Using Context Information to Evaluate Cooperativeness

Friederike Klan
Institute of Computer Science
Friedrich-Schiller-University
Jena
Ernst-Abbe-Platz 1-4
07743 Jena, Germany
friederike.klan@informatik.uni-jena.de

Mathias Röhl
University of Rostock
Albert-Einstein-Str. 21
18059 Rostock, Germany
mroehl@informatik.uni-rostock.de

Birgitta König-Ries
Institute of Computer Science
Friedrich-Schiller-University
Jena
Ernst-Abbe-Platz 1-4
07743 Jena, Germany
koenig@informatik.uni-jena.de

Adelinde M. Uhrmacher
University of Rostock
Albert-Einstein-Str. 21
18059 Rostock, Germany
lin@informatik.uni-rostock.de

ABSTRACT

In contrast to traditional enterprise service-oriented systems, service announcement, storage, and lookup in MANETs is imposed on the network participants itself. Consequently, cooperation among network participants is crucial to allow for a properly working system. In this context, protocol-based reputation methods have been proposed. Though very successful in many cases, those techniques cannot effectively deal with all observable types of misbehavior. In this paper, we propose a general reputation scheme for evaluating cooperativeness according to a higher-level protocol in MANETs. It complements protocol-based reputation mechanisms by monitoring techniques, to evaluate cooperativeness of the network participants based on currently and formerly (message context) overheard messages. We exemplarily describe and simulatively evaluate a trust building mechanism based on context information for the Lanes overlay protocol for service announcement and discovery.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network monitoring; C.2.1 [**Network Architecture and Design**]: Store and forward networks,Wireless communication; I.6 [**Simulation and Modeling**]: Miscellaneous; I.6.8 [**Types of Simulation**]: Discrete event; H.3 [**Information Storage and Retrieval**]: Miscellaneous

General Terms

Reliability, Security

Q2SWinet'08, October 27–28, 2008, Vancouver, BC, Canada.

1. INTRODUCTION

Collaboration and sharing of resources is a key issue in distributed and mobile systems. Service-oriented computing has proved to be an appropriate paradigm to enable this. It allows to provide functionality or information as stand-alone services, that can be described, published, automatically discovered, selected, composed, and executed. However, traditional enterprise service-oriented systems cannot be simply applied to mobile ad-hoc networks, since those environments pose a number of additional challenges. One important difference is that service announcement, storage and lookup is no longer managed by a small number of more or less central and trustworthy repository providers. Instead, this work is imposed on the network participants itself. For that reason cooperation, e.g. when forwarding service offers or service requests of the devices, is essential for a properly working system. Moreover, it should be ensured that the burden is shared equally between network participants.

In this context, a number of incentive mechanisms have been proposed to stimulate cooperativeness between autonomous devices. Most of the approaches address the problem of stimulating honest service provision utilizing protocol-based reputation mechanisms [6], only a few apply economic incentive patterns, typically payment schemes, to stimulate active participation in the process of service announcement and discovery [13]. Though working well in many situations, those mechanisms cannot effectively deal with all types of observable misbehavior [1]. In particular, they fail in cases where misbehavior is not directly observable by the protocol. Moreover, they impose a lot of additional effort, in terms of sending and processing messages, as well as storing and managing evidences and reputation values, to the nodes. This is particularly problematic when considering mobile devices, which are restricted in terms of battery power, network bandwidth, and storage capabilities.

With respect to those problems reputation-based approaches, based on monitoring node behavior by overhearing message traffic, seem to be more promising. In this context, various solutions, particularly related to message routing, have been proposed [2, 3, 10, 11, 15]. Those approaches address several kinds of misbehavior during route discovery

and forwarding [4]. Though many types of misbehavior during service announcement and discovery are closely related to forwarding, most of the detection mechanisms, e.g. those for detecting packet dropping, packet modification as well as packet fabrication, are not directly applicable to service trading. This is mainly for two reasons:

1. non-forwarding is legal in some cases and should not be punished and

2. misbehavior cannot always be observed directly but can only be detected when also taking messages lying far in the past (message context) into account.

We will provide examples for both categories in Section 2.

In this paper, we propose a reputation scheme for evaluating cooperativeness according to a higher-level protocol in MANETs. It complements protocol-based reputation mechanisms by monitoring techniques, that allow to detect misbehavior by indirectly inferring conclusions about the cooperativeness of devices based on currently and formerly overheard messages (message context) of other devices. Since each network participant has only a partial view of the overall network traffic, those conclusions are associated with a relatively high degree of uncertainty. For that reason, trust building is based on observations of a number of nodes. In addition, we take link availability between the observing and the observed node into account. We propose a lightweight approach in letting each node solely evaluate overheard standard messages, i.e. we do not extend communication between nodes by adding mobility information as positions and past movement vectors to lower level protocols [12]. Since the design of appropriate overhearing techniques is tightly coupled to the underlying service discovery protocol, we exemplarily describe a solution for the Lanes overlay protocol [7] and the incentive mechanisms designed to stimulate cooperation of the Lanes participants [13].

2. THE LANES PROTOCOL

Due to the lack of infrastructure in ad-hoc networks, decentralized approaches for service announcement and discovery are required. The Lanes overlay protocol is an efficient solution that provides those capabilities [7]. As depicted in Fig. 1, nodes are logically organized in lanes. Due to the rules governing the login to and the maintenance of the structures, the Lanes overlay reflects the underlying network structure to some degree, i.e. logical neighbors tend to be physically close, too. While moving, nodes may join or leave lanes according to their location. Services are announced to and stored by all members of the lane containing the service provider. Service discovery is done across lanes via anycast. Thus, the protocol allows to efficiently find all services offered in the network.

Cooperative behavior of the participating nodes is crucial for this approach. However, autonomous devices tend to be uncooperative for several reasons. Particularly, mobile devices might be uncooperative to preserve their scarce resources. At the same time network participants may unintendedly misbehave, e.g. due to low battery power. A third class of misbehavior is closely related to the nodes' role as service providers. Obviously, there is a natural incentive to passively or actively prevent actions that are advantageous for competing providers.

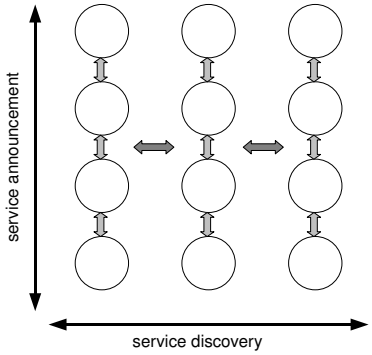

Figure 1: The lanes overlay.

The incentive patterns proposed in [13] stimulate cooperative behavior according to the Lanes protocol. They provide incentives for participating in service announcement and service discovery by introducing remuneration and protocol-based reputation patterns as well as digital signatures. However, those mechanisms cannot cope with all types of observable misbehavior. In the following sections we characterize possible types of misbehavior with respect to the Lanes protocol and demonstrate how to detect them by overhearing message traffic.

3. MISBEHAVIOR

Besides malicious behavior, the main reason for network participants to cheat is motivated by the wish to preserve resources. Different messages of a particular higher layer protocol cause different work loads. The first step for judging on cooperativeness is thus to weight messages according to the workload they impose on a node.

Table 1 lists messages of the Lanes protocol. There are messages for which the receiver is automatically motivated to react accordingly. Not reacting to these messages will cost the defector more resources than a proper reaction. *LoginOffer* and *Ping* are examples for this kind of messages. If a login offer will not be processed by a receiver, he will simply not be able to join the overlay. Not sending ping messages will cause the other nodes to start an error detection protocol, which will either cost the node more resources than sending ping messages or eventually lead to the exclusion of the node. Furthermore, there are messages for which no reaction is required, like *ServiceRequestResult*. All these messages are assigned a weight of 0.

A special case is *LoginRequest*. As it is usually among the first messages a node sends, reactions to this message are hardly observable. It is not feasible to assess, which node should have received the request and thus is expected to react with a *LoginOffer*. Reaction to login requests should therefore be rewarded explicitly. Appropriate remuneration mechanisms have been proposed in [13].

The most interesting messages for evaluating cooperativeness are messages that require the receiver to spend some of its resources to react, i.e. $weight > 0$, and which are usually observable by neighboring nodes. In these cases, overhearing of network traffic is an alternative to protocol-based incen-

tive schemes, which cause additional network load. Into this category fall messages that need to be propagated through the Lane: *ServiceOffer*, *ServiceRevoke*, *ServiceRefresh*, *ServiceRefreshComplete*. Note that forwarding is not needed in case a node is the end of a lane. However, a selfish node may claim to be the end of a lane to be relieved from forwarding. A *ServiceRequest* needs to be forwarded to the neighboring lane and to be answered by a *ServiceRequestResult*. In this context, a misbehaving node might answer those requests negatively without processing them, or it is not able to answer those requests, because of having not stored service descriptions received by others. Those types of misbehavior are not directly observable, but can be detected by overhearing techniques as we will see later on. Our evaluation is based on a simulation model of a MANET.

4. SIMULATION MODEL

MANETs are typically simulated with network simulators [9], which concentrate on the lower layers of the OSI protocol stack, e.g. to evaluate routing protocols on the third layer. With these simulation systems, models for representing movement and service behavior are usually calculated based on stochastic distributions [16]. Network performance in MANETs is sensitive to local accumulation of nodes and temporal accumulation of network traffic [16]. To evaluate the Lanes trading protocol, which was designed as a lightweight overlay mechanism [7], the general purpose modeling and simulation system James II [5] was employed to simulate MANETs containing user models whose motion and service behavior are both based on currently pursued activities [14].

Figure 2 depicts the top-level structure of the MANET model, represented as a coupled PDEVS model. PDEVS (Parallel Discrete EVent System specification) is one of the formal approaches to discrete event modeling and simulation stemming from general systems theory [18]. PDEVS distinguishes between atomic and coupled models. While atomic models define dynamic behavior based on states and transitions, coupled PDEVS models support the hierarchical, modular construction of complex models. From the point of view of the coupled *Manet* model, the *Node*, *Map*, and *Network* models are black boxes, which become coupled according to their interfaces, i.e. according to the type of events they are able to send and receive.

Network nodes move in a spatial environment and announce and request services. The model *Map* represents the geographical environment. It receives movement notifications from nodes and keeps track of all node positions. Based on positions and given radio ranges a connectivity graph is calculated. Changes in connectivity become propagated, such that other models, e.g. the network model, can adapt to these changes. Furthermore, the map model connects nodes via their *Frame* ports according to the physical network structure. As routing behavior is not at the focus of our evaluation, individual nodes are not equipped with a full-fledged routing protocol. Instead, routing capabilities are provided by a centralized model: the *Network*, which manages a routing table based on connectivity information received from the map. The network model receives route requests from nodes and answers these with route replies. Routes are calculated as shortest paths between nodes. In contrast to the *Map* model and *Network* model, which are realized as atomic models, the *Node* model contains itself

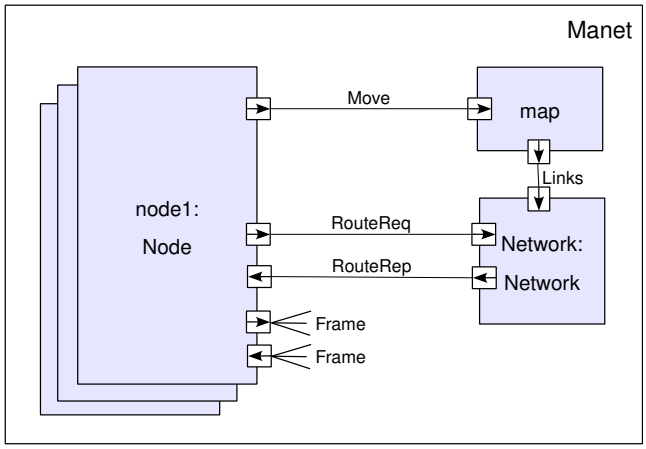

Figure 2: Structure of the MANET model

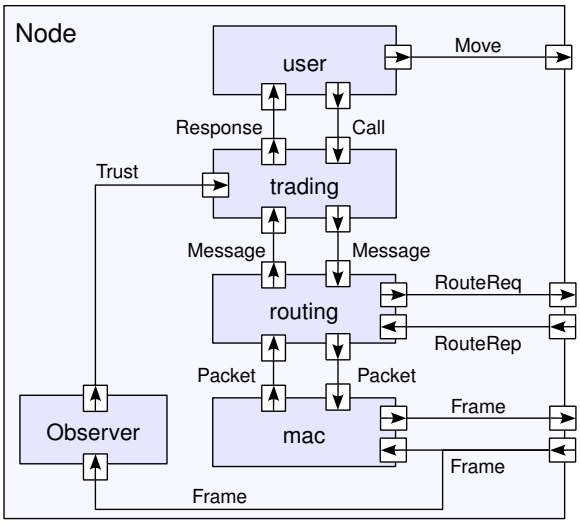

Figure 3: Structure of the node model

sub models and becomes therefore itself a coupled model. Figure 3 visualizes the internal structure of each node. A node contains models of a *User*, a *Trading* protocol, a *Routing* protocol, a *Mac* protocol, and an *Observer*. Within the scope of this paper, nodes use always the Lanes protocol for trading. *User* models initiate communication by sending calls to the trading protocol. A call initiates to offer, to revoke, or to search for a certain service. The trading protocol reacts, after having performed all necessary actions, to calls of the user with according responses. Trading protocols communicate over the network with messages of arbitrary content. Messages are passed to the lower layer, i.e. the network layer, which requests a route from the *Network* model. After having received a route reply, the message is passed down to the *Mac* model. The *Mac* model creates frames and puts them on its output port for frames. Thereby, frames are sent to all nodes that are in transmission range. Frames reaching other nodes are passed to the MAC model, where they are analyzed, whether they have to be received, to be forwarded, or to be dropped. In addition to the processing of frames by the *Mac* model, frames received by a node are automatically passed to the *Observer* model. The ob-

Name	Description	Possible Misbehavior	weight
LoginRequest	indicates wish to join the overlay	not sending a LoginOffer	5
LoginOffer	offer by a Lane member to a node willing to join	-	0
LoginAccept	accept a LoginOffer	not sending a LoginConfirmation	5
LoginConfirmation	member node confirms login	-	0
ServiceOffer	announce service description	Receiver does not forward	2
ServiceRevoke	withdraw a service description	not forwarding	2
LogoffInformation		-	0
RemoveLane	no forwarding needed		3
ServiceRefresh	periodically refresh announced service descriptions	not forwarding	3
ServiceRefreshComplete	Back-propagate available services	not forwarding	3
ServiceRequest	look up a certain service	not forwarding to neighboring lane	3
		not answering	10
ServiceRequestResult	Reaction to ServiceRequest		0

Table 1: Message of the Lanes protocol and potential misbehavior

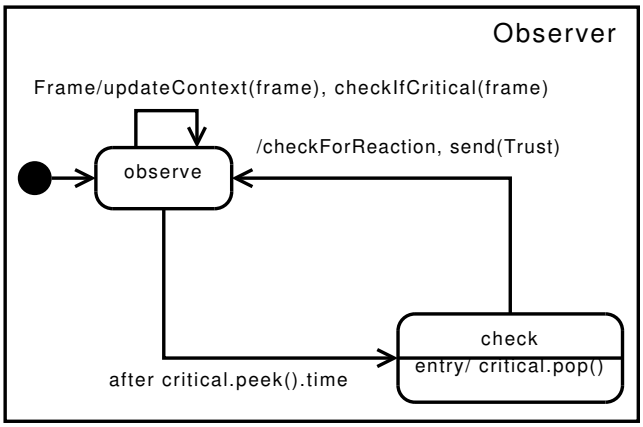

Figure 4: Trust model

server calculates trust values and passes them to the trading protocol, where appropriate actions, e.g. exclusion of an uncooperative node, may be initiated.

5. MODELING THE OBSERVER

Data transmission in wireless networks is broadcast in nature. Nodes in the transmission range of another node receive all sent frames and thereby automatically overhear communication. The model *Observer* does, in contrast to the *Mac* model, not drop communication not intended for its node but evaluates each frame. Figure 4 visualizes the behavior of the *Observer* model as a state chart.

5.1 Collect and Update Context Information

To be able to estimate availability of links, each overheard frame is used to update a node's knowledge about its context. A bidirectional communication between two nodes is assumed, if a content frame becomes acknowledged by the receiver. An observer interprets bidirectional communication as an intact link between two nodes. The *Observer* model stores for each link its message context, i.e. the last overheard frame paired with its transmission time. The basic problem of observability in MANETs stems from the fact that an observer cannot be sure whether he is (still) able to overhear traffic between two nodes. Figure 5 visualizes possible relations between a set of nodes. Suppose the link between node i and node r is subject to observation. Node i represents the initiator of an action and node r the reactor, i.e. i sends a message that — according to the trading protocol — needs to be answered by r. Nodes o_1, o_2, and o_3 represent potential observing nodes. The transmission ranges of node i and node r are drawn hatched. Node o_1 and o_3 lay within the transmission and receiving range of node i. Node r is connected to node i, o_2, and o_3. As node o_1 is connected to i it is able to see all traffic emitted by i. But o_1 is not able to see everything i receives, e.g. messages sent by r. Respectively, o_2 is able to overhear all messages sent by r but o_2 does not see messages sent by i. Concerning the traffic between i and r, only node o_3 is able to overhear the connection bidirectional. As nodes move unpredictably for an observer, node o_3 may leave the crosshatched area at some point in time. If o_3 overhears a message from i to r, which is expected to be answered by r, node o_3 waits for an according reaction by r. If o_3 does not see an expected reaction by r there are two possible explanations. First, node o_3 is not in the position to see messages sent by r anymore, or node r did simply not behave properly.

To decide on the cooperativeness of r comes down to the question, how sure o_3 can be concerning its knowledge about its relation to r. Links are assumed to disappear exponentially. For $t_o(r) \geq 0$ the IID (Independent and Identically-Distributed) link availability between two nodes o and r (from o's perspective) is calculated as

$$availability_o(r) = e^{-t_o(r)/\beta},$$

where $t_o(r)$ is the time passed since the last message between o and r has been observed by o. According to [17] the mean is set to $\beta = 100$ seconds.

5.2 Identify Critical Messages

Which overheard messages to analyze in more depth and to use as a basis for deriving context information partly depends on the higher level protocol to observe. With the Lanes protocol messages indicating a service announcement have to be forwarded by a receiving node, except the node is

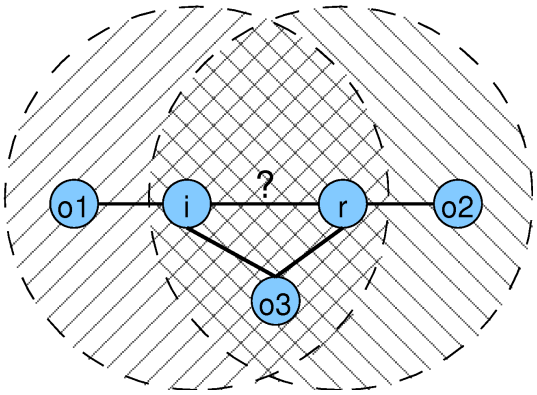

Figure 5: Radio ranges, connectivity, and observability

the start or the end of a lane. Whether a node is the start or the end of the lane can be derived from overheard ping messages. If an observed message is considered to have reached its destination, i.e. the receiver of an overheard frame is the receiver of the contained message, the observer checks whether the message is a critical one or not. Based on the past observations it is checked, whether the frame contains a message that requires a subsequent observable action by the receiving node, e.g. forwarding of a *ServiceOffer* or answering a *ServiceSearch* according to Table 1. If this is the case, the frame is added to a time-ordered queue of critical frames paired with a deadline when an reaction of the node is expected to be observed. Currently the expected max. time window for a reaction is set to 0.5 seconds. Each overhearing node o counts the total number of critical messages for all other nodes $r \neq o$. Each overheard critical message is weighted according to Table 1:

$$total_o(r) = \sum_{c \in Critical(r)} weight(c)$$

5.3 Check for Appropriate Reactions

After the deadline for a reaction to a critical message has passed, the observer has to decide on the reaction with respect to the suspect node. Whether an observer o is able to see an action of another node r depends on the availability of the link between r and o. To not pose additional requirements on other nodes, link availability is evaluated based only on the information a node usually is able to perceive. Thus, no additional mobility information like positions and past movement vectors [12] has to be announced by nodes. This would require additional conventions between nodes and thereby limit the applicability of the approach.

The number of suspicious messages for each observed node r as seen by observing node o is based on the set of missing messages, i.e. it is given by the number of critical messages for which no proper reaction has been observed. Each critical message for which no proper reaction has been observed is added to the set *Missing*. The effect of an observation is derived from the probability by which the action can be observed, i.e. it is based on the estimated availability of the according link:

$$suspicious_o(r) = \sum_{m \in Missing(r)} availability_o(r) * weight(m) \quad (1)$$

Cheating probabilities can now be calculated as the fraction of critical messages that have not been reacted to appropriately.

$$cheatingProb_o(r) = \frac{suspicious_o(r)}{total_o(r)} \quad (2)$$

A node o updates its trust value for an observed node r after the timeout for a critical message has passed and an according behavior has (not) been observed. Thus, observation of expected behavior has a "positive" effect, i.e. it becomes an exculpation. Formally, 0 is added to the number of suspicious messages and *cheatingProb* is updated — having the effect of being decreased.

5.4 Detection of Cheating Nodes

The total number of frames observed by a node $total_o(r)$ indicates the certainty of node o to judge about the cooperativeness of node r.

The overall trust of a node r in a network is based on the observations of all nodes except node r itself. Observations of each node o are weighted according to their certainty about r normalized according to the sum of certainties

$$total(r) = \sum_{i \neq r} total_i(r)$$

such that the overall cheating probability of a node r is given by

$$cheating(r) = \sum_{i \neq r} cheatingProb_i(r) * \frac{total_i(r)}{total(r)} \quad (3)$$

If both $cheatingProb_i(r)$ and $total_i(r)$ are above a certain threshold, an observing node o may initiate a voting to exclude r from the overlay. All nodes having gathered observations about r may then propagate their cheating probabilities and total observation counts. Whether an accused node will really be excluded depends on the value of $cheating(r)$.

6. EXPERIMENTS

For evaluating trust building, user models represent students moving as pedestrians on a campus and performing service operations. All experiments use an activity-based user model. The concept of this user model was described in [8] and the implementation discussed in [14]. Users arrive uniformly distributed over one hour, log into the network and are online until the end of the simulation. Transmission and receiving range of all nodes is set to a constant of 100 meters resulting in always bidirectional connections between nodes. The network is protected from partitioning. For a node without connection to any other node a dummy connection is established until it re-enters the transmission range of another node again.

Cheating behavior is introduced in the simulation model by letting a certain fraction of all nodes behave uncooperatively. Cheating nodes forward messages and answer service requests only with a certain probability, as opposed to cooperative nodes that always react properly. Regarding the

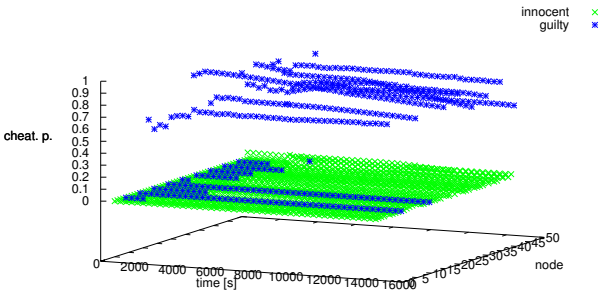

Figure 6: Black hole scenario: development of individual cheating probabilities

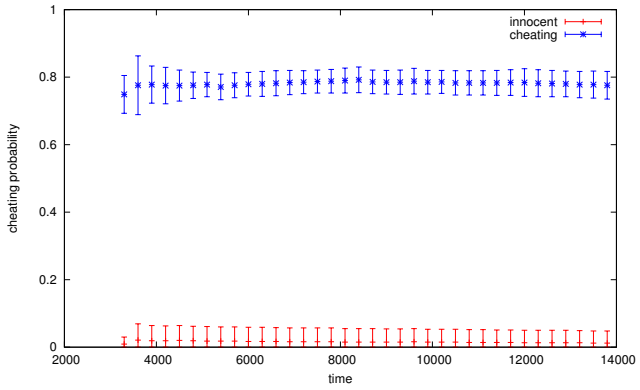

Figure 7: Black hole scenario: development of mean cheating probabilities

observer model, cheating nodes themselves prioritize consumption of resource and therefore spend no resources to observe other nodes and do not participate in voting.

6.1 Black Hole

In a first step, the simulation model is used to identify the simplest form of misbehavior: black hole attack [4]. Cheating nodes do never react to critical messages. Cheating nodes drop 100% of critical messages, i.e. forwarding / answering probability of cheating nodes is set to 0.

Figure 6 shows the development of cheating probabilities for 40 innocent nodes (node numbers $1, 2, 3, 4, 6, \ldots$) and 10 guilty nodes (node numbers $5, 10, \ldots$) during a simulation run. Except at the very start of the simulation cheating probabilities for all innocent nodes are below 0.1. Contrary, guilty nodes are clearly recognized as their cheating probabilities are above 0.7 – except for cheating nodes that are situated at the start or end of a lane. If an uncooperative node is the start or end node of a lane, it does not need to forward messages. These potentially cheating nodes will not be detected. However, as for these nodes, not forwarding is no misbehavior, they simply got no chance to cheat with respect to forwarding. It is only possible to detect these nodes, if they cheat in other cases, e.g. not reacting to a service request. Mean values together with standard deviations for innocent and cheating nodes are depicted in Figure 7. Black hole attacks of nodes are easy to detect as innocent and cheating nodes can be clearly told apart.

6.2 Gray hole

A better strategy for defective nodes than cheating all the time, is to drop (not forwarding, not answering) only a certain portion of critical messages, called gray hole attack [4].

Figure 8 shows the development of mean cheating probabilities for innocent nodes and guilty nodes, where the latter drop critical messages by 50%. While the overall cheating probabilities for guilty nodes are considerably lower than with the black hole attack, cheating nodes can still be clearly detected. The explanation for this is given by Table 2. Simply averaging cheating probabilities as observed by all nodes would only have detected few (node 35 and 40) of the cheating nodes. Many nodes out of the total number of nodes are not able to observe a particular node under question.

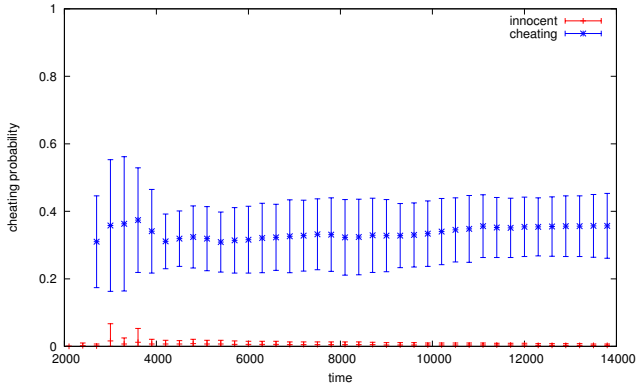

Figure 8: Gray hole scenario: development of mean cheating probabilities)

Observations made by nodes nearby a cheating node will be averaged out by the number of nodes that were not even able to observe the node. This situation changes, if individual observations are weighted by the total number of observed critical messages — as given by Equation 3. All except three cheating nodes are clearly detected. Figure 9 shows the total number of observed critical messages for all nodes. For potentially defective nodes that are not identified (e.g. 10, 45, and 50), there was simply no observation of critical messages. These nodes are at the start or end of a lane and thereby did not behave uncooperatively.

The question remains, how much effort has to be spent for observing and detecting cheating nodes. Table 3 lists the mean number of critical frames that had to be analyzed by observing nodes. The last column shows the mean number of frames overheard by a node. While the number of overheard frames is high, the number of messages to be analyzed in depth is moderate.

7. CONCLUSION

We introduced a monitoring-based reputation scheme for evaluating cooperativeness according to a higher level protocol in MANETs. In a first step we identified critical messages and assigned weights according to the work load they impose on the receiver. In a second step, we explained how

Node	weighted cheat. p.	mean cheat. p.	mean certainty
5	0.438	0.054	1.612
10	0.000	0.000	0.000
15	0.476	0.013	0.240
20	0.248	0.018	0.433
25	0.410	0.021	1.650
30	0.411	0.064	4.566
35	0.408	0.243	7.533
40	0.410	0.225	3.502
45	0.000	0.000	0.000
50	0.000	0.000	0.007

Table 2: Cheating probabilities and certainty values after 2 hours of observation.

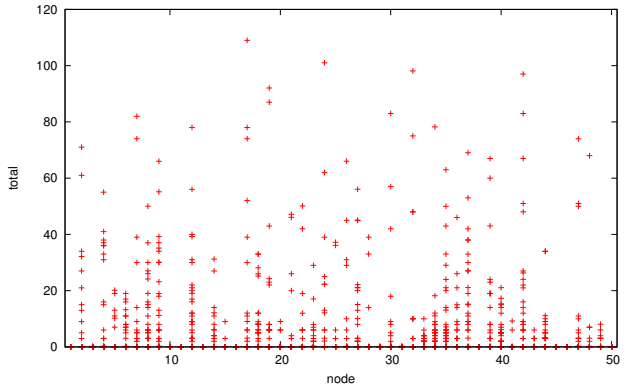

Figure 9: Total number of observed critical messages for each node

nodes	tot. critical	overheard
25	45.42	5968.7
50	74.07	16468.9
75	120.47	35205.9

Table 3: Average effort for 2h overhearing

to detect misbehavior based on overhearing those messages and already received messages (context). Our approach exploits unavoidable one hop broadcasts in MANETs, i.e. we overhear all traffic. In opposition to other approaches it requires no extension to the MAC layer, e.g. additional information like position information. To reduce the inherent uncertainty that is associated with misbehavior detection based on message traffic analysis, we considered not only observations of a single node but also those of others. We also took probabilities for link availability between the observing and the observed node into account.

Our evaluation shows that misbehavior can be effectively detected with our approach. However, many frames are received and have to be processed by an observer increasing its work load significantly. In our future work we will thus quantify the effort for the observation and compare it to the potentially saved effort by identifying defective nodes. Additionally, we will compare the extra effort introduced by our approach to the overhead generated by protocol-based reputation mechanisms like remuneration. Moreover, we will develop and implement a voting algorithm that allows to exclude uncooperative nodes based on their collectively observed behavior. We will also investigate more sophisticated types of misbehavior, including those cases where nodes untruthfully report about (mis-)behavior of other nodes.

8. REFERENCES

[1] S. Buchegger and J.-Y. L. Boudec. Self-policing mobile ad-hoc networks by reputation systems. *IEEE Communications Magazine, Special Topic on Advances in Self-Organizing Networks*, 43(7), 2005.

[2] Sonja Buchegger, Cedric Tissieres, and Jean-Yves Le Boudec. A test-bed for misbehavior detection in mobile ad-hoc networks - how much can watchdogs really do, no. ic2003. Technical report, 2003.

[3] Taco Dijkstra. Stimulating cooperation in ad-hoc networks. Technical Report DTC.5966 Freeband/CACTUS Project, TU-Delft, February 2004.

[4] Oscar F. Gonzalez, Michael P. Howarth, and George Pavlou. Detection of packet forwarding misbehavior in mobile ad-hoc networks. In F. Boavida et al., editor, *Wired/Wireless Internet Communications*, volume 4517 of *LNCS*, pages 302–314. Springer, 2007.

[5] Jan Himmelspach and Mathias Röhl. *JAMES II - Experiences and Interpretations*, chapter 20. Taylor and Francis, 2008 (to appear).

[6] Audun Josang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decis. Support Syst.*, 43(2):618–644, March 2007.

[7] Michael Klein, Birgitta König-Ries, and Philipp Obreiter. Lanes – a lightweight overlay for service discovery in mobile ad hoc networks. In *3rd Workshop on Applications and Services in Wireless Networks (ASWN2003). Berne, Swiss*, July 2003.

[8] Birgitta König-Ries, Michael Klein, and Tobias Breyer. Activity-based user modeling in wireless networks. *Mob. Netw. Appl.*, 11(2):267–277, 2006.

[9] Stuart Kurkowski, Tracy Camp, and Michael Colagrosso. MANET simulation studies: The incredibles. *ACM's Mobile Computing and Communications Review*, 9(4):50–61, 2005.

[10] Sylvie Laniepce, Jacques Demerjian, and Amdjed Mokhtari. Cooperation monitoring issues in ad hoc networks. In *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*, pages 695–700, New York, NY, USA, 2006. ACM.

[11] Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Mobile Computing and Networking*, pages 255–265, 2000.

[12] A. Bruce McDonald and Taieb F. Znati. Statistical estimation of link availability and its impact on routing in wireless ad hoc networks. *Wireless Communications and Mobile Computing*, 4:331–349, 2004.

[13] Philipp Obreiter, Birgitta König-Ries, and Georgios Papadopoulos. Engineering incentive schemes for ad hoc networks - a case study for the lanes overlay. In *First International Workshop on Pervasive Information Management (EDBT-Workshop)*, Heraklion, Greece, 2004.

[14] Mathias Röhl, Birgitta König-Ries, and Adelinde M. Uhrmacher. An experimental frame for evaluating service trading in mobile ad-hoc networks. In *Mobilität und Mobile Informationssysteme (MMS 2007)*, volume 104 of *Lect. Notes Inform.*, pages 37–48, 2007.

[15] Avinash Srinivasan, Joshua Teitelbaum, Huigang Liang, Jie Wu, and Mihaela Cardei. *Reputation and Trust-Based Systems for Ad Hoc and Sensor Networks*. Wiley & Sons, 2008 (to appear).

[16] Desney S Tan, Shuheng Zhou, Jiann-Min Ho, Janak S Mehta, and Hideaki Tanabe. Design and evaluation of an individually simulated mobility model in wireless ad hoc networks. In *Communication Networks and Distributed Systems Modeling and Simulation Conference 2002*, San Antonio, TX, 2002.

[17] Jianxin Wang, Xianman Zhu, and Jianer Chen. Link availability at any time in MANET. In T. Kunz and S.S. Ravi, editors, *Ad-Hoc, Mobile, and Wireless Networks*, volume 4104 of *LNCS*, pages 184–196. Springer, 2006.

[18] Bernard P. Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of Modeling and Simulation*. Academic Press, London, 2nd edition, 2000.