GDAŃSK UNIVERSITY OF TECHNOLOGY

FACULTY OF ELECTRONICS, TELECOMMUNICATIONS AND INFORMATICS

The author of the PhD dissertation: Alan Turower
Scientific discipline: Informatics

DOCTORAL DISSERTATION

Title of PhD dissertation: Trust Management Method for Wireless Sensor Networks

Title of PhD dissertation (in Polish): Metoda zarządzania zaufaniem w bezprzewodowych sieciach czujników

Supervisor *signature*	Second supervisor *signature*
prof. dr hab. inż. Janusz Górski	
Auxiliary supervisor *signature*	Cosupervisor *signature*

Gdańsk, 2016

Acknowledgment

During the years leading to write this thesis many people influenced me and my dissertation. I would like to express thanks for the received help and support.

The supervisor of my Master thesis, Dr. Andrzej Wardziński, proposed me to continue my studies and gave me first lessons in writing papers. I am very grateful for his support in the University and advices. He also introduced me to trust management subject and to Professor Janusz Górski.

Professor Janusz Górski gave me opportunity to write this thesis and taught me how to do research and how to work on papers. I thank him for countless comments and constructive critique on drafts of this dissertation that greatly improved its quality.

I also thank many scientists, who I had the honour to meet at conferences and discuss my work.

I am also grateful to Anetta Gorloff-Żukowska, who I could always ask for advice in academic procedures.

Finally, I would like to give special thanks to my Family, who gave me constant support and believed in me for all these years.

Abstract

A Wireless Sensor Network (WSN) is a network of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, pressure, etc. and to cooperatively pass their data to the main location.

The first wireless network that bore any real resemblance to a modern WSN is the Sound Surveillance System (SOSUS), developed by the United States Military in the 1950s to detect and track Soviet submarines. Currently, WSN are viewed as one of the most important technologies for the 21st century [1]. European Union supports programmes connected with WSN utilization and China have involved WSNs in their national strategic research programmes [2]. The commercialization of WSNs are also being accelerated by companies [3].

As the WSN are part of variety complex systems, it become important to ensure security of these networks. Copying the best practices from the conventional networks is not practical as sensor nodes are subjected to severe limitations of their resources and cannot afford running sophisticated security mechanisms which are often significantly resource consuming.

To cope with this problem, the concepts of trust and trustworthiness are employed. Trust management provides for distinguishing between trustworthy and untrustworthy nodes which enables collaborative decisions leading to isolation and exclusion of the nodes with a very low level of trust. It allows to improve the security of the network using fewer resources comparing to security mechanisms used in conventional networks.

It this dissertation a new trust management method for distributed wireless sensor networks called WCT2M is presented and its performance analysed. It is explained how WCT2M works in the network applying the fully synchronized sleep scheduling pattern. Such networks were subjected to the analyses with the help of a specially created laboratory and a dedicated WCT2M simulator. The results of conducted experiments allow to ascertain, that the proposed method reliably and efficiently recognizes untrusted nodes and prevent information from these nodes to spread in the network, using reasonable amount of resources.

Table of Contents

List of abbreviations

ACK	Acknowledgment packet
AODV	Ad hoc On-demand Distance Vector
ARPANET	Advanced Research Projects Agency Network
CH	Cluster Head
DADS	Detect And Defend Spam
DARPA	Defense Advanced Research Projects Agency
DAS	Defend Against Spam
DSN	Distributed Sensor Networks
FA	Fault Announcement
FCFS	First-Come-First-Served
FIFO	First-In-First-Out
FFD	Full Function Device
GTS	Guaranteed-Time Slot
IEEE	Institute of Electrical and Electronics Engineers
LEACH	Low-Energy Adaptive Clustering Hierarchy
MAC	Media Access Control
MANET	Mobile Ad-hoc Networks
OSI	Open Systems Interconnection
PAN	Personal Area Network
RFD	Reduced Function Device
S-MAC	Sensor MAC
SOSUS	Sound Surveillance System
T-MAC	Timeout MAC
TDMA	Time Division Multiple Access
UV	Ultraviolet
WCT2M	WSN Cooperative Trust Management Method
WCTMS	WSN Cooperative Trust Management Simulator
WSN	Wireless Sensor Network

List of figures

List of tables

1. Introduction

1.1 Context

A Wireless Sensor Network (WSN) is a network of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, pressure, etc. and to cooperatively pass their data to the main location. More advanced networks are bi-directional, also enabling control of sensor activity. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance. Recently such networks are used in various industrial and consumer applications, such as industrial process monitoring and control, patient monitoring, and so on [4]. It is also a broad research topic with applications in many sectors, like industry, home computing, agriculture, environment, and others, based on the adoption of fundamental principles, specification characterisations, modelling, simulations and state-of-the-art technology [5].

As sensor networks become more complex and provide more sophisticated services, the diversity of roles of network nodes increases, in addition to simple sensor nodes including: routers, heads and base stations. Such nodes, which mission is broader than just sensing the environment, can have considerable computing power and accomplish advanced tasks. As the size and complexity of the networks grows, managing them becomes more difficult to reconcile security with performance and flexibility. Moreover, individual nodes or sub-networks can be managed by different persons or organizations. Dependability of such networks becomes a difficult issue as in addition to technical imperfections and human faults, malicious actions have to be taken into account.

1.2 Problem statement and thesis proposition

The number of implemented usages of wireless sensor networks is growing. Now it is not only scientific demonstrations, but also large industrial networks. The applications range widely from military surveillance to civilian applications such as health monitoring [6]. First sensor systems were implemented as one device, which delivered a measured value from the monitoring object to the receiver object without wire connection. The progress in creating sensors resulted in new networked embedded systems - the wireless sensor networks collect data and deliver to the management control systems. These systems can be used to measure, process and communicate with each sensor node over the wireless communication. WSNs can support intelligent operations and work in smart environments [7].

These complex systems are often build without any security provisions or with multiple security leakages [8] [9]. Moreover, many solutions proposed for wired networks are improper for wireless

sensor networks or cannot be directly applied [8]. Ensuring security of sensor networks by just copying the best practices from the conventional networks is not practical as sensor nodes are subjected to severe limitations of their resources and cannot afford running sophisticated security mechanisms which are often significantly resource consuming.

To cope with this problem, the concepts of trust and trustworthiness are employed. Object A *trusts* object B if A makes some (positive) assumptions about the state and behaviour of B (for instance, A assumes that the data sent by B is genuine). B is *trustworthy* if A has in its disposal the evidence sufficient to justify its trust in B. With these definitions, *trust management* is understood as collecting the evidence about trustworthiness and based on this, making decisions about trust.

In sensor networks, trust management has a great importance as it provides for distinguishing between trustworthy and untrustworthy nodes which enables collaborative decisions leading to isolation and exclusion of the nodes with a very low level of trust. It allows to improve the security of the network using fewer resources comparing to security mechanisms used in conventional networks.

The objective of this work is to develop a new trust management model for distributed wireless sensor networks (WSN) and to analyse its performance. The proposed method should recognize untrusted nodes and prevent information from these nodes to spread in the network.

The thesis proposition is formulated as follows:

> ***The proposed method allows to effectively and efficiently manage trust in wireless sensor networks.***

1.3 Importance and the relevance of the problem

Wireless sensor networks are getting sophisticated enough to become important in numerous application areas, including healthcare, defence, safety monitoring, environment monitoring and others. Although these networks share a number of security requirements with traditional networks, due to their specific limitations they may need different solutions. First of all, they have limited resources such as memory, power supply and computational ability. Resource limitations result from the need of low cost devices. This restricts application of conventional protection mechanisms (like advanced crypto schemes) and excludes complex security solutions [10].

Moreover, the protocols used in WSNs are built without devoting much attention to security problems. *ZigBee*, one of the most popular specifications for communication protocols, may serve as

an example. ZigBee has been developed by The ZigBee Alliance, which is an association of companies working together to develop standards (and products) for reliable, cost-effective, low-power wireless networking. ZigBee is embedded in a wide range of products and applications across consumer, commercial, industrial and government markets worldwide. ZigBee builds upon the IEEE 802.15.4 standard [11] which defines the Physical and Media Access Control (MAC) layers for low cost, low rate personal area networks. ZigBee specification provides a security service that allows to protect transmitted data, however it does not solve many security flaws:

- Applications on one ZigBee node are not separated and each application can have access to the whole node and call any procedure. Lower ZigBee layers are fully accessible for all applications installed on a given node. The possibility that network node software will be corrupted and an unauthorized application will be able to send messages using trusted security keys due to *open trust model* implemented on ZigBee nodes [12] cannot be excluded.
- Some data (e.g. medical data) can be incorrect due to measurement error or device failure and such erroneous data should be identified as close to its source as it is possible, before it spreads over other nodes and locations. If such data is detected, then further data from its source should be discarded unless the cause is identified and removed. This issue is especially important for medical and personal data.
- Various threats may result from incompatible hardware and software from different vendors, incorrect network configuration changes, software updates or users' mistakes.

It is possible to prevent these flaws by implementing more advanced security schemes, protection mechanisms and error detection mechanisms and building nodes with more precise and reliable sensors and antennas. However, this requires more expensive and physically bigger hardware what restricts the usage of WSNs. Another solution is to use trust management approach. It allows to increase the security level of the network by minimizing issues present in WSN protocols having limited resources in consideration.

1.4 Research approach

The research approach adopted in this report is as follows.

First, a thorough review of the already proposed methods for trust management in wireless sensor networks is made.

In the next stage an inventory and analysis of threats and possible attacks in wireless sensor networks is made together with the classification of these attacks. It includes spam attack, black hole attack and message modification attack.

Then a new method for trust management in multi-layer wireless sensor networks composed of clusters, called *WSN Cooperative Trust Management Method* (WCT2M), is proposed. The method is based on a distributed trust management model in clustered networks because it allows to limit performance problems connected with network expansion.

To assess the proposed WCT2M method, the following approach was used.

- A case study was used as a reference to explain WCT2M and to provide a comprehensive example of its application. This case study has been derived from the Advanced Network embedded platform as a Gateway to Enhanced quality of Life (ANGEL) project [13]. ANGEL was a STReP project performed in years 2006-2009 within the 6. European Framework Programme.
- Criteria for assessing the effectiveness of the WCT2M are formulated and metrics for assessing these criteria are proposed.
- A laboratory WSN system was developed to demonstrate feasibility of WCT2M. The system is based on ZigBee nodes. To deploy WCT2M on these nodes, a dedicated software in C language was developed with the help of Eclipse IDE [14] and mspgcc compilator [15].

  The laboratory system implements a small WSN with structure derived from ANGEL case study. WCT2M is deployed on each node of the system. The system allows to demonstrate the feasibility of WCT2M and was used to evaluate the method effectiveness for different attack scenarios.
- Because of the scale limitations of the laboratory system, to assess WCT2M performance in a larger scale, simulations were used. After reviewing existing simulators (including the assessment of their availability and cost), it was decided to develop a dedicated simulator for assessment of WCT2M. It was written in NetBeans IDE [16], using Java 1.7 [17] with JGraph library [18].
- The dedicated WCT2M simulator was used to analyse WCT2M performance for larger networks and for different threat scenarios.

The results of the laboratory and simulation experiments were used to justify the thesis proposition.

1.5 Dissertation outline

The dissertation is organized as follows:

- Section 2 explains, what are Wireless Sensor Networks (WSN) and the basic concepts used in this work. It also introduces the concept of sleep scheduling.
- Section 3 describes the case study used in this work to show the rules of trust management. The example network and exemplary threats to that network are presented.
- Section 4 focuses on WSN security. A survey of known attacks with attack scenarios is presented and known methods of protecting networks against attacks are described.
- Section 5 introduces the concept of trust and trust management. WCT2M is presented and its features discussed.
- Section 6 presents existing WSN simulators and describes analytic tools used to assess WCT2M. WCTMS simulator and specially created laboratory are described.
- Section 7 presents and discusses the results achieved while assessing WCT2M using previously described tools.
- Section 8 summarizes the related works and compares the results presented in the previous section with the results achieved using other related methods.
- Section 9 summarizes the results and suggests directions for further research.

2. Wireless Sensor Networks

2.1 History of WSN

Wireless sensor networks are quickly gaining popularity due to the fact that they are potentially low cost solutions to a variety of real-world challenges [6]. The first wireless network that bore any real resemblance to a modern WSN is the Sound Surveillance System (SOSUS), developed by the United States Military in the 1950s to detect and track Soviet submarines. This network used submerged acoustic sensors – hydrophones – distributed in the Atlantic and Pacific oceans. This sensing technology is still in service today, albeit serving more peaceful functions of monitoring undersea wildlife and volcanic activity [19].

The next milestone in WSN research can be traced back to investments made in the 1960s and 1970s to develop hardware for today's Internet. As a result, at around 1980 the Distributed Sensor Networks (DSN) program at the Defense Advanced Research Projects Agency (DARPA) was started. By this time, the ARPANET (Advanced Research Projects Agency Network) had been operational for a number of years, with about 200 hosts at universities and research institutes [20].

Governments and universities began using WSNs in applications such as air quality monitoring, forest fire detection, natural disaster prevention, weather stations and structural monitoring. When the engineering students graduated and were employed by technology giants, such as IBM and Bell Labs, they began promoting the use of WSNs in heavy industrial applications such as power distribution, waste-water treatment and specialized factory automation [19].

Even though early researchers on sensor networks had in mind the vision of a DSN, the technology was not quite ready. The sensors were rather large (i.e. shoe box and up) which limited the number of potential applications. Further, the earliest DSNs were not tightly associated with wireless connectivity [3].

Recent advances in computing, communication and micro-electromechanical technology, starting from 1998, have caused a significant shift in WSN research. Research focused on networking techniques and networked information processing suitable for highly dynamic ad hoc environments and resource-constrained sensor nodes. Further, the sensor nodes have been much smaller in size (e.g. pack of cards to dust particle) and much cheaper in price, and thus many new civilian applications of sensor networks such as environment monitoring, vehicular sensor network and body sensor network have emerged [3]. Then DARPA launched an initiative research program called SensIT which provided the present sensor networks with new capabilities such as ad hoc networking,

dynamic querying and tasking, reprogramming and multi-tasking [21]. At the same time, the IEEE noticed the low expense and high capabilities that sensor networks offer. The organization has defined the IEEE 802.15.4 standard for low data rate wireless personal area networks. Based on IEEE 802.15.4, ZigBee Alliance has published the ZigBee standard which specifies a suite of high level communication protocols which can be used by WSNs [3].

Currently, WSN are viewed as one of the most important technologies for the 21st century [1]. European Union supports programmes connected with WSN utilization. An example can be the ANGEL project which aims at providing methods and tools for building complex heterogeneous systems in which WNS and traditional communication networks cooperate to monitor and improve the quality of life in common habitats [22]. Countries such as China have involved WSNs in their national strategic research programmes [2]. The commercialization of WSNs are also being accelerated by companies [3].

2.2 Distinguishing features of WSN

A wireless sensor network is a collection of nodes organized into a cooperative network. Each node consists of processing capability (one or more microcontrollers, CPUs or Digital Signal Processing chips), may contain multiple types of memory (program, data and flash memories), have a RF transceiver, have a power source (e.g., batteries and solar cells), and accommodate various sensors. The nodes communicate wirelessly and often self-organize after being deployed in an ad hoc fashion [23].

The *ad hoc network* is a collection of wireless nodes which can rapidly set up a network when needed [24]. A sensor network is also considered an ad hoc network in which nodes are extended with sensing capability.

Design of sensor networks differs from other ad hoc networks in some aspects [8].

Firstly, sensor node is usually a small device with a low-speed processor, limited memory and a short-range transceiver. For example, Berkley Mica Motes [25] are tiny sensor nodes largely used in practical WSN experiments. Figure 1 shows the board of the Mica2 dot Mote near a two Euro coin for size comparisons purposes, but there can be even smaller devices, called ‘smart dust’ due to their size.

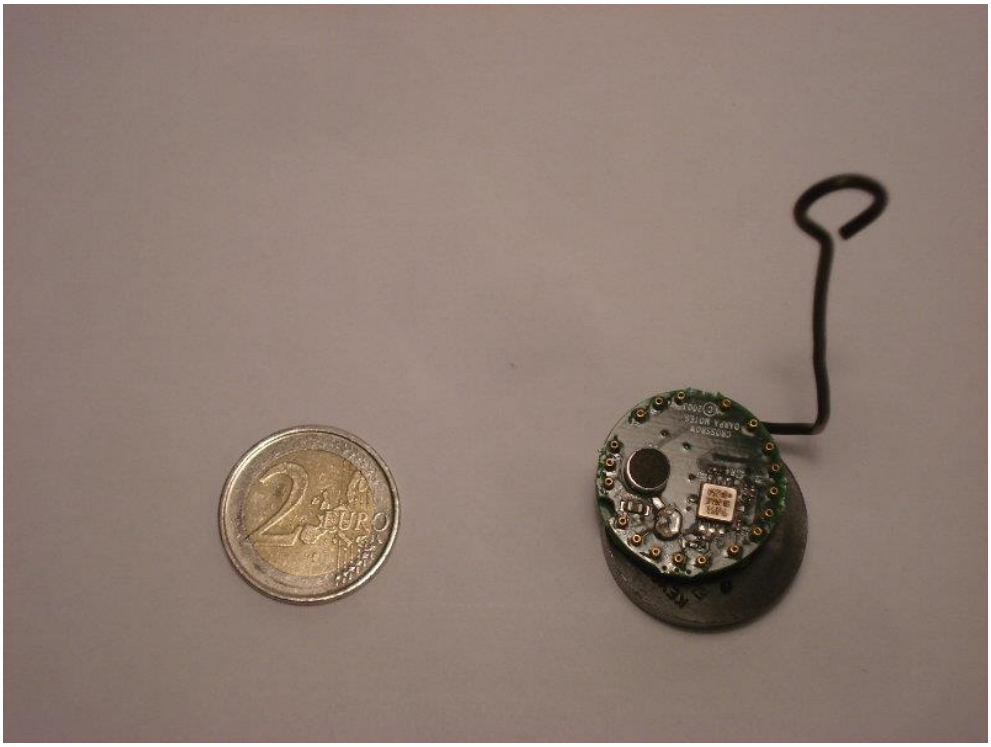

Figure 1 Mica2 dot Mote board near 2 Euro coin [26]

According to these limitations, protocols and algorithms for WSN need to be simple.

Secondly, energy consumption is a critical issue while designing sensor networks, because the nodes are usually powered by batteries, which also need to be small. Moreover, the communication patterns in sensor networks may differ from those used in other ad hoc networks.

The concept of wireless sensor networks is based on a simple equation:

`Sensing + CPU + Radio = Thousands of potential applications` [27].

Unlike traditional wireless devices, wireless sensor nodes do not need to communicate directly with the nearest high-power control tower or base station, but only with their local peers. Instead of relying on a pre-deployed infrastructure, each individual sensor becomes part of the infrastructure. Peer-to-peer networking protocols provide a mesh-like connections to send data between the thousands of tiny devices in a multi-hop fashion. The flexible mesh architectures dynamically adapt to support introduction of new nodes or expand to cover a larger geographic region. Additionally, the system can automatically adapt to compensate for node failures [27].

2.3 Network organization

2.3.1 Nodes and topology

Protocol IEEE 802.15.4 defines three types of nodes occurring in Wireless Sensor Networks:

- *Personal Area Network (PAN)*[1] *coordinator* - the main network controller, responsible for the nodes identification. It provides global coordination services between other nodes in the network by transmitting *beacon frames* containing the identity of the PAN and other relevant information. It can be also described as *base station*,
- *coordinator* – provides local synchronizing services for a part of its PAN (the same functions as the PAN coordinator, but restricted to a part of PAN).
- *slave* – simple node without any functions of coordination. It is connected to PAN coordinator or to a coordinator to synchronize with the other nodes in the network.

The first two types must implement all of the functionality defined by the IEEE 802.15.4 protocol to ensure synchronization and network management. Such devices are defined as *Full Function Device* – FFD. Slave nodes can have a minimal implementation of the IEEE 802.15.4 standard and are called *Reduced Function Device* – RFD. Of course FFD devices can also act as RFD devices.

Each PAN must have at least one FFD type device, acting as PAN coordinator, ensuring that the network functions properly. When the network starts its operations, the PAN coordinator sends beacon frames, connects and disconnects other nodes, provides synchronization services and ensure that the *Guaranteed-Time Slot*[2] (GTS) mechanism functions properly.

IEEE 802.15.4 specification defines two basic topologies of PANs (presented in Figure 2):

- *star* – one node works as PAN coordinator and all other nodes are connected to it. If a node wants to send data to another node it must send it through the central node. Consequently, the PAN coordinator needs more energy than other nodes and it is recommended for that node to be connected to a permanent power supply.
- *peer-to-peer* – this topology has also one PAN coordinator, but communication is decentralized and creates a *mesh network* – a network in which each node has the right to communicate and exchange data with any other node in its range.

[1] A personal area network (PAN) is a network used for data transmission among personal devices such as computers, telephones, personal digital assistants and body sensors.

[2] Mechanism used instead of CSMA/CA, dedicated for reservation of bandwidth of the channel. In a beacon-enabled network topology, the PAN coordinator reserves and assigns the GTS to devices on a first-come-first-served (FCFS) basis in response to requests from devices [121] [122].

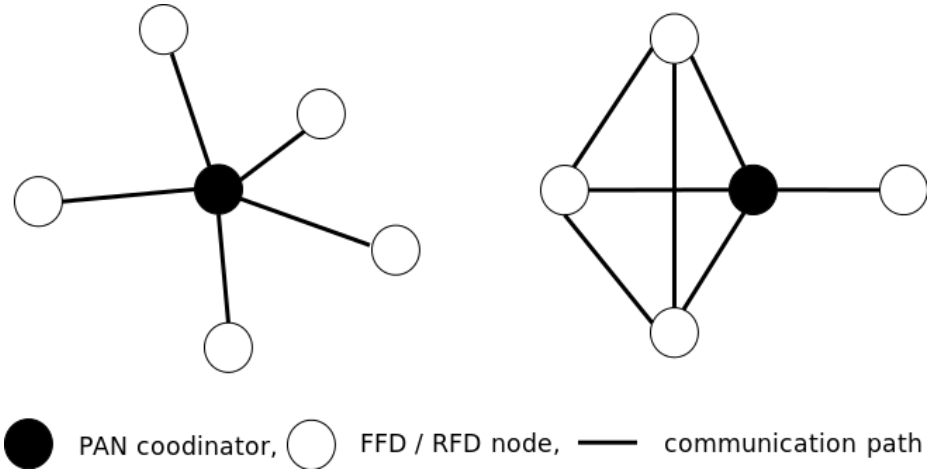

Figure 2 Topologies in IEEE 802.15.4

A special case (not defined by the standard, but given as a possibility) is a cluster tree. It happens when PAN coordinator designates some of the other nodes as coordinators and forms clusters. PAN coordinator forms the first cluster by making itself its head (called *Cluster Head* – CH) and then assigns new *PAN identifiers* (PAN ID) to the chosen coordinators which creates next clusters. For larger physical environments, the cluster tree is a good way to aggregate multiple basic star networks into one larger network [28].

The *network distance* is the shortest path from one node to another. The *level* of the node is its network distance (the minimal number of hops) to the base station. The *height* of the network is the maximal network distance (the maximal level) from a node to the base station in the network. For example, both networks presented in Figure 2 have the height equal 1.

2.3.2 Division into clusters

Cluster is a group of nodes created for more efficient network functioning. Each cluster has the cluster head – a coordinator node selected in a way determined by the clustering protocol. Cluster head aggregates data from nodes of the cluster and communicates with other cluster heads. Clustering results in dividing the network into *tiers*. A tier of a network is a set of its nodes which cluster heads belong to the same, higher level, tier. The highest tier is created by nodes that communicate with the base station as their cluster head. The simplest case is a one tier network (all network nodes form a single cluster). In a more complex situation network has two tiers: the tier of cluster heads and the tier of cluster nodes. Network can have even more tiers, if the heads of the lowest tier are aggregated into clusters to form a middle tier.

Example 1

Example distributions of one-tier and two-tier networks are shown in Figure 3. Solid lines in Figure 3 b) present division into 16 clusters. Cluster heads are marked as empty circles and the base station is empty circle outside square with the nodes. Solid arrows present routes of messages from nodes to the base station sent in the lower tier and dotted arrows present routes of messages from nodes to the base station sent in the higher tier. It can be noticed, that some nodes are connected in different way after dividing the network into clusters.

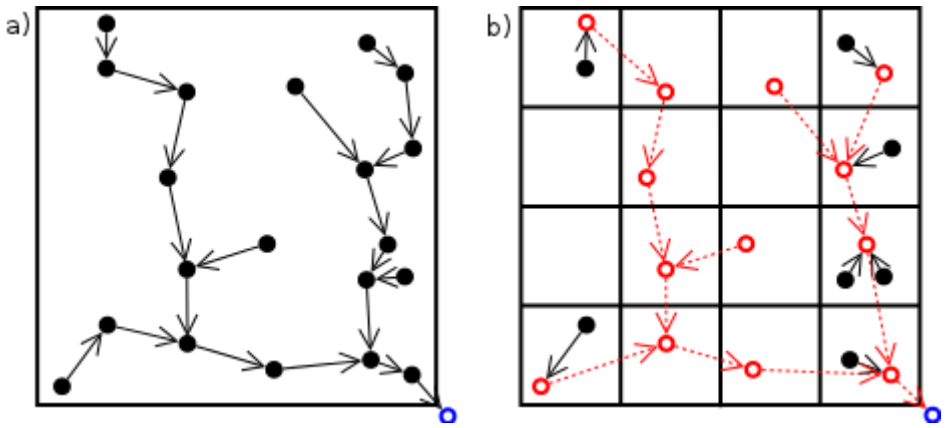

Figure 3 Example distribution of one-tier (a) and two-tier (b) networks, 20 nodes each.

Cluster heads can transmit data in longer distances than just to the members of the own cluster, because they have to communicate with other heads. This results in greater demand for energy. Therefore, the cluster protocols periodically change the cluster head or even recluster the network to distribute the load uniformly over all nodes in the network [29].

The most popular clustering protocols in WSN are based on LEACH (Low-Energy Adaptive Clustering Hierarchy) protocol, proposed by Heinzelman et al. [30]. LEACH is a clustering-based protocol that utilizes randomized rotation of local cluster base stations (cluster-heads) to evenly distribute the energy load among the sensors in the network. LEACH uses localized coordination to enable network scalability and robustness of dynamic networks, and incorporates data fusion into the routing protocol to reduce the amount of information that must be transmitted to the base station.

LEACH defines two tiers of nodes:

- Nodes in the lower tier are grouped in clusters. In each cluster, one node acts as the head of the cluster (this role can be exchanged while the network configuration change or power supply drains). The nodes communicate only with their neighbours which belong to the same cluster.

- Each cluster head is a member of the higher tier. Members of this tier act as routers of information to the network base station – which is the head of the higher tier. The base station is a machine plugged to the power network and it has high computing abilities. It is assumed that the base station remains trustworthy as long as it is available.

Figure 3 b) is an example of LEACH network.

Handy et al. [31] propose to modify the LEACH stochastic cluster-head selection algorithm by a deterministic component. Their simulation results show that in such situation an increase of network lifetime by about 30 % can be accomplished, depending on the network configuration.

Clustering reduces channel contention and packet collisions, resulting in better network throughput under high load. Moreover, clustering has been shown to improve the whole network lifetime, it provides network scalability, resource sharing and efficient use of constrained resources that gives network topology stability and energy saving attributes [29] [32]. For example, some WSN applications require only an aggregated value to be reported to the observer. In this case, sensors in different regions of the field can collaborate to aggregate their data and provide more accurate reports about their local regions and save resources by transferring smaller amounts of data [29]. For example, an average humidity for a region is important in habitat monitoring, not the list of all observed values [33].

2.4 Sleep scheduling

Most of existing *contention-based*[3] WSN Media Access Control (MAC) protocols reduce idle listening which is one of the most common sources of energy loss in WSN [34]. *Time Division Multiple Access*[4] (TDMA) reservation based protocols distinguish fixed time interval for nodes to communicate. Splitting node activity into communication and idle intervals reduces the channel contention and energy costs [35]. The periodic time sequence of fixed number of communication preceded with idle intervals of fixed length is called *sleep scheduling cycle*. The communication interval is called *wakeup period* and idle period is called *sleep period*.

[3] Contention is a media access method that is used to share a broadcast medium. In contention, any device in the network can transmit data at any time (first come-first served). The major drawback of these protocols is low throughput, which is caused by packet collision. When more than one user sends packets at the same time, their packets will collide and none can be correctly received [123].

[4] TDMA allows several devices to share the same frequency channel by dividing the signal into different time slots. The devices transmit in rapid succession, one after the other, each using its own time slot.

Sensor MAC (S-MAC) [36] and Timeout MAC (T-MAC) [37] are contention-based protocols focused on reducing idle radio listening by concentrating the network's data transmissions into a smaller active period and then transitioning to sleep for the remainder of the sleep scheduling cycle. An energy efficient and low latency DMAC [38] is an efficient data gathering protocol for sensor networks where the communication pattern is restricted to a unidirectional tree.

In the ZigBee stack, there is a fixed wakeup/sleep scheduling method: in each sleep scheduling cycle, the nodes wake up twice, firstly to receive packets from their children and secondly, to transmit to their parents / children in a ZigBee beacon-enabled tree network [39].

All these protocols allow to use *sleep scheduling* (*wakeup patterns*). This method provides effective communication and power usage minimization. Researchers in ad-hoc and sensor networks continue to search for new wakeup patterns to save power without suffering the large latency penalties associated with the wakeup process [40]. Current methods can be divided into two main categories:

- *Scheduled wakeups*: in this class, the nodes follow deterministic (or possibly random) wakeup patterns. Time synchronization among the nodes in the network is assumed. However, asynchronous wakeup mechanisms which do not require synchronization among the different nodes are also categorized in this class. Although asynchronous methods are simpler to implement, they are not as efficient as synchronous schemes, and in the worst case their guaranteed delay can be very long [41].
- *Wakeup on-demand* (*out-of-band wakeup*): It is assumed that a node can be signalled and awakened at any point of time and then a message is sent to the node. This is usually implemented by employing two wireless interfaces (radio receivers). The first radio is used for data communication and is triggered by the second ultra-low-power (or possibly passive) radio which is used only for paging and signalling [41]. Although these methods can be optimal in terms of both delay and energy, they are not yet practical. The cost issues and weak selection of available hardware result in limited range and poor reliability. Stringent system requirements prohibit the widespread use and design of such wakeup techniques [40].

To use scheduled wakeups, a time synchronization protocol is necessary to ensure time synchronization between all nodes. It is essential that the nodes are able to wake up at the same time to be able to exchange information [42].

Kumar et al. [43] presented a comprehensive survey of scheduling algorithms for TDMA protocol. They conclude that there is no a single protocol accepted as a standard. One of the reasons for this is

that the MAC protocol choice is, in general, application dependent, which means that there is no single standard MAC for sensor networks. Another reason is the lack of standardization at lower layers (physical layer) and the (physical) sensor hardware.

Keshevarzian et al. [40] presented the full survey concerning sleep scheduling patterns. The patterns differ in shift between wakeups. It affects the delays or power consumption in the network, depending on the network destination and type of traffic.

In the next section the main sleep scheduling patterns described in Keshevarzian et al. [40] are reviewed.

2.4.1 Fully Synchronized Pattern

The fully synchronized pattern is illustrated in Figure 4. It is related to a network of the height = 3. The sleep period is denoted T, the wakeup period is denoted A. The arrows present the fastest possible scenario of sending a message from a node on the level 3 to the base station.

All nodes of the network wake up at the same time moment defined by a simple periodic pattern. Every wakeup lasts for a fixed period A, and the next wakeup occurs after every fixed period T. Nodes process messages only during A period.

When node from level n sends a message to the base station, this message needs (n – 1) (T + A) periods to reach the base station. Sending a message from the base station to n-th level node takes the same amount of time.

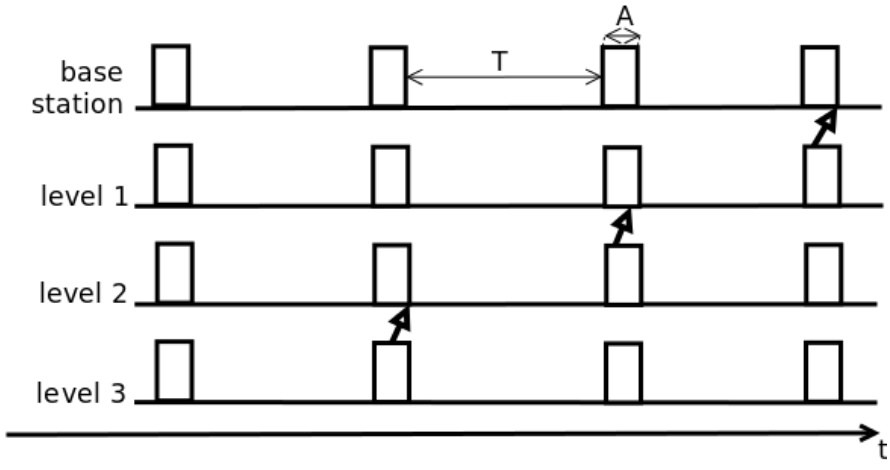

Figure 4 Fully Synchronized pattern

2.4.2 Shifted Even and Odd Pattern

This pattern is derived from the previous one by shifting the pattern for the even level nodes by (T + A) / 2 (Figure 5). Sending a message takes half of the time in comparison to the Fully Synchronized

pattern and is the same for both directions. During T period nodes can wake up to receive a message, but the message is waiting for the nearest A period to be processed. These periods are denoted in dotted grey lines.

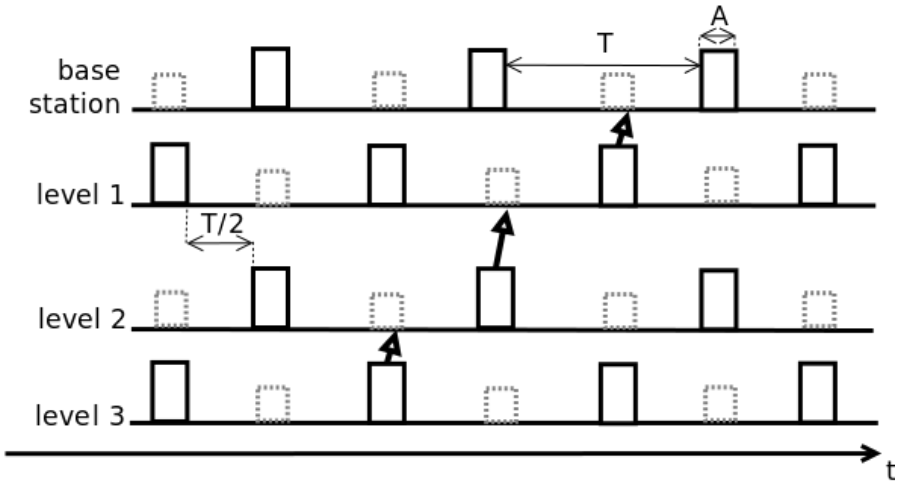

Figure 5 Shifted Even and Odd pattern

2.4.3 Ladder Pattern

At each level, the nodes follow the periodic pattern derived from the previous cases but the wakeup patterns of different levels are shifted in forward (Figure 6) or backward direction (Figure 7). Lu at. al. [44] explains it as the idea similar to the common practice of synchronizing the traffic lights to turn green (wake up) just in time for the arrival of vehicle (packet) from the previous intersection (level). For example, in forward ladder pattern, sending a message in one direction is 2 times faster in comparison to the fully synchronized pattern, but sending a message in a reverse direction is 2 times slower. Moreover this scheme requires two wake-ups of middle-level nodes (L_1,..., L_{h-1}), where *h* is the height of the network, during their T period to ensure all messages are received.

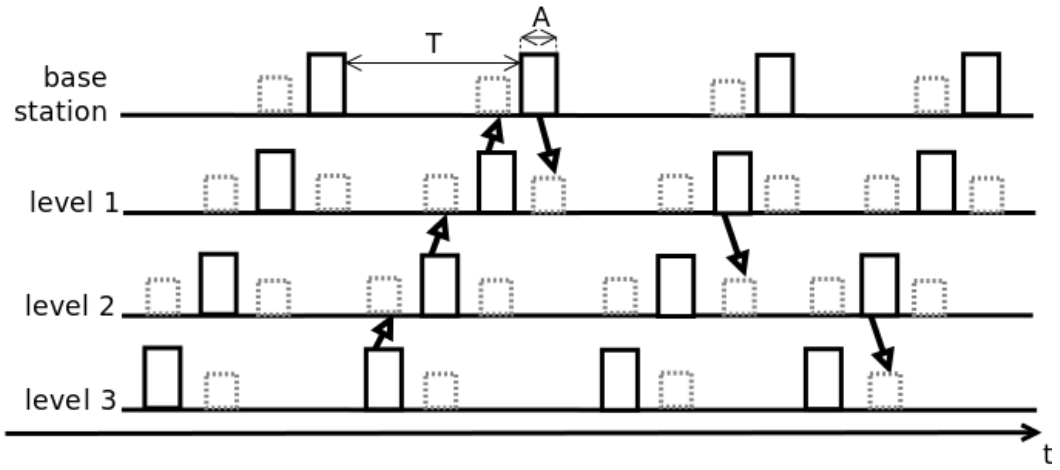

Figure 6 Ladder pattern

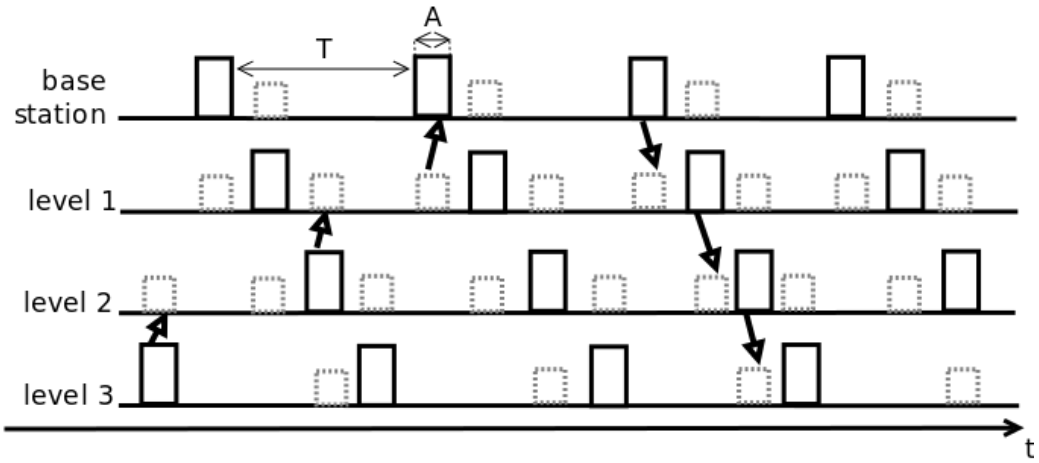

Figure 7 Backward ladder pattern

2.4.4 Two-Ladders Pattern

The previous pattern can be improved by minimizing the delay in both directions by combining a forward ladder with a backward ladder (Figure 8). This idea means that nodes in the middle levels (L_1,..., L_{h-1}), where *h* is the height of the network, wake up twice in every period T, so the effective wakeup period is T_{eff} = T / 2. This pattern allows to forward messages from any level of the network to the base station and in reverse direction in (T + A) / 2 period. However, it is less energy efficient comparing to the previous patterns, because nodes need to wake up relatively more often. Ladders can be also crossed with each other, in different levels (*Crossed-Ladders pattern*).

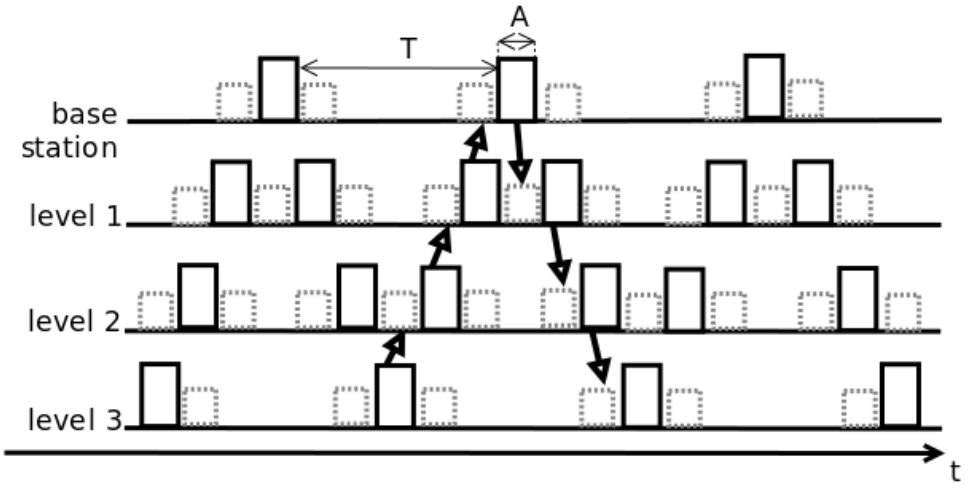

Figure 8 Two-Ladders pattern

3. Case study

Health care is an important area of WSN application. There are many benefits that can be achieved, such as freeing the patient from uncomfortable wires what make him/her feel better or treating less sick patients at their homes, thus reducing the cost of medical care and causing less stress to patients. But it is also the area where many sensitive data are processed, so there is great need for adequate security mechanisms.

3.1 ANGEL scenario

Example use of WSN in the patient's home was presented in the demonstration scenario of the ANGEL project [45]. Illustrative representation of this scenario is shown in Figure 9.

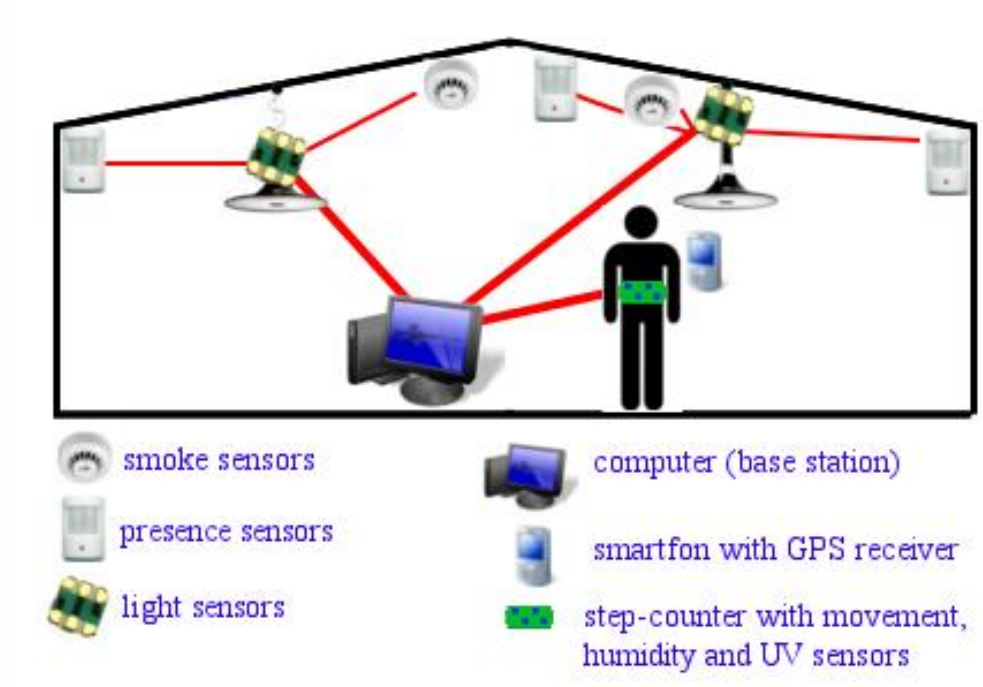

Figure 9 Patient in home environment

Angel platform is one of the results of ANGEL project [13]. It is a set of methods and tools for monitoring and improving the quality of life in common habitats, e.g., home, car and city environment. In particular, it is addressed for the maintenance of the personal health potentiality.

The scenario is as follows [45].

> *Bob has bought the Angel platform, and owns a TV set and a mobile phone, both acting as an Angel compliant Gateway[5]. Bob wants to keep his ideal weight, but the results achieved so far*

[5] Angel Gateways are devices authorized to access data collected by Angel Platform.

with different diets where not completely satisfactory, because anyway he has a tendency to gain weight during autumn, while during spring he cannot fully recover his shape. So every year he accumulates some kilos that he is not able to lose.

When Bob went to a healthcare professional to have assistance on his diet he has been told that he could have been easily supported by the Angel Platform. Through this platform, indeed, he can benefit of a Light Therapy service and of a Training Monitoring service. To follow the Light Therapy, Bob buys some wireless light sensors and special wireless lamps, all compliant with the ZigBee Home Automation Profile [12]*. Bob needs just to place the devices in his house wherever he prefers.*

Now Bob is ready to use the platform. In particular, during winter and autumn, at wakeup moment the lights of the house, in the room that Bob occupies, are set to simulate the dawn with the time and speed of the summer season. If Bob changes the room during the simulated dawn the light follows him in the different rooms of the house.

To be supported in doing his physical exercises, Bob bought also a special step-counter, which embeds also a movement, humidity and UV (Ultraviolet) sensors, able to detect whether Bob is exercising indoor or outdoor. Considering in fact that Bob wears his step-counter all day long, not all the activities measured by the Step-Counter are real exercises, some of them are just steps done during the daily life (i.e. from one room to another, stairs etc.).

Bob also bought and connected to the system smoke and presence sensors, so he feels more safe when he is not at home.

After the initial configuration, Bob can live his normal life, bringing always with him his step-counter. If Bob wants to, he can connect to a web site and design a training plan that he is willing to follow, or he can eventually ask to a professional to fill it for him (i.e. the personal trainer, the nutritionist, the doctor etc.). The platform can send some messages to Bob to remind him to follow his exercises, as well as some messages of positive or negative feedbacks about the compliance with the decided plan.

The step-counter is worn by Bob during the whole day and always sends the training data to the base station, transparently to the user. If during an outdoor training session a heart attack is identified, using temperature, movement and pulse sensors in step-counter, the platform will immediately notify the emergency and inform about the whereabouts of Bob using data from the GPS receiver in Bob's smartphone.

3.2 The example network

Figure 10 presents a network corresponding to the case study presented in Figure 9. BS is the base station. It corresponds to the computer in Figure 9. The nodes in the network are numbered:

- nodes 1 and 8 represent light sensors;
- nodes 2, 4 and 9 represent presence sensors.
- nodes 3 and 10 represent smoke sensors;
- nodes 5, 6 and 7 represent movement, humidity and UV sensors from Bob's step-counter.

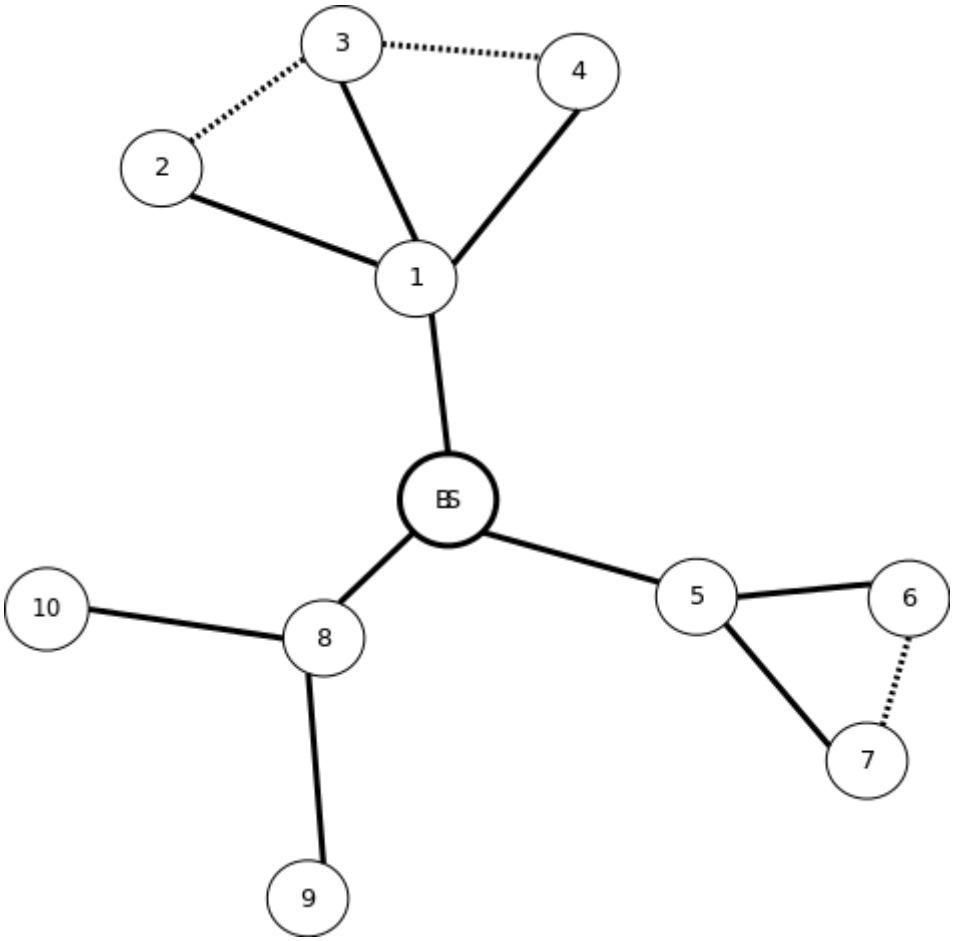

Figure 10 An example network with N=2

The solid lines represent the routes leading from the sensors to the base station. In addition, the dotted lines connect the nodes being in their direct range. For example, node #2 is able to communicate directly with nodes #1 and #3, and node #1 is its router to communicate to the base station.

In this example network, the number of tiers N = 2. The tiers are as follows:

- tier 1 = { BS, 1, 5, 8 },
- tier 2 = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }.

The height of this network is h = 2. The levels of nodes are as follows:

- level 1 = { 1, 5, 8 },
- level 2 = { 2, 3, 4, 6, 7, 9, 10 }.

The nodes of tier 1 (without base station) are routers – they forward messages from the base station to their child nodes and from their child nodes to the base station.

It is assumed that the network applies the *Fully Synchronized* sleep pattern [40] [46] presented in Figure 11. The arrows represent the fastest possible scenario of sending a message from a node of the second level of the network to the base station, for example from the presence sensor (node #2), to the Bob's computer (base station).

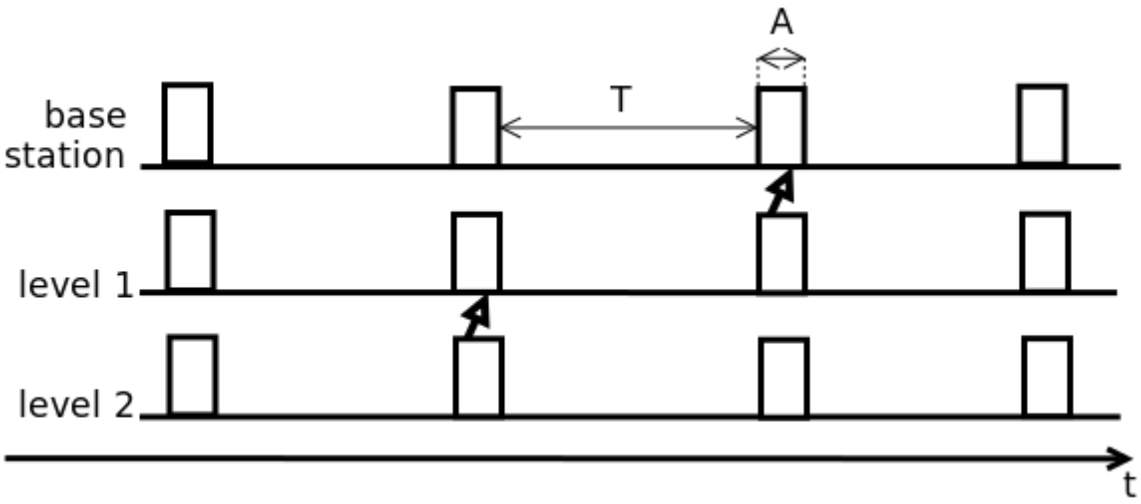

Figure 11 Fully Synchronized sleep pattern for network from Figure 10

3.3 Example threats

Using the ANGEL platform can be insecure without proper security mechanisms. The following scenarios present examples of possible threats that may affect Bob, and describe how trust management can help Bob to discover such risks and counteract them.

3.3.1 Unfair services supplier

Bob may want to insert to the system another device, or to add new services to the existing ones. There is a risk that software embedded in such new device could have a harmful effect on the system, and thereby endanger the health and, in extreme situations, even Bob's life. With some level

of authorization, the new device could also have access to Bob's personal information which could lead to violation of Bob's privacy.

3.3.2 Malignant neighbour

Bob has a neighbour who does not like him. The neighbour has some technical knowledge and he bought an equipment allowing him to reprogram WSN nodes. The neighbour wants to change the behaviour of Bob's nodes to cause some trouble. It can be not only harmful for the system, but also for Bob's life and can allow the neighbour to access sensitive data.

3.3.3 Faulty nodes

A sensor can fail and stop transferring data or it transmits incorrect data. Lack of transmission can be detected relatively easily – neighbours of the node that stopped to transmit data after the specified time warn the base station and it informs Bob about the problem. In case the node transmits corrupted data, the consequences could be worse.

3.3.4 Using trust management to defend threats

It is assumed that the network nodes are equipped with testing capabilities sufficient to detect incorrect data before the data reaches the base station of the network. They are also able to detect actions not permitted by other nodes role and / or prohibited by the network policies. With the trust management functionality on, such actions result in reducing the sender's trust value what can eventually lead to cutting the source of incorrect data off. For example, if the light sensor sends information about light intensity exceeding the values set in the network policy or sends information that it is not supposed to send, it will be detected before the lamps in Bob's apartment are set incorrectly. The culprit sensor will be excluded from the network and other nodes will not accept data from it until Bob decides to fix the problem and restore trust to that node. Adequate notice sent to the base station and displayed on the TV or on the Bob's smartphone allows him to quickly learn about the problem associated with the improperly working device.

4. WSN security

The low cost of WSN devices allows to deploy large sensor arrangements capable of performing both military and civilian tasks in a variety of conditions. But sensor networks also have severe resource constraints due to their lack of data storage and power. These are major obstacles to the implementation of traditional computer security techniques in a wireless sensor network. The unreliable communication channel and unattended operation make the security defences even harder [47]. Wireless sensors often have the processing characteristics of machines that are decades (or longer) old, and the industrial trend is to reduce the cost of wireless sensors while maintaining similar computing power [48]. Walters at al. distinguish 3 major obstacles of security in WSN [47]:

- Limited resources: limited memory and storage space, power limitation;
- Unreliable communication: unreliable transfer, packet conflicts, latency and problems with synchronization;
- Unattended operation: exposure to physical attacks, remote management.

Most of the threats and attacks against security in wireless networks are similar to their wired counterparts while some are exacerbated with the inclusion of wireless connectivity. It can be explained that wireless networks are usually more vulnerable to various security threats than wired network [49]. Moreover, traditional security mechanisms with high overhead are not feasible for resource constrained sensor nodes [50].

The researchers dealing with WSN security have proposed a number of various security schemas optimized for WSN. Also many routing protocols and data aggregation protocols have been proposed which claim to be secure and effective in WNS environment. Due to decentralized nature of the network some researchers have proposed schemas involving node collaboration and trust models. The intention is to provide solution to the problems that cannot be resolved by traditional cryptography methods.

4.1 Known vulnerabilities

Wireless Sensor Networks are vulnerable to attacks for many reasons. The main reasons are [8] [51]:

- No need of physical access to the device to connect to it. If an attacker can wirelessly reach the node, he/she can attack it. Lack of central infrastructure enforces security mechanisms implemented on every node in the network.
- The autonomy of nodes - nodes make decisions about routing and data processing themselves what increases the risk of data leakage in the event of a physical takeover and

reprogramming. In addition, the larger the network, the more difficult is to trace and monitor a single node.

- Decentralisation of decision-making – the lack of a central authority allows the attacker to use techniques which allow to break the cooperation algorithms.
- Open structure of the network – each device in the range of one of the network nodes may initiate the procedure of connecting to the network. It allows devices not compliant with network policies to connect and influence the network operation.
- Physical device constraints – WSN nodes are small devices, usually with an internal, low power source, which usually has a small computational, memory and transmission resources. Therefore, it is impossible to use the same cryptographic methods as in wireless networks for more powerful devices (e.g. IEEE 802.11 standard).
- Elimination of some security solutions e.g. based on static configuration. Due to the WSN mobility and constantly changing topology, the network nodes must continually discover and evaluate new nodes appearing within their range.

4.2 Example attacks

Three popular attacks were chosen to demonstrate the possible threat they can cause. To illustrate the described attack scenarios, there will be made references to the example network and threats introduced in Section 3.3.

Each attack will be briefly described and attack scenario will be given. Attack description will end with description, when an attack can be considered successful.

4.2.1 Spam attack

Spam attack happens when unnecessary and useless messages are generated and spread over the network. This attack is intended to waste network resources - energy, bandwidth – especially of those nodes that receive the packets and resend them (*routers*). This type of attack can quickly isolate router nodes from the rest of the network. In this way, the whole network may be destroyed, despite the fact that most nodes are still operational.

The attack can be the result of connecting a new node to the network (the unfair services supplier threat), the result of intentional actions (malignant neighbour threat) or the result of a node breakdown (the faulty node threat).

Attack scenario

The attacker node in its wakeup period A sends up to n messages to the base station. Figure 12 illustrates the spam attack – the node on the first level of the network sends more messages than it is supposed to do in its wakeup period.

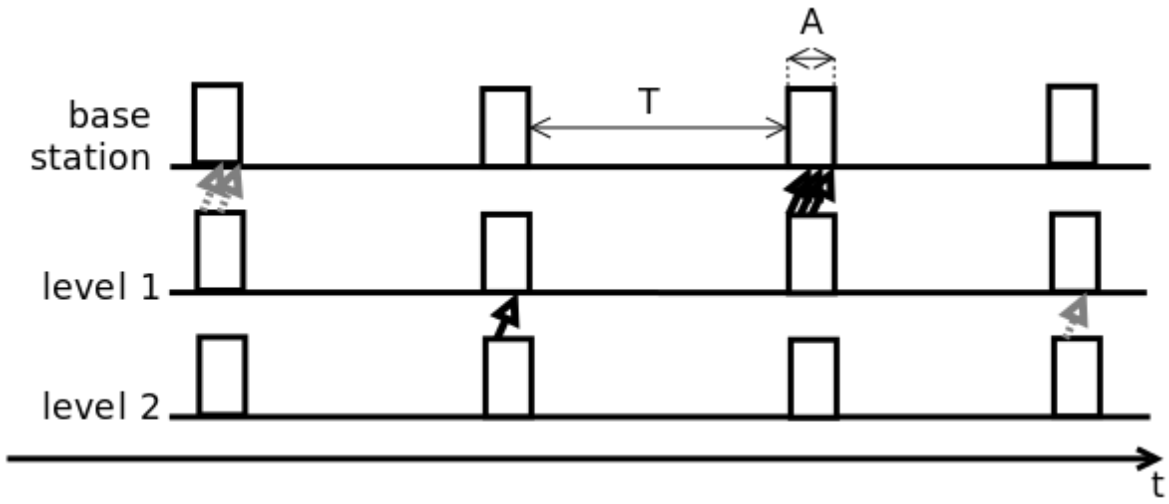

Figure 12 Spam attack

Attack success criteria

The attack is considered successful, if the attacker manages to deliver spam messages and remains undetected.

4.2.2 Black hole attack

In its simplest form the black hole attack is losing the received packets to prevent their further propagation. This situation is easy to detect due to the specificity of sensor networks – a network must be ready for sudden disappearance of a node, e.g. due to an exhaustion of its energy source. Therefore, most protocols have a mechanism that periodically checks all the paths – if one of the paths stopped working it becomes re-created. This action eliminates a malicious node from the path and stops it from sabotaging the network operation. For this reason, black hole attacks use a mechanism of random transmission of packets to confuse the maintenance mechanism. This attack affects router nodes only.

The attack can be the result of connecting a new node to the network (the unfair services supplier threat), a result of intentional actions (malignant neighbour threat) or a result of a node breakdown (the faulty node threat).

Attack scenario

The attacker node does not forward any messages or forwards only some of them (to the base station and to the other nodes). All not-forwarded messages are dropped. Figure 13 shows the black hole attack led by a node on the first level of the network. The messages which are dropped in the black hole are crossed out in red.

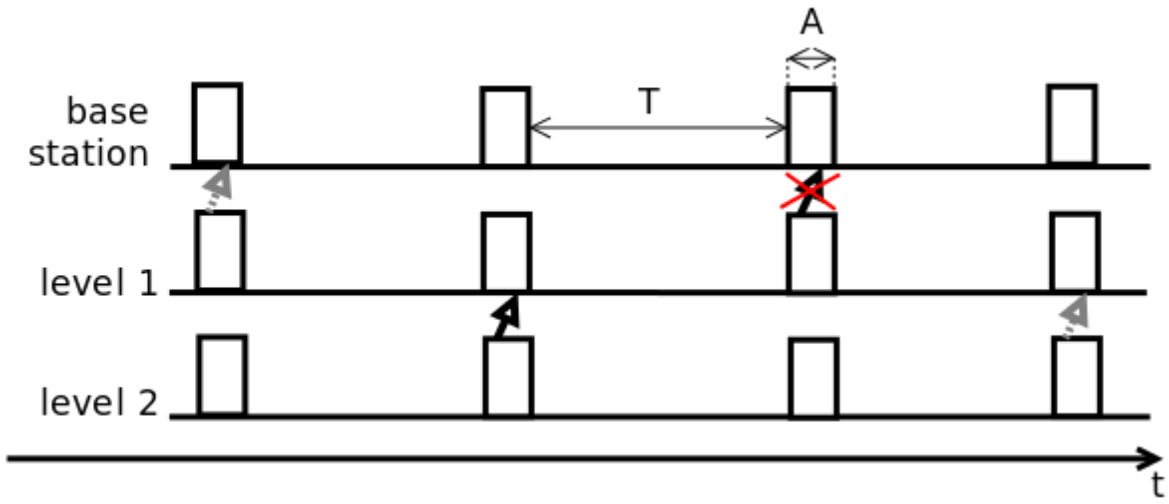

Figure 13 Black hole attack

Attack success criteria

The attack is considered successful, if the malicious node lying on at least one routing path blocks messages and remains undetected.

4.2.3 Message modification attack

Message modification attack involves modification of a message data and resending the message. The attacker can modify a content, information about the receiver node and/or information about the sender node. This attack targets router nodes only.

The attack can be the result of connecting a new node to the network (the unfair services supplier threat) as well as the result of a node breakdown (the faulty node threat).

Attack scenario

The attacker node changes all or some messages that it forwards (to the base station and to other nodes). All other messages are forwarded without any modification that breaks the network policy (some message attributes can change during standard forwarding). Figure 14 shows a message modification attack – the messages that were maliciously modified by the node on the first level of the network are marked in blue.

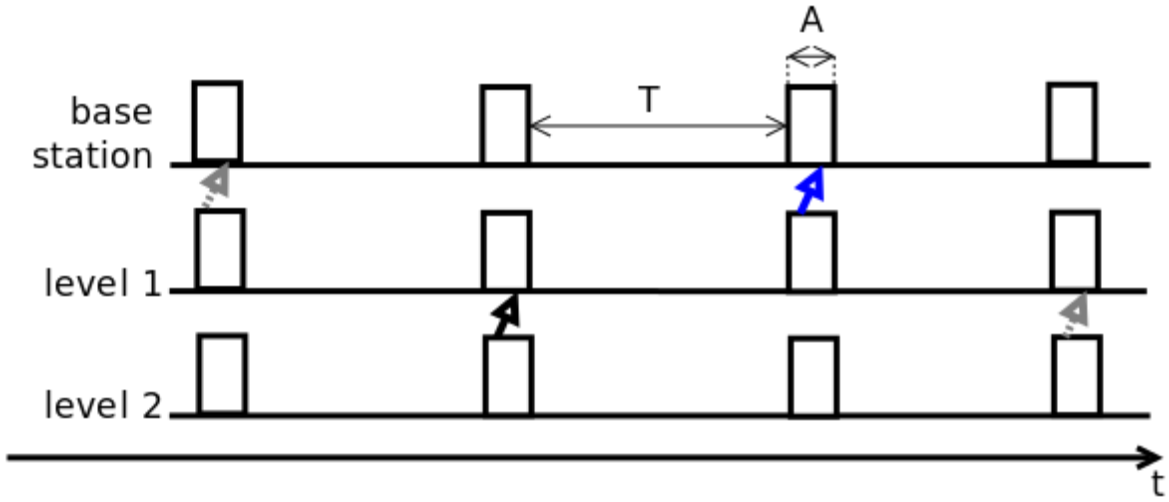

Figure 14 Message modification attack

Attack success criteria

The attack is considered successful, if the malicious node lying on at least one routing path modifies messages and remains undetected.

4.3 Defence against attacks

To defend against the attacks presented in the previous section, nodes can be equipped with security mechanisms presented below.

4.3.1 Spam attack

As the target of the spam attack in a wireless sensor network are usually router nodes, the defence methods should focus mainly on this type of nodes. This approach is used by Detect And Defend Spam (DADS) [52]. It proposes the concept of 'quarantine regions' which isolates the attackers. A dedicated remote node is responsible for detecting the spammer nodes. Their detection can be conducted in three ways:

- Filtering the incoming messages to detect a node which regularly is the sender of wrong messages.
- Referring to the average frequency of sending messages for the nodes that belong to the region.
- Monitoring the rate of packet generation by each node.

When the number of packets arriving at the router node exceeds a certain threshold, the router broadcasts an alarm message, called Defend Against Spam (DAS) message. The primary objective of DADS is the isolation of the spammer node by its neighbours. Each node that receives the DAS, starts its local timer. While timer countdown, the node forwards authenticated messages only. If it receives

a non-authenticated message from another node, it asks this node to send the message again, this time with authentication. In case of failure, it considers that it is in the *quarantine area* and enters a mode in which it only accepts and sends authenticated messages (even after countdown ends). In this way, slowing down the messaging occurs only among neighbours of the node and among the nodes, from which the only way to reach the router goes through the quarantine area. The rest of the network can send messages normally.

In order not to leave the network in a constant state of quarantine (which in extreme cases due to spammer nodes mobility could cover the entire network), if for the specified period of time there has been no failed attempts to authenticate, the node returns to the normal mode.

4.3.2 Black hole attack

To defend against black hole attacks the schema using watchdog and path rating can be used. It was described by Marti et al. [53]. Watchdog mechanism acts as a listener and checks if a message recipient sends the message to the next-hop-recipient. Each node has a counter of not resend messages for each of its neighbours. After exceeding a threshold value, the node considers the neighbour as being harmful and reports this to the base station. The mechanism of paths evaluation calculates the ratio of the successful to unsuccessful packet delivery on each path thus making the selection of the most reliable path available. The combination of these two mechanisms allows to avoid paths containing malicious nodes as well as their detection.

Another solution of the black hole attack has been described by Avramopoulos et al. [54]. It uses the following mechanism:

- Source routing – the source node specifies in each packet a sequence of nodes through which the packet should be sent.
- Confirmation of target – the recipient node sends an acknowledgment packet (ACK) to the source along the same path (in reverse order) after receiving a packet.
- Timeouts – the source and each intermediate node sets a timer which specifies the time interval at which it expects an ACK from the recipient or Fault Announcement (FA) from another intermediate node.
- FA – when the timer counts down to zero and does not receive an ACK, it generates FA and transmits it to the source.

Additionally, all data, ACKs and FAs are authenticated using message authentication code, which ensures that they are not forged by a malicious node. Detection of FA means a potential problem and allows to select a different path.

4.3.3 Message modification attack

To detect modification of a message the scheme proposed by Sivanantham et al. [55] can be used. They propose a method, which can identify misbehaving router nodes that drop or modify packets, by continuously monitoring the behaviours of the nodes in the network. This scheme contains three techniques:

- Node Monitoring: to locate and identify packet modifiers (or droppers), nodes are continuously monitored for forwarding behaviours and reputation of every node is published among the network and maintained in the base station.
- Packet Sealing: when the sensor data are transmitted by nodes to the base station, each packet sender or forwarder seals the data by adding a small number of extra bits called packet seals, from which the base station could obtain useful data related to the transmission. Based on the packet seals, the base station can figure out the dropping ratio of every sensor node.
- Node Classification: the base station identifies and classifies the nodes that are modifiers (or droppers). The behaviour of nodes is traced in variety of scenarios and with the information accumulated in base station, it classifies the nodes as modifiers (or droppers).

The similar technique with packet sealing has been proposed by Vijayalakshmi et al. [56].

5. Proposed approach to trust management

5.1 Basic concepts

A dictionary definition states that *trust* is a belief or confidence in the honesty, goodness, skill or safety of a person, organization or thing [57]. Another definition [58] says that: trust is a bet that those entities, which you cannot control, will act in a predictable manner that is favourable to your cause. Generally, trust is a relation between the *trustor* (a trusting subject) and the *trustee* (a subject being trusted).

It is suggested that each node of a network should be examined if it can be trusted and that all nodes should cooperate in that process [10] [59]. The objective of trust management system is to distinguish between trustworthy network nodes and untrustworthy ones. Then, the trustworthy nodes can cooperate to provide trustworthy network services and the untrustworthy nodes are excluded from the network [60].

Trustworthy network services can be provided if they are based on trustworthy information. Therefore, there is a need for a mechanism for assessing if a data item is trustworthy before it is subjected to further processing and passed through the network. Distrusted data are discarded and the trustworthiness assessment of the source of this data is being lowered. To limit the potential damage, it is important to assess the trustworthiness as early as possible to prevent distrusted data from further processing. For large networks, centralized assessment by a dedicated node would lead to performance problems and excessive concentration of network traffic. Therefore, it is assumed that every node in the network is involved in trustworthiness assessment and the trust related decisions are distributed.

For the purposes of this dissertation the following definition is assumed: trust is an act of acceptance of a message received from a network node which results from the assessment of the *trustworthiness* of the message and its source.

A network node acts in a dual role – as a trustor and a trustee:

- for outgoing communication, the node acts as a trustee – other nodes judge if it can be trusted,
- for incoming communication, the node acts as a trustor – it makes a real-time decision if the sender can be trusted.

The distinction is illustrated in Figure 15.

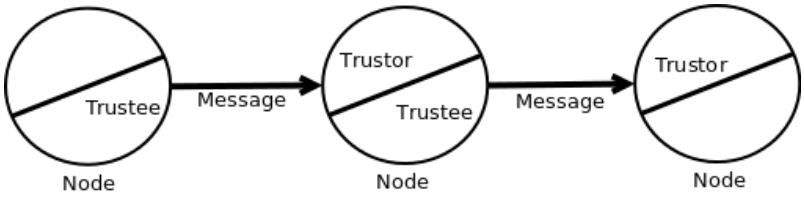

Figure 15 Idea of trustor and trustee role

A message sent from one node to another is always a trustee – the receiver node (the trustor) makes a real-time decision if the message is valid and can be trusted.

5.2 WCT2M trust management method

In this section a new trust management method called *WSN Cooperative Trust Management Method - WCT2M* is introduced.

5.2.1 Steps of WCT2M – instantiation of the method

WCT2M is represented by a software package called *WCT2M software.* To run WCT2M in a network, the network administrator needs to implement the following steps:

Step 1: Install WCT2M software on the network nodes, including the base station. Set parameters of the method. During installation choose if Trust History and/or Action History features are to be enabled on the nodes. If so, set the parameters of these features.

Step 2: For each node, configure access of WCT2M software to the routing information (to provide for distinguishing the base station) and to the synchronization protocol run by the network.

Step 3: For each node, interface WCT2M software to the software installed on the node to provide access to incoming messages and to the results of their assessment by security mechanisms enabled on the node.

Step 4: Start the network and allow to propagate trust information between nodes.

To join a new node/nodes to the existing and running network, the network administrator needs to follow the steps 1-3 and then the new node/nodes should be joined to the running network.

5.2.2 Data types of WCT2M

It is assumed, that all nodes use the same scale called *trust scale*. There are three characteristic values related to this scale:

- *full trust level* – means that the node is fully trustworthy,

- *initial trust level* – is the initial credit given to a node (for instance when the node joins the network and its trustors have yet no other evidence related to its trustworthiness),
- *cut-off level* – is the trust level below which the node is considered untrustworthy.

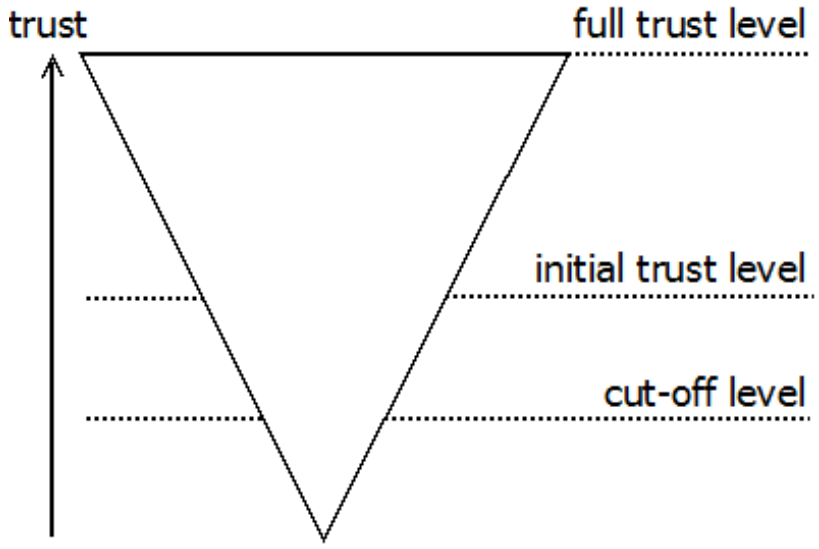

Figure 16 The trust scale

Figure 16 presents a graphical form of the scale. A value from the trust scale assigned to a given node by another node is called its trust value. A new node N in the network is credited with the initial trust in the trust scale. This determines its initial trust value. Then, depending on the behaviour of N, its trust value can change. When the trust value drops below the cut-off level, N is perceived as untrustworthy and the messages received from this node are distrusted.

The method assumes that if node's trust value drops below the cut-off level, the node cannot regain trust unless the base station resets its trust value the initial level.

It is assumed that the trust scale is mapped on the interval of real numbers [0..1] where full trust = 1 and the other characteristic points could be for instance:

- cut-off level = 0,2;
- initial trust = 0,5.

Each network node participates in the trust management process and maintains data on the reputation of other nodes. The corresponding data structure is called *trust table*. The trust table structure with example entries is presented in Figure 17. Every entry in the trust table is assigned a trust value from the trust scale.

Example 2

The trust table maintained in node 1 of the example network shown in Figure 10 could look as presented in Table 1.

Table 1 Trust table example

Node ID	Trust value
2	0.78
3	0.52
4	0.93
5	0.79
6	0.81
7	0.91
8	0.87
9	0.29
10	0.83

Recommendation is an entry of a local trust table sent to another node. A node can recommend any other node except itself.

5.2.3 Method description

WTC2M is embodied by a software package to be installed on network nodes. The class model of this software package is presented in Figure 17.

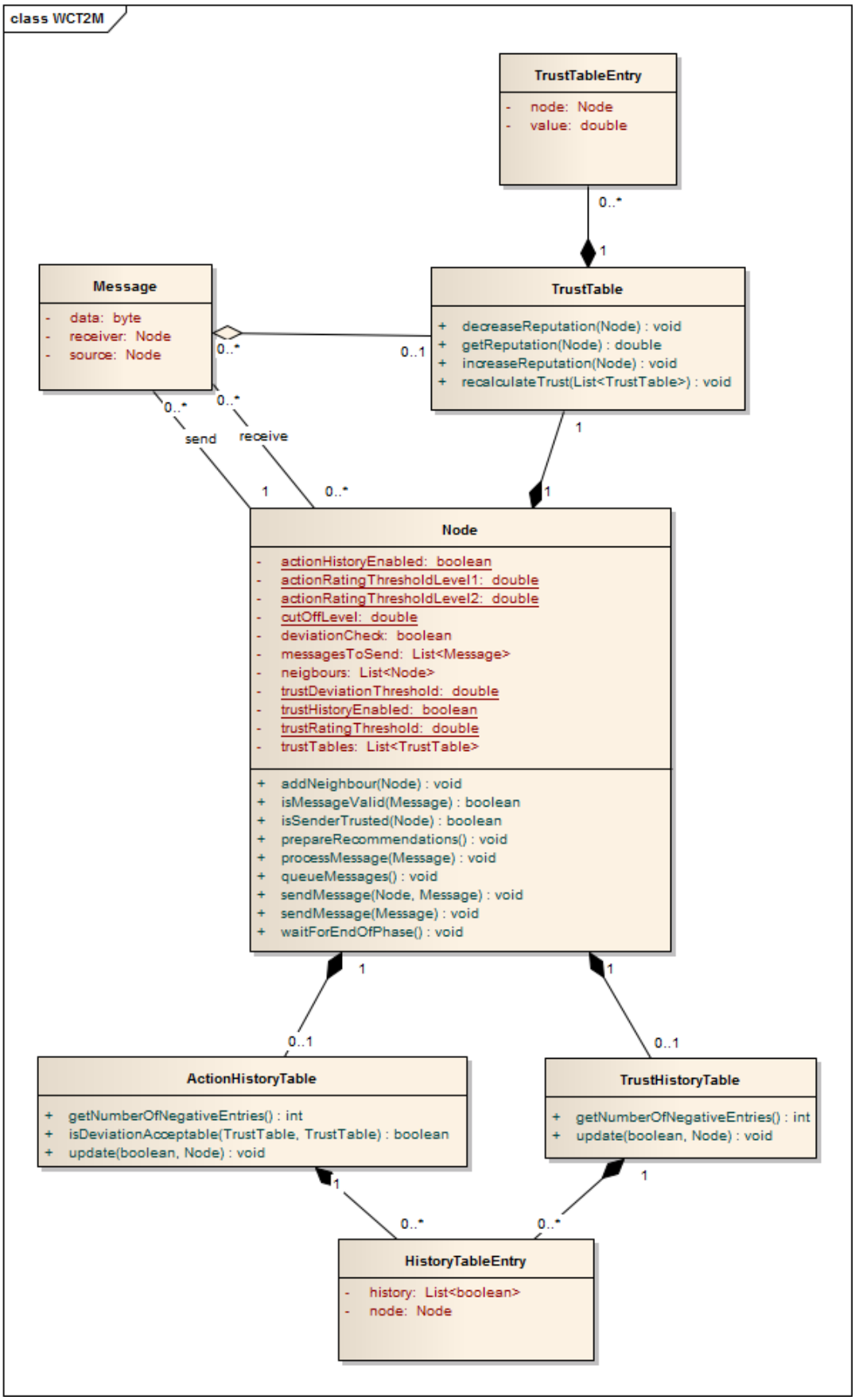

Figure 17 WCT2M basic class model

The model consists of the following classes (in the following text entities stored at the node executing a given method will be prefixed by 'own', e.g. 'own trust table'):

- `Message` represents a message (`data` attribute) exchanged between nodes (`source` and `receiver` attributes), including the base station.
- `TrustTable` represents trust values assigned to other nodes by a given node. It allows to execute the following methods:
  - `decreaseReputation(Node)` sets a new, decreased, trust value (calculated from the current trust value) to `Node` in the own trust table;
  - `getReputation(Node)` returns current trust value for the given `Node`;
  - `increaseReputation(Node)` sets a new, increased, trust value (calculated from the current trust value) to `Node` in the own trust table;
  - `recalculateTrust(List<TrustTable>)` updates the values stored in the own trust table based on the recommendations received from neighbour nodes (the values from the trust tables passed as the parameter of this method.
- `TrustTableEntry` represents a single entry of a TrustTable. It indicates a node and a trust value assigned to it.
- `Node` represents a node of the network (including the base station). It stores the following attributes (fixed attributes set during instantiation of WCT2M, see Section 5.2.1, are underlined):
  - `actionHistoryEnabled` – if action history is enabled;
  - `actionRatingThresholdLevel1` – action rating threshold level 1 used to calculate values inserted to `ActionHistoryTable`;
  - `actionRatingThresholdLevel2` – action rating threshold level 2 used to calculate values inserted to `ActionHistoryTable`;
  - `cutOffLevel` – cut-off level;
  - `deviationCheck` – auxiliary variable storing deviation check result;
  - `neighbours` – list of own neighbours;
  - `messagesToSend` – list of `Messages` to send;
  - `trustDeviationThreshold` – trust deviation threshold used to calculate values inserted to `ActionHistoryTable`;
  - `trustHistoryEnabled` – if trust history is enabled;
  - `trustRatingThreshold` – trust rating threshold used to calculate values inserted to `TrustHistoryTable`;
  - `trustTables` – auxiliary variable storing list of valid recommendations.

`Node` allows to execute the following methods:

- `addNeigbour(Node)` adds a given `Node` to the own `neighbours` list.
- `isMessageValid(Message)` communicates with the security mechanisms interfaced to WCT2M software during instantiation (configured during step 3 of instantiation of WCT2M, described in 5.2.1) and checks if a given message is valid in the light of the security policies implemented at the own node;
- `isSenderTrusted(Node)` checks in the own trust table if a given node is considered as a trusted (its trust value is above the cut-off level);
- `prepareRecommendations()` prepares new `Message` containing own trust table to send as recommendations and adds it to the `messagesToSend` list;
- `processMessage(Message)` processes the incoming `Message` in accordance with the own node objectives;
- `queueMessages()` adds `messages` that the node needs to send (new `messages` created by the node or `messages` received by the router node to be forwarded) to the `messagesToSend` list;
- `sendMessage(Node, Message)` sends `Message` to the node specified by first parameter of this method and removes it from the `messagesToSend` list;
- `sendMessage(Message)` sends `Message` to all neighbours of the own node and removes it from the `messagesToSend` list;
- `waitForEndOfPhase()` pauses till the end of the current phase (execution of this method depends on the synchronization protocol which is assumed for the considered network - see Section 5.2.4).

- `TrustHistoryTable` maintains the history of validity evaluations (with the help of isMessageValid `(Message)` method) of the incoming messages. The result of each such evaluation (positive or negative) is stored in a FIFO (First In, First Out) queue of a fixed size called *trust history table*. Every node maintains a separate copy of this table for each of node it knows. If for a given node – after inserting a new value to its history – the number of negative assessments in the trust history exceeds a given threshold, called *trust rating threshold*, the trust value of this node is decreased by calling `decreaseReputation(Node)` method on the own trust table. This check and the resulting action (if needed) is executed after every incoming message validity evaluation made by the node.

  The class allows to execute the following methods:

- `getNumberOfNegativeEntries(Node)` returns the number of *False* values in the `HistoryTableEntry` of `Node` in the `TrustHistoryTable`;
- `update(boolean, Node)` inserts to the `TrustHistoryTable` of `Node` the first parameter (boolean) on the FIFO basis.

- `ActionHistoryTable` represents the history of deviation evaluations obtained using `isDeviationAcceptable(TrustTable, TrustTable)` method. When node A receives recommendations from another node B it calls `isDeviationAcceptable(TrustTable`$_A$`, TrustTable`$_B$`)` where `TrustTable`$_A$ is its own trust table with trust values of the known nodes and `TrustTable`$_B$ is the trust table with recommendations received from B. The `isDeviationAcceptable` method calculates the deviation of the recommendations received from B with respect to the trust values maintained by A in accordance to the following formula:

$$deviation_{A-B} = \frac{\sum_{n=1}^{N} abs(Reputation_{A\to n} - Recommendation_{B\to n})}{N}$$

where N is the cardinality of the intersection of two sets: the set of entries of A's trust table (the nodes known to A) and the set of entries of the trust table received from B. If $deviation_{A-B}$ is lower than a given threshold, called *trust deviation threshold*, the method returns *True*, otherwise *False*. The returned value (positive or negative) is stored in a FIFO queue of a fixed size called *action history table*. Every node maintains a separate copy of this table for each node it knows. If for a given node – after inserting a new value – the number of negative assessments in the action history table exceeds a given threshold, called *action rating threshold level 1*, the trust to the node is decreased by calling the `decreaseReputation(Node)` method on the own trust table. If the number of negative assessments in the action history table exceeds another threshold, called *action rating threshold level 2* (where *action rating threshold level 2* > *action rating threshold level 1*), the trust to the node is decreased again by another call of `decreaseReputation(Node)`. This check and the resulting action (if needed) is executed always after receiving a valid (positively verified by `isMessageValid(Message)` method) recommendation from another node. The example for this feature is presented in Section 5.2.4.

The class allows to execute the following methods:

- `getNumberOfNegativeEntries(Node)` returns the number of *False* values in the `HistoryTableEntry` of `Node` in the `ActionHistoryTable`;

- `isDeviationAcceptable(TrustTable, TrustTable)` calculates a deviation of the received trust table from the own trust table and checks if it is lower than the trust deviation threshold;
- `update(boolean, Node)` inserts to the `ActionHistoryTable` of `Node` the first parameter (boolean) on the FIFO basis.

- *`History Table Entry`* represents a single entry of a `TrustHistoryTable` or `ActionHistoryTable`. It indicates a `Node` and a list of boolean values assigned to it.

5.2.4 Execution model

Execution of the WCT2M enabled network is structured into *WCT2M cycles*, where each cycle is divided into two *phases*:

- *Data phase:* Each node sends/receives 'regular' messages to/from the base station and communicates with its neighbours. Each node also forwards messages received from other nodes, if it is a router node. The receiver node (the trustor) evaluates the sender nodes and the incoming messages (the trustees) if they are valid and can be trusted and then updates trust values of these nodes accordingly.
- *Recommendation phase:* Each node broadcasts recommendations to its neighbours. The recommendations are related to the end of the data phase (the trust tables exchanged as recommendations are the snapshots taken at the end of the data phase, after all updates related to the incoming messages were inserted). The receiver node (the trustor) evaluates the sender nodes (based on its own trust table) and the incoming recommendation messages (the trustees) if they are valid and can be trusted, then the receiver updates trust values of the sender nodes accordingly and recalculates its trust table referencing to the received valid recommendations (the recommendations from untrusted nodes and the recommendations assessed as being invalid are excluded).

The exact length of the data phase and the recommendation phase depends on the MAC protocol used in the network, sleep scheduling pattern adopted and the density of the network[6]. It means the length of cycles can vary during whole life of a network. The idea of WCT2M cycles and phases is presented in Figure 18.

[6] Two interfering nodes cannot transmit at the same time because of message collisions [124], so in dense networks recommendation exchange takes more time than in sparse ones.

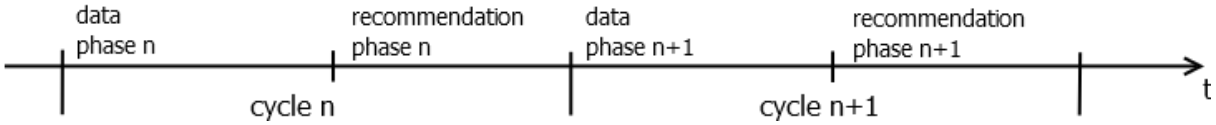

Figure 18 WCT2M cycles and phases of the WTC2M enabled network

WCT2M assumes that every node of the network participates in the synchronization protocol so that it is able to synchronize with other nodes to work in such cycles. The method also assumes that the synchronization protocol controls when a given phase ends and none of the actions included in this phase lasts forever. Moreover, each node can skip sending messages during data phase and it does not block the phase end.

Each node runs the Network Algorithm which is presented in Figure 19. It defines how the nodes in the network cooperate to provide trust management.

```
start data phase:
queueMessages()
for Message in messagesToSend
      sendMessage(Message.receiver, Message)
end
waitForEndOfPhase()
start recommendation phase:
prepareRecommendations()
for Message in messagesToSend
      sendMessage(Message)
end
for every trust table received during this phase
      if actionHistoryEnabled == true
            deviationCheck :=
actionHistoryTable.isDeviationAcceptable(trust table received during this
phase, own trust table)
            actionHistoryTable.update(deviationCheck, sender of assessed
trust table)
            if actionHistoryTable.getNumberOfNegativeEntries(sender of
assessed trust table) >= actionRatingThresholdLevel1
                  trustTable.lowerReputation(sender of assessed trust
table)
                  if actionHistoryTable.getNumberOfNegativeEntries(sender
of assessed trust table) >= actionRatingThresholdLevel2
                        trustTable.lowerReputation(sender of assessed trust
table)
                  end
            else
                  trustTable.recalculateTrust(received trust table)
            end
      else
            trustTable.recalculateTrust(received trust table)
      end
end
waitForEndOfPhase()
goto start data phase
```

Figure 19 WCT2M Network Algorithm

The method assumes that each node of the network, including the base station, runs the Node Algorithm which is presented in Figure 20. The Node Algorithm is executed on a node each time the trustor receives a message from another node (the trustee). In the algorithm,

- `incoming_message` denotes the message being processed by the algorithm,
- `sender_of_incoming_message` denotes the sender of this message,
- `sender_of_assessed_trust_table` denotes the sender of the trust table maintained in the current entry of `trustHistoryTable`.

```
if isSenderTrusted(sender of incoming message)
      if isMessageValid(incoming message)
            trustTable.increaseReputation(sender of incoming message)
            if trustHistoryEnabled == true
                  trustHistoryTable.update(true, sender of incoming
message)
                  if trustHistoryTable.getNumberOfNegativeEntries(sender of
assessed trust table) >= trustRatingThreshold
                        trustTable.decreaseReputation(sender of incoming
message)
                  end
            end
            processMessage(incoming message)
      else
            trustTable.decreaseReputation(sender of incoming message)
            if trustHistoryEnabled == true
                  trustHistoryTable.update(false, sender of incoming
message)
                  if trustHistoryTable.getNumberOfNegativeEntries(sender of
assessed trust table) >= trustRatingThreshold
                        trustTable.decreaseReputation(sender of incoming
message)
                  end
            end
      end
end
```

Figure 20 WCT2M Node Algorithm

The Node Algorithm involves decisions based on trustworthiness assessment of the sender (`isSenderTrusted(Node)` method) of the message and the message itself (`isMessageValid(Message)` method). The assessment is based on two complementary approaches:

- *Policy-based approach*: trustor evaluates trust value of the trustee assessing the trustee's state or its observed behaviour and its conformance with agreed policies, notably the security policy.
- *Reputation-based approach*: trustor takes into account information from other nodes if they trust the trustee (`TrustTableEntry` for trustee from `TrustTable`).

The idea of this assessment is illustrated in Figure 21.

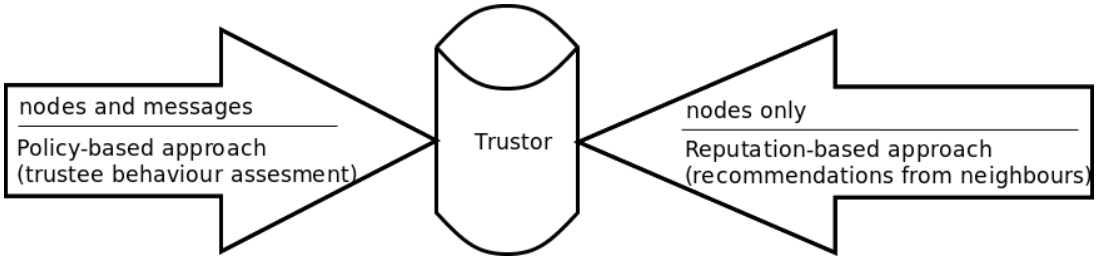

Figure 21 Two sources of trust assessment

The following are examples of evidence considered while deciding about trusting or distrusting the trustee:

- security check of a message;
- formal correctness of a message;
- the right of the sender to send messages of a given type to the recipient node;
- message content check;
- sender's trust value maintained by the trustor.

Depending on the trust assessment result the trustor performs appropriate actions according to the WCT2M Node Algorithm (Figure 20).

The trust management policy of WCT2M requires that the network nodes obey the following rules:

- local trust table can be sent on demand – for instance a new node joining the network asks its neighbours for recommendations;
- each node periodically broadcasts its local trust table (during recommendation phase);
- each local trust table or some of its entries can be reset as the result of a special command from the central node of the network (acting as the trust manager);
- only these recommendations are taken into account that come from the trusted nodes (the nodes with trust value higher than the cut-off level).

Nodes exchange trust tables only with their neighbours. If WCT2M is used in the clustered network, nodes exchange trust tables only with their neighbours belonging to the same cluster.

Example 3

In the network shown in Figure 10, node 2 exchanges recommendations only with nodes 1 and 3. However node 1 exchange recommendations with nodes 2, 3 and 4 (it is the cluster

head of the lower tier) and with the base station (as the cluster head of the lower tier it is a member of the higher tier).

Trustee's trust value depends on two factors: policy-based and reputation-based approaches, discussed earlier. The influence of these factors is characterized by the *Cooperation Factor* (CF), same for all the nodes in the network. CF assumes values from 0 to 1, where 0 means that recommendations are discarded in trust value calculation and 1 means that the trust value is solely based on recommendations, as illustrated in Figure 22.

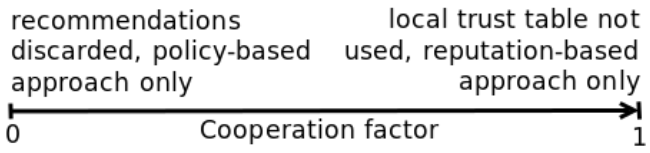

Figure 22 Cooperation Factor

Assuming that node A has received N recommendations (where N > 0) from its trusted neighbours (other than B) in a given WCT2M cycle, at the end of the cycle A will recalculate trust value of B in accordance to the following formula:

$$TrustValue_B := \frac{\sum_{n=1}^{N}(I_n \times Recommendation_{n \to B} + (1 - I_n) \times TrustValue_B)}{N}$$

where

$$I_n = CF \times Reputation_n,$$

$TrustValue_n$ denotes trust value of node n maintained by node A,

$TrustValue_B$ denotes trust value of node B maintained by node A, and

$Recommendation_{n \to B}$ – denotes the recommendation concerning node B sent by node n to node A.

The formula means that every valid recommendation concerning node B is used by node A to calculate the new trust value towards node B. However, the scale of the change depends on how trusted is the recommending node in opinion of node A.

Example 4

If node 1 of the example network shown in Figure 10, having the trust table presented in Table 1, with CF = 0.4, received in a given WCT2M cycle recommendations concerning node 4 from its trusted neighbour 2 ($recommendation_{2 \to 4}$: 0.82) and 3 ($recommendation_{3 \to 4}$: 0.95), the new trust value of node 4 will be calculated in the following way:

$$TrustValue_4$$
$$= \frac{0.4 \times 0.78 \times 0.82 + (1 - 0.4 \times 0.78) \times 0.93 + 0.4 \times 0.52 \times 0.95 + (1 - 0.4 \times 0.52) \times 0.93}{2}$$
$$= 0{,}91492$$

Trust value of the sender of a message maintained by the receiver of the message can also change in effect of the assessment of an incoming message. If node B sends a message to node A and A assesses that the message is against the assumed policies (such message is called *a spoiled message* and event of purposely sending such message is called *malicious action*), B's trust value maintained by A is decreased by $change_{negative}$ factor:

$$Reputation_B -= Reputation_B \times change_{negative}$$

In case the message agrees with the agreed policies, the trust value of the sender increases:

$$Reputation_B += (1 - Reputation_{B)} \times change_{positive}$$

For instance, $change_{negative}$ = 0,01 means that the trust value will decrease by 1% of its previous value.

$change_{negative}$ and $change_{positive}$ are the real numbers [0, 1].

Example 5

The network shown in Figure 10 has the action history feature enabled, the ActionHistoryTable size is set to 5 and action rating threshold is set to 3. Assume that at the beginning of a given WCT2M cycle node 1 has the ActionHistoryTable for node 3 with the following values: [negative, negative, positive, positive, negative].

- If the assessment is negative, the ActionHistoryTable will contain 2 positive and 3 negative entries, so (in addition to the 'standard' trust decrease resulting from the negative assessment), the trust value to node 3 will be additionally decreased by $change_{negative}$ factor.
- However, if during this WCT2M cycle the message received from node 3 is assessed positive, the ActionHistoryTable will contain 3 positive and 2 negative entries, so trust value to node 3 will be increased due to positive assessment and the additional action will not be executed.

5.2.5 Method security

Trust management helps to make the whole network more resistant to attacks, but it can be a target of attacks itself. Four popular attacks on trust management (Sun et al. [61]) and possible prevention methods are described below.

Decreased frequency of attack (On-Off attack)

Attack description:

To confuse trust management model, the attacker can decrease frequency of the attack. If the malicious actions are performed rarely enough, the trust value does not decrease to cut off the attacking node because the other non-malicious actions of that node increase it. The exact number of malicious actions and the schema of executing them (e.g. one malicious action every *n* WCT2M cycles or three malicious actions every *m* WCT2M cycles) that allows to delude trust mechanism depends on the parameters used by WCT2M, the number of nodes in the network or number of nodes that participate in the attack process.

Attack mitigation:

To minimize the influence of such behaviour, the WCT2M action history should be enabled. As described in Section 5.2.3, each node stores in the ActionHistoryTable the result of assessment of its neighbours' past actions which provides for detecting malicious node even if it performs malicious actions with decreased frequency. There is still a possibility that the frequency of malicious actions is so small, that the node will not be detected, but such node is barely harmful.

The longer history is maintained in ActionHistoryTable, the better protection can be provided. However, it costs extra memory. It can be particularly important in dense network, because a node need to maintain ActionHistoryTables for every neighbour.

Collusion attack (Bad mouthing attack)

Attack description:

The second way of trust management cheating is collusion of malicious nodes. Several nodes collude in order to rate each other with the maximum value and at the same time decrease other nodes' trust values by giving negative recommendations about the latter [62]. Detection of colluding nodes can be achieved by calculating truncated mean of recommendations or trust value variance and detecting the values divergent more from the obtained value than a given threshold.

Attack mitigation:

To minimize the collusion attack influence, the WCT2M trust history should be enabled, especially on router nodes. As they transfer many more messages than receive trust tables, an attacker has an ability to regain its trust value by sending valid messages. As described in Section 5.2.3, the node calculates deviation for all received trust tables from its own trust table. The result of this assessment is stored in TrustHistoryTable and if the received trust table vary too much from the own table, it is

not processed (the trust to other nodes remains unchanged). If such situation repeats too often, the trust to that node is decreased. There is still possibility that the nodes will collude with frequency small enough to stay undetected or will send trust tables with slight recommendations changes, but such behaviour has little effect on trust management.

As the usage of the TrustHistoryTable is similar to the ActionHistoryTable, the similar attention should be paid to the memory problems.

Sybil attack

Attack description:

In Sybil attack a malicious node disguises itself as multiple different nodes by advertising multiple identities to the neighbours. This allows the malicious node to increase the probability of being selected as a router node by other nodes. Moreover, the faked identities take the blame which should be given to the malicious node [8].

Attack mitigation:

The defence to the Sybil attack does not rely on the design of trust management, but the authentication and access control, which make registering a faked identification difficult [61].

Newcomer attack

Attack description:

If a malicious node can easily register as a new user, trust management can suffer from the Newcomer attack. In that case, a malicious node is able to advertise itself as a new user (register as a new network user), so its trust value is reset to the default one [61].

Attack mitigation:

The defence to Newcomer attack, as in Sybil attack case, does not rely on the design of trust management, but the authentication and access control, which make registering a new identification difficult [61]. However, vulnerability of the network to Newcomer and Sybil attacks are subject to verify while running a system using trust management.

5.2.6 Adjusting security level in accordance to trust value

To choose a proper set of security mechanisms and functions, the analysts takes in consideration many aspects that have influence on system security [63]. Usually the set of chosen security measures is the result of analysis, which security services the system should guarantee [64] [65]. In many cases analysts usually choose the strongest security measures to ensure the highest possible

security which leads to the greater system load [66], greater complexity and decreases availability [67]. It is the problem especially in systems with limited resources like Wireless Sensor Networks.

Moreover, due to limited resources and other specificity comparing to computer networks, traditional security methods are not sufficient or they use more resources than it is acceptable in WSNs. Researches invent new defences adjusted to WSN, but usually these are answers to only small sector of threats.

Lindskog [68] proposed *tunable security* as a solution. He noticed, that users of systems with certain level of Quality of Service (QoS) granted can choose many desired parameters, but not connected with security. The proposed solution allows to enhance QoS architectures to include security parameters.

Księżopolski et al. [67] introduce adaptable security mechanism which can change the security level depending on particular conditions that take place at a certain moment, and in given external conditions. The proposed model allows to assess the quality of security level using risk analysis. It allows to guarantee the mutable and adequate level of security based on the threats currently possible.

WCT2M also allows to introduce in the network the idea of dynamically chosen security level depending on current trust value of a sender. As illustrated in Figure 23, WCT2M cooperate with security mechanism implemented on the node allowing them to choose the most adequate level of security at the moment.

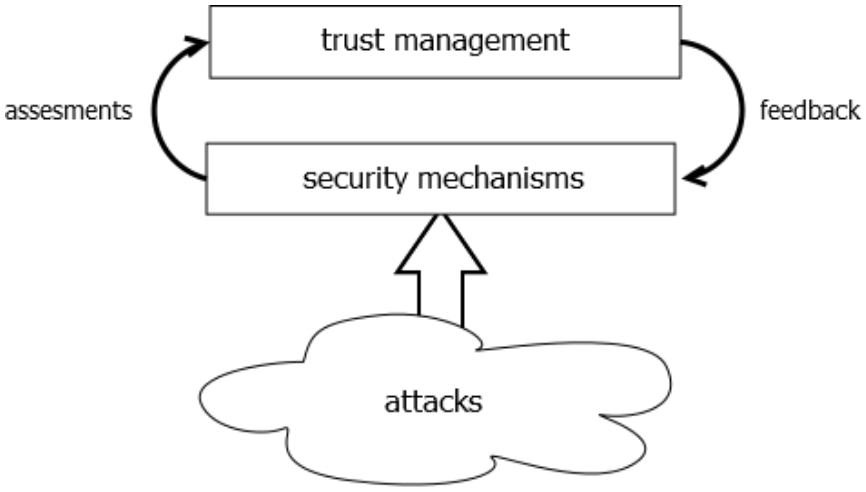

Figure 23 Defence against attacks in WSN with WCT2M

The method assumes, that WCT2M software:

- is interfaced to the software installed on the node to provide the assessment results, as described in Section 5.2.1;
- cooperating security mechanisms have access to the trust value of the nodes (using `getReputation(Node)` method) .

The cooperation is possible in two directions:

- Security mechanisms send assessment results of actions performed by other nodes to WCT2M. The method decides if the trust value of sender node should be decreased or increased.
- Security mechanisms query WCT2M about trust towards certain nodes and take suitable actions (e.g. conduct more/less detailed check of messages received from a certain node).

6. Analytic tools

To facilitate experimental evaluation of WCT2M, two analytic tools were prepared, namely the wireless sensor network laboratory and the simulator. The objective of the laboratory network was to demonstrate feasibility of WCT2M implementation in a typical WSN environment and to perform measurements of WCT2M performance. The objective of the simulator was to carry larger scale experiments using the performance parameters collected during the laboratory experiments aiming at evaluation of the scalability of WTC2M (overcoming the limitations in the number of network nodes of the laboratory environment).

In the following sections, firstly a more detailed description of the laboratory network is provided, next an overview of the presently available WSN simulators is given and then the simulator which has been created to analyse scalability of WCT2M is described.

6.1 The laboratory network

To validate the proposed method, a dedicated laboratory network was created using elements from CC2520 Development Kits [69]. CC2520 is Texas Instrument's second generation ZigBee/IEEE 802.15.4 RF transceiver for the 2.4 GHz unlicensed ISM band. This chip enables industrial grade applications by offering state-of-the-art noise immunity, acceptable link budget, operation up to 125 degrees and low voltage operation. CC2520 provides hardware support for packet handling, data buffering, burst transmissions, data encryption, data authentication, clear channel assessment, link quality indication and packet timing information. These features reduce the load on the host controller [70].

Figure 24 Elements of CC2520 Development Kits in the laboratory

The nodes were programmed with dedicated software written in C language. The Eclipse IDE [14] and mspgcc compiler [15] have been chosen as components of the development environment. This choice was dictated by the free type of the software license and support for both Windows and Linux operating systems. In addition, only this compiler is not tied to a specific IDE and is Open Source software [71]. The other considered IDEs were Code Composer Studio [72] and IAR Workbench [73]. The laboratory network nodes are presented in Figure 24.

In order to simplify the programming work, it have been decided to use additional libraries. Apart from the standard libraries included with the compiler the following libraries have been chosen:

- Vlo_rand [74] library provides a random number generator that is used for forcing the occurrence of events.
- HAL [75] library provides a partial implementation of the MAC layer in IEEE 802.15.4 protocol. It contains a number of features to facilitate the use of external interfaces and programmable devices for sending and receiving messages. However, it has some limitations, for instance: no retransmissions, the assumption about equality of all nodes, lack of association of the nodes or lack of service of beacon frames. Therefore, all these functions

were written from scratch. Moreover, the library was prepared for working with IAR Workbench IDE so its code was refactored.

In the created application, in addition to the Vlo_rand and HAL libraries, eight new modules have been distinguished (see Figure 25):

- *Configuration* module allows to set node parameters by the user, including node's role in network, the type of malicious behaviour and the related parameters, if it is a malicious node, and other communication parameters.
- *Attacks* module executes the malicious actions in accordance with the chosen role of the node in the network.
- *Transmission* module receives, sends and resends messages.
- *Synchronization* module ensures synchronization between nodes and node wakeups when scheduled.
- *Trust monitor* module calculate trust values and maintains the trust table. It also decides, if and when other nodes should be cut-off because of too low trust value.
- *Messages evaluation* module checks the incoming messages against the assumed network policies.
- *Attacks detection* module discovers malicious actions carried out by other nodes.
- *Information presentation* modules displays the node status on LCD screen.

The architecture of the resulting software modules and libraries is presented in Figure 25. Altogether they form a software package that has been installed on every node of the laboratory network. The resulting package size as *elf* file has 1142KB and the package transferred to hexadecimal form and installed on nodes has 94KB.

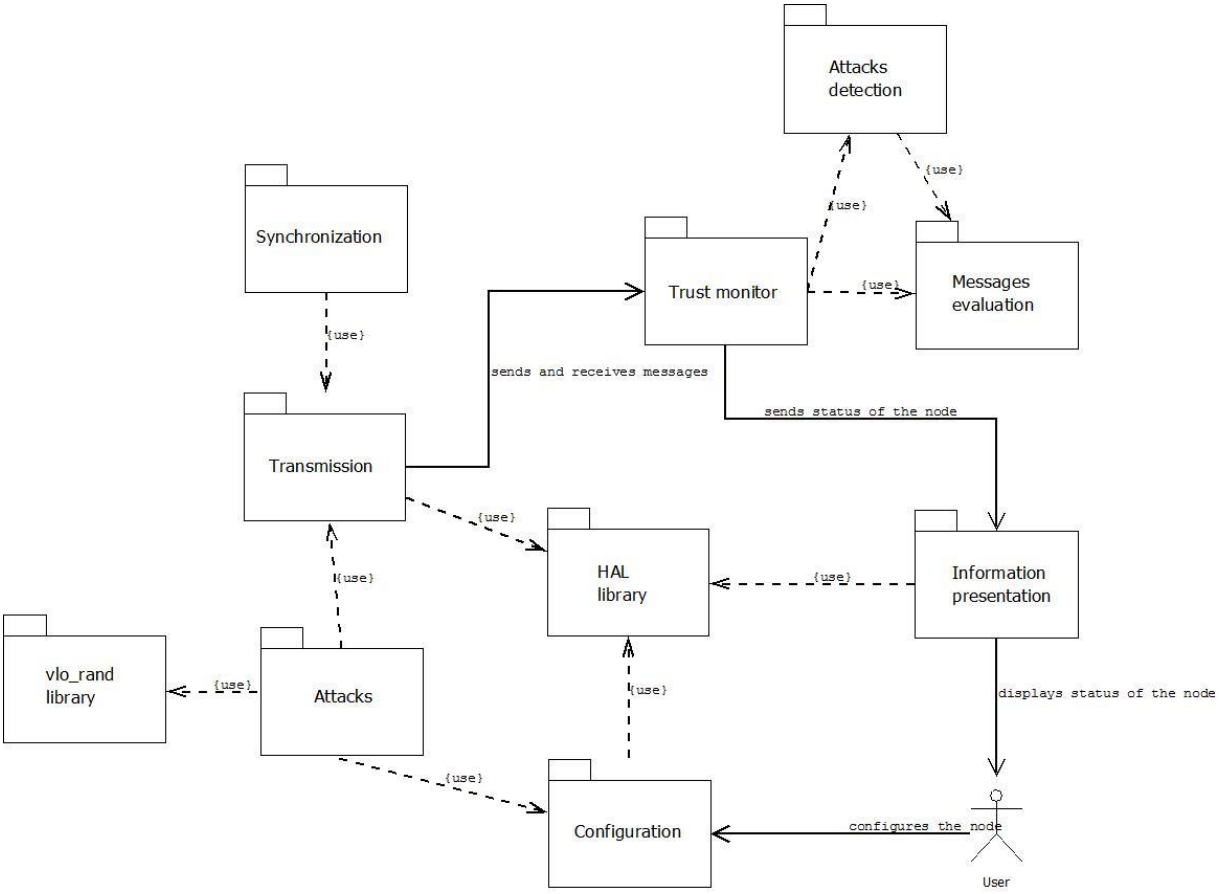

Figure 25 The software architecture of the laboratory network

To use the laboratory network, the user needs to power on the nodes used in the experiment and then using the joystick available in SmartRF05EB, choose the desired options. Every choice is approved by the button marked 'BUTTON 1' on SmartRF05EB. The configuration procedure involves three steps:

1. Choosing the node's position (this allows to distribute nodes in the laboratory without any restrictions setting the desired network structure without measuring the distances);
2. Choosing the role in the network (is it a regular node or an attacker);
3. Selecting parameters of the related malicious action (if the attacker role has been assigned to the node).

The devices can be configured in any sequence, only the base station need to be configured as the last one, because just after its configuration is done, the network synchronization process starts.

While operating, every node displays on its LCD the last part of its network address, the trust values from its trust table and some additional information (e.g. 'OF' when the node is cut-off). The example information displayed by a node during its operation is presented in Figure 26. A node does not store

any information in its memory, so in the current version of the software it is not possible to carry experiments without a human observer, who notices the relevant events, measures time and make notes. An experiment is ended when the network reaches the destination state of the experiment. (e.g. each malicious node is cut-off). Stopping the network needs that all nodes are turned off manually.

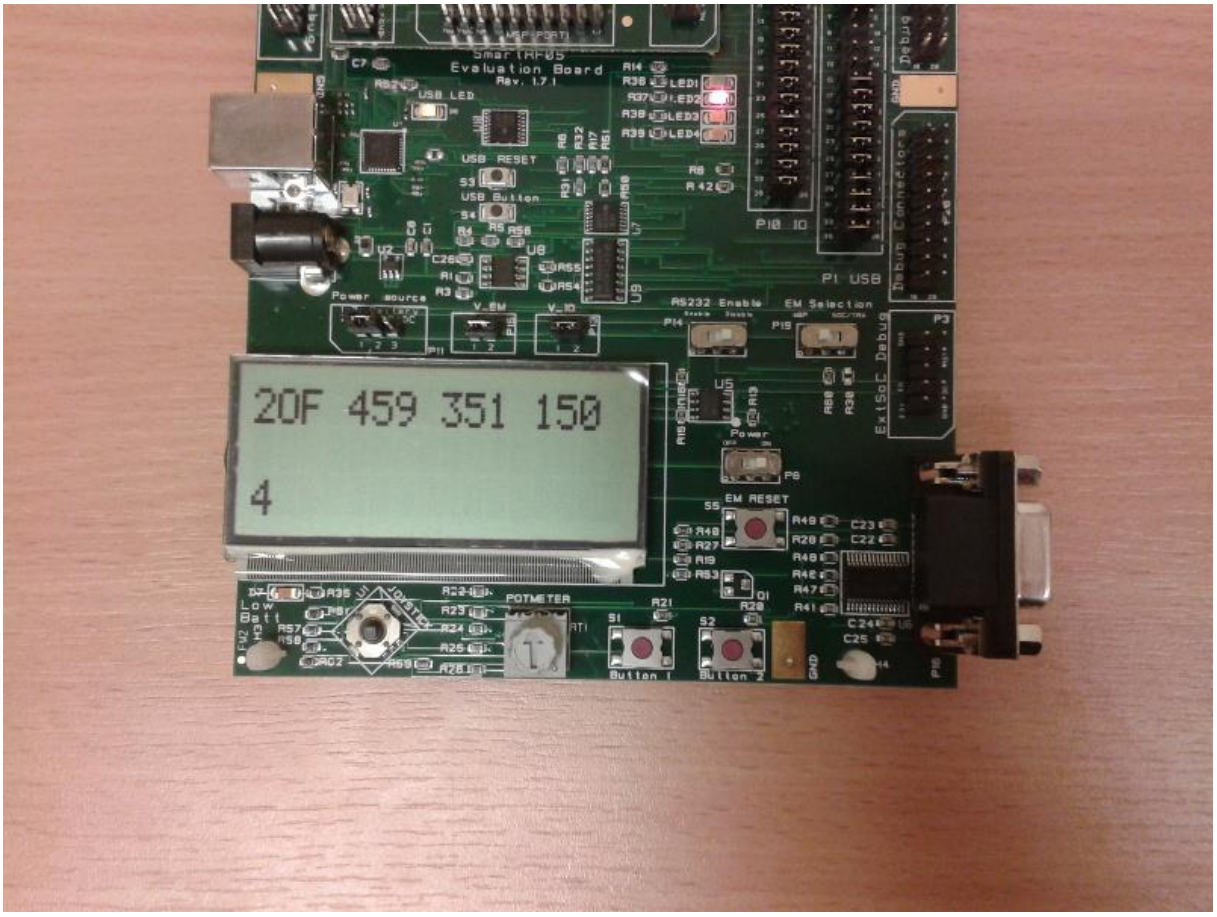

Figure 26 The LCD display of a node of the laboratory network

6.2 The simulator

6.2.1 Existing WSN simulators

It is essential to find a way to test theoretical assumptions and results. The best solution would be to do much testing in real networks, but such way is costly or even impossible if nodes are scattered on a large area, because it is hard to find them and program with new firmware. This is the reason why WSN simulators are commonly applied. Some of them are general enough to simulate many types of WSN, but often it is difficult to adjust them to a specialised application. In many cases availability of these simulators is also limited.

In this section an overview of a selection of commonly used WSN simulators is presented.

6.2.1.1 OMNeT++

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators [76]. It allows to builds networks that include wired and wireless communication networks, on-chip networks or queuing networks. OMNeT++ offers an Eclipse-based IDE, a graphical runtime environment, and a host of other tools. There are available extensions for real-time simulation, network emulation, alternative programming languages (Java, C#), database integration, SystemC integration, and several other functions. It also supports domain-specific functionality such as support for sensor networks, wireless ad-hoc networks, Internet protocols, performance modelling, photonic networks, etc. It is done by model frameworks, developed as independent projects. For WSN these can be:

- NesCT allows to simulate TinyOS-based sensor networks with OMNeT++. It translates TinyOS applications written in the NesC language to C++ simulation code. The primary aim is to provide a new simulation environment and speed up development [77].
- PAWiS is an OMNeT++-based simulation framework for the optimization of wireless sensor networks (WSN). It provides functionality to simulate the network nodes with their internal structure as well as the network between the nodes. One main feature is the contemporaneous simulation of the power consumption of every single node [78].
- Castalia is a simulator for Wireless Sensor Networks (WSN), Body Area Networks (BAN) and generally networks of low-power embedded devices. It is based on the OMNeT++ platform and can be used by researchers and developers who want to test their distributed algorithms and/or protocols in realistic wireless channel and radio models, with a realistic node behaviour especially relating to access of the radio. Castalia can also be used to evaluate different platform characteristics for specific applications, since it is highly parametric, and can simulate a wide range of platforms [79].

6.2.1.2 IAR Embedded Workbench

IAR Embedded Workbench is a set of development tools for building and debugging embedded applications using assembler, C and C++. It provides an integrated development environment including a project manager, editor, build tools and debugger. It allows to create source files and projects, build applications and debug them in a simulator or on hardware [80].

6.2.1.3 WSNSim

WSNSim is the simulation framework that attempts to emulate a true wireless environment capable of hosting multiple sensor nodes. It abstracted out the manner in which the nodes are deployed,

allowing users to use their own algorithms for deployment, rather than being restricted to the available set. Each node can be fitted with an engine that adheres to a common interface, but can also use any communication protocol to communicate between the nodes. The framework is equipped with the behaviour of the classic LEACH and HEED clustering protocols in WSN [81].

6.2.1.4 ns-2

ns-2 is a discrete-event, open-source simulator targeted at networking research. It provides support for simulation of TCP, routing, and multicast protocols over wired and wireless networks. NS can be used in the simulation of routing protocols and is heavily used in ad-hoc networking research. However, it has no GUI and modelling is a complex and time-consuming task, as it is needed to learn scripting language, queuing theory and modelling techniques [82] [83]. There are many forks of ns-2, both maintained and unmaintained, but not actively developed.

6.2.1.5 ns-3

ns-3 is newer version of ns-2, written from scratch. It is a discrete-event network simulator, targeted primarily for research and educational use. It is free software and is publicly available for research, development and use. The goal of the ns-3 project is to develop a preferred, open simulation environment for networking research: it should be aligned with the simulation needs of modern networking research and should encourage community contribution, peer review, and validation of the software. The ns-3 simulation core supports research on both IP and non-IP based networks. However, the large majority of its users focuses on wireless/IP simulations which involve models for Wi-Fi, WiMAX, or LTE for layers 1 and 2 and a variety of static or dynamic routing protocols such as OLSR and AODV for IP-based applications [84]. ns-3 is not compatible backwards and it lacks the support for some protocols, including WSN [83].

6.2.1.6 TOSSIM and EmStar

TOSSIM (discrete event) and EmStar (trace driven) are emulators specially designed to simulate WSN. TOSSIM is running on TinyOS and was first developed by UC Berkeley's TinyOS project team. EmStar includes libraries, tools, services and an extension of Linux microkernel. It was first developed by University of California [85] [86].

6.2.1.7 J-Sim

J-Sim (formerly known as JavaSim) is a component-based, compositional simulation environment. It has been built upon the notion of the autonomous component programming model. This simulator is commonly used in physiology and biomedicine areas, but it also can be used in WSN simulation. In addition, J-Sim can simulate real-time processes [87] [86].

6.2.1.8 Atemu

Atemu is an emulator of an AVR processor for WSN. It can be used to run codes on sensor nodes, debug codes and monitor program executions. It can support users to run TinyOS on MICA2 hardware. Atemu can emulate not only the communication among the sensors, but also every instruction implemented in each sensor [88] [86].

6.2.1.9 Avrora

Avrora is a research project of the UCLA Compilers Group. It is a set of simulation and analysis tools created especially for WSN for programs written for the AVR microcontroller produced by Atmel and the Mica2 sensor nodes. Avrora contains a flexible framework for simulating and analysing assembly programs, providing a Java API and infrastructure for experimentation, profiling, and analysis [89] [86].

6.2.1.10 NetSim

NetSim is a commercial stochastic discrete event simulator usually used by universities for research and in student laboratories. It is actively developed by Tetcos, in association with Indian Institute of Science. It can simulate networks using many technologies, e.g. Wireless Sensor Networks, Wireless LAN, Wi Max, TCP or IP [90] [91].

6.2.1.11 Riverbed Modeler

Riverbed Modeler (former OPNET) is another commercial discrete event simulation engine for analysing and designing communication networks. It can model many network types and technologies e.g. WSN, VoIP, TCP, OSPFv3, MPLS, IPv6. It can also analyse networks to compare the impact of different technology designs on end-to-end behaviour [92].

6.2.1.12 Assessment of the presented simulators

The summary of the presented simulators is given in Table 2. Availability specifies how easy it is to install and start to use a given simulator. Adequacy specifies how adequate a given simulator is to simulate the proposed WCT2M mechanism.

Table 2 Wireless networks simulators comparison

Simulator / criteria	Software type	GUI	Visualisation	Language	License	Extensible	Availability	Adequacy
OMNeT++	Simulation library and framework	Yes	Yes	C++, NED	Academic Public License	Yes	Very good	Medium
IAR	Project	Yes	Yes	C, C++	Paid	No (there is	Good	Medium

Embedded Workbench	manager, editor, build tools, debugger					Eclipse IDE plugin only)		
WSNSim	Simulator	Yes	Yes	Java	Apache License 2.0	No	Very poor	Good
ns-2	Simulator	No	Yes	C++, OTcl	GNU GPL	Mannasim framework available	Medium	Good
ns-3	Simulator	Yes	Yes	C++, Python	GNU GPLv2	No	Medium	Good
TOSSIM / EmStar	Emulator	Yes	No (external tools availabl e)	nesC, Python, C++	Unknown	No	Poor	Poor
J-Sim	Simulator	Yes	No	Java, Tcl	Own	Yes	Medium	Medium
Atemu	Emulator, simulator and debugger	Yes	No	C	No restrictions	No	Very poor	Medium
Avrora	Simulation and analysis tools	No	No	New language created for the project	Own	No	Poor	Medium
NetSim	Simulator	Yes	Yes	C	Paid	No	Good	Medium
Riverbed Modeller	Simulator	Yes	Yes	C, C++	Paid	No	Good	Medium

It is also possible to use hybrid simulation environment, for instance in ANGEL project, a partner, the University of Verona used ns-2 and SystemC. Another partner, the Technische Universitaet Berlin used OMNeT++ with IEEE 802.15.4 library (own model developed) [93].

The decision about selecting a simulator for the purpose of this work was made taking into consideration all attributes listed in Table 2. The summary shows that most of the compared simulators that are obtainable on a free license are difficult to set up – they usually have vestigial or stale documentation. In this group only OMNeT++ has detailed documentation and tutorials

available. Unfortunately, it is described as good for general purpose simulations, not accurate and easy choice for specific ones. Moreover, C or C++ programming skills are needed to use many of the WSN simulators (the author of this report is the Java programmer).

Taking these facts in consideration, it was decided to create a new simulator dedicated to WCT2M evaluation.

6.2.2 Introduction to WCTMS simulator

The laboratory environment described in Section 6.1 allows to validate WCT2M only in a small scale. To analyse the method performance in larger networks, a dedicated simulator was developed, called *WSN Cooperative Trust Management Simulator* (WCTMS).

Let N denotes the set of nodes of a simulated network NET and T denotes the set of values assumed by the trust tables stored in the network nodes. Let TT is a set of all possible trust tables of NET, where each $tt \in TT$ is defined as follows:

$tt: N \rightarrow T$

Let tt_n where $n \epsilon N$ denotes a trust table stored in the node n.

Then *state* of NET is defined as follows:

$S = \{\langle n, tt_n \rangle : n \in N\}$

and SS denotes the set of all possible states of NET.

Let $S_0 \in SS$ be the initial state of NET. WCTMS works in *simulation cycles*. Each simulation cycle results in changing the state of NET (modifying the trust tables stored in the nodes of NET).

Let us assume that seq=c_0, c_1,..., c_m is a sequence of simulation cycles of NET, bringing NET from its initial state S_0 to the final state S_m. Then

$DIST = m$

is called *the simulation distance* between S_0 and S_m in the sequence seq.

Let us consider a set SEQ of k simulation sequences, each bringing the network from S_0 to $S_{terminal}$ where $S_{terminal}$ is a network state that meet the criteria of a simulation termination. For each simulation sequence $seq_i \epsilon SEQ$, i=1,…,k, DIST(seq_i) denotes the simulation distance between S_0 and $S_{terminal}$ in this sequence. We sort the sequences in SEQ by DIST(seq_i), from the lowest to the highest.

Then *average simulation distance in SEQ* between S_0 and $S_{terminal}$ is defined as follows:

$$AVD = \frac{\sum_{i=1}^{k} DIST(seq_i)}{k}$$

and *median simulation distance in SEQ* between S_0 and $S_{terminal}$ is defined as follows:

$$MVD = \begin{cases} DIST\left(seq_{\frac{k+1}{2}}\right) & if\ k\ is\ odd \\ \dfrac{DIST\left(seq_{\frac{k}{2}}\right) + DIST(seq_{\frac{k+1}{2}})}{2} & if\ k\ is\ even \end{cases}$$

With the assumption that a simulation cycle corresponds to a fixed length time interval in the real network, the average simulation distance and the median simulation distance measure how much time is needed to arrive in $S_{terminal}$ while staring in S_0, provided all nodes are still attempting to communicate with their neighbours. In particular, if $S_{terminal}$ represents the network state with all failed nodes excluded from the network, theses metrics will tell us how long it will take for a given network to detect and isolate all failed nodes.

The idea of simulation distance and average distance is presented in Figure 27.

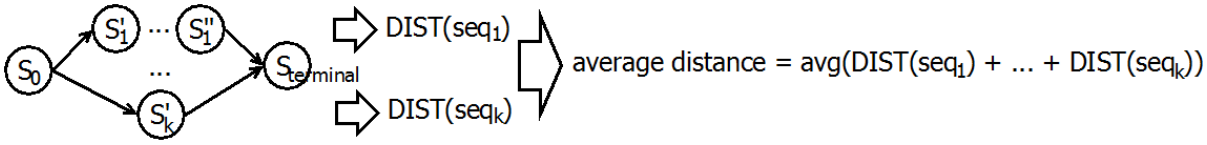

Figure 27 Dependence between simulation distance and average distance

Example 6

The trust table maintained in node 1 of the example network shown in Figure 10 could look in consecutive states S, S' and S'' as presented in Table 3.

Table 3 Example of consecutive states of node 1 in the example network

Node ID	Trust value				
	State S		State S'		State S''
2	0.78		0.81		0.79
3	0.52		0.53		0.54
4	0.93		0.93		0.95
5	0.79		0.75		0.71
6	0.81	->	0.80	->	0.80
7	0.91		0.92		0.91
8	0.87		0.88		0.89

9	0.19		0.11		0.10
10	0.83		0.84		0.86

6.2.3 Design of WCTMS simulator

The simulator was written in NetBeans IDE [16], using Java 1.7 [17] with JGraph library [18] to draw the arrangement of nodes in the simulated space. To simplify the development, WCTMS does not have any GUI – all settings are placed as variables in the code and the results are displayed in the standard output.

The class model of WCTMS is presented in Figure 28.

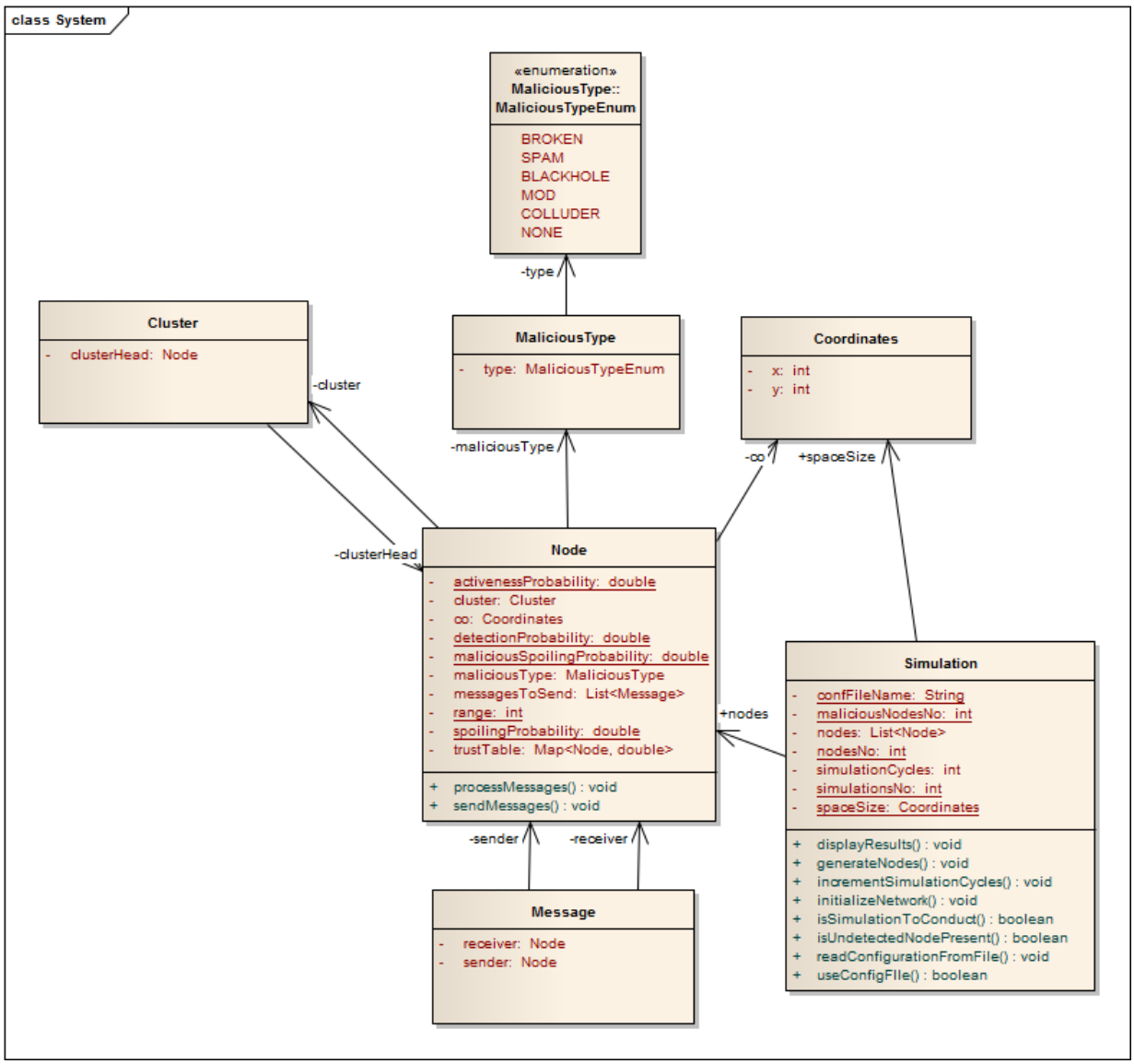

Figure 28 WCTMS class diagram

The model consists of the following classes:

- `Simulation` manages the execution of the set of `simulationsNo` simulations and calculates their length in `simulationCycles` attribute. In `nodes` attribute it stores the list of all nodes in the network. `Simulation` allows to execute the following methods:
    - `displayResults()` displays results of the conducted simulations (e.g. the average and median number of WCT2M cycles needed to detect the first and all malicious nodes) on the standard output;
    - `generateNodes()` generates a set of nodes of the size equal to the `nodesNo` attribute and randomly deploys them on a rectangular simulation space described by coordinates (0; 0) and `spaceSize`. A `maliciouNodesNo` number of nodes are set as being malicious;
    - `incrementSimulationCycles()` increments `simulationCycles` attribute by 1;
    - `initializeNetwork()` creates all objects and set all parameters needed to conduct the simulation and mark `nodes` which cannot communicate with base station (a route from them to the base station cannot be set) as cut-off. It can also stop a simulation and generate a new one if it cannot be executed (e.g. if on the beginning of the simulation all malicious nodes are cut-off from the network or most of the nodes are cut-off);
    - `isSimulationToConduct()` checks if there is another simulation to be executed(to the maximum number of `simulationsNo` attribute);
    - `isUndetectedNodePresent()` checks if in the currently simulated network there is at least one malicious `node` undetected and not cut-off from the network;
    - `readConfiguraionFromFile()` reads configuration from a XML file set under `confFileName` attribute;
    - `useConfigFile()` checks if a configuration file is set under `confFileName` attribute.
- `Cluster` represents the cluster of nodes with the cluster head represented by the `clusterHead` attribute.
- `Message` represents messages exchanged between `sender` and `receiver` (including the base station).
- `Node` represents a node of the network (including the base station). It stores the following attributes (fixed attributes set during a simulation start are underlined):

- activenessProbability – the probability that the node sends a message during a phase of a WCT2M cycle;
- cluster – the cluster the node belongs to;
- co – coordinates of the node;
- detectionProbability – the probability of spoiled message detection by the node;
- maliciousSpoilingProbability – the probability of sending a spoiled message sent by a malicious node;
- maliciousType – the node's MaliciousType;
- messagesToSend – the list of messages to send;
- range – the range of the node;
- spoilingProbability – the probability of sending a spoiled message sent by a regular node;
- trustTable – table where the node stores other nodes trust values.

Node allows to execute the following methods:

- processMessages() – processes the incoming messages in accordance with the own node objectives; messages received by the router node to be forwarded are added to the messagesToSend list;
- sendMessages() – creates new messages and adds them to messagesToSend list, sends messages from messagesToSend list to their receivers.

- Coordinates represents the node's position (x and y) in the space covered by the nodes.
- MaliciousType represents malicious behaviours that can be assumed by a node.
- MaliciousTypeEnum represents the available malicious behaviours.

WCTMS works in accordance with the Simulation Algorithm, which is presented in Figure 29.

```
start simulation
if isSimulationsToConduct()
      if useConfigFile()
            readConfigurationFromFile()
      else
            generateNodes()
      end
      initializeNetwork()
      if isUndetectedNodePresent()
            incrementSimulationCycles()
            for every cluster
                  for every node in the cluster
                        processMessages()
                        sendMessages()
                  end
            end
```

```
            for every cluster
                  for every node in the cluster
                        sendMessages()
                        processMessages()
                  end
            end
      else
            goto start simulation
      end
else
      displayResults()
      exit
end
```

Figure 29 Simulation Algorithm

WCTMS assumes that the nodes are randomly dispersed in the space of the size X by X' units. It is also assumed that each node has a Y units range (calculated as circle of radius = Y around each node). WCTMS analyses a network of ***n*** nodes and one base station. The network topology can be randomly generated or read from a file. The nodes are fixed (i.e. they do not change their position during simulation). Therefore, after initial distribution of nodes, some of them can be too far away from some other nodes or from the base station to communicate with. Such nodes are treated as being cut-off from the beginning of the simulation.

Each message (also containing recommendations) sent by a valid node (including the base station) can be received as spoiled with `spoilingProbability` probability ***e***. The malicious nodes send spoiled messages with `maliciousSpoilingProbability` probability p_s. A spoiled message is detected by the receiving node with `detectionProbability` probability ***r***.

The messages received are either accepted or discarded, depending on the sender's trust value and the local assessment if the message is spoiled by the receiver.

WCTMS assumes that the routing algorithms are in place. The route selection process takes into consideration trust values of the neighbours of a given node. If the trust value of node B stored by node A drops under the cut-off level and B leads on the route to the base station, A will try to find new route to the base station. If all neighbours of A on a way to the base station are distrusted, the node A (and its sub network) is excluded from the whole network. However, if B's route to the base station leads through A and the trust value of node B stored by node A drops under the cut-off level, node B will not be conscious of that fact and it (and its sub network) is excluded from the whole network.

A node can be permanently active which means that during a data phase of the WCT2M cycle it sends one message to the base station (and resends the messages from other nodes, if it is a router node) and in a recommendation phase it sends message with its trust table to its neighbours or can

be randomly active which means that it sends a message in the WCT2M cycle with given `activenessProbability` probability, p_{nd} (data phase) and p_{nr} (recommendation phase) for an ordinary node and p_b for the base station. At the recommendation phase of the WCT2M cycle, the nodes exchange their trust tables with their neighbours and update own trust tables accordingly.

7. Experimental evaluation of WCT2M

7.1 Evaluation plan

7.1.1 Objectives of experiments

The aim of the conducted experiments was to validate WCT2M by checking how effectively and efficiently it can detect and isolate malicious nodes in a WSN network. It is assumed that a malicious node is considered detected when one of its routers (or the base station) considers it as untrustworthy. During experiments different sizes of networks were considered, with different numbers of malicious nodes and different behavioural characteristics of these malicious nodes. The objective was to experimentally verify WCT2M resistance to different malicious actions (as described in Section 4.2).

A set of experiments was conducted using the laboratory network with nodes distribution as illustrated in Figure 10. These experiments were also repeated with the help of WCTMS to check, if the laboratory experiments and the simulations provide similar results.

Then an additional set of simulation experiments was conducted using WCTMS. These experiments were focusing on simulating larger networks which could not be directly implemented in the laboratory because of the limited resources.

Each experiment consists of a set of *test cases* which differ in input parameters. Every test case consists of a number of *simulation runs*. A simulation run starts with the input parameters characterizing its test case and ends when all malicious nodes in the network are detected or the resources devoted to the test are exhausted. The relationship between experiment, test case and simulation run is illustrated in Figure 30.

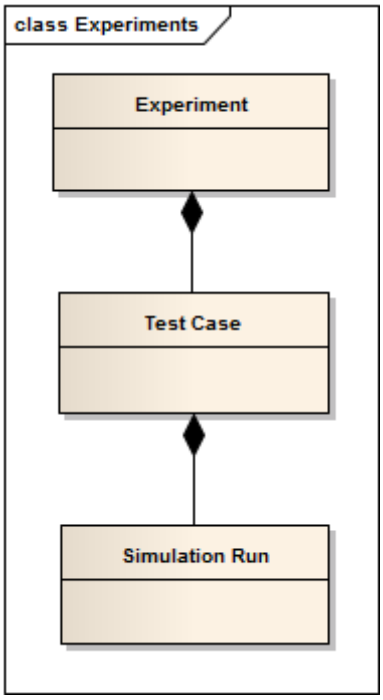

Figure 30 The relationship experiment, test case and simulation run

A set of metrics supporting the analyses was identified using the Goal-Question-Metrics (GQM) methodology [94]. GQM offers systematic approach which allows to obtain a set of metrics supporting an explicitly stated measurement goal. Using GQM, we start with the goal of measurements. Next, there is an intermediate layer of questions linking the goal to the metrics. Answering these questions helps to decide which metrics support the stated goal.

The overall goal of the experiments is defined as follows:

Analyse WCT2M for the purpose of assessment with respect to effectiveness and efficiency of malicious nodes detection.

At the lower level of GQM decomposition, the following questions were identified:

- **Q1**: What is the efficiency of malicious nodes detection?
- **Q2**: What is the effectiveness of malicious nodes detection?
- **Q3**: How precisely are the malicious nodes isolated?

The third level of GQM decomposition involves identification of metrics which are used to answer a particular question. The metrics answering the questions Q1, Q2 and Q3 are given in Table 4.

Table 4 Metrics associated with questions Q1 and Q2

Question	Metric name
Q1	First Node Detected (FND) Average FND (AFND) Median FND (MFND)
	All Nodes Detected (AND) Average AND (AAND) Median AND (MAND) Normalized MAND (NMAND)
	Improvement (Im)
Q2	Detection Quality (DQ) Median DQ (MDQ)
Q3	Cut-off Quality (CQ)

A detailed explanation of the metrics is presented below (in square brackets the domain type of the result delivered by the metric is given).

- **First Node Detected (FND):** the number of WCT2M cycles needed to detect the first malicious node in a given simulation run [Integer].
- **Average FND (AFND):** the average value of FND calculated for all simulation runs in a given test case [Real].
- **Median FND (MFND):** the median value of FND achieved in a test case [Real].
- **All Nodes Detected (AND):** the number of WCT2M cycles needed to detect all malicious nodes in a single simulation run [Integer].
- **Average AND (AAND):** the average value of AND calculated for all simulation runs in a given test case [Real].
- **Median AND (MAND):** the median value of AND achieved in a test case [Real].
- **Normalized MAND (NMAND):** the MAND value divided by the number of all nodes in the simulated network [Real].
- **Improvement (Im):** for two test cases, *new* and *old,* Im measures the changes of the AFND metric and AAND metric in accordance with the following expressions:

  $Im_{AFND} = (1 - AFND_{new}/AFND_{old})x100\%$ [%]

  $Im_{AAND} = (1 - AAND_{new}/AAND_{old})x100\%$ [%].

- **Detection Quality (DQ):** the percentage of all malicious nodes that was detected in a given simulation run [%].
- **Median DQ (MDQ):** the median value of DQ calculated for all simulation runs in a given test case [%].
- **Cut-off Quality (CQ):** for a given test case, where
    - M is number of simulation runs in the test case,
    - N is the number of malicious nodes in the analysed network,
    - *network distance*$_{nm}$ is the network distance from the n-th malicious node to the first node which cut it off in the m-th simulation run,
    - $X_m=\sum_{n=1}^{N} \frac{1}{network\ distance_{mn}}/N$

    the metric is calculated in the following way:

    $CQ = \frac{\sum_{m=1}^{M} Xm}{M}$

    CQ is the average of X_m, m=1,..,M where each X_m characterizes the inverse of the average distance between the malicious nodes and the nodes which detected them in a given simulation run [Real].

The metrics FND, AFND, MFND, AND, AAND, MAND and NMAND refer to the number of WCT2M cycles. Because each simulation cycle corresponds to a fixed time length interval in the simulated network (as described in Section 6.2.2), the simulation distance (expressed as a number of simulation cycles) represents the time distance between the referred events, for instance, the time needed to detect the first malicious node or the time needed to detect all malicious nodes. It allows to compare the results achieved using laboratory network and WCTMS.

Example 7

Let us assume the network illustrated in Figure 10 with two malicious nodes: 2 and 8. Assume that there were 2 test cases: A and B and there were 3 simulation runs in the test case A and:

- During the first run, node 8 was detected in the third WCT2M cycle by node 1 and node 2 was detected in the fifth WCT2M cycle by the base station;
- During the second run, node 8 was detected in the fourth WCT2M cycle by the base station and node 2 was detected in the tenth WCT2M cycle by the base station;
- During the third run, node 8 was detected in the fourth WCT2M cycle by node 1 and node 2 was detected in the ninth WCT2M cycle by base station.

The resulting values of the metrics are presented in Table 5.

Table 5 Metric values example

Metric	Value
AFND	3.67
MFND	4.0
AAND	8.0
MAND	9.0
NMAND	0.9
MDQ	100%
CQ	0.92

If in the test case B the following metric values were achieved: AFND = 3.42 and AAND = 7.21, then Im_{AFND} = 6,81% and Im_{AAND} = 9,88%.

The relationship between the steps of experiments and the metrics is illustrated in Figure 31.

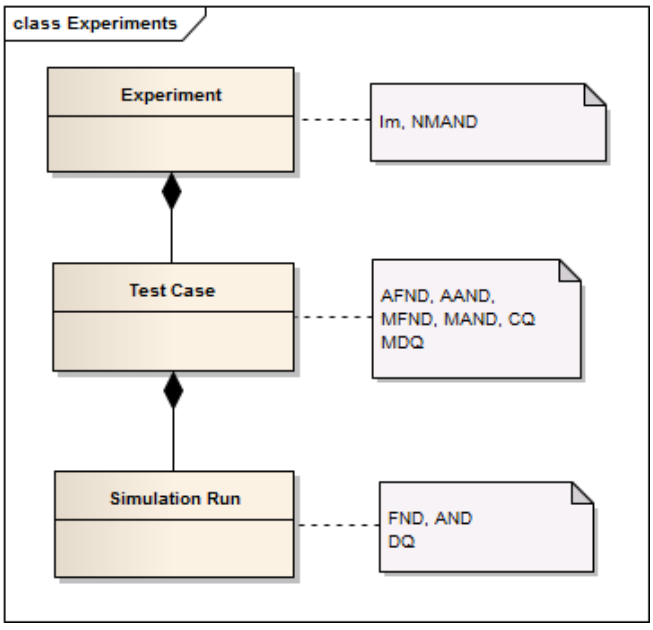

Figure 31 The relationship between the experiments steps and the metrics

To represent different malicious behaviours of nodes, the 'malicious' software modifying the behaviour of a node has been developed to represent the following functionalities: (1) *faulty nodes*, i.e. the nodes sending just damaged messages without performing any particular attack scenario and (2) the nodes performing the types of attack described in Section 4.2.

7.1.2 Scope of experiments

The experiments have been divided into the following groups. While describing these experiments the simulation distance (see Section 6.2.2) is interpreted as the measure of time delay between events related to the network.

EXP1: Feasibility of implementing WCT2M and implementing the attack models: The objective of this experiment was to demonstrate that WCT2M can be implemented in a typical WSN environment and to demonstrate the implementation of faulty nodes as described in Section 4.2. The results of this experiment are described in detail in Section 7.2.1.

EXP2: The time delay in detecting faulty nodes: The objective of this experiment was to measure the time delay needed to detect the faulty nodes in the network. There were two situations considered: (1) the faulty nodes were present from the beginning of the experiment and (2) the faulty nodes were added while the network was operating. Section 7.2.2 presents the results of this experiment.

EXP3: Relationship between nodes' activity and the time delay of detection: The objective of this experiment was to examine if and how the nodes' activity influences the time of detecting the first and all faulty nodes in the network. In the experiment, the nodes' activity was characterized by probabilities p_{nd}, p_{nr} and p_b (see Table 7). Section 7.2.3 gives the results of this experiment.

EXP4: Relationship between number of faulty nodes and the time delay of detection: The objective of this experiment was to examine how the number of faulty nodes in the network impacts the time delay of detecting them. Section 7.2.4 presents the results of this experiment.

EXP5: Resistance to decreased frequency of attack: The objective of this experiment was to examine the resistance of WCT2M to attacks repeated with decreasing frequency. In Section 5.2.3 the action history feature was introduced, which was assumed to counteract this type of attack. Section 7.2.5 presents the results of the experiment examining the effectiveness of this feature.

EXP6: Resistance to collusion attack: The objective of this experiment was to examine the resistance of WCT2M to collusion attack. In Section 5.2.3 the trust history feature was introduced. Section 7.2.6 presents the results of the experiment examining the effectiveness of this feature in detecting the colluding nodes.

EXP7: Influence of effectiveness of security mechanisms: The objective of this experiment was to examine how the effectiveness of security mechanisms implemented in the node impacts the time delay of detecting faulty nodes. Section 7.2.7 presents the results of this experiment.

One of the experiments, EXP1, was conducted using both, the laboratory network and the simulator. Its objective was to demonstrate feasibility of WCT2M implementation in a typical WSN environment and to check if the simulation results are close to the laboratory results. All other experiments were conducted with the help of the simulator (this information is included in Table 6).

Table 6 Method of conducing of the experiment

	EXP1	EXP2	EXP3	EXP4	EXP5	EXP6	EXP7
Laboratory	x						
Simulator	x	x	x	x	x	x	x

The input parameters used in the laboratory and simulation experiments are summed up in Table 7. All parameters listed in the table were already discussed in the previous sections.

Table 7 Input parameters during WCT2M validation experiments

Input Variable	Description	Discussed in Section	Scale
tiers	Number of network tiers	2.3.2	1-MAX(tiers)
n	Number of nodes in the network	6.2.3	1-MAX(n)
N	Number of malicious nodes in the network	6.2.3	0-n
CF	Cooperation factor	5.2.4	0-1
$change_{positive}$	Positive change of trust value factor	5.2.4	0-1
$change_{negative}$	Negative change of trust value factor	5.2.4	0-1
p_{nd}	Probability of sending a message by a node in the data phase of a WCT2M cycle (*nodes activity*)	6.2.3	0–100%
p_{nr}	Probability of sending the trust table to the neighbours in a recommendation phase of the WCT2M cycle.	6.2.3	0–100%
p_b	Probability of sending a message by the base station in the data phase of a WCT2M cycle (*base station activity*)	6.2.3	0–100%
p_s	Probability of spoiling a message by a malicious node	6.2.3	0–100%

p_e	Probability of sending a spam message by a malicious node	4.2.1	0–100%
p_d	Probability of blocking a message by a malicious node	4.2.2	0–100%
p_{ch}	Probability of modifying a message by a malicious node	4.2.3	0–100%
p_t	Probability of modifying a trust table by a malicious node	5.2.5	0-100%
r	Probability of recognizing a spoiled message by the receiver node	6.2.3	0–100%
e	Probability of spoiling a message during transmission	6.2.3	0–100%
t_T	Duration of the sleep period	2.4.1	[ms]
t_A	Duration of the wakeup period	2.4.1	[ms]
l_i	Initial trust level	5.2.2	0–100
l_c	Cut-off level	5.2.2	0–100
h_a	Size of the ActionHistoryTable	5.2.3	0–MAX(h_a)
h_{at1}	Action rating threshold level 1	5.2.3	0– MAX(h_{at1})
h_{at2}	Action rating threshold level 2	5.2.3	0– MAX(h_{at2})
h_t	Size of the TrustHistoryTable	5.2.3	0– MAX(h_t)
h_{tt}	Trust rating threshold	5.2.3	0– MAX(h_{tt})

The above table presents a group of probabilistic parameters connected with nodes' malicious actions: p_s, p_e, p_d, p_{ch} and p_t and parameter e connected with probability of spoiling a message during transmission. If these parameters are applied, depends on node configuration. The algorithm describing how these parameters are used is presented in Figure 32.

```
if n forwards m
      if n is a malicious node
            if n performs blackhole attack
                  block sending of m with probability pd
            else if n performs message modification attack
                  spoil m with probability pch
            else if m performs collusion attack
                  spoil m with probability ps
            else if m does not perform any attack
                  spoil m with probability ps
            end
      end
else if n sends m to the base station
      if n is a malicious node
            if n performs spam attack
                  multiply m with probability pe
            else if n performs collusion attack
```

```
                  spoil m with probability ps
            else if n does not perform any attack
                  spoil m with probability ps
            end
      end
else if n sends trust table
      if n is a malicious node
            if n performs collusion attack
                  spoil m with probability pt
            if n does not perform any attack
                  spoil m with probability ps
            end
      end
end
spoil message with probability e
```

Figure 32 Use of p_s, p_e, p_d, p_{ch} p_t and e parameters while sending a message m from node n

The values of the input parameters used in the experiments are summed up in Table 8. The values listed in each entry of the table indicate that a given experiment was conducted with these listed values. The parameter values were selected based on literature [95] [96] [97] and the observed laboratory network behaviour in particular related to the changes of achieved results after the input parameter change. During the experiments, it was assumed that input parameters do not change their values during a given simulation run.

Table 8 Input parameters used for experiments

Input Variable	**EXP1**	**EXP2**	**EXP3**	**EXP4**	**EXP5**	**EXP6**	**EXP7**
tiers	2	2	2	1, 2	2	2	2
n	10	10	10	10, 20, 50, 100, 150, 200, 300, 1000	10	20, 50, 100, 150, 200, 300	20, 50, 100, 150, 200, 300
N	1, 2	1, 2, 3	1, 2	1-10	1, 2	1-n/2	1-10
CF	0.2	0.2	0.2	0.2	0.2	0.2	0.2
***change*$_{positive}$**	0.05	0.05	0.05	0.05	0.05	0.05	0.05
***change*$_{negative}$**	0.2	0.2	0.2	0.2	0.2	0.2	0.2
p_{nd}	100%	100%	50%, 60%, 70%, 80%,	80%, 100%	100%	100%	100%

			90%, 100%				
p_{nr}	100%	100%	50%, 60%, 70%, 80%, 90%, 100%	80%, 100%	100%	100%	100%
p_b	100%	100%	50%, 60%, 70%, 80%, 90%, 100%	80%, 100%	100%	100%	100%
p_s	70%	70%	70%	70%	70%	70%	70%
p_e	40%, 60%, 80%, 100%	-	-	-	40%, 60%, 80%, 100%	-	-
p_d	50%, 80%, 100%	-	-	-	50%, 80%, 100%	-	-
p_{ch}	50%, 80%, 100%	-	-	-	50%, 80%, 100%	-	-
p_t	-	-	-	-	-	50%, 80%, 100%	-
r	100%	90%	90%	90%	100%	90%	50%, 60%, 70%, 80%, 90%, 100%

e	0%	2%	2%	2%	0%	2%	2%
t_T	7500ms	7500ms	7500ms	7500ms	7500ms	7500ms	7500ms
t_A	100ms	100ms	100ms	100ms	100ms	100ms	100ms
l_i	50	50	50	50	50	50	50
l_c	10	10	10	10	10	10	10
h_a	0	0	0	0	10	0	0
h_{at1}	0	0	0	0	3	0	0
h_{at2}	0	0	0	0	5	0	0
h_t	0	0	0	0	0	10	0
h_{tt}	0	0	0	0	0	5	0

The detailed description of the experiments is given below.

7.2 The experiments

7.2.1 EXP1: Feasibility of implementing WCT2M and implementing the attack models

7.2.1.1 Description

The software package implementing WCT2M has been developed and installed on the nodes of the laboratory network described in Section 6.1. Then, experiment EXP1 including a set of test cases was prepared. The test cases are summed up in Table 9 - Table 12.

Table 9 Parameters of spam attack scenario in experiment EXP1

Case	Spam probability (p_e)
S1	100%
S2	80%
S3	60%
S4	40%

Table 10 Parameters of blackhole attack scenario in experiment EXP1

Case	Message blockage probability (p_d)
B1	100%
B2	80%
B3	50%

Table 11 Parameters of message modification attack scenario in experiment EXP1

Case	Probability of message modification (p_{ch})
M1	100%
M2	80%
M3	50%

Table 12 Parameters of faulty nodes scenario in experiment EXP1

Case	Number of faulty nodes	Probability of message damage (p_s)
F1	1	70%
F2	1	70%
F3	2	70%

In test cases S1-S4, B1-B3 and M1-M3 always the node labelled '1' in Figure 10 (the cluster head) was faulty. Test cases F1-F3 have different faulty nodes deployment, as presented in Table 13.

Table 13 Nodes deployment for experiment EXP1

Case	Faulty nodes deployment
F1	the node labelled '1' in Figure 10 is faulty (the cluster head)
F2	the node labelled '2' in Figure 10 is faulty (the leaf node)
F3	the nodes labelled '1' and '2' in Figure 10 are faulty (the cluster head and the leaf node)

In spam attack (Table 9), blackhole attack (Table 10) and message modification attack (Table 11) there was only one malicious node (labelled '1' in Figure 10) which behaved in different ways depending on the attack type. Table 14 presents, how this malicious node acted in actions related to sending a message during each attack.

Table 14 Malicious node working mode in the attack scenarios of experiment EXP1

	Sending a message to the base station	**Forwarding a message (to the base station or to any leaf)**	**Sending a trust table**
Spam attack	Sends a message and next up to 3 series of 4 messages. Messages are valid.	Works as a non-malicious node	Works as a non-malicious node
Blackhole attack	Works as a non-malicious node	Does not forward messages	Works as a non-malicious node
Message modification attack	Works as a non-malicious node	Modifies messages (so they can be recognized as invalid)	Works as a non-malicious node

All test cases of EXP1 were conducted in the laboratory on the devices creating the example network presented in Figure 10.

Then, each test case was also repeated with the use of WCTMS, using the same parameters and layout of nodes.

For each test case, 10 runs were conducted in the laboratory and 100 simulation runs were conducted with the help of WCTMS. After each run, the AFND, MFND, AAND, MAND, MDQ and CQ metrics were calculated.

7.2.1.2 Results

The results achieved in the laboratory and the results achieved using WCTMS are summed up in Table 15 - Table 18. For spam attack, blackhole attack and message modification attack the value of AFND is equal AAND and the value of MAND is equal MFND because there is just one malicious node in the network (therefore these metrics are presented in the same column).

Table 15 EXP1 results for spam attack

Case	p_e	laboratory				simulator			
		MFND MAND	AFND AAND	CQ	MDQ	MFND MAND	AFND AAND	CQ	MDQ
S1	100%	1.0	1.0	1.0	100%	1.0	1.0	1.0	100%
S2	80%	1.0	1.5	1.0	100%	1.0	1.7	1.0	100%
S3	60%	2.0	2.3	1.0	100%	2.0	2.8	1.0	100%
S4	40%	3.5	3.75	1.0	100%	5.0	8.32	1.0	100%

Table 16 EXP1 results for blackhole attack

Case	p_d	laboratory				simulator			
		MFND MAND	AFND AAND	CQ	MDQ	MFND MAND	AFND AAND	CQ	MDQ
B1	100%	6.0	6.0	1.0	100%	6.0	6.0	1.0	100%
B2	80%	Interrupted after 30 WCT2M cycles			0%	92.0	134.41	1.0	100%
B3	70%	Not executed				4226.0	5588.73	1.0	100%

Table 17 EXP1 results for message modification attack

Case	p_{ch}	laboratory				simulator			
		MFND MAND	AFND AAND	CQ	MDQ	MFND MAND	AFND AAND	CQ	MDQ
M1	100%	2.0	2.0	1.0	100%	2.0	2.0	1.0	100%
M2	80%	3.5	3.6	1.0	100%	3.0	3.18	1.0	100%
M3	50%	7.0	7.0	1.0	100%	9.5	12.11	1.0	100%

Table 18 EXP1 results for faulty nodes

Case	p_{ch}	laboratory						simulator					
		MFND	AFND	MAND	AAND	CQ	MDQ	MFND	AFND	MAND	AAND	CQ	MDQ
F1	70%	3.0	3.2	3.0	3.2	1.0	100%	2.0	2.5	2.0	2.5	1.0	100%
F2	70%	5.0	5.2	5.0	5.2	1.0	100%	5.0	5.58	5.0	5.58	1.0	100%
F3	70%	2.0	2.8	7.0	7.2	1.0	100%	2.0	2.3	7.0	7.3	1.0	100%

The experiment demonstrated, that WCT2M can be implemented on real-world devices and is feasible in a typical WSN environment.

Most of the results achieved with the help of WCTMS are close to these achieved in the laboratory. After examining the differences, it turned out, that smaller attack probability causes the bigger differences of achieved results. It can be connected with differences in implementations of pseudorandom number generators. To check if this differences decrease with greater number of experiment executions, 10 more results were achieved in the laboratory for case F1 for faulty nodes scenarios. This case was chosen because the difference between laboratory and simulation results equals 50%. After increasing the number of experiment executions in the laboratory, the results tended to converge, as presented in Table 19. It is especially visible for MFND and MAND which are less influenced by outliers.

Table 19 EXP1 results for faulty nodes scenarios with F1 case executed 20 times

Case	p_{ch}	laboratory				simulator			
		MFND	AFND	MAND	AAND	MFND	AFND	MAND	AAND
F1	70%	2.0	2.7	2.0	2.7	2.0	2.5	2.0	2.5

7.2.2 EXP2: The time delay in detecting faulty nodes

7.2.2.1 Description

Experiment EXP2 was conducted with respect to the network presented in Figure 10 and the case study introduced in Section 3.1. Two different scenarios were considered:

- **S1:** new faulty nodes are inserted to already working network;
- **S2:** some nodes in the already working network become faulty.

The scenarios were realized by implementing the following settings:

- all non-faulty nodes are considered to be fully trustworthy (initial trust I_i=100);
- in scenario S1, faulty nodes start with initial trust I_i=50**;**
- in scenario S2, faulty nodes start with initial trust I_i=100.

For each scenario, the three test cases were distinguished. Cases C1-C3 have different faulty nodes deployment presented in Table 20.

Table 20 Nodes deployment for experiment EXP2

Case	Faulty nodes deployment
C1	the node labelled '2' in Figure 10 is faulty (a leaf node)
C2	the nodes labelled '1' and '6' in Figure 10 are faulty (a cluster head and a leaf node from another cluster)
C3	the nodes labelled '1', '6' and '9' in Figure 10 are faulty (a cluster head and leaf nodes from other clusters)

The input parameters used in the simulation are presented in Table 7.

For every test case of each scenario, 100 simulation runs were conducted with the help of WCTMS. For each test case AFND, AAND, MDQ and CQ metrics were calculated.

7.2.2.2 Results

The results of the experiment are presented in Table 21.

Table 21 EXP2 results

Scenario - case	AFND	AAND	CQ	MDQ
S1-C1	5.0	5.0	1.0	100%
S1-C2	1.0	6.0	1.0	100%
S1-C3	2.0	6.0	0.92	100%
S2-C1	12.0	12.0	1.0	100%
S2-C2	2.0	16.0	1.0	100%
S2-C3	3.0	13.0	0.9	100%

The experiment shows that the faulty nodes cumulated in one cluster (case C3) slightly influence the quality of trust management detection – not all nodes were detected by their nearest neighbours (in router hops). The experiment also shows that a faulty cluster head can be detected quickly, because it sends (forwards) multiple messages during each WCT2M cycle. The experiment also demonstrated that faulty nodes just inserted to the network can be detected about two times faster than nodes which were properly working and then became faulty.

7.2.3 EXP3: Relationship between nodes' activity and the time delay of detection

7.2.3.1 Description

To learn about WCT2M time effectiveness in detecting and isolating faulty nodes differently distributed in the network topology in networks bigger than it was possible to construct in the laboratory, experiment EXP3 with three test cases was conducted with respect to the network presented in Figure 10. The test cases have different faulty nodes deployment presented in Table 22.

Table 22 Nodes deployment for experiment EXP3

Case	Experiment characteristics
C1	the node labelled '1' in Figure 10 is faulty (a router node)
C2	the node labelled '2' in Figure 10 is faulty (a leaf node)
C3	the nodes labelled '1' and '2' in Figure 10 are faulty (a router node and a leaf node)

During each test case, different combinations of values of nodes activity (p_{nd}, p_{nr}) and base station activity p_b were assumed (see Table 8). In each test case it was assumed that p_{nd} equals p_{nr}.

For every test case, 100 simulation runs were conducted with the help of WCTMS. For each case MAND metric was calculated.

7.2.3.2 Results

The simulation results - MAND values - are presented in Table 23 - Table 25. The columns present results for given base station activity p_b [%] and rows present results for given nodes activity p_{nd}, p_{nr} [%].

Table 23 EXP3-C1 results

		Base station activity p_b[%]					
		100	**90**	**80**	**70**	**60**	**50**
Nodes activity p_{nd}, p_{nr} [%]	**100**	2	2	2	2	2	2
	90	2	2	2	2	3	3
	80	3	3	3	3	3	3
	70	3	3	3	3	3	3
	60	3	3	3	4	4	4
	50	4	4	4	4	4	4

Table 24 EXP3-C2 results

		Base station activity p_b[%]					
		100	**90**	**80**	**70**	**60**	**50**
Nodes activity p_{nd}, p_{nr} [%]	**100**	10	10,5	11	11	11	11
	90	11	12	12	12	12	12
	80	12	12	12	12	12	12
	70	14	14	14	15	15	16
	60	16	16	17	17	17	18
	50	19	20	20	20	20	20

Table 25 EXP3-C3 results

		Nodes activity p_{nd}, p_{nr} [%]					
		100	**90**	**80**	**70**	**60**	**50**
Nodes activity p_{nd}, p_{nr} [%]	**100**	11	9,5	10	10	11	11
	90	11	11	11	11	11	11,5
	80	12	12	13	13	13	13
	70	13	14	14	15	15	15
	60	17	17	17	17	17	17
	50	20	20	20	20	20	21

The experiment showed that WCT2M mechanism can detect and cut off a faulty node in a relatively short time (average time delay equals 3 in situation where the nodes are highly active and a router node fails). The detection time increases inversely to the nodes activity and for leaf nodes is longer than for router nodes.

The experiment also demonstrated that the time needed to detect and isolate a single faulty node depends mainly on the nodes activity and is less dependent on base station activity. Nevertheless, the results demonstrated that base station activity increases its influence if there are multiple faulty nodes in the network.

7.2.4 EXP4: Relationship between number of faulty nodes and the time delay of detection

7.2.4.1 Description

Experiment EXP4 with a set of test cases was prepared to assess effectiveness of WCT2M depending on the number of the faulty nodes in the network. While conducting these test cases, the nodes were assumed to be distributed in the rectangle of the size X×X' and X = X' = 100 points (a point is a distance unit). There were considered two different distribution spaces:

- the distribution space was unstructured (one-tier network);
- the distribution space was divided into 16 clusters of size Y×Y' each, where Y = Y' = 25 points (two-tier network).

The base station was placed at the point S (102, 102), outside the nodes distribution area. The node signal range was set to Z = 30 points. All nodes were fixed (they could not change their position during a given simulation). Example of the distribution space for one-tier and two-tier networks is shown in Figure 3.

The experiment test cases are characterized in Table 26.

Table 26 Simulation parameter values for experiment EXP4

Case	Number of tiers	p_{nd}	p_{nr}	p_b
C1	1	100%	100%	100%
C2	2	100%	100%	100%
C3	2	80%	80%	80%

During simulations, networks of size n = 10, 20, 50, 100, 150, 200, 300, 1000 nodes were considered and for each network of size n, the simulations were performed for different number of faulty nodes N, where $N \in [1..10]$. For each network characterized by the numbers n and N, 100 simulation runs were performed. For each test case, metrics MFND, MAND, NMAND, MDQ and CQ were calculated.

7.2.4.2 Results

The results of experiment EXP4 for the networks of 20, 100, 300 and 1000 nodes are shown in Figure 33 - Figure 40.

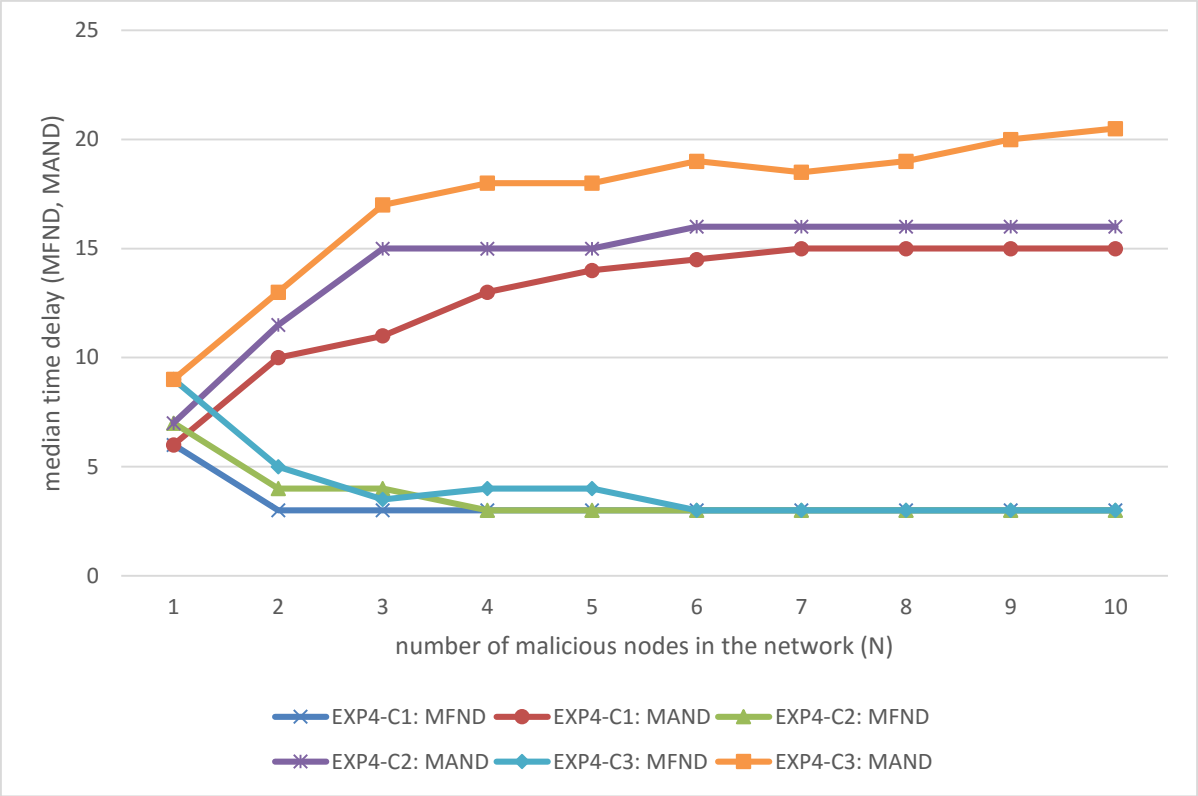

Figure 33 EXP4 results: median time delay of detecting first and all faulty nodes for n=20

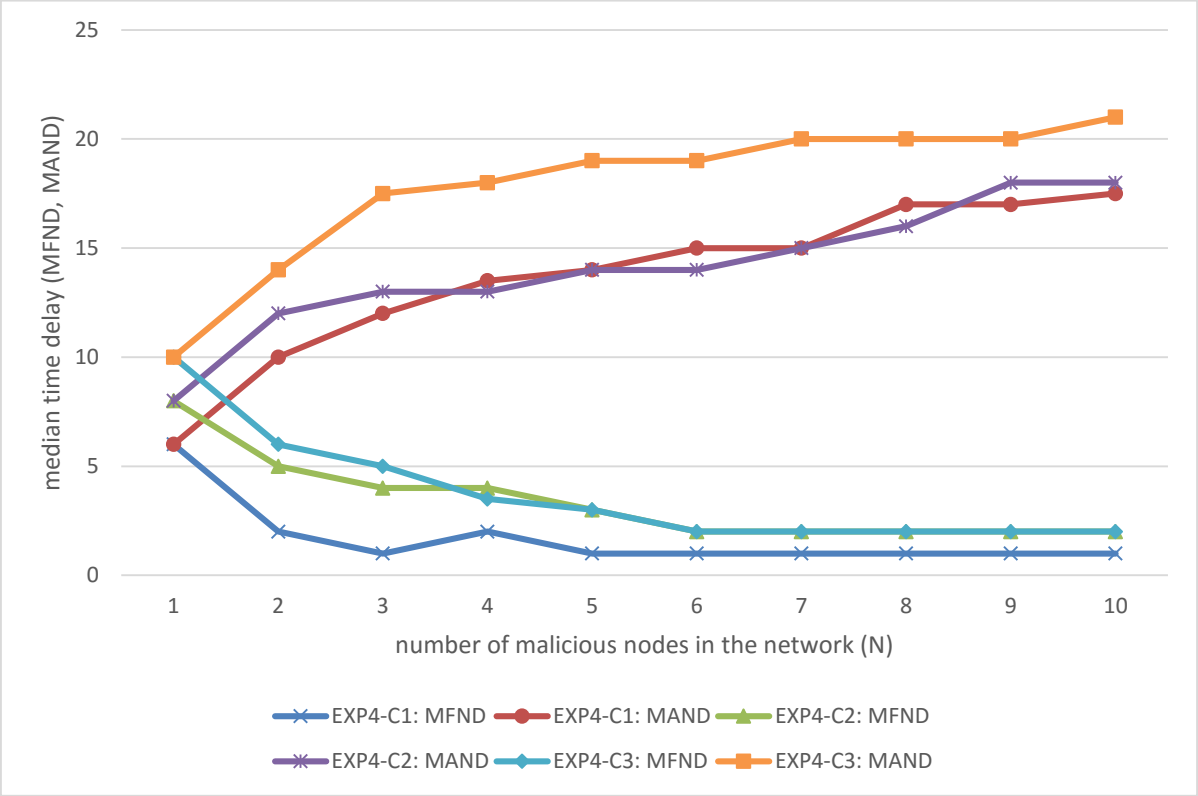

Figure 34 EXP4 results: median time delay of detecting first and all faulty nodes for n=100

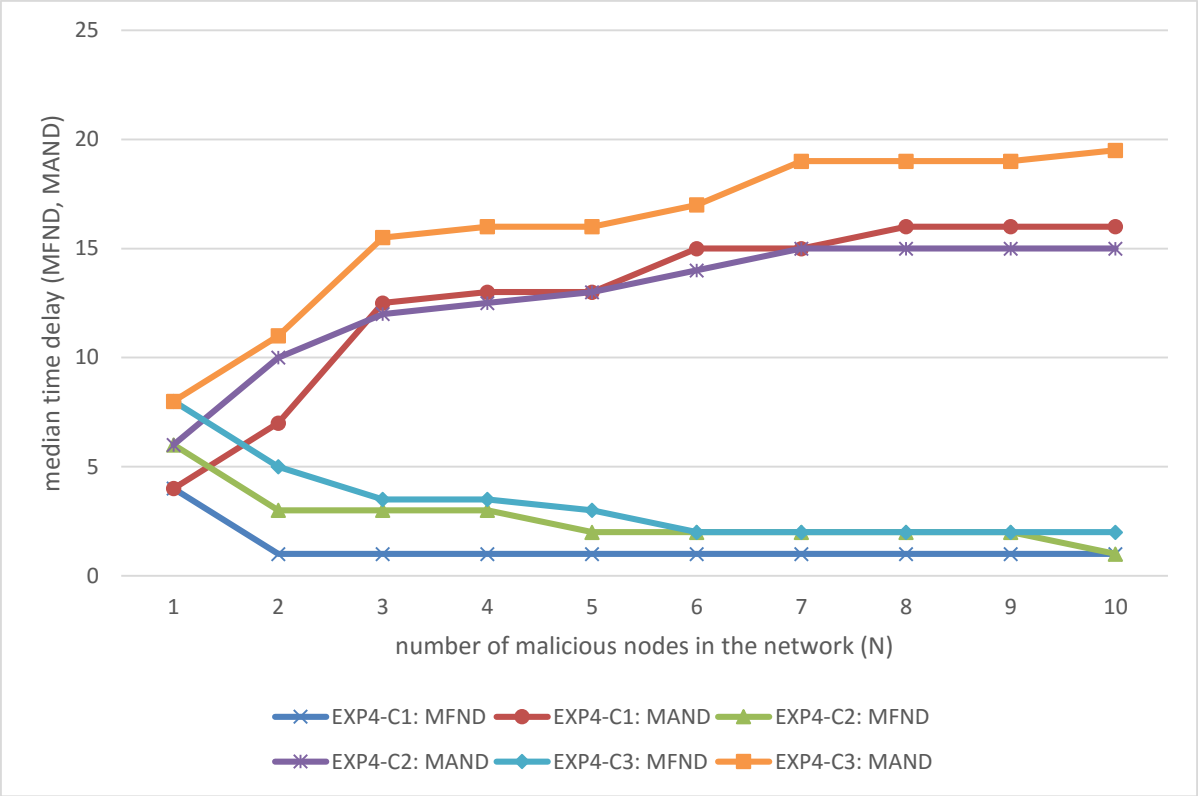

Figure 35 EXP4 results: median time delay of detecting first and all faulty nodes for n=300

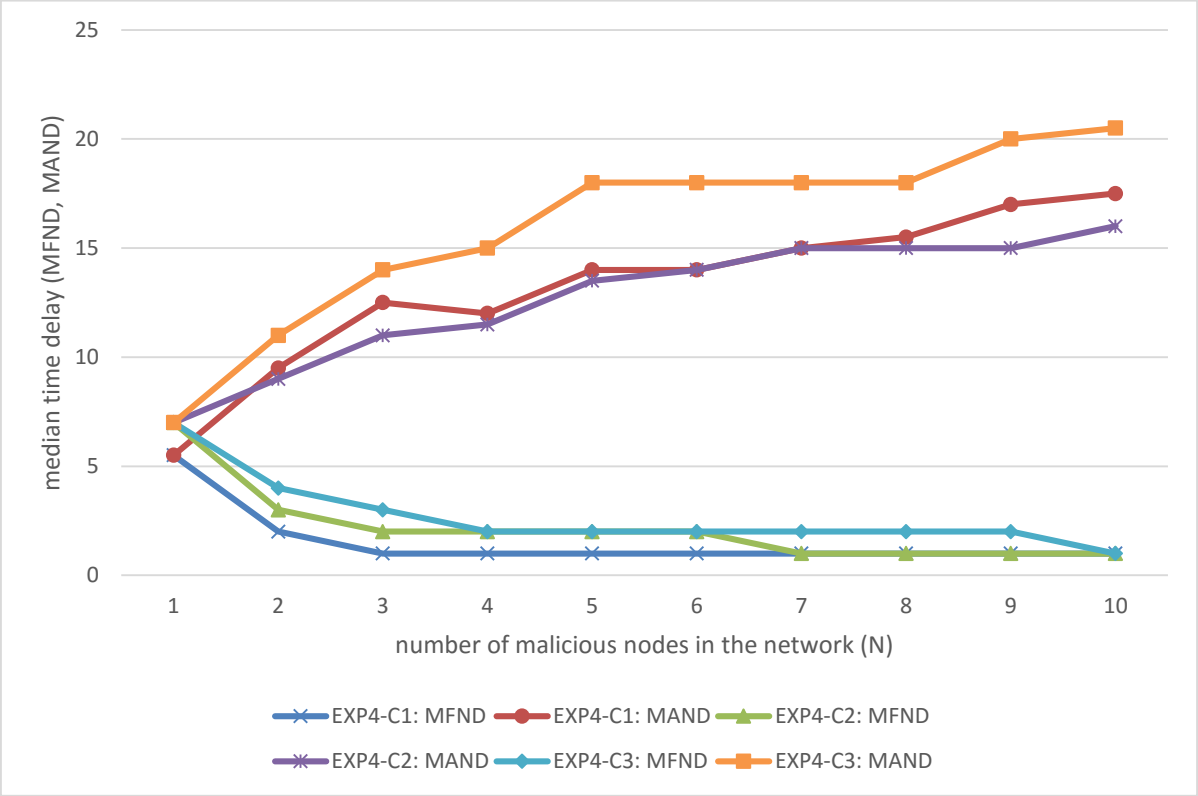

Figure 36 EXP4 results: median time delay of detecting first and all faulty nodes for n=1000

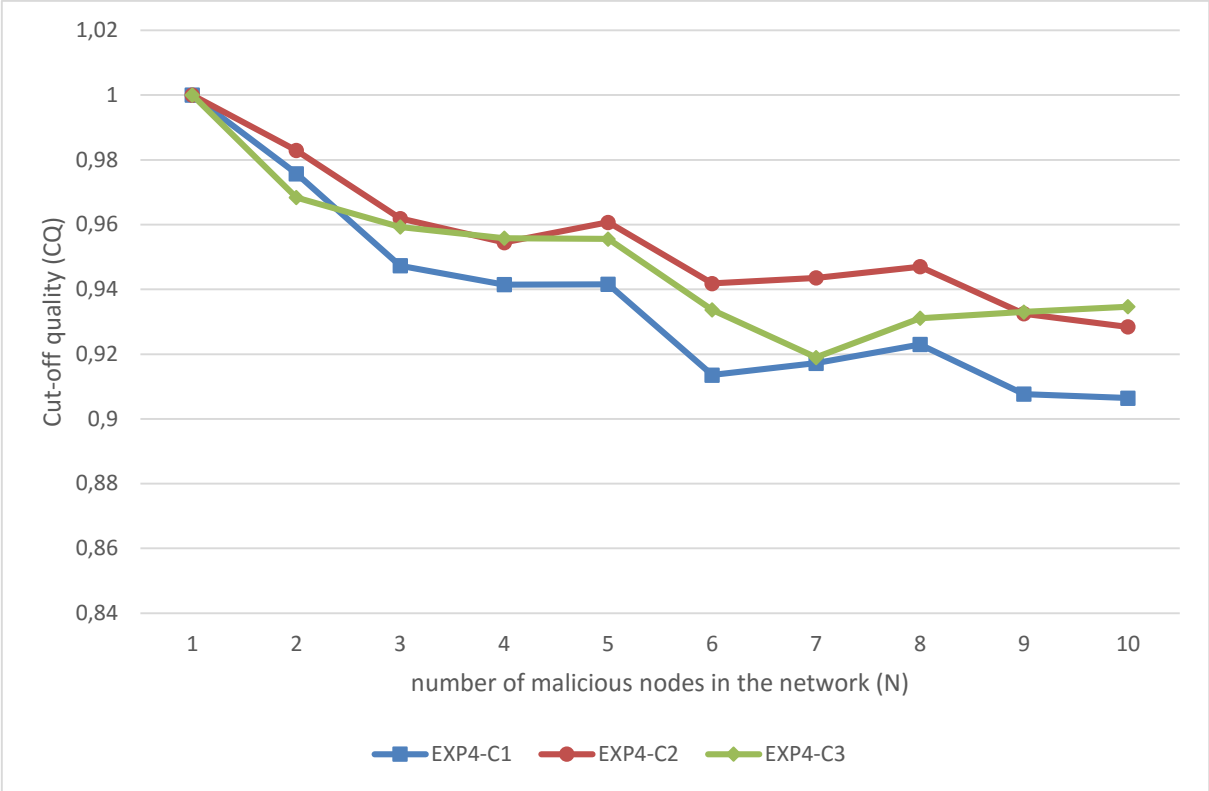

Figure 37 EXP4 results: CQ for n=20

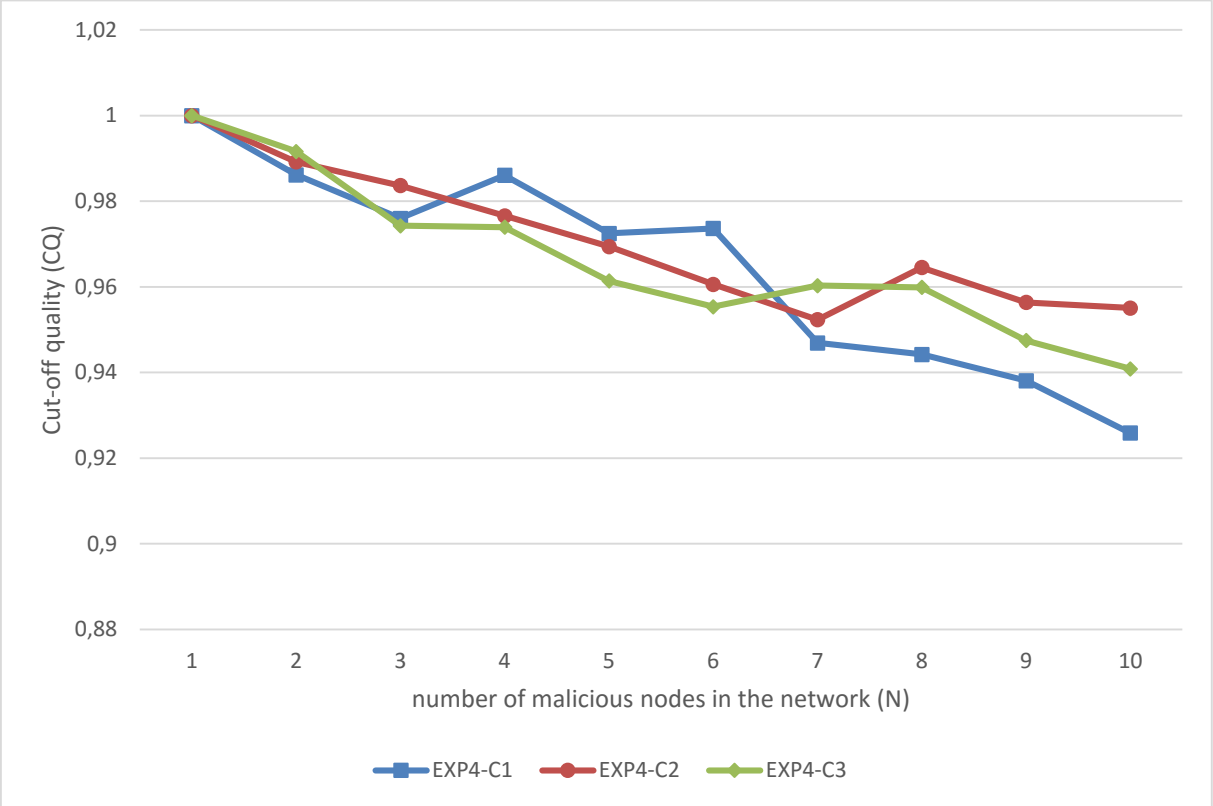

Figure 38 EXP4 results: CQ for n=100

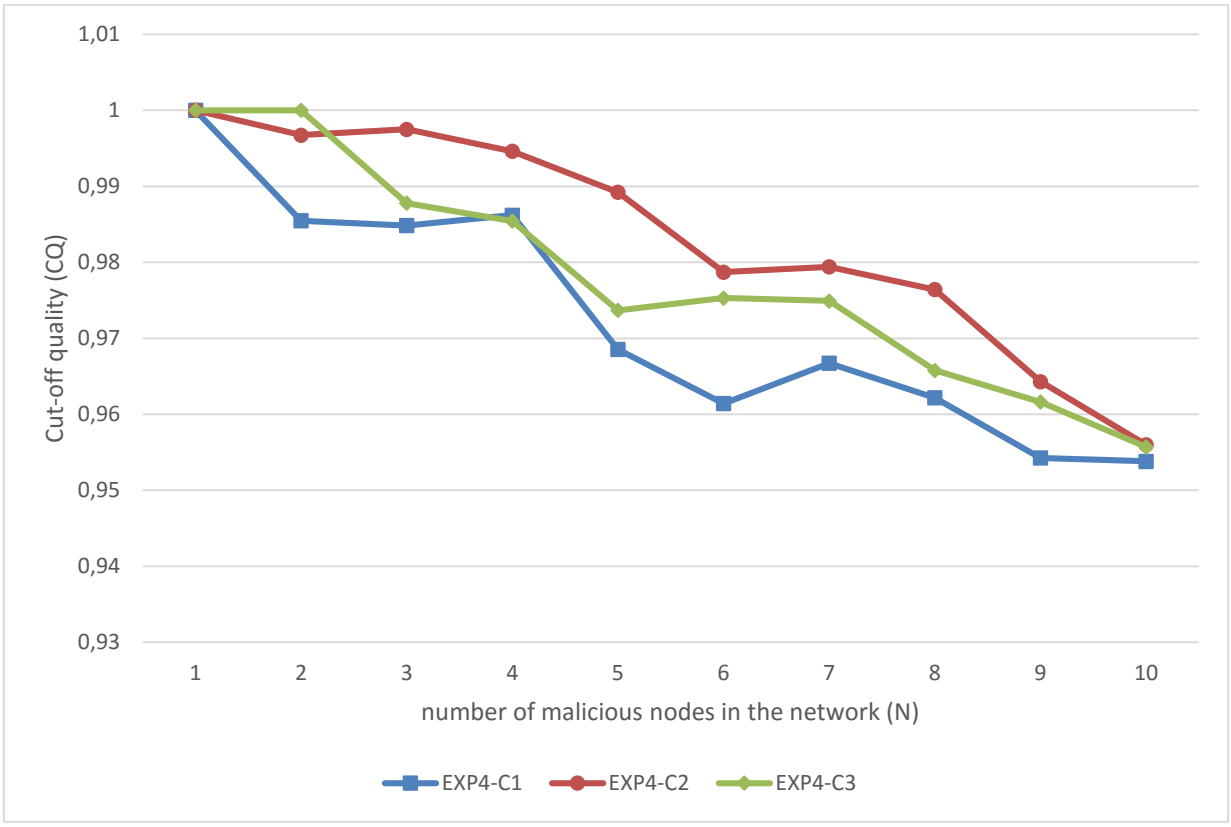

Figure 39 EXP4 results: CQ for n=300

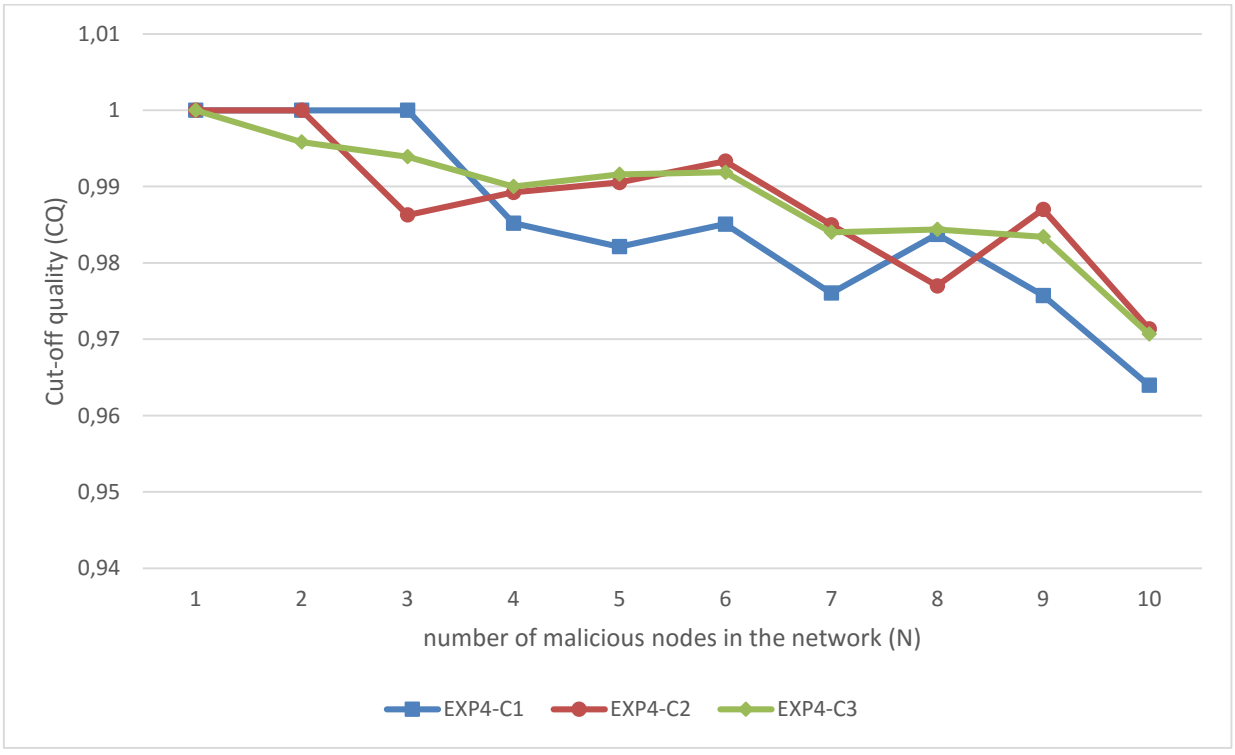

Figure 40 EXP4 results: CQ for n=1000

The results for other network show similar trends and therefore are not included. In all test cases MDQ was equal 100%. From Figure 33 - Figure 40 there can be made the following observations:

- median time delay needed to detect the first faulty node (MFND) is greater in case of one-tier network comparing to two-tier network but median time delay needed to detect all faulty nodes (MAND) is lower. This difference gets bigger with the number of nodes in the network;
- for the one-tier network, cut-off quality (CQ) is better than in two-tier network, in all cases it gets lower with number of faulty nodes in the network;
- for a two-tier network, changes of base station and nodes activity do not affect the delays needed to detect the first faulty node and all faulty nodes in a significant way and the change gets lower with number of faulty nodes in the network;
- bigger number of nodes in the network nearly does not influence the median time delay needed to detect all faulty node (MAND), but decreases cut-off quality (CQ).

To learn more about the effectiveness of detecting all faulty nodes in the network, the NMAND metric was calculated. Figure 41 presents the results for the case C2.

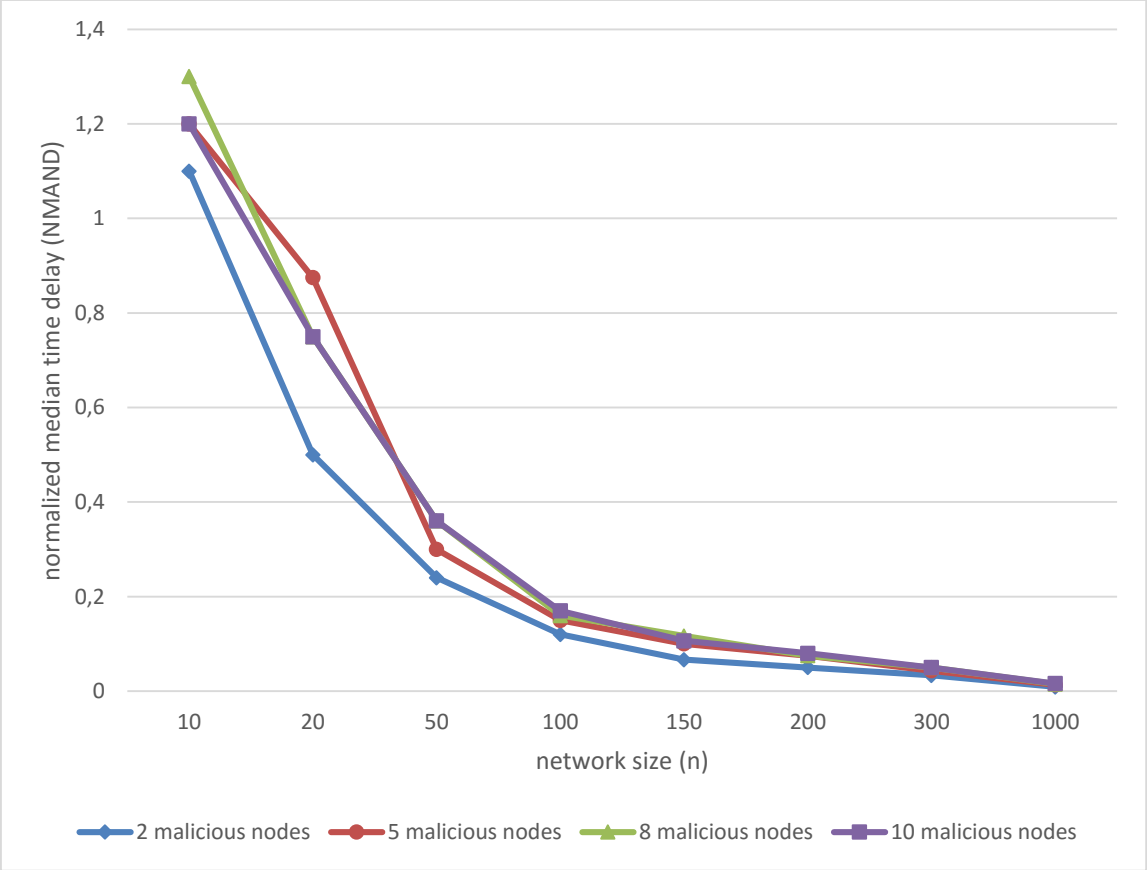

Figure 41 Median time delay of detecting all faulty nodes normalized by the number of network nodes in EXP4-C2

In Figure 41 it can be observed that NMAND decreases rapidly as the number of nodes in the network grows and then achieves a sort of saturation.

It suggests that there is an 'initial investment' that must be paid to detect a faulty node which does not depend on the number of nodes in the network. With the increase of the network size this initial investment gets distributed between the large number of nodes and its effect becomes invisible.

7.2.5 EXP5: Resistance to decreased frequency of attack

7.2.5.1 Description

The results presented in Section 7.2.1.2 for the blackhole attack scenario shows, that a decreased frequency of attack, when a malicious node performs malicious actions in a fraction of time (not continually), can be successful in the presence of WCT2M. To deal with this situation the method was enhanced by adding the *action history* attribute, as described in Section 5.2.5.

To check if the action history is effective, experiment EXP5 was conducted. During this experiment, the test cases S1-S4, B1-B3, M1-M3 and F1-F3 of experiment EXP1 (see Table 9 to Table 14) were simulated again. For every test case, 100 simulation runs were conducted with the help of WCTMS. For each test case, metrics AFND, MFND, AAND, MAND, MDQ, Im_{AFND} and Im_{AAND} were calculated.

7.2.5.2 Results

The comparison of results achieved in experiment EXP1 (Section 7.2.1.2) with results achieved using action history in experiment EXP5 is summed up in Table 27 - Table 30.

Table 27 EXP5 results for spam attack scenarios with action history

Case	p_e	without action history			with action history			Im_{AAND}
		MFND MAND	AFND AAND	MDQ	MFND MAND	AFND AAND	MDQ	
S1	100%	1.0	1.0	100%	1.0	1.0	100%	0%
S2	80%	1.0	1.7	100%	1.0	1.55	100%	8,8%
S3	60%	3.0	3.38	100%	2.0	2.28	100%	32,5%
S4	40%	5.0	8.76	100%	4.0	5.16	100%	41,1%

Table 28 EXP5 results for blackhole attack scenarios with action history

Case	p_d	without action history			with action history			Im_{AAND}
		MFND MAND	AFND AAND	MDQ	MFND MAND	AFND AAND	MDQ	
B1	100%	6.0	6.0	100%	3.0	3.0	100%	50%
B2	80%	92.0	134.41	100%	5.0	5.49	100%	95,9%
B3	70%	4226.0	5588.73	100%	7.0	7.29	100%	99,9%
B4	60%	-	-	0%	24.5	32.04	100%	-
B5	50%	-	-	0%	137.5	171.76	100%	-

Table 29 EXP5 results for message modification attack scenarios with action history

Case	p_{ch}	without action history			with action history			Im_{AAND}
		MFND MAND	AFND AAND	MDQ	MFND MAND	AFND AAND	MDQ	
M1	100%	2.0	2.0	100%	1.0	1.0	100%	50%
M2	80%	3.0	3.18	100%	2.0	1.94	100%	39,0%
M3	50%	18.0	21.48	100%	4.0	5.34	100%	75,1%

Table 30 EXP5 results for faulty nodes scenarios with action history

Case	p_s	without action history					with action history					Im_{AFND}	Im_{AAND}
		MFND	AFND	MAND	AAND	MDQ	MFND	AFND	MAND	AAND	MDQ		
F1	70%	2.0	2.5	2.0	2.5	100%	2.0	1.75	2.0	1.75	100%	30%	30%
F2	70%	5.0	5.58	5.0	5.58	100%	3.0	3.55	3.0	3.55	100%	36%	36%
F3	70%	2.0	2.3	7.0	7.3	100%	2.0	1.88	4.0	4.42	100%	18%	39%

The experiment results show that action history is an effective tool to decrease time of detecting the malicious node in the network. It is especially useful while a malicious node performs attack with decreased frequency – the lower is frequency of malicious actions, the more effective is the proposed enhancement.

However, the usage of the communication history with WCT2M needs a more capacious nodes' memory, because more information needs to be stored.

7.2.6 EXP6: Resistance to collusion attack

7.2.6.1 Description

Collusion attacks can distort WCT2M or even neutralize it. To minimize the attackers influence, WCT2M was enhanced by adding the additional trust history table, as described in Section 5.2.3.

To check if such modification of the method is effective, experiment EXP6 was conducted. The test cases of this experiment are summed in Table 31.

Table 31 Parameters for collusion attack scenarios of experiment EXP6

Case	Probability of modifying a trust table by a malicious node (p_t)
C1	100%
C2	80%
C3	50%

The simulation space was prepared in the same way as in Experiment EXP4 (Section 7.2.4). During simulations networks of size n = 20, 50, 100, 150, 200, 300 nodes were considered and for each network of size n, the simulations were performed for different number of malicious nodes N, where $N \in [1..n/2]$. For each network characterized by the numbers n and N, 100 simulation runs were performed. For each such experiment the MAND and MDQ metrics were calculated.

7.2.6.2 Results

The simulation results for experiment EXP6 for network of 20, 50 and 100 nodes are presented in Figure 42 - Figure 44. The results for other network sizes show similar trend and therefore are not included. Each figure presents what is the number of WCT2M cycles needed to detect and cut-off the malicious nodes depends on the probability of modifying a trust table by a malicious node (p_t) in different sizes of network.

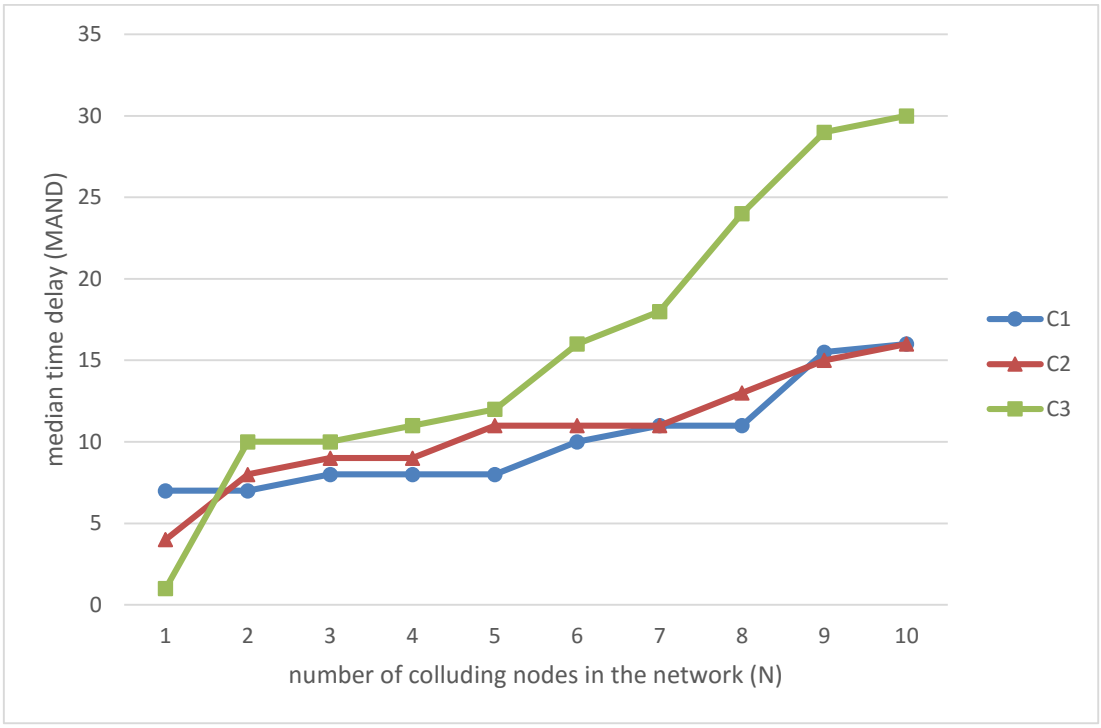

Figure 42 EXP6 results: median time delay of detecting all colluding nodes for n=20

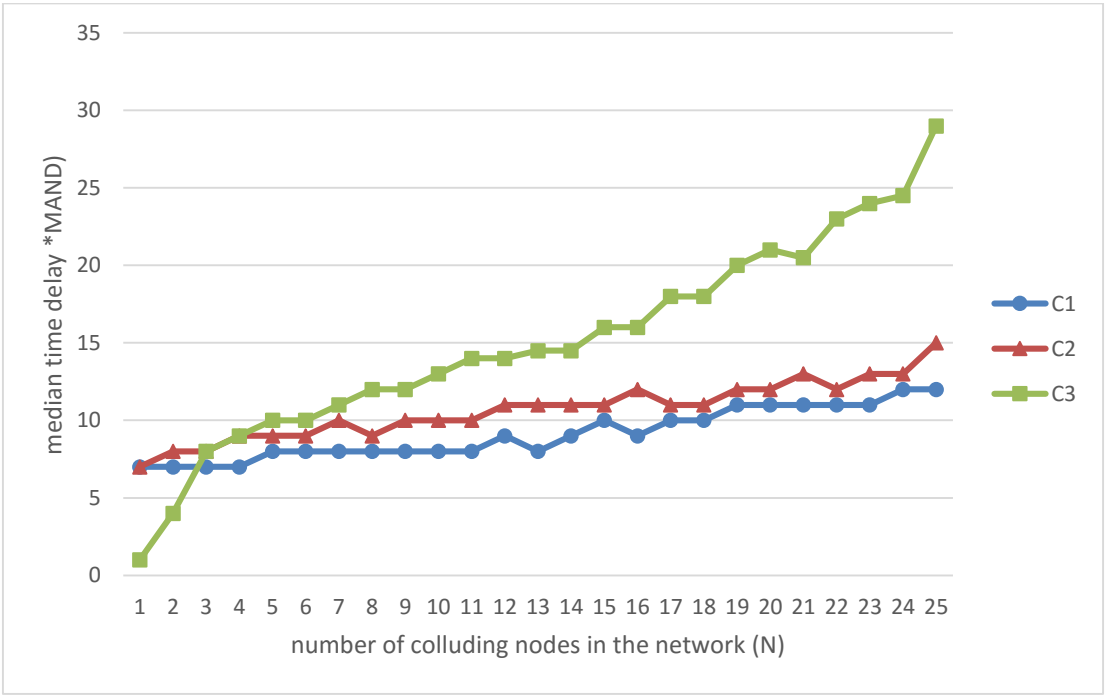

Figure 43 EXP6 results: median time delay of detecting all colluding nodes for n=50

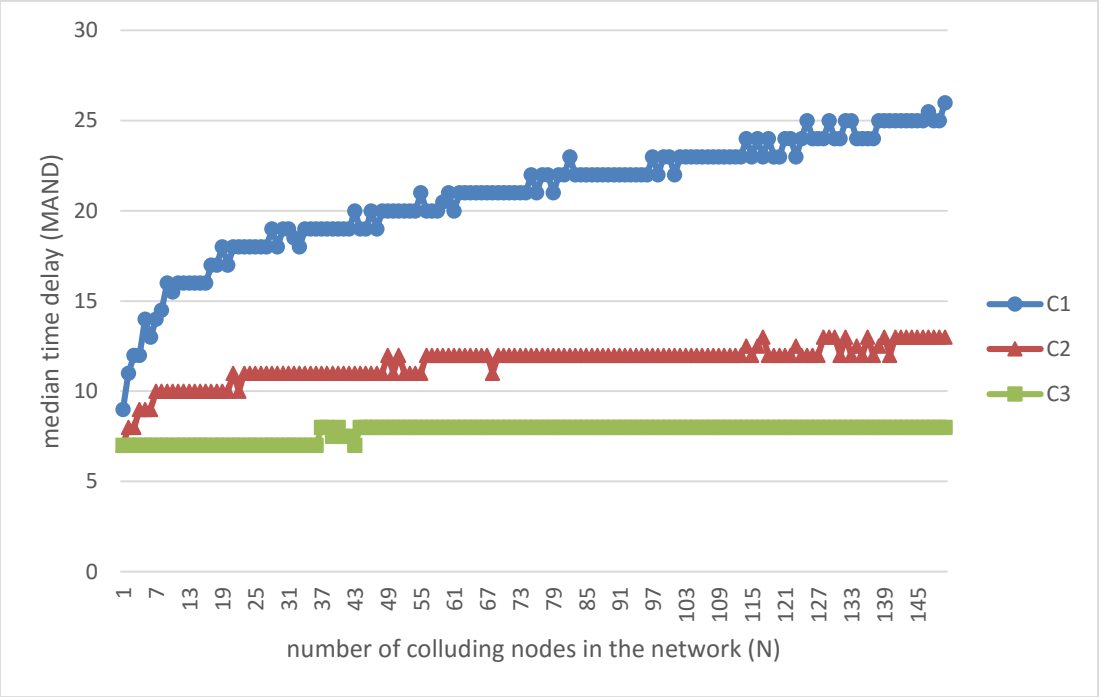

Figure 44 EXP6 results: median time delay of detecting all colluding nodes for n=300

In all test cases MDQ was equal 100%.

It can be observed that WCT2M is able to detect all colluders in acceptable time, unless the number of colluders approaches half of the whole number of nodes. This property is better visible in small networks. The smaller probability of modifying a trust table by a malicious node allowed the colluding nodes to remain undetected longer, but only in small networks.

7.2.7 EXP7: Influence of effectiveness of security mechanisms

7.2.7.1 Description

Experiment EXP7 was prepared to assess efficiency and effectiveness of WCT2M depending on the effectiveness of security mechanisms implemented in the node. This effectiveness of security mechanisms is represented by the r parameter in Table 7 and Table 8. For instance, r=90% means that there is 90% probability that the security mechanism detects that an incoming message violates the compulsory security policies. While conducting the experiment, the nodes were assumed to be distributed in the rectangle of the size X×X' and X = X' = 100 points (a point is a distance unit). There were considered two different distribution spaces:

- the distribution space was unstructured (one-tier network);
- the distribution space was divided into 16 clusters of size Y×Y' each, where Y = Y' = 25 points (two-tier network).

The base station was placed at the point S (102, 102), outside the nodes distribution area. The node signal range was set to Z = 30 points. All nodes were fixed (they could not change their position during a given simulation). Example of the distribution space for one-tier and two-tier networks is shown in Figure 3.

During simulations, networks of size n = 10, 20, 50, 100, 150, 200, 300 nodes were considered and for each network of size n, the simulations were performed for different number of faulty nodes N, where N ∈ [1..10] and different probability r of recognizing a spoiled message by the receiver , where r = 100%, 90%, 80%, 70%, 60%, 50%. For each test case characterized by the numbers n, N and r, 100 simulation runs were performed. For each test case, the metric MAND was used to measure efficiency of WCT2M and the metric CQ was used to measure effectiveness of WCT2M.

7.2.7.2 Results

The results of experiment EXP7 for the networks of 20, 100 and 300 nodes are shown in Figure 45 - Figure 50. For clarity, only results for 1, 4, 7 and 10 malicious nodes in the network are presented.

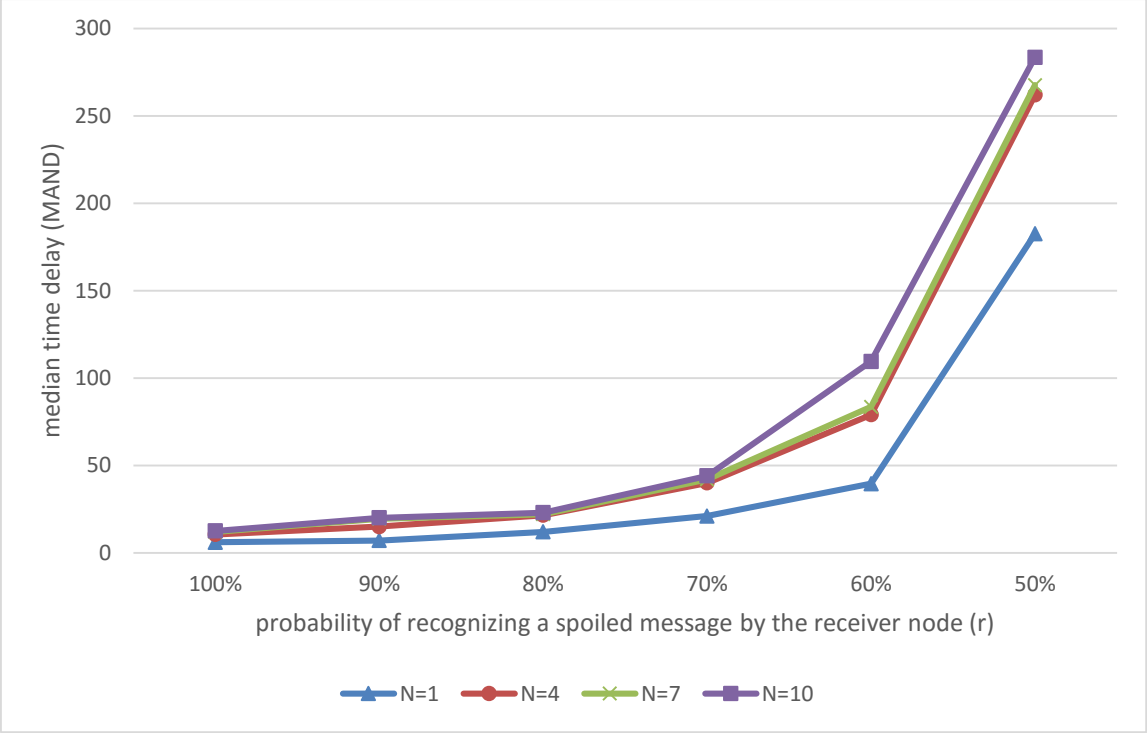

Figure 45 EXP7 results: median time delay of detecting all faulty nodes for n=20

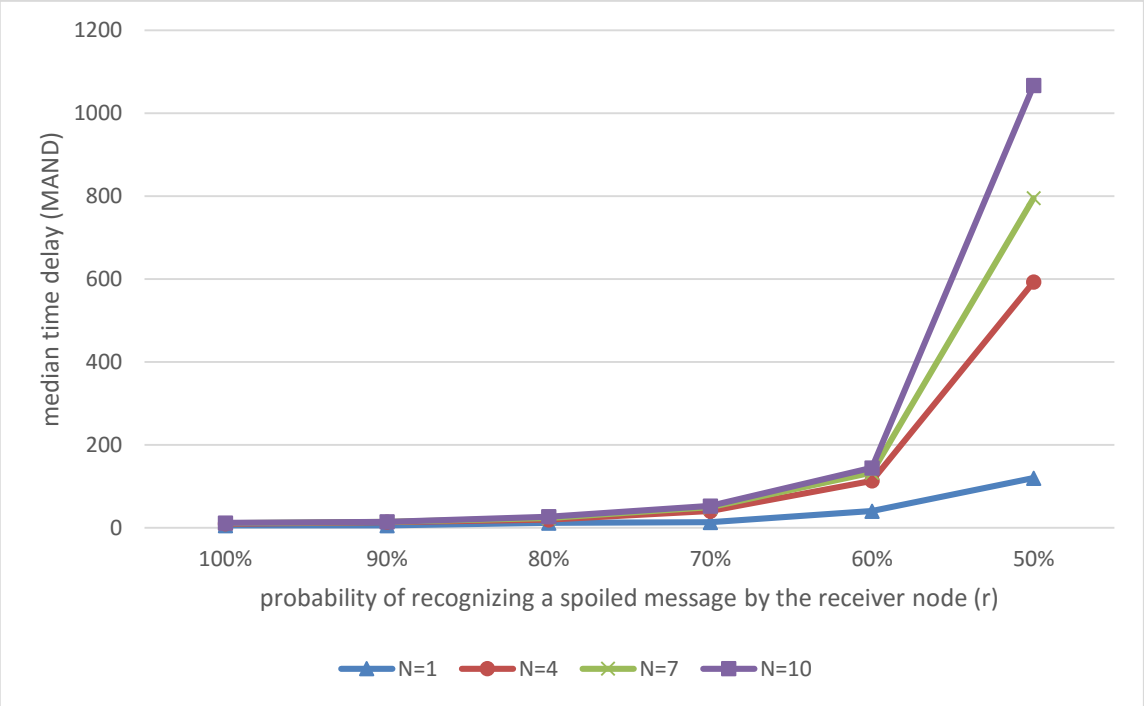

Figure 46 EXP7 results: median time delay of detecting all faulty nodes for n=100

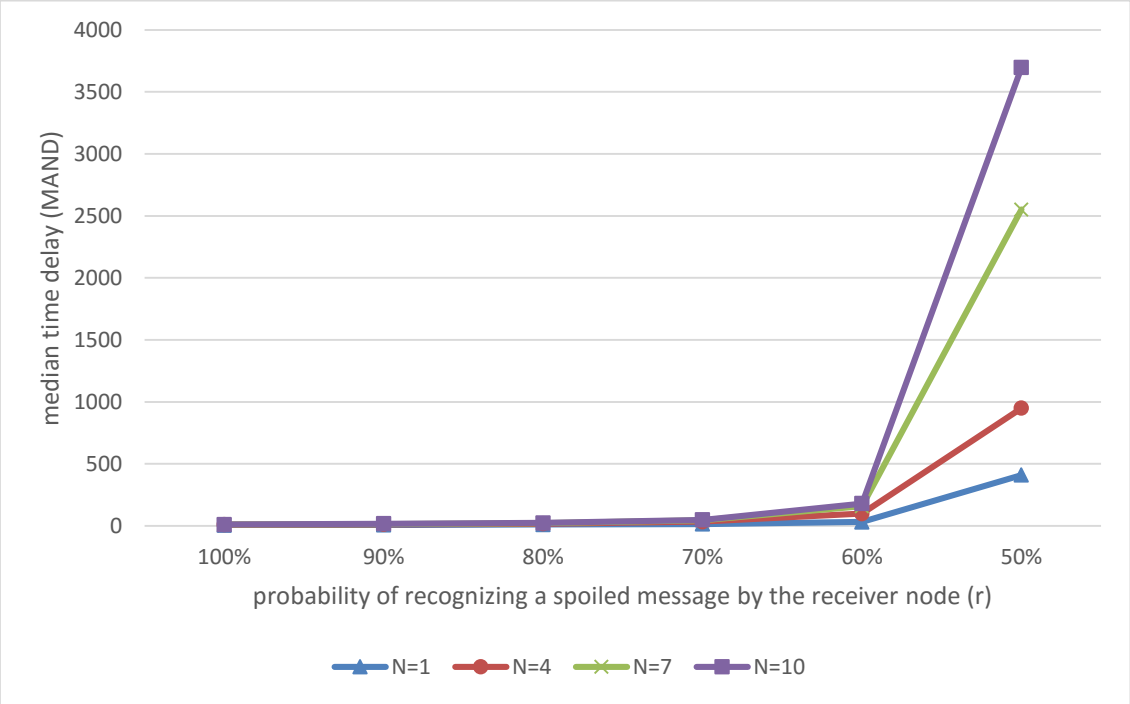

Figure 47 EXP7 results: median time delay of detecting all faulty nodes for n=300

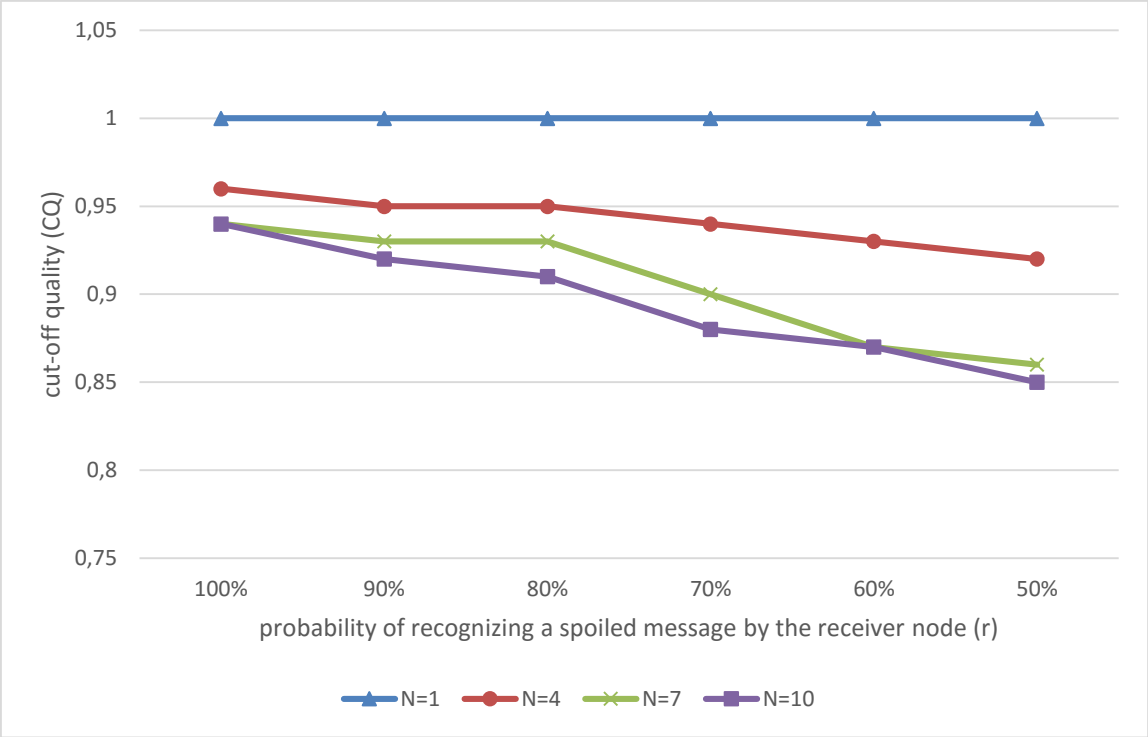

Figure 48 EXP7 results: CQ for n=20

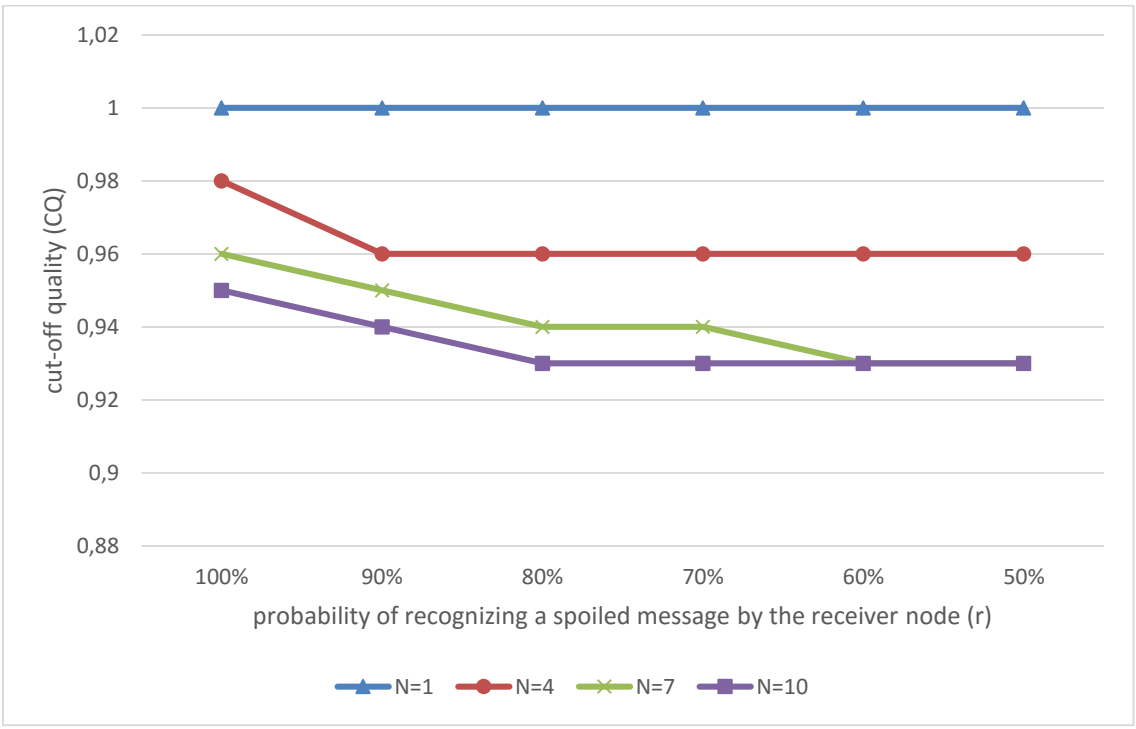

Figure 49 EXP7 results: CQ for n=100

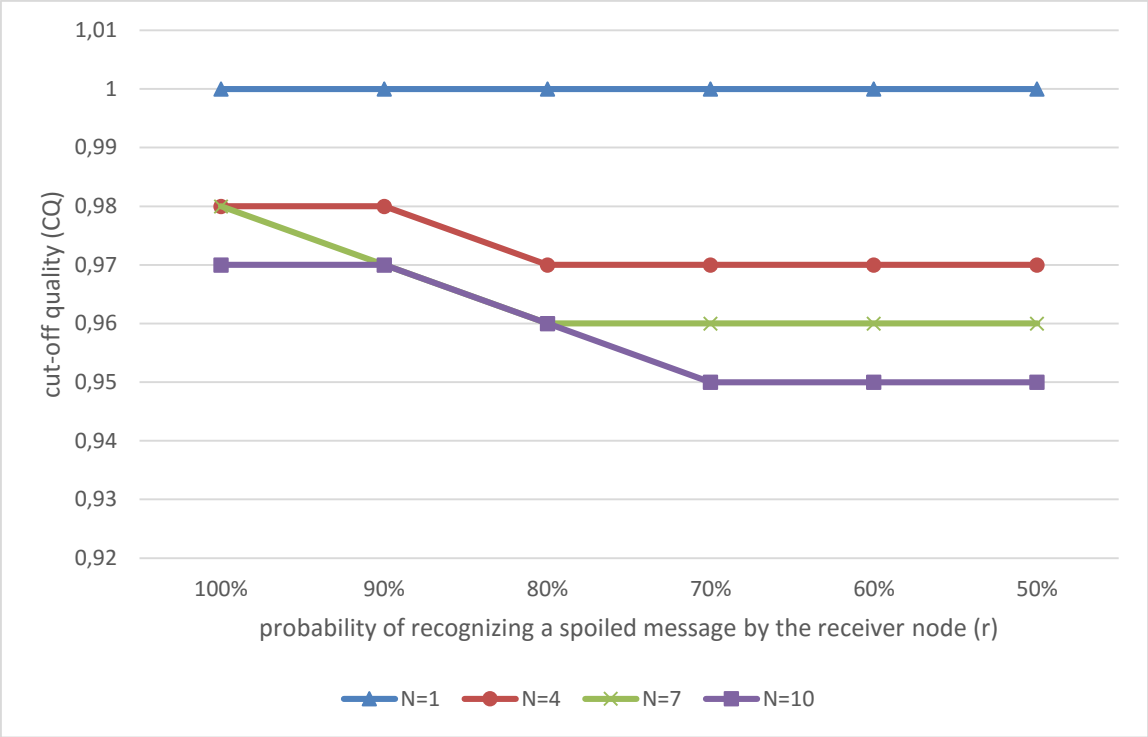

Figure 50 EXP7 results: CQ for n=300

The results for other simulated networks show similar trends and therefore are not included. From Figure 45 - Figure 47 it can be observed that the effectiveness of security mechanisms (used for evaluating received messages) represented by r parameter highly impacts efficiency of WCT2M. Moreover, it can be observed from Figure 48 - Figure 50 that r parameter has low impact on cut-off quality (CQ metric) so the effectiveness of WCT2M is not affected significantly. It means that regardless of how effective the security mechanisms are, it is most likely that a malicious node is cut off by its nearest neighbours. It can be explained by the fact, that detecting a spoiled message on every node are independent events in WCTMS simulator. Consequently, even if r = 50% the nearest neighbour detects 50% of spoiled messages, the next detects 25% of spoiled messages and so on (this is illustrated in Figure 51).

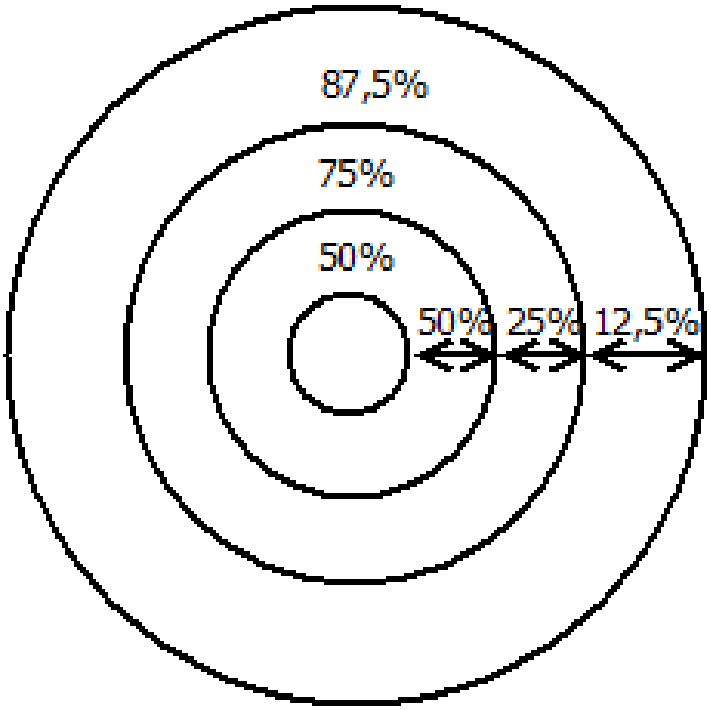

Figure 51 Probability of detecting a spoiled message by nodes on a route from a sending node, r=50%

However, in real networks detecting the same spoiled message by subsequent nodes on the route from the sending node should not be treated as independent events, because every node usually has the same security mechanisms implemented. It means that if the security mechanism tests a message without comparing its content to other received messages, and the first node does not detect a spoiled message, then it is likely that the next one also does not detect it. When security mechanisms compare content of received messages to other received messages the probability of detecting a spoiled message raises with the number of received messages (e.g. while comparing temperature values from a given area).

For this reason, the simulations for EXP7 were repeated with the following modification of the WCTMS: only the first node which is able to detect a spoiled message has r parameter assigned the value given for the test case, the all subsequent nodes have r=0%. The results for the network of 100 nodes are shown in Figure 52. The achieved results are slightly worse comparing to these achieved without this modification.

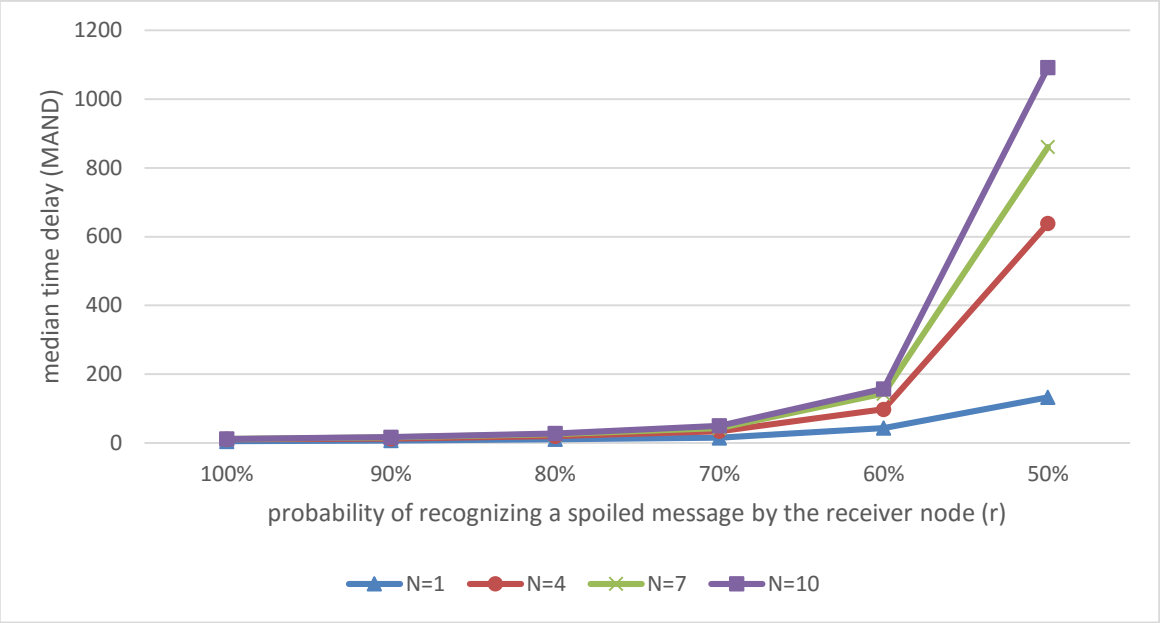

Figure 52 EXP7 results: median time delay of detecting all faulty nodes for n=100 when only first possible node detects spoiled messages

Experiment shows (Figure 45, Figure 46, Figure 47, Figure 52) that if r>=70% the malicious nodes are detected efficiently without significant delay. However, if the value of r drops below 70% the efficiency of malicious nodes detection decreases rapidly. It can be explained by the adopted method of trust change: the higher the current trust level is, the faster it decreases and slower it increases and vice versa: the lower the current trust level is, the faster it increases and slower it decreases (see Section 5.2.4). It means that the lower number of detected spoiled messages allows a malicious node to regain trust lost when spoiled messages were detected.

Example 8

Assume that node's B trust value towards node A is 0.9. Then:

- if A sends an unspoiled message, the B's trust in A is increased to 0.905 (0.05 is added);
- if A sends a spoiled message, the trust value is decreased to 0,72 (0.18 is subtracted).

Now assume that node's B trust value towards node A is 0.4. Then:

- if A sends an unspoiled message, the B's trust in A is increased to 0.43 (0.03 is added);
- if A sends a spoiled message, the trust value is decreased to 0,32 (0.08 is subtracted).

It demonstrates that in WCT2M the spoiled messages (if detected by the security mechanisms) have stronger influence on lowering the trust and the unspoiled messages have lower influence on regaining the trust.

In addition to the above differences in the influence the spoiled and unspoiled messages have on the trust value, in the experiment the probability p_s of spoiling a message by a malicious node was 70%, so 30% of the sent messages were always causing the reputation raise. The result was that the malicious nodes were losing their reputation slowly and in effect after more WCT2M cycles were isolated from the network.

7.3 Results analysis

Internal validity is the degree to which we can appropriately conclude that the changes in X caused the changes in Y [98]. To achieve the internal validity of the experiments, in every experiment no more than two inputs were changed to achieve the result. Every simulation experiment was conducted 100 times and the nodes were distributed in the way presented in Figure 10 or randomly distributed for each individual simulation. In experiment EXP3, in which two inputs were set, there were conducted tests for every combination of inputs (from the accepted range).

External validity (generalizability) is the degree to which the results can be generalized to different participants, places, time periods, etc. [99]. To ensure the external validity of the method, the described above experiments were conducted for different sizes and distributions of the network. These experiments were conducted ensuring that every nodes distribution has the same chance to occur in the experiment.

Each experiment can be interpreted as realignment of one or a mix of the threat scenarios (Section 3.3) presented for the case study (Section 3.1).

The achieved results of the experiments demonstrate the potential of the proposed trust management mechanism to detect and isolate malicious nodes in a sensor network.

The proposed AFND, MFND, AAND, MAND and NMAND metrics showed that WCT2M can efficiently detect malicious nodes in the network regardless of the total number of nodes in the network. However, to achieve such results the security mechanisms implemented in the node should detect at least 70% of spoiled messages. Exceeding this value causes a rapid decline in efficiency of WCT2M. It was presented for different behaviours of malicious nodes. The experiments showed that not every threat scenario described in Section 3.3 is detectable by WCT2M without additional features (as introduced in Section 5.2.5) enabled. The proposed Im metric demonstrated the impact of these features on method efficiency.

The values of proposed MDQ metric demonstrated that WCT2M is able to detect all malicious nodes. However, the blackhole attack, collusion attack and decreased frequency of attack require more

resources (more memory to use additional WCT2M features) what can be a problem in dense networks. Moreover, if we assume a limit of WCT2M cycles needed to detect all malicious nodes (as a limit of resources), e.g. 10, the result would be worse. In real networks, which are usually assumed to work for years without human intervention, it should not be a problem from the perspective of resources. However, the longer a malicious node work undetected, the bigger is a potential damage.

The proposed CQ metric showed that WCT2M not always cuts off the malicious nodes as near as it is possible. It can be compared to a surgeon who cuts off cancer cells and usually cuts also some healthy cells around. Moreover, experiment EXP4 demonstrated that CQ value is quite stable regardless of the number of nodes in the network. It allows to conclude that WCT2M can effectively detect malicious nodes also in networks containing large number of nodes.

8. Related works

Trust management for complex wireless sensor networks is currently an area of active research. The main question is how network nodes should behave in order to effectively identify and isolate malicious nodes and to minimize false positives and false negatives.

8.1 Architectures for trust models

There are two main architectures considered for trust models: centralized and distributed [10]. The first type distinguishes the Trust Authority (e.g. the base station) which manages trust relationships between nodes [100]. This solution is efficient and manageable, but it has problems with scalability and robustness.

A distributed trust model is considered to be suitable for large-scale sensor networks. Zhiying et al. [10] find this model appropriate for sensor network security design because a node focuses on trustworthiness of its neighbours and can assess if these nodes obey agreed security policies. They propose a corresponding security framework with different security schemes. However, their work does not take into consideration limited resources of nodes in sensor networks.

Chen et al. [101] propose a distributed agent-based trust management scheme where each agent node independently monitors the behaviour of the nodes within its radio range and broadcasts their trust ratings. They also introduce a reputation based trust model using probability, statistics and mathematical analysis and have suggested a trust system to build up a reputation space and trust space in WSNs [102].

A hybrid approach is also possible. It combines the advantages of the centralized and distributed models (but also can incorporate problems connected with each of these architectures). The network structure comprises two-levels – all nodes are divided into clusters and each cluster head is an element of so called backbone network, which enables communication of cluster heads with the base station. Boukerche et al. [103] propose trust and reputation management scheme that uses mobile agents running on each node. In this model there is a central agent launcher responsible for generating and launching agents into the network. However, there is no central repository of trust, which makes trust exchange (if there are mobile nodes) significantly more difficult.

8.2 Trust management as a solution for security issues

Trust management schemes appears to be solution for many security issues in WSN. Zahariadis et al. [104] state that cryptography and strong authentication schemes are not a solution to WSN security issues because they do not detect a large set of attacks such as selfish behaviours and black holes

while at the same time their implementation at low cost is not feasible. They survey trust models in an attempt to explore the interplay among the implementation requirements, the resource consumption and the achieved security.

Yu et al. [105] emphasize in their work importance of trust issue in wireless sensor networks as important factor in security schemes. They categorize various types of attacks and countermeasures related to trust schemes in WSNs. They also prepared an extensive literature survey by summarizing state-of-the-art trust mechanisms in two categories: secure routing and secure data. They also pay attention to the fact that using trust management methods can introduce new classes of attacks on wireless sensor networks.

Lopez et al. [106] list the best practices that they consider as essential for developing a good trust management system for WSN. They find that the nature of WSN and its vulnerabilities to attacks makes the security tools required for them to be considered in a special way.

Buccafurri et al. [107] propose special Trust and Reputation layer added to the modified version of the OSI model, under Ad hoc On-demand Distance Vector (AODV) layer. This approach allows to provide intrusion tolerant routing in WSNs. The Trust and Reputation layer uses a proactive approach, where a potentially not trusted node is explicitly tested to obtain a direct measurement of corresponding trustworthiness. This allows to collect relevant information by the trust-based architecture independently of the specific features of the considered routing protocol.

Abassi et al. [62] pay attention to the problem of collusion attacks in trust-based MANET networks. Trust management may be faked by cheaters: several nodes collude in order to rate each other with the maximum value and at the same time decrease other nodes' reputations by giving negative recommendations about the latter. In their work they propose detecting colluding nodes through the calculation of a recommendation deviation and punishing these nodes by discarding them from further communication.

8.3 Assessment of trust management methods

To present results of different WSN researches different metrics are introduced and used, which makes it difficult to compare the achieved results. Moreover, metrics and experiments conditions used are often not fully described.

Maroti et al. [108] examine the number of errors in a period of time and create histograms using real time of simulation.

Also Loscri et al. [109] use the real time in their simulation experiments. They present numbers of nodes (e.g. alive nodes) or amounts of data (sent and received) in a period of time. They also use other metrics not connected with time, e.g. number of nodes in number of data signals.

Zia [110] in the experiments uses nodes that transfer one packet every n seconds. Time needed to detect all distrusted nodes is used to assess the method effectiveness.

In comparison, Heinzelman et al. [30] measure numbers of nodes in time steps (simulation rounds), as called system lifetime.

Interesting metrics are used by Handy et al. [31]. They introduce three metrics: First Node Dies (FND), Half of Nodes Alive (HNA) and Last Node Dies (LND). These metrics are used to measure energy usage. The results are presented in simulation rounds.

8.4 Comparison to other works

To prove that achieved results ensure effective malicious nodes detection, the comparison with other WSN trust management methods was conducted.

8.4.1 Trust-based LEACH protocol (TLEACH)

Song et al. [111] propose a Trust-based LEACH (TLEACH) protocol to enhance the security of LEACH protocol [30], while preserving the essential functionalities of the original such as head-election algorithm and working phases. They add trust slots to the implemented trust-based routing module to provide better support for trust evaluation. Decisions are made based on the decision trust, which is evaluated separately and dynamically for different decisions. They also suggested a cluster-head-assisted monitoring control scheme to reduce energy consumption.

The basic algorithm of TLEACH protocol is similar to the one used in WCT2M, so achieved results when not using trust history in WCT2M for 100% nodes activity are similar. However, TLEACH allows to regain the lost trust quicker than WCT2M but nodes which are not cluster-heads can use less energy. That makes TLEACH method a hybrid one (distributed mixed with centralised), when nodes use information achieved from cluster-head rather than gathered themselves. This situation takes place when non-cluster-head node has high trust value towards its cluster-head.

8.4.2 Agent-based Trust Model in WSNs (ATSN)

Chen et al. [101] propose a distributed agent-based trust model. The model uses watchdog scheme to observe the behaviour of nodes and broadcast their trust ratings. The agent nodes use promiscuous mode to monitor the behaviour of the sensor nodes within their radio range and classify the actions. Each agent node holds several modules. Each module carries out a specific

function that can classify the collected data and mark as cooperative or uncooperative behaviour action. Moreover, the model uses aging, what means the recently obtained information is treated as more important. It allows to successfully detect nodes with lower activity. The model does not use recommendations from other nodes, so only a node's own are taken into account. That allows the malicious node to cooperate with the network even if it execute malicious actions limited to some nodes. It can be a big security problem.

8.4.3 Reputation-based Trust Management Scheme

Zia [110] proposes reputation-based trust management scheme which uses trust vote to establish trust among nodes. Value of trust vote is increased with every successful message transmission from one node to another. This trust value is compromised when a neighbouring node enters a negative vote for a particular node. Every node after sending a message which needs to be retransmitted, listens to the retransmission and increase the trust value if the message remains unchanged and decrease the trust after any change. If negative votes reach a pre-determined threshold, such voted node is declared as un-trusted node. The node which makes such declaration broadcasts information about the untrusted node. When notification reaches a cluster leader, it isolates the untrusted node from the cluster and discards any messages coming from it. The cluster leader also broadcasts a message saying that the untrusted node has been isolated, so any message originated from that node is immediately discarded by its neighbours hence isolating and removing the untrusted node from the network.

The schema description does not describe how the trust tables are updated when negative trust is broadcasted. Moreover, it gives cluster leaders (cluster heads) power to isolate a node, although the method does not assume that these nodes are always trustable. It is also unclear how cluster leader is the only node in the cluster that can force other nodes in the same cluster to discard all messages from certain nodes. It seems to be a serious security problem.

During simulations the area of 100 x 100 units was considered with randomly deployed nodes. There were a few sets of simulation executed, with minimally 100 nodes and maximally 350 nodes. Each node had transmission range of 30 units. Whenever any node detected an un-trust node, it increased its trust table by 1 and broadcasted a message to inform other neighbouring nodes. Whenever the counter reached a threshold of 3 for a specific node, its neighbours considered that node as an un-trusted node. Each node transferred one packet every 100 seconds. The probability that the node stayed awake to monitor its neighbours was set to 50%. The simulation was executed 10 times. The detection time differs from 265 seconds for 350 nodes in the network to 315 seconds for 250 nodes in the network.

8.4.4 Gaussian Reputation System for Sensor Networks (GRSSN)

Momani et al. research [112] focuses on modelling and calculating trust between nodes in a Wireless Sensor Network based on sensed events. They introduce a new trust model and a reputation system for WSNs based on a sensed continuous data (temperature) as opposed to works in which trust is calculated based on binary events. It establishes the continuous version of the beta reputation system introduced in [113] and applied to binary events and presents a new Gaussian Reputation System for Sensor Networks (GRSSN). Trust modelling represents the trustworthiness of each node in the opinion of another node, thus each node associates a trust value with every other node, and based on that trust value a risk value required from the node to finish a task can be calculated.

To assess the method several simulation experiments for different scenarios were conducted. In the work there are presented results of trust changes between previously chosen nodes. The trust of properly working node to the malicious one drops from the starting point (0.5) to 0 in more than 80 WCT2M cycles (probably it is assumed that 1 cycle equals 1 second). Trust value equals 10 (the value identical to cut-off level adopted in this work) is was achieved in 36-70 WCT2M cycles. Because no more inputs were described in results discussion, it is not possible to reliably compare both methods, but it looks like WCT2M is more efficient.

8.4.5 Node Behavioural Strategies Banding Belief Theory of the Trust Evaluation (NBBTE)

Feng et al. [114] propose Node Behavioural Strategies Banding Belief Theory of the Trust Evaluation Algorithm (NBBTE) which integrates the approach of nodes behavioural strategies and modified evidence theory. According to the behaviours of sensor nodes, a variety of trust factors and coefficients related to the network application are established to obtain direct and indirect trust values through calculating weighted average of trust factors. Meanwhile, the fuzzy set method is applied to form the basic input vector of evidence. On this basis, the evidence difference is calculated between the indirect and direct trust values, which allows to finally synthesize integrated trust value of nodes.

The simulation results show that the NBBTE algorithm is effective and is able to detect malicious node in 3-4 WCT2M cycles for 100 nodes deployed in the 100x100 units' area. Each node has 20 units range. Unfortunately, authors do not provide many important input values for simulation, e.g. how many simulations were made and if the observed malicious node was the only malicious node in the network. Moreover, all simulations were conducted in flat network (without clusters).

It seems the NBBTE algorithm is as effective as WCT2M, but nodes are forced to conduct more advanced calculations what has influence on battery lifetime.

8.4.6 Hierarchical Trust Management for Wireless Sensor Networks

Bao et al. [115] propose a highly scalable cluster-based hierarchical trust management protocol for Wireless Sensor Networks to effectively deal with selfish or malicious nodes. They consider multidimensional trust attributes derived from communication and social networks to evaluate the overall trust of a sensor node. They described a heterogeneous WSN comprising a large number of sensor nodes with different social and quality of service behaviors.

During simulations they model 900-nodes network and validate influence of proposed weight of social trust parameter as well as different methods of routing on message delivery ratio. The influence of chosen routing protocol on average delay and message overhead was examined and the optimal trust threshold for different network lifetimes was also defined.

8.4.7 Methods comparison

The descriptions of the following methods usually use different metrics than adopted to describe WCT2M and lacks parameters used in simulations so detailed comparison is not possible. However, according to the data found it was possible to prepare descriptive comparison of these methods with WCT2M. It is summarized in Table 32.

Table 32 does not cover Hierarchical Trust Management for Wireless Sensor Networks, as all experiments presented in that work differs from conducted with WCT2M.

Table 32 Estimated evaluation of the selected trust management methods comparing to WCT2M

	Time units needed to detect the first malicious node (AFND, MFND in WCT2M)	**Time units needed to detect all malicious node (AAND, MAND in WCT2M)**	**The percentage of all malicious nodes detected (MDQ in WCT2M)**	**Quality of malicious nodes detection (CQ in WCT2M)**
Trust-based LEACH protocol (TLEACH) [111]	similar	worse (when attacking nodes are not active or trust history is enabled)	similar	similar
Agent-based Trust Model in WSNs (ATSN)	similar	unknown (description covers only one	similar	worse

[101]		malicious node in simulated networks)		
Reputation-based Trust Management Scheme [110]	similar	unknown (description covers only one malicious node in simulated networks)	similar	worse
Gaussian Reputation System for Sensor Networks (GRSSN) [112]	slower	slower	similar	similar
Node Behavioural Strategies Banding Belief Theory of the Trust Evaluation Algorithm (NBBTE) [114]	similar	unknown (description shows only, that it is better than in 'weighed-average method')	similar	similar

9. Summary

9.1 Contributions

The following are the main contributions of the presented research.

9.1.1 WCT2M – a new method of trust management

A new method for trust management in multi-layer wireless sensor networks, called *WSN Cooperative Trust Management Method* (WCT2M), was proposed. The method is based on a distributed trust management model in clustered networks which allows to limit performance problems connected with network expansion.

9.1.2 Set of metrics allowing to evaluate WSN trust management method

A set of metrics was selected, using the Goal-Question-Metrics (GQM) methodology [94]. GQM offers systematic approach which allows to obtain a set of metrics supporting an explicitly stated measurement goal. It resulted in definition of 3 questions and 11 metrics, which are presented in Section 7.1.1. Then, the metrics were used to assess the results of experiments.

The metrics were used both in laboratory experiments as well as in simulator experiments.

9.1.3 Laboratory network and WCTMS simulator

A dedicated laboratory network was created using elements from CC2520 Development Kits [69]. Each node was programmed with dedicated firmware written in C programming language and Vlo_rand [74] and HAL [75] libraries. The firmware allowed to set different modes and parameters using joystick and device buttons. The firmware allowed to perform different experiments for networks composed of 11 nodes (the network presented in Figure 10). The experiments and their results are presented in Section 7.2.

Validating bigger networks using real devices would be too costly and complicated, as thousands of devices scattered on the big area would be difficult to program. Thus, a dedicated simulator in Java was written: WCTMS. It is able to draw the arrangement of nodes in the simulated space using JGraph library [18].

To check the validity of the results obtained with the help of this simulator, the results were compared with the results obtained with the help of the laboratory network (Section 7.2.1.2).

9.1.4 Experiments and their results

During this work a set of experiments were conducted. They were divided into the following groups.

EXP1: Feasibility of implementing WCT2M and implementing the attack models: The results of this experiment demonstrated that WCT2M can be implemented in a typical WSN environment which can contain faulty nodes described in Section 4.2. The experiment also showed that results achieved using the laboratory network are converging with results achieved using the WCTMS simulator.

For the detailed results see Section 7.2.1.

EXP2: The time delay in detecting faulty nodes: The experiment showed that accumulation of the faulty nodes in one cluster slightly influence the effectiveness of trust management comparing to situation when faulty nodes are placed in different clusters of the network. The experiment also shows that a faulty cluster head can be detected efficiently, because it forwards multiple messages during each WCT2M cycle.

Section 7.2.2 presents the detailed results of this experiment.

EXP3: Relationship between nodes' activity and the time delay of detection: The experiment showed that WCT2M mechanism can efficiently detect and cut off a faulty node. The detection time increases inversely to the nodes activity. The experiment also demonstrated that the time needed to detect and isolate a single faulty node depends mainly on the nodes activity and is less dependent on base station activity.

Section 7.2.3 gives the detailed results of this experiment.

EXP4: Relationship between number of faulty nodes and the time delay of detection: The experiment demonstrated that WCT2M allows to efficiently isolate faulty nodes in flat and in two-tier networks of different sizes (networks up to 1000 nodes were examined). The experiment demonstrated that median time delay needed to detect the first faulty node (MFND) is greater in case of one-tier network comparing to two-tier network but median time delay needed to detect all faulty nodes (MAND) is lower. It also showed that cut-off quality (CQ) is better for a one-tier network than in two-tier network and it gets lower with the number of faulty nodes in the network.

Section 7.2.4 presents the detailed results of this experiment.

EXP5: Resistance to decreased frequency of attack: The experiment results showed that action history is an effective tool to decrease time of detecting the malicious node in the network. It is especially useful while a malicious node performs attack with decreased

frequency – the lower is frequency of malicious actions, the more effective is the proposed enhancement.

Section 7.2.5 presents the detailed results of this experiment.

EXP6: Resistance to collusion attack: The experiment showed that WCT2M is able to efficiently detect all colluders, unless the number of colluders approaches half of the whole number of nodes in the network.

Section 7.2.6 presents the detailed results of this experiment.

EXP7: Influence of effectiveness of security mechanisms: The experiment demonstrated how the effectiveness of security mechanisms evaluating received messages impacts efficiency of WCT2M. It also demonstrated that this effectiveness has low impact on cut-off quality (CQ). It was observed that if the effectiveness of security mechanisms is greater than 70%, WCT2M is able to detected and isolate the malicious nodes without significant delay. However, if it drops below 70% the efficiency of malicious nodes detection decreases rapidly.

Section 7.2.7 presents the detailed results of this experiment.

9.2 Conclusions

Security is recently a subject of interest not only for scientists and engineers but also for ordinary people. We use more and more connected devices and Wireless Sensor Networks (WSN) become an important part of a connected world [116]. Trust management is one of the possible solutions for effective security assurance in WSN.

In Section 2 it was explained what are Wireless Sensor Networks and the basic concepts used in this work including the concept of sleep scheduling. Next, in Section 3 the case study used in this work was described to present the rules of trust management. The example network and exemplary threats to that network were presented. The survey of known attacks together with attack scenarios in WSN and known methods of protecting networks against these attacks were described (Section 4). Next, concepts of trust and trust management were introduced (Section 5) and the new trust management method for WSN – WCT2M – was proposed. To asses WCT2M, analytic tools were elaborated: WCTMS simulator and the specially created laboratory. They were described in Section 6. In Section 7 the results of experimental evaluation of WCT2M were presented. They were obtained using previously described tools. Finally, Section 8 contains comparison of WCT2M with other trust management methods described in the literature.

It was demonstrated that WCT2M allows to effectively and efficiently manage trust in sensor networks. It allows to recognize untrusted nodes and prevent information from these nodes to spread in the network, using reasonable amount of resources.

Consequently, the thesis statement formulated as:

> ***The proposed method allows to effectively and efficiently manage trust in wireless sensor networks***

has been justified.

9.3 Influence on security level

As described in Section 5.2.5, WCT2M cooperates with security mechanisms implemented on the nodes. They assess the incoming messages and submit the results to the trust management mechanism. After collecting these data, the trust management mechanism decides if the corresponding trust value should be lowered or increased. During the experiments the effectiveness of the security mechanisms was represented by parameter *r* (see Table 7).

The experiments show that security mechanisms in connection with WCT2M allow to protect the network and proved that effectiveness of security mechanisms is highly connected with effectiveness of the method. WCT2M does not have negative influence on network protocols (e.g. choosing the cluster-head), because it is decentralized (cluster heads does not have any special role in trust management) and the method works on the top of the stack of protocols. Moreover, the method can improve the work of these protocols, e.g. by taking into account the trust value to the vote influence while voting on a new cluster-head.

As described in previous sections, trust management systems, including WCT2M, are not immune to security threats themselves. However, in most cases the usage of trust management is reasonable in WSN. Lopez et al. [106] justify it as a way of providing a satisfactory solution to the problem of uncertainty. While it is not possible to know the future in an accurate way, the past actions of the nodes are reflected in the reputation and trust values. If a node behaved satisfactorily in the past performing a certain task, it is assumed that it will be reliable in the future performing the same task. As a result, a node can start cooperation with the most reliable nodes.

9.4 Dissemination of the results

The work presented in this dissertation has been published in the proceedings of six conferences. These publications are summarised below:

- [60] gives an introduction to the problem of distributed trust management in Wireless Sensor Networks. Basic concepts and definitions are explained and trust management model is presented. It also introduces WCTMS and its concepts and presents simulation results achieved.
- [117] is an extension of [60]. It presents details of the previously proposed model and further simulation results.
- [118] presents a two-tier trust management model for WSN. The model assumes that the nodes assess trustworthiness of other nodes based on the mutual observation and recommendations. By dividing the network structure into two tiers, the nodes can operate longer and more effectively.
- [119] presents a case study related to WSN application in the e-health domain. It was assumed that the network implements WCT2M which leads to detection and isolation of sensors violating the network policies.
- [120] is a continuation of [119]. To measure the effectiveness of such detection a set of metrics was derived in a systematic way, using Goal-Question-Metrics approach. The network was simulated with the help of WCTMS and the resulting data were used to obtain values of the metrics which demonstrate how effectively the broken nodes are eliminated from the network.
- [46] presents the results of the time effectiveness assessment of WCT2M in a fully synchronized Wireless Sensor Network. It introduces some basic types of synchronization patterns in WSN based on the idea of sleep scheduling, then explains how WCT2M works in the network applying the fully synchronized sleep scheduling pattern. Such networks were subjected to the analyses with the help of WCTMS to investigate the time delays needed to identify and isolate the network nodes which depart from the assumed behavioural characteristics. The results of these simulations were presented to demonstrate the time effectiveness of WCT2M.

Table 33 shows how the above publications are related to the chapters of this dissertation.

Table 33 Mapping from the publications to the chapters

Publication	Chapters
[60]	5, 5.2.4, 6.2.3
[117]	5.1
[118]	2.3, 7.2.4
[119]	3

[120]	3, 7.1.1, 7.2.2
[46]	2.4, 7.2.3

The paper summarizing all results acquired in the course of this research is currently under preparation with the intention of submitting it to an international journal.

9.5 Directions of further research

In the scope of the further research there is investigation of networks with mobile nodes. Such behaviour enforces frequent clusters reformation. It can allow malicious nodes to change clusters and access only these where they have higher trust value.

It is also planned to research gossip problem. Every node is a neighbour of only few nodes. After few WCT2M cycles every node in the network know each other because of exchanging trust tables. It means the most of the nodes know other only from kind of 'gossip'. But this indirect knowledge is exchanged also in second direction – towards assessed node. It is interesting to examine, how this 'feedback' of a gossip information influence the efficiency of the method and is it possible to use this feature by malicious node to attack WCT2M.

9.6 Epilogue

As wireless sensor networks become more complex and provide more sophisticated services, the problem of security becomes more important. Dependability of such networks becomes a difficult issue as in addition to technical imperfections and human faults, malicious actions have to be taken into account.

One of the solutions to this problem is trust management. It allows to distinguish between trustworthy and untrustworthy nodes which enables collaborative decisions leading to isolation and exclusion of the nodes with a very low level of trust. It allows to improve the security of the network using fewer resources comparing to security mechanisms used in conventional networks.

Bibliography

[1] P. Coy and N. Gross, "21 Ideas for the 21st Century," *Business Week,* p. 82, 30 08 1999.

[2] L. M. Ni, "China's national research project on wireless sensor networks," *Proc. of the 2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'08),* 2008.

[3] Q. Wang and I. Balasingham, "Wireless Sensor Networks - An Introduction," *Wireless Sensor Networks: Application-Centric Design,* 2010.

[4] I. Wikimedia Foundation, "Wireless sensor network," [Online]. Available: http://en.wikipedia.org/wiki/Wireless_sensor_network. [Accessed 17 05 2014].

[5] N. P. Mahalik, "Preface," in *Sensor Networks and Configuration*, Springer, 2007.

[6] I. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A survey on sensor networks," *Communications Magazine,* vol. 40, no. 8, 2002.

[7] D.-S. Kim and S. Y. Shin, "Wireless Pet Dog Management Systems," in *Sensor Networks and Configuration*, Springer, 2007.

[8] Y. Wang and Y. Tseng, "Attacks and Defenses of Routing Mechanisms in Ad Hoc and Sensor Networks," 2006.

[9] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe and I. Seskar, "Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study," *USENIX Security 2010,* 2010.

[10] Y. Zhiyng, K. Daeyoung, L. Insun, K. Kiyoung and J. Jongsoo, "A Security Framework with Trust Management for Sensor Networks," *IEEE SecureComm,* 5–9 Sept. 2005.

[11] Institute of Electrical and Electronics Engineers, Inc., "Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs)," IEEE Press, New York, 2003.

[12] ZigBee Alliance, "ZigBee Home Automation Public Application Profile specification," 2007.

[13] J. Górski et al., "Angel System Specification," *"Angel project report: Angel System Specification", Deliverable 1.2., ANGEL Project,* 2007.

[14] Eclipse Foundation, "Eclipse," [Online]. Available: http://www.eclipse.org/. [Accessed 03 07 2014].

[15] mspgcc Community, "mspgcc," [Online]. Available: http://sourceforge.net/apps/mediawiki/mspgcc/index.php?title=MSPGCC_Wiki. [Accessed 04 10 2013].

[16] Oracle Corporation, "Welcome to NetBeans," [Online]. Available: https://netbeans.org.

[Accessed 05 02 2015].

[17] Oracle, “Java,” [Online]. Available: http://www.java.com. [Accessed 06 07 2014].

[18] JGraph Ltd, “JGraph,” [Online]. Available: http://www.jgraph.com/. [Accessed 06 07 2014].

[19] Silicon Labs Inc., “The Evolution of Wireless Sensor Networks,” [Online]. Available: http://www.silabs.com/Support%20Documents/TechnicalDocs/evolution-of-wireless-sensor-networks.pdf. [Accessed 18 01 2015].

[20] C. Y. Chong and S. P. Kumar, “Sensor networks: Evolution, opportunities, and challenges,” *Proceedings of the IEEE ,* vol. 91, no. 8, 2003.

[21] S. Kumar and D. Shepherd, “Sensit: Sensor information technology for the warfighter,” *Proc. of the 4th International Conference on Information Fusion (FUSION’01),* 2001.

[22] Urząd Publikacji Unii Europejskiej, “Advanced Networked embedded platform as a Gateway to Enhance quality of Life,” [Online]. Available: ftp://ftp.cordis.europa.eu/pub/ist/docs/dir_c/ems/angel-v1.pdf. [Accessed 07 03 2015].

[23] J. A. Stankovic, “Wireless Sensor Networks,” 2006.

[24] H. Deng, W. Li and D. Agrawal, “Routing security in wireless ad hoc networks,” *IEEE Comm. Magazine,* vol. 40, no. 10, 2002.

[25] J. Hill and D. Culler, “A wireless embedded sensor architecture for system-level optimization,” UC Berkeley, 2002.

[26] E. P. De Freitas, Cooperative Context Aware Setup and Performance of Surveillance Missions Using Static and Mobile Wireless Sensor Networks, 2011.

[27] J. L. Hill, 2003.

[28] J. Adams, “Meet the ZigBee Standard,” [Online]. Available: http://www.sensorsmag.com/sensors-mag/meet-zigbee-standard-733. [Accessed 25 10 2014].

[29] O. Younis, M. Krunz and S. Ramasubramanian, “Node Clustering in Wireless Sensor Networks: Recent Developments and Deployment Challenges,” *Network, IEEE ,* vol. 20, 2006.

[30] W. Heinzelman, A. Chandrakasan and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks,” *Hawaii International Conference on System Sciences,* 2000.

[31] M. J. Handy and M. Hass, “Low Energy Adaptive Clustering Hierarchy with Deterministic Cluster-Head Selection,” *roc. 4th IEEE International Workshop on Mobile and Wireless Communications Network (MWCN '02),* 2002.

[32] V. Kumar, S. Jain and S. Tiwari, “Energy Efficient Clustering Algorithms in Wireless Sensor Networks: A Survey,” *IJCSI International Journal of Computer Science Issues,* vol. 8, no. 2, 2011.

[33] W. Heinzelman, A. Chandrakasan and H. Balakrishnan, “An Application-Specific Protocol Architecture for Wireless Microsensor Networks,” *IEEE Trans. Wireless Commun,* vol. 1, no. 4,

2002.

[34] M. I. Brownfield, K. Mehrjoo, A. S. Fayez and N. J. Davis IV, “Wireless Sensor Network Energy-Adaptive MAC Protocol,” *IEEE CCNC,* 2006.

[35] V. Rajendran, K. Obraczka and J. Garcia-Luna-Aceves, “Energy-efficient MAC: energy-efficient collision-free medium access control for wireless sensor networks,” *SENSYS,* 2003.

[36] Y. Wei, J. Heidemann and D. Estrin, “An energy-efficient MAC protocol for wireless sensor networks,” *INFOCOMM,* 2002.

[37] T. van Dam and K. Langendoan, “Energy-efficient MAC: An adaptive energy-efficient MAC protocol for wireless sensor networks,” *ACM SENSYS,* 2003.

[38] G. Lu, B. Krishnamachari and S. Raghavendra, “An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks,” *Proc. Parallel and Distributed Processing Symposium,* 2004.

[39] O. Incel, A. Ghosh and B. Krishnamachari, “Scheduling Algorithms for Tree-Based Data Collection in Wireless Sensor Networks,” *Theoretical Aspects of Distributed Computing in Sensor Networks,* 2011.

[40] A. Keshavarzian, H. Lee and L. Venkatraman, “Wakeup Scheduling in Wireless Sensor Networks,” *Proceeding of the 7th ACM International Symposium on Mobile ad hoc networking and computing,* 2006.

[41] J. D. Angelin and X. V. Arul, “Optimizing Delay For Critical Event Monitoring In Wireless Sensor Networks,” *International Journal of Engineering Research & Technology,* vol. 2, 2013.

[42] M. Sichitiu and C. Veerarittiphan, “Simple, Accurate Time Synchronization for Wireless Sensor Networks,” *IEEE Wireless Communications And Networking,* vol. 2, 2003.

[43] S. Kumar and S. Chauhan, “A Survey on Scheduling Algorithms for Wireless Sensor Networks,” *International Journal of Computer Applications,* no. 5, 2011.

[44] G. Lu, B. Krishnamachari and C. Raghavendra, “An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks,” *IEEE Proceedings of the 18th International Parallel and Distributed Processing Symposium,* 2004.

[45] J. Górski et al., “Description of Final ANGEL Demonstrator,” *"Angel project report", Deliverable 5.2., ANGEL Project,* 2007.

[46] J. Górski and A. Turower, “Assessing the time effectiveness of trust management in fully synchronised wireless sensor networks,” *SCR 2013,* 2013.

[47] J. Walters, Z. Liang, W. Shi and V. Chaudhary, “Wireless Sensor Network Security: A Survey,” in *Security in Distributed, Grid, and Pervasive Computing*, Auerbach Publications, CRC Press, 2006.

[48] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen and D. E. Culler, “Spins: security protocols for sensor networks,” *Wireless Networking,* vol. 8, no. 5, 2002.

[49] A. Pathan, H. Lee and C. Hong, “Security in Wireless Sensor Networks: Issues and Challenges,” *International Conference on Advanced Communication Technology,* 2006.

[50] J. Sen, “A Survey on Wireless Sensor Network Security,” *International Journal of Communication Networks and Information Security,* 2009.

[51] M. Prasant and K. Srikanth, “Ad Hoc Networks Technologies and Protocols,” Springer, 2004.

[52] S. Sancak, E. Cayirci, V. Coskun and A. Levi, “Sensor wars: detecting and defending against spam attacks in wireless sensor networks,” *IEEE International Conference on Communications,* vol. 1, 2004.

[53] S. Marti, T. J. Giuli, K. Lai and M. Baker, “Mitigating routing misbehavior in mobile ad hoc networks,” *ACM/IEEE Internationa Conference on Mobile Computing and Networking,* vol. 1, 2000.

[54] I. Avramopoulos, H. Kobayashi, R. Wang and A. Krishnamurthy, “Highly secure and efficient routing,” *IEEE INFOCOM,* vol. 1, 2004.

[55] S. Sivanantham, K. Kirankumar and V. Akshaya, “Detection and Avoidance of Intrusion, Packet Drop and Modification in WSN,” *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 2, no. 11, 2013.

[56] S. Vijayalakshmi, R. Kurinjimalar and S. Prakash, “Detection of Packet Dropping and Modification in Wireless Sensor Network,” *International Journal of Computer Science & Engineering Technology,* vol. 4, no. 4, 2013.

[57] C. U. Press, “Cambridge Advanced Learner's Dictionary,” [Online]. Available: http://dictionary.cambridge.org/. [Accessed 19 03 2011].

[58] N. I. Batt, “Operational Trust: A New Look at the Human Requirement in Network Centric Warfare,” *9th International Command and Control Research and Technology,* 2004.

[59] H. Baldus et al., “WSN Trust Management Model,” *"Angel project report: WSN trust architecture and security protocols", Deliverable 3.2., ANGEL Project,* 2007.

[60] J. Górski, A. Turower and A. Wardziński, “Distributed Trust Management Model for Wireless Sensor Networks,” *DepCoS,* 2011.

[61] Y. Sun, Z. Han and K. Liu, “Defense of Trust Management Vulnerabilities in Distributed Networks,” *Communications Magazine,* no. 46, 2008.

[62] R. Abassi and S. G. El Fatmi, “Securing a Trust-based MANET against Collusion Attacks,” *SAFECOMP,* vol. 1, 2014.

[63] M. Theoharidou, P. Kotzanikolaou and D. Gritzalis, “A multi-layer criticality assesment methodology based on interdependencies,” *Computers & Security,* vol. 6, no. 29, 2010.

[64] C. Lambrinoudakis, S. Gritzalis, F. Dridi and G. Pernul, “Security requirements for e-goverment services: A metholodical approach for developing a common based PKI-security policy,” *Computer Communications,* no. 26, pp. 1873-1883, 2003.

[65] ISO and IEC, ISO/IEC 27002:2005 - Information technology - security techniques - Code of practise for information security management, 2005.

[66] B. Księżopolski, Z. Kotulski and P. Szałachowski, "Adaptive approach to network security," in *Computer Networks*, Springer, 2009, pp. 233-241.

[67] B. Księżopolski and Z. Kotulski, "Adaptable security mechanism for dynamic environments," *Computers & Security,* pp. 246-255, 2007.

[68] S. Lindskog, Modelling and Tunning Security form Quality of Service Perspective - PhD Thesis, Chalmers University of Technology, 2005.

[69] Texas Instruments, "CC2520 Development Kit," [Online]. Available: http://www.ti.com/tool/cc2520dk. [Accessed 03 07 2014].

[70] Texas Instruments, "CC2520 Development Kit User's Guide," [Online]. Available: http://www.ti.com/lit/ug/swru138/swru138.pdf. [Accessed 03 07 2014].

[71] M. Kurowski, "Zarządzanie zaufaniem jako metoda zabezpieczenia rozproszonych bezprzewodowych sieci sensorów przed atakami," 2013.

[72] Texas Instruments, "Code Composer Studio IDE," [Online]. Available: http://www.ti.com/tool/ccstudio. [Accessed 04 10 2013].

[73] IAR Systems, "IAR Workbench IDE dla Texas Instruments MSP430," [Online]. Available: http://www.iar.com/en/Products/IAR-Embedded-Workbench/TI-MSP430/. [Accessed 04 10 2013].

[74] codeforge.com, "Vlo_and library," [Online]. Available: http://www.codeforge.com/read/146658/vlo_rand.h__html. [Accessed 09 03 2014].

[75] Texas Instruments, "HAL Library and example applications for MSP430," [Online]. Available: http://www.ti.com/litv/zip/swrc090b. [Accessed 09 03 2014].

[76] OMNeT++ Commnunity, "OMNeT++ Simulation Framework," [Online]. Available: http://omnetpp.org/. [Accessed 29 06 2014].

[77] NesCT Community, "NesCT," [Online]. Available: http://nesct.sourceforge.net/. [Accessed 29 06 2014].

[78] PAWiS Community, "PAWiS Simulation Framework," [Online]. Available: http://pawis.sourceforge.net/. [Accessed 29 06 2014].

[79] Australia's Information Communications Technology Research Centre of Excellence, "Castalia Wireless Sensor Network Simulator," [Online]. Available: https://castalia.forge.nicta.com.au/index.php/en/. [Accessed 29 06 2014].

[80] IAR Systems, "IAR Embedded Workbench," [Online]. Available: http://www.iar.com/. [Accessed 29 06 2014].

[81] WSNSim Community, "WSNSim," [Online]. Available: http://sourceforge.net/projects/wsnsim/.

[Accessed 29 06 2014].

[82] NS-2 Community, “NS-2,” [Online]. Available: http://nsnam.isi.edu/nsnam. [Accessed 29 06 2014].

[83] Wikimedia Foundation, “ns (simulator),” [Online]. Available: http://en.wikipedia.org/wiki/Ns_(simulator). [Accessed 29 06 2014].

[84] NS-3 Community, “NS-3,” [Online]. Available: http://www.nsnam.org/. [Accessed 29 06 2014].

[85] TOSSIM Community, “TOSSIM,” [Online]. Available: http://tinyos.stanford.edu/tinyos-wiki/index.php/TOSSIM. [Accessed 29 06 2014].

[86] F. Yu, “A Survey of Wireless Sensor Network Simulation Tools,” [Online]. Available: http://www.cse.wustl.edu/~jain/cse567-11/ftp/sensor/index.html. [Accessed 29 06 2014].

[87] J-Sim Community, “J-Sim,” [Online]. Available: https://sites.google.com/site/jsimofficial/. [Accessed 29 06 2014].

[88] atemu community, “atemu,” [Online]. Available: http://www.hynet.umd.edu/research/atemu/. [Accessed 29 06 2014].

[89] UCLA Compilers Group, “avrora,” [Online]. Available: http://compilers.cs.ucla.edu/avrora/. [Accessed 29 06 2014].

[90] Tetcos, “NetSim - network simulator,” [Online]. Available: http://tetcos.com/netsim_gen.html. [Accessed 29 06 2014].

[91] Wikimedia Foundation, “NetSim,” [Online]. Available: http://en.wikipedia.org/wiki/NetSim. [Accessed 29 06 2014].

[92] riverbed, “Reverbed Modeler,” [Online]. Available: http://www.riverbed.com/products/performance-management-control/network-performance-management/network-simulation.html. [Accessed 29 06 2014].

[93] D. Quaglia ed., “Complete co-simulation framework and refined models of the components of the Angel platform,” *"Angel project report", Deliverable 2.3., ANGEL Project,* 2008.

[94] R. van Solingen and E. Berghout, The Goal/Question/Metric method: A practical guide for quality improvement of software development, McGraw-Hill Publishing Company, 1999.

[95] E. Bulut, Z. Wang and B. K. Szymanski, “The Effect of Neighbor Graph Connectivity on Coverage Redundancy in Wireless Sensor Networks,” *Communications (ICC), 2010 IEEE International Conference on,* pp. 1 - 5, 2010.

[96] F. Delicato, F. Protti, J. F. De Rezende, L. Rust and L. Pirmez, “Application-driven node management in multihop wireless sensor networks,” *Lecture Notes in Computer Science,* pp. 569-576, 2005.

[97] A. Jhumka and L. Mottola, “On Consistent Neighborhood Views in Wireless Sensor Networks,” *Reliable Distributed Systems, 2009. SRDS '09. 28th IEEE International Symposium on,* pp. 199 -

208, 2009.

[98] H. Seltman, Experimental Design for Behavioral and Social Sciences, 2014.

[99] M. L. Mitchell and J. M. Jolley, Research design explained, Wadsworth, 2013.

[100] R. Perlman, "An overview of PKI trust models," *IEEE Network,* vol. 13, no. 6, 1999.

[101] H. Chen, H. Wu, X. Zhou and C. Gao, "Agent-based Trust Model in Wireless Sensor Networks," *8th ACIS Int.Conf. on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing,* 2007.

[102] H. Chen, H. Wu, X. Zhou and C. Gao, "Reputation-based Trust in Wireless Sensor Networks," *International Conference on Multimedia and Ubiquitous Engineering MUE'07,* 2007.

[103] A. Boukerche and X. Li, "An Agent-based Trust and Reputation Management Scheme for Wireless Sensor Networks," *IEEE Globecom 2005,* 28 Nov – 2 Dec 2005.

[104] T. Zahariadis, H. C. Leligou, P. Trakadas and S. Voliotis, "Trust management in wireless sensor networks," *InterScience,* 2010.

[105] Y. Yu, K. Li, W. Zhou and P. Li, "Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures," *Journal of Network and Computer Applications,* vol. 35, 2012.

[106] J. Lopez, R. Roman, I. Agudo and C. Fernandez-Gago, "Trust management systems for wireless sensor networks: Best practices," *Computer Communications,* vol. 33, 2010.

[107] F. Buccafurri, L. Coppolino, S. D'Antonio, A. Garofalo, G. Lax, A. Nocera and L. Romano, "Trust-Based Intrusion Tolerant Routing in Wireless Sensor Networks," *SAFECOMP,* vol. 1, 2014.

[108] M. Maroti, "The Flooding Time Synchronization Protocol," *proc. of 2nd ACM Conference on Embedded Networked Sensor Systems,* 2004.

[109] V. Loscri, "A Two-Levels Hierarchy for Low-Energy Adaptive Clustering Hierarchy (TL-LEACH)," *Vehicular Technology Conference,* 2005.

[110] T. Zia, "Reputaton-based Trust Managmenet in Wireless Sensor Networks," *Intelligent Sensors, Sensor Networks and Information Processing,* 2008.

[111] F. Song and B. Zhao, "Trust-based LEACH Protocol for Wireless Sensor Networks," *Future Generation Communication and Networking FGCN '08,* 13–15 Dec 2008.

[112] M. Momani and S. Challa, "Trust Management in Wireless Sensor Networks," *Proc. of 5th ACM Conf. on Embedded Networked Sensor Systems,,* November 6 – 9, 2007.

[113] A. Josang and R. Ismail, "The BetaReputation System," *15th Bled Electronic Commerce Conference,* 2002.

[114] R. Feng, X. Xu, X. Zhou and J. Wan, "A Trust Evaluation Algorithm for Wireless Sensor Networks Based on Node Behaviors and D-S Evidence Theory," *Sensors,* vol. 11, no. 2, 2011.

[115] F. Bao, I.-R. Chen, J.-H. Cho and J.-H. Cho, "Hierarchical Trust Management for Wireless Sensor Networks and its Applications to Trust-Based Routing and Intrusion Detection," *IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT,* pp. 169-183, 6 2012.

[116] "Evolution of wireless sensor networks towards the Internet of Things: A survey," in *Software, Telecommunications and Computer Networks (SoftCOM), 2011 19th International Conference on*, Split, IEEE, 2011, pp. 1-6.

[117] J. Górski and A. Turower, "Using trust management model for detection of faulty nodes in Wireless Sensor Networks," in *ICT Young vol. 1*, Gdańsk, Politechnika Gdańska, 2011, pp. 471-476.

[118] J. Górski and A. Turower, "Two-tier distributed trust management model for wireless sensor networks," *II Ogólnopolskie Seminarium: Forum Innowacji Młodych Badaczy,* 2011.

[119] J. Górski and A. Turower, "Zarządzanie zaufaniem w bezprzewodowych sieciach czujników - studium przypadku," in *Studia Informatica vol. 33*, Gliwice, Wydawnictwo Politechniki Śląskiej, 2012, pp. 125-137.

[120] J. Górski and A. Turower, "Trust management in WSN – case study evaluation," in *ICT Young vol. 2*, Gdańsk, Politechnika Gdańska, 2015, pp. 581-588.

[121] A. Mishra, "On Scheduling Guaranteed Time Slots for Time Sensitive Transactions in IEEE 802.15.4 Networks," *Military Communications Conference,* 2007.

[122] H. Labiod, H. Afifi and C. De Santis, Wi-Fi™, Bluetooth™, Zigbee™ and WiMax™, Springer, 2007.

[123] K. Chan, L. K. Yeung and W. Shao, "Contention-based MAC protocols with erasure coding for wireless data networks," *Ad Hoc Networks,* vol. 3, no. 4, 2005.

[124] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 3, 2002.

Appendix I: WCT2M simulator user guide

WCT2M simulator (WCTMS) is written in Java 1.7 and does not have any GUI. Every modification is set in the source code and then the simulator is run. To use WCTMS follow these steps:

1. Import source code to the chosen Integrated Development Environment (IDE) or modify files in any text editor. The source code is delivered with NetBeans IDE project.
2. Modify settings (final static parameters defined in the beginning of a file) in:
    a. Main.java file: general setting like simulated network sizes and number of simulations;
    b. Simulation.java file: specific simulation settings.

   Every setting is commented to facilitate modifications.
3. In any place of Simulation.java file a call to the following methods can be added:
    a. `printSituation()` displays on standard output current values of trust tables for each network node;
    b. `System.out.print(node)` displays on standard output current value of the trust table for the given node;
    c. `new Graph(nodes)` displays new Swing window showing how nodes are scattered on a simulation are and how are currently connected.
4. Run simulator. In NetBeans IDE it can be done by clicking F6 button. Results will be displayed on standard output.

Appendix II: Metoda zarządzania zaufaniem w bezprzewodowych sieciach czujników

Rozszerzone streszczenie

A.1. Wprowadzenie

Bezprzewodowa Sieć Czujników (ang. *Wireless Sensor Network*, WSN) to sieć autonomicznych czujników rozmieszczonych w przestrzeni w celu monitorowania warunków fizycznych lub środowiskowych, takich jak temperatura, ciśnienie, dźwięk itp. oraz wspólnego przekazywania danych do głównej lokalizacji. Bardziej zaawansowane sieci są dwukierunkowe, a także umożliwiają sterowanie aktywnością czujnika.

Pierwszą siecią bezprzewodową, była Sound Surveillance System (SOSUS), opracowana przez wojsko Stanów Zjednoczonych w 1950 roku w celu wykrywania i śledzenia radzieckich okrętów podwodnych. Sieć składała się z zanurzonych czujników akustycznych rozmieszczonych w Atlantyku i Pacyfiku. Ta technologia służy do dziś, choć pełni bardziej pokojowe funkcje monitoringu podmorskiej fauny i aktywności wulkanicznej [19].

Rządy i uczelnie zaczęły używać bezprzewodowych sieci czujników w monitoringu jakości powietrza, wykrywaniu pożarów lasów, zapobieganiu katastrofom naturalnym, na stacjach meteorologicznych i do monitoringu strukturalnego. Następnie giganci technologiczni, tacy jak IBM i Bell Labs, zaczęli promowanie wykorzystania bezprzewodowych sieci czujników w zastosowaniach przemysłowych, przykładowo w dystrybucji energii, oczyszczaniu ścieków i wyspecjalizowanej automatyce przemysłowej [19].

Najnowsze osiągnięcia w informatyce, komunikacji i technologiach elektromechanicznych spowodowały znaczące zmiany w badaniach nad bezprzewodowymi sieciami czujników. Badania koncentrowały się na technikach sieciowych i przetwarzaniu informacji w sieci, wspierających bardzo dynamiczne środowiska ad hoc i czujniki o ograniczonych zasobach. Ponadto węzły stały się znacznie mniejsze (zaczynając od rozmiaru talii kart kończąc na cząstce pyłu) i znacznie tańsze, a tym samym pojawiło się wiele nowych zastosowań dla cywilnych sieci czujników, np. monitorowanie środowiska, sieci czujników w pojazdach i sieci czujników na ciele ludzkim [3]. Następnie amerykańska agencja rządowa Defense Advanced Research Projects Agency (DARPA) rozpoczęła program badawczy nazwany SensIT, którego rezultaty wzbogaciły WSN o nowe możliwości takie jak sieci ad hoc, dynamiczne zapytania i zadania, możliwość przeprogramowania oraz wielozadaniowość [21]. W tym

samym czasie organizacja IEEE zauważyła niski koszt i możliwości, które oferują bezprzewodowe sieci czujników. Organizacja zdefiniowała standard IEEE 802.15.4 dla bezprzewodowych sieci osobistych niskiej przepływności. W oparciu o IEEE 802.15.4 ZigBee Alliance opublikowała standard ZigBee, który określa zestaw protokołów komunikacyjnych wysokiego poziomu, który może być używany w bezprzewodowych sieciach czujników [3].

Obecnie bezprzewodowe sieci czujników postrzegane są jako jedna z najważniejszych technologii XXI wieku [1]. Unia Europejska wspiera programy związane z wykorzystaniem WSN. Przykładem może być projekt ANGEL [13], który miał na celu dostarczenie metod i narzędzi do tworzenia złożonych systemów heterogenicznych, w których bezprzewodowe sieci czujników i tradycyjne sieci komunikacyjne współpracują w celu monitorowania i poprawy jakości życia [22]. Chiny umieściły WSN w swoim badawczym programie strategicznym [2]. Komercjalizacja bezprzewodowych sieci czujników jest również napędzana przez przedsiębiorstwa [3].

Sieci czujników stają się coraz bardziej złożone i zapewniają coraz bardziej zaawansowane usługi. Różnorodność ról węzłów sieci rośnie, oprócz prostych węzłów wyszczególniamy: routery, głowy (ang. *heads*) i stacje bazowe. Takie węzły, których zadania wykraczają poza wykrywanie parametrów środowiska, mogą mieć znaczną moc obliczeniową i realizować zaawansowane zadania. Ponieważ wielkość i złożoność sieci rośnie, zarządzanie nimi staje się coraz trudniejsze, w szczególności problematyczne staje się pogodzenie bezpieczeństwa z wydajnością i elastycznością. Ponadto poszczególne węzły lub podsieci mogą być zarządzane przez różne osoby lub organizacje. Niezawodność takich sieci staje się trudnym zagadnieniem, oprócz wad technicznych i ludzkich, należy brać także pod uwagę celowe działania destrukcyjne.

Bezprzewodowe sieci czujników są z wielu powodów narażone na ataki. Głównymi problemami są [8] [51]:

- Nie ma potrzeby fizycznego dostępu do urządzenia, aby się z nim połączyć. Atakujący może bezprzewodowo dotrzeć do węzła i go zaatakować. Brak scentralizowanej infrastruktury wymusza implementację mechanizmów bezpieczeństwa w każdym z węzłów sieci.
- Autonomia węzłów – węzły samodzielnie podejmują decyzję o routingu i przetwarzaniu danych, co zwiększa ryzyko wycieku danych w razie fizycznego przejęcia i przeprogramowania węzła. Ponadto, im większa sieć, tym trudniejsze jest śledzenie i monitorowanie pojedynczego węzła.
- Decentralizacja podejmowania decyzji – brak centralnego ośrodka decyzyjnego umożliwia atakującemu zastosowanie technik służących przełamaniu algorytmów współpracy.

- Otwarta struktura sieci – każde urządzenie znajdujące się w zasięgu choć jednego z węzłów może zainicjować procedurę łączenia się z siecią. Umożliwia to urządzeniom niespełniającym polityki sieci wpływanie na działanie sieci.
- Ograniczenia fizyczne urządzeń – węzły WSN są małymi urządzeniami, zwykle wyposażonymi w wewnętrzne źródło zasilania, mają zazwyczaj niską mocą obliczeniową, niewielką pamięć i małe zasoby transmisyjne. W związku z tym niemożliwe jest zastosowanie tych samych zaawansowanych metod kryptograficznych, które używane są w sieciach bezprzewodowych tworzonych dla bardziej zaawansowanych urządzeń (na przykład w standardzie IEEE 802.11).
- Eliminacja pewnych rozwiązań zabezpieczających, np. bazujących na konfiguracji statycznej. Ze względu na mobilność WSN i ciągle zmieniającą się topologię sieci, węzły muszą stale wykrywać i oceniać nowe węzły pojawiające się w ich zasięgu.

A.2. Cel rozprawy

Liczba wdrożonych bezprzewodowych sieci czujników rośnie. Obecnie są to nie tylko demonstracje naukowe, ale również duże sieci przemysłowe. Zakres wykorzystania jest niezwykle szeroki: od zastosowań w celach wojskowych do cywilnych, takich jak monitorowanie zdrowia [6]. Pierwsze systemy czujników były realizowane jako jedno urządzenie, które przesyłało zmierzoną wartość z monitorowanego obiektu do odbiornika bez wykorzystania połączenia przewodowego. Postęp w konstruowaniu czujników spowodował powstanie nowej kategorii sieciowych systemów wbudowanych – bezprzewodowe sieci czujników zbierają dane i dostarczają je do systemów kontroli zarządczej (ang. *management control systems*). Systemy te mogą być stosowane do pomiaru, przetwarzania danych i komunikowania się z każdym węzłem sieci czujników przy wykorzystaniu komunikacji bezprzewodowej. WSN mogą wspierać wyrafinowane operacje i pracować w inteligentnym otoczeniu [7].

Te złożone systemy są często budowane bez zastosowania żadnych środków bezpieczeństwa lub z wieloma lukami bezpieczeństwa [8] [9]. Co więcej, wiele rozwiązań proponowanych dla sieci przewodowych jest niewłaściwych dla bezprzewodowych sieci czujników lub nie mogą być bezpośrednio zastosowane [8]. Zapewnienie bezpieczeństwa sieci czujników przez proste kopiowanie najlepszych praktyk z tradycyjnych sieci nie jest możliwe, ponieważ węzły czujników są ograniczone swoimi zasobami, przez co nie wspierają zaawansowanych mechanizmów bezpieczeństwa.

Aby poradzić sobie z tym problemem, zostały wprowadzone pojęcia zaufania i wiarygodności. Obiekt A *ufa* obiektowi B, jeśli A czyni pozytywne założenia dotyczące stanu i zachowania B (na przykład A zakłada, że dane wysyłane przez B są prawdziwe). B jest *wiarygodny* jeśli A dysponuje

dowodami wystarczającymi, aby uzasadnić swoje zaufanie do B. Zgodnie z tymi definicjami *zarządzanie zaufaniem* jest rozumiane jako zbieranie dowodów dotyczących wiarygodności i na tej podstawie podejmowanie decyzji o zaufaniu.

W sieciach czujników zarządzanie zaufaniem ma wielkie znaczenie, ponieważ pomaga rozróżnić zaufane i niezaufane węzły. Umożliwia to podejmowanie wspólnych decyzji prowadzących do izolacji i wykluczenia węzłów z bardzo niskim poziomem zaufania. Takie działanie poprawia bezpieczeństwo sieci przy użyciu mniejszej ilości zasobów w stosunku do mechanizmów zabezpieczeń stosowanych w tradycyjnych sieciach.

A.3. Teza rozprawy

Celem niniejszej pracy jest opracowanie nowego modelu zarządzania zaufaniem dla rozproszonych bezprzewodowych sieci czujników (WSN) i przeanalizowanie jego skuteczności i wydajności. Proponowana metoda powinna wykrywać węzły nieprzestrzegające polityki sieci, oceniać je jako niezaufane i zapobiegać rozprzestrzenianiu się w sieci informacji pochodzących od tych węzłów.

Teza rozprawy została sformułowana następująco:

> ***Proponowana metoda umożliwia skuteczne i wydajne zarządzanie zaufaniem w bezprzewodowych sieciach czujników.***

A.4. Znaczenie podjętego problemu

Jak wspomniano wcześniej, bezprzewodowe sieci czujników stają się wystarczająco zaawansowane, by zyskać istotne znaczenie w wielu obszarach zastosowań. Chociaż te sieci współdzielą szereg wymogów bezpieczeństwa z tradycyjnymi sieciami, ze względu na ich szczególne ograniczenia mogą wymagać innych rozwiązań. Przede wszystkim są one ograniczone zasobami takimi jak pamięć, zasilanie i zdolności obliczeniowe. Ograniczenia zasobów wynikają z konieczności zapewnienia niskich kosztów urządzeń. To zawęża stosowanie konwencjonalnych mechanizmów ochrony (np. zaawansowanych systemów kryptograficznych) i utrudnia stosowanie kompleksowych rozwiązań bezpieczeństwa [10].

Co więcej, protokoły stosowane w WSN są budowane bez poświęcania większej uwagi problemom związanym z bezpieczeństwem. *ZigBee*, jedna z najbardziej popularnych specyfikacji protokołów komunikacyjnych, może służyć za przykład. ZigBee został opracowany przez ZigBee Alliance, stowarzyszenie wspólnie pracujących firm w celu opracowania standardów (i produktów)

dla niezawodnych, oszczędnych i zapewniających niski pobór mocy sieci bezprzewodowych. ZigBee jest osadzony w szerokiej gamie produktów i aplikacji dla rynków konsumenckich, handlowych, przemysłowych i rządowych na całym świecie. ZigBee opiera się na standardzie IEEE 802.15.4 [11], który określa warstwę fizyczną (ang. *Physical*) i warstwę kontroli dostępu do medium (ang. *Media Access Control; MAC*), potrzebne do stworzenia tanich sieci osobistych (ang. *Personal Area Networks; PAN*). Specyfikacja ZigBee zapewnia usługę bezpieczeństwa, która umożliwia ochronę przesyłanych danych, jednak nie rozwiązuje wielu problemów bezpieczeństwa:

- Aplikacje na jednym węźle ZigBee nie są rozdzielone, w związku z czym każda aplikacja może mieć dostęp do całego węzła i wywołać każdą procedurę. Niższe warstwy ZigBee są w pełni dostępne dla wszystkich zainstalowanych na danym węźle aplikacji. Nie można zatem wykluczyć skorumpowania oprogramowania węzła sieci i spowodowania, że nieuprawniona aplikacja będzie w stanie wysyłać wiadomości używając zaufanych kluczy bezpieczeństwa [12].
- Niektóre dane (na przykład dane medyczne) mogą nie być prawidłowe z powodu błędów pomiarowych lub uszkodzenia urządzenia. Takie błędne dane należy rozpoznać tak blisko ich źródła, jak to możliwe, zanim rozprzestrzenią się na inne węzły i lokalizacje. Po wykryciu tych danych należy odrzucić kolejne informacje z ich źródła, chyba że przyczyna została zidentyfikowana i usunięta. Kwestia ta jest szczególnie istotna w przypadku danych medycznych i osobowych.
- Różne zagrożenia mogą wynikać z niekompatybilnego sprzętu i oprogramowania od różnych producentów, nieprawidłowych zmian w konfiguracji sieci, aktualizacji oprogramowania lub błędów użytkowników.

Możliwe jest uniknięcie tych problemów poprzez wdrażanie bardziej zaawansowanych systemów bezpieczeństwa, mechanizmów ochronnych i mechanizmów wykrywania błędów oraz budowania węzłów przy użyciu bardziej precyzyjnych i niezawodnych czujników i anten. Wymaga to jednak droższego i fizycznie większego sprzętu, co ogranicza wykorzystanie WSN. Innym rozwiązaniem jest zastosowanie metody zarządzania zaufaniem. Umożliwia to zwiększenie poziomu bezpieczeństwa sieci poprzez zminimalizowanie problemów występujących w protokołach WSN, uwzględniając ograniczone zasoby.

A.5. Metody badawcze

Podejście badawcze przyjęte w niniejszej rozprawie jest następujące.

Na początku wykonano szczegółowy przegląd zaproponowanych już sposobów zarządzania zaufaniem w bezprzewodowych sieciach czujników.

W kolejnym etapie została wykonana inwentaryzacja i analiza zagrożeń oraz możliwych ataków w bezprzewodowych sieciach czujników wraz z klasyfikacją tych ataków. Obejmuje ona atak spamu, ataku czarnej dziury i atak modyfikacji wiadomości.

Następnie zaproponowano nową metodę zarządzania zaufaniem w wielowarstwowych bezprzewodowych sieciach czujników złożonych z klastrów, zwaną *WSN Cooperative Trust Management Method* (WCT2M). Metoda ta opiera się na modelu rozproszonego zarządzania zaufaniem w sieciach klastrowych, ponieważ umożliwia ograniczenie problemów wydajnościowych związanych z rozbudową sieci.

Aby ocenić proponowaną metodę WCT2M użyto następującego podejścia:

- W celu wyjaśnienia podstaw WCT2M oraz przedstawienia przykładu jej zastosowania jako odniesienia użyto studium przypadku. Zostało ono wyprowadzone ze studium przedstawionego w projekcie *Advanced Network embedded platform as a Gateway to Enhanced quality of Life* (ANGEL) [13]. ANGEL był projektem *Specific Targeted Research Project* (STReP) przeprowadzonym w latach 2006-2009 w ramach 6. Europejskiego Programu Ramowego, z udziałem Katedry Inżynierii Oprogramowania Politechniki Gdańskiej.
- W celu wykazania stosowalności WCT2M opracowano system laboratoryjny WSN. System jest oparty na węzłach ZigBee. Aby wdrożyć WCT2M na tych węzłach, opracowano dedykowane oprogramowania w języku C z pomocą Eclipse IDE [14] i kompilatora mspgcc [15].

  System laboratoryjny implementuje małą sieć WSN o strukturze pochodzącej ze studium przypadku projektu ANGEL. WCT2M został zainstalowany na każdym z węzłów w systemie. System demonstruje stosowalność WCT2M oraz został wykorzystany do oceny skuteczności metody dla różnych scenariuszy ataków.
- Ze względu na ograniczenie możliwości rozbudowy systemu laboratoryjnego, użyto symulacji w celu oceny skuteczności i wydajności WCT2M w większej skali. Po dokonaniu przeglądu istniejących symulatorów (łącznie z oceną ich dostępności i kosztów), zdecydowano się opracować specjalny symulator do oceny WCT2M. Został on napisany w środowisku programistycznym NetBeans IDE [16], używając Java 1.7 [17] oraz biblioteki JGraph [18].
- Dedykowany symulator WCT2M użyto do analizy skuteczności i wydajności WCT2M dla większych sieci oraz dla różnych zagrożeń.

Wyniki eksperymentów laboratoryjnych i symulacyjnych wykorzystano do uzasadnienia tezy niniejszej rozprawy.

A.6. Metoda WCT2M

Zaproponowana w niniejszej rozprawie metoda WSN Cooperative Trust Management Method (WCT2M) umożliwia zarządzanie zaufaniem w wielowarstwowych bezprzewodowych sieciach czujników złożonych z klastrów.

Zastosowanie WCT2M w sieci wymaga, aby administrator sieci zainstalował ją na każdym węźle, łącznie ze stacją bazową, a następnie ustawił wymagane parametry. Konieczne jest również umożliwienie metodzie na dostęp do informacji z protokołu routingu i protokołu synchronizacji. Na końcu należy skonfigurować wymianę informacji pomiędzy WCT2M a dostępnymi mechanizmami bezpieczeństwa i uruchomić sieć.

Każdy z węzłów sieci uczestniczy w zarządzaniu zaufaniem i utrzymuje dane dotyczące reputacji innych węzłów. Każdemu ze znanych węzłów przypisana jest *wartość zaufania* reprezentowana przez liczbę rzeczywistą z zakresu [0..1]. Dane te utrzymywane są w strukturze nazwanej *tablicą zaufania*.

Przychodzące od innych węzłów wiadomości są oceniane przez mechanizmy bezpieczeństwa, a binarny wynik oceny przekazywany do WCT2M. Na tej podstawie podwyższana lub obniżana jest wartość zaufania do węzła-nadawcy ocenianej wiadomości. Ponadto każdy z węzłów rozsyła swoją tablicę zaufania do sąsiadów, te zaś modyfikują swoje tablice zaufania, biorąc pod uwagę otrzymane rekomendacje.

Szczegółowy opis metody oraz zastosowanych algorytmów znajduje się w Rozdziale 5.2.

A.7. Stan badań w zakresie zarządzania zaufaniem w WSN

Architektura modeli zarządzania zaufaniem

Istnieją dwie podstawowe architektury używane w modelach zarządzania zaufaniem: scentralizowana i rozproszona [10]. Pierwszy typ wyróżnia instytucję zaufania (ang. *Trust Authority*), np. stację bazową, która zarządza relacjami zaufania pomiędzy węzłami [100]. To rozwiązanie jest wydajne i łatwe w zarządzaniu, lecz występują w nim problemy ze skalowalnością i odpornością na ataki.

Model rozproszony jest uważany za odpowiedni dla sieci czujników o dużej skali. Zhiying et al. [10] postrzegają ten model za odpowiedni dla projektowania bezpieczeństwa bezprzewodowych sieci

czujników, ponieważ węzły skupiają się na wiarygodności swoich sąsiadów i mogą ocenić, czy owi sąsiedzi przestrzegają polityki bezpieczeństwa. Autorzy proponują ramy bezpieczeństwa z różnymi dostępnymi schematami bezpieczeństwa. Jednakże ich praca nie uwzględnia limitowanych zasobów dostępnych dla węzłów sieci czujników.

Chen et al. [101] sugerują rozproszony, bazujący na agentach, schemat zarządzania zaufaniem, w którym każdy agent niezależnie monitoruje zachowanie w swoim zasięgu i propaguje swoje rankingi zaufania. Wprowadzają także model zaufania bazujący na reputacji, używając prawdopodobieństwa, statystyki i analizy matematycznej oraz sugerują wprowadzenie systemu zaufania bazującego na przestrzeni reputacji i przestrzeni zaufania w bezprzewodowych sieciach czujników [102].

Możliwe jest również podejście hybrydowe. Łączy ono zalety modelu scentralizowanego i rozproszonego (lecz również wady powiązane z każdą z tych architektur). Sieć taka zawiera dwa poziomy – węzły podzielone są na klastry i każdy węzeł będący głową klastra jest elementem tak zwanej sieci szkieletowej, która umożliwia komunikację głów klastrów ze stacją bazową. Boukerche et al. [103] proponują schemat zarządzania zaufaniem i reputacją, który wykorzystuje mobilnych agentów uruchomionych na każdym z węzłów. W tym modelu istnieje centralny system uruchamiania agentów odpowiedzialny za generowanie i umieszczanie ich w sieci. Jednakże nie istnieje centralne repozytorium zaufania, co czyni wymianę danymi o zaufaniu (o ile istnieją mobilne węzły), znacznie trudniejszą.

Wybrane metody zarządzania zaufaniem

Song et al. [111] przedstawiają protokół *Trust-based LEACH* (TLEACH) w celu rozszerzenia bezpieczeństwa oferowanego przez protokół LEACH [30], przy zachowaniu najważniejszych funkcji oryginału, takich jak algorytm wyboru głowy klastra i fazy pracy. Dodają sloty zaufania (ang. *trust slots*) do zaimplementowanego modułu routingu bazującego na zaufaniu, aby zaoferować lepsze wsparcie dla oceny zaufania. Decyzje są podejmowane na podstawie tzw. zaufania decyzji (ang. *decision trust*), które jest obliczane niezależnie i dynamicznie dla każdej z podejmowanych decyzji. Autorzy sugerują również schemat kontrolowania monitoringu wspierany przez głowy klastrów w celu obniżenia zużycia energii. TLEACH jest modelem hybrydowym, gdyż węzły potrafią wykorzystać informacje o zaufaniu przekazane przez głowy klastrów zamiast uzyskane samodzielnie. Sytuacja taka ma miejsce, gdy węzeł niebędący głową klastra ufa głowie swojego klastra.

Bazowy algorytm protokołu TLEACH jest podobny do tego zastosowanego w WCT2M. Zaletą tego rozwiązania jest mniejsze zużycie energii przez węzły niebędące głową klastra, jednakże powoduje, że znacznie szybciej odzyskują one zaufanie.

Chen et al. [101] proponują model zaufania *Agent-based Trust Model* (ATSN) bazujący na rozproszonych agentach. Model ten używa schematu *watchdog* do obserwacji zachowania węzłów i rozsyłania swoich rankingów zaufania. Węzły z zainstalowanym agentem wykorzystują tryb nasłuchu (ang. *promiscuous*) do monitorowania zachowania węzłów w swoim zasięgu i klasyfikują akcje. Każdy z agentów dysponuje kilkoma modułami. Każdy z modułów ma zaimplementowaną funkcję umożliwiającą klasyfikowanie zebranych danych i oznaczanie akcji jako współpracujących bądź niewspółpracujących. Ponadto model nadaje nowszym danym większy priorytet w porównaniu do starszych klasyfikacji. Umożliwia to udane wykrywanie węzłów o niższej aktywności.

W odróżnieniu od WCT2M model ten nie wykorzystuje rekomendacji przesyłanych przez inne węzły, a brane pod uwagę są wyłącznie własne oceny węzła.

Zia [110] przedstawia schemat *Reputation-based trust management scheme* który wykorzystuje głosowanie w celu ustalenia poziomu zaufania pomiędzy węzłami. Wartość głosu jest zwiększana wraz z każdą udaną transmisją wiadomości od jednego węzła do drugiego, natomiast wartość zaufania jest obniżana, gdy węzeł oddaje negatywny głos na któregoś ze swoich sąsiadów. Każdy z węzłów po wysłaniu wiadomości, która powinna zostać następnie przesłana dalej, nasłuchuje komunikacji i zwiększa poziom zaufania, gdy wiadomość pozostała niezmieniona, obniża zaś poziom zaufania, gdy nastąpiły zmiany wiadomości. Jeżeli liczba negatywnych głosów osiągnie predefiniowany próg, węzeł z takimi głosami jest oznaczany jako niezaufany. Węzeł, który podejmuje taką deklarację, propaguje ją dalej. Gdy notyfikacja dotrze do głowy klastra, izoluje on niezaufany węzeł i zaczyna ignorować każdą wiadomość pochodzącą od niego. Głowa klastra jednocześnie rozsyła informację o niezaufanym węźle, co umożliwia jej dotarcie do każdego sąsiada niezaufanego węzła i izolowanie każdej pochodzącej od niego wiadomości.

Zaproponowany schemat, w odróżnieniu od WCT2M, umożliwia głowom klastrów podejmowanie scentralizowanych decyzji o izolacji węzła.

Badania Momani et al. [112] skupiają się na modelowaniu i obliczaniu zaufania w bezprzewodowych sieciach czujników bazując na wykrytych zdarzeniach. Wprowadzają nowy model zaufania i system zarządzania reputacją w bezprzewodowych sieciach czujników. Bazuje on na wykrywaniu ciągłych danych (np. temperatury), w odróżnieniu do podejścia zastosowanego m.in. w WCT2M, w którym zaufanie wyliczane jest na podstawie zdarzeń binarnych. Powstał w ten sposób *Gaussian Reputation System for Sensor Networks* (GRSSN), który jest nową wersją *Systemu Reputacji Beta,* zaprezentowanego w [113], stosowanego do zdarzeń binarnych. Model zaufania reprezentuje wiarygodność węzła w opinii pozostałych węzłów, poprzez przydzielenie przez każdy węzeł wartości

zaufania każdemu innemu węzłowi. Bazując na tej wartości zaufania, można wyznaczyć wartość ryzyka powiązaną z wykonaniem zadania przez dany węzeł.

Feng et al. [114] proponują algorytm *Node Behavioural Strategies Banding Belief Theory of the Trust Evaluation Algorithm* (NBBTE), który łączy strategie behawioralne i zmodyfikowaną teorię dowodów. Tworzone są rozmaite czynniki zaufania i współczynniki związane z zastosowaniami sieciowymi zgodne z zachowaniem czujników, w celu uzyskania bezpośrednich i pośrednich wartości zaufania poprzez obliczenie średniej ważonej czynników zaufania. Jednocześnie metoda rozmytego zestawu (ang. *fuzzy set method*) jest stosowana do stworzenia podstawowego wejściowego wektora dowodów. Na tej podstawie jest obliczana różnica dowodów między pośrednimi i bezpośrednimi wartościami zaufania, co w rezultacie umożliwia syntezę zintegrowanej wartości zaufania węzłów.

Zaproponowany algorytm umożliwia skuteczne wykrywanie złośliwych węzłów, wymaga to jednak wykonania bardziej zaawansowanych obliczeń w porównaniu do WCT2M, co wpływa na zużycie energii przez węzeł.

Bao et al. [115] proponują skalujący się, hierarchiczny protokół zarządzania zaufaniem bazujący na klastrach dla bezprzewodowych sieci czujników w celu skutecznego radzenia sobie z samolubnymi i złośliwymi węzłami. Opisali niejednorodną bezprzewodową sieć czujników zawierającą dużą liczbę węzłów o różnych społecznych zachowaniach oraz o różnej jakości usług. Rozważają wielowymiarowe atrybuty zaufania wyprowadzone z komunikacji i sieci społecznościowych w celu wyznaczenia ogólnego zaufania dla każdego z węzłów.

Zastosowanie tychże atrybutów odróżnia ten protokół od WCT2M.

A.8. Wkład rozprawy w rozwój dziedziny

Poniżej zaprezentowano wkład rozprawy w rozwój dziedziny.

WCT2M – nowa metoda zarządzania zaufaniem

Została zaprezentowana nowa metoda umożliwiająca zarządzanie zaufaniem w wielowarstwowych bezprzewodowych sieciach czujników, nazwana *WSN Cooperative Trust Management Method* (WCT2M). Metoda bazuje na rozproszonym modelu zarządzania zaufaniem w sieciach wielowarstwowych (klastrowych), co umożliwia ograniczenie problemu wydajności związany z rozbudową sieci.

Zbiór metryk umożlwiających ocenę WCT2M

Przy użyciu metodologii Goal-Question-Metrics (GQM) [94] wybrany został zbiór metryk. GQM oferuje systematyczne podejście umożliwiające uzyskanie zbioru metryk wspierających zdefiniowany cel badań. Rezultatem jest zdefiniowanie 3 pytań oraz 11 metryk zaprezentowanych w Rozdziale 7.1.1. Następnie uzyskane metryki zostały użyte do oceny zarówno wyników eksperymentów laboratoryjnych, jak również symulacyjnych.

Sieć laboratoryjna i symulator WCTMS

Zostało stworzone dedykowane laboratorium przy użyciu elementów zestawów CC2520 Development Kit [69]. Każdy węzeł został zaprogramowany dedykowanym oprogramowaniem napisanym w języku C, przy wykorzystaniu bibliotek Vlo_rand [74] oraz HAL [75]. Oprogramowanie umożliwia wybór jednego z kilku predefiniowanych trybów działania wykorzystując joystick i guziki urządzenia oraz wykonanie różnych eksperymentów w sieci złożonej z 11 węzłów (sieć została zaprezentowana na Rysunku Figure 10). Eksperymenty i ich wyniki zostały przestawione w Rozdziale 7.2.

Walidacja większych sieci przy użyciu prawdziwych urządzeń byłaby zbyt kosztowna i skomplikowana, jako że setki urządzeń rozproszone po dużej powierzchni byłyby trudne do programowania. Stworzono zatem dedykowany symulator w języku Java: WCTMS. Symulator, oprócz prowadzenia doświadczeń, umożliwia wyświetlenie graficznej prezentacji rozmieszczenia węzłów na symulowanej przestrzeni za pomocą biblioteki JGraph [18].

Aby ocenić poprawność wyników uzyskanych przy pomocy symulatora WCTMS, zostały one porównane z wynikami uzyskanymi w sieci laboratoryjnej. Rezultat porównania przestawiono w Rozdziale 7.2.1.2.

Eksperymenty i ich rezultaty

W trakcie pracy przeprowadzono eksperymenty podzielone na następujące grupy.

EXP1: Wykonalność implementacji WCT2M i implementacja modeli ataku

Celem tego eksperymentu było wykazanie, że WCT2M możne zostać zastosowana w typowym środowisku WSN i pokazanie zachowania uszkodzonych węzłów, zgodnie z opisem zamieszczonym Rozdziale 4.2.

Wyniki tego eksperymentu wykazały, że WCT2M może zostać zastosowana w typowym środowisku bezprzewodowych sieci czujników, które może zawierać wadliwe węzły.

Eksperyment pokazał również, że wyniki osiągnięte przy użyciu sieci laboratoryjnej są zbieżne z wynikami osiągniętymi przy użyciu symulatora WCTMS.

Szczegółowy opis eksperymentu i jego wyników znajduje się w Rozdziale 7.2.1.

EXP2: Opóźnienie w wykrywaniu uszkodzonych węzłów

Celem tego eksperymentu było zmierzenie opóźnienia potrzebnego do wykrycia uszkodzonych węzłów w sieci. Zostały rozważone dwie sytuacje:

1) wadliwe węzły były obecne od początku eksperymentu;

2) wadliwe węzły dodano w trakcie działania sieci.

Doświadczenie wykazało, że nagromadzenie uszkodzonych węzłów w jednym klastrze wpływa w niewielki sposób na wydajność WCT2M w porównaniu do sytuacji, w której uszkodzone węzły znajdują się w różnych klastrach sieci. Doświadczenie pokazało również, że uszkodzona głowa klastra może być wydajnie wykryta, ponieważ przesyła wiele wiadomości w trakcie każdego cyklu WCT2M.

Szczegółowy opis eksperymentu i jego wyników znajduje się w Rozdziale 7.2.2.

EXP3: Zależność między aktywnością węzłów i opóźnieniem w wykrywaniu uszkodzonych węzłów

Celem tego eksperymentu było zbadanie, czy i jak aktywność węzłów i stacji bazowej wpływa na czas wykrycia pierwszego i wszystkich uszkodzonych węzłów w sieci.

Eksperyment wykazał, że WCT2M może skutecznie i wydajnie wykrywać i odcinać od sieci uszkodzone węzły. Czas detekcji wzrasta odwrotnie proporcjonalnie do aktywności węzłów. Doświadczenie wykazało również, że czas potrzebny do wykrycia i wyizolowania pojedynczego uszkodzonego węzła zależy głównie od aktywności węzłów, zaś w mniejszym stopniu zależy od aktywności stacji bazowej.

Szczegółowy opis eksperymentu i jego wyników znajduje się w Rozdziale 7.2.3.

EXP4: Zależność między liczbą uszkodzonych węzłów w sieci i opóźnieniem w ich wykrywaniu

Celem tego eksperymentu było zbadanie, w jaki sposób liczba uszkodzonych węzłów w sieci wpływa na opóźnienie w ich wykrywaniu.

Eksperyment wykazał, że WCT2M umożliwia skuteczną izolację uszkodzonych węzłów w jednopoziomowych oraz dwupoziomowych sieciach o różnych rozmiarach (badaniu zostały poddane sieci do 1000 węzłów). Doświadczenie wykazało, że mediana opóźnienia potrzebnego do wykrycia pierwszego uszkodzonego węzła (MFND) jest większa w przypadku sieci jednopoziomowej w porównaniu do sieci dwupoziomowej, ale mediana opóźnienia potrzebnego na wykrycie wszystkich uszkodzonych węzłów (MAND) jest mniejsza. Okazało się również, że jakość odcięcia (CQ) jest lepsza w sieciach jednopoziomowych niż w sieciach dwupoziomowych i staje się coraz gorsza wraz ze wzrostem liczby uszkodzonych węzłów w sieci.

Szczegółowy opis eksperymentu i jego wyników znajduje się w Rozdziale 7.2.4.

EXP5: Odporność na zmniejszoną częstotliwość ataku

Celem tego eksperymentu było zbadanie odporności WCT2M na ataki wykonywane ze zmniejszoną częstotliwością. W Rozdziale 5.2.3 zostało opisane narzędzie historii akcji, które ma na celu przeciwdziałanie atakowi tego rodzaju.

Wyniki eksperymentu pokazały, że historia akcji jest skutecznym narzędziem, umożliwiającym zmniejszenie czasu wykrywania złośliwych węzłów w sieci. Jest to szczególnie przydatne, gdy złośliwe węzły wykonują atak ze zmniejszoną częstotliwością – im mniejsza jest częstotliwość szkodliwych działań, tym skuteczniejsze jest proponowane rozwiązanie.

Szczegółowy opis eksperymentu i jego wyników znajduje się w Rozdziale 7.2.5.

EXP6: Odporność na atak zmowy

Celem tego eksperymentu było zbadanie odporności WCT2M na atak zmowy. W Rozdziale 5.2.3 zostało opisane narzędzie historii zaufania, które ma na celu przeciwdziałanie atakowi tego rodzaju.

Eksperyment wykazał, że WCT2M jest w stanie skutecznie wykryć węzły wykonujące atak zmowy, chyba że liczba takich węzłów zbliża się do połowy ogólnej liczby węzłów w sieci. Szczegółowy opis eksperymentu i jego wyników znajduje się w Rozdziale 7.2.6.

EXP7: Wpływ skuteczności mechanizmów bezpieczeństwa

Celem tego eksperymentu było zbadanie, w jaki sposób skuteczność mechanizmów bezpieczeństwa zaimplementowanych w węźle wpływa na opóźnienie wykrywania uszkodzonych węzłów.

Eksperyment pokazał, że skuteczność mechanizmów bezpieczeństwa oceniających odebrane wiadomości silnie oddziałuje na wydajność WCT2M. Zademonstrowano również, że skuteczność ta ma niewielki wpływ na jakość odcięcia (CQ). Zaobserwowano, że jeśli skuteczność mechanizmów bezpieczeństwa jest większa niż 70%, WCT2M jest w stanie wykrywać i izolować złośliwe węzły bez znacznego spadku wydajności. Jednakże, jeśli skuteczność ta spadnie poniżej 70%, wydajność wykrywania złośliwych węzłów gwałtownie maleje.

Szczegółowy opis eksperymentu i jego wyników znajduje się w Rozdziale 7.2.7.

A.9. Upubliczniony dorobek badań

Praca zaprezentowana w niniejszej rozprawie została opublikowana w sprawozdaniach z sześciu konferencji. Publikacje te zostały podsumowane poniżej:

- [60] wprowadza do problemu rozproszonego zarządzania zaufaniem w bezprzewodowych sieciach czujników. Wyjaśnione są podstawowe koncepty i definicje oraz zaprezentowany jest model zarządzania zaufaniem. Zaprezentowany jest również symulator WCTMS i jego założenia oraz zaprezentowane są uzyskane wyniki symulacji.
- [117] jest rozwinięciem [60]. Prezentuje szczegóły poprzednio zaproponowanego modelu oraz kolejne wyniki symulacji.
- [118] prezentuje dwuwarstwowy model zarządzania zaufaniem dla bezprzewodowych sieci czujników. Model zakłada, że węzły oceniają wiarygodność innych węzłów bazując na wzajemnych obserwacjach i rekomendacjach. Podział struktury sieci na dwie warstwy umożliwia dłuższe i bardziej efektywne działanie węzłów.
- [119] prezentuje przypadek użycia powiązany z zastosowaniem bezprzewodowej sieci czujników w domenie e-zdrowia. Założono, że sieć implementuje WCT2M, która umożliwia wykrycie i izolację węzłów nieprzestrzegających polityki sieci.
- [120] jest kontynuacją [119]. W celu zmierzenia efektywności wykrywania węzłów opisanej w [119] został wyprowadzony zestaw metryk używając metodologii Goal-Question-Metrics. Sieć została zasymulowana przy użyciu symulatora WCTMS i uzyskane dane zostały

wykorzystane do wyznaczenia wartości metryk demonstrujących skuteczność metody w zakresie wykrywania i eliminacji z sieci uszkodzonych węzłów.

- [46] prezentuje rezultaty oceny efektywności WCT2M w pełni synchronizowanej bezprzewodowej sieci czujników. Wprowadza kilka podstawowych typów wzorców synchronizacji w WSN, bazując na idei harmonogramu snu, następnie wyjaśnia jak WCT2M działa w sieci wykorzystującej wzorzec pełnej synchronizacji harmonogramu snu. Sieci te zostały poddane analizie przy wykorzystaniu symulatora WCTMS w celu zbadania opóźnień potrzebnych do wykrycia i odizolowania węzłów sieci, których zachowanie odbiega od charakterystyk przyjętych dla danej sieci. Rezultaty tych symulacji zostały zaprezentowane w celu demonstracji efektywności WCT2M.

Tabela 1 prezentuje, jak powyższe publikacje są powiązane z rozdziałami niniejszej rozprawy.

Tabela 1 Powiązanie publikacji z rozdziałami

Publikacja	Rozdziały
[60]	5, 5.2.4, 6.2.3
[117]	5.1
[118]	2.3, 7.2.4
[119]	3
[120]	3, 7.1.1, 7.2.2
[46]	2.4, 7.2.3

Publikacja podsumowująca wszystkie rezultaty uzyskane w trakcie badań jest w trakcie przygotowywania do opublikowania w międzynarodowym czasopiśmie naukowym.

A.10. Kierunki przyszłych badań

W ramach kolejnych badań planuje się analizę sieci składających się z mobilnych węzłów. Takie zachowanie wymusza częste przeformowania się klastrów. Umożliwia to złośliwym węzłom zmianę klastra na taki, w którym wartości zaufania do niego są wysokie.

Planuje się także zbadanie problemu plotki. Często węzeł ma co najmniej kilku sąsiadów. Po kilku cyklach WCT2M każdy z węzłów w sieci ma informacje o pozostałych węzłach ze względu na wymienianie się tablicami zaufania. Oznacza to, że węzły znają większość pozostałych węzłów sieci ze względu na „plotki". Ta niebezpośrednia wiedza jest wymieniana także w przeciwnym kierunku – tablice zaufania wysyłane są do bezpośrednich sąsiadów węzła. Ciekawym problemem jest kwestia

w jaki sposób tak uzyskane „informacje zwrotne” z plotek wpływają na efektywność metody i czy możliwe jest wykorzystanie tej własności przez złośliwe węzły, aby zaatakować WCT2M.

A.11. Podsumowanie

Bezpieczeństwo jest w ostatnim czasie przedmiotem zainteresowania nie tylko naukowców i inżynierów, ale także zwykłych ludzi. Używamy coraz więcej połączonych urządzeń (idea internetu rzeczy, ang. *internet of things*) i bezprzewodowe sieci czujników (WSN) stają się ważną częścią globalnej sieci [116]. Zarządzanie zaufaniem jest jednym z możliwych rozwiązań dla skutecznego zapewnienia bezpieczeństwa w bezprzewodowych sieciach czujników.

W Rozdziale 2 zostało wyjaśnione, czym są bezprzewodowe sieci czujników i podstawowe koncepcje wykorzystane w tej rozprawie, między innymi harmonogram snu. Następnie, w Rozdziale 3, przypadek użycia wykorzystywany w niniejszej rozprawie został opisany w celu zaprezentowania reguł zarządzania zaufaniem. Zaprezentowano przykładową sieć i opisano przykładowe zagrożenia mogące wystąpić w tejże sieci. W Rozdziale 4 została przestawiona lista znanych ataków w bezprzewodowych sieciach czujników wraz ze scenariuszami tych ataków oraz opisano znane sposoby ochrony przed tymi atakami. Następnie, w Rozdziale 5, wprowadzono pojęcia zaufania i zarządzania zaufaniem oraz zaproponowano nową metodę zarządzania zaufaniem: WCT2M. W celu oceny WCT2M opracowano narzędzia analityczne: specjalnie stworzone laboratorium oraz symulator WCTMS, które zostały opisane w Rozdziale 6. W Rozdziale 7 zaprezentowano wyniki eksperymentalnej oceny WCT2M, uzyskane przy pomocy wcześniej opisanych narzędzi. Na końcu, w Rozdziale 8, zawarto porównanie WCT2M z innymi metodami zarządzania zaufaniem w bezprzewodowych sieciach czujników opisanymi w literaturze.

Zademonstrowano, że WCT2M skutecznie i wydajnie zarządza zaufaniem w bezprzewodowych sieciach czujników. Umożliwia rozpoznanie niezaufanych węzłów i zapobiega rozprzestrzenianiu się w sieci informacji pochodzących z tych węzłów, zużywając rozsądną ilość zasobów.

W konsekwencji, teza rozprawy sformułowana następująco:

> ***Proponowana metoda umożliwia skuteczne i wydajne zarządzanie zaufaniem w bezprzewodowych sieciach czujników.***

została udowodniona.