WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2011; **11**:371–391
Published online 6 July 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.984

SPECIAL ISSUE PAPER

A security framework for wireless mesh networks†

Parag S. Mogre*, Kalman Graffi, Matthias Hollick and Ralf Steinmetz

Multimedia Communications Lab, Technische Universität Darmstadt, Rundeturmstr. 10, 64283 Darmstadt, Germany

ABSTRACT

The class of Wireless Mesh Networks (WMN) supports an ample set of applications including wireless community networks, radio access networks in rural or metropolitan areas, or wireless backbones for factory/process automation. Guaranteeing security is crucial for within these application scenarios. While contemporary wireless technologies, such as the IEEE 802.16 or the IEEE 802.11s standard, provide the basic protocol mechanisms for mesh networking, they lack in comprehensive security mechanisms. Additionally, novel security features of the above standards such as per-link encryption break existing security solutions that rely on overhearing of the wireless channel. We close this gap by developing a holistic approach toward securing WMNs with particular focus on the network layer. We perform a threat analysis and then develop solutions (1) guaranteeing the integrity and authenticity of routing messages, (2) to locally and globally detect misbehavior of nodes in forwarding data or routing messages even for settings that do not allow for overhearing the channel, and (3) to dynamically manage reputation of nodes throughout the network. The combination of these building blocks enables to provide for secure, self-organizing WMNs. As a proof-of-concept, we tailor and implement our solutions for the setting of a realistic IEEE 802.16 mesh network; we discuss the protection achieved and assess selected performance trade-offs for the developed mechanisms. Copyright © 2010 John Wiley & Sons, Ltd.

KEYWORDS

wireless mesh networks; security; dependable routing; misbehavior detection

***Correspondence**

Parag S. Mogre, Multimedia Communications Lab, Technische Universität Darmstadt, Rundeturmstr. 10, 64283 Darmstadt, Germany.
E-mail: pmogre@kom.tu-darmstadt.de

1. INTRODUCTION

Wireless networks have been tremendously successful during the recent decade. The class of multihop wireless networks promises to supplement existing cellular (wireless single-hop) networks and to increase the achievable capacity, reach, and throughput of these networks. In particular, Wireless Mesh Networks (WMNs) support a variety of applications including wireless community networks, radio access networks in rural or metropolitan areas, or wireless backbones for factory/process automation (see e.g. Reference [3] for a survey on WMNs).

Self-organizing features to establish and maintain the network allow for organic growth of WMNs, while keeping the operational costs to a minimum. The network can either build on user-controlled devices (e.g. in community network scenarios), or utilize dedicated mesh nodes (e.g. in provider-operated networks). In contrast to *ad hoc* networks, we assume the nodes to form a quasi-static topology and not to be resource constrained.

Guaranteeing security is crucial for within the given application scenarios. The multihop nature of WMNs makes provisioning of security mechanisms specially challenging, though. WMNs are based on the collaboration of nodes for routing and forwarding of data, the wireless channel is particularly vulnerable to attacks, malicious or selfish nodes can severely affect the dependability of the network, etc.

Despite the fact that contemporary wireless technologies as specified in the IEEE 802.16 or the IEEE 802.11s standard provide the basic protocol mechanisms for mesh networking, they lack in comprehensive security mechanisms. Additionally, novel security features of the above standards such as per-link encryption break existing security solutions on other layers, e.g. mechanisms to detect routing/forwarding misbehavior by means of overhearing the wireless channel. Hence, several challenges coupled to securing WMNs persist. We argue that a holistic perspective is necessary to secure WMNs. This includes prevention of, detection of, and reaction to attacks or malfunctioning of nodes.

† This paper is an extended version of the research presented in our paper originally presented at the IEEE LCN 2007 [1], it incorporates also results from our paper presented at GlobeCom 2007 [2].

Copyright © 2010 John Wiley & Sons, Ltd.

We address WMN security by defining and realizing a comprehensive security framework. Our contribution is as follows:

(1) We present a secure routing solution for WMNs. It comprises a probabilistic, proactive multipath routing algorithm, which is resilient against forging, modification and dropping attacks. It guarantees the integrity and authenticity of the routing messages.
(2) We develop a mechanism to locally detect forwarding misbehavior (for both routing as well as data messages). It is able to cope with per-link encryption. Both malicious packet dropping and packet modification at misbehaving nodes are detected, the misbehaving nodes are identified.
(3) A lightweight global (end-to-end) mechanism for detection of colluding misbehaving nodes is designed. It complements the local mechanism and allows to protect the global integrity of the network's operation.
(4) A reputation management system glues together the above components of the security framework. It allows nodes to assess the trustworthiness of neighboring nodes and is used to trigger corrective actions if a misbehaving node is detected.

This paper is a revised and extended version of References [1,2]. It adds a security analysis of the proposed framework and gives additional results demonstrating the performance of the schemes. We further extended the discussion of the state-of-the-art and included details on the authenticated network entry process.

As a proof of concept, we develop and implement a concrete solution for our security framework, which is tailored for the setting of the IEEE 802.16 MeSH mode. We have chosen a probabilistic routing scheme (*AntNet*) as a suitable instantiation of WMN routing. We design and develop (1) *AntSec*, (2) *WatchAnt*, (3) *LeakDetector* as well as (4) *AntRep* to embody our framework. The probabilistic, stigmergy-based solutions presented are able to optimize their performance with increasing time of operation of the WMN if we assume the network topology to be quasi static as, e.g. in community WMNs.

The remainder of this paper is structured as follows. In Section 2 we define the scope of our work and outline the security goals for our framework. In Section 3 we investigate related work, classify it and identify the shortcomings of existing solutions w.r.t. their applicability for contemporary WMN technologies. Section 4 describes basic security mechanisms for our solution. In Section 5 and 6 we present the individual building blocks of our security framework. We particularly focus on the description of the four core mechanisms developed. We outline the interaction of the mechanisms and discuss pros and cons. Sections 7 and 8 provide an evaluation of our proposed solutions. This includes the performance evaluation of selected parts of the framework as well as a security analysis of the system. This is followed by a conclusion and pointers for further research in Section 9.

2. SCOPE, RESEARCH PROBLEM, AND SECURITY GOALS

We next illustrate selected open research challenges in the area of WMN security. For within this work, we focus on addressing these challenges comprehensively. Subsequently, we state the security goals for the mechanisms within our security framework.

2.1. Scope and system overview

WMNs come in different forms such as subscription-based or subscription-less, open or closed networks. The term open network refers to the possibility that new nodes can join the existing WMN in an organic manner. The term subscription-based identifies a WMN which is deployed by a network provider and only nodes which are registered with the network provider are allowed to join the network. In contrast, in a subscription-less network there exists no network provider, but arbitrary nodes are allowed to join. Here, we focus on open, subscription-based mesh networks and assume the existence of a trusted third party (the network provider). Each node that wants to gain access to the network has to be authorized out-of-band to allow us to punish misbehavior. An example for such a WMN could be a mesh network using the IEEE 802.16 standard's MeSH mode (see Reference [4] for an introduction to the MeSH mode of the IEEE 802.16 standard [5]).

We further assume that the communication takes place over a shared wireless medium, where the nodes are able to both send as well as receive data. Links between nodes are assumed to be bidirectional, i.e. given a link $L(A,B)$ between nodes A and B in the WMN there exists the link $L(B,A)$. The WMN may deploy per-link encryption at the Medium Access Control (MAC) layer. Thus, data transmissions on link $L(A,B)$ can be encrypted by A such that only neighbor B is able to decode the transmitted data. Please note that our solution is applicable to mesh networks with and without per-link encryption, the mechanisms can be extended to operate in subscription-less networks.

We address the security in WMNs from a holistic perspective. To this end, we propose a security framework that consists of four main building blocks. It comprises (1) a secure routing protocol for within WMNs, which is resilient against forging, modification and dropping attacks; (2) a mechanism to locally detect node misbehavior; (3) a global (end-to-end) mechanism for detection of colluding misbehavior of nodes, which cannot be locally detected; (4) a reputation management system that glues the above components together.

Please note that with our work we do aim at addressing as yet unsolved security challenges in WMNs, hence in the following we mainly focus on the unique and novel aspects

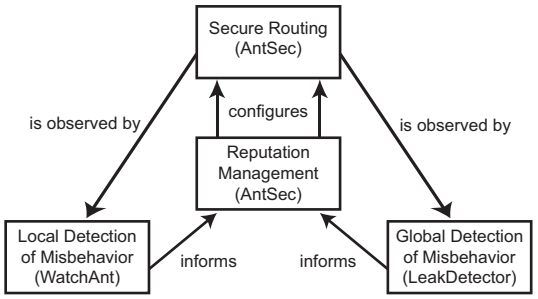


Figure 1. WMN security framework overview.

of our work. The novelty of the solution presented lies in the consideration of (A) probabilistic routing schemes and (B) the ability to operate in WMNs technologies using per-link encryption. As one possible instantiation of our security framework, protocols and system components are designed and prototypically implemented. These are: (1) AntSec; (2) WatchAnt, (3) LeakDetector, and (4) AntRep. Figure 1 gives an overview of the structure of our framework.

2.2. Secure routing

Multihop routing forms the basis of WMNs. Messages are transferred between the end systems using hop-by-hop forwarding *via* intermediate nodes. Various security solutions exist to provide a secure routing functionality (see also Section 3). For example, several secure routing algorithms have been designed to protect the integrity and authenticity of routing messages, and for preventing these messages to be maliciously dropped. While secure routing has been extensively studied in related work for deterministic routing protocols, the class of probabilistic routing algorithms is not yet covered in greater detail. In our work, we propose solutions for the latter class of routing protocols, since it is well suited for WMNs.

To aid our misbehavior detection mechanisms, we design the routing protocol such that for each packet a node N receives, the node is able to obtain information about the previous two hops along which the packet was forwarded to the node N. For example, if the packet travels along a path S-N_1-N_2-N_3-N then the node N knows that the two previous nodes which forwarded the packet are nodes N_2 and N_3.

2.3. Local and global misbehavior detection

To ensure the correct operation of the routing system, the nodes in WMNs need to cooperate and forward messages of other nodes according to the protocol specification. However, for an individual node it might be beneficial *not* to forward messages of other nodes, as this requires the expense of resources like computational power, energy, and bandwidth. Thus, from an opportunistic node's perspective it can be more valuable to silently drop messages of other nodes or to avoid being part of routes between two other end

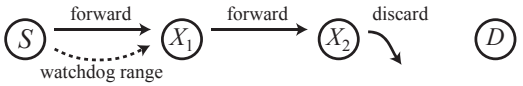


Figure 2. Example of colluding misbehavior: X_1 forwards messages to X_2, which drops these messages. S observes 'correct' behavior of X_1. X_1 ignores X_2's misbehavior.

systems. The prevention of or reaction to node misbehavior is important to ensure the dependable, i.e. well performing and uninterrupted operation of the WMN. In addition to secure routing, for detecting forwarding misbehavior, additional mechanisms are needed to monitor the forwarding of data messages. We distinguish these into local and global mechanisms.

In Reference [6], Marti *et al*. propose a mechanism called *watchdog* to detect node misbehavior. The watchdog mechanism is employed by each node individually to observe the messages sent by neighboring nodes. Comparing the overheard messages with a list of messages that have to be forwarded reveals, whether the observed node is forwarding messages appropriately or not. However, the reach of solutions incorporating the watchdog principle is limited to the one-hop neighborhood only. They fail to detect forwarding misbehavior if malicious nodes collude (or form a malicious subnet). Messages are accepted from the malicious node/subnet, but dropped as soon as no benign[‡] node is able to observe the routing behavior.

We define the following behavior of a node X_2 as *maliciously colluding*: 'A node X_2 acts *maliciously colluding*, if it (selectively) drops messages received from a neighboring malicious colluding node X_1 and the messages have not been originated by colluding malicious nodes'. Messages originated by benign nodes and received by malicious colluding nodes are forwarded in the first instances to another malicious colluding node which then drops the messages.

Let us consider the scenario presented in Figure 2. Here, a mechanism to detect one-hop forwarding misbehavior fails.

- Source S is sending packets *via* X_1 to destination D. S recognizes that X_1 is forwarding all packets to X_2.
- X_1 forwards all packets received from benign nodes (in this case to X_2).
- X_2 drops packets which were not generated by malicious nodes, but received from a colluding malicious node.
- X_1 is able to detect the misbehavior of X_2. Since X_1 and X_2 collude, X_1 silently accepts the misbehavior, which goes unnoticed for the benign nodes S and D.

One-hop mechanisms that detect forwarding misbehavior of neighboring nodes are functional in this scenario as well; however, in this scenario, the detecting node is colluding with the dropping node. Thus, the misbehavior is

[‡] 'Benign' or 'well-behaved' nodes operate as specified by the respective routing protocol, thus supporting the routing process.

not punished. We see that two colluding nodes can conceal their forwarding misbehavior from other nodes, which has to be considered a severe attack on the functionality of WMNs. Depending on the deployed routing algorithm, malicious collusion might prohibit regular communication over multihop routes.

2.4. Reputation management

A central management instance is necessary to put together the information from the misbehavior detection schemes and to react to node misbehavior by appropriately configuring the secure routing scheme. In related work a number of reputation management approaches both centralized as well as decentralized have been explored (see also Section 5). For within our work, a reputation management system has to honor the probabilistic nature of the routing protocol. Also, since WMNs build on potentially unreliable wireless communication links, we have to assume imprecision of the misbehavior detection component. Thus, we aim to provide a reputation management solution that is adaptable and self-stabilizing, i.e. a solution that is able to deal with the probabilistic nature of the routing scheme that is robust to misclassifications. For the case of open networks, the integration of new nodes and the re-integration of formerly malicious nodes into the network is a desired feature as well—if these start collaborating again to contribute to the welfare of the network.

2.5. Security goals

The security goals we consider when designing our solution encompass goals for nodes individually as well as goals for the entire routing system/network (i.e. control-plane and data-plane). They are as follows:

- Authenticity and authorization of the source and destination node can be verified by all nodes on the route.
- Authenticity and authorization of neighbors can be verified by nodes en-route.
- Correct routing functionality shall be maintained (e.g. loop free routing, up-to-date routes, etc.).
- Forging of routing messages in the name of other nodes shall have no effect on routing.
- Manipulation or dropping of routing messages shall be detected.
- Manipulation or dropping of data packets shall be detected.
- Misbehaving nodes shall be detected and identified, so that various punishment methods can be applied. This includes also collusion of nodes.

In addition to meeting these security goals, a solution should be as efficient in terms of computational complexity as possible: i.e. the usage of asymmetric cryptography should be avoided for frequent operations such as packet forwarding. Routing overhead should be kept as low as possible, unnecessary transmissions should be avoided, etc. To harness the power of decentralized operation of the network, the developed solutions should base routing and security decisions on local information, wherever this is possible.

3. RELATED WORK

The detection of malicious nodes that refuse to forward messages is a challenging task in decentralized, open networks. This is particularly true for wireless multihop networks such as WMNs. In the area of mobile *ad hoc* networks, recently a number of security solutions have been discussed. Various approaches build on reactive routing protocols for wireless multihop networks: Dynamic Source Routing (*DSR*) [7] and Ad-hoc On-demand Distance Vector (*AODV*) [8] and reach a variety of specialized security goals, typically by utilizing asymmetric and/or symmetric cryptography in combination with protocol mechanisms. See References [9] and [10] for an up-to-date survey on security issues and solutions in such networks.

Secure routing algorithms typically protect the route setup and maintenance phase. They counter various attacks such as forging, modifying, or dropping of routing messages. For example, *Secure DSR* [11], *Ariadne* [12], *ARAN* [13], and *Secure AODV* [14] provide mechanisms to enable route establishment such that malicious nodes cannot cause inappropriate routes. However, these routing schemes only protect the control plane, i.e. the routing control messages, but do not secure the forwarding of data messages.

Some approaches exist, that secure both the routing and casual data traffic. *TESLA* [15] provides a mechanism of self-authenticating packets. The authenticity of packets is revealed only after a defined period of time which is unacceptable for instantaneous processing of received data. *Ariadne* [12] is a routing protocol based on *DSR*, which can use digital signatures, HMACs and *TESLA* for authenticating routing packets. Using *TESLA* results in a source routing algorithm with delayed authentication of the packets used in the route discovery phase. *TIK* [16] is a mechanism to detect wormhole attacks by verifying that the distance a packet has covered is tolerable. To measure the distance a packet has traveled geographical leashes or temporal leashes are applied. Both concepts have limitations, which results in very strict constraints on route discovery, end-to-end delay, and clock synchronization. *SRP* [17] is a routing protocol based on *DSR*, with minimalistic security features. *SRP* assumes pre-established secret keys between every pair of nodes in the network. With these keys, HMACs to the routing messages are computed. *SEAD* [18] uses hash chains to guarantee that the distance values broadcasted in route update messages are not decreased invalidly.

To detect misbehavior a number of approaches exist. Marti *et al*. introduced in 2000 the idea of *Watchdog* and *Pathrater* [6] in order to solve the problem of malicious nodes. The main idea of the mechanism is to store an identifier for every packet forwarded to a neighbor and, by

overhearing, validate whether the neighbor forwards this packet or not. Promiscuous listening on the wireless channel comes with several limitations. Scenarios exist, where collisions occur on the physical layer and the behavior of neighboring nodes cannot be verified. Also, efficient bandwidth utilization might rely on planning of simultaneous transmissions within a two-hop neighborhood, which also prohibits reliable promiscuous listening. Despite these limitations, *Watchdog* is still one of the most common building blocks in various proposed security frameworks.

Alternatives to *Watchdog* are limited. *Nuglets* [19] and *Sprite* [20] are both incentive-based solutions that are based on accounting of the forwarding service. In *Nuglets*, accounting is done locally, but requires tamper-proof hardware, while in *Sprite* receipts for delivered packets are saved and a trusted server is used for accounting these receipts. This results in high computational and storage requirements. Kargl presents in Reference [21] *Iterative Probing*, a mechanism to detect blackholes on routes. Kargl assumes a field in each packet that contains encrypted information only decipherable by a single node on the route. This node has to acknowledge the receipt of the packet. Starting from the destination node, the source node iteratively addresses every single node on the route. Malicious nodes that drop messages can, thus, be identified by means of the probing mechanism: the last acknowledgment is either from the malicious node in the route or from its precursor. However, Kargl's approach is limited to source routing and necessitates changes on the network layer. In Reference [22] Djenouri *et al*. propose the usage of signed two-hop acknowledgments. This results in high traffic overhead even after optimization.

In References [23] and [24] Buchegger *et al*. analyze the effects of selfish behavior in MANETs. They conclude that selfish behavior will spread and dominate in a system without punishing mechanisms, this endangers the operability of the network. Therefore, they propose the *CONFIDANT* system [25], which extends the *Watchdog* and *Pathrater* concept by distributing and using second hand reputation as well. Using false second hand reputations as an attack is not discussed. *OCEAN* [26] follows the idea of *Watchdog* and *Pathrater* and defines an extension on how malicious nodes can be punished. However, *OCEAN* cannot differentiate between malicious behavior and punishment for that. Thus, two neighboring nodes may get into a deadlock by mutually punishing each other for being punished. Michiardi *et al*. propose in Reference [27] a observation and reputation system called collaborative reputation mechanism (*CORE*). *CORE* has two basic components, a watchdog mechanisms to detect misbehavior and a reputation table maintaining the various types of reputation of all known nodes. However, *CORE* assumes global reputation values for the nodes, which is not realistic with varying malicious behavior toward different neighbors. Zhang *et al*. propose in Reference [28] an intrusion detection system that detects anomalies or violations of rules in networks. The components of the mobile intrusion system can be divided in local observation, evaluation and reaction and so called global counterparts, which take evaluation results of neighboring nodes into account. The detection criteria given by the authors in Reference [29] are very complex and need training data for the intrusion detection system in order to learn valid behavior.

Current state-of-the-art wireless technology poses strong constraints that have not sufficiently been considered in related work. In particular, hop-by-hop encryption on link layer prohibits overhearing of forwarded messages. Moreover, there exist reservation-based MAC layers, which might only possess limited broadcast-capabilities, thus, making the use of flooding-based reactive routing protocols prohibitively expensive. We identified that stigmergy-based routing algorithms fulfill most of the requirements of current and upcoming WMNs. Stigmergy-based routing protocols imitate the routing behavior of insects such as ants, which randomly explore the landscape until they discover food resources. To inform their colony, they return on the path traveled toward the food and lay a pheromone trail that attracts further ants to use this route. Routing algorithms based on this principle were introduced in Reference [30] by Di Caro *et al*. The advantage of this class of routing algorithms is that they do not demand broadcasting capabilities from the underlying MAC layer and routing decisions are made entirely localized, only based on probabilistic ratings of individual links.

Reviewing stigmergy-based routing algorithms we can identify the advantages of probabilistic routing algorithms based on local decisions. In contrast to centralized or source routing algorithms, route errors can be bypassed through the usage of alternative routes or by locally launching route discovery mechanisms. Probabilistic routing adapts the routes to changes in network topology and speeds up the integration of new nodes. Also, local observation of e.g. misbehavior can be taken instantly into account.

Ant Based Control [31] is a routing protocol which was designed for stigmergy-based routing in telephone networks. Thus, it is optimized for static networks in which congestion may appear but the set of participating nodes is fixed. *ARA* [32] is a reactive routing algorithm for mobile, multi-hop *ad hoc* networks. Its main design goal is to reduce the overhead for routing. We follow a proactive approach, thus routes can be instantly used. *PERA* [33] is a proactive routing algorithm designed for mobile *ad hoc* networks with broadcasting capabilities. We follow a proactive approach, thus routes can be instantly used. *Termite* [34] is a more recent approach, which uses regular data packets for route maintenance. Each data packet leaves pheromones for its source node while being routed to the destination node, using the pheromone based routing tables in each node. For security reasons it is more complex to counteract the misuse of all data packets than the misuse of special routing messages. We follow a proactive approach, thus routes can be instantly used. *AntHocNet* [35] and *HOPNET* [36] are hybrid routing algorithms combining reactive route setup with proactive route probing and exploration. This hybrid approach is taken to combine the best of both worlds. This idea inspired us to introduce two types of Forward

Ants: Discovery FANTs and Maintenance FANTs. With this approach link quality aspects and security considerations can be considered in parallel. For further details on benefits of stigmergy-based routing protocols, see References [32] and [31]. Security aspects for this class of algorithms are not discussed in sufficient detail, yet. In Reference [37] Hung and Evans have explored some of the security issues for stigmergic systems in general. In *AntTrust* [38] a secured ant routing protocol is presented which assumes broadcast functionality in the network, however.

In summary, we witness a lack of feasible security solutions to detect node misbehavior within WMNs. This is especially true, if we consider state-of-the-art wireless technology. We propose a security framework to deal with the aforementioned challenges. In particular, we propose AntSec, a secure stigmergy-based routing protocol. AntSec works in close collaboration with WatchAnt, which provides a mechanism similar to *Watchdog*. However, it can also cope with encrypted links on the MAC layer. Moreover, it is designed to synergistically exploit the characteristics of the underlying stigmergy-based routing protocol. For detection of colluding malicious nodes, we propose LeakDetector and as a connector between the three schemes we further introduce a reputation management system, AntRep.

4. AUTHENTICATED NETWORK ENTRY

In the following sections we present our security framework for wireless mesh networks. We first introduce necessary steps for secure network entry and for establishing a cryptographic key infrastructure. The security mechanism presented in the following sections operate on this key infrastructure.

As stated in the introduction, we have identified open subscription-based and open subscription-less based network. We focus on open subscription-based mesh networks, assuming the existence of a trusted third party (the network provider). Each node which wants to gain access to the network has to be authorized out-of-band, so that each user is identified and misbehaving can result in consequences for the user. We assume that devices are equipped with a certificate (IdentCert) issued by their manufacturer containing their Public Key and MAC address. Devices are registered at the network provider, which thus knows all valid device certificates. The credentials of the network provider are also stored on the device, e.g. by means of a SIM card. Besides the device certificates, we also use (daily) authorization certificates which are handed out during network entry to valid users.

During the network entry phase a new node has both to join and also to authenticate itself, the network and its new neighboring nodes. In order to join the WMN a new node attaches to nodes which are already part of the network. A new node we call Candidate Node (CN), the particular contact node we call Sponsor Node (SN). The SN acts as intermediary between the CN and the Authorization Node (AN) of the network provider. At the end of the joining phase, all nodes included in the procedure are authenticated and the CN becomes a full member, receiving a certificate of registration (RegCert). The RegCert contains the network Node-ID, an IP address, the MAC address of the node, and a hash of the node's Public Key, it is issued by the Authorization Node.

4.1. Process of network entry

In the first step, both nodes need to mutually authenticate themselves. In a second step, the new node establishes contact to its neighboring nodes and they authenticate themselves.

The SN sends its RegCert, proving the authorization to participate in the network and the validity of the authorization in this current time period. All communication during the network phase is timestamped and signed, in order to provide integrity and authentication and to block man-in-the-middle attacks. The CN sends its IdentCert and a signed timestamp to prove its identity. In the case that the SN cannot validate the IdentCert of the CN directly it can download the manufacturer's Public Key of CN from AN *via* the network and validate the IdentCert of the CN subsequently. After this the SN and the CN are mutually authenticated.

In the next steps, the CN performs its authentication. It requests and receives its RegCert of authorization, which indicates that it is authorized to participate in the network for a limited time period (e.g. 1 day). The Registration Certificate (RegCert) contains the CN's MAC, its newly assigned IP, Node ID, (Hash of PubKey) and a validity period. This certificate is small and can be used by the CN to prove its authorization to other nodes in the network.

In summary the network entry phase is secured using digital signatures against man-in-the-middle attacks. Timestamps are used to prevent replay attacks. Each participating node is authenticated. Based on this, the subsequent protocol steps can rely on these properties.

4.2. Neighborhood establishment

In the last section the Candidate Node has been authorized to join the network, using mutual authentication with the Sponsor Node and the Authentication Node. At this point of the protocol, only these three nodes know whether CN is authorized or not. In the next step, we propose mutual authentication for the direct neighbors of the node.

To implement a mutual authentication mechanism, we have to accomplish two different requirements: First, mutual authentication of the adjacent nodes. Second, proving the neighbor relation to authenticated nodes. This second step allows us to check whether the given node is in proximity to the checking node and the messages are not tunneled by a malicious node.

Without addressing the second point the following attack would be feasible:

$$Nbr(A) \leftrightarrow A \leftrightarrow bC \iff Ca \leftrightarrow B \leftrightarrow Nbr(B)$$

The nodes bC and Ca are both belonging to an attacker C, bC echoes each packet B transmits and Ca echoes every packet A transmits. The tunnel is only opened for communication between A and B, bC and Ca refuse communication to other nodes pretending disinterest. Thus a man-in-the-middle attack involving A and B can be carried out by node C. Relying on our first requirement only, the mutual authentication of the adjacent nodes would not detect the given case. We suggest as solution to prove the neighbor relation to a common neighbor by signing a list of Node IDs of adjacent nodes.

We present a four-way handshake for our mutual authentication protocol between neighbors. The protocol considers the neighborhood of the nodes and allows the verification of the neighborhoods. Additionally, a symmetric key is exchanged using Diffie–Hellman [39] for further usage.

(1) $A \rightarrow B : RegCert_A,\ IdentCert_A$
(2) $B \leftarrow A : RegCert_B,\ IdentCert_B,$
$((DH_B, NonceB, ListOfNbrs)SigB\ )encPub_A$
(3) $A \rightarrow B : ((NonceB + 1, (NonceA)encDH_{Key},$
$DH_A,\ ListOfNbrs)SigA\ )encPub_B$
(4) $B \leftarrow A : (NonceA + 1)encDH_{Key}$

We next discuss the four-way handshake.

(1) The first message is informative, it shows which identity (node A) shall be authenticated (by node B). The receiving node B verifies the validity of the information in the Registration Certificate and Identity Certificate of node A.
(2) B sends its certificates in return and additionally a message encrypted with A's Public Key containing B's part of a new Diffie–Hellman Key, a number-used-once (Nonce) and a signed list of the Node IDs of B's neighbors. Sending a list of the neighbors prevents attackers spoofing B in another location. The signature proves B to be the creator of this message.
(3) A's identity is characterized by the possession of the Private Key belonging to the Public Key identified in the certificates of A. A proves its identity by decrypting NonceB and sending it back. Now, A affirms its neighbor relations by sending a signed list of the Node IDs of its neighbors. A sends also its part of the Diffie–Hellman Key which allows to calculates the complete Diffie–Hellman Key of A and B. NonceA is encrypted using the new Diffie–Hellman Key and the whole message is encrypted using the Public Key of B.
(4) The last message in the four-way handshake proves B's possession of the Private Key corresponding to the Public Key in B's certificates, thus the identity of B. NonceA+1 can only be calculated if B decrypted NonceA using the shared Diffie–Hellman Key of A and B.

Therefore, after performing this four-way-handshake both nodes are mutually authenticated and a symmetric key is exchanged for further usage. This mutual authentication is done during link establishment with a selection of nodes from the direct neighborhood, so our new node is now fully integrated into the network and ready to participate in routing.

5. ANTSEC, WATCHANT, AND ANTREP

In this section we first outline the components of our security framework and give an overview of the interactions among the different components. This is followed by a detailed description of the working of the individual components.

Figure 1 shows the components of our security framework and the interaction among the different components. AntSec is a stigmergy-based routing algorithm which builds up on AntNet 1.1 [40] and provides several security extensions. WatchAnt is a challenge–response based misbehavior detection mechanism, which is inspired by the work of Marti *et al.* [6]. Unlike contemporary watchdog mechanisms found in literature, WatchAnt is able to detect misbehavior of neighboring nodes even in the presence of encrypted wireless links. AntRep, the reputation management system that complements AntSec and WatchAnt, uses a multiple-threshold based system to classify neighboring nodes into different categories based on their (mis)behavior observed by WatchAnt. Figure 1 shows the interaction among the above components. AntSec is responsible for updating the routing tables at the node and for acquisition and maintenance of the routes. The routing table contains entries per-destination and neighbor; these values denote the probability that the respective neighbor is selected as next hop for the particular destination. AntSec additionally uses information from AntRep to adapt these probabilities. WatchAnt observes the routing (control) packets sent by AntSec as well as data transmissions sent to neighboring nodes and uses the routing information to perform checks to detect misbehaving neighbors. The misbehavior information is then fed to AntRep, the reputation management system as shown in Figure 1. We will next discuss the functioning of the individual components.

5.1. AntSec: securing ant routing

AntSec is a proactive, probabilistic, multipath, stigmergy-based, distributed, non-broadcast based, secure routing algorithm. AntSec is one of the first contributions for securing stigmergy-based routing algorithms (see the work of Zhong and Evans [37] for earlier work to study security vulnerabilities of stigmergic systems). Our work has to be

seen in this context. Here, we present the vital components of the routing algorithm.

Using AntSec each node maintains a routing table where for all tuples (*Destination Node*, *Neighboring Node*) a routing probability is maintained denoting the probability of choosing the *Neighboring Node* as the next hop for a packet destined to the *Destination Node*.

AntSec uses the following routing messages:

Discovery Forward Ant (DFANT): DFANTs are periodically sent to random destinations to find and establish new routes. DFANTs contain a registration certificate, and a public key hash authenticating the source node. The registration certificate and keys are obtained by nodes from a trusted third-party (network provider) as stated in our assumptions. In addition DFANTs have a path list containing all visited nodes. Flags in the DFANT allow the nodes on the route to request the registration certificate and public key of the destination node.

Maintenance Forward Ant (MFANT): MFANTs are sent periodically to keep the current routes active, reinforce active routes, and adapt the routing probabilities to the current state of the network. Similar to DFANTs, MFANTs contain a list of visited nodes. However, as all the nodes on the route have received the registration certificate of the source and the destination during route setup from the DFANTs, MFANTs contain only a unique hash of the source registration certificate.

Backward Ant (BANT): BANTs are sent by the destination nodes in response to received forward ants (DFANTs and MFANTs). Critical parts of the forward ants are signed and added to the corresponding BANT. Additionally, the public key and registration certificate of the destination node is added to the BANT if requested in the forward ants. BANTs also contain a complete list of visited nodes. All types of ants have an AntID, which uniquely identifies the ant. Due to the proactive nature of AntSec, routes are established before they are used. As only authenticated and authorized nodes shall participate, their registration certificate is contained in every DFANT.

Only authorized nodes are allowed to participate in the network. Therefore, upon receiving a DFANT, the registration certificate of the source node is checked for validity and stored if valid. In MFANTs the registration certificate is not contained in order to save bandwidth. During route establishment the certificate is propagated once and stored by every node on the route. The same holds for the Public Key and registration certificate of the destination node. Once a route is established, the validity of the destination node is checked and the Public Key of the destination node is distributed, there is no further need to provide this information in MFANTs. Invalid certificates and Public Keys can be detected easily, as the certificates may contain a hash of the Public Key and the information about the current network identifier of the corresponding node. With this approach only authenticated and authorized nodes can establish routes.

In order to guarantee the integrity of routing messages a two-step mechanism is used. BANTs are signed by the destination node, so that every node on the way back to the source node (using the path history) can verify the integrity of the BANT. To achieve this, the Public Key of the destination node is provided in the BANT upon request. The integrity of FANTs is guaranteed by comparison of the invariant fields of the bypassing FANT during the route discovery and the bypassing BANT at the route establishment. Before forwarding a FANT, each node stores a hash of the critical fields of the FANT, i.e. the path history up to the current node, the certificates, source and destination identifiers. Each node, upon receipt of a BANT, checks the BANT's path history up to this node's occurrence and other immutable parts of the routing message, to see whether the corresponding FANT is known and has not been tampered with. Any invalid modification of the FANT results in a BANT which cannot be associated to its corresponding FANT. In case of a valid match, the integrity of the FANT is assured. The matching entry is then deleted from the stored memory. In case a corresponding FANT cannot be found, the integrity of the original FANT must have been compromised or a new FANT has been forged, in this case the BANT is dropped. For any unexpected error the previous hop (node) is punished with a reputation decrease. In case of a valid BANT the routing tables are updated.

We see that forging and invalid modification of routing messages have no effect. Even replays of old FANTs and BANTs do not cause harm, as replayed BANTs are dropped due to the missing corresponding FANT. Replayed FANTs have the effect of new FANTs, so benefit is gained from this attack (by helping to update routing tables). Only routing information taken from the BANT is used to update the routing tables, when the integrity and authenticity of the routing messages is assured.

Malicious nodes may try to cause inefficient routes or even loops, but effects of such attacks are limited, because each node can only determine the next hop of the routing packet.

One additional advantage of the stigmergy-based security approach is that attacks have to be performed several times to have effect. Routing probabilities change only significantly after several routing table updates. With increase in number of attacks, malicious behavior is easier to detect.

5.2. WatchAnt: watching the extended neighborhood

In this subsection we present details about WatchAnt, a novel misbehavior detection mechanism for WMNs. To the best of our knowledge, WatchAnt is one of the first schemes proposed which can detect node misbehavior even in the presence of per-link encryption at the MAC layer. Detection of misbehavior in forwarding data by a neighboring node is a difficult task in WMNs. Even when the wireless links are not encrypted, parallel transmissions scheduled within a two-hop neighborhood to increase spatial reuse (as done by the TDMA based MeSH mode of the 802.16 standard) make promiscuous listening difficult, if not impossible.

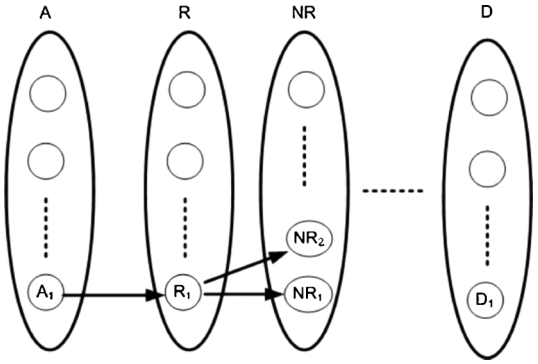


Figure 3. WatchAnt working principle.

WatchAnt is a challenge–response based scheme for detecting forwarding misbehavior of neighboring nodes. To make the presentation intuitive we will explain the functioning of WatchAnt with the help of the schema shown in Figure 3.

Consider the sets of nodes *A*, *R*, *NR*, and *D* as shown in Figure 3. *A* will represent the set of nodes generating the packet or forwarding the packet and wanting to verify the forwarding behavior of the next hop for the packets. *R* denotes the set of relay nodes (next hops for packets transmitted by nodes in set *A*). *NR* denotes the set of next hops for the set of nodes in *R*; i.e. packets transmitted by nodes in set *A* to nodes in set *R* will be forwarded to nodes in the set *NR* on their way toward the destination (nodes in set *D*). The set *D* is assumed to be the set of destinations for the packets originated or initially transmitted/forwarded by the nodes in set *A*. Note that, just to keep the discussion simple, we assume that the sets are disjoint, they could in practice be non-disjoint. We use *P* to denote the set of packets originated or initially forwarded by nodes in set *A*. P_x^{xy} denotes the last packet transmitted by a node *x* to node *y*. P_{x-i}^{xy} thus, denotes the *i*th packet preceding the current packet transmitted by the node *x* to node *y*. We assume the presence of a mathematical function $hashsum(h1, h2)$ which is basically a mapping $(\{H\},\{H\}) \longmapsto H$, where $H = B^{160}$ $h_i \in H$ for $i \in N$ and $B = \{0, 1\}$. The set *H* can be considered to be a set of 160 bit hashes computed for individual packets using a hash function. Assume that the function $hashsum()$ is commutative as well as associative. If $hashsum()$ is in addition a one-way hash function it is beneficial for our mechanism, however, this is an optional feature. To simplify the notation we will use $hashsum(P_i, P_j, \ldots, P_k)$ to denote the repeated application of *hashsum*, $hashsum(h_i, hashsum(hashsum(P_j, \ldots hashsum(P_{k-1}, P_k))))$, where h_i corresponds to the hash value for packet P_i.

We next give an overview of the challenge–response mechanism. The node (A_1) wishing to verify the forwarding behavior of its neighbor (R_1) transmits a challenge (WatchAnt Request [challenged node's ID (R_1), packet count (i)]). The challenge identifies the addressed node, and requests it to reply with the forwarding information for the last *i* packets sent by the challenger to the challenged node. In this example, R_1 is requested to reply specifying information about the forwarding for the packets $P_{x-i}^{A_1R_1} \ldots P_x^{A_1R_1}$.

R_1 then sends a response (WatchAnt Reply [(previous hop (A_1), next hop node ID ($\in NR$), num. of packets forwarded (j), *hashsum* for the j packets]). The response consists of a set of tuples identifying in each case the previous hop (the challenger), the next hop for a set of packets, the number of packets forwarded to the next hop, the hashsum for all the packets sent to the next hop. To make the example more clear, assume that A_1 had sent a challenge as specified above. Further assume that the node R_1 has forwarded the packets only to a single next hop (NR_1) and the challenge had asked for the last two packets. The response then looks like $[A_1, NR_1, 2, hashsum(P_x^{A_1R_1}, P_{x-1}^{A_1R_1})]$. This response is transmitted as a broadcast message without encryption. We denote the number of previous packets for which the $hashsum()$ is to be computed as *WaReqNum*. In the above example, $WaReqNum = 2$. This parameter determines the probability of detecting forwarding misbehavior of neighboring nodes.

Thus, in our scenario, when R_1 transmits the response, it will be received by its neighbors in the sets *A* and *NR*. Nodes which are addressed in the response will process the reply. In the above example, the nodes A_1 and NR_1 will process the received reply. By verifying the reply, either A_1 or NR_1 or both will be able to detect forwarding misbehavior of node R_1 in case it is misbehaving. In general, we can say the the WatchAnt reply sent by the relays (set of nodes *R*) will be verified by the challengers (set of nodes *A*) and the reported next hops after the relays, i.e. the set of nodes *NR*. Both these sets of nodes need to be able to verify the reply, the set of nodes *R* needs to generate a reply. Hence, each of the above sets of nodes needs to maintain certain data structures, which are described next.

Each node maintains two lists *InList* (for information about packets received from the neighbor), and *OutList* (for storing information about packets sent to the neighbor) for each neighboring node. Let $hashIm(P)$ denote the hash of the immutable parts of packet *P*. An entry in the *InList*, for a packet received (*P*), contains $hashIm(P)$, the node identifier of the neighbor which transmitted this packet (*previous node*), and the node identifier of the node which transmitted the packet prior to the *previous node*. In addition, a field in the *InList* can be used to enter the information about the next hop for the packet.

The *OutList* contains for the packet transmitted (*P*), $hashIm(P)$, and the node identifier of the node to which the packet was forwarded. The number of entries in the above two lists can be limited to some maximum value. In addition, to be able to verify the WatchAnt reply, we need information about the previous two hops for the packets. Therefore, for each transmitted packet, a previous node identifier field is set in addition to the transmitter's node identifier. On receiving or transmitting a packet, the *InList* or the *OutList* are updated and all the fields in these lists are set as specified previously. A node periodically issues WatchAnt requests (challenges) as explained previously asking for the infor-

mation about the previous *WaReqNum* packets sent to the neighbor. The challenged node then uses the *InList* to find out the next hops for the last *WaReqNum* packets received from the challenger. Using the hash values for the packets in question found from the *InList* and the corresponding next-hops, the challenged node uses the *hashsum* function to generate the WatchAnt reply. The WatchAnt reply is then transmitted. The challenger uses its *OutLists* to determine whether the *hashsum* reported by the challenged node matches the *hashsum* for the packets sent to the node. Other nodes (corresponding to the set *NR*) receiving the WatchAnt reply and addressed in the reply use their *InList* to check if the node really forwarded the packets it reports as forwarded. It is seen that a malicious node which lies and tries to manipulate the reply can fool only the challenger or the next hop but not both simultaneously and, hence, its forwarding misbehavior will be detected.

The parameter *WaReqNum* plays an important role in determining the ability of the WatchAnt mechanism to detect forwarding misbehavior. Consider that a malicious node drops packets with a probability of P_{Drop} instead of forwarding them. Now the probability that a packet is not dropped is given by $(1 - P_{Drop})$. Given a WatchAnt request asking for information about the last *WaReqNum* packets is addressed to a malicious node, its malicious behavior will not be detected if and only if it has not dropped a single packet in the last *WaReqNum* packets. The probability that the node has not dropped a single packet in the last *WaReqNum* packets is given by $(1 - P_{Drop})^{WaReqNum}$, which is equal to the probability that a malicious node will go undetected. Thus, the probability that a malicious node, dropping packets with a probability P_{Drop}, will be detected is $1 - (1 - P_{Drop})^{WaReqNum}$.

5.3. AntRep: managing reputation for stigmergic systems

To maintain a current state of its neighborhood, each node relies on AntRep as a reputation management system. For our scenario, reputation management is carried out in a distributed and decentralized fashion. In particular, AntRep represents all information gathered by WatchAnt about the well-behavior of the extended neighborhood of a node as node—value pairs. These reputation values are updated periodically when positive or negative observations are made by WatchAnt. We next describe the characterization of the reputation values, the system policies to react if certain thresholds are reached and the detailed process of updating reputations. We summarize the subsection by highlighting similarities and differences with related work in the area of reputation management.

If a node joins the network, its reputation value is 0. We define the following thresholds to apply for our reputation system.

- 25 Maximum reputation value
- 0 Initial reputation value
- −25 No-reputation-bonus threshold
- −40 Punishment threshold
- −60 Minimum reputation value

The symmetric range from [−25;25] describes the notion of normal operation of neighbors. The nodes' strategy is to perform normal routing operations to neighbors within this reputation range. Also, positive as well as negative observations lead to the below detailed change of the reputation value if within this range. Reputations below −25 indicate that a neighbor behaves maliciously. Reputation changes toward a better reputation value are no longer commenced as feedback to WatchAnt, but the reputation is only allowed to increase according to the restoration process described below. As soon as the reputation falls under −40, two changes take effect. (1) The node is excluded from routing, i.e. its probability of being chosen as next hop to arbitrary destinations is set to the minimal value. (2) The node is denied service, i.e. messages generated by this node are no longer processed.

We distinguish between the thresholds at −25 and −40 to be able to adequately treat selfish behavior of nodes, which might try to constantly operate with a bad reputation to avoid forwarding of packets for other nodes. From the threshold −25 on, these nodes rely on the (slow) mechanism of reputation fading to get back into normal operation, they are living on the edge of exclusion. In contrast, inactivity of a node is not considered harmful. To enforce continuous positive behavior from benign nodes and to allow nodes identified as malicious to (slowly) recover their reputation, the reputation of a node is periodically updated as follows:

- If current reputation value *oldRep* is positive: $newRep = 0.9 \cdot oldRep$
- If current reputation value *oldRep* is between −40 and 0: $newRep = 0.98 \cdot oldRep$
- If current reputation value *oldRep* is smaller than −40: $newRep = 0.99 \cdot oldRep$

The above models the reputation fading (or second chance) mechanism in our misbehavior detection systems. Misbehaving nodes can return to normal service after an appropriate timeout. Thus, without any other triggering change, the reputations of all neighboring nodes converge to the initial reputation value (0). As seen in this subsection, reputations are maintained locally, representing the subjective view of one node observing its neighbors. The reputation value of a single node maintained by two different neighbors can be completely different (also this single node can behave differently with respect to its neighboring nodes). Each node decides based on its local reputation table, how to cope with each of its neighboring nodes. Mechanisms can be devised to use the local subjective observations and spread them as second-hand reputations (see Buchegger [41]). Second-hand reputations have been shown to increase the speed of detection of malicious nodes. For the results presented in this paper, we do not employ second-hand reputations, but rely only on

local-observations and decisions to minimize the protocol overhead.

In addition to positive or negative reputation updates based on the WatchAnt replies, we identified elements of the AntSec protocol which can be used to update the reputation of neighboring nodes. Symptoms for node misbehavior that can be detected by a benign nodes are, for example, routing loops in FANTs, invalid BANT entries and BANTs received for which no corresponding FANT has been seen. Benign nodes receiving invalid protocol packets will not forward them. Thus, when an invalid packet is received from a neighbor, the neighbor is punished by decrementing its reputation value. There are only two events which trigger a rise in the reputation. (1) The referenced outgoing node receives a WatchAnt reply with correct information. (2) The node starting the examination process (WatchAnt requester) receives a WatchAnt reply containing authenticated nodes and correct information about the forwarded packets. In this case the reputation rises, too.

In summary, AntRep provides a localized view of the reputation of the nodes in our stigmergy-based routing system. The necessary reputation input is provided by AntSec and WatchAnt feedback.

6. LEAKDETECTOR: SOLUTION TO THE COLLUDING MISBEHAVIOR PROBLEM

Colluding misbehaving nodes are a severe threat to the correct routing functionality in MANETs and WMNs. Before presenting *LeakDetector*, our solution for detecting colluding misbehaving nodes without the use of cryptography, we discuss the assumptions we made while designing the solution.

6.1. Assumptions

The detection of misbehaving nodes depends on the underlying routing algorithm. For our scheme, we assume the following characteristics for this routing algorithm.

(1) *Distributed and Unicast*: Each node autonomously calculates the next hop node; for each individual packet a single next hop neighbor is chosen.
(2) *Proactive*: The routing mechanism periodically refreshes the routing information.
(3) *Secure Route Information*: Message integrity and authenticity for routing messages is guaranteed; routing messages contain the information for the entire routing path.
(4) *Multipath Routing*: Various paths from source to destination exist; *LeakDetector* compares these paths in order to identify malicious nodes.
(5) *Single-hop Monitoring*: A watchdog (or similar) mechanism is in use for detecting routing misbehavior in the one-hop neighborhood.

6.2. Leak detection mechanism: protocol

The main idea of *LeakDetector* is that the destination node of a route builds up a virtual graph, which models the multipath from the source node to the destination node. Periodic traffic information (which can be piggybacked on the proactive routing messages) enables the destination node to calculate the ratio of incoming and outgoing traffic—corresponding to the multipath routing information—for each participating node. Using graph theory, traffic leaks are identified. In particular, the destination node compares per route the incoming ratio with the outgoing ratio for each node participating. When the deviation is too large, the node is assumed to be malicious. The description of the leak detection mechanism and the actions and behavior of the individual nodes is as follows.

6.2.1. Source node.

Each source node maintains a traffic counter per route (source-destination combination) denoting the amount of traffic (in bytes), which has been sent to the destination node.

We assume that the periodic proactive routing messages provide two fields, which are relevant for this task: T_{total} is used to describe the total traffic for this route (2 bytes); for each visited node i, T_i denotes the fraction of traffic that passed the node (1 byte per node) in comparison to the total traffic sent by the source node.

6.2.2. Intermediate node.

On its way from the source node S to the destination node D, the routing messages are forwarded by the intermediate nodes N_i. Let's assume the packet is forwarded from node N_1 to node N_2. Then N_2 performs the following steps: N_2 appends its own information to the visited node list, where the T_{total} field is already set. N_2 calculates the amount of traffic received from its precursor N_1 for the route $S \rightarrow D$. This amount of traffic is set in relation to the total traffic for this route (denoted in the T_{total} field of the routing message). The relation represents the fraction of traffic for this route sent from N_1 to N_2. N_2 sets the respective value in the T_{N1} field of the visited node entry. With the given parametrization of one byte for the T_{N1} field, we obtain a resolution of $100/255 = 0.4$ for the obtained fraction.

6.2.3. Destination node.

The destination node collects the traffic information from incoming routing messages and creates a virtual graph. Each vertex represents a node participating in a route from S to D. The directed edges between two nodes N_1 and N_2 represent the fraction of traffic that travels *via* $N_1 \rightarrow N_2$ on its path from S to D. The destination can also infer the amount of traffic sent from N_1 to N_2 corresponding to this route.

If D recognizes that the number of bytes received differs significantly from the number of bytes originated by

the source, the *LeakDetector* enables the detection of the malicious node. The graph is further maintained and the amount of incoming traffic and outgoing traffic is updated with every incoming routing message for the corresponding nodes. If the values of a specific node X_1 differ significantly due to the outgoing traffic being far less than the incoming data, the destination node D assumes that X_1 is malicious.

6.2.4. Detection criteria.

We assume the source S and the destination D to be non-malicious. Moreover, if less than 50 packets have been processed to a given route, our scheme is not active. If the *inflow* of the node is smaller than 5% of total traffic or the difference of the *inflow* and the *outflow* of the node is smaller than 5% of total traffic, detection is deferred.

If a node does *not* fit in the latter two categories, the node is considered malicious if:

- $in_{node} > \alpha \cdot out_{node}$, with α being a tuning parameter for the *LeakDetector*.

If none of the aforementioned cases is applicable, a node is considered benign.

6.2.5. Maintenance of counter and reconciliation of false detections.

Periodic initialization of the traffic counter (e.g. every 10 min) is necessary to allow the detection of nodes that switch to malicious behavior, but have previously cooperated. With a long-term history only, the system would slowly react to such nodes. Resetting the counter should be loosely synchronized; in a time window of 30 s each node resets its internal traffic counter for the current route to 0. The destination node D of the route rebuilds the virtual graph.

6.2.6. Reaction to malicious nodes.

Once the destination node detects a node en-route as malicious, various strategies can be applied. For example, the destination node may propagate this information to the source node, using a proactive route reply that uses a disjoint path. The source node could maintain a blacklist of nodes to avoid for routing/forwarding purposes. Also, the destination node can affect the route establishment and maintenance directly by marking or dropping routing messages that list malicious nodes in their path history. Another strategy would be to maintain reputation information in a distributed manner and to use this information to decide which paths to choose for a route and/or which nodes to punish.

6.3. Example

We give an example of a virtual graph that may be observed at node D in Figure 4. A proactive routing message using the path $S \to N_1 \to N_2 \to N_3 \to N_6 \to N_9 \to D$ carries the information on the fraction of traffic corresponding to this route received from the previous hop: $(T_{\text{total}})S \to (0.45)N_1 \to (0.45)N_2 \to (0.4)N_3 \to (0.1)N_6 \to (0.2)N_9 \to (0.4)D$.

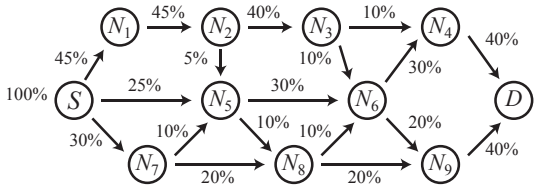


Figure 4. Sample virtual graph built by the destination node D.

Table I. Comparison of the in and out traffic per node.

	$N_{1,2}$	N_3	$N_{4,5}$	N_6	$N_{7,8}$	N_9	D
In	45%	40%	40%	50%	30%	40%	80%
Out	45%	20%	40%	50%	30%	40%	

The destination node obtains a clear picture of the relation between inflow and outflow per node on the route by periodically updating the virtual graph. Table I shows one possible example. D can observe that the deviance in node N_3 is obvious. Depending on the parameter α, node N_3 is going to be considered as malicious.

7. EVALUATION

As a proof-of-concept, we implemented our framework in a simulation environment. We next present a performance evaluation of our security solution. To this end we performed a set of simulation studies for selected experimental designs, which have been chosen to scrutinize the individual mechanisms developed. The goals of the performed experiments are as follows.

(1) We analyze the performance of our secure probabilistic routing scheme AntSec in the setting of a realistic WMN. We particularly focus on the successfully delivered packets (delivery ratio) in the network under attack, which demonstrates the effects of attacks and countermeasures from an application level perspective.
(2) We investigate the performance of the local misbehavior detection scheme WatchAnt in combination with the reputation management AntRep. We chose a basic topology and model nodes that randomly drop packets with a given probability, which allows us to evaluate the detection quality of the system and its dynamics. We investigate the detection quality using the reputation values assigned to the misbehaving nodes. The overall performance is assessed by means of the obtained delivery ratios.
(3) We analyze the performance of the global misbehavior detection scheme LeakDetector. We again

Table II. Average packet delivery ratio of the secure routing scheme AntSec in Experiment 1 with standard deviation.

Routing algorithm	Fraction of malicious nodes					
	0%	10%	20%	30%	40%	50%
AntSec	97.9±1.1%	81.2±11.8%	65.4±6.2%	46.7±8.2%	35.3±7.8%	25.2±6.1%
Ant	97.3±0.8%	68.4±11.9%	48.7±8.4%	32.9±7.5%	27.0±7.0%	21.9±6.2%
DSR	100.0±0.0%	63.2±13.9%	44.4±12.7%	30.7±10.3%	23.2±8.4%	17.7±8.0%

investigate the measures of delivery ratio and detection quality, albeit for a scenario with colluding misbehaving nodes.

7.1. Simulation setup

We utilize a consolidated version of the JiST/SWANS [42] discrete event simulator. The extended simulator contains an implementation of the MeSH mode of IEEE 802.16 (MAC and PHY), which has been used for our study. The PHY was configured to comply with ETSI 3.5 MHz channel, OFDM 256 modulation, $n = 8/7$ oversampling factor.

For our study, we compare the developed AntSec [43,1] protocol with its insecure ancestor AntNet [44,40] as well as the Dynamic Source Routing (DSR) protocol that serves as a performance baseline. The protocols were modified such that they could operate on top of the reservation-based MAC protocol of IEEE 802.16. AntSec is a proactive, probabilistic, multipath, stigmergy-based, distributed, non-broadcasting, secure routing algorithm inspired by the routing behavior of ants. Despite its probabilistic nature of the protocol, we consider AntSec as a representative protocol for the class of probabilistic routing algorithms for WMNs. Additionally the developed schemes for local and global misbehavior detection (WatchAnt, LeakDetector) as well as the reputation management (AntRep) are analyzed.

We next present the individual experiments along with the results obtained. The duration of the individual tests allowed to reach the steady state of the network after the initial network entry procedure and the flow and route setup have been completed. We perform 20 replications for each experiment, the simulation time is 1000 s if not mentioned otherwise.

7.2. Evaluation of the secure routing scheme

In Experiment 1, we study the performance of the secure routing scheme using a randomly generated WMN topologies comprising 50 nodes. The nodes are stationary and the node degree is bounded to at most five neighbors. The workload has been chosen to be 25 Constant Bit Rate (CBR) flows between randomly selected (source, destination) node pairs, each flow generating a rate of 10 packets of size 512 byte each per second. We vary the fraction of malicious nodes that are randomly placed in the network from 0 to 50%. Goal is to observe the effects on the packet loss rate as well as the effectiveness of our scheme to identify misbehaving nodes. As described in Section 5.1 AntSec is by design immune to attacks on the routing (control) data itself. Hence, we model the malicious nodes to drop all data packets not related to themselves, but to process routing packets. We focus our study to investigate the influence of the routing scheme employed; in particular, we compare AntSec, AntNet (denoted as Ant) and DSR.

Table II shows the mean delivery ratio over a varying fraction of malicious nodes in the network. If malicious nodes are absent, all three routing protocols feature delivery ratios close to 100%.[§] For an increasing fraction of malicious nodes, AntSec outperforms the other routing protocols, which demonstrates the effectiveness of our security framework to detect and exclude malicious nodes from the network. Despite this fact, the routing performance for all protocols deteriorates with an excessive amount of malicious nodes in place. Due to its probabilistic nature, AntNet is inherently able to constantly adapt its routes, thus, partially avoiding malicious nodes. This results in a slightly better performance compared to DSR. We can conclude that the features of stigmergy-based protocols such as self-stabilization are not sufficient to counteract node misbehavior, but need to be complemented with security mechanisms to combat malicious nodes. AntSec lays the groundwork for further research in this area.

The improved performance comes at a price, however. The average routing overhead of AntNet is around 20% of the total number of bytes transported in the WMN (measured over all experiments). Adding the security mechanism in AntSec leads to an average control overhead of 35%. DSR serves as a baseline for the overhead: because of the static setup, routes are discovered only once. It's overhead was measured to be less than 2% for within Experiment 1, which accounts for flooding the network once for each route to be established. Though, optimization of AntNet and AntSec, e.g. using the adaptation of the emission rate of DFANT and MFANT messages, can lead to a significant reduction in overhead, which is outside the scope of this work.

Additionally we analyzed the average hop count observed in the network. Control packets observe longer paths when

[§] Please note that the stigmergy-based protocols show a small amount of packet loss if the time-to-live for packets is exceeded, which can happen due to the probabilistic nature of these schemes.

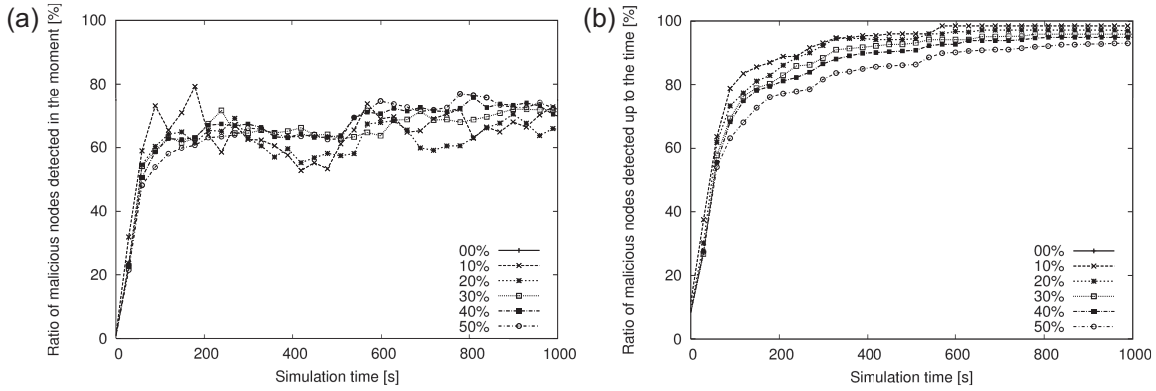


Figure 5. Quality of detection of malicious nodes over time in Experiment 1. (a) Fraction of malicious nodes identified at a given time. (b) Cumulative fraction of malicious nodes identified at least once until the given time.

using AntSec, because they are not forwarded to malicious nodes. Over the entire experiment set, the established paths followed by the data packets are in turn prolonged for AntSec with an average length of 4.53 hops, which is an increase of 1.21 hops over AntNet. This behavior is typical for multihop networks under attack, where it has been shown that in case of node misbehavior the probability of establishing longer routes is diminished; here AntSec counteracts this and is able to maintain longer routes.

7.3. Evaluation of the reputation management scheme and the local misbehavior detection

We perceive the detection quality of malicious nodes for our local misbehavior detection scheme to be of high interest. Figure 5(a) shows the fraction of identified malicious nodes at the given time. The network shows a stable detection quality of around 70% of all malicious nodes (i.e. in 70% of the cases at least one neighbor finds out a node is malicious). The remaining 30% account for nodes that are either false negatives (i.e. non-detected malicious nodes) or malicious nodes that have re-obtained an acceptable reputation over time. As described earlier, we consider such a second chance as vital to allow nodes to revert to positive behavior, thus also mitigating false positives (wrongly excluded well-behaving nodes). Figure 5(b) shows the cumulative number of correctly identified malicious nodes at least once up to a given point in time. It can be seen that nearly 100% of the malicious nodes are detected over time for all setups. The results shows that AntSec is able to detect malicious nodes even for very high fractions of misbehaving nodes and, as a result, to adapt and improve the routes in the network.

The benefit of the successive identification of malicious nodes by AntSec can be observed in Figure 6, which shows the mean delivery ratio accumulated until the given point in time for Experiment 1. The figure shows an increase in trend in the obtained delivery ratio over time for all tests. This implies that with increase in deployment time AntSec

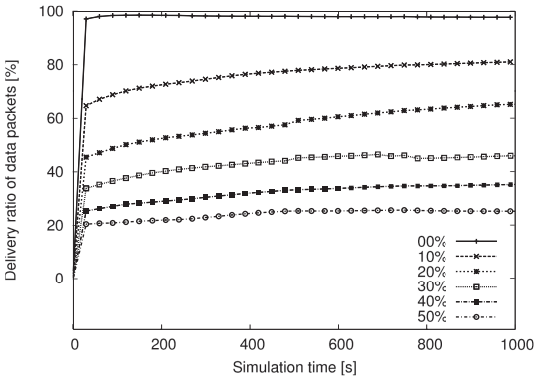


Figure 6. Mean fraction of data delivered cumulatively up to a given time Experiment 1.

selects improved routes (avoiding malicious nodes) and, thus, improves the delivery ratio. In fact, we observed that when one considers the delivery ratio for the last 100 s of the simulation only, AntSec shows delivery ratios that are up to 8 per cent higher than the average delivery ratio over the entire simulation duration of 1000 s.

We perform a second set of experiments to get insights into the reputation management system as well as the effectiveness of the local misbehavior detection. In contrast to Experiment 1, we do neither use a random topology nor random traffic flow end points. Instead the basic WMN topology shown in Figure 7 is employed in Experiment 2. We consider two CBR flows with a data rate of 10 packets of size 512 byte each per second. The two flows considered are $N_S \rightarrow N_D$, and $N_D \rightarrow N_S$. Node X marks the malicious

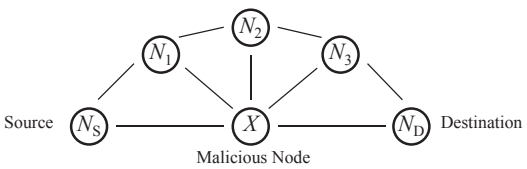


Figure 7. Topology for Experiment 2.

Table III. Average packet delivery ratio of the secure routing scheme AntSec in Experiment 2 with standard deviation.

Routing algorithm	Drop ratio of malicious nodes				
	20%	40%	60%	80%	100%
AntSec	92.2±3.3%	91.0±5.0%	87.8±7.6%	82.2±9.5%	82.5±8.6%

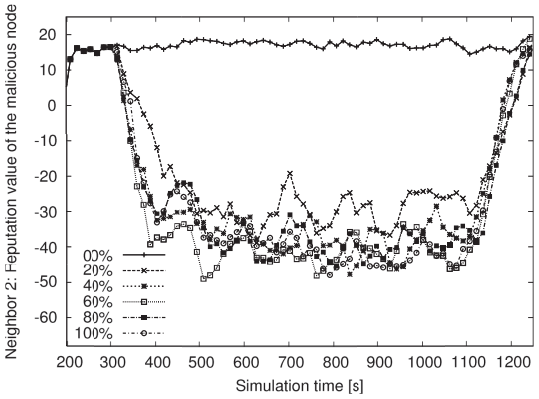


Figure 8. Reputation of the malicious node X as computed by node N_2 for Experiment 2.

node. It is active in the time-interval [200 s;1200 s] and acts maliciously in the interval [300 s;1100 s]. Thus, the node is malicious 80% of the time. We consider the performance of AntSec for malicious nodes that exhibit probabilistic misbehavior, i.e. these nodes do not drop 100% of the data packets, but try to avoid detection by randomly dropping only a certain fraction of packets. We vary the degree of maliciousness during the latter period of time.

Table III shows the mean fraction of data delivered to the destination using AntSec for differing drop rates of node X. It is seen that even with 100% packet drop rate of the malicious node AntSec achieves a mean delivery ratio of 82.5%. As a baseline, for the latter example AntNet routing produces a mean delivery ratio of only 21.9% (which is only marginally better than the sustained delivery ratio of 20% that can be reached solely during the non-malicious time interval).

Figure 8 shows the value for the reputation for the malicious node as computed by node N_2. The figure also illustrates the detection speed of WatchAnt. The reputation value of the malicious node maintained by each one of its five neighbors decreases rapidly after entering the time interval [300 s;1100 s], in which the malicious node drops packets. Having a very low reputation (i.e. -40 to -60) means that the detected node is not utilized in routing and no service of forwarding packet is provided for them by the detecting nodes. As they are not utilized they can obtain a better reputation again, are included into the network, and after a short while detected as malicious again. So the state of being detected fluctuates as shown in Figure 8. When the malicious node stops dropping packets (from 1100 s on) its reputation recovers. Hundred seconds are enough to gain a very good (i.e. $+20$) reputation again. For the case of

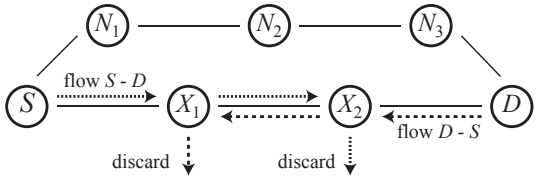


Figure 9. Topology for Experiment 3.

the malicious node acting malicious only on 20% of the processed packets the decrease in reputation is slower, still its reputation reaches -20 roughly 100 s after starting the attack. Starting from then the reputation fluctuates between -20 and -40, the node's exclusion from the network is impending.

We can conclude that the drop ratio of the malicious node makes essentially no difference to its detection probability in AntSec if it exceeds a certain threshold. For all studied drop rates, on average 4.3 and 4.55 neighbors detected node X to be malicious, i.e. almost all out of its five neighbors. This effect is due to the design of the WatchAnt mechanism as shown in the following example. As discussed earlier, the parameter *WaReqNum* is used during the creation of WatchAnt requests, and describes the number of packets for which a reception report is requested. If one of these *WaReqNum* packets has not been forwarded, the WatchAnt reply is false, and hereby it is not relevant whether 20% or 100% of these packets have been dropped. Thus, by adjusting the parameter *WaReqNum* it is possible to influence the detection quality of WatchAnt. For the above simulation we have chose $WaReqNum = 12$.

7.4. Evaluation of the global misbehavior detection

In order to show the effects of colluding malicious nodes in the network, in our Experiment 3 we choose a basic topology and place two malicious nodes (X_1, X_2) on the shortest path between the nodes S and D, between which two unidirectional flows are established (see Figure 9). Overall, we transfer 20 000 packets (2 flows with each generating 10 packets per second over 1000 s each).

With all nodes being benign, 100% of all packets are successfully delivered. If we activate non-colluding misbehavior of nodes X_1 and X_2 the delivery ratio drops to a mere 7.49% with AntNet (no security mechanisms). In this basic topology, the activation of AntSec and WatchAnt (one-hop misbehavior detection) restores the delivery ratio to be a high as 97.5%. Activation of colluding misbehavior

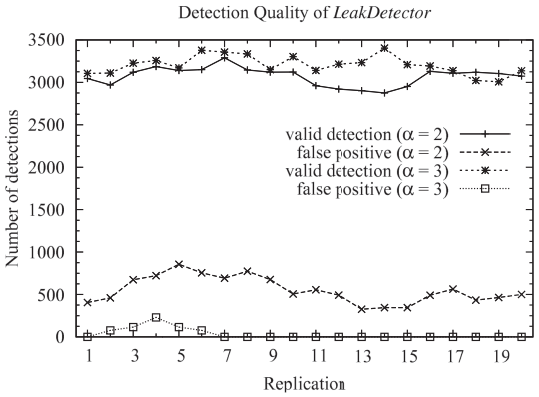


Figure 10. Number of valid and invalid detections of colluding malicious nodes.

brings the delivery ratio again down to 34.75%, even with AntSec and WatchAnt (one-hop misbehavior detection) activated. These results demonstrate that collusion is an effective strategy for malicious nodes to avoid detection by local mechanisms, which are limited to verify the correct forwarding behavior of neighboring nodes, only.

We next activate our LeakDetector component to globally detect misbehavior. Based on the traffic information transmitted in the MFANTs the destination node of a route builds a virtual graph modeling the route. The destination node calculates for each node, which participates in the route, the inflow and outflow. Based on the inflow and outflow of each node we implemented rules to decide whether a node is to be considered malicious or not. Figure 10 shows the detection quality of AntSec, WatchAnt, and LeakDetector in combination. For the parametrization of the detection mechanism we analyze values of α to be $\alpha = 2$ and 3.

For $\alpha = 2$, we have 3071 valid detections in 1000 s simulation time, i.e. a malicious node has been correctly identified as misbehaving 3071 times upon receipt of a proactive routing message. However, there have also been 551 invalid detections, i.e. a benign node has been suspected rather often for being malicious.

The results for $\alpha = 3$ are significantly better. There have been 3204 valid detections and only 34 invalid detections in average. Only very few benign nodes have been suspected falsely as malicious. This demonstrates that the *LeakDetector* provides a very good detection quality of colluding misbehavior, if tuned correctly. Its precise observations can be used to improve routing.

7.5. Summary

We have analyzed the performance of our security framework comprising AntSec, WatchAnt, LeakDetector, and AntRep for different scenarios. Our results are very promising and show that our framework is able to achieve the intended goal, namely to detect malicious nodes in WMNs that operate with the constraint (feature) of encrypted links between mesh nodes. We can also conclude that stigmergy-based secure routing mechanisms are a viable alternative to existing secure routing schemes, especially if we consider organically growing networks. Additional results and a more detailed analysis of results can be found in Reference [43].

8. SECURITY ANALYSIS

In the previous section we evaluated the the security framework both in terms of performance and cost. An exhaustive discussion of potential attack strategies and their effects in our security framework is given in this section. The discussion addresses the security challenges for WMNs stated in Section 2 and shows that all issues are either detected with our security framework or without effect.

We assume three types of threats/attackers: selfish, malicious, and defective nodes. Selfish nodes aim at conserving resources and try to cheat in protocols in order to reach this goal. Malicious nodes also aim at attacks on the operability of the whole network. Defective nodes are randomly or systematically biased in their operation. They can only do attacks that selfish or malicious nodes can also do intentionally. Therefore, we do not look further at attacks done by defective nodes, these attacks will be considered at the analysis of the attacks done by selfish or malicious nodes.

8.1. Analysis of attacks by selfish nodes

For the following we discuss attacks through selfish nodes, as listed in Table IV.

S.1a.1/S.1a.2: No Forwarding of FANTs / BANTs: This attack is detected by WatchAnt. Routing messages and data packets are both taken into account by the WatchAnt mechanism, so that loss of these packets is detected.

S.1a.3/S.2b: Modifications on the Hopcount / TTL: WatchAnt detects this attack. Therefore it stores the hashes of each packet in a way that the hash can later be recognized. However, modifying the TTL is valid in routing protocols, the TTL-field is decreased with each hop. This is taken into account, by storing various hashes of a packet with varying valid TTLs. WatchAnt detects invalid modifications.

Table IV. Potential attacks of selfish nodes.

Selfish behavior: conserving resources.
OR S.1 No participation in the routing process
OR S.1a No forwarding of routing packets (FANT, BANT)
S.1a.1 No forwarding of FANTs
S.1a.2 No forwarding of BANTs
S.1a.3 Setting of Hop-Limit / TTL to 0 or 1
OR S.1b Modification of Routing Information
S.1b.1 Worsen the quality metrics in the FANT
S.1b.2 Modifying the Backward Ant
OR S.2 Do not forward data packets
S.2a Drop data packets
S.2b Set Hop-Limit / TTL to 0 / 1

Table V. Potential attacks of malicious nodes.

M. Malicious nodes: attacks on the operability of some/all nodes
M.1 Destruction of some/all data packets
OR M.1a Disturbance of the frequency-band (traffic jamming)
M.1b Producing collisions with small effort
M.2 Generating an overload on bandwidth, CPU, energy
OR M.2a Overload of direct neighbors
M.2b Overload of arbitrary nodes on a route in the network
OR M.2b.1 Route discovery storm (many route requests)
M.2b.2 Replay old BANTs / FANTs
M.2b.3 Replay old data packets
M.2b.4 Route packets over bad (not optimal) paths
M.2b.5 Generate routing loops
M.2b.6 Forwarding all packets over a single neighbor
M.2c Overload by claiming all bandwidth for own usage
M.3 Disturbance the accurate routing
OR M.3a Packet loss/dropping (black hole routing)
M.3a.1 Send the packets to a fake neighbor
M.3a.2 Drop any packet, act as black hole
M.3b Modify routing packets
M.3c Forge routing packets
M.4 Participate unauthorized in the network
OR M.4a Use the same Node ID at different locations
M.5 Avoid being detected by WatchAnt
OR M.5a Send false information in the WatchAnt Reply (lie)
M.5b Lie to every neighbor in same frequency
M.5c Drop with a very low frequency
M.6 Use the AntRep for attacks
OR M.6a Effect benign nodes to suspect each other
M.6b Broadcast false reputations
M.7 Avoid being detected by the LeakDetector
OR M.7a Report false about information passing traffic
M.7b Drop data packets and forge invalid data packets

S.1b.2: Modifying the Backward Ant: This attack is not effective in AntSec, as BANTs are signed. Any modification to a BANT will be detected as the signature cannot be verified.

S.2a Drop data packets: WatchAnt has been designed to counteract this attack, dropping of data packets is detected.

8.2. Analysis of attacks by malicious nodes

Now we discuss the attacks listed in Table V. This list describes the possible actions of malicious nodes, which aim at threatening the operability of the network.

M.1: Destruction of some/all data packets by forging arbitrary invalid packets: Handling attacks on the PHY level are hardly to antagonize. Detecting permanent collisions could be a signal for such an attack. However, attacks on physical layer are out of scope of security mechanisms on the MAC and network layer.

M.2a: Overload of direct neighbors: Typically there is no easy way to overload neighboring nodes. For both nodes in a communication link on MAC layer the efforts needed to communicate are similar. Both need to invest transmit and receive time, both nodes have to process a transmitted packet. Thus an attack does not pay off and is not distinguish from valid traffic.

M.2b.1: Route discovery storm: In contrast to various routing algorithms AntSec does not use broadcasting. In many other routing algorithms route discovery storms are a problem, as with a single request the network can be flooded. In AntSec a route request causes the generated FANT to be routed using random walk to a specific destination node. Still in AntSec Discovery Forward Ants are sent quite frequently (400 ms). Therefore any node processes Discovery Forward Ants from any of its neighbors only if these are sent in an interval higher or equal 400 ms.

M.2b.2/M.2b.3: Replay of old packets (FANT / BANT / data): Replay of old packets can be seen as forgery of new invalid packets. The same solution counts here, too. The WatchAnt mechanism checks the correct transmission of a set of packets. If packets were forged or replayed, then the set of forwarded packets contains invalid packets and the WatchAnt reply becomes invalid and malicious nodes are detected.

M.2b.4: Route packets over bad (not optimal) paths: This attack effects longer paths and even loops may occur. As AntSec is a probabilistic routing algorithm, forwarding packets over non-optimal paths may occur and cannot be detected as attack. The effect of this attack is limited, as a malicious node can only chose the next hop for a packet, not the complete route.

M.2b.5: Generate routing loops: Rooting loops cannot be generated by one single node. This is due to the fact that AntSec is a decentralized routing algorithm, each node decides about the next hop of a packet. A single node can generate a loop only by sending a back over the hop it came. This simple attack is detected immediately. For the case that alternative nodes to the previous hop exist, it is not allowed to send the packet back.

Malicious colluding nodes can generated routing loops by passing packets to each other until the TTL limit is reached. This attack is equal in its effect to the dropping attack of colluding malicious nodes. The counteraction to this attack is defined by the LeakDetector, which detects leaks on routes caused by colluding malicious nodes.

M.2b.6: Forwarding as many packets as possible over a specific neighboring node: A malicious node may forward any packet it receives to a specific neighboring node to stress it. This cannot be detected as misbehavior, as it may also be protocol conform. Each node has only local sight on the network, so due to the topology it may be that some neighboring nodes are stronger utilized in routing than others.

M.2c: Overload by claiming all free bandwidth for own usage: A malicious node can try to reserve all free bandwidth for itself and decrease by this the remaining bandwidth for all other nodes. This attack is performed on the MAC layer. With our security framework focusing on dependable routing we cannot detect this. The MAC layer should provide mechanisms to detect invalid reservations.

M.3a.1: Send the packets to a fake neighbor: It is ineffective for an attacker to denote to have sent packets to a node,

which is not existing in reality. The WatchAnt Reply may contain the Node ID of this fake node, but the WatchAnt requesting node checks whether the listed nodes are known and authenticated. If not, the WatchAnt Reply is invalid. A fake node cannot be authenticated as there exists no Registration Certificate. Malicious nodes may try to use a Registration Certificate of a distant node to imply that this node is in its near. However, this attack is addressed is addressed during the neighborhood establishment phase. In that phase, nodes validate their direct neighborhood.

M.3a.2: Drop any packet, be a black hole: The dropping of routing and data messages is detected by WatchAnt. Additionally AntSec decreases the reputation of a neighboring node when a FANT was sent over this neighboring node and no BANT arrives.

M.3b: Modify routing packets: The modification of FANTs is not detected on the route to the destination. But as the signed BANT returns on the same path the modification is detected since there is no corresponding FANT to the received BANT. BANTs modifications are detected, as BANTs are signed and easy to verify.

M.3c: Forge routing packets: Forward Ant can be forged undetected. A FANT does not contain any information assuring its authenticity and integrity. A forged or spoofed FANT is processed regularly. The corresponding BANT is also processed regularly. However, the malicious effect is limited as the effect of a forged packet is equal to the effect of a regular FANT sent from the malicious node. The processing of FANTs and BANTs does not take into account the sender of the packets. Processing a BANT modifies the route to each node on the path to the destination node, but the source node is irrelevant.

Forging of BANTs is futile as BANTs need to be signed and verified before processing. Spoofing other nodes is impossible, as their signature cannot be created by any other node. The malicious node may create a BANT to a non-existent FANT to reinforce specific paths. However, this BANT is dropped by the first benign node since no corresponding FANT is known.

M.4a: Use the same NodeID at different locations: If different nodes use the same Registration Certificate and misbehave, then complaints about this node are faster and in larger amount reported. The problem occurs when nodes using these multiple identities are well behaving and do not behave suspiciously. In this case malicious nodes can participate with this identity on several places in the network. However, routing cannot be guaranteed for this malicious node, as packets addressed for this identity may be routed to any copy of this identity.

M.5a, M.5b: Avoid being detected by WatchAnt by lying in the WatchAnt reply: A malicious node may drop all data packets and report in the WatchAnt Reply to have sent all packets to one single neighbor. The node, which has sent the WatchAnt Request assumes that all packets have been forwarded correctly. The only node detecting the misbehavior is the node listed in the WatchAnt Reply.

The malicious node could use two different strategies. In the first case, the malicious node lists every time the same neighboring node as next hop for all packets received. This neighboring node detects the misbehavior, may even deny any service for the malicious node. However, the detecting node may be irrelevant for the malicious node, thus no real punishment is experienced. In the second case, the malicious node reports every time another neighbor of having received all packets. In this case the local reputation of the malicious node is decreased periodically in each neighbor by a little, but with recovery the negative effect may be reduced to none. In both cases WatchAnt can be optimized by modifying the parameters, e.g. query frequency, to be more sensitive on frequent false WatchAnt replies. Still using second hand reputations can improve the detection quality.

M.5c: Drop with a very low frequency: By dropping with a very low frequency a malicious node may try to drop packets to which no report is requested. Nevertheless, in order to avoid dropping packets in a requested window the frequency must be very low rate. As misbehavior is rated stronger than good behavior, the reputation recovery time has to be considered as well. Thus, the attack is not effective as avoiding detection requires a very low (irrelevant) drop rate. This case is investigated in the second experiment in Section 7.

M.6a: Effect that benign nodes suspect each other: The strongest punishment done by nodes in the network is to deny service to a neighboring node, which is suspected of being malicious. A malicious node may try to effect that benign nodes suspect each other of being malicious and mutually deny service. Very few actions can be done to lower the reputation of a third party node. The WatchAnt mechanism which has the greatest impact on the reputation of a node cannot be used to discredit a neighboring node from the perspective of another node. Any action that may discredit third party nodes, involves the degradation of the own reputation in a stronger scale.

M.6b: Broadcast false reputations: Broadcasting false reputation values can be used to worsen the reputation of neighboring nodes. Taking second hand reputations into account requires mechanisms to detect invalid second hand reputations coming from malicious nodes. Sonja Buchegger presents in Reference [41] various strategies to cope with unverified second hand reputation values in routing scenarios.

M.7a: Report false information about passing traffic: Using LeakDetector requires the nodes to report the amount of traffic on the route they have received from the previous hop of the FANT. By reporting a false information malicious nodes may try to accuse the previous hop for packet dropping. The destination node does not detect this attack, the information is considered as valid in the Leak Detector. The attack is detected on in the next step. The BANT is sent back by the destination node using the route on which the corresponding FANT came. As the BANT arrives at the node which was the previous hop for the FANT before the malicious node the invalid modification is detected. This falsely accused node assumes that up to its successor no modification has been detected as the BANT was

not dropped. Therefore the accused node drops the invalid BANT and decreases the reputation of its successor: the malicious node.

M.7b: Dropping and forging of data packets by colluding malicious nodes: Sending of invalid or corrupted packets with spoofing of source addresses is a common problem. The main function of routing is provide dependable end to end communication. The content of the transferred packets is not of interest for the routing layer. WatchAnt detects forged and dropped data and routing packets in one hop range. LeakDetector detects dropped packets, i.e. leaking traffic, in end-to-end communication. However, both WatchAnt and LeakDetector do not detect fake traffic generated through colluding nodes which also drop real traffic. This case happens if for every packet dropped in a cloud of colluding malicious nodes a new packet is forged.

9. CONCLUSION

Guaranteeing security is a crucial requirement for WMNs, thus enabling the support for networks such as wireless community networks, meshed radio access networks or mesh networks for factory/process automation. Contemporary wireless technologies such as the IEEE 802.16 or the IEEE 802.11s standard lack in comprehensive security mechanisms supporting the establishment of mesh networks, however. We have proposed a security framework to address the security challenge in WMNs in a holistic manner. Our framework combines secure routing mechanisms with a reputation management scheme as well as mechanisms for local and global misbehavior detection. The combination of these building blocks enables to provide for secure, self-organizing WMNs. Amongst others, our solution it is able to operate under the challenging setting of per-link encryption deployed in WMNs.

As a proof-of-concept, we implemented and showcased one instantiation of our framework for the case of an IEEE 802.16 MeSH network. By means of a simulation study we tested the individual components of our system and showed its feasibility. Our contribution does not mark the end of research in the field of WMN security, though, but rather the beginning, since it opens up new avenues of research for the challenging area of multihop wireless networks.

ACKNOWLEDGEMENTS

The authors would like to thank the reviewers for their helpful feedback and comments which has led to improvements of the quality of the paper.

REFERENCES

1. Mogre P, Graffi K, Hollick M, Steinmetz R. AntSec, WatchAnt and AntRep: innovative security mechanisms for wireless mesh networks. In *IEEE LCN 07: Local Computer Networks 2007*, 2007.
2. Graffi K, Mogre P, Hollick M, Steinmetz R. Detection of colluding misbehaving nodes in mobile ad hoc and mesh networks. In *IEEE GLOBECOM '07*, 2007.
3. Akyildiz I, Wang X, Wang W. Wireless mesh networks: a survey. *Computer Networks* 2005; **47**(2): 445–487.
4. Mogre P, Hollick M, Steinmetz R. The IEEE 802.16-2004 MESH Mode Explained. Multimedia Communications Lab, Technische Universität Darmstadt, Technical Report KOM-TR-2006-08, Dec 2006.
5. I. IEEE Computer Society, 802.16 Standard for Local and Metropolitan Area Networks, Part 16: Air Interface for Fixed Broadband Wireless Access Systems, 1 Oct. 2004.
6. Marti S, Giuli T, Lai K, Baker M. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *Sixth Annual International Conference on Mobile Computing and Networking*, 2000, pp. 255–265.
7. Johnson D, Maltz D. Dynamic source routing in ad hoc wireless networks. In Mobile Computing, Imielinski T, Korth H (eds). Kluwer Academic Publishers: Dordrecht, The Netherlands, 1996.
8. N. W. G. Internet Engineering Task Force. RFC 3561 Ad Hoc On-Demand Distance Vector (AODV) Routing, http://www.ietf.org/rfc/rfc3561.txt, 2003.
9. Djenouri D, Khelladi L, Badache AN. A survey of security issues in mobile ad hoc and sensor networks, *Communications Surveys & Tutorials, IEEE* 2005; **7**(4): 2–28.
10. Siddiqui MS, C. S. v. Security issues in wireless mesh networks. *International Conference on Multimedia and Ubiquitous Engineering*, vol. 0, 2007; pp. 717–722.
11. Kargl F, Geiss A. Secure Dynamic Source Routing. *Hawaiian International Conference on System Sciences 38, Hawaii, USA*, Jan. 2005.
12. Hu Y, Perrig A, Johnson D. Ariadne: a Secure On-Demand Routing Protocol for Ad Hoc Networks. In *8th ACM International Conference on Mobile Computing and Networking*, Sept. 2002.
13. Sanzgiri K, Dahill B, Levine B, Belding-Royer E. A Secure Routing Protocol for Ad Hoc Networks. *International Conference on Network Protocols (ICNP), Paris, France*, Nov. 2002.
14. Zapata MG. Secure Ad hoc On-Demand Distance Vector Routing. *ACM Mobile Computing and Communications Review (MC2R)* 2002; **3**: 106–107.
15. Perrig A, Canetti R, Tygar D, Song D. The TESLA Broadcast Authentication Protocol. In *Cryptobytes*, vol. 5, no. 2. RSA Laboratories, 2002, pp. 2–13.
16. Hu Y, Perrig A, Johnson D. Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks. Department of Computer Science, Rice University, Technical Report, Dec. 2001.

17. Papadimitratos P, Haas Z. Secure Routing for Mobile Ad Hoc Networks. *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference*, 27 Jan. 2002.
18. Hu Y, Johnson D, Perrig A. SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks. In *4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002), IEEE, Calicoon, NY*, June 2002.
19. Buttyan L, Hubaux J-P. Enforcing Service Availability in Mobile Ad Hoc WANs. In *Proceedings of the First IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, 2000.
20. Zhong S, Yang Y, Chen J. Sprite: A Simple, Cheat-proof, Credit-based System for Mobile Ad Hoc Networks, 2002.
21. Kargl F, Schlott S, Klenk A, Geiss A, Weber M. Advanced Detection of Selfish or Malicious Nodes in Ad Hoc Networks. Proceedings of 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004), Springer Lecture Notes in Computer Science, Heidelberg, September 2004.
22. Djenouri D, Badache N. Cross-Layer Approach to Detect Data Packet Droppers in Mobile Ad-Hoc Networks. In *Proc. of Int. Workshop on Self-Organizing Systems '06*, ser. Lecture Notes in Computer Science, de Meer H, Sterbenz JPG (eds). Vol. 4124. Springer, 2006; 163–176.
23. Buchegger S, Boudec J-YL. Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. In *Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing*. Canary Islands, Spain: IEEE Computer Society, January 2002, pp. 403–410.
24. Buchegger S, Boudec JL. Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. In *Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing*, IEEE Computer Society, Ed., Jan. 2002, pp. 403–410.
25. Buchegger S, Boudec J-YL. Performance Analysis of the CONFIDANT Protocol: Cooperation of Nodes—Fairness in Dynamic Ad Hoc Networks. In *Proceedings of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*. Lausanne, CH: IEEE, June 2002.
26. Bansal S, Baker M. Observation-based Cooperation Enforcement in Ad Hoc Networks. Stanford University, Tech. Rep., 2003.
27. Michiardi P, Molva R. Core: A COllaborative REputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks. EurecomResearch, Technical Report RR-02-062, Dec. 2001.
28. Zhang Y, Lee W, Huang Y. Intrusion Detection Techniques for Mobile Wireless Networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, 2002.
29. Lee W, Xiang D. Information-Theoretic Measures for Anomaly Detection. In *Proceedings of the 2001 IEEE Symposium on Security and Privacy*, May 2001, pp. 130–143.
30. Di Caro G, Dorigo M. Antnet: Distributed Stigmergetic Control for Communications Networks. *Journal of Artificial Intelligence Research* 1998; **9**: 317–365.
31. Schoonderwoerd R, Bruten JL, Holland OE, Rothkrantz LJM. Ant-Based Load Balancing in Telecommunications Networks. *Adaptive Behavior* 1996; **5**(2): 169–207.
32. Günes M, Spaniol O. Routing Algorithms for Mobile Multi-hop Ad Hoc Networks. In *Net-Con 2003*, 2003, 120–138.
33. Baras J, Mehta H. A Probabilistic Emergent Routing Algorithm for Mobile Ad hoc Networks. In , March 3–5, 2003. *WiOpt'03: Modeling and Optimization in Mobile Ad Hoc and Wireless Networks*, 3 Mar. 2003.
34. Roth M, Wicker S. Termite: Emergent Ad-Hoc Networking. In *Second Mediterranean Workshop on Ad-Hoc Networks*, 2003.
35. Di Caro G, Ducatelle F, Gambardella LM. *AntHocNet: an Ant-Based Hybrid Routing Algorithm for Mobile Ad Hoc Networks*, ser. Lecture Notes in Computer Science. Springer-Verlag: Birmingham, UK, Sept. 2004, no. 3242, best paper award.
36. Wang J, Osagie E, Thulasiraman P, Thulasiram RK. Hopnet: A Hybrid Ant Colony Optimization Routing Algorithm for Mobile Ad Hoc Network. *Ad Hoc Networks* 2009; **7**(4): 690–705, I. Bio-Inspired Computing and Communication in Wireless Ad Hoc and Sensor Networks.
37. Zhong W, Evans D. When Ants attack: Security Issues for Stigmergic Systems. University of Virgina, Department of Computer Science, Tech. Rep. CS-2002-23, Apr. 2002.
38. Melchor CA, Salem BA, Gaborit P, Tamine K. AntTrust: A Novel Ant Routing Protocol for Wireless Ad-hoc Network Based on Trust between Nodes. *Availability, Reliability and Security, International Conference on*, vol. 0, pp. 1052–1059, 2008.
39. Diffie W, Hellman ME. New Directions in Cryptography. *IEEE Transactions on Information Theory* 1976; **IT-22**(6): 644–654.
40. Baran B. Improved AntNet Routing. *SIGCOMM Computer Communication Review* 2001; **31**(2): 42–48.
41. Buchegger S. Coping with misbehavior in mobile ad-hoc networks. Ph.D. dissertation, Swiss Federal Institute of Technology (EPFL), April 2004.
42. Barr R, Haas Z, van Renesse R. JiST: an Efficient Approach to Simulation using Virtual Machines: Research Articles. *Software Practical Experiments* 2005; **35**(6): 539–576.

43. Graffi K. A Security Framework for Organic Mesh Networks. Master's thesis, Technische Universität Darmstadt, Germany, 2006.
44. Di Caro G, Dorigo M. Antnet: a Mobile Agents Approach to Adaptive Routing. Universite Libre de Bruxelles, IRIDIA, Tech. Rep. 12, 1997.

AUTHORS' BIOGRAPHIES



Parag S. Mogre obtained the M.Tech. degree in Computer Science and Engineering from the Indian Institute of Technology, Guwahati in 2004. His Masters Thesis has won the KuVS prize in 2005. Since October 2004 he is active as a researcher at the Multimedia Communications Lab (KOM), TU Darmstadt, Germany, and is pursuing his Ph.D. there. His research focuses on medium access control, routing and resource management in wireless mesh and *ad hoc* networks. He has over 10 patent applications in the area of WiMAX based mesh networks, and has been awarded the invention achievement award for 2007/2008 by the Department of Electrical Engineering and Information Technology, at the TU Darmstadt. He is leading the mobile communications and sensor networking group at KOM, TU Darmstadt.



Kalman Graffi graduated with diploma degrees in both computer science and mathematics in 2006 at the Technische Universität Darmstadt, in Germany. Since 2006 he is a researcher and Ph.D. student at the Multimedia Communications Lab (KOM), TU Darmstadt. His research focuses on security, monitoring and management of the quality of service of peer-to-peer systems.



Dr.-Ing Matthias Hollick is heading the Secure Mobile Networking lab at the Computer Science department of TU Darmstadt, Germany. He received his doctoral degree (Dr-Ing) in 2004 from the TU Darmstadt. His research focuses on secure and quality of- service-aware communication for mobile and wireless *ad hoc*, mesh, and sensor networks. Dr Hollick has been working and researching at TU Darmstadt, UC3M, and the UIUC. In 2005, Dr Hollick has received the Adolf-Messer Foundation award.



Dr.-Ing Ralf Steinmetz is professor and chair of the Multimedia Communications Lab at the TU Darmstadt. Together with more than 30 researchers, he is working toward the vision of 'seamless multimedia communications'. He has contributed to over 400 refereed publications. He is an ICCC Governor, Fellow of both the IEEE and ACM. Professor Dr Ralf Steinmetz is a member of the Scientific Council and president of the Board of Trustees of the international research institute IMDEA Networks.