A METHOD OF TRUST MANAGEMENT IN WIRELESS SENSOR NETWORKS

Janusz Górski and Alan Turower

Department of Software Engineering, Gdańsk University of Technology, Gdańsk, Poland

ABSTRACT

The research problem considered in this paper is how to protect wireless sensor networks (WSN) against cyber-threats by applying trust management and how to strengthen network resilience to attacks targeting the trust management mechanism itself. A new method, called WSN Cooperative Trust Management Method (WCT2M), of distributed trust management in multi-layer wireless sensor networks is proposed and its performance is evaluated. The method is specified by giving its class model in UML and by explaining the related attributes and methods. Different attacks against the network and against WCT2M deployed in the network are considered. The experimental evaluation of WCT2M involves laboratory experiments and simulations using a dedicated simulator. The evaluation focuses on efficiency of detecting and isolating the malicious nodes that implement different attack scenarios in the network and on the method's sensitivity to the changes in effectiveness of the security mechanisms deployed in the network nodes.

KEYWORDS

Wireless Sensor Network; trust management; security threat; attack scenario; simulation

1. INTRODUCTION

Wireless sensor networks (WSN) are important in different domains, including healthcare, defense, security, environment monitoring and others. Although WSNs share a number of security requirements with traditional networks, due to their specific limitations they may need different solutions. WSNs have limited resources such as memory, power supply and computational ability. Resource limitations stem from the need of tiny and low cost devices. This restricts applicability of conventional protection mechanisms (like advanced crypto schemes) and excludes complex and resource-consuming security solutions [1]. Moreover, the protocols used in WSNs are often designed without devoting much attention to security requirements. ZigBee [2], one of the most popular specifications for WSN communication protocols, may serve as an example. ZigBee is built upon the IEEE 802.15.4 standard [3] which defines the Physical and Media Access Control (MAC) layers for low cost, low rate personal area networks.

Figure 1 presents an example of a multitier WSN - this case study was borrowed from the ANGEL project [4]. The base station is labelled BS and the nodes are labelled by natural numbers. Solid lines represent the routes leading from nodes to the base station. In addition, the dotted lines connect 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 with BS. The nodes of this network are divided into the following *clusters*: cluster_1 = {1, 2, 3, 4}, cluster_2 = {5, 6, 7}, cluster_3 = {8, 9, 10} and cluster_4 = {1, 8, 5}. Cluster_4 groups the heads of clusters _1, _2 and _3 and together with the base station forms tier_1 = {BS, 1, 5, 8}. The remaining nodes form tier_2 = {2, 3, 4, 6, 7, 9, 10}. The nodes of tier_1 (except the base station) are routers – they forward messages from BS to their child nodes and from their child nodes to BS. The level of a node is the shortest distance (in hops) from this node to the base station. The levels of the nodes are as follows: for nodes 1, 5 and 8, level = 1 and for nodes 2, 3, 4, 6, 7, 9 and 10, level = 2.

The height h of a network is the maximal level of nodes of the network. The height of the network of Figure 1 is h = 2.

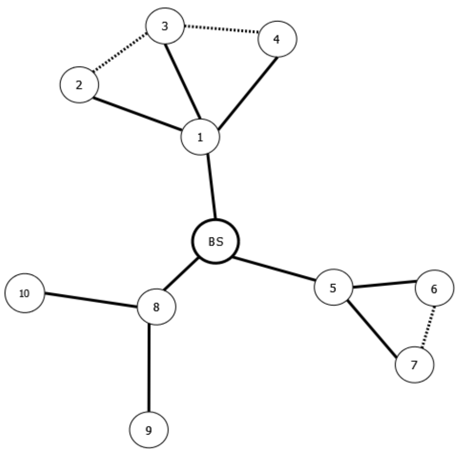
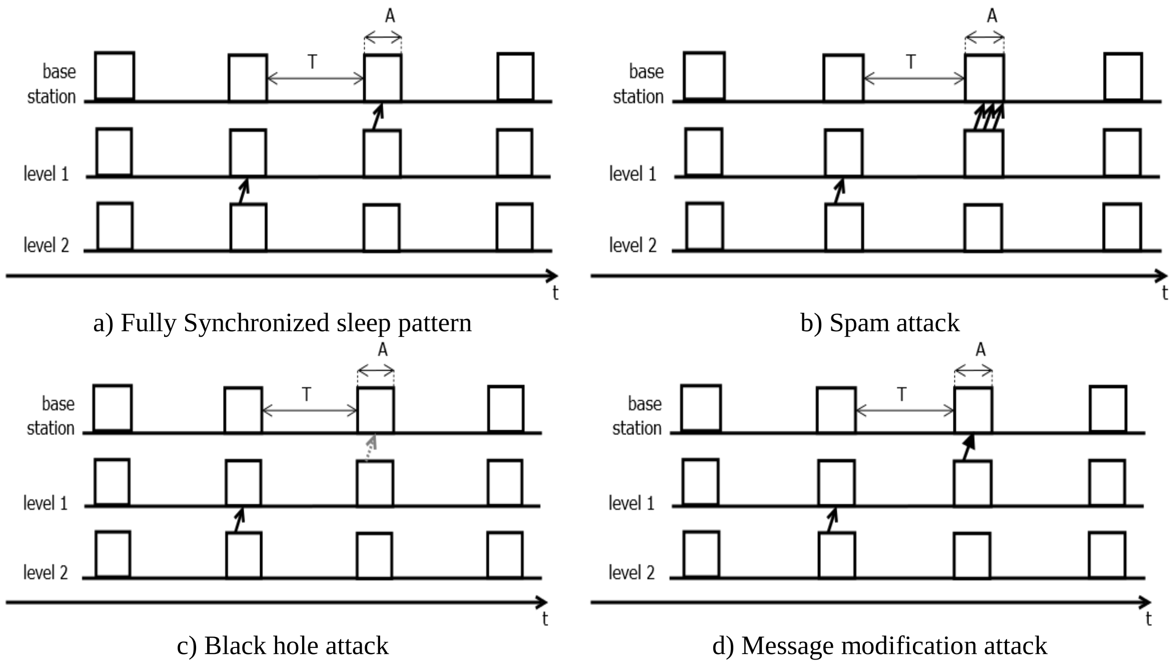


Figure 1. An example two-tier network

Let us assume that the network applies *fully synchronized sleep pattern* [5] as presented in Figure 2a [6]. Node operation is split into *sleep* (labelled: T) and *wakeup* (labelled: A) periods. The arrows represent the fastest scenario of sending a message from a node of level 2 to the base station.



a) Fully Synchronized sleep pattern

b) Spam attack

c) Black hole attack

d) Message modification attack

Figure 2. Fully Synchronized sleep pattern and related attacks

The network as presented in Figure 1 can be subjected to various attacks. For instance, a malicious router node, say node 8, may modify the forwarded messages or even drop them selectively. In more complex situations the malicious nodes can involve themselves in even more advanced attack schemes.

An alternative that can help to cope with the above problems is *trust management*. *Trust* can be defined as a bet that those entities, which you cannot control, will act in a predictable manner that is favourable to your cause [7]. Generally, trust is a relation between a *trustor* (the trusting subject) and a *trustee* (the subject being trusted). The objective of trust management in WSN 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 [8].

Trust management schemes are proposed as a solution to many security issues in WSN. For instance, [9] asserts 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.[10] emphasizes the importance of trust as an important issue of security schemes in wireless sensor networks. They categorize various types of attacks and countermeasures related to trust schemes in WSNs. They also present an extensive literature survey by summarizing state-of-the-art trust mechanisms in two categories: secure routing and secure data. [11] lists the best practices being 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. [12] proposes a 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 providing 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 collecting relevant information by the trust-based architecture independently of the specific features of the considered routing protocol. [13] pays attention to the problem of collusion attacks in trust-based MANET networks (mobile ad hoc 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' reputation by giving negative recommendations about the latter. The authors propose detecting the colluding nodes by calculating recommendation deviations and punishing these nodes by discarding them from further communication.

To strengthen protection against various attack schemes we propose a new trust management method called WCT2M which is implemented by a software package to be installed on each node of the network. WCT2M assumes that each node N has a potential to detect that an incoming message sent by another node, say M, violates the security policies implemented by N and this information becomes available to WCT2M together with the M's identity (no assumptions about the 'strength' of such security policies are being made). The information that a malicious message has been received from M is used by N to compute the trust level assigned to M. In addition, the trust level assigned to M by node N takes into account the trust levels assigned to M by other nodes and communicated to N as recommendations. Then, node N communicates to other nodes (in a form of recommendation) the computed trust level assigned to M and depending on this trust level N decides whether it continues its cooperation with M or not. Together with the trust levels assigned to its neighbouring nodes, N maintains historic data related to the previous assessments of the messages received from M (called validity history) and the deviations between the own assessment of trustworthiness of M and the recommendations related to the trustworthiness of M received from other nodes (called deviation history). Both, the validity history and the deviation history are taken into account by N while computing a new trust level of M, before the recommendation related to M is being sent to other nodes. The functionality of WTC2M software package was specified using UML and then this specification was used to implement the package. This implementation was used to demonstrate WTC2M in a laboratory environment and to run experiments of a limited scale. To provide for analysing performance of the method in larger networks that are subjected to various attacks (against the network and against the trust management mechanism itself) we developed a dedicated simulator. During simulations we assumed a probabilistic model for both, the effectiveness of mechanisms enforcing the security policies implemented by the network nodes and the attacks against the analysed network and its trust management mechanisms.

The remainder of this paper is organized as follows. In section 2 we review the related works and in section 3 we introduce a range of different attacks against a network and the attacks against a trust management mechanism deployed in the network, and state the problem we are dealing with in this paper. In section 4 we introduce WCT2M, a new trust management method for WSN and describe the way of method instantiation and the parameters and data types used by the method. Section 5 gives more details of WCT2M and in particular presents the class model and the execution model of the method. In section 6 we present the assumed evaluation strategy, the tools used during evaluation and the scope of evaluation experiments. In section 7 we present the results of the evaluation experiments and section 8 present the discussion of the results achieved and the directions of further research.

2. 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 and efficiently identify and isolate malicious nodes in the network. There are two main models considered for trust management: centralized and distributed [1]. The centralized model distinguishes the Trust Authority (this could be a role of the base station) which manages trust relationships between nodes [14]. This solution is efficient and manageable, but it has problems with scalability and robustness. The distributed trust model is considered to be suitable for large-scale sensor networks. [1] finds this model appropriate for sensor network security design because a node focuses on the trustworthiness of its neighbors and assesses if these nodes obey the 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. [15] proposes 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 [16].

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 where 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. [17] proposes a 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 significantly more difficult. [18] explore the Dempster-Shafer Theory of Evidence (DST) as part of a framework called DST-SDF and discuss some of its advantages and disadvantages. It differs from existing reputation schemes in that the well-known but in their opinion faulty watchdog mechanism is dispensed with, and end-to-end acknowledgments are used instead. [19] states that trust based only on reputation is not sensitive enough to perceive suddenly spoiled nodes that may launch intelligent attacks against a trust-establishment mechanism. It presents a Trust Model based on Risk evaluation (TMR) to effectively deal with conflicting behaviours of malicious nodes. However, risk factor evaluation causes the trust model less reliable because it becomes more sensitive to on-off attack. [20] states that encryption and authentication can prevent the external attacks on Wireless Sensor Networks, and trust management schemes are better approaches for defending against the internal attacks in WSN. They describe the internal reputation time-varying attack, in which the reputation value of nodes is manipulated to adjust dynamically by the compromised nodes or a malicious attacker. The authors propose a Time-window-based Resilient Trust Management Scheme (TRTMS), based on BETA distribution. In the scheme, the behaviours of compromised nodes are analyzed for a period, and then the difference judgment and the trend analysis are utilized to identify the abnormality of nodes' reputation.

A distinguishing feature of WCT2M is that it is implemented on top of the stack of protocols and remains neutral with respect to the protocols used in the network. The method can coexist with any security mechanism that implements specific security policies at network nodes to assess validity of the incoming messages. WCT2M makes no initial assumptions concerning the effectiveness of such assessment. It simply uses the result of validity assessment as a factor influencing the trust value (also called reputation) assigned to the sender of the incoming message. The other factors influencing this trust value are the recommendations received from the neighbouring nodes and the validity and deviation histories related to the sender of the message.

3. PROBLEM STATEMENT

WSN networks are subjected to various attacks and, if protected by a trust management mechanism, they can also be exposed to the attacks targeting the trust management mechanism itself.

3.1. NETWORK ATTACKS

Three different attack models were chosen to demonstrate possible threats to WSN. They are representative for a number of more concrete network attack scenarios. We describe them below referring to the example network of Figure 1.

3.1.1 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 the 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 remain operational. Attack scenario: The attacker node, in its wakeup period, sends multiple messages to the base station. Figure 2b illustrates the spam attack – in its wakeup period, a node on level 1 of the network sends more messages than it was supposed to do. Attack success criteria: The attack is considered successful if the attacker continues to deliver spam messages and remains undetected.

3.1.2 Black hole attack

In its simplest form the black hole attack involves losing the received messages to prevent their further propagation. Such situation is not unusual even during a normal operation of sensor networks – a network must be ready for sudden disappearance of a node, for instance due to the exhaustion of its energy source. Therefore, most protocols have a mechanism that periodically checks all paths, and if a path stops working, it becomes re-created. This action eliminates malicious nodes from the path (if any) and prevents them from sabotaging the network operation. For this reason, to avoid detection, black hole attacks use a mechanism of random transmission of packets to confuse the maintenance mechanism. Another variant can be the selective forwarding attack where a malicious node selectively drops sensitive messages [21]. These attacks target routers because only router nodes forward messages and can drop them becoming a black hole. Attack scenario: The attacker node does not forward any messages or forwards only some of them (to the base station and to other nodes). All not-forwarded messages are dropped. Figure 2c shows the black hole attack led by a node on level 1. A message which has been dropped in the black hole is represented by a dotted arrow. Attack success criteria: The attack is considered successful, if a malicious node located on at least one routing path blocks forwarding of messages and remains undetected.

3.1.3 Message modification attack

Message modification attack targets router nodes and involves unauthorized modification of a message data and then resending the message. The attacker can modify the content, the information about the receiver node and/or information about the sender node. Attack scenario: The attacker node modifies all or some messages being forwarded to the base station or to other nodes. All other messages are forwarded without any modification that contradicts the network policy (some message attributes can change during 'normal' forwarding). Figure 2d illustrates a message modification attack – the message that was maliciously modified by a node on level 1 is represented by a solid arrow. Attack success criteria: The attack is considered successful if a malicious node located on at least one routing path modifies messages and remains undetected.

3.2. Trust Management Mechanism Attacks

Trust management helps to make the whole network more resistant to attacks, but it can be the target of attacks itself. Two popular attacks on trust management mechanism are described below [22].

3.2.1 Decreased frequency attack

To confuse the trust management mechanism an attacker can decrease frequency of its attack against the network. If malicious actions are performed rarely enough, the malicious node can regain its trustworthiness before being detected and still remains in the network.

3.2.2 Collusion attack

Trust management mechanism can get confused by collusion of malicious nodes. Some nodes can collude in order to mutually increase their trustworthiness and at the same time decrease the trustworthiness of other (innocent) nodes [13].

3.3. Dependence On Effectiveness Of Security Mechanisms

Security depends on trust (for instance in public key infrastructures) however, trust can also be built on top of security. Consider a WSN node with some security mechanisms on board which implement a chosen security policy. If an incoming message violates this policy, it consequently decreases the trust in its source (the message originator) and this information is then spread to other network nodes in a form of recommendation. Such recommendation can affect trust of these other nodes in the message originator even if they were not the direct recipients of the spoiled message. The resulting changes of trust eventually lead to the exclusion of the sender of the spoiled messages from the network. A question arises to what extent the efficiency of trust management (understood as a time delay needed to exclude the malicious nodes from the network) depends on the effectiveness of the security mechanisms installed on network nodes. Of particular interest is the situation where we still can have efficient trust management even if the effectiveness of the security mechanisms is relatively low (note that less effective security is usually cheaper in terms of resource consumption, which could be an attractive alternative for WSNs).

3.4. The Problem

The research problem considered in this paper is how to protect a WSN equipped with incoming message validity checking mechanisms of limited effectiveness deployed on the nodes, against various attack scenarios, by applying a trust management mechanism that is resilient to typical attacks targeting this mechanism.

To solve the above problem we proposed a new method of trust management in multi-layer wireless sensor networks called WCT2M (WSN Cooperative Trust Management Method). We focused on clustered networks to limit performance problems related to network expansion, as clusters allow achieving high scalability and high energy efficiency of large wireless sensor networks [23].

The main contribution of this paper is WCT2M, a new method of trust management in multi-layer clustered wireless sensor networks. The method implements a distributed trust management model. A laboratory implementation and a dedicated simulator have been developed and used to evaluate WCT2M efficiency in protecting against selected attacks in networks of realistic size.

4. WTC2M – A METHOD OF TRUST MANAGEMENT IN WSN

WCT2M implements a distributed trust management model. No central entity is being assumed after a WTC2M enabled network starts its operation. The trust related decisions made at a node are solely based on local assessment of the received messages and on the recommendations received from other nodes.

4.1. INSTANTIATION OF WCT2M

WCT2M is implemented by a software package called WCT2M software. To run WCT2M software, the network administrator needs to perform the following steps:

Step 1: Install WCT2M software on each network node, including the base station, and assign values to the parameters of the method;

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 communication software installed on the node to provide access to the incoming messages and to the results of their assessment by security mechanisms enabled on the node;

Step 4: Start the network and allow propagating trust information between its nodes;
To join a new node to the existing network, the network administrator needs to follow steps 1-3 with respect to the node and then the new node can join the running network.

4.2. PARAMETERS OF WCT2M

In a WCT2M enabled network each node acts in a dual role: as a *trustor* (for incoming messages it makes real-time decisions if a sender can be trusted) and a *trustee* (for outgoing messages other nodes judge if the sender can be trusted).

To represent trust, all nodes running WCT2M software use the same scale called *trust scale*. There are three characteristic values related to this scale: *full trust* – means that the node is fully trustworthy, *initial trust* – is the initial credit given to a node and *cut-off* – is the trust level below which the node is considered untrustworthy.

Each network node participates in the trust management process and maintains data on its trust in other nodes. The corresponding data structure maintained at each node is called *trust table*.
Each node can communicate its trust in other nodes by sending *recommendations*. Recommendation is an entry of the local trust table sent to other nodes. A node can recommend any other node except itself.

While receiving a message, the node makes decisions based on trustworthiness assessment of the sender of the message and the message itself. The assessment is based on two complementary approaches: *Policy-based approach* – the trustor evaluates trustworthiness of the trustee assessing its observed behaviour and its conformance with agreed policies, notably the security policy, and *Reputation-based approach* – the trustor takes into account recommendations from other nodes if they trust the trustee. The influence of these two viewpoints is controlled by the *Cooperation Factor* (CF), same for all nodes in the network. The value of CF is being defined during Step 1 of WCT2M instantiation. CF assumes values from 0 to 1, where 0 means that recommendations are discarded in calculations and 1 means that the trust assessment is solely based on recommendations.

Trust value assigned to the sender of a message and maintained by the receiver of the message can change in effect of the assessment of an incoming message. If node B sends a message to node A and the message violates the security policies assumed by A, B's trust value maintained by A is decreased by $change_{negative}$ factor:

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

and in case the message agrees with the security policies at the receiving node, the trust value of the sender increases by $change_{positive}$ factor:

$$Reputation_B += (1 - Reputation_B) change_{positive}$$

where both, $change_{negative}$ and $change_{positive}$ are real numbers belonging to [0, 1]. WCT2M software running on a node can call a service returning the result of security assessment of an incoming message. This service is not a part of WCT2M software. WCT2M assumes that the result of this assessment is a binary value where 0 means that the message violates the corresponding security policy and 1 means the opposite. At each node, *validity history* is a data structure representing, for each sender node, the history of such evaluations of the incoming messages. The validity history is implemented as a FIFO queue of fixed size, called *validity history table*.

In addition to its own assessment of the trustworthiness of its neighboring nodes, each node receives recommendations representing trust assessments from the other nodes' perspective. WCT2M assumes that each node compares such recommendations with the own assessments and the results of such comparisons are represented in a data structure called *deviation history*. At each node and for each neighbouring node, the deviation history is a queue of (binary) values indicating if the recommendation received from the neighbouring node deviates from the own trust assessment of the recommended node. The deviation history is represented at each node as a FIFO queue of fixed size, called *deviation history table*.

5. METHOD DESCRIPTION

This section gives details of WCT2M software structure and operation.

5.1. CLASS MODEL OF WCT2M

The class model of WCT2M software is presented in Figure 3.

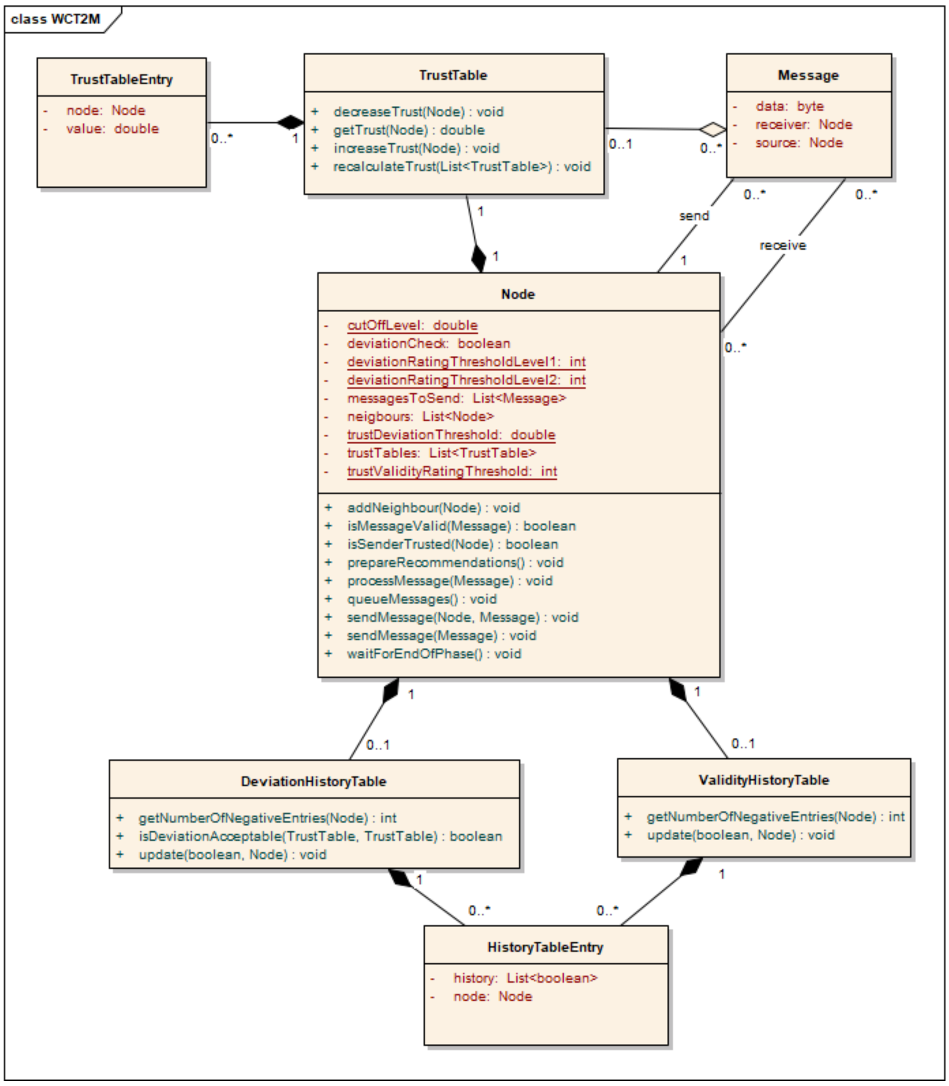

class WCT2M
TrustTableEntry
- node: Node
- value: double
TrustTable
+ decreaseTrust(Node) : void
+ getTrust(Node) : double
+ increaseTrust(Node) : void
+ recalculateTrust(List<TrustTable>) : void
Message
- data: byte
- receiver: Node
- source: Node
0..*
1
0..1
0..*
send
receive
Node
- cutOffLevel: double
- deviationCheck: boolean
- deviationRatingThresholdLevel1: int
- deviationRatingThresholdLevel2: int
- messagesToSend: List<Message>
- neigbours: List<Node>
- trustDeviationThreshold: double
- trustTables: List<TrustTable>
- trustValidityRatingThreshold: int
+ addNeighbour(Node) : void
+ isMessageValid(Message) : boolean
+ isSenderTrusted(Node) : boolean
+ prepareRecommendations() : void
+ processMessage(Message) : void
+ queueMessages() : void
+ sendMessage(Node, Message) : void
+ sendMessage(Message) : void
+ waitForEndOfPhase() : void
DeviationHistoryTable
+ getNumberOfNegativeEntries(Node) : int
+ isDeviationAcceptable(TrustTable, TrustTable) : boolean
+ update(boolean, Node) : void
ValidityHistoryTable
+ getNumberOfNegativeEntries(Node) : int
+ update(boolean, Node) : void
HistoryTableEntry
- history: List<boolean>
- node: Node

Figure 3. WCT2M class model

The model consists of the following classes (the entities stored at the node executing a given method of the specified class are prefixed by 'own', for instance 'own trust table' refers to the trust table stored on the node that executes the method):

- Class `Message` - represents a message (data attribute) exchanged between nodes (`source` and `receiver` attributes)
- Class `TrustTable` - represents trust values assigned to other nodes by a given node. It allows to execute the following methods:

  `decreaseTrust(Node)` - assigns a new, decreased trust value (calculated from the current trust value) to `Node` in the own trust table;

  `getTrust(Node)` returns current trust value assigned to `Node` in the own trust table;

  `increaseTrust(Node)` assigns 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 (the trust tables) received from the neighbouring nodes.

- Class `TrustTableEntry` - represents a single entry of `TrustTable`. It indicates a node and a trust value assigned to it.
- Class `Node` - represents a node of the network (including the base station). It stores the following attributes (the parameters assigned during Step 1 of instantiation of the WCT2M are underlined):

  <u>`cutOffLevel`</u> – cut-off level of the trust scale;

  <u>`deviationCheck`</u> – auxiliary variable storing the deviation check result;

  <u>`deviationRatingThresholdLevel1`</u> – the primary (level 1) deviation rating threshold used to control the number of values inserted to `DeviationHistoryTable`;

  <u>`deviationRatingThresholdLevel2`</u> – the secondary (level 2) deviation rating threshold used to control the number of values inserted to `DeviationHistoryTable`;

  <u>`messagesToSend`</u> – list of Messages to be sent;

  <u>`neighbours`</u> – list of own neighbors;

  <u>`trustDeviationThreshold`</u> – parameter used to calculate values inserted to `DeviationHistoryTable`;

  <u>`trustTables`</u> – auxiliary variable storing a list of recommendations (trust tables received from other nodes);

  <u>`trustValidityRatingThreshold`</u> – threshold used to control the number of values inserted to `ValidityHistoryTable`;

  Class `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) 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 `Node` is considered trusted (its trust value is above the cut-off level);
  `prepareRecommendations()` - prepares a new `Message` containing own trust table to be sent as a recommendation and adds it to the `messagesToSend` list;
  `processMessage(Message)` - processes the incoming `Message` in accordance with own objectives;
  `queueMessages()` – adds messages that the node needs to send (the new messages created by the node and/or the messages received by the router node to be forwarded) to the `messagesToSend` list;
  `sendMessage(Node, Message)` – sends `Message` to `Node` and removes it from the `messagesToSend` list;
  `sendMessage(Message)` - sends `Message` to all own neighbors 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).

- Class `ValidityHistoryTable` - each node maintains a separate copy of this table. For each neighboring node, the table maintains the history of validity evaluations of the messages received from this node. Results of such evaluations (positive or negative) are stored in a FIFO queue of a fixed size. If for a given node the number of negative evaluations exceeds `trustValidityRatingThreshold`, the trust value of this node is decreased by calling `decreaseTrust(Node)` method on the own trust table. Such check and the resulting action (if needed) are being executed after every incoming message validity evaluation performed by the receiving node.

The class allows executing the following methods:

`getNumberOfNegativeEntries(Node)` returns the number of *False* values in the `HistoryTableEntry` of `Node` in the own `ValidityHistoryTable`;

`update(boolean, Node)` inserts to the own `ValidityHistoryTable` the boolean value at the entry pointed to by `Node` (on the FIFO basis).

- Class `DeviationHistoryTable` - each node maintains a separate copy of this table. For each neighboring node, the table maintains the history of deviations resulting from the comparison of the own trust table with the trust table received as a recommendation from this node. When node A receives a recommendation from node B, it calls `isDeviationAcceptable(TrustTableA, TrustTableB)` where `TrustTableA` is its own trust table and `TrustTableB` is the trust table received as a recommendation from B. Then the deviation is calculated in accordance with 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 (the nodes known to B), $Reputation_{A\to n}$ represents the trust value assigned to node *n* in `TrustTableA` and $Recommendation_{B\to n}$ represents the trust value assigned to node *n* in `TrustTableB` . If $deviation_{A-B}$ is below `trustDeviationThreshold`, the deviation evaluation is *True*, otherwise is *False*. For each neighboring node, `DeviationHistoryTable` stores the results of deviation evaluations in a FIFO queue of a fixed size. If for a given node N, the number dev of *False* evaluations stored in this queue satisfies the condition exceeds

deviationRatingThresholdLevel1 < dev≤ deviationRatingThresholdLevel2

where deviationRatingThresholdLevel2 > deviationRatingThresholdLevel1,

the trust to the node is decreased by calling the `decreaseTrust(N)` on the own trust table. If the number of negative assessments in the deviation history table exceeds `deviationRatingThresholdLevel2`, the trust to the node is decreased twice by two calls of `decreaseTrust(N)`. This check and the resulting action (if needed) are executed after receiving a valid (positively verified by `isMessageValid(Message)`) recommendation from N.

The class allows to execute the following methods:

`getNumberOfNegativeEntries(Node)` returns the number of *False* values in the *`HistoryTableEntry`* of `Node` in the own `DeviationHistoryTable`;

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

- Class *`HistoryTableEntry`* - represents a single entry of `ValidityHistoryTable` or of `DeviationHistoryTable`.

5.2. Execution Model Of WCT2M

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

- *Data phase*: Each node sends/receives ‘regular’ messages to/from its neighbours and forwards messages received from other nodes, if it is a router node. Based on the received messages, the receiver node (the trustor) evaluates the sender nodes (the trustees) if they can be trusted and then updates the corresponding trust values in its own trust table.
- *Recommendation phase*: Each node sends recommendations to its neighbours. The recommendations are related to the end of the data phase (the trust table entries exchanged as recommendations are the snapshots taken at the end of the data phase, after all updates related to the incoming messages have been inserted). The receiver node (the trustor) evaluates the sender nodes (the trustees) based on its own trust table and on the incoming recommendation messages. If they are valid and can be trusted, the receiver recalculates its trust table referring to the received valid recommendations (the recommendations from untrusted nodes and the recommendations assessed as being invalid are discarded).

The exact lengths of the data phase and the recommendation phase depend on the MAC protocol used in the network, the sleep scheduling pattern adopted and the density of the network (two interfering nodes cannot transmit at the same time because of message collisions [33], so in dense networks recommendation exchange takes more time than in sparse ones). It means that the time length of cycles can vary during the whole life of a network. The idea of WCT2M cycles and phases is presented in Figure 4.

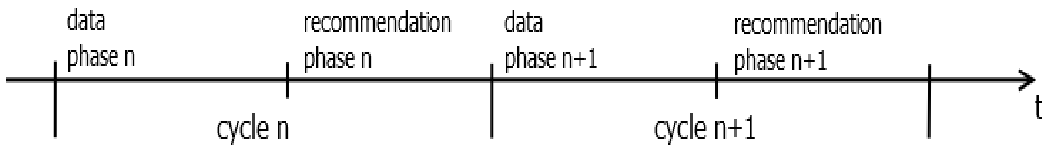


Figure 4. Cycles and phases of the WTC2M enabled network

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. If needed, a node can skip sending messages during a given phase (to avoid delaying the phase termination).

6. Experimental Evaluation

To assess WCT2M we applied the following approach: (1) Identified criteria and the corresponding metrics for assessing the efficiency of WCT2M; (2) Developed a laboratory WSN system to demonstrate feasibility of WCT2M (the laboratory system implements a small WSN of Figure 1); (3) Because of scale limitations of the laboratory system, we used simulations to assess the performance of WCT2M for larger networks. After reviewing the existing simulators

(including the assessment of their availability and cost), we decided to develop a dedicated simulator. The laboratory system was used to calibrate the simulator and then the simulator was used to evaluate WCT2M efficiency for different attack scenarios and different network sizes; (4) A set of experiments has been carried out to assess WCT2M efficiency in detecting and isolating malicious nodes implementing different attack scenarios and to assess how the effectiveness of the security mechanisms deployed on network nodes influences WCT2M efficiency.
The laboratory network was created using elements of CC2520 Development Kit [24]. CC2520 is a Texas Instrument second generation ZigBee/IEEE 802.15.4 RF transceiver for the 2.4GHz unlicensed ISM band.

The dedicated simulator, called WSN Cooperative Trust Management Simulator (WCTMS), was written in NetBeans IDE [25], using Java 8 [26] with JGraph library [27]. WCTMS simulates the execution model of WCTM where each cycle of the network (see Figure 4) corresponds to *a simulation turn* during simulation run. Assuming that the cycles of the network have the same time duration, the number of simulation turns can be interpreted as a measure of time delay in the simulated network. And for a given simulation run and two different states of the simulated network, S1 and S2, the number of simulation turns transferring the network from state S1 to state S2 is called *simulation distance* between S1 and S2.

The simulator implements probabilistic model of network operation and of different threats against the simulated network: p_{nd} – is a probability of sending a message by a node in the data phase of a cycle (nodes activity), p_b – is the probability of sending a message by the base station in the data phase of a cycle (base station activity), p_{nr} – is the probability of sending the trust table to the neighbors in a recommendation phase of a cycle, p_s – is the probability of spoiling a message by a malicious node, p_e – is the probability of sending a spam message by a malicious node, p_d – is the probability of blocking a message by a malicious node, p_{ch} – is the probability of modifying a message by a malicious node, p_t – is the probability of modifying a trust table by a malicious node.

In addition: r – is the probability of recognizing a spoiled message by the receiver node (effectiveness of the security mechanisms installed on network nodes) and e – is the probability of spoiling a message during transmission.

6.1. Strategy Of Evaluation

A set of laboratory experiments was conducted using the network illustrated in Figure 1 and these experiments were also repeated with the help of the WCTMS simulator to check if the results of laboratory experiments and the simulations converge. Then an additional set of simulation experiments was conducted using WCTMS. These experiments involved larger networks which could not be directly implemented in the laboratory because of the limited resources. The experiments were focused on evaluating the efficiency of isolating malicious nodes in the network.

Metrics supporting the analyses were identified using the Goal-Question-Metrics (GQM) methodology [28]. The overall goal of the experiments was defined as follows: analyse WCT2M for the purpose of assessing the efficiency of malicious nodes isolation for different attack scenarios. Below we give details of two metrics used in the experiments (for more details see [29]): ***Average All Nodes Detected (AAND)*** [Real]: average number of WCT2M cycles needed to detect all malicious nodes, calculated for all simulation runs of a given test case; ***Median All Nodes Detected (MAND)*** [Real]: median number of WCT2M cycles needed to detect all malicious nodes, calculated for all simulation runs of a given test case.

6.2. Scope Of Experiments

The experiments were divided into groups. During simulations, the simulation distance was interpreted as a measure of time delay between the initial state of the network and the state where all malicious nodes have been successfully detected. The groups of experiments were as follows.

EXP1: Feasibility of implementing WCT2M: The objective of this experiment was to demonstrate that WCT2M can be implemented in a typical laboratory environment. The resistance of WCT2M to network attacks: spam attack, black hole attack, message modification attack and decreased frequency attack was assessed experimentally. The laboratory experiments were then repeated using WCTMS simulator to verify if the results of laboratory and simulation experiments converge.

EXP2: Resistance to various network attacks (with possibly modified frequency of the attack): The objective of this experiment was to examine the efficiency of malicious nodes detection and isolation with respect to different network attacks assuming that the malicious nodes can modify frequency of the attack to avoid detection. Each simulation run involved a network of up to 1000 randomly distributed nodes.

EXP3: Resistance to collusion attack: The objective of this experiment was to examine the resistance of WCT2M to collusion attack. Each simulation run involved a network of up to 300 randomly distributed nodes.

EXP4: Influence of the effectiveness of security mechanisms: The objective of this experiment was to examine how the effectiveness of security mechanisms implemented in the nodes impacts the time delay of detecting the malicious nodes. Each simulation run involved a network of up to 300 randomly distributed nodes.

7. Results Of Experiments

In EXP1, the WCT2M software package was installed on the nodes of the laboratory network and then numerous test cases were run, representing different attack scenarios. Then, each test case was also repeated using WCTMS, with the same parameters and layout of nodes. For each test case 100 simulation runs were executed and the results were compared with the results obtained in the laboratory to verify if the laboratory and the simulation results converge. The experiments demonstrated that WCT2M is feasible and can be implemented on real-world devices in a typical WSN environment. The experiment results also demonstrated that WCT2M is able to protect WSN against typical attacks and that the deviation history mechanism of WCT2M is an effective tool to counteract the decreased frequency attack against the trust management mechanism.
In EXP2, the nodes were distributed in the square area (100 x 100 distance units) divided into 16 clusters of equal size (two-tier network). The base station was placed outside the nodes distribution area. The node signal range was set to Z_{cl} = 30 distance units for both, the base station and the cluster heads, and Z_n = 5 distance units for the remaining nodes. It was assumed that during a given test case, the nodes do not change their positions.

The results for a network of 1000 nodes and for 10 malicious nodes are shown in Table 1.
Experiment EXP3 was to assess WTC2M efficiency while the validity history mechanism was enabled. During experiments, the influence of the collusion attack was represented by parameter p_t (probability that the trust table gets modified in effect of collusion). The experiments were conducted for networks of up to 300 nodes and for different numbers of colluding nodes (up to the half of the network size). Each test case involved 100 simulation runs.

Table 1. EXP2 results (the deviation history mechanism on)

Test case	Parameter value	Metrics

		MAND	AAND
Spam attack	$p_e = 80\%$	3.0	6.93
Black hole attack	$p_d = 80\%$	15.0	29.70
Message modification attack	$p_{ch} = 80\%$	29.5	47.69

Figure 5 presents the result - the median time of detecting all colluding nodes. The underlying assumption was that the probability of an adverse modification of the own trust table by a malicious node p_t=50%. The simulated network comprised 300 nodes.

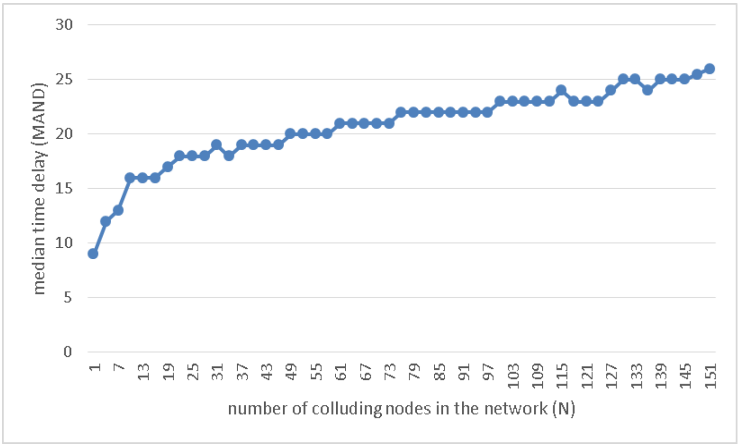


Figure 5. Median delay of detecting all colluding nodes (validity history mechanism on)

The objective of EXP4 experiment was to assess efficiency of WCT2M depending on the effectiveness of the security mechanisms implemented in the nodes (this effectiveness is represented by *r* parameter in Table 2 and Table 3). For instance, *r* = 90% means that there is 90% probability that the security mechanism detects that an incoming message violates the compulsory security policies. During this experiment we examined networks of different size (up to 300 nodes) for different values of r (100%, 95%, 90%, 85%, 80%, 75%, 70%, 65%, 60%, 55%, 50%) and for different numbers of malicious nodes in the network. Figure 6 presents the MAND metric for a network of 300 nodes (10 of which are being malicious).

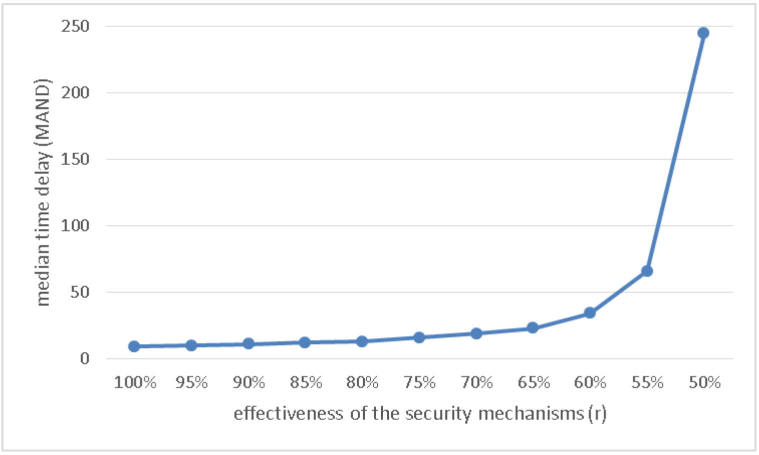


Figure 6. Median time delay of detecting all faulty nodes

From Figure 6, it can be observed that the effectiveness of security mechanisms highly impacts efficiency of WCT2M, if r drops below 65%.The experiment shows that if $r \geq 65\%$, the malicious nodes are detected with the delay which is inversely proportional to the value of *r*. However, if the value of *r* drops below 65%, the efficiency of malicious nodes detection decreases rapidly.

The explanation of this observation is as follows. The mechanism adopted in WCT2M for changing the trust level in trust tables is not 'linear' what is presented in Figure 7. The trust increase curve illustrates the increase of trust in effect of positive events (where a positive event is this which affects trust positively, for instance a positive recommendation or a positive assessment of the incoming message). The trust decrease curve illustrates how trust decreases in effect of negative events (the events that lead to trust decrease, like negative assessment of an incoming message or receiving a negative recommendation). On the X-axis we have events affecting trust (positive or negative) and on Y-axis we have trust levels. For instance, from Figure 7 we can see that 7 subsequent negative events are sufficient to drop from full trust (trust level = 1) down to the cut-off level (trust level = 0.2). And we need 25 subsequent positive events to increase trust from the level = 0.5 to the level 0.9.

The increase of trust in effect of a positive event is bigger if the current level of trust is lower, and then it gets 'saturated' (the subsequent positive events have less impact on trust). In contrary, the loss of trust (in case of negative events) is much faster and much less dependents on the current level of trust. In effect, the same numbers of positive and negative events can result in a significant trust decrease, depending on the current level of trust. And if the number of positive events grows (comparing to the number of negative events), at some point they can 'outweigh' the influence of the negative events resulting in that the isolation of the malicious node is getting delayed.

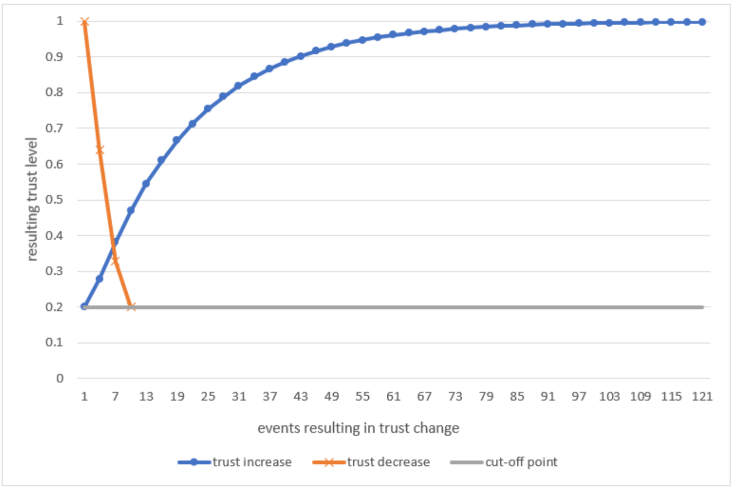


Figure 7. Trust level depending on number of events causing trust change

In addition, in the experiment illustrated in Figure 6, the probability p_s of spoiling a message by a malicious node was 90% which means that 10% of the sent messages were non-malicious and caused the rise of the reputation of the sending node. The result shown in Figure 6 is the cumulative effect of the influence of the negative events and the positive ones (which number was growing reverse proportionally to the value of parameter *r*).

8. CONCLUSIONS

The paper presents WTC2M, a new method of trust management in clustered wireless sensor networks. The method is based on a distributed trust management model. The main innovation of WTC2M is that in addition to two trust related mechanisms, namely the local assessment of the validity of incoming messages and the recommendations received from other nodes it introduces two additional mechanisms, the deviation history and the validity history, and integrates them all into a single trust management mechanism able to efficiently counteract the attacks against the network and against the trust management mechanism. The mechanism takes into account the histories of local assessments and of the received recommendations (the lengths of these histories

are the parameters of the method). It increases the efficiency of WTC2M in situations where the malicious nodes modify their behaviours to avoid detection. The efficiency of WCT2M in detecting and isolating the malicious nodes for different attack scenarios has been evaluated in laboratory and simulation experiments. Four different experiments were conducted, each involving multiple test cases and multiple simulation runs for each test case.
The considered attacks are representative for a wide range of more specialized attacks. For instance, the selective forwarding attack (dropping a part or whole of a forwarded message) can be considered as a combination of black hole attack (dropping a message) and message modification attack (modifying a message). Node capture is a prerequisite for all attack types considered in the paper and therefore is a part of each of them.

WCT2M is neutral with respect to network protocols because it is decentralized (cluster heads do not have any special role in WCT2M based trust management) and is implemented on top of the stack of protocols. In some cases however, the method can have positive impact on these protocols, for instance by taking into account the current trust value while voting on a new cluster-head.

During experiments it was assumed that the nodes use broadcasting to distribute recommendations. If this is a problem, WCT2M can be easily modified to sending the recommendations directly instead of broadcasting them. In this research we did not address the problem of when to switch between broadcasting and direct sending of recommendations (which can be a topic to be addressed in further research).

There are numerous questions which are in the scope of further research. So far, the experiments were restricted to the fixed networks and it would be interesting to investigate how WCT2M performs in mobile networks which may involve clusters reformation. In such situation, malicious nodes could assume the strategy of changing clusters to follow these where they have the highest trust values. Another interesting research issue is the gossip problem. As each node is a neighbour of just few other nodes, it learns about the remaining nodes indirectly, by exchanging trust tables received as recommendations. It means that most of the nodes know others only from a kind of ‘gossip’. This indirect knowledge is exchanged in both directions, to and from a node, and it would be interesting to examine how this gossip information may influence the precision and efficiency of WTC2M. Another issue is to investigate WCT2M behaviour while facing more intelligent attacks. The present attack models implemented in WCTMS simulator are based on the probabilistic characterization of network behaviour where the probabilities are static in the sense that they do not change during simulations (researching towards more intelligent attack models, including attacks being a combination of several simpler attacks will require further expansion of WCTMS).

ACKNOWLEDGEMENTS

This work was partially supported by the Statutory Grants 2014-2017 by Polish Ministry of Science and Higher Education.

REFERENCES

[1] Y. Zhiyng, K. Daeyoung, L. Insun, K. Kiyoung and J. Jongsoo, "A Security Framework with Trust Management for Sensor Networks," in Proc. Workshop of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks, Athens, IEEE, 2005, pp. 184-192.

[2] ZigBee Alliance, "Standards: ZigBee Specification," 28 01 2017. [Online]. Available: http://www.zigbee.org/download/standards-zigbee-specification/. [Accessed 05 11 2017].

[3] IEEE, "Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs)," IEEE Press, New York, 2003.

[4] ANGEL, "Description of Final Angel Demonstrator," 6th FR STREP Project: Advanced Networked embedded platform as a Gateway to Enhance quality of Life (ANGEL), Deliverable 5.2, 2008.

[5] A. Keshavarzian, H. Lee and L. Venkatraman, "Wakeup Scheduling in Wireless Sensor Networks," in Proceeding of the 7th ACM International Symposium on Mobile ad hoc networking and computing, Florence, Italy, ACM, 2006, pp. 322-333.

[6] J. Górski and A. Turower, "Assessing the time effectiveness of trust management in fully synchronised wireless sensor networks," in Design, development and implementation of real-time systems; Trybus L. & Mastalerz M.W. (eds), Szczecin, Poland, PTI, 2013, pp. 31-42.

[7] N. I. Blatt, "Operational Trust: A New Look at the Human Requirement in Network Centric Warfare," in 9th International Command and Control Research and Technology Symposium (ICCRTS): Coalition Transformation: An Evolution of People, Processes and Technology to Enhance Interoperability; Leoni Warne (ed), Copenhagen, Danmark, Command and Control Research Program (US), 2004, pp. 1-17.

[8] J. Górski, A. Turower and A. Wardziński, "Distributed Trust Management Model for Wireless Sensor Networks," in Monographs of system dependability: Problems of dependability and modelling; Mazurkiewicz J., Sugier J., Walkowiak T. & Michalska K. (eds), Wrocław, Poland, Oficyna WYdawnicza Politechniki Wrocławskiej, 2011, pp. 83-94.

[9] T. Zahariadis, H. C. Leligou, P. Trakadas and S. Voliotis, "Trust management in wireless sensor networks," Transactions on Emerging Telecommunications Technologies, vol. 21, pp. 386-395, 2010.

[10] 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, vol. 35, pp. 867-880, 2012.

[11] J. Lopez, R. Roman, I. Agudo and C. Fernandez-Gago, "Trust management systems for wireless sensor networks: Best practices," Journal of Network and Computer Applications, vol. 35, vol. 33, pp. 1086-1093, 2010.

[12] 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," in Computer Safety, Reliability, and Security. SAFECOMP 2014, Lecture Notes in Computer Science; Bondavalli A., Di Giandomenico F. (eds), vol. 1, Florence, Italy, Springer, 2014, pp. 214-229.

[13] R. Abassi and S. G. El Fatmi, "Countering the Collusion Attack in a Trust-based MANET," in Proceedings of the European, Mediterranean & Middle Eastern Conference on Information Systems 2015; Kostantinos Lambrinoudakis, Vincenzo Morabito & Marinos Themistocleous (eds)., vol. 1, Athens, Greece, 2015, pp. 575-585.

[14] R. Perlman, "An overview of PKI trust models," IEEE Network, vol. 36, vol. 13, no. 6, pp. 38-43, 1999.

[15] H. Chen, H. Wu, X. Zhou and C. Gao, "Agent-based Trust Model in Wireless Sensor Networks," in Proc. International Conference on Multimedia and Ubiquitous Engineering, Busan, Korea, IEEE, 2008, pp. 150-154.

[16] H. Chen, H. Wu, X. Zhou and C. Gao, "Reputation-based Trust in Wireless Sensor Networks," in Proc. International Conference on Multimedia and Ubiquitous Engineering, Seoul, Korea, IEEE, 2007, pp. 603-607.

[17] A. Boukerche and X. Li, "An Agent-based Trust and Reputation Management Scheme for Wireless Sensor Networks," in Proc. IEEE Global Telecommunications Conference, St. Louis, USA, IEEE, 2005, pp. 1857-1861.

[18] J. Konorski and R. Orlikowski, "DST-Based Detection of Non-cooperative Forwarding Behavior of MANET and WSN Nodes," in ireless and Mobile Networking. IFIP Advances in Information and Communication Technology, vol 308, Berlin, Springer, 2009, pp. 185-196.

[19] N. Labraoui, "A reliable trust management scheme in wireless sensor networks," in 12th International Symposium on Programming and Systems, Algiers, Algeria, IEEE, 2015, pp. 1-6.

[20] W. Fang, W. Zhang, Y. Yang, Y. Liu and W. Chen, "A resilient trust management scheme for defending against reputation time-varying attacks based on BETA distribution," Science China Information Sciences, vol. 60, no. 4, pp. 1-11, 2017.

[21] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," in First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA 03), Anchorage, Alaska, IEEE, 2003, pp. 113-127.

[22] Y. Sun, Z. Han and K. Liu, "Defense of Trust Management Vulnerabilities in Distributed Networks," Communications Magazine, vol. 46, no. 46, pp. 112-119, 31 March 2008.

[23] B. Mamalis, D. Gavalas, C. Konstantopoulos and G. Pantziou, "Clustering in Wireless Sensor Networks," in RFID and Sensor Networks: Architectures, Protocols, Security, and Integrations, L. Y. Y. Zhang (ed.), CRC Press, 2009, pp. 324-353.

[24] Texas Instruments, "CC2520 Development Kit," 10 07 2017. [Online]. Available: http://www.ti.com/tool/cc2520dk. [Accessed 05 11 2017].

[25] NetBeans, "NetBeans IDE Features," 10 07 2017. [Online]. Available: https://netbeans.org/features/index.html. [Accessed 05 11 2017].

[26] Java, "Java," 10 07 2017. [Online]. Available: http://www.java.com. [Accessed 05 11 2017].

[27] JGraph Ltd, "JGraph," 10 07 2017. [Online]. Available: http://www.jgraph.com/. [Accessed 05 11 2017].

[28] 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.

[29] A. Turower, "Trust Management Method for Wireless Sensor Networks (Doctoral dissertation)," 2017. [Online]. Available: https://mostwiedzy.pl/publication/download/1/trust-management-method-for-wireless-sensor-networks_16168.pdf. [Accessed 5 11 2017].

[30] 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.

[31] 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.

[32] 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.

[33] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in Proc. INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, New York, USA, IEEE, 2002, pp. 1567-1576. [2] Gizem, Aksahya & Ayese, Ozcan (2009) Coomunications & Networks, Network Books, ABC Publishers.

AUTHORS

Janusz Górski is a professor at the Department of Software Engineering, Gdansk University of Technology in Poland where he is leading the Information Assurance Group (IAG) http://iag.pg.gda.pl/iag/.

Alan Turower is presently a cloud software developer and a design lead at Intel Technology Poland company.