Detection of Colluding Misbehaving Nodes in Mobile Ad hoc and Wireless Mesh Networks

Kálmán Graffi, Parag S. Mogre, Matthias Hollick, and Ralf Steinmetz
Email: {graffi,pmogre,mhollick,rst}@kom.tu-darmstadt.de
Multimedia Communications Lab (KOM), Technische Universität Darmstadt,
Merckstr. 25, 64283 Darmstadt, Germany

***Abstract*—Ubiquitous network connectivity and mobile communications have recently attracted remarkable attention. Wireless multihop networks such as Mobile Ad hoc Networks or Wireless Mesh Networks have been proposed to cater to the arising needs. Various security challenges persist, esp. because these networks build on the premise of node cooperation. Secure routing protocols and mechanisms to detect routing misbehavior in the direct neighborhood exist; however, collusion of misbehaving nodes has not been adequately addressed yet. We present *LeakDetector*, a mechanism to detect colluding malicious nodes in wireless multihop networks. In combination with proactive secure multipath routing algorithms, *LeakDetector* enables the calculation of the packet-loss ratio for the individual nodes. We perform an experimental analysis, which shows the excellent detection quality of *LeakDetector*.**

I. Introduction

In a world where ubiquitous computing is emerging, broadband wireless networks address the continuously increasing demand for fast and reliable mobile network access.

Mobile Ad hoc Networks (MANET) or Wireless Mesh Networks (WMN) promise several benefits in comparison to the traditional single-hop wireless networks such as cellular networks. In MANETs and WMNs the range/capacity of the network is extended by active cooperation of the participating nodes. Nodes in the network act as routers and forward messages on behalf of the other nodes. The premise of node cooperation induces various challenging security issues.

One of the main issues in the process of routing of messages in the aforementioned class of networks is that the cooperation of nodes cannot be assumed in general. It can be beneficial for nodes to misbehave during the process of routing/forwarding, e.g. to save resources such as energy. A common attack is to drop messages of other nodes. Several approaches to detect routing/forwarding misbehavior in a node's one-hop neighborhood have been proposed. However, considering only the one-hop neighborhood disregards an important security problem: the collusion of misbehaving nodes.

Colluding misbehaving nodes are able to cloak the actions of each other in order to prevent detection of misbehavior. In Section II, we describe the problem of colluding misbehaving nodes in detail. We survey existing solutions to detect routing misbehavior (with and without collusion of misbehaving nodes) and highlight the shortcomings of these solutions to detect collusion of attackers in Section III. In Section IV, we state the assumptions for our solution to the problem and introduce *LeakDetector*, a mechanism to detect colluding misbehaving nodes in the network. Our solution comes with a low overhead and at no additional computational cost, as it requires no further cryptography. Section V presents the evaluation of our mechanism. We show that the *LeakDetector* is a very precise mechanism to detect misbehaving nodes that maliciously drop messages. Finally, in Section VI, we conclude our work.

II. Defective Routing: A Problem Statement

MANETs and WMNs are based on wireless multihop communication. Messages are transferred between the end systems using hop-by-hop forwarding via intermediate nodes. To ensure the correct operation of the routing system, the nodes need to cooperate and forward messages of other nodes according to the protocol specification. However, for an individual node it might be benefical *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 systems.

Various security solutions exist to provide a secure routing functionality. E.g., several secure routing algorithms are designated to prevent forging, invalidly modifying, and dropping of routing messages. However, to detect forwarding misbehavior, additional mechanisms are needed to monitor the forwarding of data messages, as well. In [1], Marti et al. propose a mechanism called *watchdog* to solve this issue. 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 forwarded reveals, whether the observed node is forwarding messages appropriately or not.

However, the reach of solutions like watchdog 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[1] node is able to observe the routing behavior. We define the following behavior of a node X_2 as *malicious and colluding*.

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

"A node X_2 acts *maliciously colluding*, if it (selectively) drops messages m received from a neighboring node X_1 in the case that X_1 is a colluding malicious node and the messages m have not been originated by malicious nodes colluding with X_2, e.g., X_1."

Let us consider the scenario presented in Fig. 1. 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.

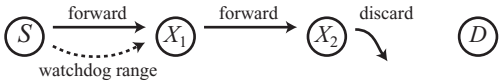


Fig. 1. Example of colluding misbehavior: X_1 forwards messages to X_2, who drops these messages. S observes "correct" behavior of X_1. X_1 ignores X_2's misbehavior.

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 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 MANETs and WMNs. Depending on the deployed routing algorithm, malicious collusion might prohibit regular communication over multihop routes, because message drops are excessively high.

In this paper we present a solution to detect colluding misbehavior and to identify malicious nodes. This information can be used to adapt routing strategies and to enable more dependable routing in MANETs and WMNs.

III. Related Work

In this section we give a brief introduction to secure routing algorithms proposed for MANETs and discuss the limitations of these algorithms. We briefly introduce mechanisms to detect forwarding misbehavior in a node's one-hop neighborhood. Finally, we present solutions in literature to detect misbehavior of colluding nodes.

Secure routing algorithms typically protect the route setup and maintenance phase. They counter various attacks such as forging, modifying, or dropping of routing messages. E.g., *Secure DSR* [2], *Ariadne* [3], *ARAN* [4], and *Secure AODV* [5] 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.

Several mechanism to detect forwarding misbehavior exist. As already mentioned, watchdog [1] is a widely discussed solution to detect forwarding misbehavior. Watchdog relies on overhearing the communication of neighboring nodes (one-hop neighbors), which limits its applicability. E.g., collisions on the wireless medium can lead to misinterpretations whether a neighbor is behaving according to the specifications or not (see [6] for further explanations). Also, the watchdog mechanism is inapplicable in state-of-the-art wireless technology such as the IEEE 802.16 MeSH Mode [7], because link-level encryption and the highly optimized MAC-layer hinders a proper overhearing of the channel.

Djenouri et al. [8] propose an alternative approach for detecting forwarding misbehavior. The authors suggest the usage of authenticated two-hop acknowledgments per message, which inform the two-hop precursor node if the intermediate node is behaving correctly. However, this solution causes a high traffic overhead and fails to address collusion among misbehaving nodes, i.e., a colluding node may incorrectly acknowledge messages that have been dropped.

Kargl introduces in [9] a mechanism called *iterative probing*. A probe is an encrypted field in a modified packet structure that reveals information only to a specified node on the route. The field is encrypted with a pre-shared key between the probed node and the source node. Fig. 2 shows the basic principle of this approach. The source S sends probes to every node on the route starting with the destination node D. As the malicious node (here: X_1) drops every data packet, it drops the probes, too. The first node that answers the probe request is, thus, either the malicious node itself or its predecessor.

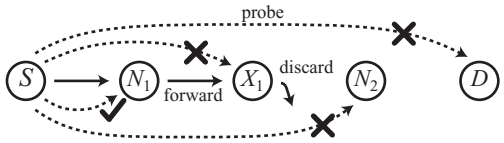


Fig. 2. Working principle of iterative probing

Iterative probing provides a solution to detect malicious nodes en-route even if they are colluding. However, this comes at a high price: the packet format has to be extended and pairwise cryptographic keys have to be negotiated between the source node and each individual node en-route.

In summary, we conclude that several security issues of routing in MANETs and WMNs are adequately addressed; however, these networks have still be considered as vulnerable if nodes are colluding in order to hide their misbehavior.

IV. *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.

A. 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 & 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.

B. 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:

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.

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 parameterization of one byte for the T_{N1} field, we obtain a resolution of $100/255 = 0.4$ for the obtained fraction.

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.

4) Detection Criteria: a node in the route is not considered malicious if:

- the node is source S or destination D of the route.
- less than 50 packets have been received for this route (a minimal set of observations is required).
- 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.

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 also considered benign.

5) Maintenance of Counter and Reconciliation of False Detections: Periodic initialization of the traffic counter (e.g., every 10 minutes) 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 only slowly react to such nodes. Resetting the counter should be loosely synchronized; in a time window of 30 seconds 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) Reaction to Malicious Nodes: once the destination node detects a node en-route as malicious, various strategies can be applied. E.g., 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.

C. Example

We give an example of a virtual graph that may be observed at node D in Fig. 3. A proactive routing message using the path $S \rightarrow N_1 \rightarrow N_2 \rightarrow N_3 \rightarrow N_6 \rightarrow N_9 \rightarrow D$ carries the information on the fraction of traffic corresponding to this route received from the previous hop: $(T_{total})S \rightarrow (0.45)N_1 \rightarrow (0.45)N_2 \rightarrow (0.4)N_3 \rightarrow (0.1)N_6 \rightarrow (0.2)N_9 \rightarrow (0.4)D$.

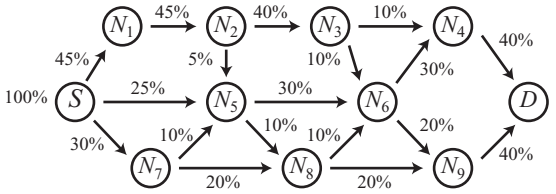


Fig. 3. Sample virtual graph built by the destination node D.

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.

TABLE I
COMPARISON OF THE IN AND OUT TRAFFIC PER NODE

	S	$N_{1,2}$	N_3	$N_{4,5}$	N_6	$N_{7,8}$	N_9	D
In		45%	40%	40%	50%	30%	40%	80%
Out	100%	45%	20%	40%	50%	30%	40%	

V. EVALUATION

In order to evaluate the quality of our solution, we perform a simulation study studying *LeakDetector* together with a secure proactive routing algorithm.

A. Goals and Metrics

We consider the following metrics and goals.

- *Throughput, Loss*, to show the application level effect of collusion of malicious nodes.
- *Detection Quality of LeakDetector*, to study whether colluding malicious nodes are detected, and if they are detected how many valid/invalid detections occur.

B. Simulation Setup

For our study, we use AntSec [6], [10], a secure version of AntNet [11], [12]. AntSec is a proactive, probabilistic, multipath, stigmergy-based, distributed, non-broadcasting, secure routing algorithm inspired by the routing behavior of ants. AntSec fulfills the assumptions stated in Subsection IV-A; without loss of generality, we use AntSec as a representative algorithm for the aforementioned class of routing algorithms[2].

For our simulation, we use a consolidated and extended version of the JiST/SWANS [13] discrete event simulator.

[2]We do not expect a significant influence of the routing algorithm on the detection quality of *LeakDetector*, because we measure only the state of routing paths and do not provide feedback to the routing algorithm, yet.

TABLE II
SIMULATION PARAMETERS

Parameter	Value
Number of nodes	7
PHY/MAC Layer	IEEE 802.16 MeSH Mode, ETSI spec. (n = 8/7 oversampling factor, 3.5 MHz channel, OFDM 256)
Node Placement	see topology in Fig. 4
Node Mobility	stationary nodes
Application	Constant Bit Rate (CBR)
Number of flows	2
Flow pairing	fixed: $S \rightarrow D$, $D \rightarrow S$
Sending rate per flow	10 packets per sec.
Packet size	512 bytes
Simulation time	1000 s
Replications	20

Without loss of generality, we use the IEEE 802.16 MeSH mode [14] as MAC layer, which is a standardized, state-of-the-art MAC layer for WMNs. The list of simulation parameters for our study is shown in Table II.

In order to show the effects of colluding malicious nodes in the network, we choose a small fixed topology and place 2 malicious nodes (X_1, X_2) on the shortest path between the nodes S and D, between which two unidirectional flows are established (see Fig. 4).

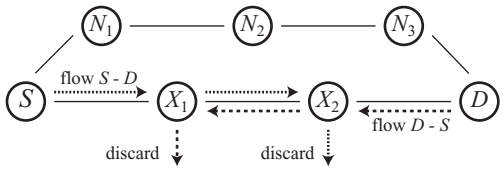


Fig. 4. Simulation topology: 2 flows, from S to D and vice versa, have been established. Nodes X_1, X_2 are (colluding) malicious nodes.

C. Results

Fig. 5 shows the effect of non-colluding and colluding misbehaving nodes X_1, X_2. By comparing the second and third bar in Fig. 5 we can see that AntSec effectively detects misbehavior and increases throughput in the non-colluding, malicious case. Here, throughput increases from 7.49% when the misbehaving mechanisms for the direct neighborhood (AntSec + one-hop misbehavior detector) is disabled to 97.5% when it is enabled. The malicious nodes decrease the throughput in the network from 97.5% to 34.75% when colluding, i.e. dropping data packets of benign nodes when received from the partnering malicious node. Throughput decreases from 97,5%, when the one-hop misbehavior detector is enabled and malicious nodes are not colluding, to 34,75% in case that the one-hop misbehavior detector is enabled and malicious nodes are colluding.

Our results demonstrate that collusion is an effective strategy for malicious nodes to avoid detection by mechanisms, which are limited to verify the correct forwarding behavior of neighboring nodes, only.

Please note that the *LeakDetector* is a passive element in our simulation scenario, i.e., it does not give any feedback to the routing system. In Table III and Fig. 6 we present the

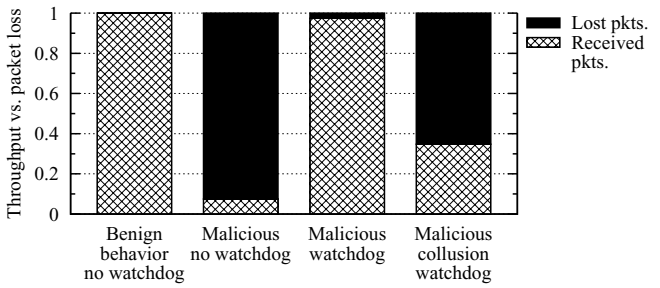


Fig. 5. Impact of colluding malicious nodes on application-level performance

results for the tuning parameter α of the *LeakDetector* to be α = 2 and 3.

For α = 2, we have 3071 valid detections in 1000 seconds 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 α = 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, if tuned correctly. Its precise observations can be used to improve routing. However, we perceive there is still room for further optimization.

VI. Conclusion

Colluding malicious nodes are a severe risk for MANETs and WMNs, which rely on node collaboration. By working together, malicious nodes are able to trick well-behaving nodes. Their misbehavior is revealed only to other malicious nodes. However, since colluding nodes work together, their misbehavior is not detected and, hence, goes unpunished. Existing work in literature describes this problem, but presents only very expensive solutions (e.g., iterative probing [9]) that can be considered infeasible for the studied class of networks.

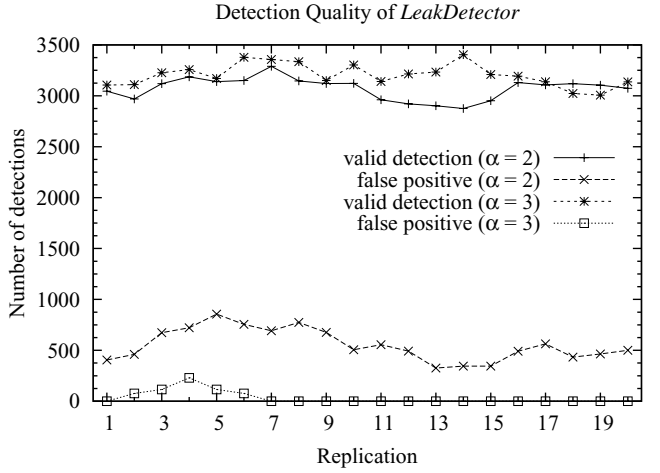


Fig. 6. Number of valid and invalid detections of colluding malicious nodes

TABLE III
Average number of valid and invalid detections of malicious colluding nodes on the route.

Param.	Valid detections	Std. dev.	Invalid detections	Std. dev.
α = 2	3071.6	310.7	551.4	473.4
α = 3	3204.4	357.8	34.4	153.8

We developed the (*LeakDetector*) mechanism to detect colluding malicious nodes. It can be used in combination with any proactive, multipath, non-broadcasting, secure routing algorithm. We implemented our proposed solution and performed a simulation study using the contemporary IEEE 802.16 MeSH mode to quantify the effects of colluding misbehaving nodes in WMNs.

The *LeakDetector* is able to efficiently tackle this problem and presents good detection results, as seen from the evaluation. The *LeakDetector* is one of the first mechanisms for addressing the problem of malicious colluding nodes in WMNs.

References

[1] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," in *Proceedings of ACM MobiCom '00*, 2000.

[2] F. Kargl and A. Geiss, "Secure Dynamic Source Routing," in *Proceedings of HICSS 38*, 2005.

[3] Y. Hu, A. Perrig, and D. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," in *Proceedings of ACM MobiCom '02*, 2002.

[4] K. Sanzgiri, B. Dahill, B. Levine, and E. Belding-Royer, "A Secure Routing Protocol for Ad Hoc Networks," in *Proceedings of IEEE ICNP '02*, 2002.

[5] M. G. Zapata, "Secure Ad hoc On-Demand Distance Vector Routing," *ACM Mobile Computing and Communications Review (MC2R)*, vol. 6, no. 3, pp. 106–107, July 2002.

[6] K. Graffi, "A Security Framework for Organic Mesh Networks," Master's thesis, Technische Universität Darmstadt, Germany, 2006.

[7] IEEE Computer Society, IEEE Microwave Theory and Techniques Society, "802.16 Standard for Local and Metropolitan Area Networks, Part 16: Air Interface for Fixed Broadband Wireless Access Systems," 1 Oct. 2004.

[8] D. Djenouri and N. Badache, "Cross-Layer Approach to Detect Data Packet Droppers in Mobile Ad-Hoc Networks," in *Proceedings of IWSOS '06*, 2006.

[9] F. Kargl et al., "Advanced Detection of Selfish or Malicious Nodes in Ad Hoc Networks," in *Proceedings of ESAS '04*, 2004.

[10] P. Mogre, K. Graffi, M. Hollick, and R. Steinmetz, "AntSec, WatchAnt and AntRep: Innovative Security Mechanisms for Wireless Mesh Networks," in *Proceedings of IEEE LCN '07*, 2007.

[11] G. Di Caro and M. Dorigo, "Antnet: a Mobile Agents Approach to Adaptive Routing," Universite Libre de Bruxelles, IRIDIA, Tech. Rep. 12, 1997.

[12] B. Baran, "Improved AntNet Routing," *SIGCOMM Computer Communication Review*, vol. 31, no. 2, pp. 42–48, 2001.

[13] R. Barr, Z. Haas, and R. van Renesse, "JiST: an Efficient Approach to Simulation using Virtual Machines: Research Articles," *Software–Practice & Experience*, vol. 35, no. 6, pp. 539–576, 2005.

[14] P. Mogre, M. Hollick, and R. Steinmetz, "The IEEE 802.16-2004 MeSH Mode Explained," Multimedia Communications Lab, Technische Universität Darmstadt, Germany, Tech. Rep. KOM-TR-2006-08, Dec 2006. [Online]. Available: ftp://ftp.kom.tu-darmstadt.de/pub/TR/KOM-TR-2006-08.pdf