentication in High Security Wireless Sensor Networks

Antony Chung
B.Sc. Hons.

Final submission for the degree of Doctor of Philosophy in November 2011.
Supervised by Utz Roedig. Reviewed by Matthias Hollick and Andrew Scott.


Supporting materials are available from `http://www.tonychung.net/`.

# Efficient Authentication in High Security Wireless Sensor Networks

**Antony Chung**

Thesis submitted for the degree of Doctor of Philosophy
November 2011

## Abstract

Wireless sensor networks (WSNs) promise to greatly enhance and simplify the collection of sensor data in many applications. Individually, nodes are relatively limited, with minimalist computational power, communication bandwidth and energy. Lots of effort continues to be made to manage these limitations whilst providing a powerful overall system. Unfortunately, these limitations and solutions introduce new security challenges that must be solved. This work enhances authentication in WSNs for high security scenarios.

A review of available security solutions for wireless sensor networks found an over-emphasis on link-layer security. This is insufficient as attackers can easily imitate any node if a single key in the network is compromised. End-to-end security offers an improvement by allowing the sink to authenticate the source of a message as well as its integrity. The impracticality of using public key cryptography for all communication requires that different symmetric keys are shared between the sink and individual nodes. This can cause significant communication overhead in the network, unbalanced energy use and network lifetime reduction.

The first contribution addresses this problem with the concept of Broadcast Key Establishment (BKE). BKE allows the sink to distribute key material using a broadcast that is used to securely generate different keys on each node. The evaluation shows that this method significantly reduces overheads, extends the life of the network and causes less disruption.

The combination of wireless communication and exposed resources on nodes has resulted in new attack threats. For example, attackers can inject arbitrary messages and waste computational resources via cryptographic algorithms. The second contribution, Distance-Based Message Authentication, focuses on physical layer security to reject messages, based on distance measurement, as early as possible. Practical experiments evaluate ranging accuracy and optimisations.

This work therefore improves WSN authentication by efficiently distributing keys, for end-to-end authentication, and protects resources against depletion attack.

*A university should, I believe, provide an experience of living as well as an opportunity for learning, Without this, education is dehumanized, the student himself defrauded.*

- Albert Sloman, 1963 [1].

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

**Wireless sensor networks (WSNs) are an emerging technology with a wide variety of sensing applications. Examples include pollution monitoring, flood detection, building evacuation support and smaller networks such as body area networks on patients. The underlying scenarios vary considerably, resulting in technological differences between networks, but all are highly constrained. These constraints include limited processing power, energy availability and bandwidth. Applying existing security protocols in this environment is challenging. Worse still, new security threats continue to emerge that now target the resources directly. This chapter provides an introduction to these issues and an outline of the thesis.**

## 1.1 The Challenge of Wireless Sensor Networking

Sensor networks allow data, recorded by numerous distributed sensors, to be collected at a central point known as the 'sink'. Organisations that wish to deploy a sensor network face two challenges. The first challenge is financial; the network needs to be affordable to purchase and install in the first place. The second challenge is maintenance; the network should require as little maintenance as possible after it has been installed. These challenges are especially important if the organisation wishes to deploy a large number of nodes.

An organisation could deploy a cabled network, but this would require the installation of cables, which is not only expensive but can also be prohibitive in some environments. Without cables, nodes need to be independently powered and connected by wireless communi-

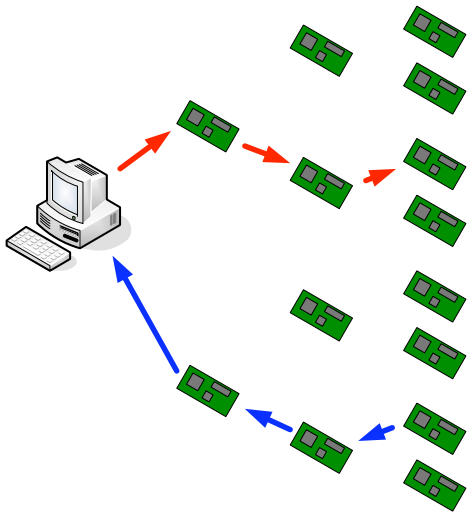


Figure 1.1: Multi-hop communication in a WSN.

cation. This could be satisfied using conventional computing equipment, but such equipment can be undesirably expensive and is not optimised for long-term battery operation; frequent visits to change batteries would be necessary. The wireless sensor network (WSN) philosophy seeks to address these issues by reducing the cost of sensor nodes and providing significant energy efficiency to allow long intervals between servicing.

The multi-hop networking architecture, where nodes forward messages on behalf of other nodes (see Figure 1.1), is found in most WSNs. This approach eliminates the need for a communications infrastructure whilst enabling the use of low-power radio communication. Energy efficiency is therefore even more important since individual nodes need to survive for long periods, not only to perform their own tasks, but also to forward messages on behalf of other nodes.

These properties have resulted in a deliberately constrained architecture with limited computational and communication capabilities. In turn, this necessitated a fundamentally different approach to hardware and software design compared to conventional computers. These system characteristics mean that conventional computer network and security protocols do not work well in WSNs. The majority of research has thus focused on addressing these differences, particularly to extend energy reserves and to maximise the efficiency of communication channels.

WSNs are now at the point where the original academic scenarios are being replaced with industrial scenarios. These industrial scenarios have greater security demands. For

example, data generated by the nodes has to be authenticated to prove that it has not been injected or modified by an adversary. In other cases there may be a need to encrypt data transmissions to protect the privacy of the sensory subjects. Key distribution has to be managed correctly in order to properly support this. As is common in many research areas, a great deal of WSN research has not considered security in detail. Existing cryptographic countermeasures have been adapted that are not matched to the WSN architecture, resulting in serious inefficiencies, infeasibilities and new attack vectors.

Whilst these security problems can be tolerated in some applications, *high security applications* are now emerging. These include scenarios where there is a real risk of an attack leading to consequences such as death, major contamination or huge financial loss. Examples include wireless security systems and industrial process control in installations such as oil refineries. The potential impact of an attack is thus very severe, providing a clear motivation for a review of WSN security and the introduction of revised and additional countermeasures.

Although a number of high security applications exist that this work is applicable to, the main application motivating this work is the use of WSN technology to provide the sensing and communication aspects in physical intrusion detection (PID) systems. One such PID system is trialled in an office complex at the University; it is discussed in more detail in Chapter 2.

In the remainder of this chapter, the limitations of WSN architecture are introduced in greater detail with an overview of the research challenges posed. The countermeasures proposed within this research are then outlined along with a summary of the contributions made into the research community.

## 1.2 Research Challenges

In high security scenarios, it is a strong requirement to authenticate the source and integrity of sensor reports. The WSN architecture presents two problems. Firstly, existing countermeasures do not work well. Second, attackers can aim to deplete resources and prevent network operation. These issues are now examined briefly.

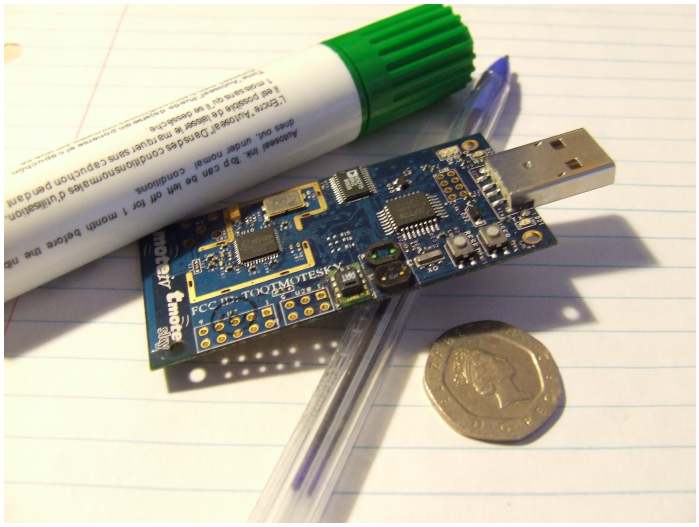


Figure 1.2: Telos Revision B sensor node (branded Tmote Sky),

The constrained WSN node architecture is a result of design goals that favour a low-cost and low-power platform. A low-cost platform means a greater number of nodes can be deployed. A low-power platform helps to reduce the need for battery changes and therefore maintenance. The constrained node architecture has a number of implications that affect all subsequent layers of design and has resulted in a radically different hardware and software approach.

A typical node is designed to operate from a pair of AA power cells that, together, provide a $3$ Volt power supply and a typical energy capacity of $1500$ mAh. The target energy lifetime is usually in the order of years, rather than the hours that might be expected from a computing device such as a laptop or PDA. Typical node hardware is thus highly constrained to reduce energy requirements; for example, the popular *Telos revision B* node design (also known as the Tmote Sky [2], see Figure 1.2) is used within this research and utilises a $4$MHz microcontroller, $10$kB of RAM, $48$kB of program memory and a $1$mW wireless transceiver. This contrasts significantly with a conventional computer system, which may have an effectively unlimited energy supply, a $2$GHz processor, $4$GB of RAM, $500$GB of program memory and a $32$mW wireless transceiver.

The resulting WSN architecture hinders WSN security for three principle reasons:

1. Security protocols require computational resources, in particular to execute cryptographic algorithms. Computational resources and operating system features are scarce in WSNs.

2. Extra communication burden is needed by security protocols to add authentication or negotiate keys. Communication bandwidth, payload length and channel availability can be very limited in a WSN.

3. The wireless communication medium provides an attack vector that is open to anybody with a transmitter.

Some WSN security problems have already been tackled with some success. Industry-strength cryptographic authentication is often based around common block ciphers and hash functions. The popular AES (Rijndael) block cipher and the SHA hash family have been implemented in WSNs for a number of years and are computationally efficient enough for regular use. A number of solutions for distributing and managing the necessary keys have also been devised. Public key algorithms that support these protocols, or can be used independently, have been optimised significantly. However, some important issues have been overlooked.

Too much focus in existing research has been on link-layer security, but in *high security* scenarios the sink needs to authenticate the source and integrity of messages on an end-to-end basis. End-to-end authentication can be useful such that the compromise of a node does not affect the connectivity of other nodes when its keys are revoked. End-to-end cryptography requires that the keys be shared differently. In the link-layer case, keys have to be established between neighbouring nodes; in the end-to-end case they must now be established such that each node shares a *different* key with the sink. A re-appraisal of key management protocols was therefore required as it was clear that there was a lack of straightforward and efficient protocols for this scenario.

Aside from the complexity of protocols, the communication overhead of the protocol has to be minimised to avoid affecting the application. When keys need to be established, a message needs to be sent between endpoints. Protocols tend to do this for every single relationship: at least one message would need to be sent for each node in the network, resulting in a bottleneck effect. Nodes closer to the sink are forced to use more resources than those further away; their energy is used more quickly and overall network connectivity is put at risk.

The first contribution of this research was to propose and evaluate the concept of Broad-

cast Key Establishment, where individual keys (shared between each node and the sink) can be privately replaced using a *single broadcast* message sent to all nodes. This better balances energy and communication overheads, reduces the load on important nodes and is much easier to integrate into existing protocols and networks than existing solutions.

Another authentication issue arises from the communications medium used. Any attacker is theoretically able to inject a packet into the network and *drain resources*. Such messages might waste communication resources if forwarded across the network before being cryptographically rejected at the destination. More seriously, the potential to waste computational resources is considerable if expensive cryptographic mechanisms are invoked. These mechanisms might be part of authentication protocols or key establishment schemes such as Diffie-Hellman. Public key mechanisms, such as modular exponentiation, are particularly troublesome on typical WSN nodes since they require many seconds of computation and introduce real time performance difficulties. An additional layer of protection would be beneficial to help protect from this new breed of *denial-of-service* threat.

Recent security developments in WSNs have embedded defences into the *physical layer*. The second contribution of this research builds on these developments by proposing and evaluating the concept of *Distance-Based Message Authentication*. This protocol involves securely measuring the distance between a sender and receiver during message exchange at each hop. If the measured distance is below an acceptable threshold, the message is accepted, otherwise it is rejected. This mechanism allows messages to be rejected more easily, thus protecting higher layers from denial-of-service attack in scenarios where attackers can be excluded from the deployment area by means of physical boundaries.

## 1.3 Main Contributions

There are two main contributions in this research.

1. The concept of Broadcast Key Establishment has been proposed, implemented and evaluated to address the inefficiency of end-to-end key establishment in high security WSNs. The initial protocol based on the Diffie-Hellman protocol was implemented in conjunction with an efficient broadcast loss recovery approach. The protocols have

been found to better balance load, extend the life of nodes closer to the sink and keep the network connected to the sink for longer. See Chapter 5 for more details.

2. Distance-Based Message Authentication has been proposed, simulated, implemented and evaluated to provide a new low-level message authentication mechanism based on secure distance measurement. The concept allows messages to be rejected based on a distance measurement threshold rather than costly cryptographic mechanisms. The initial simulation showed that the communication overhead is similar to existing MAC protocols, particularly those using long-preamble duty-cycle approaches. The issues of ranging accuracy and optimisation strategies were also investigated. Chapter 6 details this work.

## 1.4 Publications

Publications have been made on the Broadcast Key Establishment aspect of this work. Diffie-Hellman Broadcast Key Establishment (BKE/D) was first briefly proposed as *DHB-KEY* [3] at EuroSSC, then again [4] at EWSN'08. An evaluation of DHB-KEY was then carried out [5] at IEEE WSNS'08 (IEEE MASS'08).

Publications have been made on the Distance-Based Message Authentication aspect of this work. The concept was first proposed as the Round-Trip-Time Message Authentication Protocol (RTTMAP) [6] at IFIP N2S'09 with a simulation-based feasibility analysis. An implementation of RTTMAP (RTTMAP-N) using Nanotron transceivers and chirp spread spectrum was then presented [7] at EWSN'10. An evaluation of the ranging accuracy in RTTMAP-N was then conducted and optimisation strategies involving link selection were proposed and evaluated [8] at IEEE WSNS'10 (IEEE MASS'10).

Additional publications were also made to convey the research to more general audiences [9] and also in collaborative works such as IPsec 6LoWPAN [10].

## 1.5 Outline

This thesis is organised as follows.

**Chapter 2 Case Study** Presents a case study on the primary motivational application: physical intrusion detection systems. Existing systems are reviewed in terms of security. The performance and security of WSN technology is then explored using an experimental test bed.

**Chapter 3 Performance Evaluation Fundamentals** Measures and characterises the computational and communication performance of WSN technology for later sections.

**Chapter 4 Background** Provides a review of existing research and development in WSNs, and in security approaches available before this work began and developed concurrently. It motivates the need for both efficient key establishment and physical layer authentication.

**Chapter 5 Broadcast Key Establishment** Details the first contribution of this thesis: Broadcast Key Establishment. The various approaches including Diffie-Hellman Broadcast (BKE/D) are evaluated in terms of energy and communication performance to determine the benefits of the concept.

**Chapter 6 Distance-Based Message Authentication** Details the second contribution of this thesis: Distance-Based Message Authentication, more specifically Round-Trip-Time Message Authentication (RTTMAP). The implementation is detailed along with an evaluation of the performance of ranging in the real world and various measures to optimise the protocol.

**Chapter 7 Conclusion** Concludes the work with analysis and new research questions.

# Chapter 2

# Case Study

**The core application for this work is physical intrusion detection (PID). PID systems have strong security needs as the application itself is used for security purposes. Using wireless communication for PID systems carries an increased level of risk as the communication channels are exposed. This chapter presents a case study on PID systems, the usefulness of WSNs in that context and the related security challenges. The experimental test bed, deployed for evaluation purposes, is then detailed. The security analysis shows the benefits offered by the main contributions of this thesis: Broadcast Key Establishment and Distance-Based Message Authentication.**

## 2.1 Physical Intrusion Detection

*Physical intrusion detection* (PID) systems vary slightly in design and purpose, but they all share a common goal: to monitor a set of areas for changes and to report any changes to users. They generally share a common architectural design with numerous *sensors* linked to a central *control unit* (analogous to a *sink* in a WSN). The control unit is responsible for interpreting sensory events, such as a door opening or movement being detected, and raises alarms depending on its settings.

Systems mostly differ in complexity, size and context. Smaller systems, installed in many homes and businesses, provide straightforward alarm systems. These are armed by users and then triggered by sensor events. The objective is to deter burglars and then to attract attention if a burglary actually takes place. Complex systems can alert relatives, key holders

or the police using telecommunication links. Larger systems cover premises, such as airports, or even national borders, with much firmer security requirements. Active at all times, these are continually monitored by security personnel at a control station. It is common for these larger systems to be integrated with closed circuit television (CCTV), access control and fire detection systems.

Very often, PID systems are referred to simply as *Burglar Alarms*. This term is perhaps misleading for larger PID systems; however, the underlying technology, design objectives and threat model are very similar. A PID system provides three core features:

**Monitoring** A set of areas are monitored for changing circumstances using sensors. Any events or faults are reported to a control unit via a low latency communication mechanism.

**User Interface** A user interface is provided for authorised users. The user can configure settings and be notified about events.

**Tamper Detection** Tampering has to be detected or avoided.

### 2.1.1 Threat Scope

The previous section discusses the intended functionality. An attacker, by contrast, will aim to attack these features such that an attack can go unnoticed. These three main threats are:

**Avoidance** The attacker avoids detection in the first place, meaning that no attack against the system is needed.

**Physical Tampering** The attacker physically accesses and alters the system in such a way that it cannot function correctly.

**Electronic Attack** The attacker remotely attacks the system to obtain unauthorised access or inject false data.

Whilst the work of this thesis helps with all these points, the later threat is the primary focus: secure communication within PID systems; especially those used in large, industrial-scale, deployments where there is a need for high security. It is still useful to demonstrate the insecurity of the simplest systems, so these are also discussed in the coming sections.

### 2.1.2 Attacker Capability

To carry out an electronic attack, an attacker has a number of capabilities that can be used either individually or in combination. A review of the current systems and current WSN security will later relate to these capabilities. The assumed capabilities are purposefully strong as this thesis focuses on *high security applications* where attackers have strong motivation and lots of resources. The following assumptions are therefore made:

**Eavesdropping** The attacker can record all communication after the system is deployed.

**Injection** The attacker can transmit any number of messages to any node[1], on any frequency and at any power level from the outside.

**Power** The attacker has superior (but realistic) computing power and access to abundant, and portable, energy.

**Equipment** The attacker can procure and use any feasible hardware or software, including clones of reverse-engineered devices[2].

### 2.1.3 Electronic Security Requirements

To avoid these electronic attacks, a number of countermeasures are needed. The following countermeasures are either already available or need to be developed. These are based on the review of existing research work (see Chapter 4) and the review of the existing technology (see below). Some of these countermeasures are quite common (discussed by Schneier [11, p. 2] for example) whilst others are more specific to the scenario.

**Source Authentication** When the control unit receives a message it must be sure it is from the intended sensor and not injected by an attacker.

**Integrity** The control unit must be able to determine that a message sent by a genuine sensor has not been modified or replayed by an attacker.

**Trust Isolation** The control unit must be able to trust the rest of the network in the event that certain participants have become untrustworthy.

[1] whether or not that message is accepted is another matter
[2] although without confidential data, such as keys

**Jamming Detection** The system should be able to raise an alarm in the event of a jamming attack.

**Cryptanalysis Resilience** Any electronic mechanism and codes used to protect the system must be difficult to break by an attacker.

**Key Security** It must be possible for the system to replace existing keys in the network in a secure fashion.

**Tamper Detection** Attempts to physically attack a sensor must be detected so that the sensor and its keys can be revoked or other action taken.

**Resource Protection** An attack mounted by an adversary should not result in significant resource loss unless the attacker expends considerable resources in carrying out the attack.

In addition to these requirements, there are a number of requirements that must be met by any security mechanism. These are particularly relevant to the constrained wireless platform.

**Energy Efficiency** Participants that operate from batteries must not require battery changes frequently.

**Scalability** The network should be scalable without requiring additional infrastructure.

**Upgradability** The network should be capable of being reprogrammed to allow changes to protocols, security mechanisms and sensory applications.

## 2.2 Existing Systems

Until recently, the complexity of PID systems largely depended on the sensors. For example, some sensors can reject false detection caused by animals. Others implement complex algorithms to monitor intruder proximity to fences. The vast majority of existing PID systems are mains powered, perhaps with some battery backup, and use sensors that are connected with cables to the control unit. This design has been widely available for decades, but wireless systems are gaining momentum for two principle reasons.

**Expense and Complexity** Installation of cables can be expensive and difficult. Installation along, and through walls, can cause damage and breach planning laws (for example, in listed buildings). In industrial deployments, particularly near flammable substances, cables may need to comply with standards that significantly increase expense. In temporary deployments, the need for cable runs can prove inconvenient and issues such as health and safety quickly come into play. It may also be desirable to hide sensors to avoid alerting intruders to their location.

**Reliance on Cables** All cabling must remain intact, which leads to two problems. First, an attacker often need only snip wires to disable the system, or at least render it unable to distinguish between fault and attack. Avoiding this can permit 'confirmed' alarms that result in faster and more accurate emergency response. Second, cabling can be damaged by accidents and disasters. This results in false alarms, re-installation cost and business disruption. Obviously this problem can be avoided by increasing cabling redundancy, but at additional cost and complexity.

Wireless networking lends itself to this problem by introducing a communication medium that does not rely on physical links and is therefore potentially cheaper, less complex and more robust.

### 2.2.1 Strengths and Weaknesses

Some wireless systems have already been marketed and begin to address the issues outlined. The risk of *electronic attack* must be tackled because the communication channels are more easily accessed when compared to secured cables. It is therefore important to either protect against, or detect, *electronic* attacks. These safeguards vary considerably between systems.

Insurance companies and monitoring centres use a security grading system to measure these safeguards. This is included in the European standard *EN50131* [12], which defines the minimum technical capabilities that a system must satisfy. These grades scale from Grade 1, for simple deterrent systems, to Grade 4, for advanced systems used in scenarios like government installations and bullion stores. As the grading increases the assumed abil-

ities of the attacker are matched by more stringent technical requirements. For example, a Grade 4 system must detect the substitution of a device within $10$ seconds, whilst in a Grade 3 system such detection is optional. The standard does not define specific mechanisms, leaving such decisions to the manufacturer. The size of the system is irrelevant.

Cabled systems are beneficial from a communication and security point of view. They use a fixed communication structure, which is either *star* based, where all sensors have unique cables connected to the control unit, or *bus* based where sensors are connected to a common communications cable, such as in the *iD Plus* system [13]. This can be tested on deployment and usually remains reliable unless damaged. The cables can be armoured, hidden and secured using electronic techniques to detect tampering or false sensors. Unfortunately, these properties do not translate well into wireless systems. Critically, since the security grade of a system is set at the lowest grade present, hybrid systems do not benefit from cabled security if the wireless portion is less secure.

In larger scenarios, cabled systems are therefore more commonplace as they need to comply with very strong security requirements that wireless systems do not yet meet, are easier to test and remain stable once installed. This is particularly important as the requirements form a foundation for '*confirmed activations*', which are mandated by some police forces before they will respond.

### Deployed Security Mechanisms

It is generally difficult to obtain detailed standards information or implementation details of deployed security systems as many organisations hide these for security[3] or commercial reasons. Nevertheless, it is possible to determine some properties by analysing radio transmissions and evaluating published documentation, attacks and reviews. In this section, the terms *key* and *code* are used interchangeably.

Wireless systems are generally not graded to a high level because assurance is needed that a highly capable attacker cannot circumvent or manipulate the system. Although considerable effort can be made to cryptographically protect transmissions, wireless remains a significant challenge as exposure to jamming and injection attack is significantly higher.

[3] Or, as any security researcher will tell you: obscurity...

No wireless system has yet achieved Grade 4 and only one, a ZigBee and IEEE 802.15.4 based system [14], claims to achieve Grade 3. This situation has occurred since EN50131 mandates the use of message authentication codes (MACs) at Grade 3 and above, yet many systems do not use them.

At lower grades, the standard merely mandates a given number of codes and a percentage probability that an attacker will *discover* a code [12]. MACs provide a far stronger approach because they do not simply include the code as the signature directly. The code is used as a key to generate the signature, based on the content of the message; then that signature, not the code, is appended to the message. Since the key is never transmitted over the air, and the signature changes for different message payloads, attackers are forced to use a different method to compromise the key in order to spoof messages; guessing or eavesdropping is infeasible. Unfortunately, some systems do not even meet Grade 1, let alone Grade 4.

Simple wireless systems use unidirectional communication and have limited security functionality. There are three simple approaches commonly used in these systems to prevent electronic manipulation: pre-distributed network keys or 'house codes', pre-set node keys with learning mechanisms and rolling codes. These approaches have been shown to achieve varying degrees of success.

The *house code* concept is little more than a network group ID but is sometimes missmarketed as a network encryption key. This mechanism really only ensures that multiple systems in the same area can co-exist on the same radio channels, which is an approach also used in systems such as door bells and home automation. In some systems this 'key' is only $8$ bits in length and set using hardware DIP switches within the control panel, sensors, key-fobs and siren unit. Some users might think that having eight switches provides a large number of combinations, but in reality an $8$-bit code only provides $256$ different combinations. By contrast, bicycle locks often provide at least $1000$ combinations and require the attacker to be physically present to break them. Obviously, it is not ideal to protect a site with less security than a bicycle lock. Such systems do not even meet Grade 1 of EN50131, which mandates $100,000$ codes as a minimum.

Another approach is the concept of *pre-set codes*, which are usually much longer; for

example, $16$ million combinations from $24$ bits. Rather than using a network-wide code, each sensor has a unique code. These codes are either preset in the factory or reset each time the batteries are replaced. When a sensor is *joined* to the system, a *learning mode* is enabled in the control panel for a short period. Data sent to the control unit during that period allows it to recognise new codes, and thus those devices, in future. This approach helps to avoid the additional cost of bidirectional communication or user interfaces on individual sensors.

Weaker systems transmit their codes in *plaintext* alongside sensor data. Therefore, it is easy for a capable attacker to recover the code. This is part of the reason such systems do not achieve higher security grades. More advanced mechanisms, for example, message authentication codes could be used with the key; but there is still a risk of offline attack. Recovering the codes is far more feasible with $24$-bit codes than it is with proper cryptographic keys, which are usually $128$ bits or greater. Another risk is replay attack using recorded, genuine, messages.

These attacks are avoided by more advanced systems that use the concept of *rolling codes* where the codes are changed in a deterministic fashion on both sensor and control unit. If an attacker overhears a code, it is immediately useless provided the control unit has received it. Knowledge of earlier codes should not give the attacker knowledge of future codes. Unfortunately the security of this mechanism has been subject to implementation faults by various manufacturers; for example, the *KeeLoq* mechanism used extensively in car key-fobs was found to be vulnerable to attack [15]. Another issue is that the attacker can selectively jam the communications channel during a physical intrusion, capture valid codes and then use them for carefully timed false transmissions.

It is not unusual for security to be stronger in one element of a system compared to another. For example, the security provided to key-fob communication is often strong to avoid *user imitation*. But, this level of protection is not always provided for sensor communication. Therefore, an attacker is left with a harder time disabling the system by spoofing the user but he can still selectively jam sensor transmissions and inject personalised replacements.

As a deterrent, it is possible for a manufacturer to *encrypt* all communication using a site (network) key; however, this makes adding new devices harder as the site key would need to be programmed into new devices by the user. Recall that an $8$-bit key is insufficient. Large

and expensive input mechanisms are therefore needed, unless the devices can be connected to a computer. Whilst it may be tempting to use a *manufacturer key*, this approach is insecure as the keys can be extracted by reverse-engineering any available device. Although difficult to achieve for an amateur, a more determined professional might have access to the necessary equipment. Indeed, this approach was recently taken [16] against the anti-piracy mechanisms built into the PlayStation 3 games console. Whilst the PlayStation attack could have been prevented using public key cryptography, such an approach would not be workable in a security system as it is self-contained and therefore would require both private and public keys to be present.

In systems installed by qualified engineers it is obviously possible for complex configuration to be carried out, enabling procedures such as *key pre-distribution*. However, should keys (codes) need to be replaced the process has to be repeated, or new key material has to be negotiated over the network, which increases communication and, sometimes, computational *overhead*.

Then there are other problems. If an attacker can generate deliberate *false* alarms, he can force the user to either disable certain features of the alarm system or stop using it all together. For example, an attacker may be able to add his own devices during the learning period with the objective of causing false alarms. Or, he may deliberately jam the channel to force the users to disable jamming detection; a function that systems sometimes provide to avoid false alarms in the presence of strong nearby signals. This makes it easier to jam the system and inject custom messages.

### General Limitations

*Scalability* is an issue in all PID systems, but many wireless systems are less scalable than expected. Unless all the devices are within a few dozen metres, most systems do not operate. Some manufacturers have added repeater devices, but multi-hop networking is not possible as the sensors only have transmitters, not receivers. Some *mesh network* systems have emerged, such as the *Ricochet* [17] system from Texecom that can support very large-scale deployments, and enhance security by providing multiple pathways. Others rely on a *hybrid* approach where a bus cable is installed and wireless devices are added using access

points, much like the design of Wi-Fi networks.

The lack of bidirectional communication results in a transmission approach known as '*shout-and-pray*' [14] since the sensors are not able to determine if transmissions reach the control unit. This is particularly critical to real-time constraints as sensors may employ power saving measures that restrict transmission intervals, therefore resulting in high energy demand after detections or undesirable delays in alarms. There is clearly a need for bidirectional communication to help solve these problems, amongst others described earlier.

*Battery life* is of concern to manufacturers for other reasons. It is generally recognised that the transmission interval for *supervisory* or *control* messages has to be minimised as much as possible to extend battery life. This often results in transmissions occurring as little as once per hour, conflicting with the minimum specified by EN50131 and thus lowering the system grading. One particular manufacturer [18] uses separate 'wake-up', video and data channels to avoid waking idle sensors when transmitting unrelated data.

Finally, should users require new functionality, the sensors cannot normally be upgraded with new software, for example, to add new features or permit inter-operation with new devices. Complete replacement of major components, or even the whole system, is often required.

From this evaluation of physical intrusion detection systems, the following are a set of issues identified that require attention. This list is not exhaustive, particularly since security attacks evolve over time and differ between applications:

- Codes should be used as cryptographic authentication keys rather than plaintext passwords.[4]
- Keys should be set locally, either by users or automatically, rather than being set by the manufacturer.
- Key length should be sufficient to prevent brute-force attack.
- Keys should be changed regularly to avoid cryptanalysis.
- Jamming needs to be more gracefully handled.

[4]The EN50131 standard mandates message authentication codes in Grade 3 systems and above.

- Bidirectional communication is necessary.
- Multi-hop communication is beneficial.
- Real-time performance needs improvement.
- Energy efficiency is vital.
- Software upgrades are desirable for new functionality, security and protocols.
- False and inconclusive alarms should be minimised.
- 'Confirmed activations' need to be reported with high reliability.

Wireless sensor network technology can help solve these problems. WSNs are low-power and obviously offer wireless communication. However, new possibilities are added due to the re-programmability of nodes and the presence of a transceiver rather than transmitter. It is therefore easier to change security protocols, improve routing, replace keys and add other functionality. For example, multi-hop mesh networks can be introduced to overcome jamming and respond better to failures.

## 2.3 Necessary Improvements

Existing commercial systems are generally effective in energy efficiency and nodes generally last half a decade before battery changes become necessary. Some manufacturers have even designed devices such as key-fobs that do not have removable batteries and are thus replaced entirely. Energy efficiency has been a major focus of WSN research and development; an energy lifetime of several years has already been realised via use of low-power MAC protocols and minimalist architecture.

Other non-security areas are also well developed. Tamper detection of the devices themselves is well supported by manufacturers. Jamming detection could be conducted locally by sensors as well as the control unit. Scalability is a natural ability of WSNs as individual nodes can both transmit and receive, allowing the implementation of a variety of network protocols. Upgradability is also relatively straightforward as WSN node design is more general-purpose

than the average embedded system. In addition, various over-the-air reprogramming protocols have been proposed that are energy-efficient, reliable and secure. However, in other areas there are shortcomings surrounding the wireless communication medium.

Using established security principles it is possible to immediately improve wireless security, although not to an acceptable degree. Source authentication and message integrity can be provided by using message authentication codes (MACs). The necessary industrial-strength cryptographic primitives, such as AES-256, are already implemented in common transceiver chips, such as the Chipcon CC2420, and popular operating systems like TinyOS.

The main problem in WSN systems is that these primitives are used at the link-layer; messages are encrypted or authenticated for individual hops only. This approach is necessary when nodes need to co-operate to support aggregation functions, for example, but it is insufficient protection when compromise occurs. The security of multiple nodes and keys along the communication path cannot be guaranteed and compromise allows impersonation of any node in the network.

Revoking a compromised key prevents it from being used, but the lack of trust isolation results in large portions of the network becoming disconnected as that node is no longer allowed to forward messages. It is important to keep the network in operation, especially if an attack is underway, so that un-compromised nodes can still report data about the attack. Disconnecting large sections of the network is simply unacceptable. Node specific keys and end-to-end security solves this problem by isolating trust to individual end-points and permitting continued network operation even if parts have been compromised.

The next problem is that keys have to be replaced to avoid cryptanalysis and allow use of finite security mechanisms (such as counters). The existing key establishment mechanisms available for WSNs would require a large number of messages in the network, which raises a number of issues as discussed and tackled in Chapter 5. It is also not possible to use mechanisms from, for example, Wi-Fi (IEEE 802.11) or the Internet; these networks have different communication patterns, system constraints and weaknesses.

To protect resources and energy from attack, countermeasures have to be put in place that have minimal resource requirements themselves. Because WSN nodes are able to receive as well as transmit messages, attackers are provided with a potential vector to launch

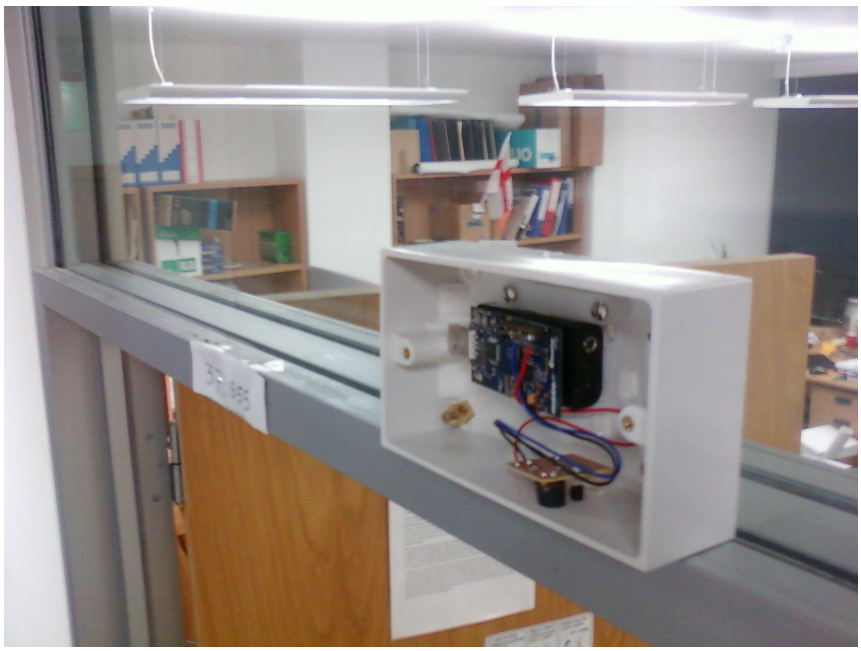


Figure 2.1: Intrusion detection node with passive infrared movement sensor.

energy-draining attacks that have been proven against some security protocols; this is discussed in Section 5.8. One way to close this attack vector is to use improved security at the physical layer, which is the subject of Chapter 6.

## 2.4 Demonstration Test Bed

A physical intrusion detection system was constructed using WSN technology to prove the concept and act as a test bed for experiments. A TinyOS application was developed for the sensor nodes such that they provide four core functions. (1) Participation in a multi-hop wireless network, forwarding messages on behalf of other nodes. (2) Management of security sensors and generation of sensor reports in response to detections. (3) Periodic transmission of *heartbeat* messages to confirm their health and connectivity. (4) Security services to provide secure authentication and key management. This application was modified to test different security enhancements.

Up to $25$ sensors were deployed at any one time in an office complex at the University. Some nodes were fitted with magnetic door contacts and others with infrared movement detectors. An example of a movement detection node is shown in Figure 2.1.

A PC, running Linux, was used as the control unit, or ‘sink’ in WSN terminology. A

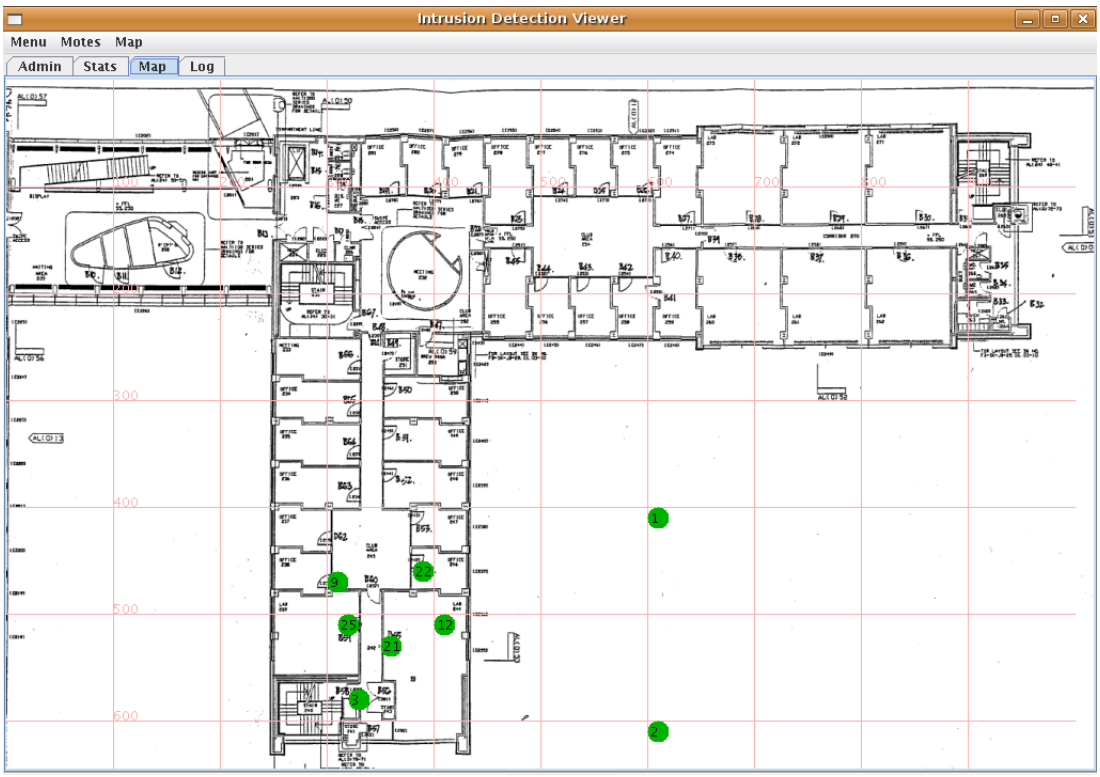



Figure 2.2: Intrusion detection software running on a PC.

node, connected using USB, acted as a network interface card (NIC). The sink software was written in Java and provided a graphical overview of the system as shown in Figure 2.2. The software shows the deployment overlaid on a site graphic and node icons flash in the event of an event or fault. Additional software components were added as needed.

The system provides two alarm types:

**Intrusion Alarm** The sink receives an intrusion detection message. This alarm indicates a *confirmed* breach of security[5]. The location of the intruder is known.

**Fault Alarm** The sink does not receive a heartbeat message for a specified period. This alarm indicates a *possible* breach of security as the alarm could be the result of jamming or have a non-security related cause. The exact location of the problem is unknown.

If an attacker needs at least a few minutes to leave a building after entering a restricted area, a delay of many seconds before reacting to a fault alarm is reasonable. Experiments (see Section 5.6) show that a wireless sensor system in a building experiences a fluctuating

[5]Under the assumption that detectors have no detection errors.

link quality and message losses occur frequently. The delay allows the system to avoid reacting to fault alarms caused by temporarily unavailable communication links. The network might recover during the delay period and the fault alarm can be cancelled.

### 2.4.1 Requirements Met

The use of the test bed and a review of existing research allowed for the requirements to be evaluated. WSNs already meet some of the requirements by implementing bidirectional communication and cryptographic algorithms, for example. However, in other areas this research has addressed shortcomings in the following ways:

**Transport Layer Authentication** The use of end-to-end authentication replaces link-layer authentication as the primary cryptographic security mechanism. A message authentication code, using a key specific to the sensor, is applied over the data and header fields. This satisfies the requirements to provide source authentication, integrity, trust isolation and revocation. An attacker cannot inject or modify a message without the specific sensor key. If the key is compromised, it can be revoked without affecting the trust of messages originating from other sensors.

**Efficient Key Distribution** Keys are changed regularly. Changing the sensor keys is necessary to achieve cryptanalysis resilience. This also covers the use of finite primitives such as anti-replay counters. However, changing the keys on nodes must be done carefully to improve energy efficiency, avoid scalability problems and because the application itself needs priority network access. Efficient key distribution has been provided by using the concept of Broadcast Key Establishment as described in Chapter 5.

**Physical Layer Security** Resource protection is provided to protect two main areas, which are energy and keys. The cryptographic functions used to authenticate messages and generate new keys can be abused by an attacker to waste energy. An attacker can waste resources by injecting arbitrary messages, which need to be processed in order to reject them. There is also a risk that keys can be stolen and used by a remote attacker to inject messages. Physical layer security prevents this by adding an additional authentication layer, which must succeed before exposing the node to these

attacks. The concept of Distance-Based Message Authentication was added to the system design to prevent the injection of messages by an attacker located outside the deployment area. See Chapter 6.

There are two areas that were not investigated in detail. Tamper detection has only been investigated in a communications sense. No effort was made to physically protect nodes such the theft of keys could be avoided or detected; however, this could be easily achieved with specially engineered packaging. Jamming detection has not been investigated, although it is noteworthy that a WSN is no worse placed to handle this when compared to existing wireless systems.

## 2.5 Summary of Findings

Wireless physical intrusion detection systems are beneficial due to the reduced reliance on cables, the ability to recover from partial connection failure, less expense in some installations and lower complexity.

Existing commercial systems use insufficient security safeguards to prevent electronic manipulation by hostiles. Some users are forced to disable jamming detection functions. Some systems use unbalanced security protection with secure key-fobs but insecure sensor communication, for example. There is also a need for significantly improved key management, with the correct type of keys and authentication protocols.

Other issues with existing systems included the lack of scalability, an absence of multi-hop networking and a lack of software update functionality. An implementation was created to show that using a WSN to solve this problem is feasible. However, the existing WSN security algorithms are not matched to this domain. Although some aspects can be modified, there is a penalty in communication overhead and there are vulnerabilities to energy-draining attacks.

## 2.6 Conclusion

This chapter has shown that existing wireless physical intrusion detection systems lack basic features such as sufficient security, scalability and re-programmability. The concept of using a WSN to solve this problem has been shown using a real WSN deployment. Problems regarding the security approach have been identified as motivation for the contributions of this thesis.

# Chapter 3

# Performance Evaluation Fundamentals

**When security protocols are applied in a WSN, they have an impact on resources. This resource consumption is linked to two key performance areas: computational performance as a result of applying cryptography and communication performance as a result of sending messages. This chapter introduces these performance fundamentals, discusses the underlying theory and performs experiments to enable the review of schemes throughout the thesis. In particular, the popular CC2420 transceiver and MSP430 microcontroller are scrutinised. The newer NA5TR1 transceiver is also investigated.**

## 3.1 Node Architecture

A conventional computer with conventional networking is simply unable to meet the minimalist WSN energy and cost requirements. Tackling this issue has resulted in a chain of fundamental redesign at almost all levels. The core requirements have lead to a hardware architecture that is highly constrained, which in turn has led to heavily constrained communication and software approaches.

The hardware design on WSN nodes[6] is driven by the demands of low power and cost. The traditional approach in embedded systems, such as PDAs, is to replace resource-hungry processors, such as those based on the Intel x86 architecture, with those in the *MIPS* and *ARM* family. One particular WSN example is the original *GumSTIX* node design; this utilised

[6]The terms *node* and *mote* tend to be used interchangeably.

an Intel XScale PXA255 [19] processor, a member of the ARM family. This delivers a favourable power consumption of around $400$mW, compared to desktop-class processors at around $50$W, but it still drains a typical battery in a matter of hours. Whilst *PIC*-class devices, such as the Microchip PIC [20], can achieve a power consumption of $1$mW or less, they are too constrained to be useful. Instead node designers aimed for the middle ground with chips such as the Texas *MSP430* [21] that provide 'just' sufficient computing capability, but with an acceptable power consumption of around $6$mW.

A similar approach was taken with the communications hardware. Rather than using high-throughput *IEEE 802.11* 'Wi-Fi' transceivers, WSN nodes tend to use lower power *IEEE 802.15.4* [22] transceivers. A popular example is the Chipcon *CC2420*[23], with a typical power consumption of 19mA when active. The penalty in lower power is primarily in bandwidth, which is only a theoretical maximum of $250$ kbps rather than upwards of $10$ Mbps possible in Wi-Fi.

Obviously WSN developers must be able to design and exchange software for this hardware with ease. To support this, a number of standardised designs emerged. Popular designs include the *Telos* [2] and *MICA* [24] nodes. These designs specify the exact hardware specification and interconnections to create a working sensor node. Several less popular designs have emerged from industry, such as the *SunSPOT* and Eneida [25] nodes. Some are more modular in nature, such as the Tyndall stack system [26] or the Coalesenses iSense [27], allowing fast prototyping with different parts like *FPGAs* and sensors. *Multi-processor* node architectures, such as in [28], have been proposed to support parallel processing and energy balancing. Smaller designs have been attempted, such as the *1-cc* [29] that fits into $1$cm$^3$ without batteries.

## 3.2 Computational Performance

The constrained hardware architecture of WSN nodes is inadequate to support 'conventional' embedded operating systems such as 'Embedded' Linux or Windows CE, for example. Programmers are thus forced to program much closer to the hardware level. This difference results in more severe effects when algorithms perform poorly. This section details these

differences and then measures the performance of cryptographic algorithms.

A new breed of minimalist operating systems were developed for WSNs, the most popular being *TinyOS* [30] and *Contiki* [31]. These lack many of the features in conventional operating systems[7], such as full multi-tasking and memory management. Additionally, applications are often compiled at the same time as the operating system, which is radically different to normal operating systems that are compiled separately.

In particular, there is a greater focus on *event-driven* and *task-based* programming rather than multi-tasking. Event-driven programming is characterised by a lack of polling in function design and a greater emphasis on interrupts and callbacks. Task-based programming schedules tasks in a queue and executes them consecutively rather than concurrently. Special programming language extensions have been developed to support this, such as *nesC* [32].

These differences mean that slow operations either block system resources undesirably or have to be split between different task invocations; a complex challenge in some cases. Such longer bursts of processing are a fundamental concern, particularly in *public key* cryptography where operations can last several seconds on constrained platforms.

Special operating system extensions have been developed to attempt to support this. Contiki has *protothreads* [33]. TinyOS has extensions to support *task pre-emption* [34] that allows low-priority tasks, such as public key computation, to be suspended when high priority tasks are scheduled.

### 3.2.1 Cryptographic Cost Principles

Raw computational energy cost can be calculated by multiplying the current drain of the microcontroller by the processing duration of the relevant algorithm. Different algorithms may use different microcontroller features resulting in differing current drain, but this has been abstracted for simplicity reasons. This section first provides the formulae to generate the data shown in Table 3.1. The variables in Table 3.2 are then populated with values obtained from the data sheet of the microcontroller and timing experiments with the algorithms themselves.

This thesis utilises three classes of cryptographic function: encryption, hash and public

[7] Arguably this means they are not 'proper' operating systems.

key. The delay for each must take two issues into account. (1) The overhead delay of the function and (2) the length of any input data. The input data length is irrelevant for some operations, but the length of the input data in the case of encryption and hash functions can obviously change.

| Purpose | Variable | Units |
|---|---|---|
| Cost of hashing or encrypting one byte | $e_e$ | mAs |
| Cost of hash/cipher function overhead | $e_g$ | mAs |
| Cost of hashing or encrypting one message including overhead | $e_E$ | mAs |
| Cost of scalar point multiplication | $e_M$ | mAs |

Table 3.1: Computational performance evaluation output variables. Note the use of capital letters to indicate that the encryption cost applies to a whole message, rather than just a single byte.

| Purpose | Variable | Units |
|---|---|---|
| Average current drain of microcontroller when active | $m_a$ | mA |
| Processing duration for hashing or encrypting one byte | $d_e$ | sec. |
| Setup duration for hash/cipher function | $d_g$ | sec. |
| Processing duration for scalar point multiplication | $d_m$ | sec. |

Table 3.2: Computational performance evaluation input variables.

### Public Key Operations

The *scalar point multiplication (SPM)* is the most expensive operation required by the elliptic curve cryptography functions found in this thesis. The expensive function found in traditional public key algorithms is *modular exponentiation*. Although the input data is of a fixed length, the values can have some bearing on the execution time. A well-designed cryptographic function will have a fixed execution time, for different key material, to avoid *side channel* attack.[8] However, the processing duration $d_m$ is specific to the configured key length and curve parameters. Thus:

$$e_M = m_a d_m \tag{3.1}$$

[8]Where an attacker can detect the duration, or other property, of the computation to shorten an attack.

**Encryption and Hashing**

The total cost $e_E$ to encrypt or hash a whole message can be obtained by adding an overhead cost $e_g$, for setup functionality, to the main cost for processing bytes. The main cost is derived from a per-byte cost $e_e$ and the length of the message $n$.

$$e_E = e_e n + e_g \tag{3.2}$$

The overhead cost $e_g$ is calculated based on the overhead delay $d_g$ multiplied by the energy drain $m_a$ of the microcontroller:

$$e_g = m_a d_g \tag{3.3}$$

The per-byte cost $e_e$ is derived from the per-byte delay $d_e$:

$$e_e = m_a d_e \tag{3.4}$$

Care must be taken since some functions work in terms of *blocks* and there may be no performance gain from encrypting or hashing a partial block. Therefore the length of the message $n$ may need to be rounded up as appropriate.

### 3.2.2 WSN Microcontroller Performance

The above equations require the energy drain of the microcontroller and the relevant delays. The energy performance of the microcontroller can be derived from its data sheets; this is shown in Table 3.3. Although data is available on the performance of cryptographic functions on WSN platforms, such as by Granjal et al. [35], newer developments have led to faster performance that must be considered. Reference implementations were therefore required to obtain the computational delays, and in turn, the energy consumption figures. These were tested on the Tmote Sky platform to obtain performance on the *MSP430* at $4$ MHz.

As the SHA and AES algorithms are so fast, the accuracy of software time measurement is an issue due to software delays. Therefore, the timing measurements used an oscilloscope to measure general-purpose lines that were flipped between each operation. The

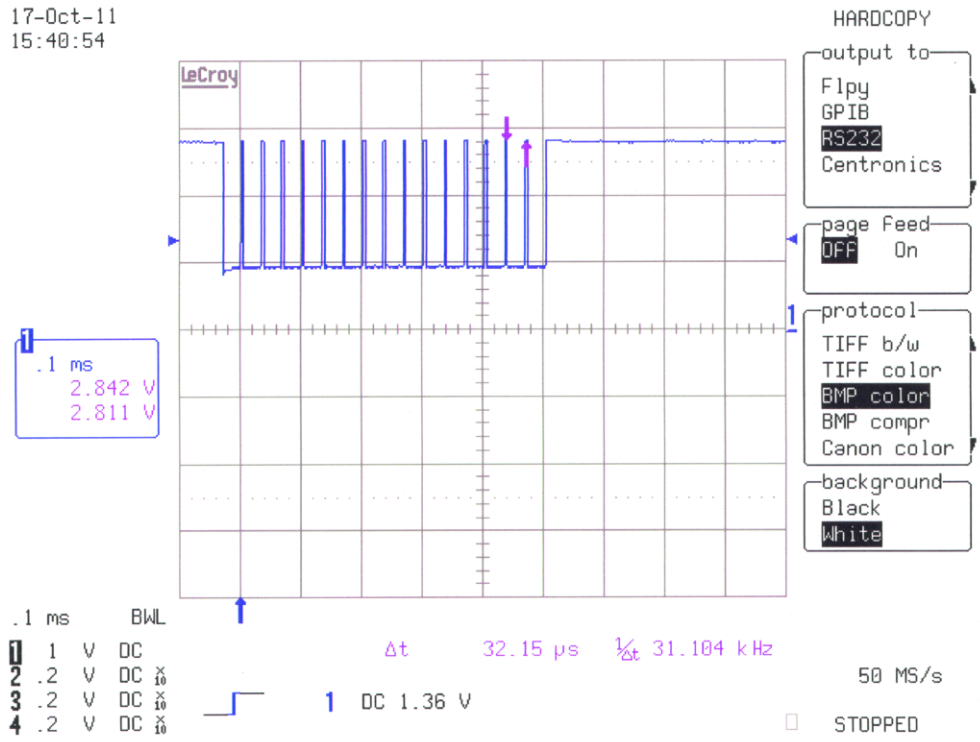



Figure 3.1: Oscilloscope output showing the delay ($d_e = \Delta t = 32\mu$s) for the SHA-256 algorithm to process a byte. $16$ bytes are shown in the output.

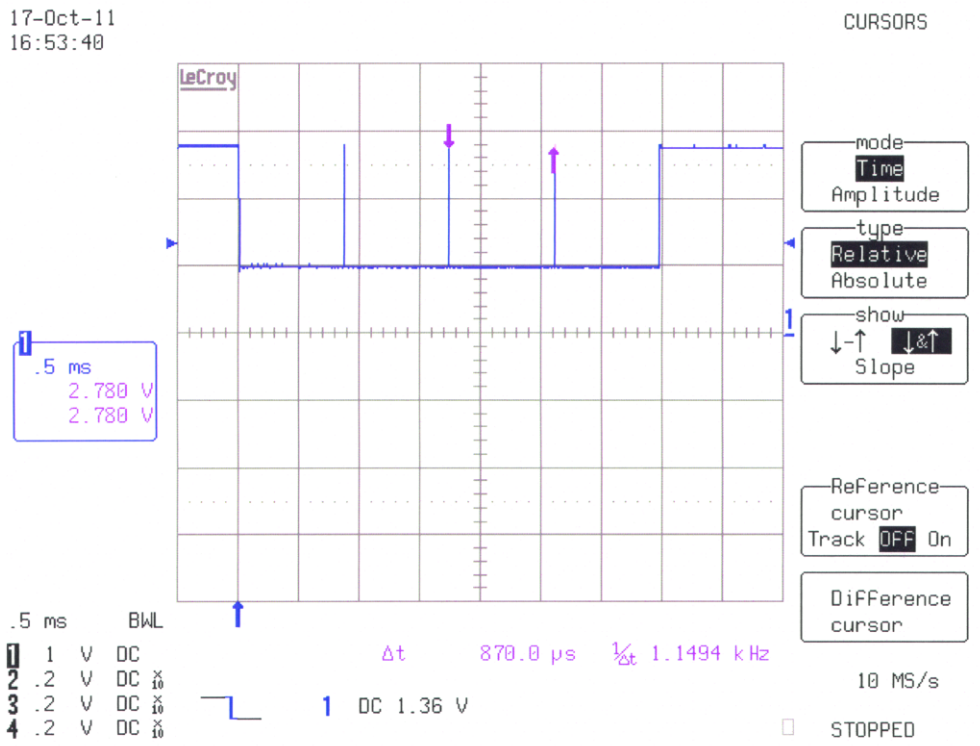



Figure 3.2: Oscilloscope output showing the delay ($d_e = \Delta t = 870\mu$s) for the AES-128 algorithm to encrypt a block of $16$ bytes. $4$ blocks are shown in the output.

time taken to process each byte in SHA-256 is $32\mu$s, shown in Figure 3.1[9]. The time taken to encrypt each block in AES-128 is $870\mu$s, shown in Figure 3.2. To check the results, the operations were then run multiple times and less accurate timer functions from TinyOS were used to confirm the measurements. Since it is obviously impossible to test every combination – it would constitute a brute force attack – the correct output of the functions was verified using relevant *test vectors*. For example, the *AES Known Answer Test* (KAT), from NIST, contains sample input and output block values for given keys.

The *MIRACL* [36] library provides an efficient implementation of the most recent cryptographic functions. It supports elliptic curve public key cryptography, symmetric ciphering using AES and the SHA hash family.

The performance of these is shown in Tables 3.4, 3.5 and 3.6. For convenience the energy costs have also been shown. These values form the first findings of this thesis and are used as part of the evaluations found in Sections 5.5, 5.7 and 6.7.

*AES* is a block cipher that encrypts, or decrypts, in blocks of $16$ bytes. It supports key lengths of $128$, $192$ and $256$ bits. In the MIRACL implementation, the functions have to be initialised and then each block is processed individually. If AES is used purely in *electronic code book* (ECB) mode then this initialisation function need only be called when keys change since no data from previous block operations is used in subsequent operations. Thus the overhead per message is zero ($d_g = e_g = 0$). In the event that initialisation is required, an overhead delay is incurred as shown in Table 3.5.

As AES operates in blocks, the time taken for each block operation was recorded. The values shown in Table 3.5 have to be divided by $16$ to obtain the per-byte delay; however, if the number of bytes $n$ is not a multiple of $16$, then $n$ has to be rounded up to the next multiple of $16$.

For SHA, it is always necessary to initialise the function and then the function operates in blocks of $1$ byte. However, a calculation is then made at the end, which forms the overhead. This calculation has to be repeated every $512$ bytes; therefore, the figures shown in the table can be taken as is if the message is less than $512$ bytes in length. $d_g$ and $e_g$ need to be multiplied otherwise.

For the elliptic curve scalar point multiplication (SPM), the *EccM* implementation was

[9]Not $320\mu$s, which was incorrectly shown in a former publication [6].

utilised using the *sect163k1* parameters. The EccM implementation is slow by modern standards. A newer implementation, *NanoECC* [37], is available but was not verified by these experiments.

## 3.3 Communication Performance

Transceiver performance is of concern throughout this thesis; it has a significant impact on the performance of any protocols that communicate and thus the WSN as a whole. For example, the transceiver performance is a core dependency in calculating the costs incurred if additional messages have to be sent or if messages have to be extended.

Low-power transceiver standards and chips have emerged as it is not feasible to use conventional wireless transceivers in WSNs; the required energy drain is simply too great to be supported. Communication performance is still of concern as it has a high *proportional* impact on resources, since low-power microcontrollers and low-capacity energy supplies are used in WSNs.

Rather than trying to discuss the large number of protocols, this section instead focuses on the transceiver-related costs and discusses *approaches* rather than specific protocols. Many surveys, such as by Demirkol et al. [38], detail numerous subsets of these protocols.[10]

This section first introduces the approach taken to calculate transmission cost. WSN-grade transceiver hardware is then introduced and the performance calculated using the CC2420 and NA5TR1 transceivers.

### 3.3.1 Communication Cost Principles

The energy cost at the transceiver level can be split into two subtypes: (1) baseline cost and (2) transmission cost. The baseline cost is incurred as a result of being part of the system and being able to receive messages. The transmission cost is specific to transmitting messages[11]. This thesis focuses on transmission cost as it is more influenced by the

[10] Readers are warned that many of these protocols do not consider security and have not faced proper scrutiny. The sheer number of protocols, and application cases, make overall security scrutiny and general comparison particularly difficult.

[11] This concept is analogous to the 'line rental' (baseline cost) and 'call charges' (transmission cost) of a telephone, although this evaluation obviously deals with energy instead of money.

| **Microcontroller** | $m_a$ mA | **Clock Speed** MHz | **Voltage** V |
|---|---|---|---|
| MSP430 | 1.9 | 4 | 2.3 |

Table 3.3: Microcontroller performance values.

| **Implementation** | **Mode** | $d_m$ secs. | $e_M$ mAs |
|---|---|---|---|
| EccM | SPM sect163k1 | 60 | 114 |

Table 3.4: MSP430F1611 scalar point multiplication performance.

| **Key Length** | **Init.** secs. | **Block Operation** secs. | **Init.** mAs | **Block Operation** mAs |
|---|---|---|---|---|
| 128 (MIRACL) | 0.042 | 0.00087 | 0.0798 | 0.001653 |
| 192 (MIRACL) | 0.0496 | 0.001 | 0.09424 | 0.0019 |
| 256 (MIRACL) | 0.0576 | 0.001104 | 0.10944 | 0.0020976 |

Table 3.5: MSP430F1611 AES performance (per block).

| **Implementation** | $d_g$ secs. | $d_e$ secs. | $e_g$ mAs | $e_e$ mAs |
|---|---|---|---|---|
| AES128 (MIRACL) | 0 | 0.000054375 | 0 | 0.0001033125 |
| AES192 (MIRACL) | 0 | 0.0000625 | 0 | 0.00011875 |
| AES256 (MIRACL) | 0 | 0.000069 | 0 | 0.0001311 |
| SHA256 (MIRACL) | 0.01017 | 0.000032 | 0.019323 | 0.0000608 |

Table 3.6: MSP430F1611 symmetric function performance.

communication requirements of the protocols and applications.

The objective of this section is to derive formulae that can be used to generate the performance variables shown in Table 3.7. The performance variables are all expressed in milliamp seconds (mAs). The formulae depend on numerous variables as shown in Table 3.8.

Transmission energy cost is derived from the modulation duration and required energy drain in either absolute or relative terms. The *current* drain during transmit may be the same as, or less then, that during reception; however, transceivers are rarely left active at all times, making the baseline receive cost in terms of *energy* significantly less. Low-power MAC protocols, such as FrameComm [39] and B-MAC [40], can also involve higher transmission cost due to extended preambles or repeated transmission. Therefore, this thesis uses absolute values for transmission cost.

Most modern transceivers transmit messages in frames. Each frame is made up of a number of bytes, including the message itself and protocol-specific extensions for functionality such as preambles and error handling. The raw byte costs are therefore handled first, followed by frame-specific costs.

Some MAC protocols use a different duration to transmit; for example, they may transmit for longer and this is handled below. The values are obviously specific to given transceivers and will vary with different frequencies and coding schemes.

| **Type** | **Transmit** mAs | **Receive** mAs |
|---|---|---|
| Single-Byte | $e_t$ | $e_r$ |
| Frame | $e_T$ | $e_R$ |

Table 3.7: Communication performance evaluation output variables. Note the use of capital letters to indicate that the cost applies to a whole frame, rather than just a byte.

| **Purpose** | **Variable** | **Units** |
|---|---|---|
| Current drain of transceiver when idle | $m_i$ | mA |
| Current drain of transceiver when receiving | $m_r$ | mA |
| Current drain of transceiver when transmitting | $m_t$ | mA |
| Modulation duration of one byte | $d_b$ | sec. |
| Modulation duration of frame overhead | $d_o$ | sec. |
| Transmit duration | $d_p$ | sec. |

Table 3.8: Communication performance evaluation input variables.

### Byte-oriented Cost

The cost for a single byte is useful in the event that a raw, un-framed, communications protocol is used, or if frames have to be extended or modified, as is the case in RTTMAP (see Chapter 6).

Assuming the transmitter and receiver are both active simultaneously, and return to idle when done, the cost to transmit $e_t$ or receive $e_r$ a single byte is derived from the modulation duration $d_b$ for one byte and required energy drain in the relevant state ($m_t$ for transmit and $m_r$ for receive).

$$e_t = m_t d_b \tag{3.5}$$

$$e_r = m_r d_b \tag{3.6}$$

### Frame-oriented Cost

The total cost for a frame has to take into account two parts: the frame overhead and the payload. The frame overhead duration $d_o$ is incurred by headers and preambles. The payload duration is calculated by multiplying the per-byte delay $d_b$ by the number of payload bytes $n$. The total duration is then multiplied by the energy drain in the relevant state; the answer provides the energy cost for a full-frame transmission $e_T$ or reception $e_R$. Note the use of capital letters to denote frames, rather than bytes.

$$e_T = m_t \cdot (d_b n + d_o) \tag{3.7}$$

$$e_R = m_r \cdot (d_b n + d_o) \tag{3.8}$$

### Duration-oriented Cost

In some MAC protocols, the cost of sending a frame is not related to the length of the frame, but instead on epoch duration. For example, some duty-cycled MAC protocols transmit for a fixed-length epoch whilst others transmit numerous attempts until the receiver responds.

Thus, the calculation is based on a specific duration (usually an average) rather than the message length. The simplified transmit ($e_T$) and receive ($e_R$) costs are therefore as follows.

$$e_T = m_t d_p \tag{3.9}$$

$$e_R = m_r d_p \tag{3.10}$$

### 3.3.2 WSN Transceiver Performance

The CC2420 and NA5TR1 are two low-power chips targeted at WSNs. They use a tenth of the current compared to typical low-power IEEE 802.11 'Wi-Fi' chip-sets, such as the Broadcom BCM4326 [41].

The CC2420 supports IEEE 802.15.4 and is found on the common Telos revision B nodes. The NA5TR1 is a newer design that supports chirp spread spectrum (CSS) in a non-standard variant of IEEE 802.15.4a. The NA5TR1 supports combined communication and ranging; this was important for the DBMA experiments in Chapter 6. Both transceivers operate in the $2.4$GHz band.

The low-power design of these chips has resulted in a peculiarity: transmit drain is usually lower than receive drain. Protocols that do not idle the transceiver or that are time synchronised will incur no energy overhead from transmission. However, very few protocols leave the transceiver in receive mode at all times as this is wasteful.

In order to use the equations in the previous section, is it necessary to obtain values for $m_r$, $m_t$, $d_b$ and $d_o$. Table 3.9 shows these values for the CC2420 and NA5TR1.

All current values, shown in Table 3.9, assume a system voltage of $2.3$ Volts. The modulation duration for a byte $d_b$ is derived from the raw bit-rate of the transceiver using the formula $d_b = 8\frac{1}{\text{bps}}$. The values for $e_t$ and $e_r$ have also been computed in Table 3.10.

The preamble duration is taken to include all modulation incurred *at the PHY level*, excluding the frame headers and payload. Transceiver start-up and synchronisation are not taken into account for reasons of simplicity. In the case of IEEE 802.15.4, a $6$ byte PHY (including a $4$ byte preamble, $1$ byte sync byte and $1$ length field) is followed by up to $39$ bytes in header and footer fields. This translates to an overhead of $45$ bytes. The NA5TR1 is not

IEEE compliant, but the Nanotron CSS proposals include a similar format [42]. Therefore, this comparison assumes identical byte overhead, although these are modulated at $1000$ kbps on the NA5TR1 rather than $250$ kbps as on the CC2420.

| **Transceiver** | $m_i$ mA | $m_r$ mA | $m_t$ mA | $d_b$ sec. | $d_p$ sec. |
|---|---|---|---|---|---|
| CC2420 | 0.020 | 18.8 | 17.4 | 0.000032 | 0.001440 |
| NA5TR1 | 0.002 | 34 | 30 | 0.000008 | 0.000360 |

Table 3.9: Transceiver performance values showing current drain when idle ($m_i$), receiving ($m_r$) and transmitting ($m_t$). The modulation duration for a single byte ($d_b$) and for frame overhead ($d_p$) are also shown.

| **Transceiver** | $e_r$ mAs | $e_t$ mAs |
|---|---|---|
| CC2420 | 0.0006016 | 0.0005568 |
| NA5TR1 | 0.000272 | 0.00024 |

Table 3.10: Transceiver energy performance values for receiving ($e_r$) or transmitting ($e_t$) individual bytes.

## 3.4 Summary of Findings

WSN hardware is considerably more efficient than conventional computing hardware due to the minimalist features offered, such as reduced speed. The performance of a WSN security scheme can be measured in terms of energy cost; this has to take both computation and communication into account.

The computational cost of a cryptographic function depends on the algorithms used and the energy drain of the microcontroller; the length of the message has to be carefully considered in algorithms that process data in blocks. The computational limitations in WSNs can cause difficulties, particularly with public key operations that take several seconds to execute on WSN microcontrollers. If the operating system does not support multi-tasking then nodes may not be able to meet real-time requirements during that time.

WSN transceivers are low-power, but still need to be powered down as much as possible to save energy; power-saving MAC protocols have been developed to do this, but they can increase the cost of transmission. The communication cost, at the link-layer, therefore

depends on the MAC protocol employed as well as the length of messages (or epochs), and the energy drain of the transceiver[12].

## 3.5 Conclusion

This chapter has evaluated the performance of typical WSN nodes in terms of communication and cryptography to assist with the evaluation of the main contributions of this thesis.

[12]The cost from the microcontroller is also a consideration, but for simplicity has been omitted as it drains less current than the transceiver

# Chapter 4

# Background

**Research and development in the area of wireless sensor networks has been gathering pace over the past decade. Several journals and large conferences now publish annually, and whilst industry is yet to mass-produce a generic WSN product, some elements of the research are being drawn into bespoke and application-specific products. As these industrial systems emerge, the security requirements are becoming more important and better understood. This chapter provides an overview of wireless sensor networking, with a strong focus on the security mechanisms currently available or proposed.**

## 4.1 Applications for Wireless Sensor Networks

Wireless sensor networks (WSNs) provide significantly greater flexibility and cost savings compared to cable-based solutions. This makes WSNs quite desirable in some applications whilst also enabling the conception of a new generation of applications.

Traditionally WSNs used conventional computing devices, more closely resembling laptops or PDAs, networked with existing technologies such as wireless modems, IEEE 802.11 ('Wi-Fi') or cellular telecommunication systems like GSM. Although this is workable in some scenarios, such as in short-term scientific deployments, it exposed a number of shortcomings. These networks are expensive to build and require frequent battery changes; the scalability is therefore minimal and the range of potential applications limited. Modern WSNs tackle this problem using a low-cost and low-power platform to facilitate the benefits without

those drawbacks.

WSN applications vary in ways such as size, density, data patterns and node relationship; the security requirements also differ considerably. The following are some common examples.

**Environmental Monitoring** Simple deployments gather basic data such as temperature or light and forward it to a base station. The benefit of wireless, self-powered, systems in large or sparse deployments is obvious. In the case of outdoor fieldwork, only a handful of nodes may be needed [43]. Such networks can be bespoke and use expensive batteries as there is no real economic benefit in using optimised architectures. Indeed, more practical issues, such as livestock eating the antennae outweigh the need for extreme power efficiency. Militaries and industry have shown a desire for larger scale networks, using thousands of nodes, but such systems have yet to materialise.

**Medical Body Area Networks** MBANs already exist in two converging forms. One is a wireless extension to existing equipment such as pacemakers [44]. Another is the use of body sensors in a home or hospital to provide alarms in an emergency. These networks are small, but have a greater chance of widespread take-up and thus cost-reduction and standardisation. The motivation for energy efficiency is based on the need for small batteries and difficulties in changing them, especially if embedded in the body.

**Industrial Monitoring and Control** Industrial systems exploit WSNs to reduce cabling expense and improve communication resilience. These networks involve real time requirements and harsher operating environments. Existing standards have been retrofitted with WSN technology; for example, HART has been extended to WirelessHART [45]. Research is particularly focused on integration with existing technology, such as in CONET [46], and deterministic performance, such as Ginseng [47]. Organisations are investigating standards and component reuse, such as at Thales [48] and BT [49]. There are a wide variety of potential topologies, from a mesh network in a factory to a chain topology monitoring a pipeline or railway. The term *supervisory control and data acquisition* (SCADA) is often used to refer to this overall area.

**Wireless Security Systems** Wireless security systems have grown from domestic wireless security products to those covering entire facilities. The benefits of rapid deployment, ease of camouflage, integration with existing WSN systems and the potential for continued operation whilst under attack, for example by using mesh networking, make the approach valuable. This area was discussed in detail in Chapter 2.

WSNs can also be found in a growing number of other areas, and some of these are beginning to become considerable research areas in their own right. Vehicular networks (VANETS) link road vehicles together to exchange traffic flow and safety data. Hybrid RFID systems, such as in stock control, are emerging that combine elements of WSN technology with passive RFID tags [50].

The security threats are very much application specific. There is almost no need for security in the environmental case since there is less motivation to attack the system; failure of the system is only likely to affect scientific experimentation rather than causing catastrophic failure of an industrial process. The need for security in medical networks (MBANS) is particularly important in the presence of confidential patient data and controllable actuators like pace makers [44].

In *high security* applications, the risk moves towards severe repercussions including financial loss, environment damage or loss of life. The Deepwater Horizon explosion in the Gulf of Mexico [51] illustrated the potential of an industrial incident; it led to the direct death of eleven people, cost billions of dollars, polluted a huge area of ocean and put at risk the livelihood of many residents. This thesis classes wireless security systems in the same domain since the system may protect such an installation.

The necessary minimalist approach to WSN design hinders the implementation of security mechanisms. Not only do the security mechanisms need to meet the requirements of the application, but they also need to work on the constrained platform typical in WSNs, protect from attacks against that platform and not use excessive resources. For more details about the threats and security requirements for the wireless security system, see Chapter 2.

## 4.2 Overview of Security Mechanisms in WSNs

Security in WSNs is very different from security in a conventional network. The exposed nature of the nodes, the larger size of the network, the need for extreme efficiency and the highly constrained architecture all combine to create a new security research area.

Existing security protocols for conventional networks cannot necessarily be directly applied since they were not designed for such scenarios. For example, security in an Ethernet-based network is commonly provided using IPsec or transport layer security (TLS), as well as physical security. These protocols were designed for large transfers, exploiting much larger packets compared to those in a WSN. The communication properties are also different. In a WSN, data flows predominately towards a central point using multi-hop mesh networking; in an Ethernet the topology tends to be based on dedicated backbone hardware in a star configuration. WSNs cannot be easily physically isolated either, due to the use of wireless communication; a potential mechanism to handle this is later expanded upon in Chapter 6. As WSNs tend to be application specific, a large number of security protocols have been proposed as designers tailor and optimise schemes for particular scenarios.

This section provides an overview of the current security mechanisms and protocols implemented in WSNs. The cryptographic building blocks that are commonly used for WSN security are first introduced, beginning with public key and then symmetric cryptographic mechanisms. Common applications of these cryptographic mechanisms are then explored in terms of authentication. This section motivates the need for new key management and physical layer security schemes, but it does not cover them directly. Detailed discussion of such matters follows in subsequent sections. The potential attack vectors against a physical intrusion detection system, that were covered in Chapter 2, are used as part of this review.

### 4.2.1 Public Key Cryptography

The main cryptographic requirement in a physical intrusion detection system is authentication. Authentication mechanisms are needed to allow the control unit to verify that received messages were generated only by genuine nodes and not altered whilst in transit.

From a cryptographic viewpoint, one of the most ideal methods would be to use public key

cryptography. *Public key cryptography*, and in particular the *RSA* [52] method, allows one node to send messages signed with a *digital signature*, using a private key. All other nodes can openly share the corresponding public key and use it to authenticate the messages. The combination of both keys is known as a *key-pair*. Since only the sender has the private key, no other node is able to imitate the sender. This concept eliminates overheads, such as encryption, in key distribution as the public key need not be confidential.

In the envisaged physical intrusion detection scenario, each node could be deployed with an individual private key and the public keys could be openly distributed. In the event that the private key is replaced, the public key could be sent to the sink without the need for confidentiality.[13]

The main problem with this approach is computational complexity. Classical RSA uses a process known as *modular exponentiation*, which involves the repeated multiplication of numbers. As both the multiplicands and exponents are very large, usually over $2000$ bits, conventional functions and microcontroller instructions cannot handle them; this results in complexity and a long duration, over a minute, of computation on WSN class devices. Unsurprisingly, there is an appetite to avoid the use of such cryptography [53].

A newer approach based on *elliptic curve cryptography* (ECC) was proposed by Miller [54] and then Koblitz [55]. This method allows for equivalent security to RSA, but with smaller key sizes, reduced computational overhead and better scalability when security levels rise. Expensive modular exponentiation found in classical RSA is replaced with *scalar point multiplication* (SPM).

SPM still takes many seconds to compute, even with the multitude of algorithmic optimisations available [56, 57, 58]. WSN implementations of SPM now exist, such as in EccM [59] that completes in about a minute or NanoECC [37] that is optimised to complete in a few seconds. An HTTPS stack [60] that provides a secure web server even exists.

There are some optimisations to public key operations that may reduce their security strength. One such example is the use of small exponents, such as in TinyPK [61], where the performance increase is caused by the need for fewer multiplications but reduces the security level [62].

More appropriate optimisations include transferring the computational load away from

[13] Properly exchanging new keys is a little more complex than this, but this serves for illustration.

constrained nodes. For example, some protocols perform only the cheaper public key, compared to private key, operations on the nodes [61] to reduce the computational load; however, this is not possible in all protocols. Another approach is to pre-compute some values in advance, which is the case in some ECC approaches like NanoECC [37].

Even the optimised public key implementations represent an overhead in terms of energy drain. The processing delay also represents a challenge in task-based operating systems that do not implement a process scheduler. The system may have to block whilst the calculation is completed, causing disruption to real-time performance and network operation unless handled, for example, by task pre-emption [34].

Hardware implementation is an alternative approach to solving the computational load problems. Some proposals, such as SecFleck [62], now involve the use of *trusted platform modules* (TPMs). TPMs have at least one pre-installed private key, typically of $4096$ bits or more, and highly efficient hardware functions for cryptography using that key. TPMs can be more secure as the private keys are not accessible over the communications bus. Security questions remain; the private key may be permanent and set by a potentially untrusted third party. Many computer systems now ship with a TPM, for example to support whole-drive encryption, and this gain in popularity has reduced the cost of the component. However, even this reduced cost may remain undesirable in the minimalist WSN design philosophy.

There is also another aspect about public key cryptography: attackers can inject random messages, forcing nodes to run public key algorithms in order to reject them. This represents a serious resource-drain attack on constrained platforms. If the microcontroller is normally sleeping, and only wakes for a few milliseconds a minute, an attacker can accelerate energy depletion by forcing the microcontroller to stay active for many seconds and aggravate real-time performance by tying up microcontroller resources.

### 4.2.2 Symmetric Cryptography

Symmetric cryptography provides encryption or authentication based on symmetric *block ciphers* and *hash functions*. Symmetric cryptography takes its name as both end points use the same key. Symmetric operations complete in milliseconds, on WSN class microcontrollers, rather than the seconds or minutes found in public key cryptography. This is mainly

as a result of symmetric functions being based on logical operations rather than computationally expensive mathematical functions. This helps to solve many of the processing-related issues inherent in public key cryptography. Further optimisations exist, with one example using the cryptographic functions in the CC2420 transceiver [63]. Clearly the use of symmetric cryptography is more desirable.

Symmetric block ciphers take a block (of bytes) with a key and then generate an encrypted result as an output block. Decryption is the reverse operation. The most popular block cipher is the Rijndael cipher [64], selected by NIST to become the *AES* standard. AES succeeded the *DES* standard, which had become outdated due to its small key length and advancing attacks. Other block ciphers exist, such as *RC5* and *Skipjack* that are used in WSNs as part of TinySec [65].

Block ciphers are used in a *block mode* to achieve secure encryption or authentication. Block modes divide the data into blocks, sequence the operations of the cipher and control the data passed back and forth. The later is crucial as it alters the behaviour of individual block operations, preventing the construction of dictionaries of plain text blocks and corresponding cipher text blocks that may assist in cryptanalysis. The specifics of such cryptanalysis are beyond the scope of this thesis.

As the focus of this work is on authentication, the block modes that are most interesting are those that generate *message authentication codes* (MACs). These are roughly equivalent to signatures in RSA. A MAC is a secure digest appended to a message that can only be computed by holders of the symmetric key. Any change in the protected element of the message results in a different MAC result. If an attacker tries to alter, or inject, a message, he cannot generate a valid MAC without the key and is left to try to guess a valid MAC; the probability of successful attack is very low. Two parties that wish to communicate can therefore use these MACs to assure that no other party has forged or tampered with their messages.

There are many block modes that generate a MAC. One is the *cipher block chaining* (CBC-MAC) mode used in TinySec [65], with RC5 or Skipjack ciphers, and in IEEE 802.15.4 [22] with the AES cipher. CBC-MAC is only safe if the message length is constant [65], else attackers can add additional content to the message without the key. Alternative modes have

to be considered for variable length messages.

Hash-based MAC (HMAC) functions are also symmetric, but are based solely on *hash functions*. The key is combined with the message rather than being supplied as a separate input. The hash function produces a short version of the message that forms the MAC; this is regarded as secure if it is difficult to compute *collisions*[14].

A popular example is the mandatory *HMAC-SHA1-96* standard [66] used in IPsec. This uses the SHA1 hash function in the HMAC mode. The output is truncated to $96$ bits.

Although there is a significant performance gain when compared to public key cryptography, the critical difference is that both parties must use the same key. Secure key management is therefore critical because confidentiality is mandatory. A full review of key management is handled in Section 4.3.

### 4.2.3 Cryptographic Authentication

The most common, and often only, authentication implemented in a multi-hop WSN is at the *link layer*. MACs are generated when a node sends a message; the MAC is subsequently checked by the next node in the path, before being processed or forwarded with a new MAC. This process is repeated for each hop. The network software has to provide matching keys at each end of individual links. Transceiver chips supporting IEEE 802.15.4 provide AES-CBC-MAC in hardware. Alternatively, software-based link-layer security is provided in schemes such as TinySec.

This approach is useful when intermediate nodes need to aggregate data; however, it cannot provide true end-to-end security in a multi-hop network. If a single key is compromised, messages can be injected using that key and they are subsequently provided with new MACs by genuine nodes and then forwarded to the destination without requiring the attacker to know the other link keys.

Ideally, compromised nodes should lose their ability to *generate* or alter authentic messages, but should still be able to assist in network communication by *forwarding* messages. This aim is not possible using the outlined approach; if a node is found to be compromised, any revocation of its keys results in its disconnection from the multi-hop network topology.

[14]Collisions occur when multiple inputs are found that generate the same output. This is slightly different to one-way security, where a hash input should be difficult to determine from the output.

Any nodes that rely on it to forward messages are also disconnected unless they have alternative routes and can be instructed to use them. End-to-end security, by contrast, assigns keys to end-points, allowing messages to be rejected by such end-points if intermediate nodes modify or inject them. Thus, message protection bridges multiple hops and the network can remain connected if a node is compromised but continues to forward messages. This is useful in the physical intrusion detection system; compromised nodes can be left enabled, and still perform useful forwarding.

There are some IP capable gateways, for example from ArchRock [67], that allow communication between sensor nodes and conventional IP hosts. Although these are marketed as providing *end-to-end* IPsec security, they currently only apply this between the gateway and the IP hosts. Within the network, the security still relies on link-layer security.

Technically, all *IPv6* implementations were mandated to support IPsec (see RFC4294 [68]). IPsec is an example of a protocol that can use the outlined symmetric ciphers, like AES, within its sub-protocols known as *Encapsulating Security Payload* and *Authentication Header*. *6LoWPAN* is a compressed version of IPv6 for sensor networks. However, the 6LoWPAN standard does not currently support IPsec, even though it claims to be fully IPv6 compliant. Recently, a compressed version of IPsec has been added to the Contiki implementation of 6LoWPAN [10].

WirelessHART [45] crucially adds end-to-end encryption and authentication using session keys that are provided to authorised nodes by a *security manager* host when they join the network. WirelessHART assumes that all nodes in the network maintain security and the emphasis is on authenticating joining devices [69].

It is important to consider that point-to-point authentication is not the only authentication relationship present in a WSN. Broadcasting, or more generally point-to-multi-point, communication is common in the WSN domain to support the distribution of code updates, commands and search queries. *Broadcast authentication* has become a major research area; the limited resources of nodes, such as energy and caches, are vulnerable to resource-drain attacks. Broadcast authentication is a non-trivial problem, which is described in the remainder of this section.

Broadcast authentication might be seen as an obvious candidate for public key cryptog-

raphy. The sink could sign messages with a private key and nodes could verify signatures with the public key. Although this works, it has the drawbacks outlined earlier: it is computationally expensive and vulnerable to resource-drain attacks.

Broadcast authentication cannot be solved using *normal* symmetric authentication. All nodes would need to hold the same key, thus they could imitate each other. Link-layer security cannot solve this problem either; it does not provide end-to-end security, so any node in the system could inject broadcasts.

$\mu$*TESLA* was proposed [70] to solve this using delayed key disclosure from a key chain. Before deployment, a *key chain* is generated on the sink. The chain contains a list of keys obtained from a one-way hash function, seeded with a random key. The last key is then pre-loaded on all nodes and the keys are used, in reverse order, to sign broadcasts. The key for a broadcast is released once all nodes have the message, thus an attacker cannot use the key. The released key is authenticated by applying a hash function to reveal the former key. Nodes can thus authenticate both the key and message. The one-way chain forces an attacker to perform an infeasible brute force attack to find valid keys. $\mu$TESLA requires that nodes cache messages until keys are released as the messages cannot be authenticated until the keys are available for local use. Attackers can thus flood the caches with false messages, creating a new resource-drain attack.

A few protocols, especially for code updates, use *hash chains* (similar in concept to a key chain) in combination with public key cryptography to avoid these problems. A common approach [71, 72] is to sign the first part of a broadcast. Each part contains a hash of the next part, allowing subsequent parts to be authenticated using hash functions instead of public key cryptography. The resource-drain problem is reduced since public key functions can be disabled until the end of the update. The entire message must be known in advance; so whilst the sink could generate a large number of updates in advance, any significant increase in delay between each message (hours rather than seconds) increases the chance of an attacker calculating a collision by brute force.

As resource-drain attacks are a difficult problem to solve completely, *resource protection* proposals exist that *mitigate* the problem. Some are integrated into broadcast protocols, others protect lower layers. The intention in both cases is to reject a large portion of malicious

messages before they reach the vulnerable broadcast authentication protocol.

*AQF-PASS* [73] uses multiple *1-bit* MACs in broadcasts. Symmetric keys are randomly distributed through the network and nodes deterministically map their keys to each bit. Although attackers have a $50\%$ chance of correctly guessing bits, the scheme significantly limits the spread of false broadcasts. The key mapping changes with each message, thus attackers cannot predict how far a message will propagate before it is dropped. AQF-PASS provides good protection for nodes far from the sink, but not those that are closer.

*Message specific puzzles* [74] introduce a processing requirement at the broadcast sender called a puzzle. The puzzle is formed of a message and a key from a one-way chain. The solution requires time to complete, but once a solution is provided it can be easily verified. Since the solution requires knowledge of the next key in the chain, an attacker cannot immediately begin solving the puzzle. Thus an attacker has less opportunity to inject messages since it takes too long to complete the puzzle before the next solution has been used.

### 4.2.4 Physical Layer Security

In an effort to avoid attacks on communication protocols, security schemes in WSNs are increasingly being applied at the physical layer. Attackers often exploit the physical layer to their own advantage, but honest parties rarely do the same [75].

Physical properties can be used to provide authentication and other security services. These properties can be used to generate keys, for example from RF frequency selectivity [76] or sound side channels [77]. Of particular interest are those that use the physical properties to provide authentication directly.

Jamming, which can be carried out just by scanning for signals and then jamming the appropriate frequencies [78], can be applied in a more intelligent way to achieve distributed authentication. *Jamming for Good* [75] does this in a WSN by observing channel characteristics, specifically received signal strength, and disrupts the *Start Frame Delimiters* sent from malicious nodes. This prevents the reception of malicious messages.

*Location limitation* can be used to deliberately confine communication signals to physical areas in order to avoid eavesdropping and injection. Physical contact can be included in protocols to deliver this security property [79]. LED light is used in KeyLED [80] to convey

keying material. Sound can also be used, even using friendlier sounds such as tweets [81] if necessary.

Authentication can also be applied based on distance. Research in the smart card domain has evaluated *distance bounding*, based on RF distance measurement, to prevent communication by distant adversaries [82, 83].

As this domain is highly complex, it is covered in more detail in Section 4.4. The most important observation is that authentication at such a low layer can mitigate, or even eliminate, many of the attacks in the higher layers. Attackers are either prevented from obtaining vital data to use for attacks or it is made physically impossible to carry out an attack.

### 4.2.5 Other Security Areas

There are, of course, many other areas in WSN security that cannot be covered in great detail.

Although not used in the envisaged high security scenarios, aggregation is used in other WSN scenarios to reduce communication overhead. In these schemes there may be a requirement to ensure that data cannot be viewed or manipulated by intermediate nodes. *Secure aggregation* can be used to avoid these issues. *Homomorphic encryption* schemes [84], for example, allow algebraic operations to be applied to ciphertext.

Given the reliance of some protocols on a global clock, time synchronisation has to be applied in WSNs to address the problems of clock drift and inactive clocks. *Secure time synchronisation* allows the time to be set with protection from the possibility of an attacker manipulating the timing values. Ganeriwal et al. [85] carry out a security analysis of existing protocols and provide a selection of potential alternatives.

Routing protocols are an essential part of WSNs. Attackers can exploit routing protocols to either intercept or modify messages. *Secure Routing* can address this problem by routing messages in an externally non-deterministic manner. *INSENS* [86] sends redundant copies of messages via different routes; forcing an attacker to compromise more nodes in order to successfully alter a message, but there is an overhead as more than one copy must be sent.

### 4.2.6 Summary of Findings

The main objective in the envisaged scenario from Chapter 1 is to provide a WSN authentication system that provides modern security strength, protects resources and offers graceful degradation when under attack.

Public key systems would offer the most convenient solution as they eliminate privacy problems. Unfortunately, even with elliptic curve optimisations, the overhead is too great and represents a resource-drain vulnerability.

Symmetric cryptography is well developed on WSN devices and offers a significant performance benefit over public key schemes. However, the need for all end-points to share the same key means that key establishment must be secure. Symmetric cryptography cannot be used directly for broadcast due to the same limitation. Key management is therefore an issue that requires investigation.

All WSN broadcast authentication schemes are vulnerable to some form of resource-drain attack. Schemes exist to mitigate this problem, but do not offer a complete solution.

Physical layer security can be used to provide authentication at the lowest layer of the network stack to reject messages before the vulnerable protocols are given the message. This thesis later proposes Distance-Based Message Authentication so that the distance between sender and receiver can be used to reject messages based on a simple threshold method.

The following sections in this chapter therefore focus on key management schemes in WSNs and physical layer security.

## 4.3 Key Management

Thus far, two requirements for the high security WSN scenario have been identified. Firstly, Chapter 1 identified the need for end-to-end security. Second, the previous sections have identified that symmetrically keyed cryptographic mechanisms are more desirable than the less-efficient public key mechanisms.

In high security scenarios, keys will have to be changed regularly for a variety of reasons, such as to avoid cryptanalysis or to enable mechanisms such as finite anti-replay counters.

Key management is thus a major concern, particularly since existing schemes tend to be optimised for link-layer keys and not end-to-end keys.

There are many key management schemes. Most are specific to particular WSN deployment scenarios, exploiting particular security relationships, network topologies and application characteristics. Re-usability is thus hard in different scenarios as the different properties may break these optimisations.

This section specifically reviews the existing WSN key management schemes proposed or implemented. There are two types of key management. One is *key distribution* (or *key transfer*), which focuses on transporting keys between different parties, and the other is *key agreement*, which focuses on computation of keys between parties without transporting the keys themselves. The section evaluates these schemes for their suitability for end-to-end key establishment, particularly between each node and the sink. The section ends by motivating the need for a simple, low-overhead, key management protocol.

### 4.3.1 Key Management Properties

The goal of symmetric key management is to ensure that all authorised end-points can obtain the key *whilst all other parties cannot*. Since the network is multi-hop, all intermediate nodes on a communication pathway can potentially eavesdrop on messages. Worse, since the medium is wireless, all adversaries in range can also eavesdrop.

There are some other important properties that are sought; these aim for modern-grade security strength and to protect the resources of the constrained nodes. These properties will be used to compare schemes:

**Authentication** ensures key material is from the intended party. This can be important for two reasons. First, it assures that messages *encrypted* with the key material cannot be decrypted by an adversary. Second, it assures that messages *authenticated* with the key material can be accepted by the intended target.

**Forward Security** ensures that compromise of a current or past key does not compromise future keys. Backward Security ensures that compromise of a current or future key does not compromise past keys.

**Communication Efficiency** aims to reduce the number of messages required for key management.

**Computational Efficiency** aims to reduce the computational overhead required to manage keys.

**Resource-drain Resistance** is necessary to ensure that an adversary cannot easily abuse the key management schemes to drain resources such as energy.

### 4.3.2 Key Transfer

Key transfer protocols involve the generation of symmetric keys by one party that are then securely transferred to the necessary end-points. Such a protocol can be initiated by one of the parties, but is often undertaken by a trusted authority. The actual transfer can be offline or online.

*Key pre-distribution* is the simplest approach. Keys are installed on nodes before deployment, often as part of the programming toolchain. This process is the most secure since attackers are unlikely to be able to compromise the programming environment.

In order to assign unique keys to each link, prior knowledge of the deployment is needed if the keys are to be used on the link-layer. This problem does not apply to end-to-end keys if there is a common end-point (the sink), but the number of keys required on each node will obviously grow and become infeasible if end-to-end security is required with multiple end-points.

To address these problems, some protocols accept that a smaller number of keys will need to be used and shared by multiple links. This removes the requirement for pre-deployment knowledge and reduces the memory requirements in the network. *Probabilistic key sharing* [87] works from a pool of keys generated by the authority. A random subset of these keys is installed on each node and nodes can negotiate with their communication partners to identify common keys.

The security level of this approach depends on the number of keys generated and the size of the subset on each node. The main problem with this approach occurs when nodes are compromised. A compromised node reveals the keys not only for its own links, but

also those of many other nodes in the network. A careful balance is thus needed between assuring that a sufficient number of common keys remain available in the event that keys need to be revoked against the impact on a single node's keys being stolen. Small subsets of keys reduce the impact, but this also reduces the chance that common keys are available.

The number of keys on each node can be reduced by using *virtual key rings* [88]. In this approach, fewer keys are installed on each node and nodes can indirectly use the keys of their neighbours as well. This approach provides better resilience against node capture, but increases communication overhead.

Fully random key distribution can help an attacker carry out *wormhole attacks* [89] since the keys might be found anywhere in the network. An existing message could therefore be deliberately sent over a faster link (a wormhole) for replay deeper in the network to disrupt protocols. To avoid this, keys can be assigned with geographic knowledge so that keys are distributed in a more localised fashion.

*Robots* can be used with global positioning systems [89] to transfer keys in a phase between deployment and activation. Nodes that are not in direct communication range are not provided with identical keys and thus an attacker cannot use captured keys anywhere else in the network. Unfortunately this scheme requires that the deployment zone is free of eavesdroppers during key transfer, which is unrealistic or at least inflexible.

Without some knowledge of the deployment geography, keys can instead be installed using *clustered key management*. This allocates keys based on network topology to maintain localism. Some protocols use a cluster structure [90] where inter-cluster messages must pass through *supernodes*. This obviously increases communication overhead in some situations (where nearby nodes are in different clusters) and introduces a point of failure, but the compromise of a node will only affect one cluster.

With the exception of the scheme using a robot for key distribution, all of the above probabilistic schemes are unsuitable for end-to-end security. In end-to-end security, it is important to assure that each node has a unique key. Clearly any scheme that reduces the number of keys and enforces sharing will not achieve this goal. Few options are left.

Another clear problem with all these protocols is that there is no specified mechanism to replace the keys. Thus they are not a solution in scenarios where the keys must be

replaced; for example, due to expiring counter values in anti-replay mechanisms. Other transfer solutions, operating over the network itself, must therefore be considered.

The approach of *WirelessHART* [45] is to deliver keys to authorised nodes from a centralised security manager [69]. This approach makes strong assumptions about the security of the whole network and obviously incurs communication overhead.

A key can be directly sent over the network in parts (using the XOR operation, for example) so that an attacker is forced to compromise multiple pathways in order to access keys [91]. This is referred to as *disjoint pathway key transfer*, although is sometimes generalised to *multipath communication*. This requires high communication overhead to achieve good security and does not provide for the scenario where an attacker has the resources to eavesdrop on the whole network.

Transferring keys over a non-radio link is another option to avoid eavesdropping. The *Interactive Guy Fawkes* protocol [79] uses a *location limited side channel* to bootstrap key material. Location limited side channels can be programming cables, light communication or even ultrasonic (sound). The main property is that it is impossible or very hard for an attacker to eavesdrop or inject messages sent over the link.

The applicability of these schemes depends on the application. Sound can be used in a human body network to provide authentication or key exchange [92, 44]. *KeyLED* [80] uses the LEDs and light sensors on nodes to transfer keys. These schemes are slightly overidealistic since such a location-limited channel must exist on an end-to-end basis. Given the necessary range, or potential involvement of intermediary nodes, it is likely that these schemes have a similar security limitation to RF communication. Light and ultrasonic may not propagate as well as radio waves but the environment must still be sealed to achieve location limitation. It could be argued that if such a channel existed, then it could be used for communication.[15]

The *recursive key establishment protocol* (RKEP) [91] addresses this problem by encrypting the radio links on an end-to-end basis via intermediate nodes. The pathways are thus protected from eavesdropping and an attacker is forced to identify the intermediaries and focus his attacks to obtain keys. RKEP assumes an underlying key agreement model.

There are some efficiency issues in the RKEP approach, as links have to be established

[15] Obviously this would depend on the speed.

with the intermediate nodes first. Different methods improve performance [93], such as providing for shorter path lengths or less communication overhead.

Clearly key transfer requires an existing mechanism for confidentiality. Key agreement protocols, where the key can be recalculated using pre-distributed or exchanged material are thus an attractive option.

### 4.3.3 Key Agreement

Key agreement schemes allow two, or more, end-points to agree a key in privacy. Rather than transferring a generated key in a secure way, these protocols instead use *key material* to *calculate* a key. This material might be pre-distributed or can be transferred directly if there is a privacy mechanism built in or no privacy requirement.

Protocols exist that allow keys to be *generated* from pre-distributed key *material*. Since the key material has to be used to compute the keys, such schemes are a form of key agreement rather than key transfer. These schemes resist compromise by tolerating a certain percentage of nodes being compromised.

In the *t-degree trivariate polynomial key system* [94], a global polynomial is created and is used to create *shares* that are distributed. Nodes can use these shares to calculate a shared key. An attacker cannot feasibly determine the global polynomial unless a certain number of nodes have been compromised. Re-keying is not possible unless multiple shares are distributed (weakening the scheme) or new shares are securely distributed. Similar schemes[95] exist with similar properties.

In an important twist to key management, *public key cryptography* can be used for two end-points to agree a secret in privacy. The secret can then be used as a key for symmetric ciphers, an approach used on the Internet within the transport layer security (TLS) protocol [96].

The classical *Diffie-Hellman* [97] protocol can be used to calculate keys by generating private keys on each end-point, exchanging the public keys and then computing a shared secret. An elliptic curve variant (ECDH) is also available [54], which makes modern-strength implementation on WSN nodes feasible. Diffie-Hellman is described in more detail in Chapter 5. Diffie-Hellman achieves privacy but it lacks authentication.

The lack of authentication means a malicious third party can generate false public keys and negotiate keys as a middleman (*man-in-middle attack*). This allows him to generate and alter messages, which is against the primary objective of the work. Another problem is that nodes have to exchange public keys with the sink when keys are refreshed, creating undesirable communication overhead.

*Group key establishment*, which is an extension of Diffie-Hellman, can be used for a larger number of nodes to agree a common secret [98]. It is less useful in an end-to-end scenario, but offers an option for cluster-based communication.

Aside from computation-based schemes, physical protocols can be used to implement *physical key agreement*.

*RF characteristic extraction* can allow two nodes to agree a key. One scheme [76] is designed so that the two end-points can obtain the same radio characteristics whilst an eavesdropper is unable to obtain any useful data. Unfortunately this scheme cannot extend to end-to-end security unless both end-points are able to communicate temporarily in a single-hop.

A similar scheme that uses accelerometers is available that involves a user shaking nodes [77]. The scheme requires some tolerance to sensor data difference. Unfortunately it requires that a user actively shakes the nodes, or some environmental stimulus is available that only shakes those two nodes privately. This approach is useful for group re-keying in a confined area, like a body network, but it is unrealistic to assume that an environmental stimulus is available in a building that is solely observable by the sink and *individual* end-points. Some location-limited mechanism, such as a cable, would be needed that defeats the purpose of a WSN.

### 4.3.4 Comparison

In Table 4.1 a comparison of the schemes from the previous two sections is presented. The comparison is undertaken on the basis of suitability for end-to-end key management, so the findings may not match those claimed elsewhere in terms of link-layer management.

Re-keyability is a core requirement in high security scenarios. Use of a key for too long can result in cryptanalysis risk. In other cases, the use of finite resources such as counter

| Scheme | Re-keyable | Exclusive | Auth. | FB Security | Communication | Computation | Resource-drain |
|---|---|---|---|---|---|---|---|
| Key Pre-Distribution | No | Yes | Yes | N/A | None | None | Very Good |
| Probabilistic Key Sharing [87] | No | No | Yes | N/A | Low | Minimal | Good* |
| Virtual Key Rings [88] | No | No | Yes | N/A | High* | Minimal | Good |
| Robot Deployment [89] | Possible | Partial | Partial | Yes | Low | None | Very Good |
| Cluster Head [90] | No | Yes | Yes | N/A | High* | None | Good |
| Disjoint Pathway [91] | Yes | Partial | No | Yes | High | Low | Good |
| Interactive Guy Fawkes [79] | Yes | Yes | Yes | Yes | Low | Low | Very Good |
| KeyLED [80] | Yes | Partial | Partial | Yes | None | Low | Very Good |
| Sound Re-keying [44] | Yes | Partial | Partial | Yes | None | Low | Very Good |
| Recursive Key Establishment [91] | Yes | Partial | No | Limited | High | Medium | Good* |
| t-degree Polynomial [94] | No | Limited | Yes | N/A | None | High | Good |
| Diffie-Hellman [97] | Yes | Yes | No | Yes | Low | High | Very Poor |
| Group Key Establishment [98] | Yes | No | No | Yes | Medium | High | Very Poor |
| RF Characteristic Extraction [76] | Yes | Yes | No | Yes | Medium | Medium | Good |
| Common Accelerometer [77] | Yes | Yes | No | Yes | None | Low | Very Good |
| Common Stimulus | Yes | No | No | Yes | None | Low | Very Good |

Table 4.1: Comparison of schemes for end-to-end key management.

values requires key replacement eventually. Thus, observe that almost half of the protocols are unsuitable since they rely on offline installation of material, which is difficult to repeat after deployment. In the case of the robot key deployment strategy, this is theoretically possible but would only be secure if the link between the robot and each node was itself secured.

Keys need to be exclusively held by each node as they are symmetric. Of the schemes that offer re-keying, it is possible to eliminate some schemes. Group key establishment is intended for use to agree a key amongst more than two end-points, so it unsuitable for the outlined need for a different key on each end-point. The common stimulus approaches cannot be relied upon as an attacker may have access to the shared stimulus. For a similar reason, KeyLED and sound re-keying are also partially unsuitable as it is hard to enforce this limitation, due to the need for soundproof rooms or the removal of windows for example. Recursive key establishment and disjoint pathways can be partially eliminated as the 'trusted' intermediaries have some knowledge of key material.

Authentication is highly desirable as it ensures that key management messages have originated from a trusted source. The only available option is Interactive Guy Fawkes as it involves interaction with a trusted person, but this requires physical intervention. Authentication thus remains an open issue.

Forward and backward Security is in all of the protocols, although it obviously does not apply if re-keying is impossible. In the case of recursive key establishment, the protection is limited since knowledge of the keys used by intermediaries can allow an attacker to decrypt keys passed through them.

In terms of communication overhead, this analysis is based on RF communication overhead; thus schemes that use side channels are considered to be free, although there may be a cost in terms of energy to use hardware, such as accelerometers or LEDs. The cost impact, as a result of the key architecture chosen, means that although the communication needed to use virtual key rings or cluster head (zoned) keying is low, all subsequent messages using the established security must involve additional communication. These have therefore been marked as High*, in the table, to avoid misleading claims.

Computation only considers cryptography and routing, not normal computation needed to merely exchange messages. The worst performers are those using constructs from pub-

lic key cryptography, as the computational load is higher than schemes using symmetric cryptographic approaches.

Resource-drain attacks are very important; these consider the impact of an attacker injecting false key management messages. Those that use only negotiation messages and involve little or no cryptography perform well. Clearly the schemes that are either offline or use side channels are very good. Those that are very poor use public key cryptography in response to key management messages, in particular the Diffie-Hellman protocol.

Interactive Guy Fawkes and common accelerometer offer the best overall performance, but both require physical intervention to work. They are therefore not an option. Since no other protocol meets all of the requirements, a trade-off will be required. If re-keyability and exclusivity are considered essential, then only Diffie-Hellman and RF characteristic extraction are left.

Improving the efficiency of Diffie-Hellman based schemes was thus the first contribution of this thesis and is covered in Chapter 5.

However, the main weakness remaining is authentication. Diffie-Hellman has high computational overhead and is thus vulnerable to resource-drain attack. One way to avoid these problems is to implement authentication at the Physical Layer, providing low-level authentication based on means other than solely cryptography.

## 4.4 Physical Layer Security

In the previous sections, a real threat of *resource-drain attack* in cryptographic protocols was identified that can have implications on the survival and usability of a WSN. The concept of *physical layer security* can help to protect these vulnerable higher-layer mechanisms by eliminating a large percentage of, or even all, malicious messages at the lowest layer in the network stack.

This section reviews currently proposed and implemented physical layer security schemes. It then focuses more specifically on secure radio frequency ranging protocols as a basis for the work of this thesis. The section finishes by motivating the potential for the use of securely measured distance as a parameter in message authentication.

### 4.4.1 Physical Layer Protection

The physical layer can be used for *steganography*, where messages are hidden such that unauthorised parties cannot even detect the message let alone access its contents. Some physical layer modulation standards, such as *ultra-wide band* (UWB) and *chaotic spread spectrum* [99] potentially offer this property. The radio frequency (RF) energy degrades in such a way that the signal disappears as noise beyond a specific range. Whilst this is obviously useful to support privacy, it offers very little in an authentication scheme; that is unless it can assure that messages from unauthorised parties will similarly degrade and be undeliverable. Such a property is highly unrealistic if the sender has greater control over the channel than the receiver.

Side channels, as observed in the previous sections, can offer *location-limitation*. The property is useful for confidentiality, but can also be applied for authentication since the restriction is symmetric. The most obvious method would be to use a cable connection, which if made tamper resistant would offer the required security but would be highly counter-productive in a wireless sensor network.

The main objective of side channels in this sense, whether they are based on light, sound or electricity, is to make the channel inaccessible to an attacker. Achieving this on a radio channel is therefore not trivial. Simply blocking all RF communication, for example with a Wi-Fi jammer [78], is not an option as it prevents friendly communication. However, the concept of jamming *only malicious* transmissions has been achieved. *Jamming for Good* [75] allows genuine nodes to disrupt the delivery of malicious messages by interfering with the Start Frame Delimiters (SFDs).

Another option is to use an *evidential communication channel* to alert users if messages are being exchanged. Researchers working on the security of implantable cardiac defibrillators used pork meat to prove that audio communication is possible through tissue, whilst being audible from the outside; this allows the user to be aware of communication that may be unauthorised [44]. Obviously this is unlikely to be applicable in a WSN as it may be unattended.

It is also possible to implement cryptography in the physical layer. *Cryptographic error coding* combines error coding with cryptographic functions. Messages can be rejected

as part of the error detection phase [100]. Thus if a message is not authentic, it can be dropped before any other layer is provided with the message. For encryption purposes, this has even greater benefit; attackers without the key cannot even read the headers, making cryptanalysis a lot harder.

### 4.4.2 Secure Ranging and Localisation

Physical layer security is closely connected to the security areas of *secure ranging* and *secure localisation*. Their objectives are to securely localise a node in either relative or absolute domains. The securely acquired localisation data can be subsequently used for security purposes.

Localisation in WSNs is motivated by the low-cost design of the nodes and the typical lack of global positioning system (GPS) hardware. Localisation is an important mechanism in some protocols that require geographical knowledge for features such as routing. Localisation is thus a large research area in WSNs; it is, however, inappropriate to conduct an in-depth analysis of the different protocols.

The main observation that can be made is that all the localisation protocols tend to take the same approach, noted by Srinivasan and Wu [101] as well as Langendoen and Reijers [102]; first measurements are taken between nodes to obtain distances, then messages are exchanged to build a network model and finally the process is repeated to improve accuracy.

The need for security in localisation is pretty obvious. If an attacker can modify the geographic model, then it is probable that he can influence node relationships and routing mechanisms.

There are generally two research areas that investigate security protection. One area aims to secure the measurement process itself, with an emphasis on physical boundaries. The other area seeks to secure the negotiation phase, so that the attacker cannot modify the exchanged values used in building the model; this area is generally based on message authentication. This work focuses on the first area.

### 4.4.3 Distance Bounding Protocols

The problem with measuring distance between two nodes is that an attacker might be one of those nodes and does not need to comply with the rules that would be followed by a genuine node.

*Received signal strength indication* (RSSI) can be used to determine distance by using a function that converts the RSSI into a distance measurement. Most research has found that RSSI is generally an inaccurate means for such ranging [103, 104]. More importantly, it is immediately possible to eliminate RSSI as a secure ranging method since an attacker can manipulate output power; this would cause incorrect range measurement on the genuine node.

Smart card security provides the basis for one method of secure range measurement. Some smart card systems need to prove that the card is near the card reader. This is motivated by attacks described by Drimer and Murdoch [82] that show the triviality of carrying out range extension attacks on schemes like the UK 'Chip and Pin' system so that a card can be used for a remote transaction. Similarly, Fajardo and Dominguez [105] show that wireless smart cards can be *bridged* so that a genuine remote card can be linked over a long distance to a local reader in a form of *virtual pick pocketing*. One popular proposal made to address this problem was to use *distance bounding*.

Distance bounding is based on *round-trip-time* (RTT) measurement between two parties. In RTT methods, a probe is sent from one node to another, and then immediately returned. The time taken for the probe to be returned is a function of the response time and radio propagation delay. Thus if the response time can be determined, the remaining time can be used to calculate distance. This process has been implemented in hardware with an efficiency profile suitable for WSN usage; one such example is the Nanotron NA5TR1 [106].

RTT is usually applied for non-security purposes, such as locating firemen using an ad-hoc network in a burning building [107]. Modern implementations tend to use either *ultra-wide band* or *chirp spread spectrum* [108].

RTT measurement is not secure since an attacker can disobey the rules and reply before receiving the probe. Protocols, one of the first being by Brands and Chaum [109], attempt to resolve this by enforcing a lower bound. The general approach of such protocols involves

including an unpredictable value called a *nonce* in the probe, so that the attacker does not have sufficient data to reply early. With the emergence of low-power *software defined radio*, such as SODA [110], it is now possible to implement hardware designs based on software instructions. The engineering challenge for an attacker is therefore easing. Over time, the protocols have been strengthened with mechanisms such as rapid bit exchanges and different modulation to make it very hard for an attacker to breach the security.

This bounded RTT method has been widely used, or extended, in many research papers. Many of these papers analyse the physical security of the scheme as applied using a variety of technologies. Examples include work by Flury et al. [111], which identifies attacks against the IR-UWB method, and work by Rasmussen and Čapkun [112], which raises privacy concerns since an eavesdropper may be able to identify the layout of the network by listening to the ranging messages.

Whilst there are specific modulation and hardware design concerns that these papers address, there is an important difference between smart card and WSN communication. Smart cards communicate at a range of centimetres from the reader. By contrast WSN devices communicate at a range of several, perhaps dozens of, metres with the potential for obstructions and reflected signals. Existing research is not concerned about *multipath* effects and the implications on ranging accuracy; this is despite clear evidence, such as by Miluzzo et al. [113] and Obayashi and Zander [114], that just the presence of *human beings* can radically alter signal propagation. Whilst multipath errors will not cause range reduction effects, they can cause range-extension effects. These effects can cause implementation challenges. Since these issues are more important in WSNs, there is a need for further evaluation.

In Chapter 6 secure RTT is integrated into the physical layer to enable distance to be measured during message exchange. Messages are then rejected if they are sent from outside of an acceptable range. This enables deployments in secured building scenarios to benefit from physical boundaries such as security fencing. Ranging accuracy is specifically addressed.

### 4.4.4 Findings

Upper layer protocols, particularly those based on public key cryptography, are vulnerable to various attacks. Preventative measures are not fully effective and can themselves be attacked. Thus protection at the physical layer is worthwhile.

There are a number of protocols that provide physical security by limiting access to the communications channel, but they require infeasible hardware such as cabling or soundproof rooms.

One way to achieve the security desired is to limit radio channel access. Intelligent jamming is one option. Another is to use distance bounding, in conjunction with message exchange, to provide a protected radio space. This process requires further evaluation as the ranges involved in WSN communication can involve reflections and thus measured distance enlargement issues.

## 4.5 Conclusion

This chapter identifies that many areas of WSN security are well developed, for example the industrial-strength block ciphers have been implemented and can be used to provide cryptographic authentication. However, two areas were identified that are less ideal. Firstly, cryptographic authentication is rarely implemented on an end-to-end basis in WSNs; many of the available key management schemes are not designed to distribute keys in this pattern. Second, once the management is in place, some schemes are vulnerable to resource-drain and key theft; additional protection is thus desirable. These findings resulted in the contributions of this thesis.

Chapter 5 proposes a key management protocol that uses the Diffie-Hellman key agreement protocol in a specific way. Individual node keys, used for transport layer security between each node and the sink, are replaced by sending a single broadcast from the sink. By using Diffie-Hellman, all nodes can arrive at a different shared secret without being able to obtain those of other nodes. In addition, the keys offer forward and backward security. The main benefit of this approach is the reduced communication overhead. The chapter evaluates these issues, finds efficiency benefits but identifies problems if the broadcast is

forged by an attacker.

Chapter 6 approaches the problem of resource-drain attacks against authentication protocols, particularly in strengthening broadcast authentication. DBMA uses the RF distance bounding principle to provide low-level Distance-Based Message Authentication. Attackers are prevented from injecting messages in scenarios where they cannot physically access the deployment environment. The chapter deals with the different propagation issues involved in long range WSN links when compared with the small ranges encountered in the original smart card scenarios.

# Chapter 5

# Broadcast Key Establishment

**In high security applications, such as the physical intrusion detection system described in Chapters 1 and 2, authentication must be carried out on an end-to-end basis. Chapter 4 found that strong symmetric authentication is feasible in WSNs, but requires secure key management to ensure symmetrical keys are shared between each sensor node and the sink. Keys should be refreshed during network operation, but the cost of doing so should be evaluated. In particular, the energy, computational and communication overheads are key areas of concern as they affect node lifetime, responsiveness and link availability. Existing schemes do not provide sufficient performance. This chapter introduces the concept of *Broadcast Key Establishment* (BKE) that uses a single sink-initiated broadcast message to set different keys on all sensor nodes. BKE preserves scarce resources such as bandwidth and energy, simplifies routing requirements and better fits available network facilities and protocols. Although different cryptographic mechanisms can be used as part of BKE, this chapter focuses on one in particular: BKE/D that uses the Diffie-Hellman protocol. Others are also introduced and tested using the popular TinyOS platform. The performance of BKE is evaluated in depth.**

## 5.1 Motivations

The physical intrusion detection system, described in Chapter 2, requires that the sink can authenticate that sensor reports are generated by a specific, trusted, node and not modified

within transit. The existing approach in WSNs is to use link-layer security, which for reasons already discussed in Section 2.3 and Section 4.2.3 needs to be replaced with end-to-end security. End-to-end security requires that cryptographic keys are negotiated between each node and the sink, which is a different pattern than seen in most existing WSN key distribution schemes.

Symmetric cryptography has to be used since public key cryptographic algorithms, like RSA, are too computationally expensive for continual use. Where public key cryptography supports open distribution of public keys, symmetric cryptography requires that the same key is present as both ends and therefore requires confidential negotiation. These keys have to be replaced regularly to avoid compromise through cryptanalysis and to allow use of finite primitives like counters to prevent replay attacks.

Key distribution has to take WSN requirements into account. A primary concern is energy efficiency as nodes have a relatively limited energy supply compared to conventional computing equipment. Communication is an obvious target for saving energy, so as few messages as possible should be required for key distribution. However, energy consumption is not the sole reason for reducing message numbers. Available network capacity should be available to transport time critical sensor data and not be (temporarily) consumed by key distribution messages.

The network structure also needs to be taken into account. Most sensor networks are optimised for upstream data transport from sensor nodes to the sink. Consequently most network resources are allocated for this traffic direction to guarantee that messages from a sensor can travel quickly to the sink. Generating a large number of downstream messages flowing away from the sink therefore causes problems; for example, less space may be reserved in caches for downward flow, resulting in potential loss and delay. Thus, it is often impractical to construct a key distribution protocol that requires bi-directional traffic flow between the sink and all sensor nodes.

Finally, packet losses occur in wireless networks. Therefore, any key distribution mechanism must be able to efficiently deal with losses and it must be possible to integrate recovery mechanisms that are not too resource hungry.

Section 4.3 identified that existing key management mechanisms proposed for wireless

sensor networks do not match the outlined requirements. However, a noteworthy method to establish keys on each node is to do so before deployment [70]. This *key pre-distribution* uses offline side channels and cannot be intercepted or tampered with by an adversary; unfortunately, it cannot be directly used to refresh keys since the out-of-band channel is lost after deployment.

In this chapter, the concepts of key pre-distribution and broadcast communication are combined to create the principle Broadcast Key Establishment (BKE). Public key cryptography is used with BKE to implement BKE/D. End-to-end keys can thus be refreshed with heightened security, feasible computation effort and efficient communication overhead. The key distribution is simple to execute and fits the communication patterns observed in sensor networks.

## 5.2 Principle of Broadcast Key Establishment

To overcome the outlined limitations, this thesis proposes Broadcast Key Establishment (BKE) as a general principle for use in networks where the security keys on individual nodes need to be individual, but shared with a common end point (usually the sink). All of the keys in the network are replaced confidentially when the sink broadcasts a single value. Each node generates a new key and no other node, except the sink, should be able to obtain it[16].

Several variants of BKE will be discussed in this chapter; they all share the same two phase principle as shown in Figure 5.1. In the first phase, offline *key material pre-distribution* ensures that cryptographic key material is shared individually between each node and the sink. In the second phase, the sink generates key material and broadcasts it in a single message. This triggers a function executed by each node to generate a new shared secret. A similar function is executed by the sink to generate the same shared secrets associated with each node. The shared secrets are then used to generate key material for the end-to-end authentication mechanisms, or any other relevant purpose depending on the scenario.

**Phase 1** A unique node key $k_n$ is generated and stored on each node. A related key $j_n$ is generated and stored on the sink. This occurs before deployment on a secure side

[16]There is obviously a very small chance that multiple nodes generate the same key, but this is not predictable by an attacker.

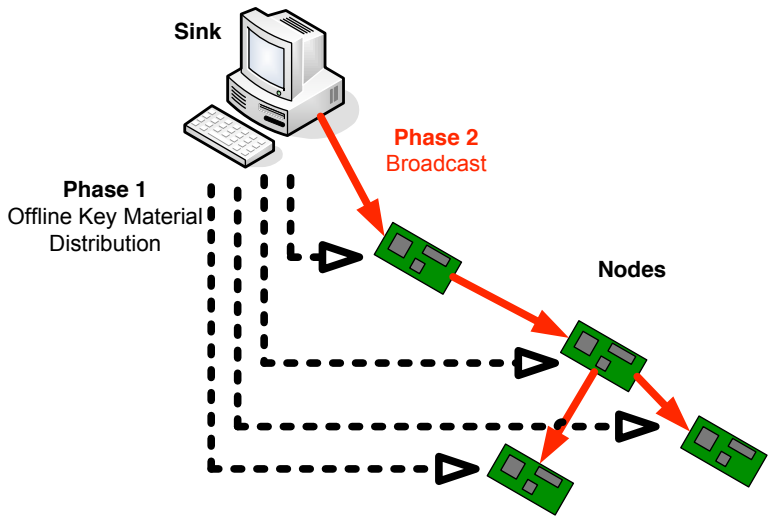



Figure 5.1: The phases of Broadcast Key Establishment.

channel.

**Phase 2a** The sink initiates a new round by generating a new secret key $b$ and a related key $a$. $a$ is broadcast to all nodes.

**Phase 2b** Each node executes function $f(k_n, a)$ to obtain shared secret $s_n$. The sink executes function $e(j_n, b)$ to obtain an identical shared secret $s_n$.

Depending on the implementation, some variables may be identical. Keys $k_n$ and $j_n$ may be identical. Sink keys $b$ and $a$ may be identical. Finally, the function $f$ could be identical to function $e$.

Implementations thus differ depending on the functions and variables chosen. Performance depends on the security properties of the protocol, delay of the functions and the length of the messages used to disseminate $a$. However, there are some common characteristics; for example, the broadcast method is identical in all cases. These issues are handled later in the evaluation.

### 5.2.1 General Benefits

BKE is simpler and requires less network features compared to alternative key distribution schemes that require unicast communication from the sink to each node. The network need only provide the capability to distribute a *broadcast* message from the sink to all sensor

nodes. It is not necessary to maintain efficient unicast routes from the sink to each node at the time of key distribution; this allows for a reduction in routing data on each node, eases management overhead and helps to reduce packet header sizes.

Many sensor networks experience strong fluctuation in link quality, which can present difficulties in the maintenance of valid optimal routes to all nodes [115]. Since the BKE key update messages can be distributed by broadcast they can be distributed in a network that has no fixed routing structure. For example, the BKE key update information can be piggy backed on a broadcast message used to setup a network structure for the following data transport from nodes to the sink. Thus nodes have keys ready for immediate use when the topology has been established.

Some sensor networks are optimised for *asymmetric* data flow, as most data flows towards the sink. In such networks it is common for very few resources to be provided to support downstream data flow. Available network capacity is generally defined by the medium access control (MAC) protocol. For example, *DMAC* [116] and *GinMAC* [117] arrange the network such that messages are transported quickly towards the sink. For messages travelling in the opposite direction a small bandwidth is allocated and these messages incur a high latency. Difficulties also emerge in areas such as store-and-forward caching. BKE is better aligned with this asymmetric property of wireless sensor networks.

## 5.3 Diffie-Hellman Broadcast Key Establishment

This section describes the principle variant of BKE called *Diffie-Hellman Broadcast Key Establishment* (BKE/D)[17]. First, the well-known Diffie-Hellman (DH) method based on elliptic curve cryptography is briefly summarised to define the syntax of DH parameters in the context of this chapter. Second, the BKE/D mechanism as a modification of the standard Diffie-Hellman key establishment is outlined. Finally, the security of the BKE/D mechanism is discussed.

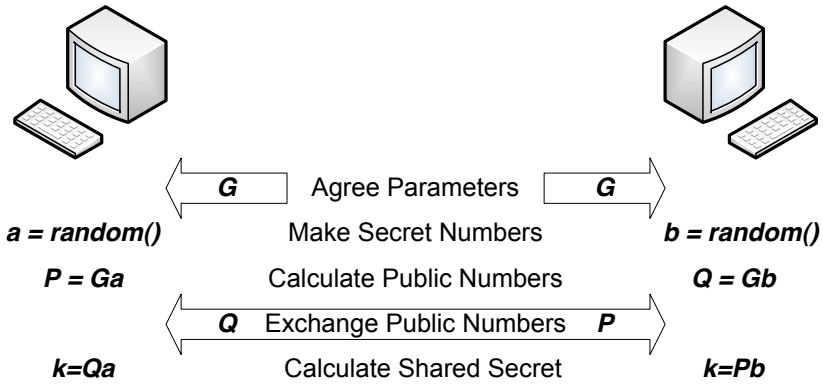



Figure 5.2: Elliptic Curve Diffie-Hellman.

### 5.3.1 Elliptic Curve Diffie-Hellman

*Diffie-Hellman* (DH) [97] is a public key cryptography construct that establishes a secret $k$ between parties $A$ and $B$ in secrecy over an open channel. This secret $k$ may then be converted into cryptographic keying material for use with symmetric ciphers.

Figure 5.2 shows a variation of Diffie-Hellman based on *elliptic curve cryptography* [54, 55]. This is known as ECDH and provides similar security to normal Diffie-Hellman with significantly shorter keys. ECDH is important in WSNs as its computational overhead is less than that of conventional DH.

This version is discussed since the variables and operations are slightly different. ECDH operates as follows.

1. $A$ and $B$ agree ECDH parameters with curve base $G$.
2. $A$ generates private number $a$ and public point $P = Ga$.
3. $B$ generates private number $b$ and public point $Q = Gb$.
4. $P$ and $Q$ are exchanged over an insecure channel.
5. $A$ generates a secret $k_a = Qa$.
6. $B$ generates a secret $k_b = Pb$.
7. The shared secret is: $k = k_a = k_b = aQ = bP = aGb$.

[17] BKE/D was formerly referred to as DHB-KEY in previous publications and was renamed to provide greater emphasis to its BKE foundation and the proposal of alternative variants.

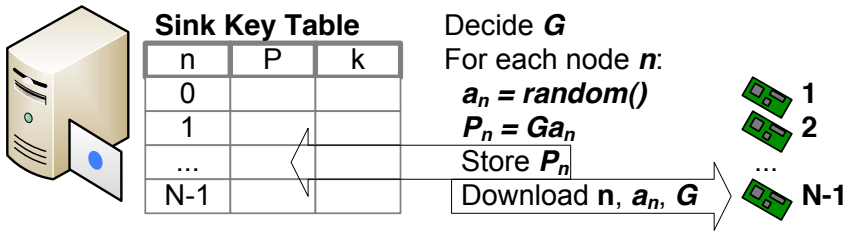



Figure 5.3: BKE/D phase 1.

A possible attacker only has access to $P$ and $Q$ (and possibly $G$), which is insufficient to feasibly calculate $k$. However, the key establishment is vulnerable to man-in-middle attacks as $P$ and $Q$ are exchanged without authentication.

### 5.3.2 BKE/D Establishment Mechanism

BKE/D is a modification of the basic ECDH key establishment mechanism using a static private number on one side and a shared ephemeral private number on the other side. The first half (phase 1) of the ECDH key establishment is undertaken before deployment and establishes static key-pairs for all sensor nodes. The second half (phase 2) of the ECDH key establishment is executed periodically using an ephemeral key-pair generated on the sink; the public key from the sink is broadcast to all nodes and then used to compute new secrets that are shared only by each individual node and the sink.

#### Phase 1

The sensor network consists of $N$ sensor nodes and a sink. $N$ private numbers $a_n (\forall 0 \leq n < N)$ are generated using a *pseudo-random number generator* and corresponding public points $P_n$ are calculated. Each sensor node $s_n$ is configured with its $a_n$ and a table on the sink is populated containing all $P_n$ (see Figure 5.3). Phase 1 is carried out once only in a secure environment before network deployment:

1. ECDH parameters with curve base $G$ are selected and shared.
2. All $a_n$ and $P_n = Ga_n$ are calculated by the sink. Each $a_n$ must be unique.
3. Each $a_n$ is stored on the corresponding node $s_n$.

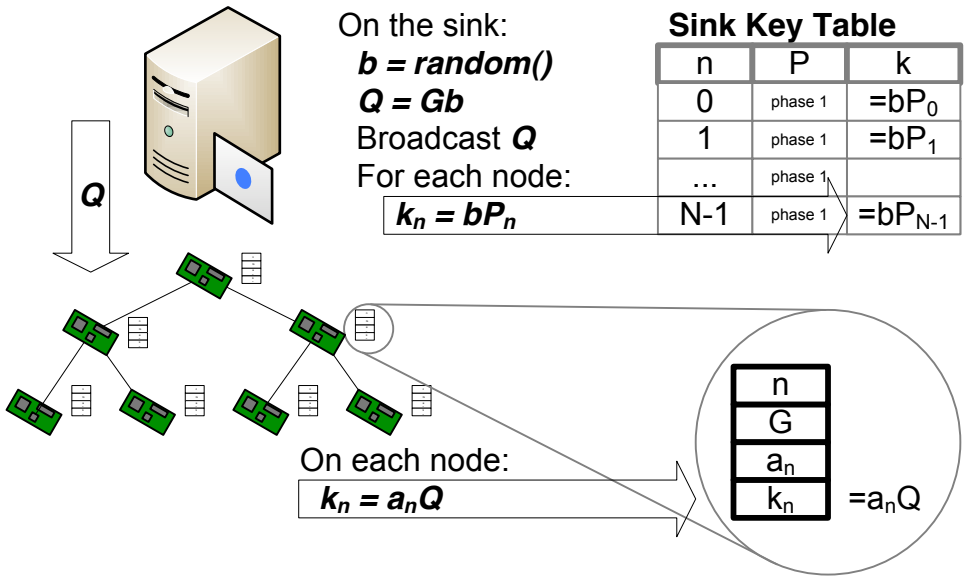



Figure 5.4: BKE/D phase 2.

4. All $P_n$ are tabled on the sink.

Because this is conducted before deployment, it is resistant to man-in-middle attacks. An attacker cannot obtain or modify any $P_n$ and so will find it infeasible to imitate either party later on.

**Phase 2**

The sink generates a new ephemeral private number $b$ and corresponding public point $Q$. $b$ must not have been used previously[18]. The public point $Q$ is distributed in the network using a single broadcast message (see Figure 5.4). All nodes $s_n$ use this value $Q$ to calculate a new individual shared secret $k_n$ using their locally stored $a_n$. This process can be repeated periodically to set keys on all nodes.

1. The sink creates $b$ and $Q = Gb$. $b$ must not have been used previously.

2. The public point $Q$ is broadcast to all nodes.

3. Each node $s_n$ recalculates the secret $k_n = a_n Q$.

4. The sink recalculates all secrets $k_n = bP_n$.

[18]Given that $b$ will be over a hundred bits in length, the range of available values is unlikely to be exhausted in most networks.

5. The secrets are shared as $k_n = a_n Q = bP_n = a_n Gb$.

Variables $k_n$, $j_n$, $b$ and $a$ in BKE map to $a_n$, $P_n$, $b$ and $Q$ in BKE/D respectively. Functions $f()$ and $e()$ in BKE map to the calculations $a_n Q$ and $bP_n$ respectively. BKE/D thus fits the BKE concept.

### 5.3.3 BKE/D Security Analysis

BKE/D is different to basic ECDH in two important aspects: (1) the sink's public point $Q$, derived from its ephemeral private number $b$, is common to all nodes in their production of shared secrets $k_n$ and (2) the nodes private numbers $a_n$ are not replaced in each key negotiation round. Even assuming that ECDH and DH are safe, it has to be analysed if these differences represent a security risk.

**One-to-Many Relationship** Diffie-Hellman is used in other protocols in a one-to-many communication relationship. For example, the Transport Layer Security (TLS) [96] protocol commonly used to secure the communication between web servers and Internet browsers can use DH key establishment. A web server can include a DH public key ($Q$) in a static certificate; all clients connecting to this server use this public key and their private key to create the shared secret $k_n = a_n Q$. As TLS is widely used and considered to be safe, it is assumed that this aspect of BKE/D does not represent a security risk.

**Static Mode** The usage of static private numbers $a_n$ on the nodes is similar to the *ephemeral-static mode* described in RFC2631 [118]. This RFC defines the usage of DH in Internet protocols and explicitly specifies a mode of operation in which one DH side uses a static private key and the other side uses a fresh private key for each negotiation. TLS also specifies this mode of operation for key negotiation. It is therefore assumed that this aspect of BKE/D also does not represent a security risk either.

**Compromised Private Keys** An attacker might be able to compromise a node and obtain the stored private number $a_n$. From that point on the attacker is able to impersonate the compromised node and retrieve previously used keys. The problem of compromised nodes is not specific to BKE/D. Communication channels cannot be protected if

endpoints are already compromised. Importantly, the use of end-to-end security and separate keys means that the compromise of $a_n$ does not damage the trust of the whole network; that node is unable to modify messages it forwards and cannot imitate others.

**Compromised Shared Secrets** Depending on the protocol, an attacker might be able to obtain $k_n$. It is assumed that $a_n$ cannot be recovered from $k_n$. ECDH does not reveal the private numbers $a_n$ or $b$ of either side even if the shared secret $k_n$ is discovered. For example in TLS the client cannot obtain the server's private key despite having calculated a shared session key. If it could, the security of TLS would be broken as soon as a client connected and the server would need to replace its private key on each connection. This is not the case since the private key is a signed element of its certificate.[19]

**Broadcast Authentication** BKE/D does not authenticate the broadcasted public point $Q$ in phase 2. Whilst an attacker cannot exploit this to authenticate with the sink, this can be exploited for *denial-of-service* attacks, some of which fall into the category of *resource-drain* attack. In particular, an attacker can maliciously inject false public points $Q'$ that force nodes to calculate invalid keys. The sink drops subsequent messages as the wrong keys are used, resources in the network are blocked and energy is wasted. This may not be problem in some scenarios as an alarm can be generated indicating an electronic attack. For other scenarios an *authenticated broadcast* message might be necessary. It is observed that existing broadcast authentication methods, such as $\mu$TESLA and public key digital signatures, are also subject to denial-of-service attacks [74]. This issue is discussed in Section 5.8.

## 5.4 Protocol Implementation and Practicalities

Broadcast Key Establishment can be used to support end-to-end authentication mechanisms that use symmetric cryptography; the physical intrusion detection application, in Chap-

[19] Knowledge of $k_n$ may not lead directly to $a_n$, but it may still assist an attacker in searching for $a_n$. Implementations should therefore still balance the security needs of the system.

ter 2, has this requirement. This section discusses a protocol that provides this authentication mechanism and conducts key management transparently to the application.

Although this protocol is used for evaluation purposes, it is also suitable for use in static networks. Obviously the protocol can be replaced and BKE used either alone or in a different protocol variant. This section first introduces the protocol and then an implementation for the TinyOS sensor node operating system. A discussion of implementation issues then completes the section.

### 5.4.1 Secure Two-Direction Routing

*Secure Two-Direction Routing* (or 'SecureTDRoute') uses the BKE/D mechanism and is tailored to the application scenario described in Section 5.1. SecureTDRoute provides minimalist routing functionality that allows unicast communication only in the upstream direction, from any node to the sink, and broadcasts downstream from the sink to all nodes. This avoids the need for complex routing algorithms, helps to reduce header length and is sufficient for the application scenario.

It was decided to couple routing functionality and the BKE/D distribution mechanism as this allows implementation of a very efficient recovery strategy for lost key broadcasts. The functionality in terms of routing, key management (based on BKE/D), authentication and reliable broadcast are discussed in the following sub-sections.

A slight derivative of SecureTDRoute called *Secure Binary Tree Routing* (SecureBTRoute) was used solely for fair comparison since SecureTDRoute does not support downstream unicast communication. SecureBTRoute adds *binary tree routing* to support this type of communication. For reasons of flow, this is defined in Appendix A.

#### Routing

Sensor nodes in the network are organised in a *tree* structure, rooted at the sink. This structure is ideal in a fixed network and avoids evaluation problems when dealing with dynamic routing protocols. Sensor nodes are statically deployed and each node is aware of its parent (upstream) node in the tree structure and its child (downstream) nodes, as shown in Figure 5.5.

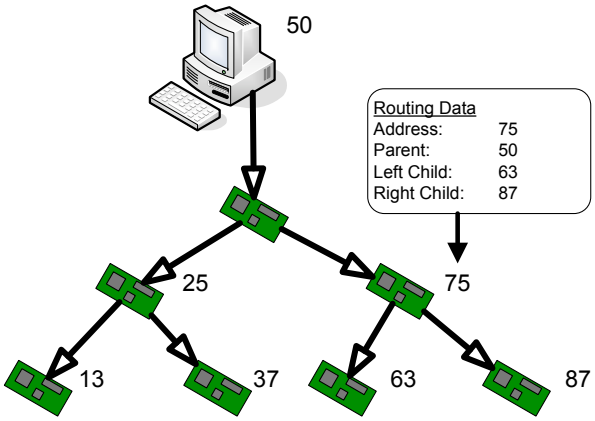



Figure 5.5: SecureTDRoute routing data.

Two types of messages can be routed: (1) downstream broadcasts from the sink and (2) upstream unicast messages to the sink. There are two types of broadcast, BCKEY for key establishment and BCMSG for application messages. There is one type of unicast message, UCMSG. The packet structures for BCMSG and UCMSG are shown in Figure 5.6. The BCKEY format is the same as BCMSG. The packet formats are intended to be carried within a link-layer packet, such as TinyOS's ActiveMessage packets. The fields are explained in Table 5.1.

| Field | Bits | Purpose |
|---|---|---|
| length | 8 | Length of data field in bytes |
| type | 2 | Data type in package (BCKEY, BCMSG or UCMSG) |
| frag | 2 | Reserved for fragmentation |
| R | 1 | Broadcast recovery flag (explained later) |
| appmux | 3 | Application multiplex |
| sender | 16 | Link-layer sender (hop sender) |
| creator | 16 | Network-layer sender (UCMSG only) |
| key ID | 16 | ID of key (BCKEY) or ID of key used to for MAC (UCMSG) |
| seq | 16 | Sequence number (UCMSG only) |
| MAC | 32 | Message authentication code (UCMSG only) |

Table 5.1: SecureTDRoute packet header fields.

Upon receiving a packet SecureTDRoute examines the sender field. If the packet is not from a parent or child node, then it is discarded. This means that SecureTDRoute actually implements a *multicast* strategy, rather than broadcast, since downstream messages are selectively accepted or rejected based on network topology constraints. Thereafter the type field is examined. BCKEY and BCMSG messages are sent downstream and UCMSG messages are forwarded upstream to the parent node. BCKEY data is used internally by

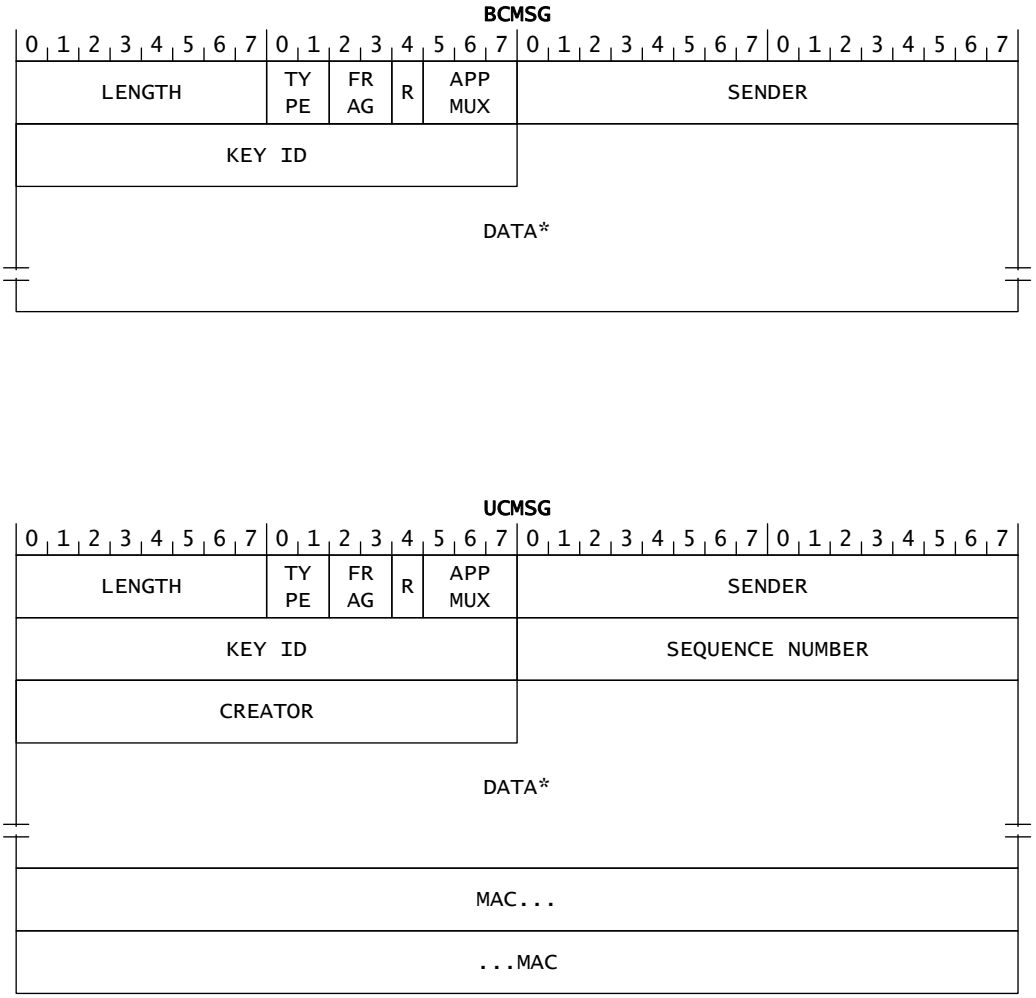



Figure 5.6: SecureTDRoute packet format.

SecureTDRoute to update key material. BCMSG data is passed to the application layer.

### Key Management

Before deployment, the sink calculates the $a_n$ and $P_n$ for each node. The private number $a_n$ is stored on each node and the public point $P_n$ is tabulated on the sink. This constitutes phase 1. Periodically, the sink generates a new private number $b$ and broadcasts the public point $Q$, with a unique key ID $i$ as a BCKEY message. The sink calculates and stores the new shared secrets $k_n$ and corresponding keys $K_n$. As each node receives the public point $Q$, the shared secret $k_n$ is calculated at each node and the symmetric key $K_n$ is derived (phase 2). Each node also stores the current key ID $i$ received in the BCKEY message.

### Authentication

Data sent by a node in a UCMSG message is secured by a $4$-byte *message authentication code* (MAC). The MAC is computed over all the fields, except the MAC field itself, the sender and the R bit. The sequence number field is used to prevent replay attacks. The sink, on receipt of the message, inspects the key ID and creator fields (see Table 5.1) to select $K_n$ and calculates the MAC. The received MAC and generated MAC are then compared and the message is dropped if they do not match. The sink may allow the usage of an old key, for a short period, as some nodes might not have received the latest BCKEY message for key updates.

The implementation uses a symmetric block cipher, with cipher block chaining *CBC-MAC*, to generate message authentication codes (MACs); these are subsequently *truncated* to $32$ bits, in order to reduce the necessary header size. The security properties are discussed later.

SecureTDRoute does not require link-layer security as all authentication is handled by the sink. This is deliberate to allow alarms to be generated when under attack. For the same reason, SecureTDRoute does not implement broadcast authentication.

Primitive *resource protection* is implemented to avoid denial-of-service attacks at the link-layer. This enforces a limited forwarding rate of messages. Link-layer security can obviously be used additionally if required, but this will require another key management

scheme. These security issues are discussed later in Section 5.8.

#### Reliable Broadcasting

Packet losses are common in wireless sensor networks. For UCMSG messages an acknowledgement on the link layer can be used to detect a lost transmission and to initiate a re-transmission. For BCMSG messages this is not an efficient option since acknowledgements increase communication overhead undesirably (see Section 5.6). A recovery mechanism is thus necessary to deal with lost broadcasts to ensure delivery of important key material.

SecureTDRoute applies a loss recovery mechanism based on *packet inspection*. Each node inspects the key ID field when forwarding a unicast message upstream. If the key ID is lower than the locally stored key ID, it can be concluded that some node downstream has not received the latest key update. The node then creates a BCKEY message using locally stored public point $Q$ and sends this message downstream. The recovery flag in the unicast message is set before the node forwards this message upstream to prevent nodes closer to the sink from initiating a repair broadcast as well. The sink might still decide to accept the unicast message with an old key if it is not deemed to be too old.

This approach exploits the single broadcast aspect of BKE. In a unicast scheme a different key update message has to be sent to each node. In BKE a single message is broadcast to the whole network. Message caching quickly becomes infeasible in the unicast case as the necessary cache size is impractical in most WSN platforms. For example, if a node is on the pathway to $100$ nodes, it would need a $5$-kilobyte cache if $50$-byte messages were used. Nodes, such as the Tmote Sky, ship with $10$ kilobytes of RAM; much of this is used by the application itself. In BKE the caching is feasible, because only a single message need be cached. Thus in-network caching allows in-network recovery.

### 5.4.2 TinyOS Implementation

SecureTDRoute was implemented on the MoteIV Tmote Sky node using TinyOS 2.0. TinyOS is a popular choice amongst WSN researchers and developers; it is an event-driven operating system for wireless sensor nodes, providing the necessary features to allow wireless

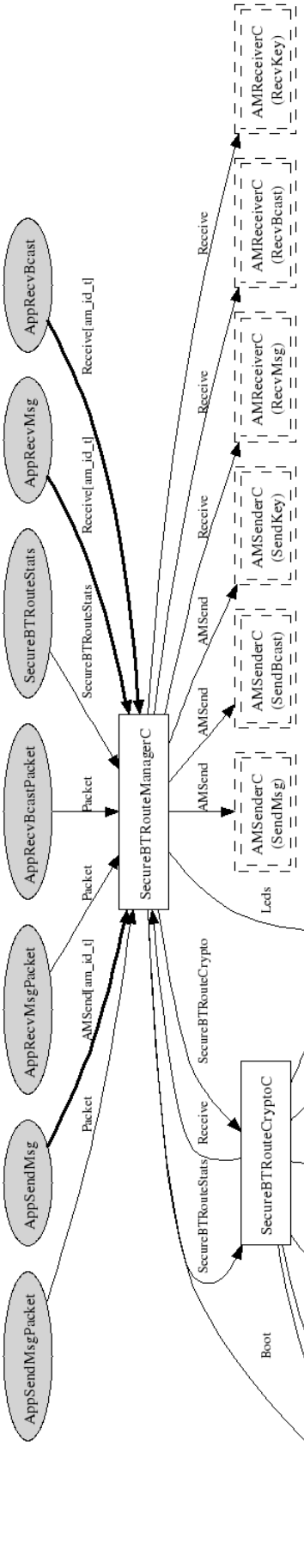



Figure 5.7: SecureBTRoute implementation in TinyOS 2.0.

communication whilst maintaining a low-power profile.

SecureTDRoute was implemented as a set of TinyOS 2 modules, providing the standard TinyOS communication interfaces `AMSend`, `Packet` and `Receive`; this allowed the reuse of existing applications, which can be reconfigured to use `SecureTDRoute` instead of the default TinyOS modules. The applications do not need to be aware of the underlying routing and security features. The BKE key management is handled transparent to the application.

To implement ECDH on nodes, an existing implementation called *EccM* [59] was used. EccM uses the *sect163k1* curve parameters [119], resulting in a $21$-byte key size and a $42$-byte public point $Q$. As a result, the default TinyOS packet capacity ($28$ bytes) had to be increased to avoid using two messages to broadcast the public point $Q$.[20]

The critical bottleneck in ECDH is the time taken to perform scalar point multiplication (SPM). Longer calculation time consumes more energy and affects system responsiveness. An EccM key calculation time of $60$ seconds was measured on the Telos revision B (Tmote Sky) nodes operating at $4$ MHz. Recent work [37] claims that this calculation time can be reduced to several seconds; however, this optimisation was not implemented in the prototype system. It is possible to roughly halve the calculation time by increasing the clock speed to $8$ MHz, but this increases overall energy consumption and so it was avoided.

The calculation delay causes nodes to continue using old keys for a short period while the calculation completes. The sink therefore has to tolerate use of old keys for a period after a refresh.

During key calculation, no other task can be carried out in the default TinyOS distribution; this is because TinyOS does not implement process scheduling, and each task must complete entirely before the next queued task can be executed. The internal design of TinyOS makes extensive use of task queuing, including when forwarding messages; this disables a node's ability to report events and participate in the network when calculating keys. The implementation therefore used *PLScheduler* [34], a TinyOS 2.x extension that adds *task preemption* to counter this problem. The elliptic curve calculation is performed in a 'low' task, which is automatically pre-empted by other tasks. Thus key calculation can be performed as a low priority background process. The alternative would be to use a process scheduler, but this has performance caveats [34] in areas such as energy and memory overhead.

[20] The implementation in initial publications used two messages.

For the generation of message authentication codes (MACs), the MIRACL library implementation of AES-256 was used to implement CBC-MAC.

### 5.4.3 Security Review

The principle cryptographic components of SecureTDRoute are ECDH for BKE/D and AES for end-to-end authentication of sensor reports. ECDH uses *sect163k1* parameters over a binary field of $163$ bits. AES is used with a $256$-bit key in the CBC-MAC mode, with the result truncated to $32$ bits.

A cryptographic algorithm is considered 'broken' if it can be compromised faster than an infeasible brute force attack. There are no reports that currently consider ECDH to be 'broken' provided it is used properly; for example, with a parameter set that avoids weak curves. Recent research using biclique cryptanalysis [120] reports that the security of AES has been reduced slightly, by about $2$ key bits; but, provided it is implemented properly, it still has considerable strength. CBC-MAC is known to be insecure if the initialisation vector (IV) or message-length are permitted to change [121]. Therefore, fixed-length messages, with a static IV, are used. C-MAC is a suitable alternative where variable length messages are required.

Obviously the security also depends on the protocol itself, as well as the foundation laid by the cryptographic algorithms. The truncation of the CBC-MAC output is beneficial for a number of reasons. Firstly it reduces the message overheads and secondly it increases the complexity of cryptanalysis as more inputs map to a single output. The second benefit makes it harder for an attacker to verify keys during an attack. Although a shorter MAC increases the probability of a successful guess by an attacker, it still remains difficult with $2^{32}$ combinations. Since there is no means for an attacker to test each candidate offline, an online test is eventually required for each candidate. This will alert the sink and raise alarms if the guess is incorrect. Thus, it is theoretically possible to have even shorter MACs if the probability of guessing is acceptable.

SecureTDRoute does not require link-layer security since all authentication is handled by the sink. SecureTDRoute does not implement broadcast authentication either. BKE/D does avoid man-in-middle attack by conducting half of the ECDH process offline. SecureTDRoute

still authenticates reports at the sink, so attackers cannot abuse this to send false sensor reports. However, attackers can still maliciously abuse the protocol to drain resources or deny service by injecting false messages, which cause computational overhead and incorrect keying. Link-layer security and a limited forwarding rate are some countermeasures possible. The broadcast authentication issue is discussed in more detail in Section 5.8 as it applies to all protocols using BKE in its default form.

Notice that SecureTDRoute does not authenticate the key ID transmitted in sensor reports to the sink. This is not possible without an additional key distribution model in place to support this. This would add considerable complexity, more overhead and quite likely be self-defeating. An attacker can thus trigger broadcasts of cached key material by abusing the loss recovery mechanism. Such attacks are very limited in scope. First, the message flags will be modified to avoid nodes closer to the sink from repeating the broadcast. Second, the broadcast will be rejected by receivers already possessing the material, self-limiting the propagation to $1$ hop. If nodes do not possess the material, the effect will be to trigger the recovery mechanism earlier than intended, which is actually beneficial.

## 5.5 Theoretical Energy Evaluation

BKE was intended to provide two core energy-related benefits, which are: (1) Lower communication overhead and (2) Better energy balance to help extend network lifetime. This section performs a theoretical energy evaluation to demonstrate why these properties are achievable.

BKE has to be compared with the closest equivalent in terms of security that does not exploit broadcasting. In the simplest such case, herein referred to as Unicast Key Establishment (UKE), individual unicast messages deliver key material from the sink to each node[21]. Such a system can avoid special expensive cryptographic functions since it is not necessary for nodes to share common (broadcast) key material. This avoids computational overhead if the special function has high processing delay (such as SPM found in BKE/D). The actual implementation is unimportant; however, the critical difference is the increase in messages required.

[21] Unicast could also deliver keys from nodes to the sink.

The energy cost of key establishment is made up of two parts; *communication cost* is the energy cost of exchanging messages in the network and *computation cost* is the cost of executing cryptographic functions. Communication cost is closely tied to the communication protocols and transceiver hardware. Computational cost is closely tied to the microcontroller architecture and cryptographic algorithm implementation. The communication and computational costs are therefore calculated separately.

The energy cost of a scheme can be expressed in two primary ways; the cost to an *individual* node and the cost to the *entire* network. The cost to the entire network is obviously of some interest, but more critically the *energy balance* has to be measured to see if some nodes expend significantly more resources than others. This section is therefore organised as follows: Section 5.5.1 will present the component parts of an energy calculation, Section 5.5.2 will present the means to calculate the overall energy cost in a network, Section 5.5.3 will present the means to calculate the critical energy balance in the network and Section 5.5.4 will discuss the comparative results between BKE and UKE.

### 5.5.1 Energy Evaluation Components

The basis of an energy calculation relies on three fundamental components: the cost to *send a message*, the cost to run the *cryptographic function* and the *number of nodes* in the network. These are shown in Table 5.2.

| Variable | Value Used | Purpose | Units |
|---|---|---|---|
| $c_t$ | $1.74$ | Energy cost of sending one message | mAs |
| $c_f$ | $114$ | Energy cost of cryptographic function $f$ | mAs |
| $N$ | - | Number of nodes in the network | qty. (exc. sink) |

Table 5.2: Key distribution energy evaluation components.

In Chapter 3, the cryptographic and communication performance of a typical WSN node was characterised and measured. The equations and variables were defined for the CC2420 transceiver and the MSP430 microcontroller.

For the energy cost of sending a message, assuming use of an extended transmission MAC protocol with an average epoch length of $d_p = 0.1$ seconds, is $c_t = e_T = 1.74$ mAs. Refer to Section 3.3 for a detailed description.

The energy cost of the cryptographic function, for BKE/D, based on the performance of a scalar point multiplication (SPM) in the EccM library, is $c_f = e_M = 114$ mAs. For UKE, *cryptographic* computational cost is not considered as existing cryptographic safeguards can be reused for this purpose.

The computational cost in routing key distribution messages is not considered; it is expected to be far less than $c_f$, as routing decisions are based on simple rule-sets rather than complex cryptographic functions. It would favour BKE anyway due to the simpler rules involved in broadcasting. The processing cost in route establishment and management protocols are similarly not considered.

### 5.5.2 Calculation of Overall Energy Cost

The overall energy cost $C$ is based on the number of transmissions plus the number of cryptographic function executions required. See Table 5.3.

| Value | Purpose | Units |
|---|---|---|
| $Q$ | Minimum key-related transmissions | qty. (exc. sink) |
| $N$ | Number of nodes | qty. (exc. sink) |
| $C_t$ | Total energy cost of message transmission | mAs |
| $C_f$ | Total energy cost of computation | mAs |
| $C = C_t + C_f$ | Total energy cost | mAs |

Table 5.3: Key distribution overall energy evaluation components.

The overall number of transmissions $Q$ has to be carefully considered for several reasons:

**Network Structure** Since WSNs are multi-hop in nature, messages have to be transmitted several times to reach their target. Costs are obviously higher where more hops are needed. Loss therefore also carries high penalties. See Section 5.6 for more details.

**Actual Overhead** The costs only include additional messages for key distribution and associated acknowledgements. Messages that *already exist* in the network are not counted. This allows for various optimisations, such as the loss-recovery approach used in Section 5.4.1.

**Sink Exclusion** The sink is considered to be external to the network and its transmissions are not counted. This is a fair approach since the sink is usually a far more capable host with a considerable energy supply.

**Topology Awareness** Leaf nodes do not *broadcast* and it is assumed that each node will be aware if it is a leaf node or not. This avoids transmission wastage.

The (theoretical) minimum number of transmissions $Q$ can be determined analytically since reliability and the need for retransmission can be ignored.

Three network structures will be assumed in these calculations. The best case applies when all nodes are within a *single hop* of the sink, which is achievable in some scenarios. The worst case applies when the network topology is a *chain*. Realistically a network formed of a tree is more likely; in particular a *binary tree* case is considered where each node has a maximum of $2$ children.

In BKE, $Q$ is based on the number of non-leaf nodes, excluding the sink. In UKE, $Q$ is the *sum* of transmissions needed to reach individual nodes. Recall that sink transmissions are *not* counted.

Table 5.4 shows the formulae to calculate the number of transmissions in each topology for a given network size $N$. The methodology behind these formulae is explained in Appendix B.

| **Topology** | **UKE** $Q$ | **BKE** $Q$ |
|---|---|---|
| One-hop | $0$ | $0$ |
| Chain | $\sum_{i=1}^{N} i - 1$ | $N - 1$ |
| Binary Tree | $\sum_{i=1}^{N} \lfloor \log_2(i+1) \rfloor - 1$ | $2^{\lfloor \log_2(i+1) \rfloor} - 2$ |

Table 5.4: Calculation of overall key distribution transmissions required.

Once $Q$ has been calculated, it is multiplied by $c_t$ to obtain the total transmission cost $C_t$. The total computational cost $C_f$ is obtained by multiplying $c_f$ by $N$. The total overall cost is then the sum of $C_t$ and $C_f$:

$$C = C_t + C_f = Qc_t + Nc_f \tag{5.1}$$

The comparison using these methods can be found in Section 5.5.4.

### 5.5.3 Calculation of Critical Energy Balance

A major benefit of BKE is the improved energy balance within the network. Better balancing the energy consumption can improve the *survivability* of the entire network. The survivability is not only dependant on the overall energy usage in the network, but also the energy usage on nodes that are critical to maintaining communication links to the sink.

In a multi-hop network, nodes are reliant on the help of other nodes to forward messages across the network. In a WSN, with a single sink, the network is particularly reliant on those nodes directly adjacent to the sink. In the worst-case situation, there may be a single node that solely provides connectivity between the sink and the rest of the network. Failure of this node results in the disconnection of the entire network. The term *critical node* is used to refer to that node.

The critical node provides a good evaluation point; it has the greatest communication burden because it is involved in all communication with the sink. Even if there are multiple nodes close to the sink, such nodes will still have higher communication burden compared to other nodes deeper in the network.

Two aspects are important. The first is obvious: how much energy must the critical node expend to re-key the network? The second is less obvious: if function $f$ is expensive, is it actually cheaper to use UKE instead? Both of these aspects are independent of the network topology, assuming the link-layer cost to transmit a message is uniform.

To answer these questions with formulae, the components from Table 5.2 are reused. Instead of computing the overall energy overhead, the energy overhead solely on the critical node $C_d$ is computed. This overhead from BKE can be used as an energy budget to determine how many messages the critical node could send using UKE without executing function $f$. These evaluation components are shown in Table 5.5 and explained below.

| Value | Purpose | Units |
|---|---|---|
| $N$ | Number of nodes | qty. (exc. sink) |
| $C_d$ | Total energy cost on critical node | mAs |
| $x$ | Number of critical nodes | qty. |
| $m$ | Number of messages possible with $C_d$ | qty. |

Table 5.5: Critical node energy evaluation components.

In BKE, the critical node is required to forward a minimum of one broadcast message,

costing $c_t$ and must compute its new key using function $f$, which costs $c_f$. The topology of the network is irrelevant in this scenario. Thus, for BKE:

$$C_d = c_t + c_f \tag{5.2}$$

In UKE, the critical node is required to forward a minimum of $N - 1$ messages, but does not have to execute $f$. Thus, for UKE:

$$C_d = c_t \cdot (N - 1) \tag{5.3}$$

Depending on the energy values $c_f$ and $c_t$, and the number of nodes $N$, either BKE or UKE will result in lesser energy cost to the critical node. A crossover point $m$ exists and if $m < N$ then BKE outperforms UKE. The crossover point can be obtained directly, adding $1$ as the critical node can receive a message for 'free':

$$m = \lfloor \frac{c_t + c_f}{c_t} \rfloor + 1$$

.

If the number of critical nodes $x$ is increased, the crossover point $m$ remains static since the size of the network is irrelevant in the calculation of $m$. This means that if the network is divided equally between the critical nodes, the divided size $\frac{N}{x}$ must be considered in the comparison with $m$. Networks that were large enough to satisfy the critical node benefit with one critical node may thus be too small to satisfy the benefit with two or more critical nodes.

### 5.5.4 Results and Discussion

Using the figures and methods from previous sections, it is possible to compute the overall energy cost $C$ of BKE and UKE in networks of increasing size $N$. BKE/D will be used as the representative example for BKE.

The results are compared for the chain case in Figure 5.8 and the binary tree case in Figure 5.9. There is very little point comparing the single-hop case as no nodes need to transmit; BKE will therefore lose by default because computation will normally cost more than no transmission.

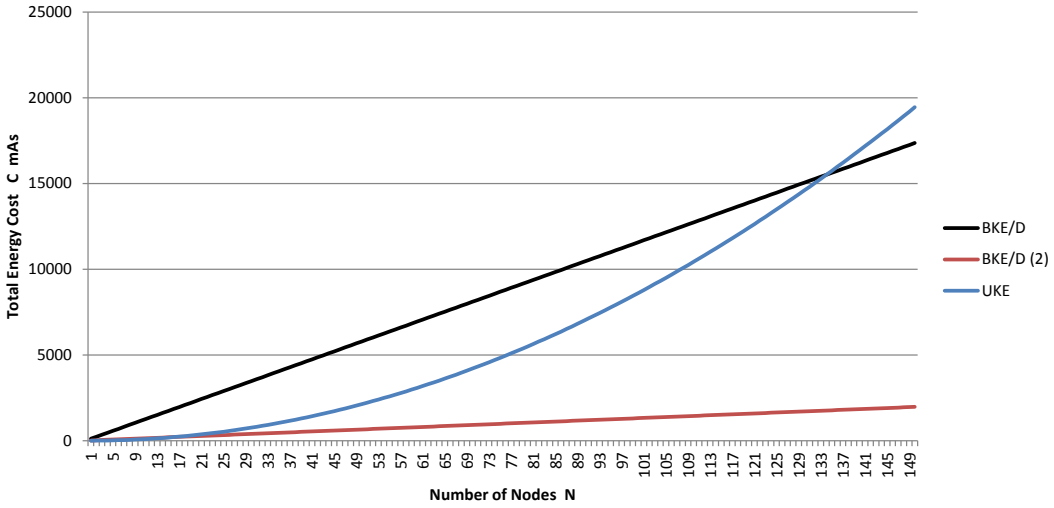



Figure 5.8: Theoretical overall energy cost $c$ comparison (chain topology).

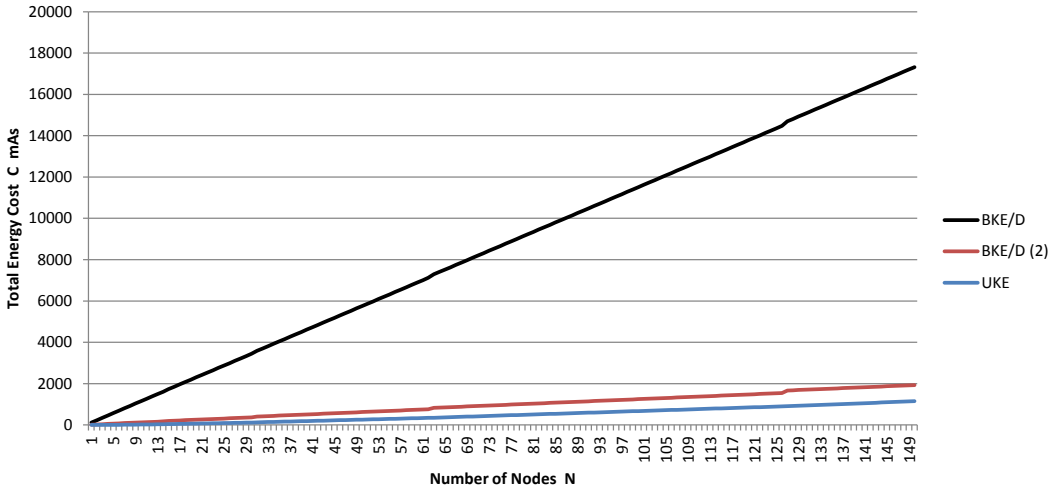



Figure 5.9: Theoretical overall energy cost $c$ comparison (binary tree topology).

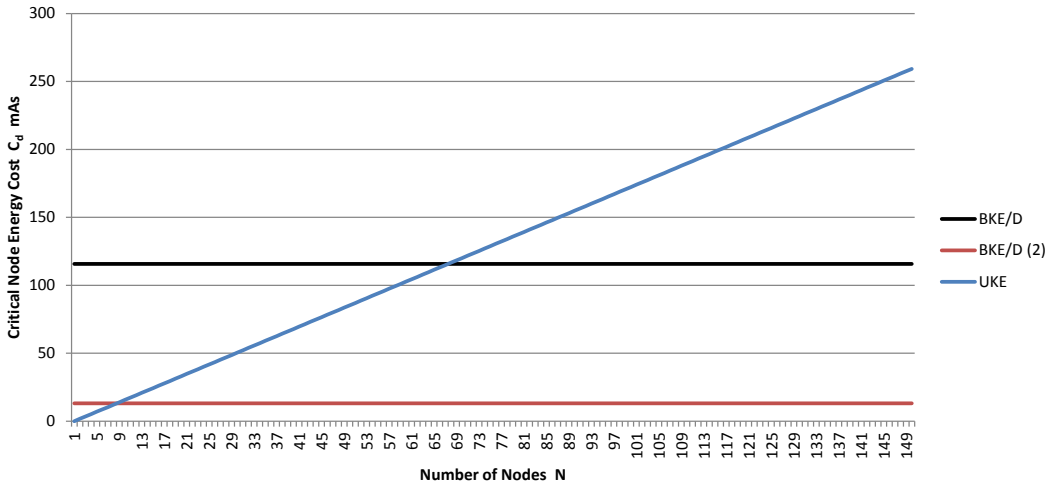



Figure 5.10: Theoretical critical node energy cost $c_d$ comparison.

These results show two immediately obvious aspects. First, BKE/D is significantly more expensive than UKE. Second, the overhead from communication in the UKE case shows an exponential growth pattern. This growth pattern results in BKE becoming more efficient than UKE once the chain network exceeds $N = 135$; this is despite the relatively undesirable computational efficiency of the SPM implemented by EccM within BKE/D.

The tested implementation of BKE/D is not ideally efficient. Work already exists that improves the performance of the SPM, such as in NanoEEC, based on MIRACL [37], which is about $10$ times faster. This is shown as BKE/D (2) in the graphs. Clearly the impact of such more efficient functions is significant.

Attention must now turn to the critical balance issue. Figure 5.10 shows the result when BKE/D is compared with UKE. The expected performance from NanoECC is shown again as BKE/D (2).

It is very clear that the cost of both BKE variants is fixed. UKE grows linearly since the critical node has to forward a minimum of one extra message for each extra node in the network.

Whilst BKE/D has a high cost, it outperforms UKE once the network increases beyond $m = 67$ nodes; this is well within a reasonably sized WSN in the near future. The topology is not relevant as the load on the critical node, in terms of forwarded key distribution messages, will not change with different topologies. Notably, the optimised BKE/D outperforms UKE from $m = 8$ meaning that the crossover point can be lowered even further if $f$ in BKE can be

optimised.

Alternative mechanisms can also replace BKE/D in some circumstances. This is discussed in more detail in Section 5.7.1. Modification of the MAC protocol, the MAC protocol configuration or use of a different transceiver will change the performance of BKE; however, this has not been evaluated in these results.

BKE therefore significantly improves the balance of overhead relating to key establishment.[22]

### 5.5.5 Findings

BKE presents a linear increase in overall overhead and better energy balance. This property is a result of the broadcast mechanism and use of the fixed cryptographic function $f$. This contrasts significantly with the overhead growth patterns observed with UKE.

The benefit of using BKE only occurs once the network size scales beyond a specific crossover point. Improvements in the efficiency of the function $f$ are expected to deliver significant benefit to the overall computational cost. Despite the high cost of the BKE/D scalar point multiplication, the tested BKE/D already outperforms UKE in chain networks of over $135$ nodes. Crucially, although BKE/D adds a significant cost to the overall network, it also provides significant balancing properties that help to preserve the resources on the critical node; this property makes BKE beneficial in smaller network sizes than would have been estimated based on overall energy cost.

This theoretical evaluation has considered the case of one critical node. The addition of more critical nodes will not change the energy cost to each critical node in BKE, however in UKE the load will be balanced between the nodes due to need to forward less messages. Since the load on each critical node will fall in the UKE case, the total network size at which BKE becomes beneficial will accordingly increase. Again, performance gains from the optimisation of function $f$ will improve matters considerably.

[22] The cost is not totally balanced because leaf nodes do not need to broadcast and thus have a slightly lower overhead.

## 5.6 Practical Evaluation

The theoretical energy evaluation used *minimum* communication in its calculations. In realistic environments, the minimum effort is unlikely to be achievable due to a complex array of issues that are hard to analyse theoretically. These include propagation, interference, hardware design, software operation, packet loss, network congestion, retransmissions and implementation specifics such as caching. This section considers those impacts by running experiments to obtain performance measurements in a real WSN implementation. Of interest are networks using contention-based MAC protocols, as collisions both have an impact on, and are affected by, key exchange. Two issues are addressed. Firstly, is the communication performance of BKE still superior to UKE under lossy conditions? Second, SecureTDRoute, specified in Section 5.4, tackles loss-recovery by performing packet inspection within the network and then retransmitting cached material when loss is detected; is such an approach effective?

Two evaluation metrics are used in this evaluation:

**Key Transfer Overhead** The actual communication overhead is the number of key-related[23] messages that must be sent, including those that are retransmissions. This gives an insight into the *actual* overhead incurred, rather than the theoretical minimum.

**Key Transfer Delay** The re-keying delay is a measure of the time delay for nodes to receive their key material from the start of the re-keying-round. Minimal re-keying delay ensures that new keys can be used as quickly as possible.

Good performance is indicated by low communication overhead and low re-keying delay. Lower communication overhead is good for energy preservation, as fewer transmissions are needed, and avoids contributing to network congestion, as there are less messages being exchanged. Faster re-keying delay means that new keys can be used as quickly as possible. Bad performance is marked by higher communication overhead and high re-keying delay. This results in greater resource consumption and longer delays before new keys can be used, which is detrimental to security. For example, if keys have to be replaced during a

[23]This includes all messages that would not normally be found in the network. Some of the distribution schemes use existing messages for acknowledgement purposes and these are not counted.

physical event (such as a burglary), the sink needs to be sure that the security mechanisms are supported by fresh keys. If re-keying causes the introduction of delay, an attacker may be able to exploit this to carry out an attack and escape before being detected.

These properties are examined in the same types of network discussed in the previous chapter: chain, tree and one-hop topologies. In each network, the reliability of message exchange is artificially reduced in the different experiments to evaluate degradation. Good performance during increased message loss will be marked by smaller increases in communication overhead.

### 5.6.1 Evaluation Principle

| Aspect | Metric | Purpose | Unit |
|---|---|---|---|
| Actual Overhead | $m_n$ | Transmissions (by node $n$) | Qty. |
| | $M$ | Transmissions (sum over whole network) | Qty. |
| Delay | $d_n$ | Dissemination delay (node $n$) | ms |
| | $D$ | Dissemination delay (max. over whole network) | ms |
| Parental Effort | $s_n$ | Min. transmissions required to reach node $n$ | Qty. |

Table 5.6: Key distribution communication performance metrics.

Table 5.6 shows the evaluation metrics that these experiments collect. In each round, three sets of data are recorded on the nodes: First, the actual overhead $m_n$ is the number of key-related transmissions, made by node $n$, including *dedicated* acknowledgements, derived from a transmission counter. Second, the delay $d_n$ is the time difference between the starting time at the sink and the local time on node $n$ when the first key message is received in that round. The clocks must be synchronised, which is discussed later. Third, the parental effort $s_n$ is derived from the downstream retry counters on nodes (including the sink) between the sink and each node individually; this gives an indication of the *required* number of transmissions, excluding acknowledgements.

Periodically, nodes transfer those values to the sink using unicast. The sink then records and aggregates them to form the total transmissions $M$ and the maximum dissemination delay $D$ for that round.

| Strategy | Mode | Retry Method |
|---|---|---|
| Unicast | U-ACK | Periodic, stopped via link-layer acknowledgement. |
| | U-SPI | Triggered, via packet inspection of reports at sink. |
| Multicast | M-ACK | Periodic, stopped via link-layer acknowledgement. |
| | M-LPI | Triggered, via packet inspection of reports at each hop. |
| | M-SPI | Triggered, via packet inspection of reports at sink. |
| Broadcast | B-LPI | Triggered, via packet inspection of reports at each hop. |
| | B-SPI | Triggered, via packet inspection of reports at sink. |

Table 5.7: Key distribution dissemination modes.

### 5.6.2 Protocol Variants

Seven protocol variants were tested, as shown in Table 5.7. These were formed from combinations of three modes and three types of reliable-control. The types of dissemination tested were *unicast* (U), *multicast* (M) and *broadcast* (B). The types of reliability control tested involved either *dedicated acknowledgements* (ACK), *link-layer packet inspection* (LPI) or *sink packet inspection* (SPI).

The unicast variants (U-ACK and U-SPI) involved the sink sending a *different* message to each node; these correspond with the unicast key exchange (UKE) example used in the theoretical evaluation. The multicast and broadcast variants involved the sink sending a *single* message to the whole network. In the broadcast protocols, dissemination messages were accepted by any node that heard them; in multicast protocols, only messages sent from a parent were accepted. This allows the broadcast protocols to opportunistically receive broadcasts, rather than strictly observing the network topology; where networks are deployed with nodes in overlapping RF space, this may help to improve the speed of dissemination.

The ACK variants acknowledged, at the link-layer, any dissemination messages received. Nodes continued to periodically re-transmit messages until they were received by the relevant immediate children.

The LPI variants performed packet inspection on forwarded sensor report messages. A field in the sensor report messages indicated the latest dissemination ID received by the source of the sensor report. If the node was out of date, the broadcast or multicast was repeated using locally cached material. The sensor report was flagged to prevent nodes

further up the tree from repeating the recovery. This approach is identical to that used in SecureTDRoute, see Section 5.4. The SPI variant is identical, although applied only at the sink and additionally supported unicast.

Two of the nine combinations were *not* tested. This thesis argues that it is infeasible to locally cache unicast packets in a large network, so the U-LPI method was not tested; see Section 5.4.1 for the argument. The B-ACK variant was also not tested as it is difficult to know which nodes should acknowledge messages at the link-layer during a broadcast.

### 5.6.3 Implementation Specifics

The experiments were run in an office environment and then in an RF-screened room. $14$ nodes, plus the sink, were deployed in the topologies shown in Figure 5.11. In the office environment, nodes were places on door frames and positioned such that they were all within one-hop of the sink. This configuration allowed a comparison of the different methods without needing to use different deployments, which may have resulted in an unfair comparison. Despite this, the area of the deployment was not artificially reduced since non-line-of-sight communication ranges of about $20$ metres were achievable. A map of this deployment is shown in Figure 5.12. This deployment arrangement was chosen as it reflected a physical intrusion detection system deployment. All experiments were repeated in an RF-screened chamber to provide clean channel conditions [24].

A network of Telos revision B nodes was utilised, running a TinyOS application. Each node used an external, plastic-coated, vertically polarised, half-wavelength, dipole antenna. Each node was configured with a pre-computed address and routing data for each of the topologies. The default TinyOS MAC was utilised; this uses clear channel assessment (CCA) to avoid collisions and has no low-power extensions. All received packets were processed in software, no hardware address recognition was used and acknowledgements were generated in software; this eliminated potentially unfair comparisons since some modes require software processing whilst others do not. The nodes did not accept messages until any previous message had been processed or forwarded.

The control of experiment parameters, such as the mode and topology of the network,

[24] and eliminate curious humans...

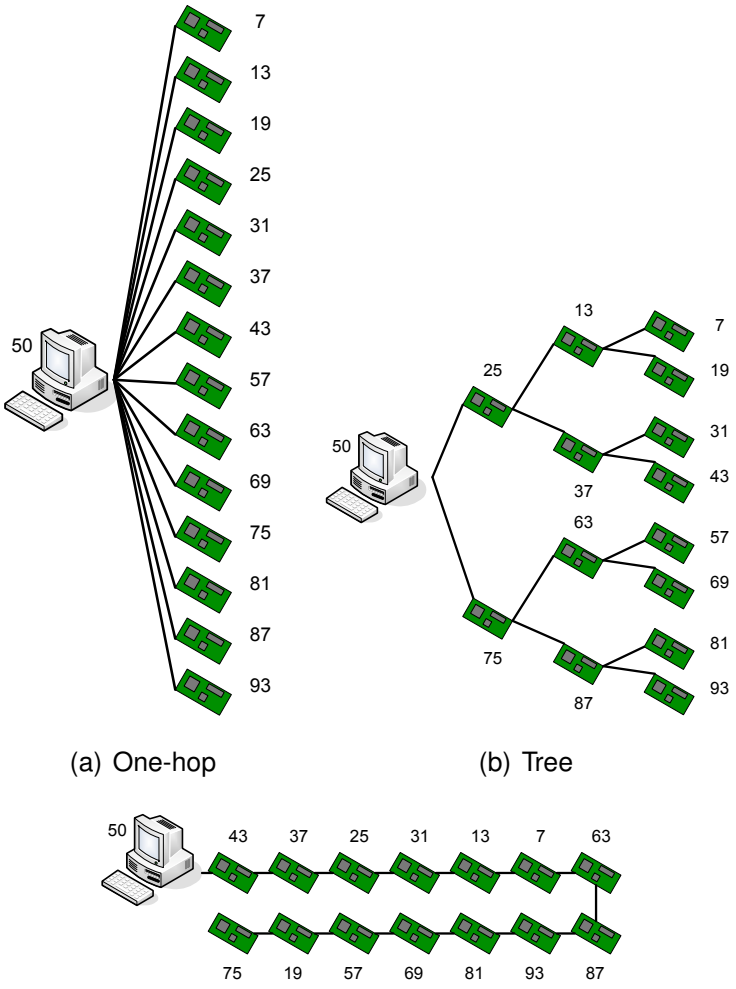



Figure 5.11: Network topology types.

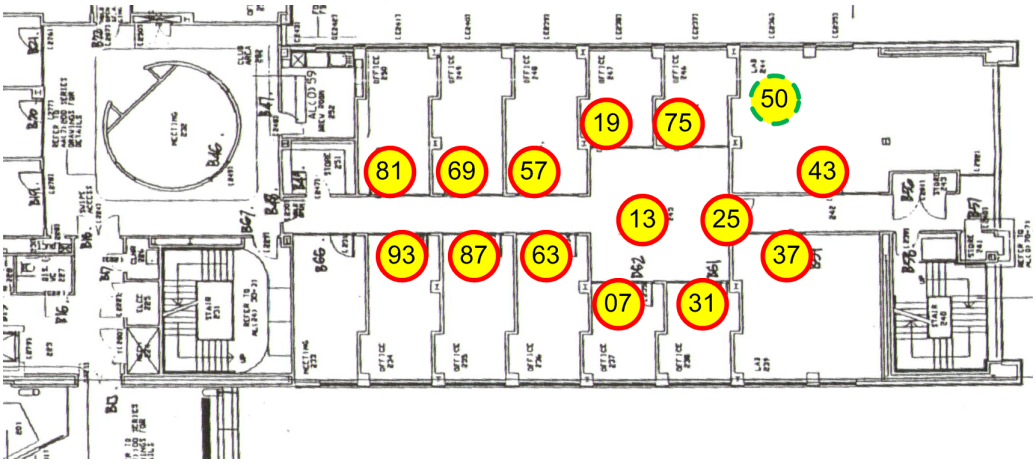



Figure 5.12: Office deployment map.

were controlled using broadcasts sent from the sink. Broadcasts were also used to synchronise the network time; the sink broadcasted its time value at the start of each experiment repeatedly until all nodes reported successful receipt. These broadcasts were sent directly from the sink to every node, without using the multi-hop network structure. This provided a $32$-bit global clock with about $10$ milliseconds of accuracy.

The transmission interval on the nodes, for sending reports and controlling retry mechanisms, was set to $250$ milliseconds with a positive *random jitter* of up to $50$ milliseconds. The jitter was necessary to avoid high-loss associated with closely synchronised nodes performing CCA simultaneously.

Each experiment comprised of $105$ tests, formed of $5$ tests for each combination of $7$ protocol variants and $3$ topologies. The sink initiated each test – equivalent to a re-keying round – and also reset the nodes, controlled node configuration and conducted time synchronisation. Each test was considered complete when each node had received at least one downstream (re-keying) message intended for that node and the performance data had been successfully recorded at the sink. The data was then processed using software for statistical analysis, with some stages automated due to the volume of data involved.

Each experiment was run using a *reliability function* that filtered messages with a specific probability, based on a pseudo-random number generator, in the receive function on each node. The experiment was first run at $100\%$ delivery reliability, followed by further experiments at $5\%$ decrements. It was found that the chain topology became infeasible once reliability dropped below $95\%$, so lower reliability levels were only run for the one-hop and tree topologies.

### 5.6.4 Key Transmission Overhead

The key-related transmission overhead is important as it demonstrates the actual number of transmissions rather than the theoretical minimum. The one-hop and chain topologies are first investigated as these give a view of the best and worst cases. The tree topology is then considered in more detail as it more closely resembles realistic networks. The chain topology cannot be discounted as unlikely as it might find uses in the future for the monitoring of structures with considerable linearity; examples include pipelines, national borders and

railways.

### One-Hop Topology

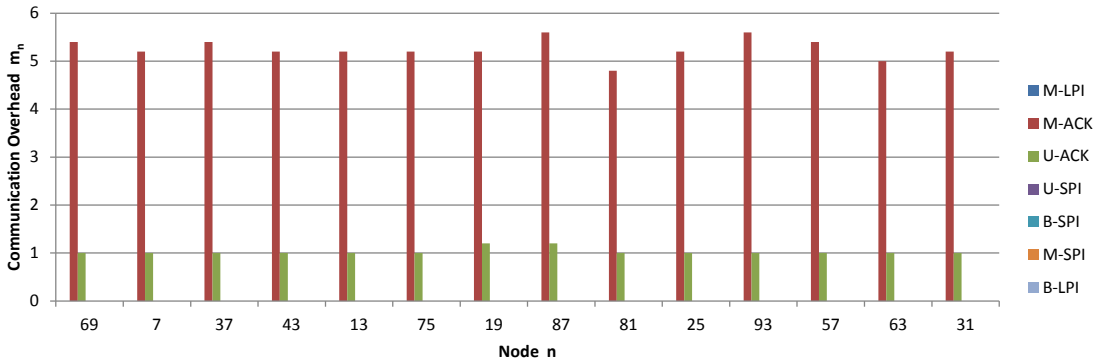



Figure 5.13: Average transmissions $m_n$ on each node (one-hop topology, reliability filter at $100\%$).

Figure 5.13 shows the average message overhead (number of key-related transmissions) $m_n$ for each mode on each node in the one-hop topology at $100\%$ reliability.

The one-hop scenario might be expected to deliver excellent performance as all of the nodes can be reached with a minimum of one transmission from the sink and do not need to make further downstream transmissions. However, the need to take dedicated acknowledgement (ACK) messages into account within some variants means that overhead is incurred, even when no loss is encountered. The SPI and LPI methods all result in no overhead as these do not involve dedicated ACK messages; the existing report messages are used to indicate success or failure.

Notice that the U-ACK method outperformed the M-ACK method. It might be expected that the number of ACK messages would be identical in both cases, but this is not the case. In the UKE method, downstream messages are sent *consecutively* to *individual* nodes from the sink. As each node receives its UKE message, it can send an ACK in normal channel contention conditions. In the multicast case, all nodes receive the *shared* downstream message *simultaneously* and then proceed to send ACK messages simultaneously. This causes *ACK collision*, resulting in loss of the acknowledgements. The sink continues to resend the multicast until it sees at least one ACK message from each node. Since all of the nodes resend the ACK messages again on receipt of the multicast message – they have no indication which ACKs were lost – the problem repeats until the round successfully completes.

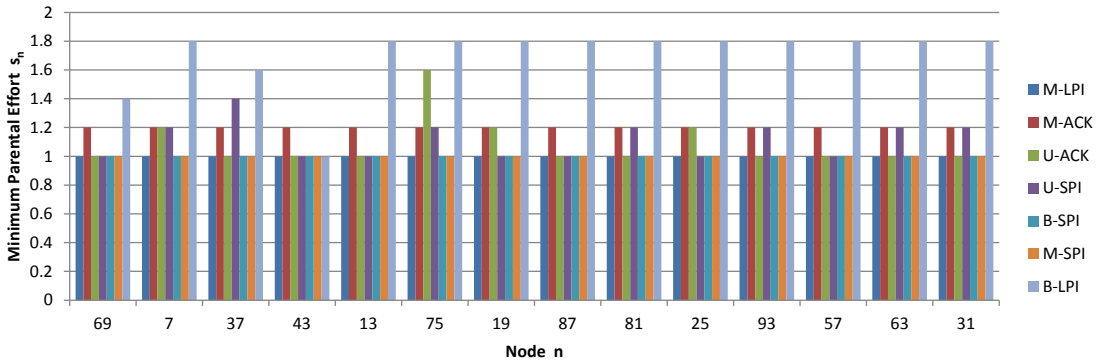



Figure 5.14: Required parent effort $s_n$ for each node (one-hop, reliability filter at $100\%$).

Figure 5.14 confirms this finding by showing the minimum number of *actually required* attempts needed for each node to receive the downstream message from the sink. This is the derived from the first attempt number seen by that node. In the case of the M-ACK method, only one downstream message was lost during the $5$ experiments for this topology. This is reflected by $s_n = 1.2$, indicating $6$ transmissions in $5$ experiments. Therefore the high overhead stems principally from the ACK mechanism.

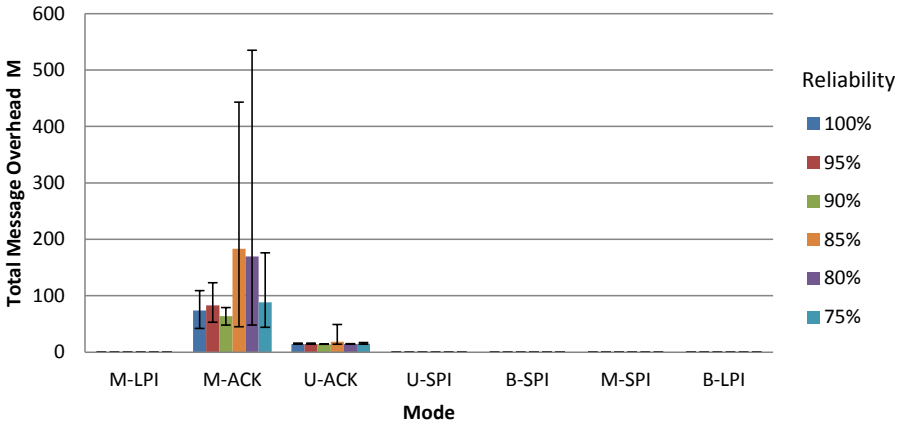



Figure 5.15: Comparison of the summed average transmissions ($M$) ordered by mode (one-hop topology).

Figure 5.15 shows the *total* message overhead $M$ for the different modes and reliability levels. Figure 5.16 shows the same data as a line graph ordered by reliability. Only M-ACK and U-ACK show any overhead in this topology; the other modes do not involve any additional, dedicated, transmissions by leaf nodes, even when loss occurs.

Notice that the M-ACK method is relatively unpredictable compared to the U-ACK method, which maintains a near constant overhead. The drop in overhead with reduced reliability is likely to be influenced by the ACK collision problem. If dissemination messages are lost, the

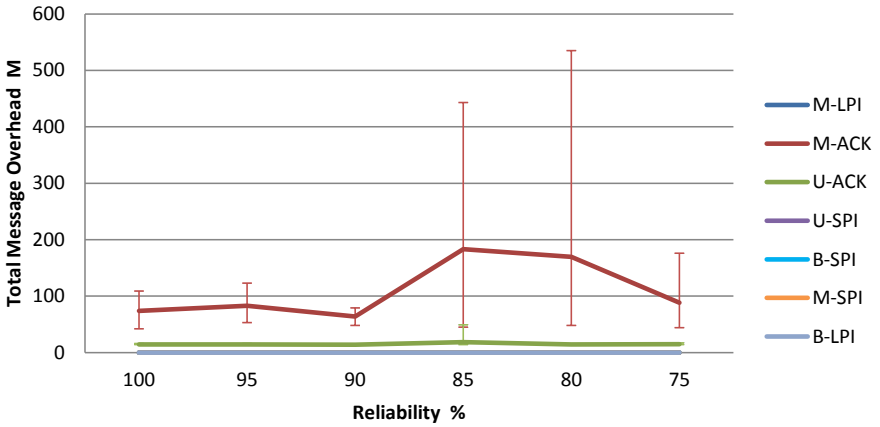



Figure 5.16: Comparison of the summed average transmissions ($M$) ordered by reliability (one-hop topology).

generation of ACK messages is reduced and less ACK collisions occur. This makes it hard to generalise the performance of M-ACK in lossy conditions.

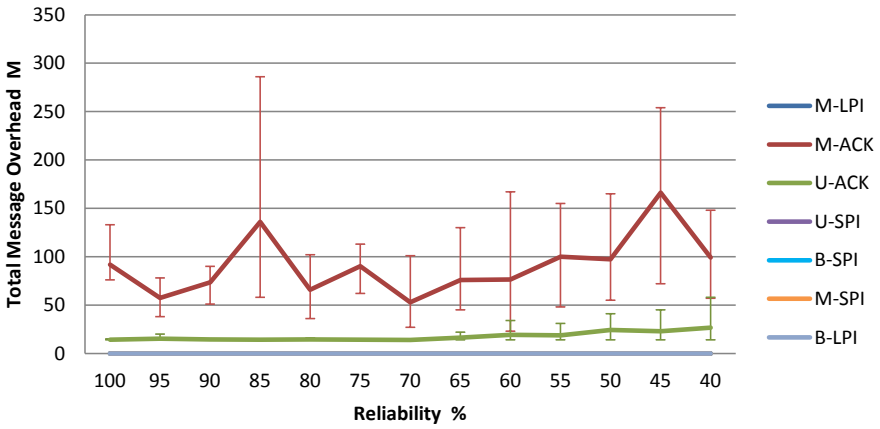



Figure 5.17: Comparison of the summed average transmissions ($M$) ordered by reliability (screened, one-hop topology).

Figure 5.17 shows the results when the same experiment was conducted in a screened chamber and able to run to $40\%$ reliability. Again, only M-ACK and U-ACK show any overhead in this topology. In this case it is clearer that the performance of U-ACK does begin to noticeably degrade once the reliability falls below $70\%$. However, the performance of M-ACK remains unpredictable despite a noticeable trend increase.

ACK based approaches are clearly inappropriate in a one-hop network as they incur high overhead even when no loss occurs. This problem is worsened by the larger number of children and would need further work to address; for example, a modified protocol design could reduce ACK collision by staggering ACK transmissions.

### Chain Topology

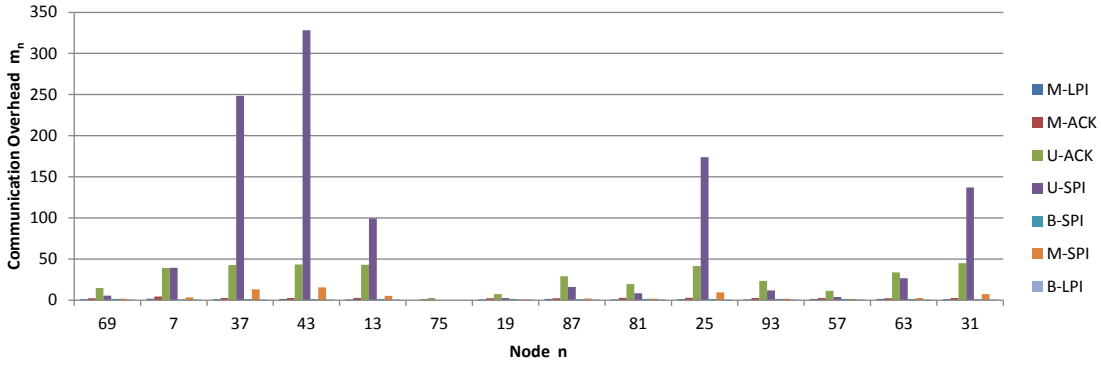
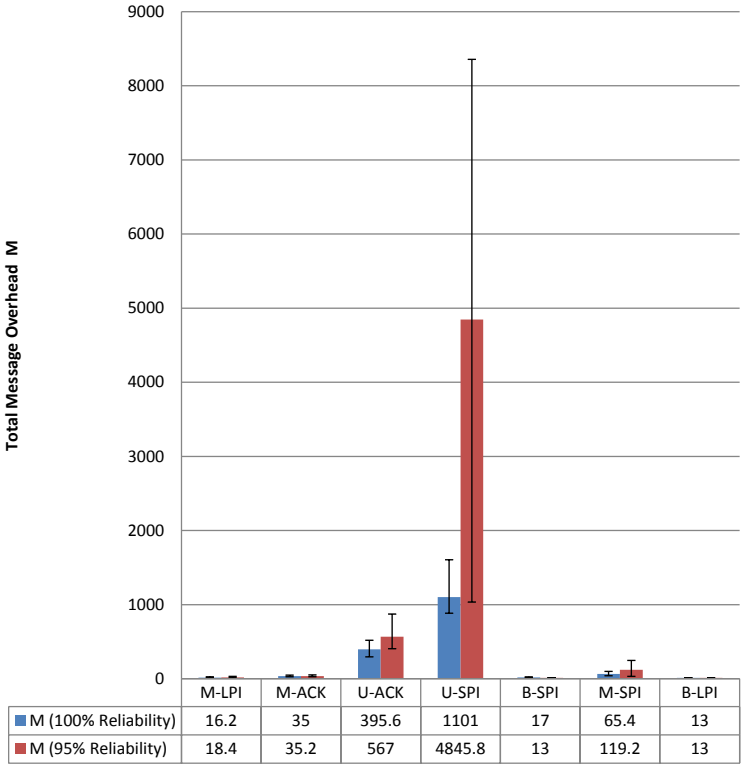



Figure 5.18: Average transmissions $m_n$ on each node (chain).




| | M-LPI | M-ACK | U-ACK | U-SPI | B-SPI | M-SPI | B-LPI |
|---|---|---|---|---|---|---|---|
| M (100% Reliability) | 16.2 | 35 | 395.6 | 1101 | 17 | 65.4 | 13 |
| M (95% Reliability) | 18.4 | 35.2 | 567 | 4845.8 | 13 | 119.2 | 13 |

Figure 5.19: Comparison of the summed average transmissions ($M$) ordered by mode (chain topology) with reliability filter set to $100$% and $95$%.

The chain topology represents the worst-case network topology. This should produce the worst-case communication performance due to the high number of hops in overlapping RF space, increasing channel contention and collision-related loss.

Figure 5.18 shows the average message overhead for each node and mode in the chain topology. Figure 5.19 shows the *total* message overhead for each reliability. Near-identical findings were encountered in the screened chamber experiment, with a slight increase in

message overhead for the UKE methods likely caused by the close proximity of the nodes.

The UKE methods are clearly the worst performers, with U-SPI requiring a thousand messages to the $14$ nodes, in the experiment. This increases by $340\%$ when reliability is dropped to $95\%$. U-ACK, by contrast, increased by $43\%$. At lower reliability levels, the experiment ran for an unacceptably long period such that it became infeasible to show results in the chain topology below that level. It is no surprise that this was the result; one node required no loss over $14$ hops, which is highly improbable in these conditions and especially when there is an additional $\frac{1}{10}$ probability of loss on each hop. The need to send acknowledgements over multiple hops is clearly ineffective and also contributes to the channel contention issue.

It is clear that the broadcast based approaches both perform the best in this topology, as shown in Figure 5.19. This is mainly because the broadcast approach is not forced to use the topology in the downstream direction and performance gains result from the overlapping communication range of each node.

Of the multicast approaches, M-SPI performs about twice as bad compared to M-ACK. This is caused by the retry mechanism; M-ACK only requires *one* hop to send an ACK whilst SPI requires several. M-LPI performs better, and similarly to B-LPI, as it is not penalised by ACK messages in general.

Unsurprisingly, there is a clear relationship between the distance to the sink and number of transmissions. In the case of U-ACK, unicast performance is approximately a hundred times worse than the methods based on broadcast. Nodes closer to the sink have a significantly higher overhead as is observed on nodes $43$, $37$, $25$ and $31$, which are respectively first, second, third and fourth in the chain.

Node $43$ also demonstrates the *critical node* concern (see Section 5.5.3); it has to send $320$ messages, which is far higher than the handful of messages sent near the end of the chain. Although this experiment did not use an extended-preamble MAC protocol, $320$ messages would cost $556$mAs in the scenario described in Section 5.5. This is equivalent to over $4$ executions of the elliptic curve scalar point multiplication. Clearly, even with the computational overhead of the SPM, the avoidance of unicast for downstream communication would be beneficial to that node.

Unicast is simply not feasible in such a network, even with just $14$ nodes. The envisaged

maximum feasible network size for UKE in a chain is therefore far lower than the $135$-node limit predicted in Section 5.5.4. This difference exists because the theoretical evaluation did not consider loss. In a real-world application, loss will occur both as a result of normal channel conditions and because of the channel contention caused by overlapping RF ranges, for example. It is therefore unsurprising that the actual limit is lower. The benefit of in-network recovery is therefore of considerable benefit as the chance of uninterrupted communication across many hops is low. Sink-based recovery mechanisms, particularly in UKE mechanisms, are clearly undesirable in this topology and motivate the use of BKE with in-network recovery mechanisms.

### Tree Topology

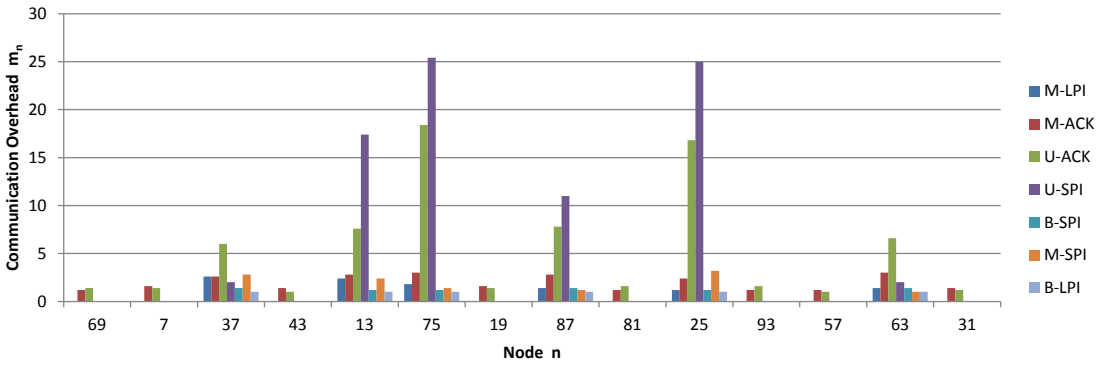



Figure 5.20: Average transmissions $m_n$ on each node (tree).

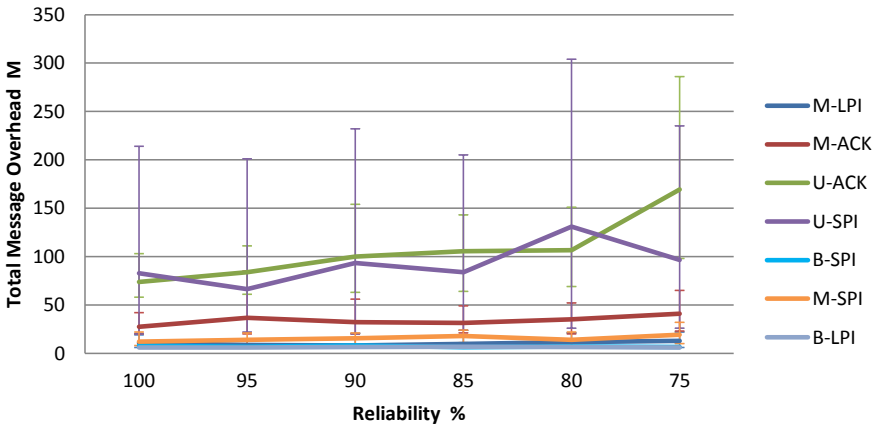



Figure 5.21: Comparison of the summed average transmissions $(M)$ ordered by reliability (tree topology).

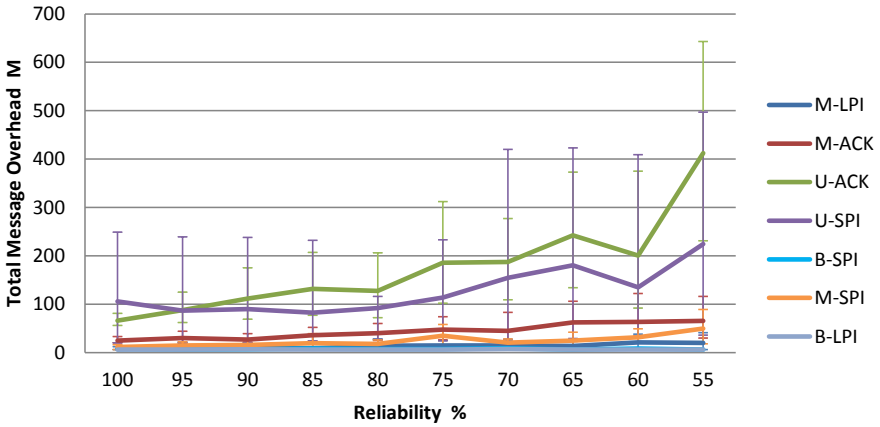



Figure 5.22: Comparison of the summed average transmissions ($M$) ordered by reliability (screened, tree topology).

The tree scenario allows a view of a more realistic network organisation and resembles the SecureTDRoute scenario. Figure 5.20 shows the average message overhead in the envisaged binary tree scenario for each node and mode. Figure 5.21 shows the total message overhead for each mode by reliability.

The performance ranking between the modes is generally identical throughout the levels of reliability, except for U-ACK and U-SPI, which swap places several times. The same ranking can be observed in Figure 5.22. Therefore the ranking is first discussed, then the impact of reliability is discussed second.

It is immediately clear that the two UKE methods are the worst performers. When investigated on a per-node basis in Figure 5.20, the performance is noticeably worse on nodes closer to the sink (such as $75$ and $25$) and the difference between U-ACK and U-SPI also widens. This is unsurprising given that any loss must be handled at the sink in U-SPI, incurring additional overhead close to the sink when messages fail to reach distant nodes. Notice that the highest overheads are found on nodes $25$ and $75$ that are immediately adjacent to the sink. Again, this highlights the effort imbalance with unicast approaches.

M-ACK involves at least a small overhead on all of the nodes due to the need to send an ACK even when loss has *not* occurred. Where no loss occurs, the M-ACK can perform slightly worse than U-ACK. This is noticeable on nodes $7$, $19$ and $31$ in Figure 5.20. Again, this can be expected due to the ACK collision problem identified earlier.

Of the packet inspection modes, all perform with zero overhead on leaf nodes. On other nodes, the overhead is generally lower than the ACK based approaches primarily due to the

lack of ACK messages. There are some similarities in performance between the different variants of the SPI and LPI modes. Note that the LPI methods always outperform, or roughly equal, the corresponding SPI method for that mode. This is to be expected in SPI modes; only the sink can retransmit and this results in multiple transmissions for some destinations. However, this is not very clear since the average number of hops is still quite small.

Also noticeable is that the broadcast methods outperform all others. This is because they utilise proper broadcast for the downstream direction and only use the topology-constrained pathways for upstream communication. Thus nodes can receive downstream messages faster, from more communication partners and produce less congestion.

The ranks of U-ACK and U-SPI swap when network reliability drops to $95\%$. This can be observed in both the office and screened experiments. The introduction of loss negates the benefit of U-SPI since loss requires retransmission from the sink, which introduces higher penalties compared to the link-layer approach U-ACK. All approaches generally degrade in performance below $90\%$ reliability, although the rate of increase in communication overhead is noticeably higher in the unicast cases.

Therefore, it can immediately be observed that unicast methods are outperformed by multicast and broadcast methods at all reliability levels, and, that unicast methods have worse degradation in lossy environments.

#### Key Transmission Overhead Findings

From these experiments, several points have been identified.

**ACK Collision** The need to send an ACK on receipt of a multicast message results in network congestion and loss due to simultaneous channel access. As a result, unicast works better than multicast when an ACK-based approach is employed. This issue could be addressed by using jitter in the ACK mechanism, but this could affect communication performance in other ways.

**Packet Inspection Reduces Communication Overhead** In packet-inspection based protocols there is no need to send an ACK when a message is successfully received. This prevents additional network congestion and allows some leaf nodes to incur no

overhead at all. Where recovery is needed, packet inspection is most effective at the link-layer.

**Unicast Performance Degradation** In a one-hop network, the unicast protocols can perform with zero overhead. However, this quickly degrades to an infeasible level as the number of nodes and hops increases. By contrast, multicast and broadcast protocols can perform with virtually no increase in local load.

**Broadcast Performance Gain** Broadcasting protocols benefit from topology 'ignorance' in the downstream direction allowing a single message to reach more nodes and reduce the need for increased transmissions. This is beneficial in environments where the communication radius of nodes overlaps nodes that are several hops away.

It is therefore clear that broadcast and multicast approaches offer the best solution in this scenario; particularly when coupled with link-layer packet-inspection recovery.

### 5.6.5 Key Transfer Delay

The time taken to update the entire network with new keys is relevant because it represents the time (a) where there is disruption within the network due to re-keying and (b) where the sink has to tolerate the use of old keys. The analysis provides a brief insight into the delays that can be incurred.

#### One-Hop Topology

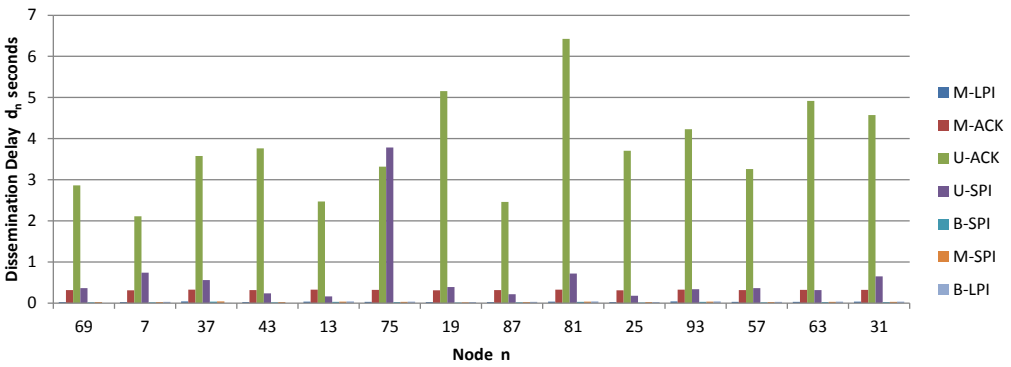



Figure 5.23: Average time delay $d_n$ on each node (one-hop).

The time delays in the one-hop topology are shown in Figure 5.23. The unicast schemes perform the worst in the one-hop topology, in contrast to their performance in terms of communication overhead. This is because the sink has to schedule the consecutive transmission of individual messages to each node as messages cannot be sent simultaneously. Notice how $d_n$ is markedly different for each node, indicating its position in the queue to receive a message.

U-ACK incurs the highest delay, with U-SPI working much better. The reason for this lies in the protocol design. The sink must wait for an ACK from each node before proceeding to send a message to the next node, U-SPI does not have this limitation as the sink can respond to report messages instead of waiting for acknowledgements. The sink can therefore proceed faster through the schedule. Even if the ordering of the retry mechanism was modified, such that it conducted retries after one attempt had been made for every destination, loss on any node would still extend the time in a similar fashion as the retry would then not be possible until after the end of the first cycle.

All the broadcast methods, and the multicast methods using packet-inspection, perform the fastest. The multicast M-ACK approach is affected by the ACK collision problem, incurring a performance penalty as a result of blocked resources both on nodes and in the radio spectrum.

The unicast time delay scales according to the number of nodes; this is a performance limit caused by the need for the sink to send the messages consecutively. This performance limit does not apply to broadcast and multicast mechanisms as they are not restricted by this scheduling problem.

### Tree Topology

Figure 5.24 shows the delays in the tree topology. Again, a similar effect caused by the unicast transmission scheduling can be observed since U-ACK performs the worst. As expected, U-SPI performs better because the sink does not have to wait for ACKs during the schedule.

The broadcast methods still perform the fastest, with the multicast M-LPI close in performance. The broadcast methods again benefit from their inherent design that is not con-

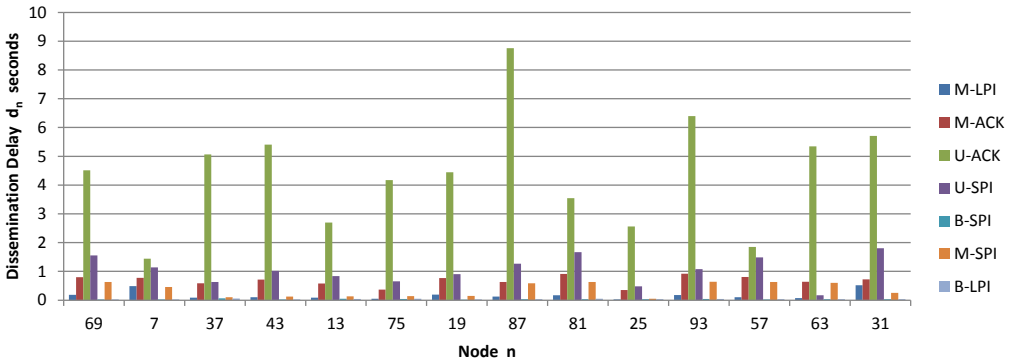



Figure 5.24: Average time delay $d_n$ on each node (tree).

strained by the network topology in the downstream-communication. M-SPI performs worse on some nodes due to the sink-based recovery issue.

### Chain Topology

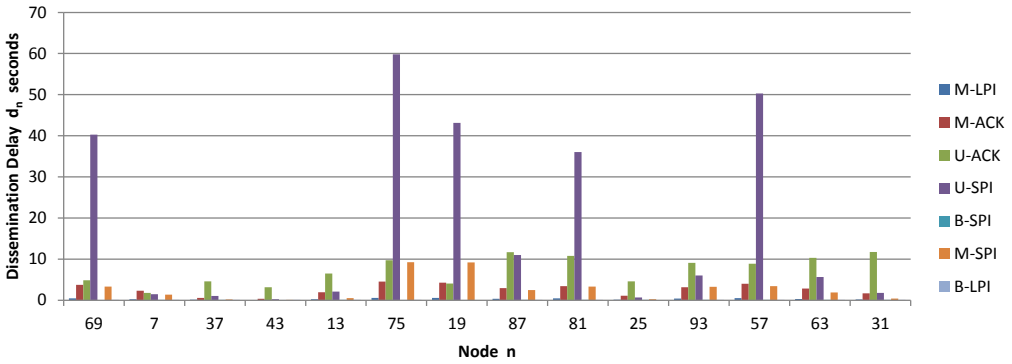



Figure 5.25: Average time delay $d_n$ on each node (chain).

Figure 5.25 shows the delays in the worst-case chain topology. The unicast schemes are still the worst performers, but the relationship between U-SPI and U-ACK has changed. This is again is due to the sink-based recovery mechanism, which leads to longer delays in lossy environments. B-SPI benefits from fewer hops in the downstream direction and therefore is generally unaffected by the topology. Notice how the dissemination time can now be measured in *minutes* for U-SPI.

### Key Transfer Delay Findings

Thus the following points can be observed:

**Unicast Transmission Scheduling** Because it is not possible to transmit a single message to all nodes, the sink has to transmit each message consecutively. This creates an unavoidable delay that increases with network size.

**Acknowledgement Overhead** ACK messages result in time delays and complexity since nodes have to be able to handle these messages, thus blocking resources.

**Sink-Based Retry Incurs Delays** Sending retries from the sink incurs a high delay penalty when the network has many hops. Local recovery mechanisms within the network operate far more efficiently.

**Broadcast Efficiency** Broadcasting will always reach nodes the fastest since it can bypass topology constraints both normally and during failure scenarios.

## 5.7 Efficiency Improvements

### 5.7.1 Alternative Cryptographic Mechanisms

BKE/D is a specific type of Broadcast Key Establishment, but other cryptographic approaches are possible with differing security properties. This section explores two alternative approaches that use cheaper cryptographic functions, but still result in confidential BKE.

Both schemes share the pre-distribution and broadcast elements of BKE as well as the goal of obtaining an identical shared secret $s_n$ at both ends; they differ only in function $f()$, function $e()$ and the type of keys used for $k_n$ and $j_n$. The first scheme uses a symmetrical block cipher. The second uses a hash function.

This section first introduces each scheme with the security differences and performance expectations. The computational cost $c_f$ of function $f()$, whether SHA-256 or AES-256, is then derived from the experiments in Section 3.2. In the case of the scalar point multiplication in BKE/D, the figure is taken directly as $c_f = 114$mAs.

**Broadcast-Symmetric-Encryption Key Establishment**

*Broadcast-Symmetric-Encryption Key Establishment* (BKE/E) uses a symmetric cipher to encrypt a sink nonce with a node key. BKE/E avoids the expense of Diffie-Hellman by using symmetric ciphers.

In the first phase, each node is assigned a unique node key that is shared by the sink and transferred offline. In the second periodic phase, the sink generates a nonce and broadcasts it to the network. The nonce is encrypted using the node key at each node and the output used to replace the shared secrets. The nonce is an unpredictable value and, in this case, is never re-used during the lifetime of a deployment. The bit-length of the nonce means that the range of possible values is considerably larger than that needed for the lifetime of the deployment.

**Phase 1** $N$ sink keys $e_n(\forall 0 \leq e < N)$ are generated. Each sensor node $n$ is configured with its $e_n$ and a table on the sink is populated containing all $e_n$. Phase 1 is carried out once only, offline:

1. All $e_n$ are calculated by the sink.
2. Each $e_n$ is stored on the corresponding node $n$.
3. All $e_n$ are tabled on the sink.

Because this is conducted offline, it provides confidentiality. An attacker cannot obtain or modify any $e_n$ and so will find it infeasible to imitate either party later on.

**Phase 2** The sink generates a nonce $x$. $x$ is distributed in the network using a single broadcast message. All nodes and the sink encrypt $x$ using the specific node key $e_n$. This process can be repeated periodically to set keys on all nodes.

1. The sink creates nonce $x$. $x$ must not have been previously used. $x$ is broadcast.
2. Each node recalculates the secret $k_n = \{x\}_{(e_n)}$ .
3. The sink recalculates all secrets $k_n = \{x\}_{(e_n)}$.

**Security** In phase 2 of BKE/E, nodes encrypt a broadcast *nonce* using a node key distributed offline in phase 1. Provided a secure symmetric cipher is used, an attacker should not be able to calculate the ciphertext (shared key) of a given nonce $x$ during the lifetime of that nonce. If the shared key is recovered, which is unlikely, he should then not be able to derive the node key used to generate it.

Nonces avoid *codebook attacks*. As nonces are unpredictable and unique, it is not possible for an attacker to build codebooks from observable traffic. Although the range of possible nonces will reduce over time, their unpredictable nature prevents an attacker from choosing to perform concentrated cryptanalysis for a specific nonce and then waiting to see the nonce re-appear. Also, at a protocol level, this design prevents replay attacks if the shared key is used for authentication purposes.

The purpose of a cipher is also to protect the confidentiality of the key, even if both plaintext and ciphertext are known. Thus, the scheme provides a similar one-way security property as Diffie-Hellman; an attacker still needs to carry out brute force attack to find the correct key.

The output from the encryption could be hashed to avoid these attacks, but a strong symmetric cipher should be able to resist brute force attacks.

It could be argued that this scheme provides little security benefit because the strength of the end-to-end cipher used with the shared secret might be identical to that used for key calculation. However, the exposure of the keys used for key calculation is lower since the key calculation occurs less frequently than end-to-end operations.

**Performance** AES has a block size of $128$ bits, thus function $f()$ involves two block encryptions using AES-256 to encrypt the nonce and produce material for a $256$-bit key. Whilst it is common to link these encryptions using a block mode, passing data from block to block to avoid cryptanalysis, this is omitted for the purpose of performance evaluation as little more than XOR operations take place.

A block encryption using MIRACL's implementation of AES-256 was found to cost $e_e = 0.0020976$mAs in Section 3.2. Thus $c_f = 0.0041952$mAs. This is significantly less than $c_f$ for SPM ($c_f = 114$mAs) and is also less than the cost of sending a single message using LPL with $100$ms epochs ($c_t = 1.74$mAs, see Section 3.3).

It is therefore not hard to imagine that BKE/E is much more feasible in energy terms.

### Broadcast-Concatenated-Hash Key Establishment

*Broadcast-Concatenation-Hash Key Establishment* (BKE/H) uses a hash function to hash a concatenation of a sink nonce and node key to produce shared secrets.

Similar to BKE/E, in the first phase each node is assigned a unique node key is shared by the sink and transferred offline. In the second periodic phase, the sink generates a fresh nonce and broadcasts it to the network. The nonce and node keys are used to replace the shared secrets, but a hash function is used rather than using a symmetric cipher.

**Phase 1** Phase 1 is identical to phase 1 in BKE/E.

**Phase 2** The sink generates a nonce $x$. $x$ is distributed in the network using a single broadcast message. All nodes and the sink concatenate $x$ with the specific node key $e_n$ and then apply a one-way hash function. This process can be repeated periodically to set keys on all nodes.

1. The sink creates nonce $x$. $x$ must not have been used before. $x$ is then broadcast.
2. Each node recalculates the secret $k_n = h(x \mid e_n)$.
3. The sink recalculates all secrets $k_n = h(x \mid e_n)$.

**Security** BKE/H concatenates the sink nonce $x$ and node key $e_n$, before applying a hash function to generate the shared key $k_n$. The security of the function needs to protect two keys: the shared secret $k_n$ and the node key $e_n$.

Since $k_n$ is a known output of the hash function, the attacker can search for $e_n$ in the event that $k_n$ is compromised. The design of BKE/H means that an attacker also knows part of the input, the sink nonce $x$. This is tolerable only if the security of the hash function does not degrade ungracefully when part of the input is known.

The security therefore relies strongly on the one-way nature of the hash function. Cryptographic hash functions are intended to make it very difficult to map outputs back to inputs in a short time. Unfortunately, some hash functions are considered ‘weak’ in that they can

be broken much faster than using brute force. BKE/E, for example with AES, and BKE/D, when used properly, do not currently have such a serious concern. This, combined with the issue of a partially known input, adds additional complexity to the choice of hash function.

**Performance** The MIRACL implementation of SHA-256 does not have a minimum size for the input, but it does have an overhead delay once every $512$ bytes. Thus function $f$ involves an initialisation cost $e_g$, followed by a per-byte cost $e_e$ to encrypt the $32$-byte nonce (see Section 3.2). Thus:

$$c_f = e_g + (32 \cdot e_e) = 0.019323 + (32 \cdot 0.0000608) = 0.0212686$$

BKE/H is thus nearly five times more expensive than the $0.004256$mAs required by BKE/E. Importantly, it is still cheaper than $1.74$mAs required to send a message and obviously cheaper than the $114$mAs needed for a scalar point multiplication. BKE/H is thus feasible in energy terms.

#### Security Differences

In BKE/D, the Phase 1 offline key material exchange comprises of the node's public key. For an attacker to perform a man-in-middle attack on Diffie-Hellman he must be able to view the existing public key and transmit his own to the sink. This allows him to later intercept the sink's public key and send his own to the node. Only if all of these are carried out can an attacker perform the attack. Since the first phase is offline, this attack is not possible.

In BKE/E and BKE/H, it is symmetrical node key material that is exchanged offline. As both of these schemes use a common symmetric operation, the attacker can perform a man-in-middle attack. The attacker could view the node key, to perform both sniffing and injection, or provide a false node key, to perform injection only. Whilst these attacks seem easier, the attacker still has to compromise the side channel. Thus to compromise these keys, it is still necessary to physically compromise the node or conduct cryptanalysis.

If the session cipher is compromised, a concern is on maintaining the security of the node private keys. In BKE/D, the attacker is left with the shared secret, which is useless to him as Diffie-Hellman is designed to protect the private key. In BKE/E and BKE/H, the attacker is left

with the ciphertext output of the cipher or hash function, respectively, in addition to the sink nonce. This may be a security risk with functions that are vulnerable to *known ciphertext* or *known plaintext* attack.

In some deployments it may be desirable to replace the private keys on the nodes. This may be to avoid cryptanalysis generally, or for more specific reasons. For example, if a node's private key is compromised then the security of all shared secrets computed using that key are compromised as well. Where the shared secrets are used for authentication, this is not such an issue as those keys expire. However, it is obviously of greater concern if the shared secrets were used for encryption; any messages encrypted using those keys would no longer be confidential. It may therefore be a requirement to limit this risk by replacing the node private keys.

The use of public key cryptography in BKE/D allows new private keys to be generated on nodes and the corresponding public keys to be exchanged over the network without privacy; it is a natural result of using Diffie-Hellman. It is sufficient security to use key material that is derived from an existing shared secret for authentication of the transmission of the public key to the sink; the shared secret will expire and the attacker will not be able to inject false public keys later. By contrast, if the symmetric secret node keys in BKE/E and BKE/H need to be replaced, it becomes necessary to provide confidentiality when those keys are transferred to the sink. Use of the existing node keys would leave the system to vulnerable to forward security issues, since the compromise problem outlined earlier would also compromise the new keys. Thus, BKE/D is stronger in this respect.

Hash functions do not have the same level of security as ciphers. Ciphers have one input for each output to permit decryption. Hash functions output a digest shorter than the input and thus produce collisions. Output values can be generated by *multiple* input values. This property can actually be a strength, if a strong hash function can be found. An attacker can obviously launch a brute force attack, storing inputs that generate an identical shared secret. But, then the attacker will have to repeat the attack with new data in order to reduce the size of the stored inputs. This requires at least as much processing compared to an attack against a cipher with only one result at the end. Note that these attacks require that the attacker first breaks each shared secret in order to obtain data for the cryptanalysis. This

is a challenge in itself.

### 5.7.2 Reduced Lifetime Ciphers

The relative efficiency of BKE and the short lifetime of the counter field may make some options that are normally considered inappropriate more appropriate.

Broadcast Key Establishment can make it feasible to replace the keys more regularly than may have been previously possible using unicast methods. This allows the use of *weaker* end-to-end ciphers in some circumstances because the shared secrets (and thus keys) are replaced more rapidly. It may also allow weaker key establishment ciphers to be used, provided that the node *static* keys are not compromised. These two issues are now discussed.

Shorter shared secret lifetime is possible if the shared secret need only remain secure for a short duration. They become useless to an attacker after a period as short as several hours if the keys are used solely for authentication purposes. There is little benefit in providing thousands of years of security, when a security level of several days would be more appropriate and cheaper. Obviously the requirements are different for encryption.

A good cryptographic cipher is resistant to all attack except for a *brute force attack*. A brute force attack involves testing every key. To illustrate the practicality of such an attack, consideration can be given to the time required to simply generate all the key combinations, without including the computation needed to test each combination.

Assuming a single machine can generate combinations at $1$GHz ($h = 1000000000$), with $s = 31536000$ seconds in a typical year, then the number of years $y$ required to generate all combinations of a $b$-bit key is:

$$y = \frac{2^b}{hs} \tag{5.4}$$

A brute force attack thus takes about $3 \times 10^{60}$ years against a $256$-bit key, about $10^{22}$ years against a $128$-bit key and about $584$ years against a $64$-bit key. By using parallel computing, such an attack can complete faster; for example, if an attacker had access to a network of a hundred thousand machines, then the time taken could be reduced by a hundred thousand. Whilst the $64$-bit key would be broken in a few days, the $128$-bit key would still have $10^{17}$ years

of security.

The Universe is believed to be about $14$ billion ($1.4 \times 10^{10}$) years old [122], which is shorter than the time required for both $256$-bit and $128$-bit keys, even with parallel computing. It could therefore be argued that a $256$-bit key is currently excessive, given that there are no reports of unmodified $128$-bit AES being broken faster than a brute force attack. Although an increase in the number of parallel processors is possible by designing custom hardware, the sheer scale required makes such an attack infeasible for the average attacker. There are developments in fields such as quantum computing that may significantly reduce the duration of such an attack, but these have yet to be realised.

Attackers are therefore forced to consider side channel attacks to obtain information about the operation of the device or its key generation. Due to the use of offline key exchange and independent power sources on the sensor nodes, some security against this is achieved.

The shared secret need only remain secure until the next round of BKE is initiated by the sink. a few days of worst-case security may be acceptable under those circumstances. Weaker ciphers such as DES, which allows a $56$-bit key, might therefore still be considered depending on the abilities of the attacker. Granjal et. al. have shown [35] that the use of 3DES can result in lower energy consumption when compared to AES. Another study [123] evaluated algorithms such as Skipjack, XTEA and Twofish.

It is also possible to choose a weaker ECDH variant, but this may necessitate key-pair replacement. This is feasible in security terms since once the key-pair has been replaced, the nature of ECDH means that there is forward security provided that the underlying authentication mechanism is secure *at the time of re-keying*. Obviously there is a higher communication overhead involved as the new key material has to be sent to the sink.

Observe that the use of one-way hash functions can protect the node secret used to generate end-to-end keys if there is a concern about the node secret confidentiality. Although the cipher used to generate keys and the one-way hash function need to be very secure, these need only be used once per key update and can thus be more expensive than that used for more regular operations.

## 5.8 Denial-of-service and Resource-draining Attacks

By introducing arbitrary values during the key broadcast phase, an attacker can cause genuine nodes to recalculate their shared secrets. This does not allow the attacker to inject data at the sink, but it does cause three problems.

Firstly, the nodes replace real keys with *false* keys preventing authenticated communication (*denial-of-service*). Second, the nodes expend valuable resources in calculating the new shared secrets (*resource-draining*). Third, nodes disseminate the false broadcast throughout the network, causing the problem to spread.

There exist several methods for protecting BKE from denial-of-service by adding authentication to the broadcast. Unfortunately, each of these methods is also vulnerable to attack. These attacks stem from the limited resources provided to WSN nodes, so acceptable approaches for other types of network do not apply.

In physical intrusion detection systems it is useful to be aware of *electronic attacks*. Alarms can be generated or security personnel can be sent to the site of attack. Thus, there might be little motivation to incorporate broadcast authentication. Other systems, however, may benefit from such protection or additional forensic information might be useful.

The first option is *public key cryptography*. If the sink *signs* the broadcast with a private key, nodes within the network can authenticate that signature by using the corresponding public key. This method avoids the establishment of false keys, but it still leaves the nodes vulnerable to resource-draining attacks (via the signature verification function) as well as at least doubling the cost of Phase 2.

Another option is to use a *time-delayed broadcast authentication* protocol such as $\mu$*TESLA* [70]. The basic principle of $\mu$TESLA is that data is broadcast, cached by all nodes, and then authenticated using a key broadcast at the end of an epoch. The main issue with $\mu$TESLA is the cache[25]. It is impossible to determine if a message is genuine until the end of the epoch, so there is potential for an adversary to *flood* the cache preventing the delivery of genuine broadcasts. An adversary can thus prevent key establishment from taking place by preventing message authentication.

A derivative of $\mu$TESLA is possible by directly broadcasting keys from the $\mu$TESLA key

[25]Time synchronisation is also required, which may in itself be a weakness.

chain; these could then be used with the alternative symmetric BKE methods described in Section 5.7.1. Since the keys are used directly, there is no broadcast payload to protect and nodes would be able to immediately use the new key without depending upon a cache. This approach would allow for broadcast authentication, provided that the full key chain could be computed by the sink in advance. Any attacker would need to lack the computational power to break the hash function at high speed. The design of public key cryptography means that it is not feasible to generate a usable public key without first knowing the corresponding private key; a key chain approach would therefore not be feasible for use with BKE/D. For example, when the sink generates the key chain by hashing one public key into another, the sink would not be feasibly able to compute the corresponding private key. The approach may, however, be viable for use with BKE/E or BKE/H.

Finally, symmetric encryption could be used to deliver the broadcast, but it would require the sharing of symmetric keys between *multiple* nodes in advance. This would be self-defeating as it would lead back to the very key distribution approach that this work aims to avoid.

The next chapter investigates the use of physical communication properties to exclude attackers based on location and this is one approach to obtaining link-layer authentication without using keys. Although the proposed method is a unicast mechanism, the increased overhead of using unicast instead of broadcast *on each hop* may be an acceptable overhead as it scales gracefully. For example, rather than sending a single broadcast, each node might need to send a unicast message to each *immediate* child. This is still an improvement over the sink needing to send a unicast message to *each and every* descendant.

## 5.9 Summary of Findings

Key distribution in a high security scenario has to support end-to-end authentication. Symmetric ciphers need to be used for efficiency reasons. Therefore, a key management protocol supporting confidentiality must be provided.

Such protocols must be able to handle the limitations of the WSN platform by limiting energy use, reducing communication, avoiding excessive computational and handling mes-

sage loss gracefully. Of the existing schemes available in WSNs, none efficiently provide for these requirements.

Many WSNs comprise of a sink and a large number of nodes. All of the end-to-end communication is between nodes and the sink. The use of a single broadcast message to set *individual* keys, shared with the sink, is an attractive means to reduce communication overhead. This in turn can assist with the other problems. For example, the use of a single broadcast allows for efficient in-network caching of key material, which is beneficial when compared with the loss-recovery burden required by unicast methods.

The use of Diffie-Hellman to provide the underlying key establishment allows the exploitation of its natural confidentiality mechanism. The elliptic curve variant allows for shorter keys and reduced computational time; these reductions benefit a WSN by reducing communication and computational overhead and are possible without sacrificing security strength.

BKE/D avoids the man-in-middle vulnerability in Diffie-Hellman by conducting half of the exchange offline. The use of shared sink key material and individual key material on the node is similar to some modes found in the commonly deployed TLS protocol. This provides some security assurances.

The communication requirements of BKE mean that a less complex routing protocol can be implemented that need only support two types of message: sink-initiated broadcast and sink-bound unicast. The necessary routing data on each node can be reduced as there is no need for node-to-node unicast.

The most expensive operation in elliptic curve cryptography algorithms is the scalar point multiplication. This causes difficulty in task-based operating systems as resources are temporarily blocked for many seconds. The use of task pre-emption is a useful mechanism to handle this problem.

Comparison with the traditional unicast approach depends on the cost of the cryptographic key establishment function and the cost of message transmission. Initially, BKE is more expensive than the unicast cases, but grows linearly. In the unicast protocols, the growth pattern depends on the network topology, so there is a benefit crossover point that differs depending on the topology and cost. The scalar point multiplication in BKE, although slow in the evaluated case, delivered better overall energy cost performance in chain topolo-

gies of over $135$ nodes compared to UKE.

Of concern is the energy cost on a *critical node*. Such a node is adjacent to the sink and provides the sole route to the entire network. In this case, BKE has a static minimum energy cost while the unicast case has a linear increase. Therefore, there is also a crossover point in this case where BKE becomes beneficial. The cost of the cryptographic function defines this point. In the case of BKE/D, as implemented, this crossover occurs at $67$ nodes. If the cryptographic function is sufficiently optimal, BKE/D becomes effective in smaller networks.

A real experiment was needed to obtain realistic communication performance data. Performance in terms of transmissions, delay and loss depends on many factors. These include the protocols, implementation, system limitations, radio interference and physical interference by people. These experiments found the broadcast methods to significantly outperform unicast schemes in normal and lossy environments. Unicast schemes, in particular, cannot benefit from simultaneous dissemination[26]. The delay in keying the whole network is therefore fundamentally related to the transmission schedule.

An effective method to provide reliable broadcast is to inspect report messages to check to see if the broadcast was fully propagated. Any loss can be handled by retransmitting the broadcast from any node, which is feasible as all nodes share the same broadcast. In the practical evaluation, this method was found to outperform methods based on dedicated acknowledgement messages in most environments.

Because BKE involves the pre-establishment of secrets, there is no need to necessarily use Diffie-Hellman. DH was intended for use with nodes that have no such opportunity. Therefore alternative key establishment functions can be used that are based on symmetric functions. These provide significant performance gains but have different security properties.

Various optimisations are possible. BKE provides the opportunity to re-key the network at more regular intervals. This allows the use of cheaper ciphers for end-to-end security when they are used for authentication rather than encryption. The risk of faster compromise of the shared secret is acceptable due to the regular re-keying and limited useful life of the keys.

Finally, BKE does not have a built-in authentication mechanism. This is not necessary for its core functionality as the node keys remain confidential and an attacker cannot establish

[26] Obviously in scenarios where the keys need to be replaced on subsets of nodes, this is not such an issue.

a shared secret with the sink for impersonation purposes. However, an attacker can exploit this to waste resources and disrupt communication. Adding authentication to a broadcast protocol in a resource-constrained environment is often self-defeating. This motivated the remainder of this thesis to investigate physical-layer security in order to better protect from such attack.

## 5.10 Conclusion

This chapter identified that there is a strong motivation for an efficient key establishment mechanism in high security WSNs. The concept of Broadcast Key Establishment (BKE) was introduced that uses a single broadcast message to set individual keys. This reduces communication requirements, allows more resources to be used by the application, better aligns with WSN communication patterns and balances energy consumption relating to key management.

The Diffie-Hellman-Broadcast Key Establishment (BKE/D) was introduced as a specific implementation of the BKE principle. The public key cryptography elements of BKE/D were found to provide the necessary security strength and are feasible on the constrained computing platform. A theoretical evaluation firstly found that BKE/D provided benefits to network lifetime due to the reduced communication overhead and better balances resources in the network. A practical evaluation supported those findings. Potential alternatives to the Diffie-Hellman variant and issues relating to resource-drain attacks were discussed.

The chapter has found that the concept of BKE is effective, secure and feasible in real WSN deployments. However, the shortcomings of cryptographic protection were also identified, motivating the following work on physical layer security.

# Chapter 6

# Distance-Based Message Authentication

**Cryptographic authentication is problematic in WSNs: it is susceptible to key compromise, requires key management and can be a target for denial-of-service and resource-drain attacks. Providing additional authentication at the physical layer can help to protect against these issues. A new generation of WSN communication transceivers are now available that support time-of-flight distance measurement. This chapter investigates the exploitation of this feature for authentication purposes. By inseparably integrating distance measurement with message transmission in certain environments, it is possible to authenticate messages based on distance rather than relying on keyed authentication. Physical access control can be utilised to enforce a secure zone where attackers are excluded. The electronic protocol then rejects messages sent from outside that zone. This introduces a significant hardware-level challenge for an attacker as they need to defeat the distance measurement. Since messages are rejected early, before any cryptographic algorithm is executed, exposure to cryptographic attack is significantly reduced.**

## 6.1 Motivations

Distance-Based Message Authentication (DBMA) has several motivations relating to the work described in former chapters. These centre on areas such as key compromise resistance, authentication cost under normal and attack circumstances, key management overhead, resource depletion attack and the use of existing physical limitations – like barriers

and rules such as the speed of light – to strengthen defence.

The existing security schemes available for WSNs were reviewed in Chapter 4; most are based on *cryptography* and provide good protection only if the keys remain secure. Once a secret key has been compromised, it can be used to inject messages into the network. Network security thus degrades when keys are obtained by an attacker. This problem is particularly damaging if the keys are used at the link-layer; an attacker can potentially inject messages using a compromised key, those messages are then provided with replacement message authentication codes by peers and forwarded through the network. This was discussed in detail in Section 4.2.3. It is therefore desirable to either avoid keys completely or to provide additional protection. Such an approach also goes a long way to offering credible protection against cryptanalysis, which is an *infeasible* threat rather than an *impossible* threat. For example, cryptanalysis is possible if sufficient computational power and optimisation is available. By contrast, Distance-Based Message Authentication is based on the measurement of distance that, depending on the implementation, may be impossible to defeat with current technology.

Physical intrusion systems are also predominantly protected using cryptography, as Chapter 2 shows. In the best case, sensors sign report messages using a secure key. Although considerable effort is possible to implement tamper resistance and secure cryptographic methods, this protection is meaningless should keys be compromised.

The difficulty in implementing cryptographic algorithms within constrained systems is a known issue, see Chapter 4 for examples. Existing work aims to improve efficiency of cryptography by employing dedicated hardware or optimised algorithms [22, 62, 124, 37]. This makes cryptographic approaches feasible but the situation is still less than ideal; especially when resources are so limited that they can be a target for attack.

The management of keys in a sensor network is also not straightforward since it requires the secure generation and distribution of keys for all nodes. This can consume scarce bandwidth and energy resources, as investigated in Chapter 5. Avoiding this overhead and the associated security risks is beneficial.

Finally, denial-of-service attacks against security protocols are a worthwhile attack vector in WSNs due to the minimalist nature of the nodes. An attacker can carry out small-effort

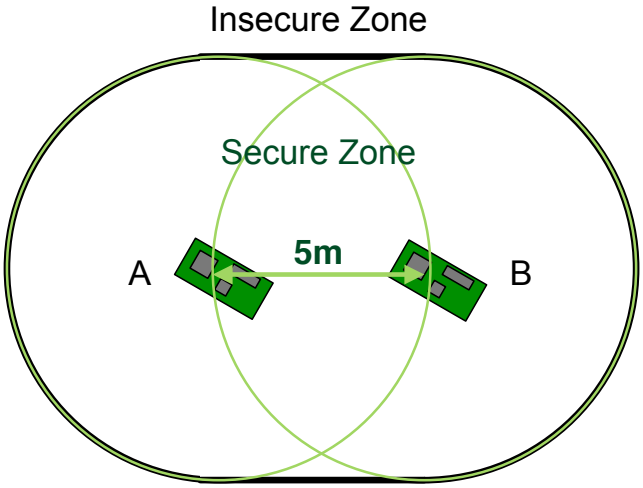



Figure 6.1: DBMA secure zone.

attacks that are insignificant in conventional networks, but cause significant problems for WSNs. One example is the attack against resources used to carry out message authentication; by injecting unauthorised messages, the attacker can force the use of cryptographic algorithms before messages can be discarded. An additional layer of security is thus desirable.

## 6.2 Principle of Distance-Based Message Authentication

DBMA exploits the new generation of communication transceivers that support distance measurement. In networks, where a secure zone can be enforced to prevent entry by an attacker or his devices, the *distance* between nodes is a useful authentication parameter. It is most useful if it can be measured during, and inseparably from, message transmission. This concept is called Distance-Based Message Authentication (DBMA).

Consider a network with two nodes, $A$ and $B$, that have the Euclidean distance of $5m$. A *secure zone* could be constructed, with a guarded barrier[27], using the union of two circles with radius $5m$ drawn around $A$ and $B$, as shown in Figure 6.1. Using DBMA, nodes $A$ and $B$ can reject communication from further away than $5m$ by using a set threshold. Thus an external attacker is unable to inject messages and this protects against attack.

DBMA is obviously only useful in scenarios where potential attackers can be excluded

[27] There are issues concerning accuracy which must be considered when placing the barrier, this is discussed in Section 6.9

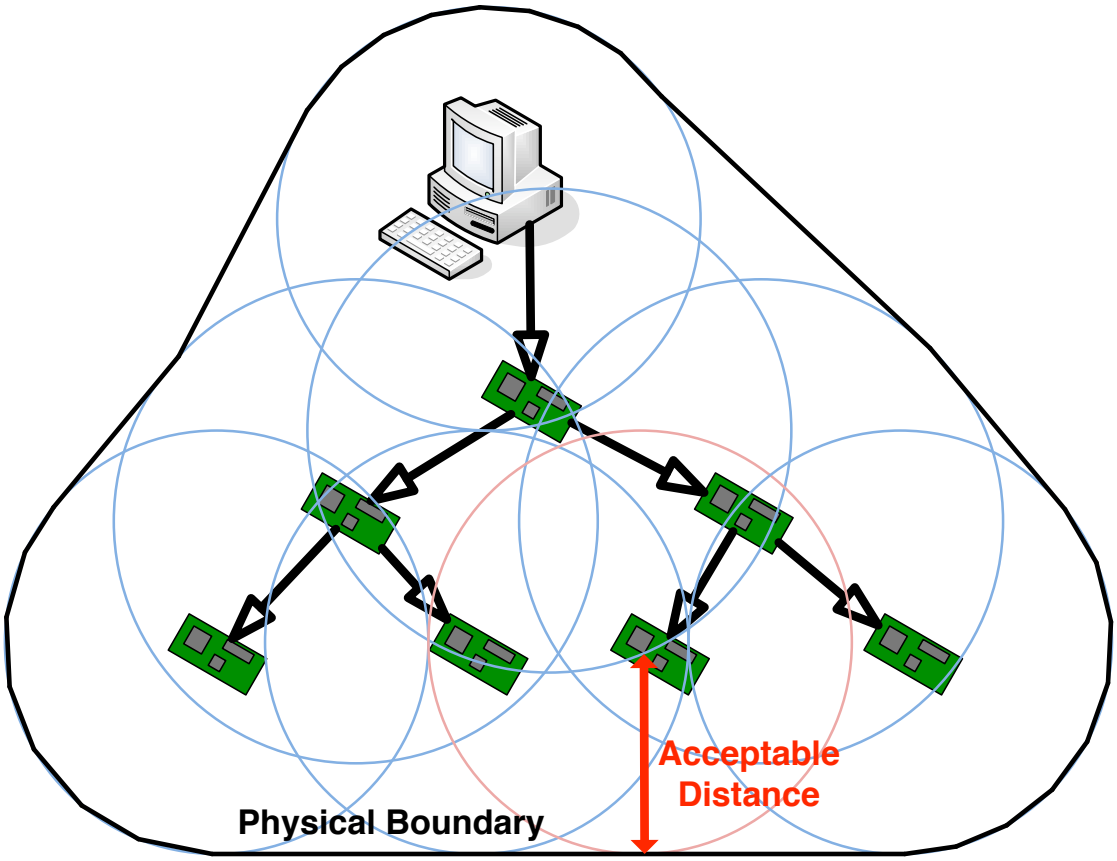



Figure 6.2: DBMA secure network.

from the WSN deployment area. This is certainly not possible in all WSN deployments; however, deployments do exist where physical access is restricted and monitored. Examples include some physical intrusion detection system scenarios (see Chapter 2), but can also include military and industrial deployments. Figure 6.2 shows how a physical boundary could be erected around a deployment.

In these scenarios DBMA prevents injection of messages from attackers located outside of the WSN deployment area either as an additional layer of defence to increase protection levels or as a complete replacement for cryptographic authentication mechanisms. DBMA does not require costly ongoing key management and this chapter shows that it can reduce the cost of rejecting unauthorised messages.

## 6.3 Research Problems

Any implementation of DBMA will need to satisfy the following properties. Several of these properties relate directly to security, whilst others are related to accuracy and integration.

**Lower Bound** An attacker should not be able to reduce the distance measurement, as this would make the concept of a boundary meaningless.

**Forced Participation** The attacker must be forced to participate in distance measurement and not be able to fool the receiver into conducting distance measurement with a different, genuine, node.

**Inseparable Integration** The exchange of a message must be integrated such that it is not possible for an attacker to send a message without also participating in the distance measurement process.

**Sufficient Accuracy** A sufficient resolution is needed to ensure that the system can positively identify whether a node is inside or outside the secure area. The accuracy also has an impact on the necessary secure zone area requirement, as discussed in later sections.

**Feasible Area Requirement** The secure zone area requirement should be as minimal as possible since maintaining larger plots of land is expensive.

**Feasible Energy Overhead** The energy cost of the scheme should be comparable to traditional cryptographic authentication and out-perform its energy cost when under attack.

**MAC Protocol Integration** DBMA makes extensive use of the lowest layers, it should be easy to integrate it with the MAC and PHY layers.

These problems are handled in the coming sections. Section 6.5 addresses the problems of inseparable integration and secure distance measurement by proposing the Round-Trip-Time Message Authentication Protocol (RTTMAP). The analysis of the feasibility of the energy overheads, security and MAC integration is the role of Section 6.6. The accuracy of the system and the impact on the safe area requirements is the topic of Section 6.9 that uses an implementation to obtain accuracy data and considers optimisation methods.

## 6.4 Secure Ranging

Cryptography is not the only approach to message authentication possible in a WSN. *Physical layer security* has been shown to provide enhanced security. One example [100] integrates keys with error correction codes. Others can be found in Section 4.4. By applying security at this layer, it is harder for an attacker to successfully deliver messages to genuine nodes. It is thus much harder to attack cryptography applied at the higher layers.

*Secure ranging* is an established WSN research area for positioning applications [101] and smart-card proximity verification (see Chapter 4). It has not been previously used for the purpose of message authentication in WSNs.

In order to be considered for use in DBMA, a secure ranging technique needs to satisfy two criteria. Firstly, it must not be possible for an attacker to participate in a measurement and reduce the distance measurement below that in reality. Second, the technique must be applicable without involving third parties and negotiation; this would be self-defeating since additional communication would be required and this too would need to be secured.

Whilst it is possible to use *ultrasound* [125] for secure ranging, obtaining range measurements using the radio transceiver eliminates the requirement for additional security hardware and allows transmissions to be coupled with ranging. There are several methods that can be used to measure distance using radio frequency (RF) methods.

**RSSI** Received signal strength indication (RSSI) is generally indicative of distance and is widely supported. However, RSSI is widely regarded as unreliable [104], and in the context of DBMA, it is unsuitable. An attacker can increase transmission power at will. Therefore, an attacker is able to artificially decrease distance measurements and appear to be located in the secure area. Countermeasures exist if clusters can be used [126], but this is not feasible in the context DBMA since it would require multiple parties and negotiation.

**NFER** Near-field electromagnetic ranging (NFER) exploits the gradual phase convergence of the electric and magnetic waves within half a wavelength from the transmitter. This can be measured [127] and used to determine range. It also has the interesting property that both fields become phase matched once the signal has propagated more

than half a wavelength from the transmitter. Thus there is no repetition that might be exploitable by an attacker; for example, he cannot appear to be within half a wavelength if he is not physically located within that region. Unfortunately, for the same reason, NFER is only usable if nodes can be positioned within half a wavelength of each other. The common frequency for WSN devices is $2.4$GHz, which has a wavelength of only $12.5$cm. It is not feasible to place devices this close. A frequency in the HF band is therefore necessary, such as $7.5$MHz with a wavelength of $40$m. NFER has the advantages in that it can be directly measured by the receiver and is sent from a single source. To be usable in a security setting, it has to be proven that an attacker cannot modify the phase relationship at the transmitter and that the attacker cannot remove the non-repetition property. NFER also has more specific antenna and receiver requirements, such as separate phase measurement of the two fields; these functions are yet to be provided in WSN class hardware.

**Timing-based** Timing-based schemes fall into three areas, all based on the near-constant propagation delay of a radio signal. Although radio signals can be delayed, they cannot currently be accelerated beyond the speed of light limitation. This gives the approach useful security properties. There are three general approaches:

**ToF** Time-of-Flight (or 'ToF'), measures the propagation delay between message transmission and message reception to obtain the range. ToF requires that the clocks of both nodes are synchronised, which is difficult to achieve securely and with sufficient accuracy. ToF also requires that the sender include a time-stamp in each message, therefore requiring trust. Manipulation is therefore possible.

**TDoA** Time-Difference-of-Arrival (or 'TDoA') involves the transmission of pulses from multiple synchronised transmitters. The receiver can measure the difference between the arrival of each pulse and use multi-lateration to determine position. TDoA has been employed in the Loran-C marine navigation system [128] since World War II. Loran-C measures the phase difference between these pulses to obtain a position. Sallai et al. [129] have implemented this in a WSN. TDoA does avoid the need for the time synchronisation between the sender and receiver; however, the pulses need to be sent from multiple transmitters or anten-

nae. Given that using multiple nodes to send a message is self-defeating – secure link-layer negotiation would be needed – and that having multiple antennae sufficiently spaced is impractical, TDoA cannot be considered. TDoA can also be implemented in reverse, where a single transmission is measured by multiple parties; unfortunately, this is still self-defeating due to the need for multiple receivers. It might be acceptable in scenarios where single-hop communication and multiple receivers are possible.

**RTT** Round-Trip-Time (or 'RTT') is similar to ToF. In RTT a PROBE message is sent to the other node and returned as a RESPONSE message. A timer is started when the PROBE is transmitted and then stopped when the RESPONSE has been received. The delay is taken directly from the timer, thus RTT does not require time-stamps or clock synchronisation. RTT is thus harder to manipulate and can therefore be considered since it only involves the communicating parties.

RTT exploits the interesting property that RF communication operates using a fixed propagation speed, the speed of light; this cannot be accelerated by an adversary due to the laws of physics. Although this property prevents acceleration, it is still necessary to ensure that a malicious responder cannot respond early. Such a response would reduce the distance measurement. Part of this problem can be eliminated by including a *nonce* (unpredictable value) in the PROBE message that must also be included in the RESPONSE. This principle, known as *secure round-trip-time*, has been applied for other purposes [130], but not for message authentication in WSNs. The other part of the problem lies in the transceiver and low-level implementation. These are implementation issues and do not have an impact on the general idea, but they must be addressed in production environments.

Older WSN standards using conventional modulation techniques such as *phase-shift-keying* (PSK) or *frequency-shift-keying* (FSK) did not perform well in RF ranging as they struggle to identify pulse edges in reflective environments, which is a key requirement when measuring time delay. The newer *IEEE 802.15.4a* standard, for use in wireless sensor networks, includes a number of new modulation amendments. These include *ultra-wide band* (UWB), *chaotic spread spectrum* and *chirp spread spectrum* (CSS). These technologies offer a number of benefits that improve radio ranging. CSS is particularly interesting because it

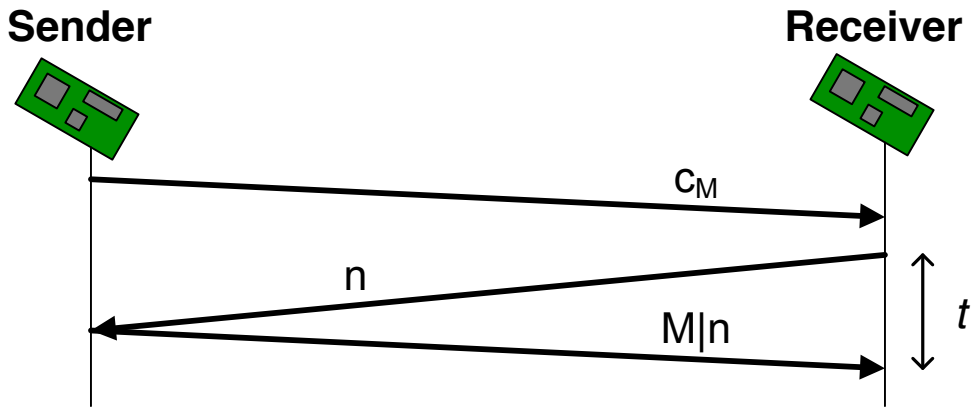



Figure 6.3: RTTMAP message exchange.

still operates in the ISM 2.4 GHz band, unlike the other schemes that use much more spectrum and thus require different antennae and significantly different hardware. New low-power transceivers have recently emerged [131] that support CSS and built in RTT ranging.

## 6.5 Round-Trip-Time Message Authentication Protocol

To satisfy the need for link-layer distance-based message authentication, the existing secure RTT measurement technique must be inseparably integrated with link-layer message exchange. An attacker must not be able to send a message without also participating in the secure RTT measurement. The Round-Trip-Time Message Authentication Protocol (RTTMAP) is believed to be the first protocol that satisfies this requirement.

RTT is self-defeating if it is measured at the *sender*, since the measurement data would need to be sent to the receiver and separately authenticated, thus the RTT principle can only be measured at the receiver side. Some form of trigger is therefore needed to inform the receiver of the intention to exchange a message.[28] RTTMAP implements the required facilities.

The transmission of the payload must be linked to the ranging process securely and without using keys. RTTMAP utilises a *hash function* as the link. The hash is computed over the payload, sent as part of the trigger and then verified by the receiver when the payload becomes available. Since the payload is sent as part of the ranging process, this procedure

[28] Care is needed when reading this section as both the *sender* and *receiver* perform both sending and receiving.

ensures that the ranging cannot be detached from message transmission.

During network planning, each node is configured with a *distance threshold value* $r$ that is determined by the deployer. $r$ is set such that all neighbours with which the node shall communicate are closer than $r$. Nodes that are excluded from communication are assumed to be always further away than $r$. $r$ has to include any measurement error, as discussed in Section 6.9.

A sender first transmits an INIT message (the trigger) to the receiver, signalling a pending data transmission. The receiver responds with a PROBE message that, upon reception, triggers immediate transmission of the DELIVERY message by the sender. Thus, the receiver is able to compute distance from a round-trip-time measurement using the PROBE and DELIVERY messages.

The sender maintains a counter, which is used to generate counter values for use in messages to prevent replay attack. The counter is not critical to meeting the security objectives discussed in the next section, but it does help to prevent attackers from building a dictionary of known messages and commitments.

RTTMAP requires the following three phases for each message exchange (see Figure 6.3).

1. The sender adds a fresh counter value $i$ to message $M$. A commitment is computed, over the whole message (including $i$), using the hash function: $c_M = h(M)$. $c_M$ is sent to the receiver.

2. The receiver caches $c_M$ and creates a fresh nonce $n$. $n$ is sent to the sender as timer $t$ is started.

3. The sender returns $M|n$ to the receiver. The receiver takes a timer reading $t$ to calculate distance $r'$. The receiver recovers $M$ from the response. $M$ is accepted only if $r' <= r$, $i$ is fresh, and $c_M = h(M)$.

The hash commitment $c_M$ prevents an attacker from hijacking the exchange, the nonce $n$ prevents an early response during ranging, and finally, the timer value $t$ allows for distance measurement. The security reasoning behind this approach follows in the next section.

### 6.5.1 Security Objectives of RTTMAP

The use of secure RTT prevents an attacker from pretending to be within the acceptable radius $r$ during the ranging process. However, it is necessary to securely link this to all control messages to avoid problems such as the hijack of ongoing message exchanges or denial-of-service. The security design of RTTMAP aims to avoid the following problems:

**(1) Cache Reset Attack** An attacker injecting malicious INIT messages in response to genuine INIT messages hoping to prevent authentication by resetting the cached $c_M$.

**(2) Timer Reset Attack** An attacker repeating the INIT message, hoping to reset the timer, achieving a lower distance measurement and successful authentication.

**(3) Opportunistic Hijack Attack** An attacker opportunistically sending DELIVERY messages, after overhearing an INIT message, in the hope of injecting a false message without properly participating.

**(4) Resource-drain Attack** An attacker participating solely to waste resources.

Problems $1$ and $2$ are avoided since nodes do not accept messages unless they are in the state where they are expected to arrive. Thus, the cached $c_M$ and timer $t$ cannot be changed until a DELIVERY message has been received or a timeout has occurred. Problem 2 is additionally avoided since the changing nonce prevents an attacker from recording valid responses and replaying them.

Problem $3$ is avoided since RTTMAP sends a one-way hash $c_M$, rather than the actual message $M$ in the first phase. An attacker is thus forced to participate in all earlier stages of the protocol. A DELIVERY message cannot be sent without first computing a hash collision within a few milliseconds and is thus infeasible. Problem $3$ is mainly prevented by the secure ranging process, but this countermeasure prevents problems if an attacker is able to inject data whilst another node is transmitting.

Problem $4$ is avoided gracefully (see Section 6.7), the authentication steps in phase $3$ are deliberately designed to leave the invocation of the hash function until after an acceptable distance measurement is taken. Later in the chapter it is found that this results in lower attack costs.

There is one problem that RTTMAP *does not solve* and which requires management. RTTMAP is not resistant to a *relay attack*. In a relay attack, an attacker will plant a device within the network and then use it to relay messages from outside. The objective of this attack is to bypass the distance measurement. This attack can be prevented by checking for such devices, which is a viable option in some deployments where strict security is already in place. In other scenarios, the use of end-to-end authentication can mitigate this risk.

### 6.5.2 Security Concerns of RTTMAP

RTTMAP relies on a secure ranging mechanism. Such implementations need to tackle the following possible attacks:

**(A) Modulation Overwrite Attack** If an attacker injects data at the modulation level to overwrite the $h$ and $M$ in a genuine exchange, he can bypass the security mechanism by hijacking an existing exchange. The requirements for this attack are harsh. An attacker needs to detect the start of a transmission, then be able to overwrite bytes and generate a valid checksum with only a few microseconds notice. The issue could be avoided using XOR functions to combine the payload and nonce, thus overlaying the fields and preventing over-writing attacks.

**(B) Fast Response Attack** If an attacker reduces the turnaround time in his hardware, then the timing measurement at the Receiver is lowered without breaking the speed of light constraint. An attacker need only reduce the turnaround by a few nanoseconds.

**(C) Modulation Trickery Attack** If an attacker can fool the demodulation systems at the Receiver, a late response can be delivered without detection. An attacker would need to specially design a transceiver system to deliver the several nanosecond benefit necessary to throw off a measurement by several metres. This attack is interesting because any such compromise can potentially be adopted by friendly parties, although at high cost. Like issue A, the use of XOR combination can help to prevent this problem.

| Purpose | Name | Units |
|---|---|---|
| RTTMAP INIT frame length | $l_i$ | bytes |
| RTTMAP PROBE frame length | $l_p$ | bytes |
| RTTMAP DELIVERY frame length (min) | $l_d$ | bytes |
| RTTMAP DELIVERY frame length (max) | $l_{d'}$ | bytes |

Table 6.1: RTTMAP frame lengths.

## 6.6 MAC Protocol Feasibility

This section performs an initial feasibility analysis focusing on the MAC layer. This information is essential because the remaining sections of this chapter rely on this to make further evaluation. Three areas are investigated. First, the integration challenges are identified where RTTMAP is applied at the MAC layer. Second, the delay incurred when passing a message using RTTMAP is calculated. Finally, the maximum theoretical channel throughput possible, when RTTMAP is utilised, is determined.

### 6.6.1 Frame Format Assumptions

The analysis that follows is sensitive to the length of each frame. Therefore, it is necessary to define this length. The precise frame format is obviously an implementation issue, but RTTMAP has minimum requirements. RTTMAP has three frame structures, one for each phase of the protocol, and the minimum envisaged are shown in Figure 6.4.

All structures include $2$-byte *source* and *destination* addresses with a $1$-byte *type* field used to differentiate the frames.

INIT messages require one extra field; the *commit* field ($4$ bytes) that contains the truncated output $c_M$ of hash function $h()$. The total length $l_i$ thus is $9$ bytes. PROBE messages require one extra field, the $4$-byte *nonce* field. The total length $l_p$ is thus $9$ bytes. DELIVERY messages require three extra fields. The *nonce* field is $4$ bytes long, containing the nonce delivered by a PROBE. The $2$-byte *CRC* field contains a checksum. There is no *length* field as the PHY layer has a length field. The *payload* is a maximum of $244$ bytes, because the PHY length field only supports an overall length of up to $255$ bytes. The total length is therefore between $l_d = 11$ and $l_{d'} = 255$ bytes.

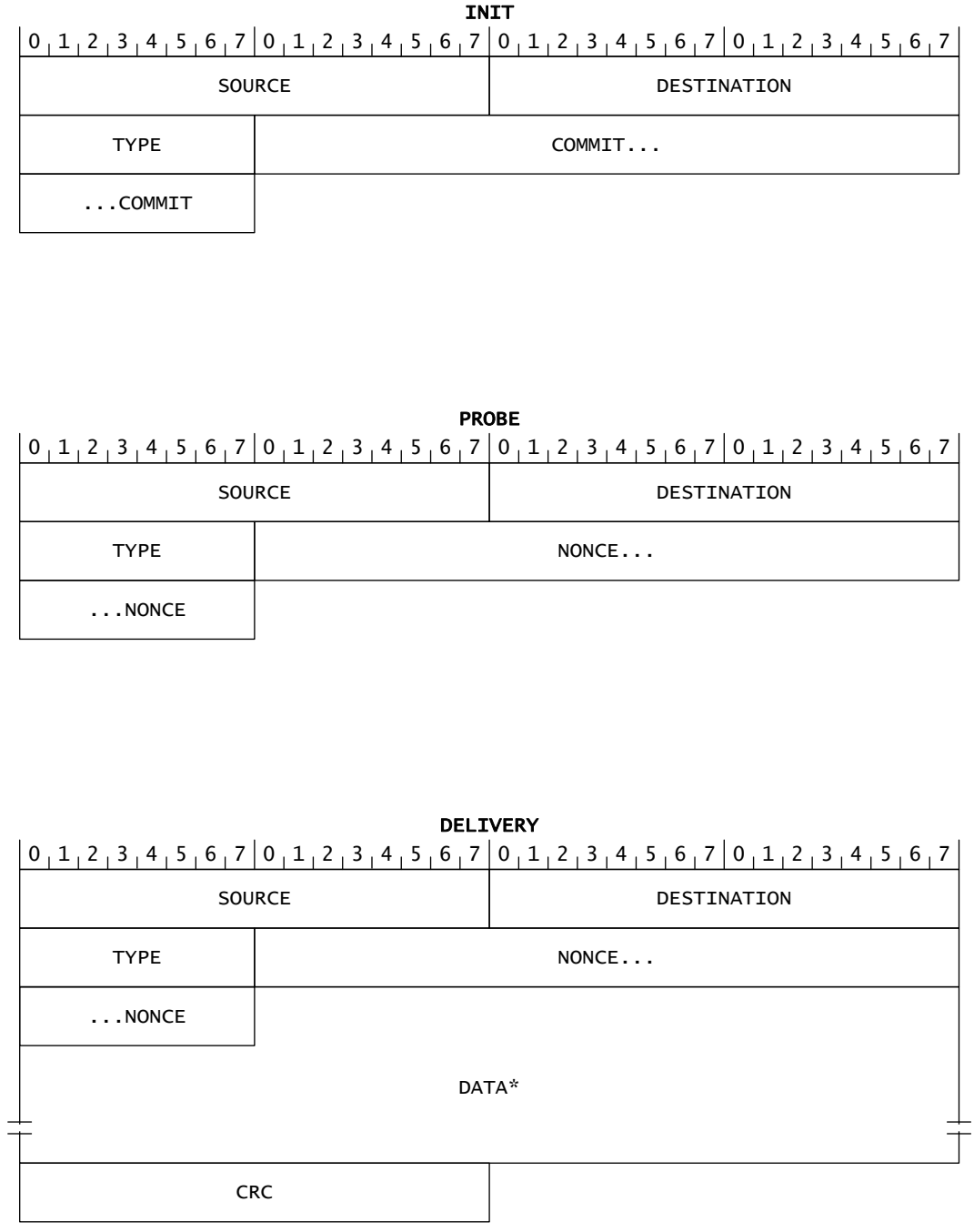



Figure 6.4: RTTMAP frame structures.

### 6.6.2 MAC Protocol Integration

RTTMAP involves close interaction between lower layers in the network stack, alongside existing functionality such as framing and addressing, to enable ranging to be carried out. RTTMAP also requires three frame transmissions to exchange a message, one of which must be sent in real-time. Since the primary function of a MAC protocol is to control access to the shared communication medium, it must be evaluated if, and how, RTTMAP affects, or is affected by, this functionality.

RTTMAP can be integrated with a scheduled protocol easily since the scheduling permits channel exclusivity and avoids the need for clear channel assessment[29]. The fact that RTTMAP uses three frames per message makes no difference in this respect, but the slot length must be sufficiently long for all three frames to be exchanged.

Where contention is necessary, a contention-based protocol will be needed. Such protocols that avoid collision by using clear channel assessment (CCA) present a challenge: CCA cannot be carried out immediately before transmitting a DELIVERY frame. The CCA would delay the frame and break the secure timing mechanism. CCA can, therefore, only be carried out before transmission of INIT and PROBE frames and the CCA period has to be extended to provide coverage of all the frames in an exchange.

This extension of the CCA period works because there is a guarantee that a successful message exchange involves transmissions by both the sender and receiver. This guarantee is similar to that within the RTS/CTS concept, described in Appendix C. RTS/CTS – normally used to avoid the hidden terminal problem – also avoids the need for CCA before the final frame is transmitted; this is because any nodes overhearing the RTS or CTS frames will back off. Thus, RTTMAP exploits this benefit offered by a RTS/CTS MAC protocol, provided the CCA period is sufficiently long to encompass the message exchange.

### 6.6.3 Theoretical Channel Occupation and Throughput

RTTMAP involves greater use of the channel. The achievable throughput depends on the contention handling protocol, but channel occupation first needs to be found. This is needed to determine the slot length in contention-free protocols or the maximum throughput in other

[29] Assuming, of course, there is no significant interference.

protocols.

The channel occupation time must encompass the modulation and propagation of the three transmissions.

The maximum frame sizes (in bytes) for the INIT, PROBE and DELIVERY messages are $l_i = 9$, $l_p = 9$ and $l_{d'} = 255$ bytes respectively. (See Table 6.1.) The resulting modulation durations are $120\mu$s, $120\mu$s and $2088\mu$s based on a modulation duration $d_b = 8\mu$s per byte and $d_o = 48\mu$s of preamble overhead[30]. The maximum total delay purely in modulation is thus $2328\mu$s, of which $240\mu$s (just over $10\%$) might be considered overhead.

Assuming a maximum range of $r_n = 300$ metres, it is necessary to add approximately $3\mu$s for radio propagation time (as there are $3$ exchanges). Thus the shortest occupation time, and thus shortest epoch duration, is $2331\mu$s. This figure does not include turnaround delays or computation. The estimated epoch duration, based on these figures is taken as $2.5$ms, which is used for subsequent calculations.

Under maximum load, with perfect synchronisation, the maximum channel throughput would be approximately $400$ exchanges per second. If nodes need to transmit once per second, this performance would seem adequate on the basis that having $400$ nodes in overlapping space has barely been realised yet. WirelessHART, for example, provides $100$ timeslots per second [45]. Obviously these values depend on how any necessary computation is handled (i.e. concurrently or consecutively). Additionally, the variables and transmission power can be manipulated to alter this performance.

### 6.6.4 Summary of Findings

The issue of MAC protocol selection is intricately tied to the contention-handling model implemented by the MAC protocol. RTTMAP has minimal impact on channel reliability when a scheduled MAC protocol is implemented; this is because of exclusive access to the channel during scheduled access. Performance in a contention-based environment is less ideal because of the difficulty in implementing clear-channel assessment in low-power protocols. Thus RTTMAP is likely to benefit from a scheduled MAC protocol, even though the overhead will be greater in terms of time synchronisation and channel occupation.

[30] Assumes a $6$-byte PHY overhead, as discussed in Section 3.3

| Name | Purpose | Units |
|---|---|---|
| $F_t$ | Energy cost of sending a full message | mAs |
| $F_r$ | Energy cost of receiving a full message | mAs |
| $H_r$ | Energy cost of a receiving a malicious message | mAs |
| $h$ | Energy cost of hashing one frame | mAs |
| $e$ | Energy cost of cryptographically authenticating one frame | mAs |
| $t$ | Energy cost of transmitting one frame | mAs |
| $r$ | Energy cost of receiving one frame | mAs |

Table 6.2: RTTMAP energy analysis components.

The NA5TR1 transceiver, with the minimum headers needed for RTTMAP to operate, is able to carry out the message transfer in less than $2.5$ms. This allows for a maximum theoretical throughput of approximately $400$ messages per second. The actual increase in channel occupation can be as little as roughly $10\%$.

## 6.7 Energy Analysis

Resource-drain attacks aim to deplete node resources, primarily in terms of energy. Before considering RTTMAP as a countermeasure against denial-of-service attack, some initial feasibility questions must be answered about the energy performance of the protocol, both when used by friendly parties and when it is attacked itself. This has to be compared with the conventional approach of using cryptographic message authentication. This section aims to answer the question: is the protocol feasible to begin with, and how does it compare to the conventional (cryptographic) approach when under attack? First, the component parts of each energy evaluation are defined, then the methods and results are discussed.

### 6.7.1 Performance Components

To carry out the energy cost analysis, the analysis values shown in Table 6.2 have to be computed using variables that correspond to the hardware and implementation characteristics. The energy cost of sending, or receiving, a full message comprises of all the actions required; this includes the transceiver cost for the necessary frames and the cryptographic cost. $h$, $e$, $t$ and $r$ are used as components within the full costs.

As these values have to be computed on frames, the minimum frame lengths are needed;

| | INIT **or** PROBE $l_i = l_p = 9$bytes | DELIVERY **min** $l_d = 11$bytes | DELIVERY **max** $l_{d'} = 255$bytes | **Full Slot** $2.5$ms | **EP** $100$ms avg. |
|---|---|---|---|---|---|
| $h$ | - | $h_d = 0.0199918$ | $h_{d'} = 0.034827$ | - | - |
| $e$ | - | $e_d = 0.0020976$ | $e_{d'} = 0.0335616$ | - | - |
| $t$ | $t_i = t_p = 0.0036$ | $t_d = 0.00408$ | $t_{d'} = 0.06264$ | $t_s = 0.075$ | $t_x = 3$ |
| $r$ | $r_i = r_p = 0.00408$ | $r_d = 0.004624$ | $r_{d'} = 0.070992$ | $r_s = 0.085$ | $r_x = 3.4$ |

Table 6.3: RTTMAP and conventional theoretical energy cost components for NA5TR1/MSP430. All figures in mAs.

these were specified in Section 6.6.1. The other dependency relates to the hardware, the NA5TR1 from Nanotron (see Section 6.8) is combined with the MSP430 microcontroller, resulting in an architecture similar to the Tmote Sky [132] sensor node for comparison purposes.

Table 6.3 shows the performance expected for each operation. The method for obtaining each of these figures was explained in Section 3.2. The following factors have been considered. For encryption, AES-256 is a $16$ byte block cipher, thus the payload length must be rounded up to the next multiple of $16$; therefore $l_d = 16$ and $l_{d'} = 256$ in the calculation of $e$. For the raw transmit or receive cost of INIT, PROBE and DELIVERY frames, the size used is increased by $6$ bytes for PHY overhead in the NA5TR1 transceiver. The cost for a full slot and the average low-power MAC protocol extended preamble (EP) is also included as this is required below.

The normal and attack costs, for RTTMAP and the conventional cryptographic approach, are comprised of different combinations of the values presented. Each is now handled separately.

### 6.7.2 Normal Transmission Performance

To *transmit* a message using RTTMAP, the sender has to execute the hash function once and then participate in the three-way handshake to exchange the message. The sender needs to keep the transceiver active until the DELIVERY frame is sent as it needs to be ready to respond to the PROBE frame; the duration is therefore taken as the estimated minimum channel occupation of $2.5$ms computed in the previous section. In the minimum case, a full-length payload is not transmitted and therefore the full slot is not used; hence, the transmission time for the payload ($r_{d'} - r_d$) is subtracted. Since the transceiver needs to both send

| **Protocol** | **Payload** | **LP Preamble** | $F_t$ mAs | $F_t$ mAs |
|---|---|---|---|---|
| RTTMAP | Minimum | No | $h_d + r_s - (r_{d'} - r_d)$ | $0.0386238$ |
| RTTMAP | Minimum | Yes | $h_d + t_x + r_s - (r_{d'} - r_d)$ | $3.0386238$ |
| RTTMAP | Maximum | No | $h_{d'} + r_s$ | $0.119827$ |
| RTTMAP | Maximum | Yes | $h_{d'} + t_x + r_s$ | $3.119827$ |
| Conventional | Minimum | No | $e_d + t_d$ | $0.0061776$ |
| Conventional | Minimum | Yes | $e_d + t_x$ | $3.0020976$ |
| Conventional | Maximum | No | $e_{d'} + t_{d'}$ | $0.0962016$ |
| Conventional | Maximum | Yes | $e_{d'} + t_x$ | $3.0335616$ |

Table 6.4: RTTMAP and conventional theoretical energy cost calculation for transmission (mAs).

*and* receive during this time, $r_s$ is used instead of $t_s$ for simplicity and as it is more expensive. Where an extended-transmission power-saving MAC protocol is in use, the average epoch duration ($100$ms) has to be added (this concept is discussed in Section 3.3).

In the conventional approach, the sender has to apply a cryptographic authentication function and then need only transmit the DELIVERY frame. The delivery frame can be embedded in the LP preamble if used, otherwise a raw transmission can be used. The necessary authentication data takes the place of the nonce in the DELIVERY message.

The transmission costs can be computed as shown in Table 6.4. The actual costs based on the reference platform are shown. It is not difficult to see that RTTMAP is more expensive than the conventional approach in all cases, this issue is discussed below.

### 6.7.3 Normal Receive Performance

To *receive* a message in RTTMAP, the receiver will activate the transceiver and leave it active until the end of the DELIVERY frame is received; this is necessary due to the ranging process. Again, as with transmission, the receiver will need to both transmit and receive during this time; the receive cost $r_s$ is used instead of the transmit cost $t_s$ for simplicity. The receiver will then execute some basic logical operations, using the timer for example, and then execute the hash function. If a power-saving MAC protocol is in use, the average epoch duration can only be omitted if there is a mechanism to recover, and signal delivery, of the INIT message early. This assumption is made below.

To receive a message in the conventional case, the receiver will need to activate the

transceiver to receive the message and then run the cryptographic hash function. Where a power-saving MAC protocol is in use, the average epoch duration is omitted since the protocols usually allow the receiver to recover the frame before the end of the epoch.

| **Protocol** | **Payload** | **LP Preamble** | $F_r$ mAs | $F_r$ mAs |
|---|---|---|---|---|
| RTTMAP | Minimum | N/A | $h_d + r_s - (r_{d'} - r_d)$ | $0.0386238$ |
| RTTMAP | Maximum | N/A | $h_{d'} + r_s$ | $0.119827$ |
| Conventional | Minimum | N/A | $e_d + r_d$ | $0.0067216$ |
| Conventional | Maximum | N/A | $e_{d'} + r_{d'}$ | $0.1045536$ |

Table 6.5: RTTMAP and conventional theoretical energy cost calculation for normal reception.

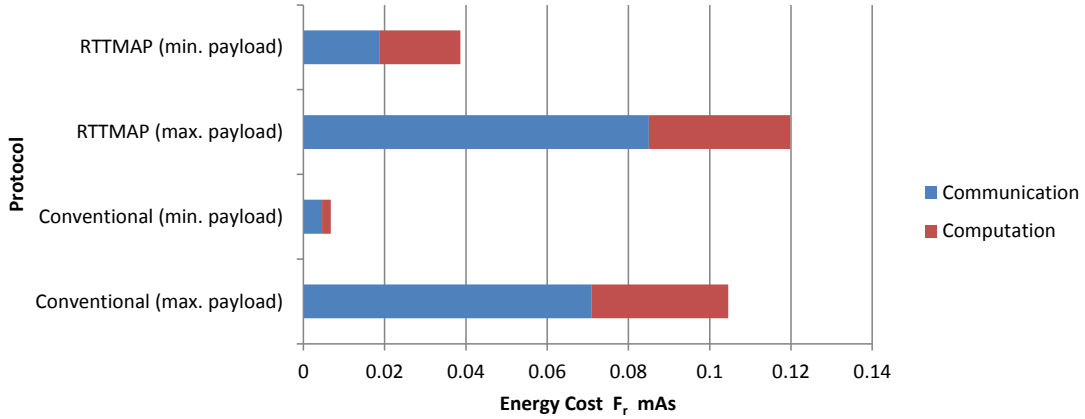



Figure 6.5: RTTMAP and conventional theoretical energy cost comparison for normal reception.

The reception costs can be computed as shown in Table 6.5. Again, RTTMAP is more expensive than the conventional case. For an empty payload, RTTMAP is about $6$ times more expensive; for a full payload, RTTMAP is about $15\%$ more expensive. The reason for this performance difference can be visualised in Figure 6.5. RTTMAP requires greater channel occupation in order to transmit the INIT and PROBE frames; thus, for a given payload length, RTTMAP can never be cheaper in terms of communication than the conventional case.

This performance means that RTTMAP is always more expensive than the conventional case, in these protocol conditions, *under friendly circumstances*; the performance under attack is different and is considered in the next section. It is important to realise that one of the motivations of RTTMAP was to protect expensive functions higher in the network stack. For example, if an elliptic curve scalar point multiplication is invoked by an attacker then it

might cost $114$ mAs; this energy drain would be hundreds of times more than the maximum RTTMAP cost, see Section 3.2. Investing additional energy in regular communication may be worthwhile if it protects from such greater energy drain.

### 6.7.4 Energy Performance under Attack

When under attack, the actions of the receiver will differ depending on the type of attack employed and the protocol in use. RTTMAP can be attacked in the following principle ways, which demonstrate further penetration into the mechanism exposed at the receiver:

**Attack 1** The attacker will inject arbitrary INIT messages and respond to the resulting PROBE messages in order to use the correct nonce. Obviously, the timing mechanism will expire and the message will be dropped. The hash function is *not* executed and the receiver can return to a sleep state early.

**Attack 2** The attacker injects arbitrary INIT and PROBE messages in quick succession to try to beat the timing mechanism. The attack stops at the nonce check (a simple comparison) since the attacker cannot feasibly ($2^{32}$ combinations) obtain the nonce in advance. The hash function is therefore *not* executed.

**Attack 3** The attacker carefully injects a partial DELIVERY message during an existing exchange (hijack attack). The nonce in the genuine message is not altered. The receiver will execute the hash function, which fails, but expends resources in doing so.

The hash function is only executed in attack 3. In that case, the cost will be equal to a normal message exchange. The difficulty of carrying out this attack is expected to be very difficult, and it relies on an existing exchange being underway; it is expected to be easier to simply jam the channel. In all other cases, the hash function is not executed. In attack 1 the node can shut down the transceiver early, avoiding energy waste. The maximum cost is therefore equal to a normal message exchange.

In the conventional case, the attacker is left with injecting arbitrary messages. The full overhead is required since the message cannot be rejected until the cryptographic result is known. This means that the energy consumption is formed of the full reception period, followed by the hash function execution.

The reception costs under attack can be computed as shown in Table 6.6. As with the previous two sections, $r_s$ is used instead of $t_s$ despite the transceiver using less energy whilst transmitting.

| **Protocol** | **Payload** | **LP Preamble** | $H_r$ mAs | $H_r$ mAs |
|---|---|---|---|---|
| RTTMAP Attack 1 | Minimum | N/A | $r_s - r_{d'}$ | 0.014008 |
| RTTMAP Attack 1 | Maximum | N/A | $r_s - r_{d'}$ | 0.014008 |
| RTTMAP Attack 2 | Minimum | N/A | $r_s - (r_{d'} - r_d)$ | 0.018632 |
| RTTMAP Attack 2 | Maximum | N/A | $r_s$ | 0.085 |
| RTTMAP Attack 3 | Minimum | N/A | $h_d + r_s - (r_{d'} - r_d)$ | 0.0386238 |
| RTTMAP Attack 3 | Maximum | N/A | $h_{d'} + r_s$ | 0.119827 |
| Conventional Attack | Minimum | N/A | $r_d + e_d$ | 0.0067216 |
| Conventional Attack | Maximum | N/A | $r_{d'} + e_{d'}$ | 0.1045536 |

Table 6.6: RTTMAP and conventional theoretical energy cost calculation for malicious reception.

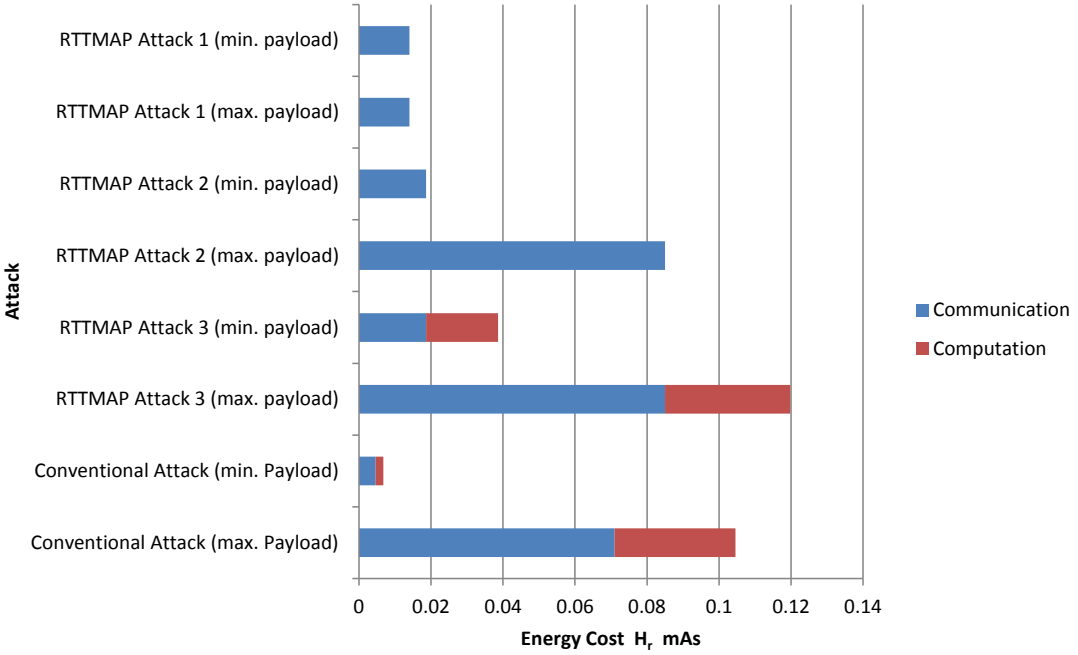



Figure 6.6: RTTMAP and conventional theoretical energy cost comparison for malicious reception.

The results show that the benefit depends on the type of attack launched, and the size of the payload. This is shown graphically in Figure 6.6. With the smallest payload, an attack on the conventional protocol causes less energy wastage than one against RTTMAP; at first sight this seems to represent bad performance, but an attacker has the option of injecting a full payload to extend transceiver reception and computation time. The conventional case

then performs much worse in attack 1; RTTMAP is able to use under a fifth of the energy in rejecting those attacks. This theoretically means that a node could preserve its resources for five times longer if it uses RTTMAP when under such an attack.

In attack 2, RTTMAP avoids use of the hash function, and so is cheaper than the conventional case that *must* run the cryptographic encryption function.

In attack 3, RTTMAP is still outperformed by the conventional case, although only by about $10\%$. However, attack 3 requires a significant level of attack skill on a repeated basis as it is only possible when the receiver is involved in an existing (genuine) exchange. It is therefore not a vector that can be used for *arbitrary* injection, unlike the conventional case.

The performance of the hash function implementation could be improved to counter this problem. Alternatively, a *weaker* hash function could be used since the hash need only remain secure for a *fraction* of a second, at most. Considerable effort and luck is likely to be required by the attacker to break even a weaker hash function, in the time available. Also, the attacker will need to repeat the effort to repeat the attack.

By contrast, the encryption function cannot be replaced with a weaker cipher without compromising security. Keyed protocols may reveal the key following a compromise and thus become totally useless after just one successful attack. This is a strength of RTTMAP. It is difficult to lower the strength of an encryption function, while in RTTMAP various optimisations are possible.

### 6.7.5 Summary of Findings

The main feasibility issue with security protocols is the energy efficiency both when used with friendly parties and when under attack. The energy efficiency can be derived from the length of individual frames and the energy effort required to transfer them using the transceiver. This feasibility analysis took such an approach by using known, and measured, performance values from available WSN communication and processing hardware.

The energy overhead of RTTMAP compared to a cryptographic scheme depends on the combined cost of communication and computation in each case. Although in this analysis the RTTMAP scheme was more expensive, adjustments to the implementation could improve this relationship. Critically, RTTMAP provides improved energy performance when

attacked, under some circumstances, in comparison to the conventional cryptographic approach; in some cases RTTMAP can use under a fifth of the resources in rejecting messages than the conventional cryptographic approach. In addition, attacks against RTTMAP are less straightforward than arbitrary injection attacks against the conventional cryptographic approach.

RTTMAP needs to keep the transceiver awake longer, but another difference in energy overhead comes from the hash function. These concerns may be irrelevant due to the physical security gain that can be made from the DBMA concept.

## 6.8 Implementation

To implement RTTMAP it must be possible to satisfy the security requirements outlined in Section 6.5.1 as well as the tackling the security issues identified in Section 6.5.2. This section discusses these implementation challenges in respect of the Nanotron NA5TR1 as a first prototype, motivates the need for a modified RTTMAP-N protocol and details the implementation used for feasibility analysis. The Nanotron DK development kit was chosen, comprising of an Atmel *ATmega128* microcontroller and the previously introduced Nanotron NA5TR1 transceiver [131].

### 6.8.1 Nanotron NA5TR1

The NA5TR1 is the first low-power transceiver that supports combined ranging and communication. It is therefore a candidate for an initial implementation of RTTMAP for practical evaluation purposes. Unlike earlier WSN transceiver designs, such as the CC2420, the NA5TR1 uses *chirp spread spectrum* (CSS) modulation. CSS is a spread spectrum technology that provides some of the benefits of ultra-wide band in the existing $2.4$GHz ISM band. It is therefore not ultra-wide band technology because the signal is spread over a relatively narrow spectrum, rather than half a gigahertz or more. In particular the use of *frequency sweeping*, rather than *shift keying*, allows CSS is perform better in Doppler environments and it provides improved resistance to frequency fading.

The NA5TR1 operates in the same power class as commonly used sensor node transceivers

such as the Chipcon CC2420. The NA5TR1 has a maximum transmit power of $1mW$ and timing provides ranging accuracy of $\pm 1m$ in ideal conditions. The NA5TR1 ships as a development kit, the Nanotron DK, which was utilised for these experiments. Currently no WSN operating system such as TinyOS supports the platform directly, so the provided C API was utilised.

The ranging provided by the NA5TR1 can be operated in two modes. The default mode actually involves two RTT measurements, initiated by both parties; these measurements are then averaged, resulting in higher accuracy. An alternative *fast ranging* mode is supported that involves a single RTT measurement and is better aligned with RTTMAP.

The Nanotron API implements fast ranging using a three-message handshake. The initial message begins the ranging process and receipt on the other node causes an immediate response. A third message is then returned which includes data about the turnaround time on that node. The receiver can then adjust the ranging measurement to obtain just the in-flight delay. Obviously the RTTMAP protocol requires that the turnaround time be constant, and minimal, and that all messages used for ranging contain a destination MAC address in the headers. However, this platform is suitable for evaluation purposes.

One issue remains that has to be resolved to enable RTTMAP. The transmission of user data within ranging transmissions is not possible with the NA5TR1. Therefore it is not possible to transmit a payload in the ranging response, nor is it possible to include a nonce in the ranging messages. The RTTMAP-N protocol, described below, aims to resolve these problems.

### 6.8.2 RTTMAP-N Implementation

To achieve RTTMAP equivalent functionality, *RTTMAP-N* modifies the MAC addresses during the ranging process. Each node is still configured with a distance threshold value $r$, but four phases for each message transmission are used (see Figure 6.7):

1. The sender adds a fresh counter value $i$ to message $M$. The source address must also be included in $M$ to avoid collision of commitments (discussed below). A commitment is computed using the hash function: $c_M = h(M)$. $c_M$ is sent to the receiver and the MAC address of the sender is then set to a *temporary address* $a'$ based on $c_M$.

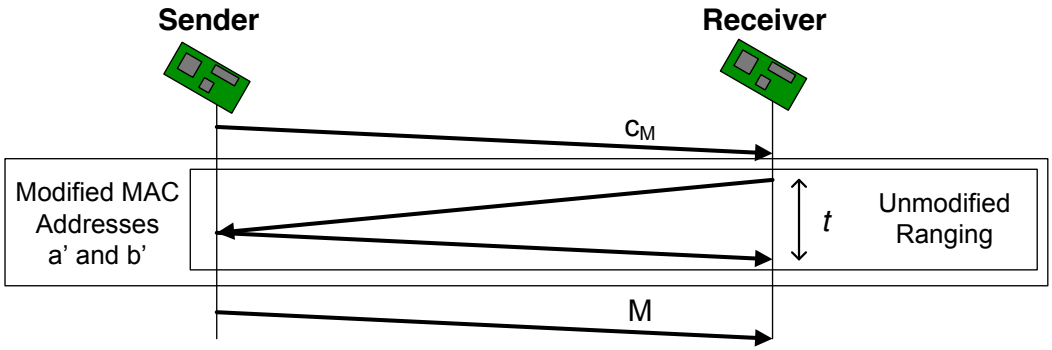



Figure 6.7: RTTMAP-N message exchange.

2. The receiver caches $c_M$ and the original address $a$ of the sender. The MAC address of the receiver is *randomised* to $b'$ and a ranging request is sent to $a'$, generated from $c_M$, as the ranging timer is started.

3. The sender sends a ranging acknowledgement to $b'$.

4. Both nodes restore the original MAC addresses $a$ and $b$. The sender then sends $M$ to the receiver. The receiver takes a timer reading $t$ to calculate distance $r'$. The receiver recovers $M$ from the packet. $M$ is accepted only if $c_M = h(M)$, $r' <= r$ and $i$ is fresh.

RTTMAP-N does not require a nonce, because the *randomised MAC address* of the receiver is not known by the sender until the ranging request is received, thus providing the same function. RTTMAP-N does not need to insert $M$ into the ranging response because both the delivery of $M$ and the ranging acknowledgement are separately linked to the secure ranging via the hash commitment (see below). The security requirements in Section 6.5 are satisfied, although the importance of the counter $i$ and hash function is greater since the message is not sent with the ranging response and thus must be securely linked to the ranging response.

The separation of the message and ranging response means that the ranging node needs an assurance that the ranging is in relation to the delivered message. By changing the MAC address of the sender based on the commitment, an attacker outside the network cannot utilise genuine nodes as they will not change their MAC addresses accordingly. It is assumed, as with RTTMAP, that the attacker has not compromised, or deployed, any nodes within the secured area.

There is a risk of address collision; if two nodes send a message with the same commitment, in overlapping RF space, then the senders will both assume the same sender address. The receiver, or receivers, will perform ranging with two nodes simultaneously due to the ambiguity in addressing. To avoid this problem, the source address is included in the message to cause the commitments to be address dependant.[31]

## 6.9 Ranging Accuracy and Secure Zone Requirements

DBMA requires a security boundary that encompasses an area known as the *secure zone*. Messages sent from outside of this zone are rejected and attackers are prevented from entering it by physical security. The question is where this boundary in relation to the WSN deployment has to be placed such that DBMA can be effective.

At first glance, it might be assumed that the fence is placed such that the distance from each node to the fence is never closer than the distance of each node to its communication peers in the WSN. Unfortunately, distance measurements contain errors. The measurements may be larger (but never shorter[32]!) than the Euclidean distance as signals may follow non-line-of-sight paths.

The fence must therefore be placed using *measured* distances and not Euclidean distances, see Figure 6.8. Fitting the secure zone to the observed worst-case measurement ensures that an attacker cannot pretend to be located in the secure zone, whilst maintaining a high probability that genuine messages will be accepted. Unfortunately this requires that the maximum measurement on each link is used as the authentication threshold, thus enlarging the secure zone requirement when links have errors.

Recall the scenario from Section 6.2 where two nodes, $A$ and $B$ are located $5m$ apart. Consider that the *measured* distance between $A$ and $B$ is actually between $5m$ and $6m$. In this case, $A$ and $B$ would reject genuine transmissions if the threshold was unchanged, since the measurement error would cause them to *appear* to originate from outside the secure zone. By constructing the secure area using $6m$ circles around $A$ and $B$ the problem can be corrected, but a *space overhead* is introduced as shown in Figure 6.8. Note that it

[31] This finding and requirement is new, and therefore not in the original publication [8].
[32] Some negative error may be present, but it is bounded as discussed later.

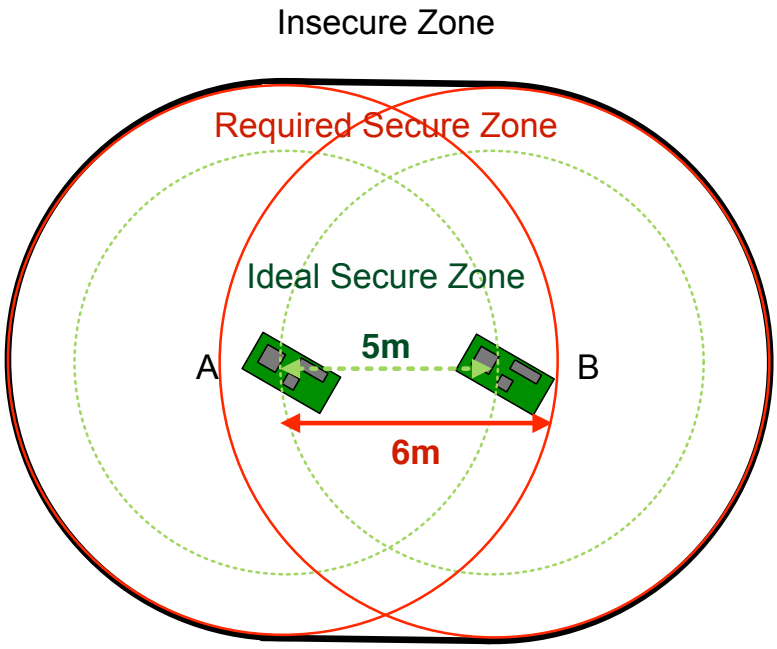



Figure 6.8: DBMA secure zone taking measurement error into account.

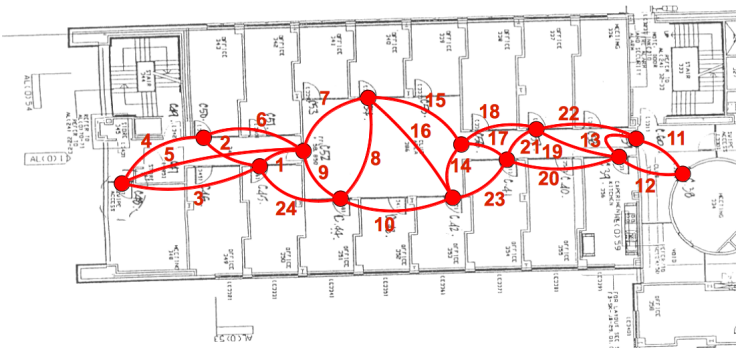
is not acceptable to simply subtract expected errors from measurements as an attacker may be able to produce error-free measurements.

A secure zone cannot be arbitrarily large in a practical WSN deployment; it may be too expensive or just impossible to acquire land. It is therefore desirable to analyse the required secure area size of a practical DBMA implementation and to investigate methods to reduce the secure zone area. This section introduces two test deployment sites, analyses experimental distance measurements and then proposes methods to reduce the secure zone requirements.

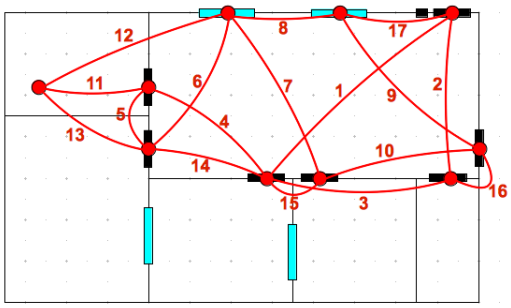
### 6.9.1 Experimental Deployments

To evaluate DBMA performance, range measurements were collected from two test deployments (see Figure 6.9). Deployment A is in a modern office building comprising of earthed metal floor and roof panels with glass office partitions containing metal blinds. Deployment B is a small radio station of traditional breeze block construction with fewer earthed and metallic surfaces. These two deployments were chosen due to the different construction techniques and therefore differences in RF propagation patterns. The figures show node positions and communication links available for topology construction.

To conduct measurements, Nanotron DK nodes were placed in the intended sensor node




(a) Deployment A




(b) Deployment B

Figure 6.9: Experimental DBMA deployments.

positions and 50 RTT measurements were carried out on each available link. These samples give a picture of the distance measurement distribution on each link. RTT distance measurement distribution is similar for both transmission directions on a link as an RTT measurement requires transmission in both directions.

### 6.9.2 Distance Measurement Accuracy

Realistic distance measurements contain two types of errors:

**Timing Errors** Timing error is related to the design of hardware and issues such as clock drift. This error can be negative or positive, but is bounded and only influenced by the receiver of the message. *An attacker is unable to increase this potential negative error*. In the case of the NA5TR1, this results in a distance error of up to $\pm 1m$.

**Propagation Errors** Propagation error is related to the signal pathway and edge detection delay in the transceiver. The error is always positive as the signal cannot travel faster than the speed of light. However, the error can be regarded as being unbounded. In these deployments, propagation errors of up to $+10m$ are observed.

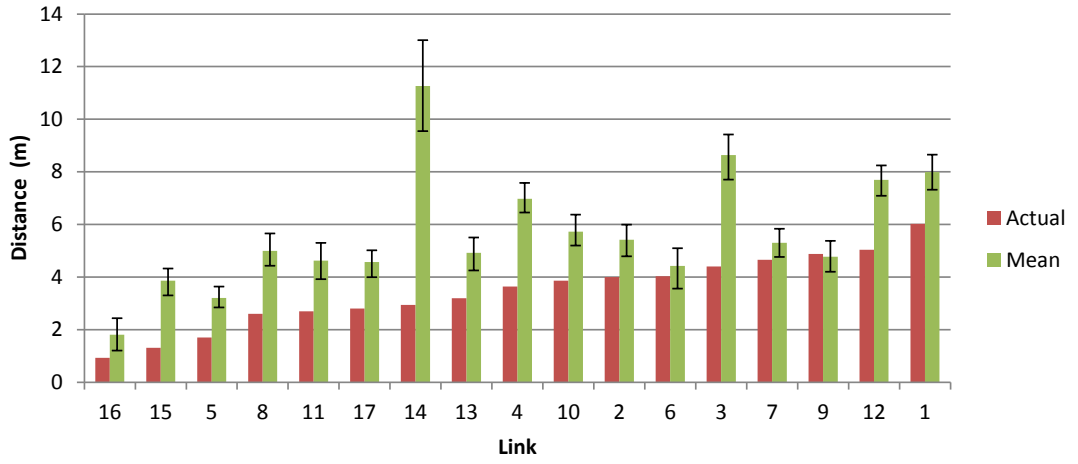



Figure 6.10: Actual vs. average distance measurement in deployment B.

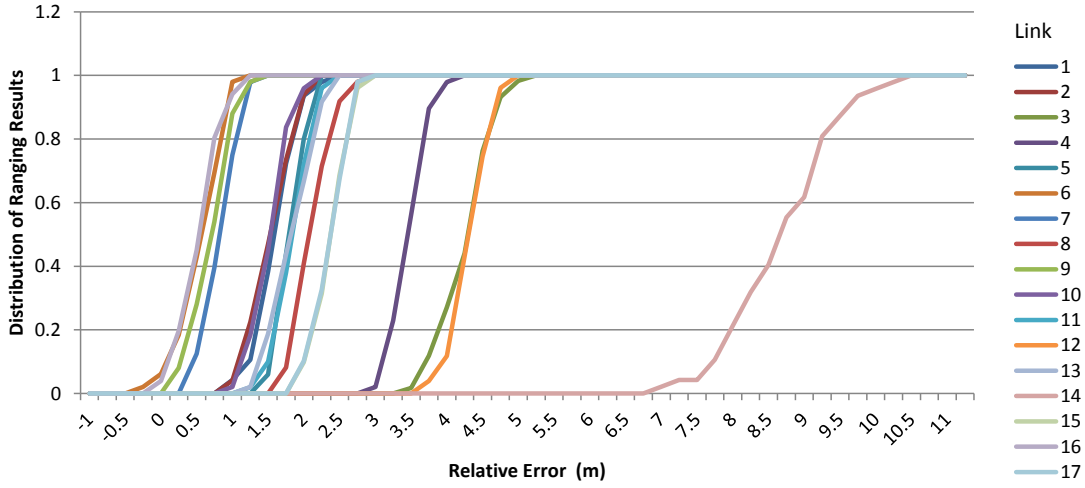



Figure 6.11: Distribution of distance measurements in deployment B as a cumulative histogram.

Figure 6.10 shows the Euclidean distance and the average measured distance for each link in deployment B. It is clear that there is no reliable correlation between the two values meaning that defining the security boundary requires test data on a per-deployment basis. This is certainly feasible, particularly given that some deployers [133] now carry out such data gathering before deployment.

Figure 6.11 shows the ranging accuracy distribution for deployment B. This shows that the distribution is contained within about $1.5m$ for each link, demonstrating that the ranging variance is generally similar to the timing accuracy range. The notable exception to this is link 14, which is distributed over a much wider area and is likely to be caused by several strong RF pathways. In all cases, note that the inaccuracy is positive. The only negative

inaccuracies are contained within less than $1m$, as expected due to the timing errors discussed above. This is also noticeable in Figure 6.10 where the average for link $9$ is actually below the Euclidean distance. Link $6$ similarly has a lower measurement bound, but not a mean measurement, below the Euclidean distance.

### 6.9.3 Secure Area Requirements and Optimisation

An *ideal* secure zone would encompass an area formed of circles around each node. Each has a radius representing the Euclidean distance to the furthest adjacent communication partner for that node.

However, as identified, this needs to be expanded to take errors in account. The *required worst-case* secure zone thus uses the maximum measurement rather than the Euclidean distance.

Figures 6.12 and 6.13 show the ideal and worst-case requirement secure area in deployments A and B. For practical reasons the secure area has the form of a square and not an arbitrary shaped form.

The worst-case secure area is more than double the ideal in both deployments. From a practical perspective this results in high financial costs as a lot of land is needed to implement the secure zone. It is therefore desirable to reduce this overhead.

## 6.10 Secure Zone Optimisations

One method to reduce the secure area requirement is to exclude links that increase the secure area requirement undesirably but are not needed to form a fully connected network. This method of optimisation is referred to as *link pruning*. For example, in deployment B it possible to exclude link $14$ (see Figure 6.9), which has huge distance measurement errors (see Figure 6.10). Nodes can forward messages along links $4$ and $5$, which have better error values. This section proposes and evaluates such approaches in the deployments introduced in the previous section.

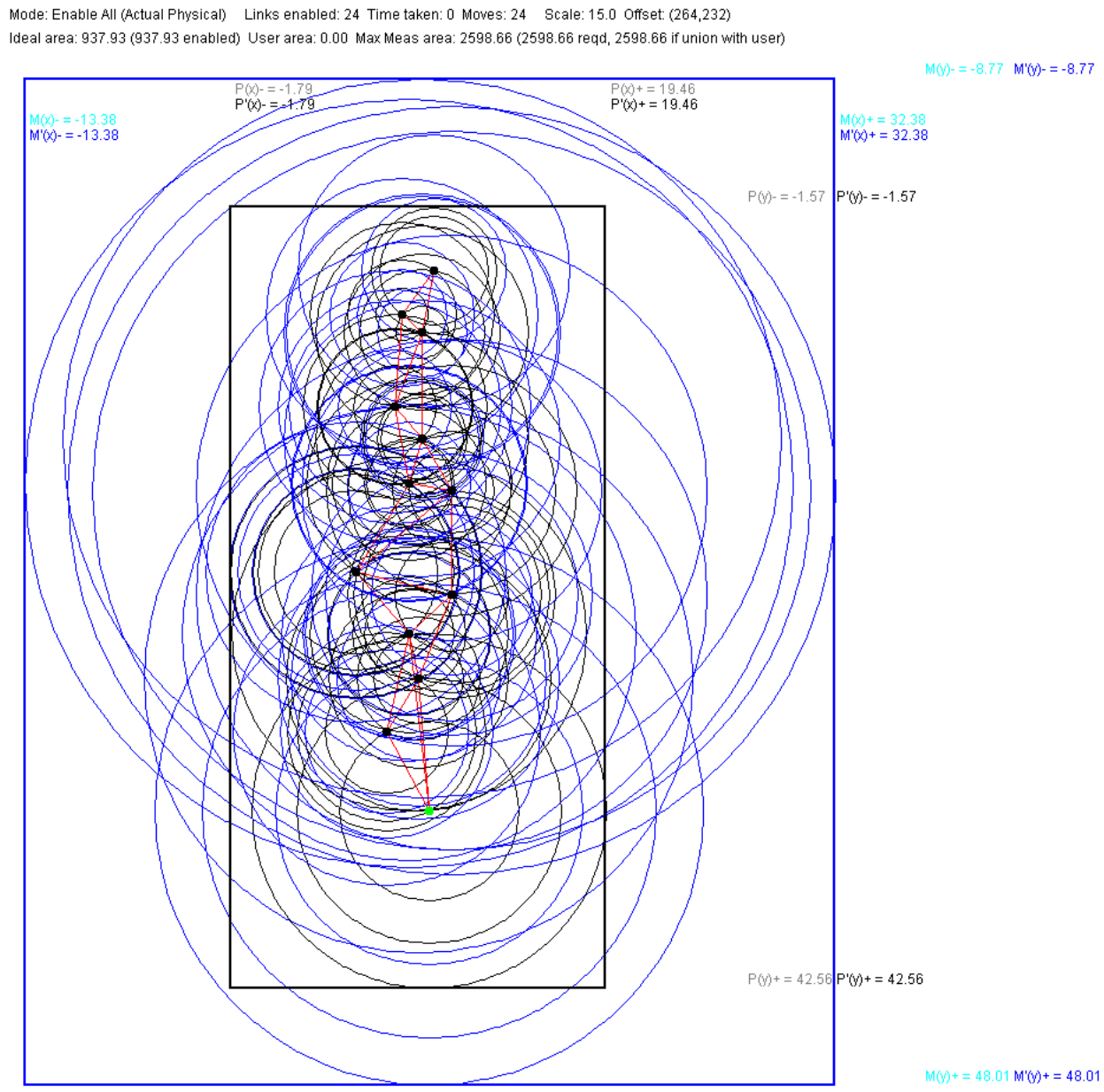



Figure 6.12: Ideal and worst-case required Area in deployment A. Black circles indicate ideal thresholds. Blue circles indicate required maximum measured thresholds. Black boundary indicates ideal secure area ($937.93m^2$). Blue boundary indicates required secure area ($2598.66m^2$).

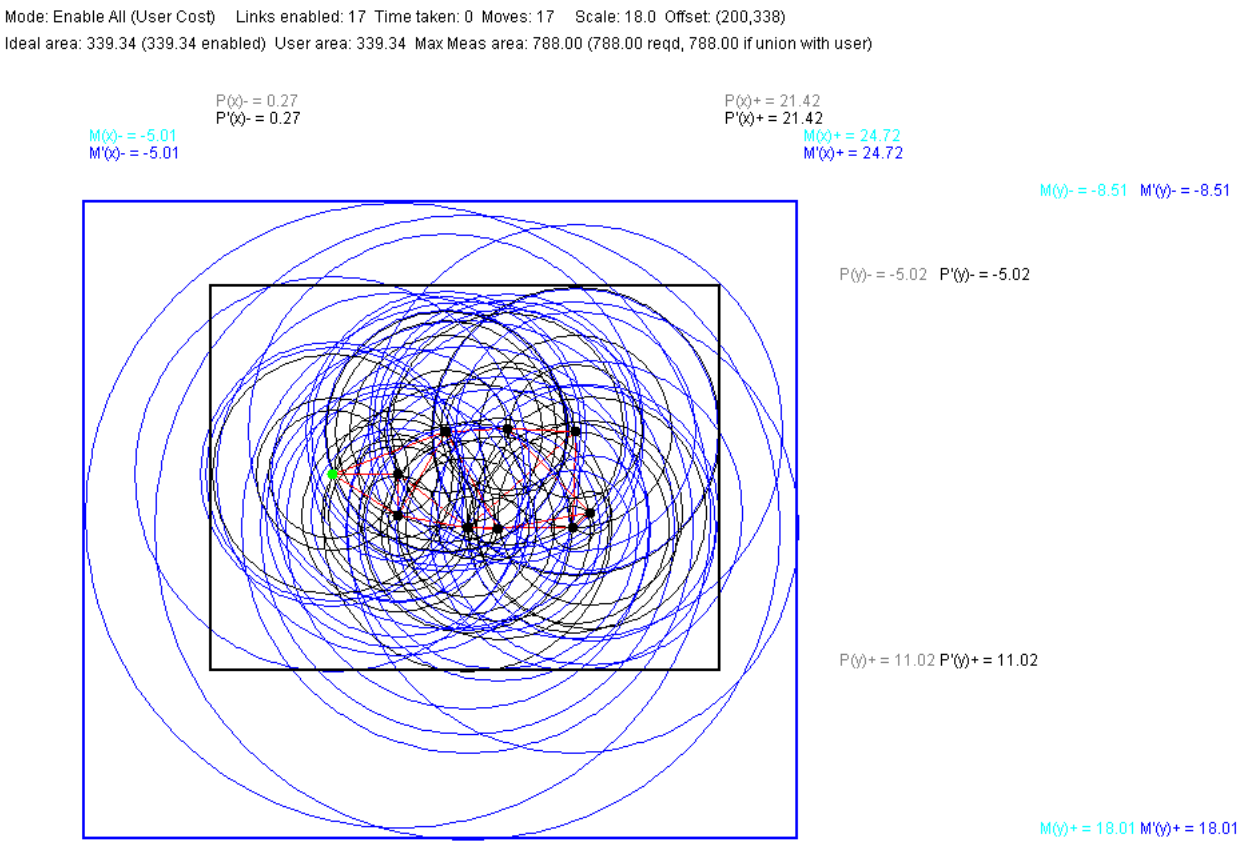



Figure 6.13: Ideal and worst-case required area in deployment B. Black circles indicate ideal thresholds. Blue circles indicate required maximum measured thresholds. Black boundary indicates ideal secure area ($339.34m^2$). Blue boundary indicates required secure area ($788m^2$).

### 6.10.1 Optimal Link Pruning Solution

The most optimal solution to obtain a fully connected network, with the least area requirement, can obviously be found by testing every possible network topology. This is known as a *brute force search*. Each topology has a unique combination of enabled and disabled links and potentially different area requirements. Since every possible combination is tested, the found solution cannot be improved upon, unless the nodes are moved or modified.

The method in Algorithm 1 performs such a search. The algorithm considers every combination, initially rejecting those that are not fully connected. A topology is considered fully connected if every node can send a message to the sink. The area requirement is then computed using a rectangle boundary encompassing all radii that represent the maximum measurements from each enabled link. No point outside the rectangle is within the maximum measurement, in effect preventing an attacker from being close enough to inject messages. The combination with the smallest area requirement is selected.

**Algorithm 1** Optimal DBMA link pruning algorithm. Tests all combinations of enabled and disabled links. The fully connected combination with the lowest area is selected.

```
LET b = infinity
LET s = NULL
FOR EACH combination c of enabled and disabled links
 IF c is not fully connected THEN ignore
 LET a = the secure area required by c
 IF a < b THEN
  LET b = a
  LET s = c
 END IF
END FOR
```

In deployment A (office), the smallest area achievable is $972.03m^2$. In deployment B (radio station), the smallest area achievable is $337.03m^2$. Obviously this is based on a rectangle area and further optimisation is possible by taking the union of the circular areas, but this is avoided for simplicity.

This results are shown graphically in Figures 6.14 and 6.15. It is easy to see that the disabled links, shown in light blue, would have had a significant impact if enabled.

The approach is highly inefficient as it does not scale well. Every possible combination of enabled and disabled links has to be generated and tested for connectivity. Then the cost of

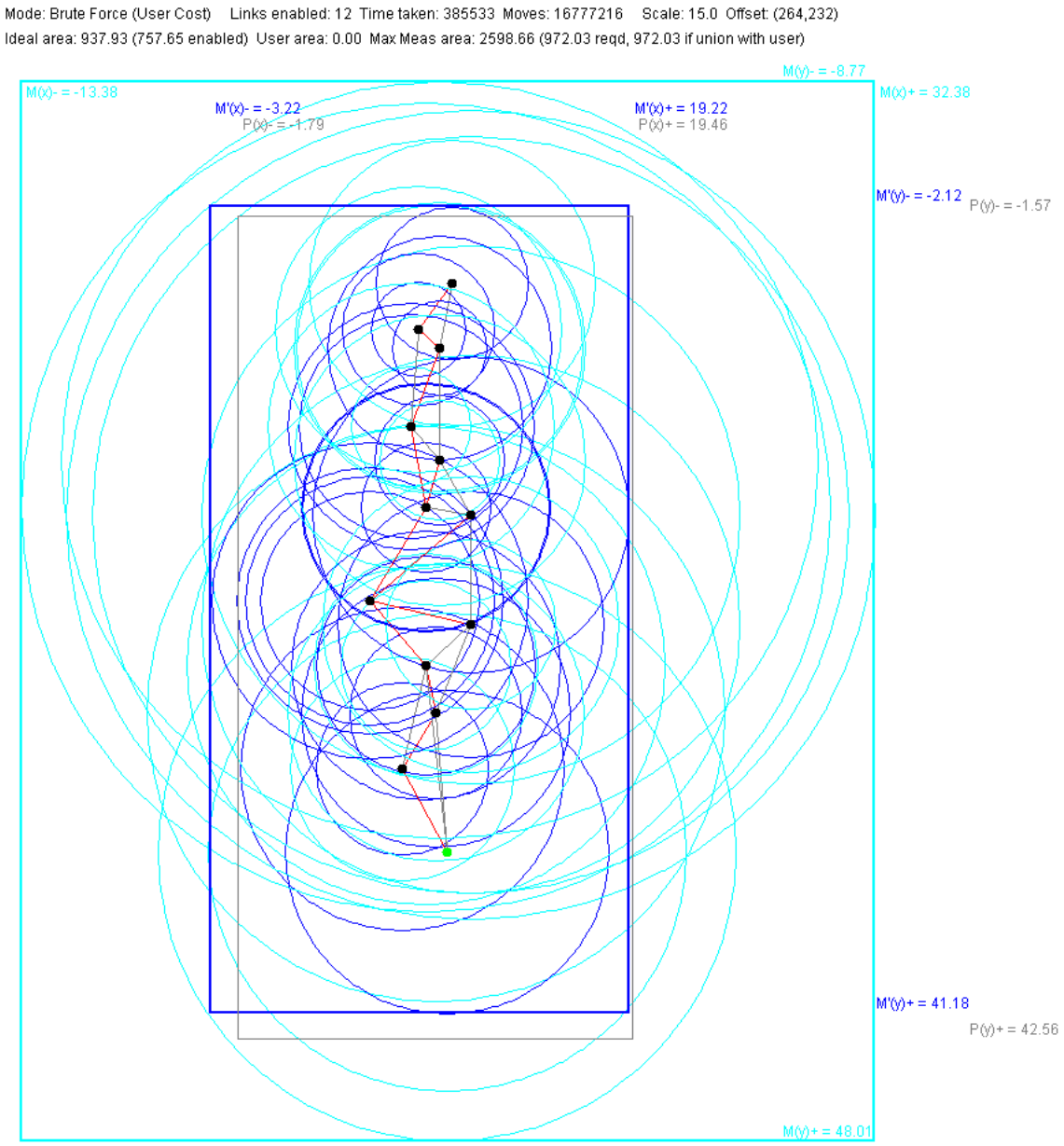



Figure 6.14: Ideal and required area with optimal DBMA link pruning algorithm in deployment A. Circles indicate required maximum measured thresholds. Dark blue circles indicate enabled thresholds and light blue circles indicate disabled thresholds. Black boundary indicates ideal secure area ($937.93m^2$). Dark blue boundary indicates required secure area ($972.03m^2$).

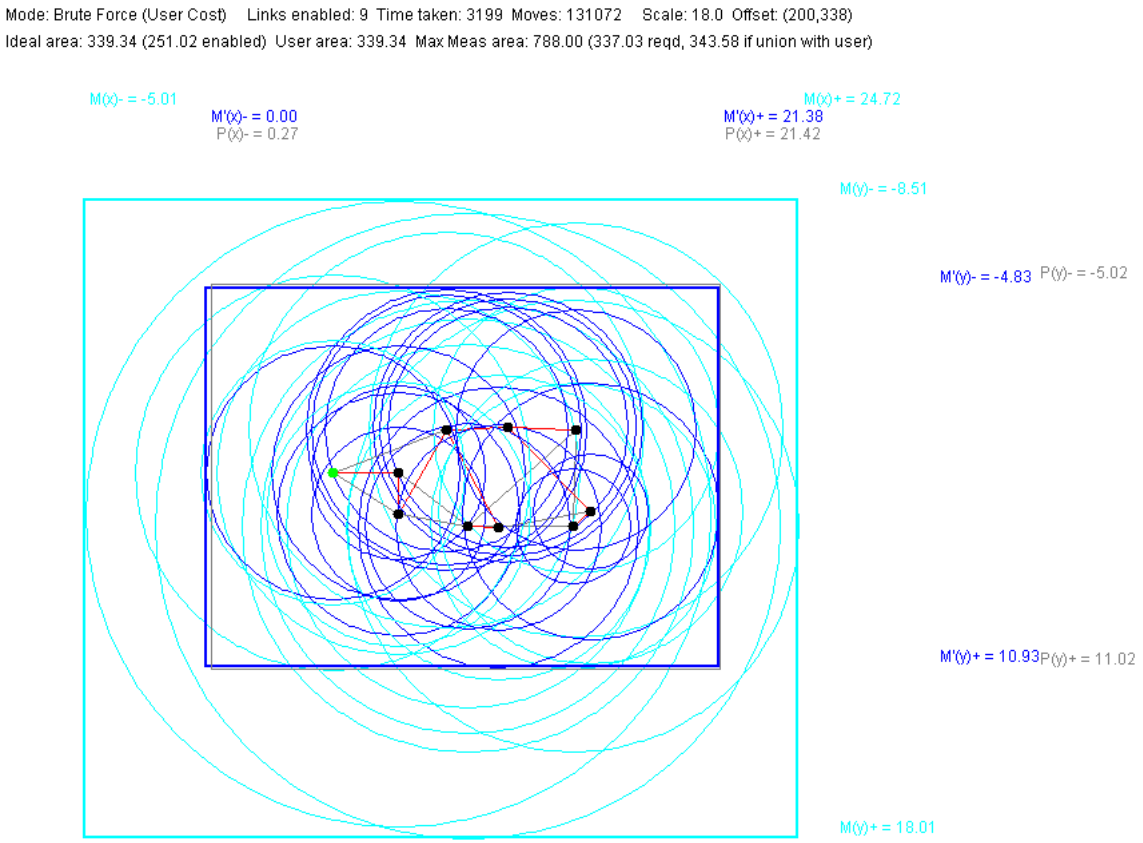



Figure 6.15: Ideal and required Area with optimal DBMA link pruning algorithm in deployment B. Circles indicate required maximum measured thresholds. Dark blue circles indicate enabled thresholds and light blue circles indicate disabled thresholds. Black boundary indicates ideal secure area ($339.34m^2$). Dark blue boundary indicates required secure area ($337.03m^2$).

every successful candidate has to be generated. Adding an additional link thus requires that all existing cases are tested twice, forming an exponential problem with complexity $O(2^n)$.

This inefficiency might be acceptable in smaller networks, but it quickly becomes infeasible when scaled. For example, the program used completed the calculation for deployment A ($17$ links) in $3$ seconds. For deployment B ($24$ links) this increased to $6$ minutes. In a network with $64$ links, $2^{64}$ combinations have to be tested. Finding the solution is thus harder than a brute force attack against a DES cryptographic key: it would take over $584$ years just to generate raw combinations at $1$ Gigahertz. This scalability problem must be addressed to allow feasible use with networks formed of hundreds of nodes.

Another issue with the optimal solution is that it must be run centrally with all the available data. The data has to be transferred to a central location and then the selected topology data transferred back to the nodes. Some optimisations might permit in-network deployment of the algorithm.

### 6.10.2 Computationally Efficient Solution

In the cryptographic world, partial and previous results do not (normally) give any clue as to what combination should be tried next. The algorithms are thus forced to fully try all combinations, leading to the security level required. This limitation does not exist in the case of this problem; the topology can be used as a search tree and previous data can be used in the heuristics. This subsection details this approach.

A solution must satisfy two goals: (A) The final result must be fully connected, such that all nodes are connected to the sink. (B) The final result should provide a secure zone requirement of comparable fitness to the optimal solution. Note that this does not necessarily mean that the absolute optimal solution is found.

This section discusses a search algorithm that searches for routes starting at the sink (the root of the map). The algorithm works by expanding nodes, following the links from that node, and recursively expanding the nodes (if required) on those links. By avoiding unnecessary node expansion, a large number of searches can be avoided.

Each node stores two data items: (1) The currently cheapest path to the sink, represented by the first link in that path. This is initialised to null. (2) The cost of that path

**Algorithm 2** Flooding-based DBMA link pruning algorithm. Sets the best path from the sink on all nodes. $l$ is the link being used. $c$ is the cost of the pathway from the sink.

```
FUNCTION offerBestPath( l, c )
 IF this node's existing c < c THEN
  RETURN
 END IF

 STORE l and c

 FOR EACH link i from n
  IF i is l THEN
   SKIP
  END IF

  LET d = other node on i
  COMPUTE new c

  offerBestPath( i, c )
 END FOR
END FUNCTION

FUNCTION main()
 offerBestPath( null, 0 ) at sink

 FOR EACH node n
  ENABLE node n's chosen l
 END FOR
END FUNCTION
```

computed based on the cost of each of its links. This is initialised to infinity to ensure a valid pathway will be set.

Every node is expanded at least once. Subsequent visits to that node are on an 'offer' basis and can be rejected. If accepted, the data items are updated and the node is expanded by recursively attempting expansion of its connected neighbours. The order of expansion is depth-first.

This translates nicely into a network protocol where an offer message can be sent over a link and either processed recursively or rejected. Obviously other approaches are possible as well, but the purpose of this section is to demonstrate that an efficient approach is possible, not in finding the most efficient approach absolutely.

### Fitness Functions

The fitness function is used to compute the cost of a pathway, with higher values representing better pathways. There are two broad approaches: (1) Those based on required *area* and (2) those based on *overhead* contribution.

The *area* approach was used in the optimal, but infeasible, algorithm. Although it was not the primary cause of the in-feasibility, the area of the network has to be computed every time, which requires a large number of operations.

The *overhead contribution* approach is computationally simpler because it uses local metrics that are pre-computed. These local metrics can include:

**Physical Length** This is the Euclidean distance between the nodes. Since there is no clear correlation between the Euclidean and *measured* distance, it is unsurprising that the results based purely on this metric are ineffective.

**Maximum Measurement** Once example measurements have been taken, the maximum can be taken as the *worst case*. Unfortunately, no account is taken of *actual cost*. Links with high error deep inside the network cannot be used even though they do not incur high overhead. The ability to use these links can potentially widen the available path options for better *overall* optimality.

**Overhead Contribution** Based on the known ideal secure area, this signifies the worst boundary expansion needed if the link is enabled. The calculation is shown in Figure 6.16. Links with errors that do not require enlargement of the secure zone are not penalised. This approach improves the number of links that can be considered, particularly deeper in the network.

At first sight it may be tempting to suggest that choosing the cheapest option on each hop results in the best result, but this is not true since the whole pathway needs to be taken into account. Although results from this work were published [8] that suggest that using the average on the pathway is effective, this actually obfuscates the impact of a given pathway. The worst-case overhead contribution on the entire path must be used as the selection value.

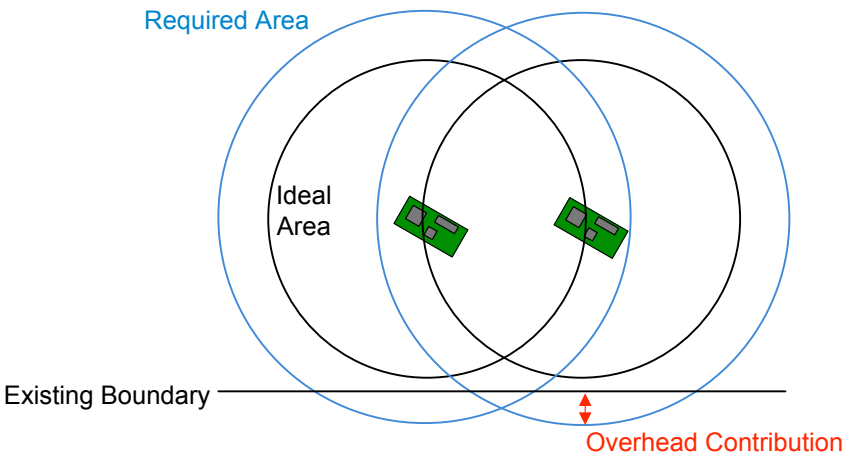



Figure 6.16: DBMA link overhead contribution. From each node on a link, and in each direction, the distance to the boundary is subtracted from the maximum measured distance (shown by the blue circles). The worst-case overlap is the maximum value found, with a minimum value of zero.

### 6.10.3 Comparison of Methods

Each of the proposed methods was executed using the test data obtained from each deployment. The comparison compares the performance of these executions based on the following properties: (1) optimality, (2) computational complexity and (3) computation time.[33]

| Indicator | Purpose | Units |
|---|---|---|
| $r$ | Area Requirement | $m^2$ |
| $i$ | Additional area required (over ideal) | % |
| $b$ | Additional area required (over optimal) | % |
| $m$ | Moves required | Moves |
| $n$ | Number of enabled links | Links |
| $t$ | Processing time | ms |

Table 6.7: DBMA link pruning algorithm performance indicators.

| Algorithm | $r$ $m^2$ | $i$ % | $b$ % | $m$ Moves | $n$ Links | $t$ ms |
|---|---|---|---|---|---|---|
| Ideal | 937.93 | - | - | - | - | - |
| Optimal | 972.03 | 3.64 | - | 16777216 | 12 | 385533 |
| Efficient (Phy) | 2285.88 | 143.72 | 135.17 | 100 | 12 | 1 |
| Efficient (Meas) | 1052.49 | 12.21 | 8.28 | 103 | 12 | 1 |
| Efficient (OHC) | 972.03 | 3.64 | 0.00 | 124 | 12 | 2 |

Table 6.8: Comparison of DMBA link pruning algorithms in deployment A.

Tables 6.8 and 6.9 show the results for deployments A and B respectively. Table 6.7 lists the performance indicators used. The optimality of a result is expressed as the area

[33]Normally computational time is not considered separately, but the desire to implement these algorithms in constrained hardware makes such a performance measure useful.

| **Algorithm** | $r$ m$^2$ | $i$ % | $b$ % | $m$ Moves | $n$ Links | $t$ ms |
|---|---|---|---|---|---|---|
| Ideal | 339.34 | - | - | - | - | - |
| Optimal | 337.03 | -0.68 | - | 131072 | 9 | 3199 |
| Efficient (Phy) | 766.43 | 125.86 | 127.41 | 84 | 9 | 6 |
| Efficient (Meas) | 337.03 | -0.68 | 0.00 | 72 | 9 | 1 |
| Efficient (OHC) | 341.92 | 0.76 | 1.45 | 70 | 9 | 1 |

Table 6.9: Comparison of DMBA link pruning algorithms in deployment B.

requirement $r$. $i$ and $b$ indicate the percentage of additional area required (wastage) in comparison with the ideal area and the optimal area respectively. $m$ indicates the number of moves needed by the algorithm, $n$ the number of links left enabled at the end and $t$ the time to complete the algorithm on the test machine.

Notice that it is possible to achieve a negative value in the case of $i$ as an optimal solution may not require use of the full ideal area. This occurs because not all the links are utilised.

Comparing the algorithms, in terms of additional overhead incurred, compared to the ideal area and optimal requirement reveals a high level of optimality in the efficient algorithm. This is shown in Figure 6.17 and Figure 6.18. In the case of deployment A, the overhead is $i = 3.64$ % and in deployment B the overhead is $i = 0.76$ %.

The flood algorithm is obviously more computationally efficient than a brute force attack. Whilst the execution times are listed, it is worth noting that the design of the Java runtime engine uses various runtime optimisations that result in differing behaviour. For example, the *just-in-time* compilation can result in subsequent executions of an algorithm completing much more quickly than the first. However, the relative magnitude between the timings remains consistent.

### 6.10.4 Network Protocol Feasibility

The flooding-based protocol could be implemented as an in-network protocol. Such a protocol would need to obtain distance measurements for each link. Assuming no topology has already been identified, the protocol could work as follows, for each node expanded, starting from the sink:

**1. Identify Neighbours** A broadcast is sent to notify available neighbours of an offer and the path.

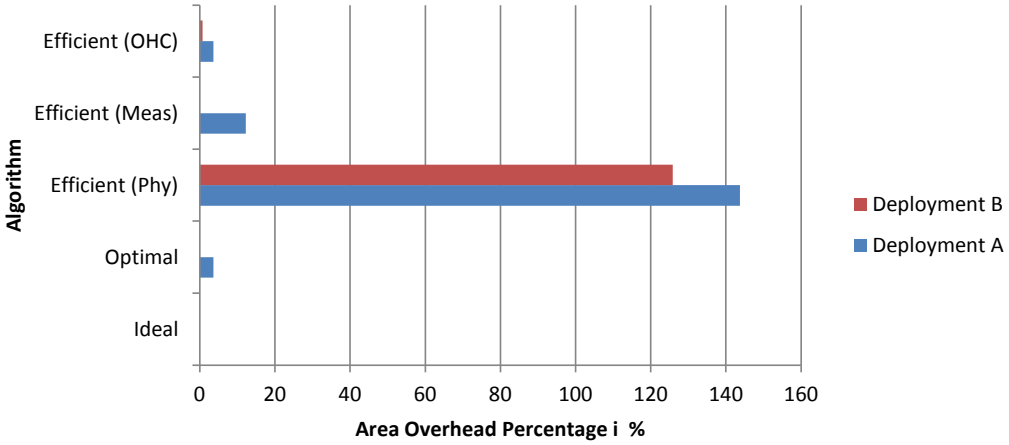



Figure 6.17: Overhead $i$% incurred, for each algorithm, over the ideal area as a result of link pruning in each deployment.

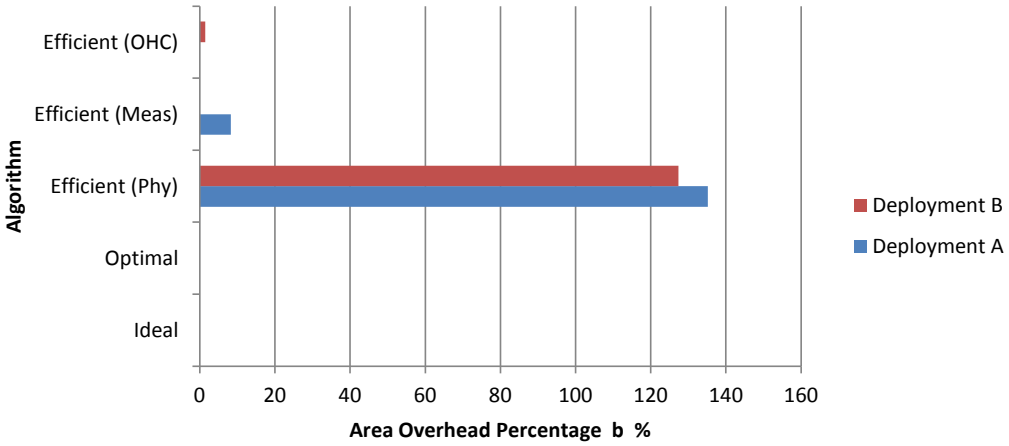



Figure 6.18: Overhead $b$% incurred, for each algorithm, over the optimal area as a result of link pruning in each deployment.

**2. Measure Distances** Each neighbour overhearing the broadcast initiates a number of secure RTT measurements against the broadcaster if it is not already in the path.

**3. Apply Fitness Function** If the link offer and measurements are better than any pre-existing selection, then that link is selected and stored with the relevant fitness value. That node then initiates phase 1.

The most important aspect is that it is not necessary to send all the data to a central point for processing: nodes can use purely local measurement data and the offer received in the broadcast. However, there are some security concerns that are now outlined.

First, the attacker must be excluded from participation. There are two approaches. If the ideal secure area has already been enforced, and if nodes know their location, then the protocol can run and apply a threshold test to ignore offers from outside the secure area. If the ideal area has not yet been selected, or nodes do not know their location, then it is necessary for the deployer to set an arbitrary limit and clear the associated area.

By performing RTT from the receiver, rather than broadcaster, an attacker is prevented from lying about his range due to the design of the secure RTT. This both prevents participation and any potential poisoning of the local routing data with false offers.

### 6.10.5 Alternative Strategies Discussion

Although this section has demonstrated that a heuristic can efficiently optimise the network, there are a number of ways in which the strategy can be modified. These are briefly introduced.

**Redundancy** These algorithms do not necessarily achieve an optimal level of link redundancy. The focus has been on minimising network area requirements, so no effort is made to enable redundant links. Nodes therefore cannot use links that did not need to be disabled. This can affect network resiliency should some links die. This could be added to either approach, or as a refinement step at the end.

**Node Relocation** It is obvious that the position of a node may have some impact on the accuracy of ranging measurements. In many sensing scenarios, a sensor may not

need to be in a precise location and some movement may be possible. The possibility of multiple node positions could therefore be considered in modified algorithms.

**Radio Design** The design of the antenna, its orientation and the selected modulation mechanisms may also have some bearing on accuracy and could also be modified. Although this might increase the cost of nodes, it may reduce the area requirement or improve performance, thus becoming a worthwhile investment.

**Union of Circles** The current area requirement $r$ is computed as a bounding rectangle, rather than as a union of the necessary communication radii. The algorithm therefore over-exaggerates the necessary requirement for each candidate. $r$ could instead be computed based on a union of circles, although this would be more computationally expensive.

**Overhead Contribution** The overhead contribution is based on one dimension rather than an area. The fitness function is therefore not accurate close to the corners of the ideal area where two dimensions are relevant. Different results would be obtained if the overhead could be based on the circular area, around each node, that extends outside of the ideal area. Again, this approach is more computationally expensive.

**Pre-existing Secure Zone** If an existing secure zone is available, this might be better taken into account when determining the required area or the overhead. For example, some more expensive candidates might be cheaper, or even free, if they are fully contained within the existing secure zone. The results also show that an optimal $r$ is not always fully contained within the zone (see Figure 6.14) so a union value is needed.

**Alternative Heuristics** Obviously the algorithm can be modified in terms of the search order and fitness functions. Indeed a slightly different fitness function was previously proposed as part of this work [8].

## 6.11 Summary of Findings

Cryptographic protocols that rely on keys present several challenges. The keys can be compromised either through theft or attack. In other cases, the protocols themselves can be

a target of resource-drain attacks. An additional layer of security to protect these protocols is therefore desirable.

Some secure deployments benefit from a secured zone that is free from attackers and their devices. By carefully placing nodes, or adjusting the zone perimeter, it is possible for node separation to always be less than their distance from the boundary. It is now feasible to measure the (radio) distance between nodes directly with the radio transceiver. This allows for integration of ranging with message exchange. This measured distance can be used to check if a node is within the secure area. An additional authentication mechanism is therefore available.

The underlying ranging mechanism must be secure, sufficiently accurate, inseparable from message exchange, deliver a feasible energy overhead and provide for a sensible secure zone area requirement. Many existing mechanisms for ranging do not support these needs because they are either infeasible or subject to manipulation by an attacker.

The Round-Trip-Time (RTT) mechanism avoids many manipulation problems since it is measured independently on one node and cannot be easily defeated without breaking the speed of light. It can be secured by including unpredictable *nonce* values in the ranging messages. RTT is therefore implementable within a message authentication protocol (such as RTTMAP).

RTT must be measured by the receiver and involves the inclusion of a payload. It requires three messages; the first message initiates the exchange, the second begins ranging and the third completes ranging and includes the payload. The use of cryptographic (but key-less) hash functions allows these three messages to be linked in such a way to prevent manipulation by an attacker.

The need for three messages increases the energy cost of RTTMAP compared to conventional schemes, although this communication-related energy cost is only a small increase. The other difference relates to the cost of the hash function in RTTMAP. The costs may be acceptable if protection is offered against the exposure of very expensive cryptographic functions, for example in public key cryptography.

Importantly, the design of RTTMAP results in a lower cost to *reject* a message; in some attacks, a malicious message can be rejected with less than a fifth of the energy resources

compared to a conventional cryptographic attack. The hash function cannot be arbitrarily attacked and often the transceiver can be powered down early. This results in a significant performance gain whilst under attack in comparison to traditional cryptographic measures.

RTTMAP affects the MAC layer by affecting clear channel assessment in *hidden terminal* scenarios. However, its design is similar to RTS/CTS mechanisms, allowing the clear channel assessment to be avoided before sending ranging responses

Although RTT is now available in WSN class devices, the ranging mechanisms on available hardware do not support the inclusion of a nonce or the injection of a payload during ranging. Some changes can be made to avoid hardware modification. It is possible to avoid using a nonce by instead changing the MAC address of the node that performs the ranging. Without knowledge of the MAC address, an attacker cannot transmit a response early. The payload can be sent in a separate payload. However, the ranging node needs an assurance that it is ranging in relation to that message. By changing the MAC address of the sender, an attacker outside the network cannot utilise genuine nodes as they will not change their MAC address accordingly.

The actual measurements carried out within RTT relate to the strongest RF pathway between the nodes. This pathway may not be line-of-sight and therefore measurement inaccuracies exist. Apart from some minor timing inaccuracy ($\pm 1m$), these errors are always positive and can reach about $10m$. They can be tolerated, but require that the secure zone is enlarged to prevent attackers using special apparatus to optimise their RF pathways.

Any increase in secure zone size can be highly impractical, so methods are needed to reduce or eliminate any expansion. One such method is to exploit network redundancy and switch off links that are not needed to form a fully connected network but which exhibit high error. It is necessary to collect real measurements for this selection process as there is little correlation between Euclidean distance and RF measured distance due to complex and unpredictable environmental issues.

The most optimal method to select such links is to try every combination of enabled and disabled links, calculating the required area for each fully connected network and then choosing the solution with the lowest area. Unfortunately this approach is centralised, scales exponentially with network size and quickly becomes infeasible. It is possible to implement

an efficient algorithm since a fitness function can be applied during the search. This provides for similar results with significantly lower computational requirements.

A core benefit of RTTMAP is the need for an attacker to obtain better hardware. The main issue is the efficiency, and security, of the turnaround at the sender during ranging. The attacker would need to respond faster or find a weakness in the communication protocols. To exploit such improvements, hardware re-design would be necessary, which may be far less practical than many cryptographic attacks. For example, many attacks against legacy Wi-Fi networks are possible using freely available software; no special hardware is normally needed. This hardware-level engineering challenge is also likely to lead to solutions for friendly parties as well as attackers. For example, if a late commit attack is developed, friendly parties could also commit late. Alternatively, if an attacker develops a device to respond a few nanoseconds earlier, the same device could be used by a friendly party. Therefore, there is likely to be a period where optimisations will be continually developed by both parties. Older devices used by friendly parties will become less useful and outdated, but the level of optimisation will eventually become harder and harder. Eventually optimisations may only be able to reduce the response time by nanoseconds, meaning that an extended boundary would solve the problem for friendly parties. At some point, these optimisations will arrive at the problem of breaking the speed of light, a barrier that nobody has yet crossed with communications technology.

## 6.12 Conclusion

This chapter has introduced the concept of Distance-Based Message Authentication to provide additional physical-layer protection for WSN protocols. An implementation using secure RTT measurement has been proposed called Round-Trip-Time Message Authentication Protocol. RTTMAP is shown to provide physical security in some scenarios by restricting access to nodes located within a defined security boundary. Although RTTMAP was found to cost slightly more in accepting genuine messages, it was found to be more efficient in rejecting malicious messages as it can terminate early.

An implementation of RTTMAP for sensor nodes equipped with the NanoLOC TRX

NA5TR1 transceiver was created and demonstrated that DBMA provides a practical additional layer of defence for WSN deployments. Using data from real deployments, the impact of propagation error on secure area overhead was evaluated. Approaches to optimise this overhead were proposed using link pruning techniques, achieving near 100% savings in two test deployments.

Future work could consider optimised implementations of the raw RTTMAP protocol in hardware.

# Chapter 7

# Conclusion

Wireless sensor networks were deliberately conceived to be architecturally limited in order to benefit from lower costs and infrequent battery changes. This architecture has imposed a number of limitations that have been the subject of academic and industrial efforts for many years. The number of *potential* applications has steadily increased, leading to a number of new security challenges as some are industrialised. In high security applications, the focus has been on both defending against existing attacks and defending against new breeds that are specific to WSNs, such as those that directly target limited node resources.

A particular focus in this thesis has been high security physical intrusion detection systems that utilise WSN technology. These systems have existed in cabled form for some time, but the move to wireless systems has resulted in security problems, leading to poor security grading, and issues with scalability and key management. Directly applying existing WSN solutions to this problem would have resulted in continuing security limitations, feasibility issues and efficiency concerns. A number of changes were necessary to achieve high security authentication with low overheads; these included the use of end-to-end cryptographic authentication, the implementation of energy-efficient key management and the inclusion of additional defences at the physical layer. These changes better protect against attacks that involve the modification of nodes and wireless attacks from outside a physically secured zone. Importantly, they were found to have feasible overheads, particularly in terms of energy consumption.

In order to properly evaluate both existing and contributed schemes, it was necessary to obtain performance metrics from common WSN hardware. Common, industry-strength,

cryptographic algorithms were tested on the popular MSP430 microcontroller to obtain processing delays and energy costs. Message transmission using two WSN transceivers, the CC2420 and the NA5TR1, was also evaluated to obtain energy costs for message exchange. The methods for characterising these costs had to be carefully considered to deal with energy-saving mechanisms employed in many modern MAC protocols.

Although existing cryptographic algorithms were available for WSNs, only the symmetric algorithms were found to be computationally feasible for repeated use. The time taken for a typical frame encryption is about $20$ms on the popular MSP430 microcontroller, whilst an elliptic curve operation can take up-to $60$ seconds. Experiments with asymmetric algorithms revealed not only issues with high computational load, but also with real-time implications in the basic operating systems present on WSN nodes. The use of symmetric algorithms required the confidential distribution of symmetric keys between end-points, which is not required when distributing public keys. Unfortunately, direct application of the existing WSN key management protocols would have been inappropriate; many of these protocols were intended for use with link-layer security and adapting them for end-to-end security would have been either impossible or carried significant computational and communication overhead. It is important to avoid such overheads both because of performance generally and also so that keys can be replaced during an attack quickly and with minimal network disruption. Otherwise, an attacker may be able to exploit this to carry out an attack and escape without being reliably detected.

Broadcast Key Exchange was proposed (see Chapter 5 and publications [3, 4, 5]), as a principle to address this problem where the sink needs to establish a separate symmetric key with each node in the network. Rather than sending a single message to each node in a unicast approach, the principle involves a single broadcast.

Whilst this reduced the communication burden on nodes, implementations of BKE are required to use algorithms to compute the keys. The Diffie-Hellman variant of BKE, for example, includes an expensive cryptographic function. This level of computational overhead is not necessarily present in conventional protocols. The need to run this algorithm increases the energy consumption of nodes when keys are replaced, meaning that BKE has a generally higher energy footprint than conventional approaches. One notable exception

was chain networks of over $135$ nodes, since the communication cost increases significantly in the conventional unicast key distribution case. However, BKE was found to better balance the combined computational and communication overhead in multi-hop networks; nodes that are closer to the sink do not need to forward a large number of messages, meaning that their energy load with BKE – even with the computational overhead – is considerably improved and extends their lifetime. A theoretical evaluation found BKE/D to be beneficial, compared to unicast rekeying, in multi-hop networks with over $67$ nodes, regardless of topology, connected through a single critical node. Improvements in cryptographic efficiency on the WSN platform continue to appear, making BKE even more beneficial. Several alternative algorithms for use with the BKE concept were also identified to avoid the need for public key operations where this is useful; such as on new, or more constrained, platforms.

Real conditions were found to involve some level of transmission loss. Even in an interference-free environment, loss can still be encountered due to implementation specifics. This loss had to be handled to ensure that the broadcast reaches all nodes. Practical experiments tested different approaches. The use of dedicated acknowledgment messages was found to be wasteful and actually increased communication losses and overheads. In one scenario, the acknowledgment messages collided frequently resulting in the need to resend the message seven times when it reached most nodes on the first attempt. Manually re-sending messages from the sink was found to carry high levels of penalty, especially in unicast scenarios. The most efficient solution was to re-use sink-bound report messages as acknowledgments and then to re-transmit cached broadcasts from *within* the network when loss was detected. The loss recovery approach within SecureTDRoute was thus validated. Communication performance of the conventional unicast approaches was also evaluated to validate the communication benefits. The levels of loss seen, particularly in the chain topology, made BKE beneficial in much smaller networks than anticipated. For example, the critical node energy cost in the chain network, which had $14$ nodes, was high enough to permit four rounds of scalar point multiplication in some unicast cases; but, the network was nowhere near the theoretical crossover point of $135$ nodes, predicted earlier.

The expensive cryptographic algorithms found in Diffie-Hellman caused some concern about the vulnerability of constrained systems against resource-drain attacks. On WSN plat-

forms, an attacker need only inject messages to waste resources on the nodes; an additional layer of protection before these algorithms are executed was thus investigated at the physical layer.

In high security scenarios, the WSN is often installed in a secured environment where attacker access can be prevented. By securely measuring the distance between nodes during message transfer, an assurance could be made about the location of communication partners; this concept (see Chapter 6 and publications [6, 7, 8]) was named Distance-Based Message Authentication. DBMA allows messages to be rejected if they originate from outside a permitted radius, located wholly within the secured zone.

An implementation of the DBMA concept called the Round-Trip-Time Message Authentication Protocol (RTTMAP) was devised and implemented. RTTMAP incurs a slight increase in channel occupation and requires more specific approaches to clear channel assessment due to its real-time design. However, it provides the opportunity to reject messages based on a secure distance measurement that cannot be reduced or hijacked by an attacker. Practical experiments with the Nanotron NA5TR1 investigated its energy overheads and ranging accuracy.

The energy overhead of RTTMAP was found to be higher in comparison to the conventional method of sending a single frame and executing a cryptographic function. However, the design of RTTMAP allows for early termination of the process when it is attacked, resulting in better performance under attack – less than a fifth of the energy – than the conventional method. The higher cost of RTTMAP in general use may be acceptable if it prevents exposure of very expensive energy attacks, such as those targeting public key cryptography. Improvements in transceiver and algorithm implementation are expected to improve this performance significantly.

The ranging inaccuracy, experienced in the real world, results in the need for an enlarged security zone in some deployments. Whilst the nodes can be moved around within the zone to avoid this, another approach investigated involved disabling links that are not needed to form a fully-connected network but which otherwise incur high area overheads. A method to find the optimal solution was evaluated and found to scale badly. An efficient method was therefore devised that is distributable in the network and significantly reduces computational

overhead whilst finding a near-optimal solution. In the tested deployments, solutions with less than $1.45\%$ overhead could be found.

This thesis has contributed two major WSN security improvements. Firstly, the use of end-to-end message authentication has been made more viable by the concept of Broadcast Key Distribution. The use of end-to-end message authentication no longer need adversely affect communication performance during re-keying and the improved balance of energy consumption increases the lifetime of important nodes. Second, the risk of attacks against the restricted resources of sensor nodes has been mitigated in some environments by providing protection at the physical layer. Attackers that are located outside of a secured zone are unable to inject messages that are subsequently processed above the physical layer, even if they have stolen or compromised keys in the network.

The use of WSN technology in physical intrusion detection systems is therefore considerably more viable as the main shortcomings identified in existing systems have been addressed. These systems can now operate with considerably stronger authentication matched to the high security scenarios.

## 7.1 Future Work

Significant portions of the overhead incurred in both BKE and RTTMAP are from the computational element. In BKE, this computational overhead has an effect on the total energy consumption in the network during re-keying as well as the size of network where BKE becomes beneficial. In RTTMAP, the computational overhead from the hash function increases the energy cost in exchanging messages between friendly parties. Although these overheads are acceptable, optimised implementations of the cryptographic protocols, either in software or hardware, would increase the benefits of both BKE and RTTMAP. Some cryptographic optimisation has been seen already, such as those mentioned in Section 4.2.1. Alternatively, the evaluation of alternative cryptographic algorithms that offer acceptable security with lower overheads would be a useful contribution.

BKE works fairly when there is a need to re-key the entire network; the useful remaining lifetime of the keys in the network is well balanced and the key management load is fairly

distributed. Unfortunately, some WSN applications involve unbalanced key use. For example, some applications might send sensor reports from some nodes more often than others. Other applications might sporadically send larger messages over the WSN; for example, images from cameras. In these scenarios some ('greedy') nodes will require new keys before others. If BKE is used in these scenarios, there may be resource wastage as many keys are discarded prematurely. The high cost of re-keying may be a motivation to maintain the life of keys for as long as possible. Methods that can efficiently and securely re-key a subset of nodes, without incurring high overheads on other nodes, might therefore be useful.

RTTMAP relies on very secure ranging such that the attacker cannot respond to a PROBE message significantly faster, or otherwise alter the behaviour of the ranging, such that the distance measurement is shorter than reality. There is always the risk that an attacker might develop new hardware do this. Whilst the implementation on the NA5TR1 was useful for performance measurement, further work is required to evaluate what changes may be needed (such as those referenced in Section 4.4) to avoid this risk. Even though this risk exists, it is important to emphasise its difficulty; an attacker cannot simply carry out a cryptanalysis attack, instead he will have to carry out complex work at a very low hardware level. This in itself is a security benefit, and it is likely such work will be useful as a defence. In time optimisations by both friendly and hostile parties will reach a limit that cannot be crossed: potentially, the speed of light barrier.

RTTMAP also uses only distance as the authentication parameter. This is useful in many scenarios, but means that the full distance radius has to be protected around a node. It would be useful to have other physical properties available. Very recently, researchers have attempted to use angle-of-arrival as an authentication method [134]. Combining these approaches would allow nodes to be sited closer to boundaries and for the above mentioned issue with secure ranging to be mitigated.

Finally, the EN50131 standard imposes considerable requirements at Grade 4 and no wireless system, at this time, is known to have achieved it. Many of the timing requirements at Grade 4 are in the region of just $10$ seconds, including the maximum channel unavailability and the maximum time permitted to detect the delay of messages. Given the need to keep communications energy overhead low in a WSN, and the exposed nature of the communi-

cations medium, it will be interesting to see if Grade 4 will ever be met by a WSN or if Grade 4 will need to be re-defined.

# Appendix A

# Binary Tree Routing

SecureBTRoute is an extension of SecureTDRoute (see Section 5.4.1) that provides downstream routing of unicast messages in a static network topology. SecureBTRoute is useful for the purposes of evaluation, but may be unsuitable in other scenarios.

SecureBTRoute uses the concept of binary tree address allocation to limit the necessary routing information held on each node and avoids the need for, potentially unpredictable, automated topology setup. The network topology must follow the binary tree rules; all children to the left of a node must have lower addresses and all those to the right have higher addresses. Routing decisions are simplified, as shown in Algorithm 3.

**Algorithm 3** Binary tree routing algorithm.

```
Let d = msg final dest, s = msg last sender
Let n = local address, p = parent address
Let l = left child, L = lowest left address
Let r = right child, R = highest right address

If d == n then pass msg to higher layer and stop
Else if d < n and d >= L then let i = l
Else if d > n and d <= R then let i = r
Else i = p

If i == s then drop msg (to avoid cycles) and stop
Else Forward msg to i
```

# Appendix B

# Communication Overhead in Key Distribution

| **Topology** | **UKE** $Q$ | **BKE** $Q$ |
|---|---|---|
| One-hop | $0$ | $0$ |
| Chain | $\sum_{i=1}^{N} i - 1$ | $N - 1$ |
| Binary Tree | $\sum_{i=1}^{N} \lfloor \log_2(i+1) \rfloor - 1$ | $2^{\lfloor \log_2(i+1) \rfloor} - 2$ |

Table B.1: Calculation of overall key transmissions required.

The computation of the minimum number of transmissions $Q$ required to disseminate material to all nodes from the sink, excluding sink transmissions, is linked to the topology and size $N$ of the network. In all cases $i$ is a unique node ID, numbered $1 \cdots N$ with $1$ being adjacent to the sink and $N$ the furthest.

The single-hop case is the most straightforward; no node, but the sink, needs to transmit, thus $Q = 0$ in both UKE and BKE. In the chain case, it is easy to see that the number of transmissions in BKE is equal to $N - 1$ since all nodes must broadcast, except for the sole leaf node. In UKE, $i$ is the depth of the node, so each node $i$ requires $i - 1$ nodes to transmit on its behalf (i.e. not itself or the sink at depth $0$).

The binary tree case is much more complex. In the case of UKE, the calculation requires knowledge of the number of hops needed to reach each node. This is obtained by computing the binary logarithm on $i + 1$ (offsetting the address as the sink is not counted), flooring the result (the number of hops) and then subtracting $1$ (to eliminate the uncounted sink hop).

In the case of BKE, the number of hops in all depths previous has to be obtained. This

value is obtained by first computing the maximum number of nodes in the current depth: the binary logarithm of $i + 1$ (again, because the sink is not counted) is first computed, the result is floored as before, and $1$ is *not* subtracted (as the true depth is needed). $2$ is then risen to the power of the result. Finally, $2$ is subtracted because the sink is not counted and each layer of a binary tree always has one more node than the sum of all nodes in previous layers. This formula is a simplification; some of the nodes in the previous layer may not need to transmit if they are leaf nodes, but this is omitted for reasons of brevity.

# Appendix C

# MAC Protocol Design Issues

The performance of WSN communication protocols is tied not only to the transceiver, but also to the MAC protocol. The MAC protocol is therefore an important consideration since its behaviour alters attributes such as collision likelihood, the number of retries, back off periods and delays caused by throughput available to nodes. This section discusses various MAC protocol design issues to assist readers to understand the issues presented elsewhere in the thesis. The section is best read in combination with Section 3.3.

## C.1 Channel Contention

A *medium access control* (MAC) protocol primarily controls access to the medium. Uncontrolled access would otherwise allow the potential for transmission by multiple nodes simultaneously and therefore data loss. This scenario is called *collision* and there are two common methods to avoid it: *contention-free* and *contention-based*.

**Contention-free** Sometimes referred to simply as *scheduled* or *time division multiple access* (TDMA), these protocols divide fixed time periods into slots, and allocate those slots for the exclusive use of individual nodes. Theoretically in a perfect, interference-free, environment this approach will provide 100% reliability and determinism.

**Contention-based** Contention-based protocols are needed if contention-free operation is infeasible. Nodes can attempt to access the channel at any time; nodes either try to either avoid collisions in the first place, or they detect collision and attempt to transmit

again when they do occur. In both cases, nodes back off for a small, usually random, period before trying again. Contention-based protocols are less deterministic than scheduled protocols because of the uncertainty about back off delay.

**Collision Avoidance** By checking the channel before transmitting using *clear channel assessment* (CCA), nodes can avoid transmitting when other nodes are using the channel. This is only successful if both nodes are *not* perfectly synchronised, as perfectly synchronised nodes will not detect the transmissions of each other leading to collisions. CCA does not handle the hidden terminal problem as the transmitter is not in the same RF space as the receiver; this is discussed below.

**Collision Detection** When a collision does occur then a new transmission can be attempted. This is often implemented by using acknowledgements, where nodes repeat transmissions if no acknowledgement is received, or repeat requests where a node that receives a damaged frame can request transmission. Alternatively, nodes may be able to detect collision during transmission.

Some MAC protocols support *beaconing*, where instead of using time synchronisation, nodes transmit according to a beacon signal. These beacons can be sent from either dedicated or negotiated nodes. Some protocols combine several of these approaches; for example, IEEE 802.15.4 allows scheduling in some slots with contention in others. This allows some nodes to benefit from guaranteed time slots whilst others that are less critical can share a smaller time period with contention.

## C.2 Hidden Terminals

The *hidden terminal problem* (see Figure C.1) is caused by the sender and receiver being located in different RF areas. When the sender performs CCA and believes that the channel is free, the channel may actually not be free at the receiver; any frames sent in this scenario will collide. It may be acceptable to lose some frames in the network, particularly when retransmission can be arranged, but other protocols aim to avoid this problem. The problem is particularly problematic when long-duration transmissions are needed as the opportunity for a collision increases with time.

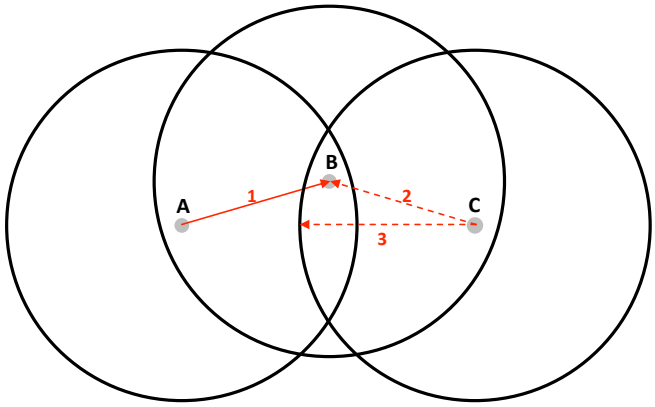



Figure C.1: The hidden terminal problem. Node A transmits a frame believing that the channel is free, but causes a collision at node B since it cannot overhear the frame being sent from node C.

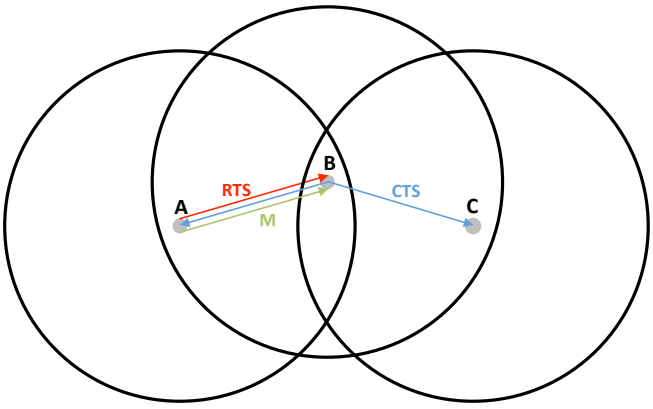



Figure C.2: RTS/CTS hidden terminal mitigation.

It is insufficient to merely extend the CCA period to handle this problem as there will be no indication from the receiver that any ongoing transmissions have begun. The RTS/CTS (ready-to-send/clear-to-send) mechanism, found in IEEE 802.11, addresses this problem. In RTS/CTS, a node that wishes to send data will first send a RTS frame to the receiver, which then responds with a CTS frame (see Figure C.2). Any node that overhears either frame will back off for a sufficient period to allow the exchange to occur uninterrupted.

The risk of collision remains during the RTS stage, but the RTS frame is smaller, thus reducing the probability. As this risk of collision is still present, RTS/CTS is not effective for *short* payloads since the chance of collision is similar. For example, IEEE 802.11 can be configured to use RTS/CTS only for long payloads. Also, RTS/CTS can fail since communication distance is not equal to *interference distance* [135].

## C.3 Energy Efficiency

Regardless of the efficiency of WSN transceiver hardware, it is desirable to put the hardware into a sleep mode as much as possible to save energy; changes at the MAC layer are usually needed to support this. *Duty-cycling* MAC protocols only wake up the transceiver for small time intervals. To allow nodes to exchange messages, it is necessary for both sender and receiver to be active at the same time. Duty-cycling is easy to implement in a TDMA scheme since the clocks are synchronised and the receiver can wake up at the time when a frame is likely to be received. In contention-based MAC protocols, no such certainty exists, and special approaches are needed.

Duty-cycling has an impact on clear channel assessment. Since the radio is not continually listening, it may miss ongoing message exchanges. Channel assessment therefore cannot be carried out *opportunistically* before a transmission is needed. The assessment period must therefore be sufficiently long to tolerate unheard transmissions. This approach avoids the need to synchronise clocks, but can reduce throughput due to the wait period.

Unfortunately, this requirement also means that RTS/CTS protocols are not immediately effective in a duty-cycled radio network; nodes are not guaranteed to overhear the RTS or CTS frames. Thus, where RTS/CTS is used, the CCA period has to be extended to

ensure that an existing exchange can complete before a RTS frame is sent. This reduces throughput.

# Bibliography


[1] A. Sloman, "The Reith Lectures 1963 - A University in the Making: Lecture 4. The Fulfilment of Lives," 1963. BBC Home Service.

[2] MoteIV, *Telos. Ultra low power IEEE 802.15.4 compliant wireless sensor module. Revision B : Humidity, Light, and Temperature sensors with USB*. MoteIV Corporation, USA, 2004.

[3] A. Chung and U. Roedig, "Efficient Key Establishment for Wireless Sensor Networks Using Elliptic Curve Diffie-Hellman," in *Adjunct Proceedings of the 2nd European Conference on Smart Sensing and Context (EuroSSC 2007)*, Springer-Verlag, 2007.

[4] A. Chung and U. Roedig, "Poster Abstract: DHB-KEY - A Diffie-Hellman Key Distribution Protocol for Wireless Sensor Networks," in *Adjunct Proceedings of the 5th European Workshop on Wireless Sensor Networks (EWSN2008)*, Springer-Verlag, 2008.

[5] A. Chung and U. Roedig, "DHB-KEY: An Efficient Key Distribution Scheme for Wireless Sensor Networks," in *Proceedings of the 4th IEEE International Workshop on Wireless and Sensor Networks Security (WSNS2008) at IEEE MASS 2008*, IEEE, 2008.

[6] A. Chung and U. Roedig, "On The Feasibility of a New Defense Layer for Wireless Sensor Networks using RF Ranging," in *Proceedings of the 1st IEEE International Conference on Network and Service Security (N2S2009)*, IEEE, 2009.

[7] A. Chung and U. Roedig, "Poster-Abstract: Implementation of Distance-Based Message Authentication for WSNs," in *Adjunct Proceedings of the 7th European Workshop on Wireless Sensor Networks (EWSN2010)*, Springer-Verlag, 2010.

[8] T. Chung and U. Roedig, "Implementation and Evaluation of Distance-Based Message Authentication," in *Proceedings of the 7th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS2010)*, pp. 519–524, IEEE, 2010.

[9] T. Chung, "Poster: Locking Out The Evil Guys - Without Using Keys," 2010. SciTech Christmas Conference 2010, Lancaster University, UK. `http://www.lancs.ac.uk/sci-tech/christmas_conference/`.

[10] Raza, S. and Duquennoy, S. and Chung, T. and Yazar, D. and Voigt, T. and Roedig, U., "Securing Communication in 6LoWPAN with Compressed IPsec," in *Proceedings of the 7th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'11)*, IEEE, 2011.

[11] B. Schneier, *Applied Cryptography*. John Wiley & Sons Inc., USA, 2 ed., 1996.

[12] BSI, "BS EN 50131-1:2006+A1:2009 Alarm Systems - Intrusion and hold-up systems - Part 1 System Requirements," 2009. British Standards Institute, UK.

[13] Castle Care-Tech, *Installing iD PLUS Systems*. Castle Care-Tech Ltd, UK, 2004. http://www.castle-caretech.com/wh/manuals/misc-id-plus.pdf.

[14] P. O'Connor, "Taking the Fear out of Wireless," 2010. Global Security Devices, Ireland. http://www.globalsecurity.ie/.

[15] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel, "A Practical Attack on KeeLoq," in *Proceedings of the 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2008)*, International Association for Cryptologic Research, 2008.

[16] BBC, "Sony launches legal action against PlayStation hackers," *BBC News Online*, January 2010. http://www.bbc.co.uk/news/technology-12171423.

[17] Texecom, *Professional Wireless Security with RICOCHET Mesh Technology*. Texecom Wireless Security, UK, 2010. http://www.texe.com/ricochet/pdfs/RICOCHETTechnologyLeaflet.pdf.

[18] B. Sims, "Honeywell: taking intruder alarms to a new Dimension," 2010. http://www.info4security.com/story.asp?storycode=4123913.

[19] Intel, *Intel PXA255 Processor: Electrical, Mechanical and Thermal Specification*. Intel Corporation, USA, 2004. http://www.intel.com/.

[20] Microchip, *PIC16F84A Data Sheet*. Microchip Technology Inc., USA, 2001. http://ww1.microchip.com/downloads/en/DeviceDoc/35007b.pdf/.

[21] Texas, *MSP430F15x, MSP430F16x, MSP430F161x Mixed Signal Microcontroller*. Texas Instruments Incorporated, USA, 2009. http://www.ti.com/lit/ds/symlink/msp430f1611.pdf.

[22] IEEE, "IEEE 802 Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)," 2006. http://standards.ieee.org/findstds/standard/802.15.4-2006.html.

[23] Chipcon, *Chipcon CC2420: 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver*. Chipcon AS, Norway, 2006. http://www.ti.com/lit/ds/symlink/cc2420.pdf.

[24] Crossbow, *MICA Wireless Measurement System*. Crossbow Technology Incorporated, USA, 2004. http://www.xbow.com/.

[25] Eneida, "Long Range Smart Active Tag," 2010. Eneida, Portugal. http://www.eneida.pt/.

[26] J. Barton, "A Modular Platform for Wireless Sensor Network Technology Development," 2009. Tyndall, Ireland. http://www.tyndall.ie/.

[27] C. http://www.coalesenses.com/, "iSense: Europe's latest Hardware and Software Platform for Wireless Sensor Networks," 2008. Coalesenses GmbH, Germany.

[28] U. Roedig, S. Rutlidge, J. Brown, and A. Scott, "Towards Multiprocessor Sensor Nodes," in *Proceedings of the 6th Workshop on Hot Topics in Embedded Networked Sensors (HotEmNets2010)*, ACM, June 2010.

[29] T. Nakagawa, M. Miyazaki, G. Ono, R. Fujiwara, T. Norimatsu, T. Terada, A. Maeki, Y. Ogata, S. Kobayashi, N. Koshizuka, *et al.*, "1-cc computer using UWB-IR for wireless sensor network," in *Proceedings of the 2008 Asia and South Pacific Design Automation Conference*, pp. 392–397, IEEE, 2008.

[30] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *ACM Sigplan Notices*, vol. 35, no. 11, p. 104, 2000. ACM.

[31] A. Dunkels, B. Grönvall, and T. Voigt, "Contiki - A Lightweight and Flexible Operating System for Tiny Networked Sensors," in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*, IEEE, 2004.

[32] D. Gay, P. Levis, R. Von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC language: A holistic approach to networked embedded systems," in *Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation*, p. 11, ACM, 2003.

[33] A. Dunkels, O. Schmidt, T. Voigt, and M. Ali, "Protothreads: simplifying event-driven programming of memory-constrained embedded systems," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pp. 29–42, ACM, 2006.

[34] C. Duffy, U. Roedig, J. Herbert, and C. Sreenan, "Adding preemption to TinyOS," in *Proceedings of the 4th workshop on Embedded networked sensors*, p. 92, ACM, 2007.

[35] J. Granjal, R. Silva, E. Monteiro, J. Silva, and F. Boavida, "Why is IPSec a viable option for wireless sensor networks," in *Proceedings of the 4th IEEE International Workshop on Wireless and Sensor Networks Security (WSNS2008) at IEEE MASS 2008*, IEEE, 2008.

[36] Shamus Software, *MIRACL User's Manual*, 2007. Shamus Software Ltd., Ireland. `http://www.shamus.ie/`.

[37] P. Szczechowiak, L. Oliveira, M. Scott, M. Collier, and R. Dahab, "NanoECC: Testing the limits of elliptic curve cryptography in sensor networks," in *Proceedings of the 5th European conference on Wireless sensor networks*, pp. 305–320, Springer-Verlag, 2008.

[38] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 115–121, 2006. IEEE.

[39] J. Benson, T. O'Donovan, U. Roedig, and C. Sreenan, "Opportunistic Aggregation over Duty Cycled Communications in Wireless Sensor Networks," in *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN'08)*, pp. 307–318, IEEE, 2008.

[40] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, ACM, 2004.

[41] Broadcom, *BCM4326 Product Brief*. Broadcom Corporation, USA, 2006. `http://www.broadcom.com/`.

[42] Lampe, J. and Hach, R. and Menzer, L., "IEEE P802.15: Nanotron Chirp Spread Spectrum Proposal," tech. rep., Nanotron Technologies, 2005. http://www.nanotron.com/.

[43] I. Marshall, M. Price, H. Li, and S. Boult, "Multi-sensor Cross Correlation for Alarm Generation in a Deployed Sensor Network," in *Proceedings of the 2nd European conference on Smart Sensing and Context*, pp. 286–299, Springer-Verlag, 2007.

[44] D. Halperin, T. Heydt-Benjamin, B. Ransford, S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. Maisel, "Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 129–142, IEEE, 2008.

[45] HART Communication Foundation, "WirelessHART Technical Data Sheet," tech. rep., HART Communication Foundation, 2007. http://www.hartcomm.org/protocol/training/resources/wiHART_resources/wirelesshart_datasheet.pdf.

[46] P. J. Marrón, "CONET: Cooperating Objects NETwork of Excellence," 2009. University of Duisburg-Essen, Germany. http://www.cooperating-objects.eu/.

[47] C. J. Sreenan, "GINSENG: Performance Control in Wireless Sensor Networks," 2009. University College Cork, Ireland. http://www.ict-ginseng.eu/.

[48] Thales, "Wireless Sensor Networks (White Paper)," 2009. Thales Research, UK. http://www.thalesresearch.com/Publications/WhitePapers/Documents/NET090901.pdf.

[49] J. Foley, "Recent Developments in the Design of Sensor Network Architectures." Presented at the 2nd European Conference on Smart Sensing and Context, 2007. British Telecommunications, UK.

[50] S. Huang and S. Shieh, "Authentication and secret search mechanisms for RFID-aware wireless sensor networks," *International Journal of Security and Networks*, vol. 5, no. 1, pp. 15–25, 2010. Inderscience.

[51] BBC, "BP oil spill: The environmental impact one year on," April 2011. http://www.bbc.co.uk/news/science-environment-13123036.

[52] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978. ACM.

[53] A. Perrig, J. Stankovic, and D. Wagner, "Security in wireless sensor networks," *Communications of the ACM*, vol. 47, no. 6, pp. 53–57, 2004. ACM.

[54] V. Miller, "Use of Elliptic Curves in Cryptography," in *Proceedings of Advances in Cryptography 1985 (CRYPTO '85)*, Springer, 1986.

[55] N. Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, January 1987. American Mathematical Society.

[56] S. Moon, "Elliptic Curve Scalar Point Multiplication Using Radix-4 Booth's Algorithm," in *Proceedings of the International Symposium on Communications and Information Technologies*, pp. 80–83, IEEE, 2004.

[57] M. Scott and P. Szczechowiak, “Optimizing multiprecision multiplication for public key cryptography,” tech. rep., Dublin City University, Ireland, 2007.

[58] P. Balasubramaniam and E. Karthikeyan, “Elliptic curve scalar multiplication algorithm using complementary recoding,” *Applied Mathematics and Computation*, vol. 190, no. 1, pp. 51–56, 2007. Elsevier.

[59] D. Malan, M. Welsh, and M. Smith, “A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography,” in *Proceedings of the 1st Annual IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, pp. 71–80, IEEE, 2004.

[60] V. Gupta, M. Wurm, Y. Zhu, M. Millard, S. Fung, N. Gura, H. Eberle, and S. Chang Shantz, “Sizzle: A standards-based end-to-end security architecture for the embedded Internet,” *Pervasive and Mobile Computing*, vol. 1, no. 4, pp. 425–445, 2005. Elsevier.

[61] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus, “TinyPK: securing sensor networks with public key technology,” in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 59–64, ACM, 2004.

[62] W. Hu, P. Corke, W. Shih, and L. Overs, “secfleck: A public key technology platform for wireless sensor networks,” in *Proceedings of the 6th European Conference on Wireless Sensor Networks (EWSN2009)*, pp. 296–311, Springer, 2009.

[63] A. Wood and J. Stankovic, “Poster Abstract: AMSecure - Secure Link-Layer Communication in TinyOS for IEEE 802.15.4-based Wireless Sensor Networks,” in *Proceedings of the 4th international conference on Embedded Networked Sensor Systems*, pp. 395–396, ACM, 2006.

[64] J. Daemen and V. Rijmen, “AES Proposal: Rijndael,” tech. rep., Proton World International and Katholieke Universiteit Leuven, 1999.

[65] C. Karlof, N. Sastry, and D. Wagner, “TinySec: a link layer security architecture for wireless sensor networks,” in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 162–175, ACM, 2004.

[66] C. Madson and R. Glenn, “The Use of HMAC-SHA-1-96 within ESP and AH.” RFC 2404 (Proposed Standard), Nov. 1998. `http://www.ietf.org/rfc/rfc2404.txt`.

[67] Arch Rock, “IP-based Wireless Sensor Networking: Secure, Reliable, Low-Power IP Connectivity for IEEE 802.15.4 Networks,” tech. rep., Arch Rock Corporation, USA, 2007.

[68] J. Loughney, “IPv6 Node Requirements.” RFC 4294 (Informational), Apr. 2006. Updated by RFC 5095. `http://www.ietf.org/rfc/rfc4294.txt`.

[69] HART, “WirelessHART Security Overview,” tech. rep., HART Communication Foundation, USA, 2010.

[70] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. Tygar, “SPINS: Security Protocols for Sensor Networks,” in *Proceedings of 7th Annual International Conference on Mobile Computing and Networks (MOBICOM 2001)*, ACM, 2001.

[71] J. Deng, R. Han, and S. Mishra, "Secure code distribution in dynamically programmable wireless sensor networks," in *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, pp. 292–300, ACM, 2006.

[72] P. Lanigan, R. Gandhi, and P. Narasimhan, "Sluice: Secure Dissemination of Code Updates in Sensor Networks," in *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*, IEEE, 2006.

[73] Z. Benenson, L. Pimenidis, F. Freiling, and S. Lucks, "Authenticated Query Flooding in Sensor Networks," in *Proceedings of the 4th Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, IEEE, 2006.

[74] P. Ning, A. Liu, and W. Du, "Mitigating DoS attacks against broadcast authentication in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 4, no. 1, 2008. ACM.

[75] I. Martinovic, P. Pichota, and J. Schmitt, "Jamming for good: a fresh approach to authentic communication in WSNs," in *Proceedings of the 2nd ACM conference on Wireless network security*, pp. 161–168, ACM, 2009.

[76] M. Wilhelm, I. Martinovic, and J. Schmitt, "Secret keys from entangled sensor motes: implementation and analysis," in *Proceedings of the 3rd ACM conference on Wireless Network Security*, pp. 139–144, ACM, 2010.

[77] R. Mayrhofer and H. Gellerson, "Shake Well Before Use: Authentication Based on Accelerometer Data," in *Proceedings of the 5th International Conference on Pervasive Computing*, pp. 144–161, ACM, 2007.

[78] A. McDaniel, J. Pitcher, K. Bivek, and J. Barry, "Reactive Diversity Array Wi-Fi Jammer," tech. rep., Georgia Institute of Technology, USA, 2008.

[79] D. Balfanz, D. Smetters, P. Stewart, and H. Wong, "Talking to strangers: Authentication in ad-hoc wireless networks," in *Proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS)*, pp. 7–19, The Internet Society, 2002.

[80] R. Roman and J. Lopez, "KeyLED - Transmitting Sensitive Data Over Out-of-Band Channels in Wireless Sensor Networks," in *Proceedings of the 4th IEEE Workshop on Wireless Sensor Network Security*, IEEE, 2008.

[81] C. Lopes and P. Aguiar, "Aerial acoustic communications," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 219–222, IEEE, 2001.

[82] S. Drimer and S. Murdoch, "Keep your enemies close: Distance bounding against smartcard relay attacks," in *Proceedings of the 16th USENIX Security Symposium*, pp. 87–102, USENIX Association, 2007.

[83] G. Hancke and M. Kuhn, "An RFID Distance Bounding Protocol," in *Proceedings of the 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks*, p. 73, IEEE, 2005.

[84] M. Önen and R. Molva, "Secure data aggregation with multiple encryption," in *Proceedings of the 4th European conference on Wireless Sensor Networks*, pp. 117–132, Springer-Verlag, 2007.

[85] S. Ganeriwal, S. Čapkun, C. Han, and M. Srivastava, "Secure time synchronization service for sensor networks," in *Proceedings of the 4th ACM workshop on Wireless security*, p. 106, ACM, 2005.

[86] J. Deng, R. Han, and S. Mishra, "INSENS: Intrusion-Tolerant Routing in Wireless Sensor Networks," tech. rep., University of Colorado, USA, 2002.

[87] L. Eschenauer and V. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pp. 41–47, ACM, 2002.

[88] T. Vu, R. Safavi-Naini, and C. Williamson, "Securing wireless sensor networks against large-scale node capture attacks," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, pp. 112–123, ACM, 2010.

[89] Q. Mi, J. Stankovic, and R. Stoleru, "Secure Walking GPS: A Secure Localization and Key Distribution Scheme for Wireless Sensor Networks," in *Proceedings of the ACM Conference on Wireless Network Security (WiSec)*, ACM, 2010.

[90] A. Ünlü and A. Levi, "Two-tier, location-aware and highly resilient key predistribution scheme for wireless sensor networks," in *Proceedings of the BCS International Academic Conference on Visions of Computer Science*, pp. 355–366, BCS, 2008.

[91] A. Wacker, M. Knoll, T. Heiber, and K. Rothermel, "A new approach for establishing pairwise keys for securing wireless sensor networks," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pp. 27–38, ACM, 2005.

[92] K. Rasmussen, C. Castelluccia, T. Heydt-Benjamin, and S. Capkun, "Proximity-based access control for implantable medical devices," in *Proceedings of the 16th ACM conference on Computer and communications security*, pp. 410–419, ACM, 2009.

[93] N. Mehallegue, A. Bouridane, and E. Garcia, "Efficient path key establishment for wireless sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2008, pp. 1–9, 2008. Hindawi.

[94] Y. Zhou and Y. Fang, "A Two-Layer Key Establishment Scheme for Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, pp. 1009–1020, 2007. IEEE.

[95] W. Du, J. Deng, and P. Varshney, "A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security*, ACM, 2003.

[96] T. Dierks and C. Allen, "The TLS Protocol Version 1.0." RFC 2246 (Proposed Standard), Jan. 1999. Obsoleted by RFC 4346, updated by RFCs 3546, 5746, 6176. `http://www.ietf.org/rfc/rfc2246.txt`.

[97] W. Diffie and M. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, pp. 644–654, 1976.

[98] I. Chatzigiannakis, E. Konstantinou, V. Liagkou, and P. Spirakis, "Design, Analysis and Performance Evaluation of Group Key Establishment in Wireless Sensor Networks," *Electronic Notes in Theoretical Computer Science*, vol. 171, no. 1, pp. 17–31, 2007. Elsevier.

[99] A. Dmitriev, E. Efremova, A. Kletsov, L. Kuzmin, A. Laktyushkin, and V. Yurkin, "Wireless ultrawideband communications and sensor networks," *Journal of Communications Technology and Electronics*, vol. 53, no. 10, pp. 1206–1216, 2008. Springer.

[100] A. Ahmad, A. Biri, and H. Afifi, "Study of a new physical layer encryption concept," in *Proceedings of the 4th International Workshop on Wireless and Sensor Networks Security*, IEEE, 2008.

[101] A. Srinivasan and J. Wu, "A Survey on Secure Localization in Wireless Sensor Networks," *Encyclopedia of Wireless and Mobile Communications*, 2008. CRC Press.

[102] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: a quantitative comparison," *Computer Networks*, vol. 43, no. 4, pp. 499–518, 2003. Elsevier.

[103] J. Krumm and E. Horvitz, "Locadio: Inferring motion and location from wi-fi signal strengths," in *Proceedings of the International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous 04)*, IEEE, 2004.

[104] K. Muthukrishnan, M. Lijding, N. Meratnia, and P. Havinga, "Sensing motion using spectral and spatial analysis of WLAN RSSI," in *Proceedings of the 2nd European Conference on Smart Sensing and Context (EuroSSC 2007)*, pp. 62–76, Springer-Verlag, 2007.

[105] J. M. Fajardo and A. P. Dominguez, "Distance-Bounding Protocols for RFID," *Security in RFID and Sensor Networks*, pp. 151–169, 2009. Auerbach Publications.

[106] Nanotron Technologies, "Real Time Location Systems (RTLS)," tech. rep., Nanotron Technologies, Germany., 2007. `http://www.nanotron.com/`.

[107] Thales, "Thales: UWB Precise Positioning," 2010. `http://www.thalesresearch.com/`.

[108] A. Springer, W. Gugler, M. Huemer, L. Reindl, C. Ruppel, and R. Weigel, "Spread spectrum communications using chirp signals," *IEEE/AFCEA EUROCOMM 2000. Information Systems for Enhanced Public Safety and Security*, pp. 166–170, 2000.

[109] S. Brands and D. Chaum, "Distance Bounding Protocols," in *Proceedings of Advances in Cryptography 1993 (CRYPTO '93)*, Springer, 1994.

[110] Y. Lin, H. Lee, M. Woh, Y. Harel, S. Mahlke, T. Mudge, and C. Chakrabarti, "SODA: A Low-power Architecture For Software Radio," in *Proceedings of the 33rd International Symposium on Computer Architecture (ISCA'06)-Volume 00*, pp. 89–101, IEEE, 2006.

[111] M. Flury, M. Poturalski, P. Papadimitratos, J. Hubaux, and J. Le Boudec, "Effectiveness of Distance-Decreasing Attacks Against Impulse Radio Ranging," in *Proceedings of the 3rd ACM Conference on Wireless Network Security*, ACM, 2010.

[112] K. Rasmussen and S. Čapkun, "Location Privacy of Distance Bounding protocols," in *Proceedings of the 15th ACM conference on Computer and communications security*, pp. 149–160, ACM, 2008.

[113] E. Miluzzo, X. Zheng, K. Fodor, and A. Campbell, "Radio characterization of 802.15. 4 and its impact on the design of mobile sensor networks," in *Proceedings of the 5th European conference on Wireless sensor networks*, pp. 171–188, Springer-Verlag, 2008.

[114] S. Obayashi and J. Zander, "A Body-Shadowing Model for Indoor Radio Communication Environments," *IEEE Transactions on Antennas and Propagation*, vol. 46, pp. 926–927, 1998.

[115] J. Benson, U. Roedig, T. O'Donovan, and C. Sreenan, "Reliability Control for Aggregation in Wireless Sensor Networks," in *Proceedings of the 2nd IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SENSEAPP2007)*, IEEE, 2007.

[116] G. Lu, B. Krishnamachari, and C. Raghavendra, "An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks," in *Proceedings of the International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks*, IEEE, 2004.

[117] T. O'Donovan, J. Brown, U. Roedig, C. Sreenan, A. Dunkels, A. Klein, S. Silva, V. Vassiliou, and L. Wolf, "GINSENG: Performance Control in Wireless Sensor Networks," in *Proceedings of the 7th Annual IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks (SECON)*, pp. 1–3, IEEE, 2010.

[118] E. Rescorla, "Diffie-Hellman Key Agreement Method." RFC 2631 (Proposed Standard), June 1999. `http://www.ietf.org/rfc/rfc2631.txt`.

[119] R. L. Brown, "SEC 2: Recommended Elliptic Curve Domain Parameters," 2010. Certicom Research, USA. `http://www.certicom.com/`.

[120] A. Bogdanov, D. Khovratovich, and C. Rechberger, "Biclique Cryptanalysis of the Full AES," in *Proceedings of the 17th Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'11)*, Springer-Verlag, 2011.

[121] M. Bellare, J. Kilian, and P. Rogaway, "The Security of the Cipher Block Chaining Message Authentication Code," *Journal of Computer and System Sciences*, vol. 61, no. 3, pp. 362–399, 2000. Elsevier.

[122] BBC, "Age of Universe confirmed," *BBC News Online*, April 2002. `http://news.bbc.co.uk/1/hi/sci/tech/1950403.stm`.

[123] L. Casado and P. Tsigas, "ContikiSec: A Secure Network Layer for Wireless Sensor Networks under the Contiki Operating System," in *Proceedings of the 14th Nordic Conference on Secure IT Systems: Identity and Privacy in the Internet Age*, pp. 133–147, Springer-Verlag, 2009.

[124] A. Liu and P. Ning, "TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks," in *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*, IEEE, 2008.

[125] R. Mayrhofer and H. Gellersen, "On the Security of Ultrasound as Out-of-band Channel," in *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS 2007)*, pp. 1–6, IEEE, 2007.

[126] J. Yang, Y. Chen, and W. Trappe, "Detecting Sybil Attacks in Wireless and Sensor Networks Using Cluster Analysis," in *Proceedings of the 4th IEEE International Workshop on Wireless and Sensor Networks Security*, IEEE, 2008.

[127] H. Schantz, “Near Field Phase Behavior,” in *Proceesings of the IEEE Antennas and Propagation Society International Symposium*, pp. 134–137, IEEE, 2005.

[128] R. Frank, “Current developments in Loran-C,” *Proceedings of the IEEE*, vol. 71, no. 10, pp. 1127–1139, 1983. IEEE.

[129] J. Sallai, A. Lédeczi, I. Amundson, X. Koutsoukos, and M. Maróti, “Using RF received phase for indoor tracking,” in *Proceedings of the 6th Workshop on Hot Topics in Embedded Networked Sensors*, ACM, 2010.

[130] S. Capkun and J. Hubaux, “Secure positioning of wireless devices with application to sensor networks,” in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM2005)*, vol. 3, IEEE, 2005.

[131] Nanotron, *nanoLoc TRX Transceiver (NA5TR1) Datasheet*, 2008. Nanotron Technologies, Germany. `http://www.nanotron.com/`.

[132] Sentilla, *Tmote Sky Low Power Wireless Sensor Module Datasheet*, 2006. Sentilla Corporation, USA. Formerly Moteiv Corporation, USA. `http://www.sentilla.com/`.

[133] P. Suriyachai, J. Brown, and U. Roedig, “Time-Critical Data Delivery in Wireless Sensor Networks,” in *Proceedings of the 6th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS10)*, IEEE, 2010.

[134] J. Xiong and K. Jamieson, “SecureAngle: Improving Wireless Security Using Angle-of-Arrival Information,” in *Proceedings of the 9th ACM Workshop on Hot Topics in Networks*, ACM, 2010.

[135] K. Xu, M. Gerla, and S. Bae, “How Effective is the IEEE 802.11 RTS/CTS Handshake in Ad Hoc Networks?,” in *Proceedings of the IEEE Global Communications Conference 2002*, IEEE, 2002.

# Index