Evaluation of trust building from contextual information in MANETs

Master Thesis

Faculty of Computer Science and Electrical Engineering
University of Rostock

Vorgelegt von: Ulrike Borchardt
Geboren am: 16.09.1982 in Greifswald
Matr.-Nr.: 002202792
Betreuer:
- Dipl. Inf. Mathias Röhl, Universität Rostock
- Dipl. Inf. Friederike Klan, Friedrich Schiller Universität Jena
Gutachter:
- Prof. Dr. rer. nat. habil. Adelinde M. Uhrmacher, Universität Rostock
- Prof. Dr. rer. nat. Clemens Cap, Universität Rostock

Abstract

Mobile ad-hoc networks are composed of nodes, which build up infrastructure autonomously. The cost of maintenance for the network is shared equally between the participants. This principle of cooperation is also required for successful service trading within a mobile network. To protect participants from misbehaviour, a distributed reputation system was introduced in the DIANE project. The trustworthiness is therewith calculated based on the information, that could be obtained from the overlay protocol, Lanes. Within this work the possibilities to gain trust were extended to routing information, to make the reputation system more reliable. To integrate this possibility into the already existing model, ad-hoc routing in the network was required. Furthermore a possibility had to be created to calculate trust values and use them to decide on participants intentions. Finally the founded solutions were evaluated in terms of efficiency.

Zusammenfassung


CR–Classification (1998)::
I.6.8[Simulation and Modelling];Types of Simulation—Discrete event, Distributed, Parallel;

Keywords:
MANET, service trading, simulation, trust building
Contents

1. Motivation 1

2. MANETs 3
   2.1. Fields of application ........................................... 4
   2.2. Protocol stack for MANETs ..................................... 4
      2.2.1. The physical layer ..................................... 5
      2.2.2. The data link layer ..................................... 6
      2.2.3. The network layer ..................................... 7

3. Ad-hoc routing protocols 8
   3.1. General overview ........................................... 8
   3.2. Examples of routing protocols ................................. 10
      3.2.1. AODV ........................................... 10
      3.2.2. DSR ........................................... 12
      3.2.3. DSDV ........................................... 14
      3.2.4. ABR ........................................... 15
      3.2.5. TORA ........................................... 16
      3.2.6. Summary ........................................... 18

4. The DIANE project 20
   4.1. Main intention ........................................... 20
      4.1.1. PDEVS - Parallel Discrete Event System Specification .... 21
      4.1.2. General model ....................................... 23
      4.1.3. Assumptions for the model ................................ 25
   4.2. Lanes - the overlay protocol ................................ 26
      4.2.1. Structure of the Lanes protocol ........................... 27
      4.2.2. Service management .................................... 30

5. Misbehaviour and how to overcome it 33
   5.1. Kinds of misbehaviour ....................................... 33
1. Motivation

Nowadays, computer networks are an inherent part in people’s communication structures [39]. For this reason static computer networks are widely researched and offer a stable infrastructure, as well as a possibility for integration to the user. Besides, this static approach, ad-hoc networks offer a more dynamic way to interact. Especially with regard to the fact that mobile devices became more and more common over the last years, more people are provided with the possibility to use ad-hoc facilities. In addition, these devices experienced an extraordinary rise in processing power and are therefore able to provide more and more sophisticated computer applications.

Devices in mobile ad-hoc networks (MANETs) [1] do not rely on a predefined infrastructure, but build up a network autonomously, where each device takes on a part of the effort necessary to maintain and hold up a network. Therefore it is of high importance that each participating node is reliable and spends a part of its valuable resources, like energy or bandwidth, for network purposes. Nevertheless, it cannot be assumed that there will not be a participant trying to conserve its resources. This selfish behaviour can result in attempts to gain a personal advantage by imposing tasks upon other participants the device should fulfil by itself [3]. However, this behaviour breaks the agreement on cooperation in an ad-hoc network. For this reason and to protect cooperative participants from misbehaviour, such has to be discovered and misbehaving participants have to be penalized. Possible actions include the introduction of rewards for non-selfish behaviour as well as penalties for misbehaviour, e.g. exclusion of a participant from the network in the worst case. Furthermore, this requires identification for each participant to be guaranteed.

Nevertheless, developing a mobile information system is a highly experimental and complex task; hence, testing with the help of simulation is highly common and widely accepted. Within a given framework new strategies can be easily explored and refined before applied to the real world.

It is the main intention of this work to reveal uncooperative behaviour based on routing information and enable the nodes to draw necessary conclusions for uncooperative participants. The challenging aspect lies within the fact that not every request, which is unanswered is uncooperative behaviour, but might occur due to changes in the topology or the loss of energy resources.
1. Motivation

There are several ways of misbehaviour in MANET’s [3]. It can be differentiated between intended and unintended attempts of cheating. Possible reasons causing unintended cheating might be the loss of connection to the network and the running out of battery power. Intended cheating instead is, e.g. the cancellation of packets asking for service provision by a participant, which is only using the network to its advantage. Those participants are to be excluded since they do not share the common costs of the network, i.e. the resources needed to maintain the connectivity. They are using resources of other participants while saving their own resources for their own purposes.

The main intention of this thesis is to find an approach to reveal open trust issues in the existing implementation of the DIANE project and afterwards evaluate the solution found. These issues should be solved by the creation of trust based on routing information. Before handling the integration of routing into the existing model, in chapter 8.2 the basic scenario of a MANET [1] with regard to protocols and communication on the lower layers of the OSI reference model [27] will be explained. Afterwards, chapter 3 will give an introduction into several existing routing protocols for ad-hoc networks. Each protocol will be described with its mechanisms. These routing protocols are explained as they present the general idea of routing in MANETs needed to create routing information. Following chapter 4 will present the DIANE project in which this thesis is settled. This chapter gives details on the system and model used for simulation of a MANET at the University of Rostock. Furthermore, it introduces the overlay protocol, Lanes, which delivers the context information used for the recent reputation system. Thereafter in chapter 5, specific scenarios, which are to be solved will be described, as well as incentive patterns. Incentive schemes are used to stimulate regular cooperation, and offer possibilities to solve the regarded trust issues. The actual approaches used to solve them will be described in detail with chapter 6. Subsequently, in chapter 7 an explanation of the found solution will be outlined and an evaluation, chapter 8 with regard to the correctness to the problem will be given.
2. MANETs

This chapter will present the general ideas of MANET. This will be especially done with regard to the functionalities, which are provided by the lower layers of the OSI reference model [27], as they should be used for the later on creation of context.

MANET is the abbreviation for mobile ad-hoc network [1], [39]. In contrast to common solutions it works without a fixed infrastructure. While WLAN still depends on access points and relies on a partly given infrastructure represented by access points where devices can log in, an ad-hoc network device will not rely on such facilities but will establish a connection to its neighbour devices on its own. Therefore, it is necessary that mobile devices that are ”transitively” in reach of radio communication and connect without configuration or maintenance to form a network [21]. This means, that nodes does not need to be in the direct transmission range of another node, but a connection over the ranges of several other nodes is necessary to establish communication.

According to [13] a MANET can be described as follows:
A mobile ad-hoc network is a collection of mobile platforms or devices, each one able to move free and independently from the others. In addition it can be recorded that it forms a packet based radio communication network, which in contrast to other networks of this kind totally sets aside an infrastructure and is able to organize itself autonomously.

Hence, every device in an ad-hoc network fulfills following characteristics [12]:

- mobile
- autonomous
- restricted in resources (energy, bandwidth)
- established with a wireless communication device
- restricted in reach
- able to route and therefore employ a routing protocol
2. MANETs

2.1. Fields of application

Potentially every network application can be found within a MANET, since MANETs live on the exchange of services between the devices. Subsequently, it is possible that as soon as a device is offering access to the Internet every other participating device can use this service, and therewith further services that rely on the Internet [1]. Yet one of the classic fields of application is conferencing. Since mobile devices enable workers to go out of their offices they leave the classic office network infrastructure, but still have the need for communication with other workers. Consequently, it is useful that mobile devices have the possibility to build up a network on their own without further support or maintenance. Nevertheless, this field still bears a lot of potential. This results from the fact, that most techniques are build on the old internet technologies and therefore are not optimized to sufficiently support ad-hoc networks, e.g. Mobile IP. Mobility is not only of interest for business conferencing but also for home networking. For example it might be of interest to take a laptop from the office home to do some work later in the evening. Nevertheless the laptop needs support and internet access and therefore may not be permanently subscribed to a network by its IP address, since this means higher maintenance. Another idea might rely on the fact, that ad-hoc networks do not need a static infrastructure [13]. They can therefore be used whenever static infrastructure may fail, as for example in cases of emergency due to natural catastrophes. Consequently, mobile devices may be a solution to hold up services until the fixed network is brought up again. They even provide a possibility to offer services in regions where no fixed infrastructure can be established like refugees camps or war zones.

2.2. Protocol stack for MANETs

Like most network types known nowadays, MANETs represent communication systems enabling the participants to communicate with each other. Networks therefore provide connections for communications, which in the case of mobile devices will be wireless. However, wireless is only a part of the system, whose design is affected by the three lower layers of the OSI reference model: the physical layer, the data-link layer and the network layer [2]. The physical layer allows the transmission of bits between two participants of a network. It provides the mechanism to modulate and demodulate electronic waves, which are used for wireless transmission. Nevertheless, wireless links established this way tend to be unreliable, leading to errors in the transmitted signals. Therefore, the data-link layer will add error correction or detection on top of the physical layer. In addition it defines how the transmis-
sion medium is shared between the participants. For wireless communication over the radio spectrum it is necessary to prevent interference with other devices; this process is referred to as multiple-access.

Until this point in the layered structure of the OSI reference model a one-to-one communication between two directly connected devices can be established. Yet a network consists of more than two directly connected devices and will seldomly be a full mesh, where all nodes are directly connected to each other. It is the task of the network layer to establish communication over several intermediary devices. Therefore, it has to determine the way information takes towards a given destination. This process is called routing. Furthermore, the third layer takes care of the quality of service and flow control, which means it tries to establish a congestion-free network. Nevertheless, it also needs layer 4, the transport layer, to provide the upper layer with a uniform access to the communication network, on which they can depend and work independently from. This layer accomplishes the actual end-to-end connection within a communication of two network participants [23], [2].

Following, short explanations on the protocols and techniques used in the physical and data link layer for the wireless communication will be given. The network layer will be covered in chapter 2.2.3, its routing protocols in more detail in chapter 3.

### 2.2.1. The physical layer

Principally the physical layer consists of the following three components as shown in figure [2.1].

![Physical layer components](image)

Figure 2.1.: Physical layer components - from [2]

There always is a transmitter, a channel and a receiver in a network on the physical layer [2]. Yet these components show different properties in a wireless system compared to a wired one.

The main functionality of the transmitter is to take the signal carrying the information and modify it in such way it can be transmitted over the channel. In a wireless system the signal must suit the limited transmission medium resources like the radio spectrum. At the same time the transmitter has to be equipped with modulation techniques that are robust and power-efficient since the network’s mobile devices are limited in battery power. Furthermore,
it already has to take care of possible interferences to avoid them. The channel in a wireless environment will simply be air, yet some characteristics can be denoted for it as well. Firstly, a channel can suffer from distortion by the interference of signals. Secondly, the property of the channel may vary due to the fact that nodes are moving or the propagation way changes. For instance barriers like buildings would change the conditions of transmission. Thirdly, air is not a dedicated medium and will be used for transmission by several devices at the same time, even other kinds of devices than involved in the actual network can interfere the radio spectrum. And finally, there is noise caused by the receiving node, because the device is working electronically and disturbs the original signal.

According to the nature of the channel the receiver will take the resulting signal and produce an estimation of the original information-bearing signal. It can only be an approximation, since compensation for interferences in the channel can never be fully guaranteed. Furthermore, the receiver is already adding some kind of error-correction to the unreliable wireless channel and it has to take care of synchronization, which is especially difficult due to the fact that the channel conditions may vary often. Altogether, it can be summarized that wireless transmission is rather a challenging process compared to wired transmission, since it holds more system inherent problems.

The standards of this layer are V.24, V.28 or X.21 [40]. They regulate the data transmission over the medium and therewith provide a general basis for the creation of networks.

2.2.2. The data link layer

Controlling multiple accesses of the media is one responsibility of the data link layer. Since MANETs are based on wireless communication they will mainly rely on the radio spectrum. Nevertheless, the spectrum is finite and only a fixed range is assigned for the use in wireless networks. This range has to be shared, which can be done by using the following strategies: FDMA, TDMA, CDMA and SDMA. It is the main objective of all of these strategies to utilize the given range as efficient as possible. To optimize their effects, these techniques are often combined in real systems [37].

FDMA stands for frequency division multiple access and dedicates parts of the range to a user either permanently or temporarily. In contrast, TDMA (time division multiple access) permits the user to make use of the whole range of the radio spectrum. However, to avoid conflicts the user will only be allowed to transmit in a dedicated time slot. CDMA (code division multiple access) assigns an individual encoding for each user, which is used
2. MANETs

to differentiate signals spread. Finally, SDMA (space division multiple access) relies on the best direct connection to another node which is realized with directional antennas minimizing interference in the channel.

Protocols defined for WLAN as used in MANETs on the data link layer are summarized in the 802.11 group of the IEEE [38].

2.2.3. The network layer

The network layer is responsible for the flow control and routing of packets over more than two nodes. Therefore it is necessary to establish a logical addressing scheme in which each possible destination can be identified. This addressing scheme can be realized for instance by using IP addresses. In addition, the layer assumes the task of path determination for the packets to be delivered. This process, called routing, involves different protocols and approaches and its protocols are explained in chapter 3. Due to the wide variety of mobile ad-hoc networks lots of different protocols are established. Besides routing, layer 3 of the OSI reference model is also responsible for error correction and data fragmentation, to divide the data into packet size.

Protocols of this layer are IP and ICMP. Both protocols work to realize a consistent addressing scheme and above a possibility to exchange messages, which contain diagnostic information. Furthermore, all routing protocols [23], several of them explained in detail in chapter 3.
3. Ad-hoc routing protocols

With regard to the fact, that contextual information is to be gained on routing information, it is necessary to integrate routing into the simulation model. Subsequently a mechanism to create routes is required. This chapter provides a general overview of routing protocols used in mobile ad-hoc networks. Chapter 6, will deliver more specific details used to make a choice, which routing protocol suites the simulation model used in this thesis best. Accordingly, the routing protocols are introduced in terms of special features, mechanisms used for maintenance of the network, support for and scalability with regard to the network size. Yet it has to be pointed out, that this is only a small subset of all existing routing protocols in use in ad-hoc networks [41].

3.1. General overview

Each node in an ad-hoc network volunteers to carry traffic and participates in the formation of the network topology. This is quite similar to the way that intermediate nodes within the Internet, or within a corporate intranet, cooperate to form a routing infrastructure. Routing protocols within a network provide the information necessary for each node to forward packets to the next hop along the way from a source towards a given destination. Routing protocols are self-starting, adapt to changing network conditions, and almost by definition offer multi-hop paths across a network from a source to a destination. This description underlines, why employing a routing protocol is desirable to manage topology changes in an ad-hoc network [8].

For ad-hoc networks a great variety of routing protocols adapted to different specific characteristics of the environment are available. To structure the variety of existing protocols for ad-hoc networks following approaches exist [12], [13]. According to figure [3.1], a categorisation can be done with regard to the used localisation information, the kind of route determination and maintenance, the location where the route is determined, and the structure of an ad-hoc network. Most common is the categorisation based on route determination and maintenance. According to this category, there are three different kinds of networking protocols: there are proactive, reactive and hybrid protocols.
Reactive, or on-demand, protocols establish a way through the network only if a participant is asking for a connection to a service or another device. In contrast, proactive protocols establish ways through the network as soon as a participant is entering it. These protocols maintain an infrastructure of the network on a regularly timed basis. As a consequence more resources are consumed. Nevertheless there is less effort for establishing a connection when a participant is asking for a special service. Hybrid protocols implement a combination of reactive and proactive protocols. How this combination is accomplished depends on the individual protocol.

Yet there are further ways of categorization possible. With regard to the kind of quality of used localisation information, there are on the one hand protocols, whose routing is based primarily on neighbourhood information, the topology. On the other hand, there exist proto-
3. Ad-hoc routing protocols

cols referring to decided fixed points and hence allow a relative position towards them as well as absolute positions. These protocols are so called position based protocols. The determination of fixed points is for example possible by using a GPS system. A main disadvantage to this system is its inaccuracy. Therefore it is only suitable for networks on the outside able to bridge these inaccuracies.
Moreover, there are source and destination based routing. Source and destination are referred to as the points in the network specifying the way of a packet through the network. The difference between these protocols is that in the case of source based routing, the source is determining the complete way of a packet through the networks whereas in the case of destination based routing the packet’s way is only determined hop by hop.
Finally ad-hoc protocols can be categorized by the structure of the network they create. On the one hand there are flat networks, where every node has to be able to fulfil all arising tasks and all nodes are equal. On the other hand, there are hierarchical networks, where the networks is divided into single domains or clusters, where one or several head nodes are taking over the routing task for the domain. This implies higher workload for the head node, yet less traffic for routing tasks in the domain, which is especially desirable for bigger networks, containing nodes that can be dedicated for this task.

3.2. Examples of routing protocols

Before making a final decision, which routing protocol to choose for the needed realization of context, several protocols will be introduced in this paragraph:

- AODV
- DSR
- DSDV
- ABR
- TORA

3.2.1. AODV

The Ad-hoc On Demand Distance Vector (AODV) routing algorithm is a routing protocol designed for mobile ad-hoc networks and other wireless networks [25]. AODV is capable of
3. Ad-hoc routing protocols

both unicast and multicast routing, but needs a special extension for anycast\(^1\) routing [36]. It is an on demand, reactive algorithm, meaning that it determines routes between participants only if requested by source nodes. AODV is, as the name indicates, a distance-vector routing protocol. It maintains its routes only as long as they are used by the sources. Additionally, AODV forms trees which connect multicast group members. The trees are composed of the group members and intermediary devices needed to connect the groups. AODV uses sequence numbers to ensure the freshness of routes. It is loop-free, self-starting, and scales to large numbers of mobile nodes.

As AODV is meant to work reactive, there is no network traffic as long as there is no need for a connection. As soon as a connection is required, AODV establishes routes by starting a connection broadcast to request a path towards the destination. This request is called route request (RREQ). Nodes receiving this broadcast packet will update their routing table information on the sending node like IP address, current sequence number and broadcast ID. In case the receiving node is neither the destination nor knows a path towards it, it will forward the RREQ. In the other case it will send a packet containing a route reply (RREP). Nevertheless, a RREP is only provided if its sequence number is not lower than the one in the RREQ. As long as the RREQ is broadcasted it may be that a broadcast is received by a node more than once. In this case the node will re-recognize the request by its sequence number and will discard it instead of forwarding it.

In case a RREP is sent it is propagated back to its requesting source. Nodes along the path back will set up forwarding pointers to the destination. As soon as the source node receives the RREP, it can start transferring data packets over the network towards the destination. This will be done by sending the packet to the next node along the path via unicast, where the next address is found in the routing table and the packet is forwarded again. This route announced with the first RREP is kept as long as there is no RREP containing a greater sequence number or the same sequence number with a smaller hop count. In both cases the old route would be banished and the new one would be used for the transfer of data packets. The route will be considered active as long as there are data packets travelling along it. This is important as only active routes will be maintained. As soon as the source node stops sending data packets the link will time out and will be deleted from the routing tables of the intermediary nodes, if it is not needed for another active route.

\(^1\) unicasting: packet for one destination; multicasting: packet for a group - sent to the closest member of the group and further propagated to all group members; anycast: packet is sent to the closest member of the group - needs no further propagation
Whenever a link breaks while a route is still active, the maintenance process takes care of the problem. The node closest to the broken link on the sending side will send a route error (RERR). The RERR will be propagated back to the source, which in case it still needs a route to the destination will start a new route discovery.

The routes for multicast are set up similarly to the ones for unicast. Hence a node which wants to participate in a multicast group broadcasts a RREQ including the destination address of the desired multicast group. In addition, the “Join” flag in the RREQ is set to indicate that this RREQ is a request to join the group. Subsequently the RREQ is forwarded until it is received by a member of the multicast group. Such a member will then send a RREP if its sequence number is high enough. This RREP will afterwards be propagated back towards the source and each intermediary node along the path will add the information, that the sender has joined the group, to its multicast routing table. When the source receives the RREPs from the multicast group members it will keep track of the route with the freshest sequence number and the smallest hop count. After a specified listening period the initiating device will decide, which route to choose and sends a unicast packet including a multicast activation message (MACT) to the group member it selected. This process activates the route. Every other device in the multicast group which has sent a RREP but does not receive a MACT will delete the pointer towards the source node in its routing table after a given time period. The participants of a multicast group are organized as trees and must therefore be maintained on a regular basis as nodes are moving and links are breaking.

A certain advantage of AODV is the fact it does not cause extra traffic for communication along links that are already established. In addition distance vector routing can be considered rather simple and does not require much memory or calculation time. Nevertheless the AODV protocol has a higher traffic emerging for initial connection establishment. Yet as long as there is no connection needed neighbours will not have to contact each other.

3.2.2. DSR

DSR stands for Dynamic Source Routing [1]. It is a routing protocol designed for wireless mesh networks, meaning networks without a hierarchy, where every device in the network is directly connected to every other device. It is based on mainly two mechanisms: route discovery and route maintenance. Together these mechanisms allow all nodes to establish source routes to arbitrary destinations in an ad-hoc network [12].

All aspects of the protocol work entirely on demand, allowing the routing packet overhead
to scale automatically to what is needed for reaction on changes of routes currently in use. This approach of DSR, behaves similar to AODV, as routes are only established if needed, whenever a device needs a concrete connection to another. Furthermore DSR is optimized to work without routing tables even though it is a routing protocol. Instead of storing all information in a table each packet to be sent contains a list with all destination addresses. Even though this appears to be very complex, it reduces the necessity of forwarding devices to store and maintain routing tables. This in turn reduces the transmission of routing data and requires less hardware performance allowing participating nodes to be structured simpler. In addition storage space for routing tables becomes less important.

Nevertheless participants need information about the network structure. This is retrieved by listening to the local network traffic. By overhearing transferred packets a device gets to know addresses of other participants in the network. Furthermore the node remembers routing activities, like requests and failures, as well as information on other nodes to be used later on.

Usually a sending device can retrieve a suitable route by searching its Route Cache. Nevertheless there might be destinations to which the source does not have a route to. To initiate a route discovery the source sends a local broadcast packet containing a Route Request. This Route Request will be forwarded as broadcast by each device it is received by until it reaches the destination. Each device along the way will add its address to the record in the message. As soon as the Route Request is received by the destination it will return this record within a Route Reply message to the initiating source, which subsequently stores this route in its Route Cache. To speed up the process a device which receives the Route Request and knows a route to the destination will route the request to the destination, so it can answer with a Route Reply.

Route maintenance is necessary for the sending node to detect a link along the way towards the destination, that is no longer working. Consequently the source route is not suitable anymore and can be discarded. When route maintenance indicates that a route is no longer available the source device can try another route or has, in case no other routes are known, to invoke a route discovery. The route maintenance checks whether each packet was delivered successfully or not. In case it was not it will send a Route Error message back to the source of the packet.

With regard to performance issues it has to be noted that in small, less frequented radio transmission networks DSR performs the same way as AODV. Yet with raising traffic load DSR only causes a third of the traffic AODV does. This is possible since information can be gained by overhearing and does not have to be requested explicitly. However overhearing
3. Ad-hoc routing protocols

causes a lot of internal data for a node requiring efficient algorithms to keep records as lean and up to date as possible.

In addition, DSR has no built-in anycast support since it relies on finding a route to an explicit destination, not to the member of the group closest to it. Therefore it is designed to allow unidirectional links and asymmetric routes.

3.2.3. DSDV

The Destination-Sequenced Distance Vector (DSDV) [8] routing protocol is a proactive routing algorithm based on routing tables maintained in each node in the network. Since DSDV is an extension to the Bellman-Ford algorithm it works as follows [41].

Each participant of the network runs a routing table where it denotes which destinations are available at the moment, as well as the hop count towards them. Nevertheless a node does not know the exact way a packet will take through the network. The packet will be unicasted only to the neighbouring node with least costs in terms of a certain destination. When the packet is received by the neighbouring node the next node along the way will be detected according to the costs to the destination node. This protocol calculates the way through the network step by step instead of using predetermination by the source. Since each node should be able to communicate with its neighbours the links established by DSDV are bidirectional. For maintenance of these bidirectional links every node in the network has to send periodical updates on its information. DSDV therefore differentiates between two kinds of updates. On the one hand, are incremental updates, on the other hand full updates. Full updates contain the complete routing table and consequently, need several packets to be spread over the network. For this reason this kind of update should be send less frequently and only during periods of low network traffic. Yet, in the meantime the network topology will not stop changing especially with regard to the fact that it contains mobile devices. To keep the network working such changes must be propagated by using an incremental update. These update packages only contain information on what has changed recently and should fit into a single packet to be transferred over the network. The size of the packet matches the one of a network protocol data unit (NPDU). This size also influences the point in time the next full update is done, since this is necessary whenever more changes are to be propagated than fit into a NPDU.

Until now the methods of DSDV do not differ from the ones used for standard distance vector protocols. The extension of DSDV is similar to a main characteristic of AODV: the sequence number. It indicates the freshness of a message. Therefore the sequence number is increased
by each node it traverses. To differentiate the sequence numbers each node has to make its sequence numbers unique. This is achieved by containing the device number in the sequence number. It also keeps the routes loop-free, since only information on nodes must be updated, where the sequence number of a route increased or the hop count towards the destination with the same sequence number decreased.

With regard to advantages of DSDV it has to be remarked that routes are available at any time and there is a fast automatic adoption to changes in the network. Yet there is a high price to be paid: nodes and their resources suffer from high emerging traffic even without real data being transferred. In addition a lot of storage space is needed as the routing tables grow fast and very large containing even routes that will never be used.

3.2.4. ABR

ABR stands for Associativity-Based Routing [12], [41]. It is a flat, topology-based, reactive routing protocol, which routes destination-based. The special characteristic about ABR is that it defines a new routing metric for ad-hoc networks. This metric represents the degree of how stable a connection is, which is mapped onto the degree of association stability. ABR chooses its routes according to this metric.

To calculate the association stability every participant has to maintain an association table for its neighbours. To deliver data used for this table every mobile node periodically broadcasts beacon packets to indicate its availability. For every beacon packet a node receives from its neighbour it increments the association counter in the association table entry for the sending node. A high counter number therefore marks a long undisturbed connection and therewith a high association stability. Yet as soon as the connection between two nodes breaks their counters for each other will be set back to zero. If the devices later build up a new connection, the counting starts again.

For route discovery ABR makes use of the same mechanism as the routing protocol DSR, where a RREQ packet stores all the addresses of the intermediary nodes. The difference in the route discovery process again lies in the metric by which ABR chooses its routes. Instead of hop count ABR relies on its degree of association stability. Only if it has to choose between two routes of the same degree, ABR will use the lower hop count to determine, which route to select. However, when it comes to the transmission of data, ABR uses a destination based routing approach, where each device in the network only forwards to the next hop along the way and packets do not carry the whole route in their header.
With regard to route repair ABR has its own mechanism. A broken link can easily be detected by all its neighbours as the beacon packets will be missing. The neighbour closest to the broken link on the sending side will try to repair the link by starting a local repair process. To achieve this, the node broadcasts a route repair as a local query to its neighbours. To keep the broadcast local its TTL (time to live) will be set to a low value, for example two. If this way the participant on the other side of the link can be reached, it will answer and the broken link can be replaced. In case the participant closest to the link on the sending side fails to find a new path to the link destination, the next participant up the path to the source will try the same process. Yet not every participant will initiate this process - only devices up to the middle of the broken path do, afterwards the source will be informed to start a new route discovery. This way the local query is also used to detect a moving device and check whether it is still reachable. In case it receives the query it will answer with the best new partial route.

The main advantage of ABR lies in the fact that it prefers stable routes. This means the probability of a broken link is reduced, which also decreases the necessity of flooding what consequently conserves bandwidth. In addition ABR first tries to repair a link locally and does not start a new route discovery process at once. Yet it has to be noted, that ABR also tends to choose longer paths if they are more stable. This preference can result in longer transmission times. Furthermore delay can be caused due to the fact that local query broadcast is used for route repair, even though a completely new route is needed.

3.2.5. TORA

TORA is the abbreviation for Temporary Ordered Routing Algorithm [22], [33]. It is a route developing unicast based protocol, which means a route needs to be verified before real data can be transmitted. If there is a route it will be stored as directed acyclic graph (DAG). Every potential destination will have a DAG to be stored by the source node. The edges of a DAG are directed and therefore allow traffic only in one direction. For this reason, they determine the direction of the messages to be routed. Consider the following DAG:

It contains all possible routes from source A to a destination G shown by directed edges. Yet in case one node along the way looses all its outgoing edges the DAG has to be corrected, since the node can no longer forward traffic in the desired direction. TORA verifies the route with the help of link reversal. This means if for instance D would loose its outgoing edge it would have no possibility to route to F. Therefore it starts to reverse the direction of all its incoming edges and turns them into outgoing edges. Since now C has only incoming edges
it will start to reverse its links as well. Yet it will not reverse the link that D has already changed, since only links can be reversed that were not changed before in the current process of link reversal. For this reason C can only change the direction of its connection to B. After this step, B will have to reverse its links that have not been reversed before and changes the direction of the connection towards A. Since A now has one incoming and one outgoing link it will not further reverse and the DAG is stable again. The final DAG will therefore look like figure [3.3 b)].

Originally TORA is a unicast routing protocol, nevertheless for its extension GeoTORA anycasting had to be enabled. Therefore it must be possible to group nodes with regard to different criteria. In the process of routing it will then be sufficient to contact the closest member of the group. Since it is only necessary to be able to send to one member of the anycast group it is sufficient to maintain one DAG for one group. In contrast to common
3. Ad-hoc routing protocols

anycast groups, the grouping in GeoTORA is defined by geographical criteria instead of by its members. By establishing this geocast regions, an explicit login and logout of a group is no longer necessary. A device joins the group automatically as soon as it crosses the borders of the geocast region and leaves it as it moves out of the region. To achieve this geographical coordinates are absolutely necessary and can be gained via GPS. Nevertheless GPS for common use leaves inaccuracies, what makes GeoTORA unacceptable for precise indoor use.

To retrieve routes in the first place and later on maintain them TORA as well as GeoTORA rely on a height metric. Therefore each device in a network contains a quintuple \((t, oid, r, h, i)\) with

- \(t\): logical time of a link failure
- \(oid\): unique node id that defines the reference level
- \(r\): reflection indicator bit
- \(h\): propagation ordering parameter
- \(i\): unique id of the node

All nodes can be ordered lexicographically according to this metric. Yet the main element of this metric to achieve this is \(h\), whereas \(t\), \(oid\), \(r\) are mainly used for failure discovery and \(i\) for the identification of the device. The value of \(h\) determines the direction of the link between two network participants. Generally a link is always directed from the higher level of \(h\) to the lower level. This is also the case, if the assumption, that 0 is lower than NULL is included. Accordingly the routes are build and maintained. For further details see [22].

TORA guarantees to be loop-free at any moment and establishes routes quickly, anyhow, the optimization is of less importance. As it is a demand driven protocol TORA is also designed to provide minimized communication overhead.

3.2.6. Summary

With regard to the variety of routing protocols developed for ad-hoc networks, it has to be pointed out, that it is a field of research where new routing protocols still evolve into the market. This makes it difficult to gain a general overview without getting lost in the specific details of a protocol. However, the variety of protocols evolved due to the variety of conditions under which
3. Ad-hoc routing protocols

MANETs can be used, since they are much more flexible than common static networks. Yet a simplification is desirable to ease the establishment of MANETs at all. It could be speculated that many individual solutions will not take hold a longer time period, when the community which is interested in them is too small. Some protocols might disappear in the course of time, yet a combined effort may organize this field instead of splitting it further.
4. The DIANE project

This chapter gives a brief overview on the DIANE project, its backgrounds, structure and the used models for the simulation, on which this work is based. It includes the current state of the art in the project and the points where this thesis ties in.

4.1. Main intention

Originally initiated at the University of Karlsruhe the main intention of the DIANE project has been defined as follows:

"It is the main intention of the DIANE project to make the distributed resources in ad-hoc networks usable. Based on the service-oriented paradigm the work in this project assumes that resources are utilized in terms of services. Subsequently mechanisms are needed to use these services efficiently and effectively." [15]

In the context of this definition DIANE stands for the German abbreviation of: Dienste in Ad-Hoc-Netzwerken (Services in ad-hoc-networks). Since it is difficult to establish ideas in a real system from the start, DIANE offers an experimental framework to develop and evaluate concepts for the efficient provision and usage of services in MANETs. The major challenge of the framework lies in the fact, that devices restricted in their resources might need to cooperate to fulfil more complex tasks [5]. Nevertheless, it has to be noted that DIANE offers a simulation framework, where a device is represented as a node, since a system with real devices is too elaborate and expensive for testing. Hence simulation always means that a real system is mapped onto a model, which always includes the process of abstraction, where real systems are restricted to their main attributes. The specific modelling formalism and according abstractions will be explained in the following sections.

Even though a whole framework is provided by the project, research at the university of Rostock is based on one model used within this framework. The model and its assumptions are described beginning with chapter 4.1.2. Further models are possible and researched at the University of Jena, where the DIANE project is situated and overseen now. they will not
be subject of this thesis.

### 4.1.1. PDEVS - Parallel Discrete Event System Specification

The DIANE project models its parts with the help of the PDEVS formalism [19]. The PDEVS formalism consists of following parts shown by this signature:

\[ PDEVS = \langle X, Y, S, \text{ta}, \delta_{\text{int}}, \delta_{\text{ext}}, \delta_{\text{con}}, \lambda \rangle \]

- **X**: set of input events
- **Y**: set of output events
- **S**: set of sequential states
- **\( \text{ta} \):** \( S \rightarrow \mathbb{R}^+_0, \infty \) time advance function, associates a time period with each state, it is existing before an internal state transition takes place
- **\( \delta_{\text{int}} \):** \( S \rightarrow S \) internal transition function
- **\( \delta_{\text{ext}} \):** \( Q \ast X^b \rightarrow S \) is the external transition function where \( Q = \{(s,e) | s \in S, 0 \leq e \leq \text{ta}(s)\} \) and \( X^b \) is a set of bags over elements in \( X \)
- **\( \delta_{\text{con}} \):** \( Q \ast X^b \rightarrow S \) is the confluent transition function, subject to \( \delta_{\text{con}} (s, \phi) = \delta_{\text{int}} (s) \)
- **\( \lambda \):** \( S \rightarrow Y \) output function

This tuple represents a single atomic model which can be depicted as shown in figure [4.1].

![Atomic model in PDEVS](image)

**Figure 4.1.: Atomic model in PDEVS**

The big box represents the actual model, the smaller ones including the arrow attached to it, stand for the ports of the model. The port on the left side of the model allows incoming events to enter the model. All possible events of \( X \) can enter this model as a bag. Subsequently, multiple occurrences of the same element are allowed. The port on the right side tags the exit point for events of the model; all possible events of the set \( Y \) can occur here.
Within the model four functions cover the possible actions. These functions depend on the actual state of the model, $S$, and events, which either come from the outside as a bag of $X$ or from the inside as a result of one of the functions. The $\lambda$ function models the output, which is is a bag of $Y$ and results from the state of the model. In contrast, $\delta_{int}$ causes internal state transitions based on the elapsed time. The $\delta_{ext}$ function calculates state transitions based on the incoming events, a bag of of elements of $X$. To resolve possible conflict when an external event and an internal state transition occur at the same time, namely when the elapsed time equals $ta(s)$, $\delta_{con}$ enables the individual to decide at the time this occurs.

The DIANE project is modelled based on the PDEVS formalism and therefore each extension made must be modelled accordingly. The description above depicts a simple model consisting of one atomic component. Yet as DIANE should provide a framework to test different models for users and environment, it is a coupled model, where different atomic models are put together. The mathematical description for the coupled model looks as follows:

$$\text{DN} = < X, Y, D, \{M_i\}, \{I_i\}, \{Z_{i,j}\} > \ [19]$$

- $X$ : input set
- $Y$ : output set
- $D$ : index set for the components of the coupled model
- $M_i$: one component and PDEVS model for each $i \in D$
- $I_i$: set of influencers of $i$ for each $d \in D \cup self$
- $Z_{i,j}$: output translation function for each $i, j \in D \cup self$

According to this formalism, the individual models are combined into the modelling framework of the DIANE project. An example of a coupled model can be seen in figure [4.2].

Each of the labelled components represents an individual atomic model, which altogether are combined into the composed model. Each component is an atomic PDEVS model. Subsequently, as the components are combined, they influence each other. The set of influencers for each atomic model is given by $I$. Furthermore, the combined model, as well as a single model needs input events and output events. In addition as the atomic models inside the coupled model produce output, which will be input for another component, $Z$ transforms the output of the first component into something reusable by the second component.
4. The DIANE project

Figure 4.2.: Composed model in PDEVS

4.1.2. General model

Figure [4.3] illustrates the general structure of a MANET in the DIANE project according to the PDEVS model. This is the MANET model used at the University of Rostock, following named Manet. In Manet multiple node model are connected with three central models; soc(cial environment), env (spatial environment) and net(work) are realized on the top level. On the next level each of these components can be refined. Each node model therefore is a coupled model, as depicted in figure [4.4]. The user model allows the adoption of different user behaviour, in general it controls when a user will move, where it will move to and which services it will request. The request of a service will be done over the serv port to the trading model, which will send an according answer over the same port. The trading model itself determines the way services are propagated and changed through the network. It represents overlay protocols, which are situated on top of layer four of the OSI reference model. The trading model in figure [4.4] can be refined according to the protocol, which is to be simulated on this layer. Until now two protocols were implemented: Lanes and Flooding. Further explanations on the Lanes protocol will be given in the next section. In addition, the trading component establishes the contact to other nodes over the transport port. This port carries message events representing the messages sent on layer 5, which needed to be exchanged over the network. Furthermore, the individual user model determines the ways a user moves, which is propagated to the central env model over the move port. Finally, the social port allows social behaviour in the model, like for instance group structures. As illustrated by the cardinality of [0..1] this behaviour is optional.

The central env(ironment) model is informed on movements of the user over the move inter-
face, which in exchange delivers information on the geographic surroundings to the user. Main task of the protocol lies in answering the requests the user has initiated via its serv port. To be able to achieve a result the protocol may need to interact with other nodes, since services are provided within the whole network without a single node knowing all services. To establish communication with other nodes the trading model contacts the central env(ironment) model via the transport port. This process represents the sending of messages over the network, yet as DIANE is based on the PDEVS formalism, these messages are actually events of the type message, which are exchanged between the different models.

Besides, the node model three central component models represent a map component, a network component and a social organization component. The map component, env model, represents the geographical area with its attributes like streets or buildings. The explicit
environment used in this work is a model of the university campus in Karlsruhe. The network component, net model, is modelled to fulfill the tasks of the transport layer in networks according to the OSI reference model. Though real systems use a complete protocol stack of the OSI reference model as explained in chapter 2.2, in the Manet all lower layers than layer 5 are combined into the central model without further redefinition. This was done as the focus of the project lay on the functioning of upper layer protocols concerned with service trading. The main functionality of message transport is provided by the net component over the transport port to the nodes. The necessary information on node connections due to their transmission reach are retrieved from the env(ironment) model over the links port. Consequently the net component works as a central routing table. But as this is no full implementation of layer three messages are directly given to the destination, without using routing.

The env(ironment) component tracks the movements of the nodes on the campus. The individual movement is retrieved over the move port from a node model. In addition, the positions of the nodes are also propagated to the soc component, where they can be used to build groups. Like the social port within each node the soc model is optional and does not need to be implemented to form a functioning model [24].

4.1.3. Assumptions for the model

To implement the model, the mapping from a real system onto the model is necessary. The abstraction for the used Manet occurs, on the one hand, in terms of the environment, in which devices are operating in real systems and, on the other hand, with regard to the devices and their functionalities themselves.

With regard to the environment it can be stated that it is simplified to a map of surroundings without further details than buildings. The main abstraction for this environment is the neglect of free moving within it. It is assumed that devices move along certain paths as it can be seen in figure [4.5] on the left side, but these modelled paths can be used randomly. This model is also referred to as a random waypoint model. The paths were chosen to pathways existing on the real campus in Karlsruhe. Under these circumstances taking the real pathways as the basic scheme for possible node movements maps the reality better than a free moving of nodes; people will tend to use given paths especially with regard to the fact, that buildings do not allow free movements. The whole area of the campus scenario has an overall dimension of 600 by 600 meters.

In addition, moving along these paths is no continuous process; but since the project is based on the PDEVS formalism each movement takes place after a discrete time step.
With regard to the devices used in ad-hoc networks, now represented by a node, following abstractions were made:

- each node has an omnidirectional antenna and equal submission power at its disposal
  - the actual transmission reach is 75 meters
- each connection between two nodes is bidirectional
- messages between two nodes are passed directly from sender to receiver
- connections are not weighted, i.e. signal strength is not considered
- all messages fit into one packet, therefore no fragmentation is necessary

### 4.2. Lanes - the overlay protocol

As mentioned before the DIANE project implements an overlay network. This is a logical structure for networks applied on top of the physical network topology. Formally an overlay is therefore a layer between the lower and the upper (user-oriented) layers of the OSI
reference model. This layer matches the communication system of the underlying network layers and the functionality to be provided. The overlay exists independently from the underlying infrastructure and usually has an address domain independent from it. Besides, an overlay usually has its own mechanism for structure determination. As DIANE was until now concentrating on the upper layers and the overlay, there was no full implementation of the actual routing process given. This was due done to the fact, that the project is focussed on service provision and usage, for which the Lanes protocol is fully sufficient. It was chosen as it represents a suiting solution for the management of services in the network and is also the basis of this thesis.

Accordingly the operation of the Lanes protocol is important to understand the overlay structure, as it determines the main functionalities of service provision and usage. Each overlay network defines a logical structure according to the task it should fulfil. This structure has to be build and maintained by the participating nodes. The structure and a short overview of the necessary operations of the Lanes protocol will be described in the following paragraphs.

4.2.1. Structure of the Lanes protocol

The Lanes protocol consists of several paths containing closely coupled nodes, called lanes. Each node within a lane only knows its predecessor and its successor in the lane and communicates only with these two. It is the main intention of the Lanes protocol to fulfil most operations within the lane. For example service descriptions are only propagated within the lane and stored by all nodes. Whenever a service is requested, the service descriptions of the lane are searched first. Yet this is a node internal action, what means less effort than a general search over all other nodes. However, if a certain service could not be found within a lane, lanes have to communicate with each other, what is realized with the help of anycast messages. To enable this inter-lane communication each lane controls a part of the anycast domain and has two specific anycast addresses, one towards its left neighbouring lane and one towards its right neighbour. Anycast messages in contrast to multicast messages are not sent to every member of the message group, but to the one node of the group being closest to the sender. However, only one anycast address is known to the neighbouring lane, depending on whether it is a right or a left neighbour. The overall structure of a MANET under the Lanes protocol is depicted in figure [4.6]. In this figure four lanes were established, which do not necessarily contain the same number of nodes.

Another important characteristic of the lane protocol is the fast adoption to topology changes caused by moving nodes. The contact to predecessor and successor is maintained by sending ping messages. These messages show changes in topology and allow the participating node to
express the wish for change. For this reason a ping message contains following information: the successor address, the state of the sender and a counter, counting the number of lane participants the ping message has already passed. An overview of all messages necessary for the building and maintenance of the Lanes protocol is given in table [4.1].
## 4. The DIANE project

<table>
<thead>
<tr>
<th>Name</th>
<th>Necessary for</th>
<th>Needs forwarding</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginAccept</td>
<td>inquiring node accepts login offer</td>
<td>no</td>
<td>200</td>
</tr>
<tr>
<td>LoginConfirmation</td>
<td>lane member accepts login, sent as reaction to LoginAccept</td>
<td>no</td>
<td>200</td>
</tr>
<tr>
<td>LoginInformation</td>
<td>inform successor in lane about the login</td>
<td>no</td>
<td>200</td>
</tr>
<tr>
<td>LoginInformation-Confirmation</td>
<td>successor confirms login</td>
<td>no</td>
<td>200</td>
</tr>
<tr>
<td>LoginOffer</td>
<td>offer for login by a lanes member as reaction to LoginRequest</td>
<td>no</td>
<td>200</td>
</tr>
<tr>
<td>LoginRequest</td>
<td>node wants to join lanes overlay</td>
<td>no</td>
<td>200</td>
</tr>
<tr>
<td>LogoffInformation</td>
<td>node gets information that predecessor or successor has logged off</td>
<td>yes</td>
<td>300</td>
</tr>
<tr>
<td>ServiceRequestResult</td>
<td>results suiting a request including service descriptions</td>
<td>no</td>
<td>400</td>
</tr>
<tr>
<td>ServiceOffer</td>
<td>node informs lane members about all services it offers</td>
<td>yes</td>
<td>200</td>
</tr>
<tr>
<td>ServiceRefresh</td>
<td>node informs lane members about all changes to services it offers</td>
<td>yes</td>
<td>300</td>
</tr>
<tr>
<td>ServiceRefreshComplete</td>
<td>last node in lane propagates refreshed information back to first node of the lane</td>
<td>yes</td>
<td>300</td>
</tr>
<tr>
<td>ServiceRequest</td>
<td>node request a certain service it cannot find within its lane</td>
<td>yes</td>
<td>400</td>
</tr>
<tr>
<td>ServiceRevoke</td>
<td>no informs lane that service is no longer available</td>
<td>yes</td>
<td>200</td>
</tr>
<tr>
<td>SplitRequest</td>
<td>request for a new lane</td>
<td>yes</td>
<td>400</td>
</tr>
<tr>
<td>SplitComplete</td>
<td>new lane is established, information is initiated by the end of the lane and will be propagated back</td>
<td>yes</td>
<td>200</td>
</tr>
<tr>
<td>MergeRequest</td>
<td>last member of a lane requests merge</td>
<td>yes</td>
<td>200</td>
</tr>
<tr>
<td>RemoveLane</td>
<td>members of a lane are informed that one border has changed</td>
<td>yes</td>
<td>300</td>
</tr>
<tr>
<td>Ping</td>
<td>keep regular contact to lane members</td>
<td>no</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.1.: Lanes protocol messages
4. The DIANE project

All messages are shown with a short description, the fact whether they need forwarding and a weight, which represents the calculation effort caused by them. This weight will be needed for trust calculation, since manipulation of message types with different workload can create personal advantage.

![Diagram of nodes in the lanes protocol](image)

Figure 4.6.: Structure of nodes in the lanes protocol - according to [5]

4.2.2. Service management

To maintain the structure of the overlay many different operations are necessary, yet since this thesis is focused on misbehaviour in service management these operations are of no concern here, for further details see [5], [3]. Nodes know all services offered within their lane, and besides an elaborated description of the service they store the unicast address belonging to each offered service. To organize this behaviour four main operations are necessary:

- new service announcements - publishing
- deletion of services - revoking
- periodical refreshments of service descriptions
- service search
To announce a new service a node sends the service description combined with its unicast address in a ServiceOffer message. This message is sent towards its predecessor as well as its successor, in case there are any. Each of these neighbouring nodes then saves the description with the address of the offering node and forwards it to its successor or predecessor depending on which node it got the information from. In case it got the ServiceOffer from its predecessor it will forward to its successor, if the message came from the successor the forwarding is done towards the predecessor.

Accordingly, a node can delete a service by sending a ServiceRevoke message which is spread the same way as a ServiceOffer. Furthermore, a ServiceRevoke as well as a ServiceOffer message can contain several services at the same time to be removed or offered, for instance when a node logs off or on.

Since there is a chance of transmission failure it cannot be guaranteed that each node within a lane has the recent level of information. Therefore it is necessary that the first node of the lane invokes a RefreshRequest after a certain period of time. Each node along the way to the end of the lane, refreshes the descriptions within the message and the last node of the lane sends the message’s content back as a RefreshComplete message.

Most of the times a node will try to find a service suiting its needs within its lane. For this reason it will check its stored service description to find a node offering a service within its lane. In case it finds a description it will contact the node directly via its unicast address stored with the service description. Yet if a service is not available within the node’s lane, the node will start a ServiceRequest to its neighbouring lanes. Therefore it will contact its left and right neighbour lane by sending a ServiceRequest message to their anycast addresses. These anycast messages will find the closest node of those neighbouring lanes, which will then compare the requested service with the services available in their lane. In case a matching service is found a ServiceRequestResult message will be send. In case no match is found, the receiving node will further forward the ServiceRequest to its neighbouring lane, if there is one. If no match is found on the edges of all the lanes an empty ServiceRequestResult message will be send. In the worst case a ServiceRequest has to be forwarded to all lanes, yet as each lane forms an anycast group, only to one of their nodes instead of to all of them. This is horizontal spread is a clear advantage of the anycast characteristic of the Lanes protocol.

An illustration of this process can be seen in figure [4.7]

The described usage of a ServiceRequest relies on an explicit service description in the ServiceRequest message. However, services may differ in small details, like the time interval for which they can provided or their design. Accordingly, it would be desirable to request a service more commonly, gather all answers and choose the most promising offer. There-
4. The DIANE project

Figure 4.7.: ServiceRequest in the Lanes protocol - according to [3]

fore, forwarding of the ServiceRequest message will be done in any case despite a possible result. This approach is used for the implementation at the University of Rostock, since more abstract service descriptions are in use here.
5. Misbehaviour and how to overcome it

As this work’s intention is to create a possibility to build trust, the main idea of trust should be clarified first. With regard to the recurring problem in MANETs, namely the question whether a node is working reliably and therefore sharing the network’s load, the following definition given by Gambetta (1988) [16] can be used:

"Trust is the subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends."

Within this work due to abstraction needed for simulation an individual is represented by a node. Subsequently, a possibility for an individual node to create an opinion about another node is needed. Usually to form an opinion on something usually a lot of information is needed. Accordingly, in this thesis trust within the field of MANET operation needs to be created, regardless of the amount of information possessed by a node. A node would need information on the context in which another node is working or not.

Before the already existing ideas to create trust are introduced a general overview about the possible misbehaviour in the Lanes protocol is given.

5.1. Kinds of misbehaviour

Within the scope of possible behaviour according to [3] following kinds of misbehaviour can occur in networks.

1. Selfish behaviour: this category contains participating nodes trying to conserve their individual resources by not fully taking part in the communication. These participating nodes do not forward messages or answer to ServiceRequests. Their behaviour is therefore categorised as passive. By not cooperating they create a personal advantage and violate the principle of cooperation on which MANETs are based.

2. Squandering behaviour: participants of this category are interested in creating a personal advantage as well as participants of the first category. Yet, they do not achieve
this by passive behaviour but by a disproportional usage of resources belonging to other participants. For example, a node uses a service of another node even though it could provide the service itself.

3. Malicious behaviour: in this case the participating non cooperative node is no longer focussed on creating a personal advantage but on harming other nodes in the network. An example is the manipulation of a node ServiceRequest message that way it leads to no result, instead of simply forwarding it.

4. Justifiable non cooperation: this case of non cooperation must be differentiated from the other possible ways of misbehaviour. It occurs due to the limited resources existing in a MANET. These limits can include small storage space, limited bandwidth or running out of energy. Especially energy affects the networks as participating nodes die silently [13], since they usually are mobile devices and have no possibility of permanent recharging.

According to these categories of malpractice it is desired to identify cases where misbehaviour is not caused by the node itself, but is due to the circumstances in the MANET. For further differentiation the kinds of malpractice in the Lanes protocol can be grouped as illustrated in figure [5.1]:

The main categorization is done regarding the fact whether the misbehaviour is detectable by the lane or not. Furthermore, misbehaviour is differentiated by the fact whether the behaviour includes active changes by the misbehaving node or results from passive behaviour like omitting actions. Most of these misbehaviours can be solved with the help of incentive schemes, which will be introduced in the following section. There is no need for routing context, yet with regard to the service management the bold framed cases need a context, since without it a resolution is hardly possible. These scenarios are covered more deeply in chapter 6.
5. Misbehaviour and how to overcome it

Figure 5.1.: Malpractice within the lanes protocol - according to [3]
5. Misbehaviour and how to overcome it

5.2. Incentive schemes

Since an ad-hoc network does not contain a fixed infrastructure, there is no centralized approach to control and punish misbehaviour. Every participant in the network will follow its own interests and will try to gain personal advantage. However, to support the Lanes protocol, as well as the general idea of MANETs to share the work for the network, an incentive is needed to motivate participants to cooperate. Therefore, incentive schemes were developed [18]. Generally remuneration is a widely accepted approach to stimulate cooperation between user and provider of a service. The kind of remuneration differs between the incentive patterns. Possible are bonds, that can be used like money. Another option is a reputation based on cooperation. In both cases participants are in need of bonds or reputation to be able to further use the services the network. According to [18] a taxonomy of incentive schemes looks like depicted in figure [5.2].

![Incentive patterns](image)

Figure 5.2.: Incentive patterns

It can be recognized, that there are two general approaches for incentive pattern. Firstly, there is the chance of stimulating cooperative behaviour by building trust between the different nodes. Secondly, remuneration can be used as stimulus. For trust based patterns there exists a differentiation between collective patterns and common patterns. Both patterns work without direct remuneration, since all participants within the domain benefit from trust and take cooperation for granted. In the collective pattern the trust is statically determined, which means that each participant in the domain gets the same trust from the moment he enters the network. Anyhow, it is necessary that before a node enters the domain it must be granted that the node earns the trust of the domain. Contrary, a community pattern wants its participants to earn their trust dynamically. Their trust is determined with every action the node takes part in. This also means that, before a node can consume services in the
network it has to offer its services to gain a certain trust value. As opposed to trust pattern, trade based incentive patterns take advantage of explicit remuneration. Remuneration can hereby be granted directly to the consumer by the supplier. In addition to a direct exchange of services, it also implies immediate exchange between consumer and supplier. However, here lies the problem of this approach, since it cannot be guaranteed that a consumer of a service can also offer the supplier something he wants at the moment. So, instead of enforcing one-to-one relations between the participants a bond based pattern allows an exchange of service between more nodes since bonds can be traded on.

5.3. S-Lanes

An incentive scheme does not only include one incentive pattern, but usually combines several of them into a suitable solution. In the diploma thesis [3] the Lanes protocol was extended with an incentive scheme. The resulting protocol is called S-Lanes. Altogether three stages to support cooperation were discussed.

The first approach grants only minimal security by adding a signature to the protocol. This way a message can be saved from corruption and the sender can be clearly identified. Respectively each receiver checks the signature of a message and discards corrupted messages.

The next stage implements a reputation system. With regard to the highly dynamic environment of MANETs a community pattern was used, where each participant builds its individual opinion on other nodes. For a more efficient reputation system the node also stores evidence on which it built its opinion. This way it can document its decisions. The individual ratings can be exchanged over the network on a regular basis to inform other participants. Based on these ratings a reputation is build for a certain node. A reputation system can be implemented with or without evidence, but the extension with evidence also allows a correction in reputation, which is not possible without them.

The highest stage integrated in S-Lanes uses a reputation system with evidence and complementary self-billing mechanisms. As incentive scheme a reputation system was recommended, where trust is dynamically generated for each participating node. This includes a trade based reward system, where bond, i.e. bills, are handed out to guarantee later actions. These bills are handed out by each node for special services another node has fulfilled for it. Following actions are to be rewarded:

- forwarding a message
- service matching
5. Misbehaviour and how to overcome it

- login offers
- support for successful login

Consequently a ServiceOffer message, which is propagated within the S-Lanes protocol includes more messages than the simple one described in chapter 4.2.2. The version including all securing measures is depicted in figure [5.3]. Next to the figure a description of all necessary actions is given, the bold entries represent the actual extension of grade three.

![Diagram of extended ServiceOffer]

Figure 5.3.: Extended ServiceOffer - from [3]
Before any kind of contextual information can be gained, it has to be clarified what exactly context is and which information can be used for its creation. According to [34], context can be defined as follows:

"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves."

Corresponding to this definition, information about the background and the circumstances, as well as the interrelations within the Lanes protocol is needed. As the assumptions about the lower layers of the OSI reference model are restricted and leave out details, the consideration was to extend the model within the DIANE model to allow more details to arise. These details should allow a node to form a more profound opinion on the reliability of other nodes in the network. All knowledge a node could gain was based on the Lanes protocol and its extension S-Lanes. However, devices in real systems have more contact with each other than within one layer protocol. To rate reliability nodes want to distinguish between misbehaviour and justifiable non cooperation. Therefore, it can be considered that a device will be able to receive all frames of nodes, in whose transmission reach it is located. Until now this consideration was not taken into account by the model. For this reason the first step in creating context was: enabling full layer two communication. Nevertheless layer 2 allows only one-hop connections, whereas the underlying MANET model requires multi hop connections, which are established in layer 3. The possibilities and variety of layer 3 routing protocols were already explained in chapter 3. The following section will present further details on this problem. The introduction of mechanisms belonging to layer 2 and 3 enable a node to overhear conversations of other nodes and gain information of it. This information allows answering following questions:

- When was the last secured contact to another node?
- Does a certain node contribute to protocol mechanisms?
6.1. Realisation of the routing protocol

As for the variety of ad-hoc routing protocols, it has to be stated that most of the protocols are adapted to very specific circumstances of their environment or very special requirements of a certain situation. For instance, the routing protocol GeoTORA was especially developed for the work with geographical data. Though it is capable to route for every network, it is destined to achieve efficiency and robustness for the routing of geographical data. Subsequently it is destined for outdoor use with GPS devices.

According to the classification of routing protocols, figure [3.1], one group could be excluded at once: hierarchical protocols. The reason for this exclusion is the flat assumed structure of homogeneous nodes working in the network simulated by the DIANE project. Nevertheless, there are further points to be considered, especially with regard to the already existing structure on layer 6, the overlay protocol. Hence, the underlying routing protocol was intended to fulfil following requirements:

- anycast capable
- reusable implementation (black box)

With consideration of the introduced routing protocols and the fact, that information should be gained by overhearing routing, a proactive, flat, topology-based routing protocol was needed. A protocol and topology-based routing protocol causes more traffic within the network and thus consumes resources, but on the other hand it delivers information on a more regular basis than other protocols.

Nevertheless, since it is not the purpose of this thesis to implement a full routing protocol, but to use routing as the underlying structure for extracting context information, a reusable implementation of a routing protocol meets this requirement.

The asked routing protocol should include a built-in anycast support, as the Lanes protocol is based on anycast groups. With regard to the anycast problem, there are adoptions to the wide spread standardized protocols enabling them to route anycast as described in [36]. Therefore it would have been possible to choose a protocol like DSR with an anycast extension, yet none of the considered protocols complied with all these requirements. For this reason, the decision was made to exclude a full routing protocol implementation.

Nevertheless, a solution had to be found to model routing behaviour, since it was needed to create a context. Therefore a closer look on real systems was desirable. Figure [6.1], shows a simplified version of four nodes in a WLAN. The circles around each node represent their...
6. Conception of context creation

transmission reach, and show, which node is in whose reach. This picture already shows the assumed nodes with equal transmission reaches, whereas in real systems reaches differ from device to device. This picture does not show a lane in its original state, but after some time, when it has eroded. In the original lane node A and D were one hop neighbours. However now it is necessary, that node B and C, as shown in figure [6.1], will have to route when A wants to send a message to D. The routing process is now introduced into the known model in figure [4.2] and [4.4], as illustrated in figure [6.2] and [6.3]. In figure [6.2], it can be seen that the Manet is complemented with two aspects. Firstly, the node model can have a link to 1 to n other nodes and secondly, a partial model named physical is introduced. These extensions to the original model allow individual nodes to be directly linked with nodes within their transmission reach. The creation of these links is accomplished by the partial model physical. It creates and removes links based on the information it retrieves over the link port from the map model. This behaviour was not provided within the original PDEVS formalism [19], it only allowed linkage at the instantiation of a model. To overcome this restriction the dynamic PDEVS [35] was created, it allows the dynamic creation, deletion and adaption of components during simulation. Assuming a node enters the network, the map model would recognize the node’s position and all other nodes within its reach of 75 meter, representing the assumed transmission reach in the model. Subsequently, the map model sends this information to the physical model, which will create connections between all concerned node models.

In addition, the net model’s work was redefined. It is now modelling the routing process, but still acts like a central routing table. The information is retrieved via the links port from the map model. Consequently, whenever a node wants to send a message to another node, it will contact the net model over the transport port, and which in turn will it provided with a route. Accordingly, the node will send its frame to the next node along the way using the
6. Conception of context creation

Figure 6.2.: Model of the reorganized Manet - from [24]

direct link, which was created between them.

Already from this description, it can be concluded that the node model itself must be extended as well. The extension is depicted in figure [6.3]. Whereas the former model only included the partial models user and trading, it now consists of four partial models. Though user is the same model as before, as well as trading, the routing and mac models are newly introduced. These two map the layer 2 and 3 of the OSI reference model. The routing model retrieves the routes from the central net model and sends a packet including all information, along with the next hop address, to the mac model. The mac model composes a frame based on this information. Moreover, it determines the right link with a frame port to the next hop node and sends the frame accordingly. At the same time it sends the frame over the outgoing frame port to all other nodes it is connected to. This way every node in the transmission reach of the sending node will be able to receive the frame.

With regard to the overhearing approach the observer should be able to catch all these frames. An extension including an observer model, therefore looks like depicted in figure [6.4]. Therewith, the observer model would be able to track all messages received over the incoming frame port and start its interpretation of possible misbehaviour on this basis. Yet this way an observing node is only able to observe messages of its neighbours, if it is located in their transmission reach and correctly linked to them.
6. Conception of context creation

Figure 6.3.: Model of a single node - from [24]

Figure 6.4.: Model of a single node extended by an observer component
6. Conception of context creation

6.2. Affected scenarios

This section gives further details on concerned scenarios and discusses possible solutions. Main focus lies on the scenarios that previously could be resolved by the already existing reputation system available in the DIANE system. Following scenarios are have to be considered. Firstly, there is the case that participants in the network structure do not send receipts for requests. This case arises with the usage of S-Lanes instead of Lanes. Secondly, there is the case when participants do not save the service announcement, and therefore are unable to answer ServiceRequests. This is shown in figure [5.1] as non lane detectable, passive malpractice. Thirdly, there is the participant device claiming it is the end of the lane even though it is not. This includes all lane detectable, passive malpractice, which includes no forwarding.

Before solutions to scenarios can be offered it is necessary to clarify, how context information can be gained. First, it has to be stated clearly that a node will receive all layer 2 messages. Nevertheless, not every frame will be unpacked, since on layer 3 according to the OSI reference model, it is checked whether the packet was destined for the node or not. On this layer packets not destined for the node are usually discarded, if no forwarding is necessary. At this point the functionality of overhearing can tie in. With regard to figure [6.5] it can be seen, that node E should receive all packets send by node A and B, while node F will only receive packets send by node A. Whenever A is now sending a packet to node B both E and F will be able to see this packet. Nevertheless, if A contacts node B, only node E will be able to see whether B responds accordingly or not. For this reason an incoming frame containing a request from node A will document to F that A is still reachable, whereas E will have to see that this is a critical message. Subsequently, E is in charge to track whether B cooperates according to some criteria.

In addition it is necessary to differentiate between the Lanes protocol and the actual routing, with all its packets sent. Considering that a lane remains in its state of creation would be the optimal case, since each neighbour would only be one hop away. In this case each message sent from a node A to its successors in the lane would need two frames, one for actually sending the message and one coming back acknowledging the frame. Therefore, twice as many than messages are sent.

However, this assumption is unrealistic as mobile devices cannot be considered to remain on the same location. They will certainly move. Therefore, over the course of time neighbouring nodes of the Lanes protocol will not be one hop neighbours any longer. Already the start of a SearchRequest will be more complicated.
In figure [6.6] it is depicted, how a node A in a lane wants to send a ServiceOffer to a node E. Yet these two are not one hop neighbours and therefore the ServiceOffer has to be routed over nodes B, C and D. So-called observing nodes are needed to decide, whether the participating nodes are cooperating. In figure [6.6] an observing node in the reach between C and D has to decide, whether there is something happening worth observing. Therefore an observing node, like G, has to detect on which layer the action takes place. To achieve this, the node will have to compare the addresses of the incoming frame to the one of the message inside. In case the destination address of the frame is the same as the one of the message, G observes a node involved in the Lanes protocol, in any other case it is watching routing and cannot draw conclusions on the behaviour of the destination node. However, there are nodes only in the transmission reach of one node, like F, which can only overhear messages sent by node C. This node will not be able to observe complete transactions at all, and therefore can ignore all incoming frames. Nevertheless, it should not discard the frames, but store them, since they might be required as evidence. For instance, even a purely routed message can give the last point in time a node was actually working.

Moreover, all messages in the network can be used to estimate the reachability of a node. All the frames sent may not be of importance for the reputation of a node, but each frame sent means a secured contact with the sending node. Therefore, the observing node G may ignore the message within the frame, but will secure it as the last known contact with node C. For every message of the Lanes protocol an acknowledgement (ACK) will be sent. This ACK is realized as an empty frame. The according response, an ACK frame from node D will secure the contact to D. When later a reputation on D must be acquired the information of this last contact may be become important. It assists G to tell reliably whether D is available or not. Yet, for storage space issues the message table of the individual node should not

Figure 6.5.: Nodes in an ad-hoc network
grow too big, not all messages should be kept permanently, as only the last one sent from a
certain node is important. This is especially important with regard to the amount of ACK
packets in the network. In figure [6.5] two ACK packets are sent, one for each message. In
contrast, the scenario in figure [6.6] needs four ACKs on the way to the destination node and
at least four more for the way back. One for each hop along the route to secure that the
packet actually arrived at the next hop. In reality, this amount will be a lot higher, since
the simulation in the DIANE project includes the abstraction that no fragmentation is needed.

The basic issue with regard to the received frames is the documentation and storage of
information, as well as the algorithms for interpretation. They form the context to build
trust values, on which a node can evaluate, whether it trusts another node or not. The
storage of messages is done by establishing an object representing a table, where all incoming
messages can be stored. This table allows different functions to gain several information.

<table>
<thead>
<tr>
<th>Time of receive</th>
<th>Frame.from</th>
<th>Frame.to</th>
<th>Frame.packet</th>
</tr>
</thead>
</table>

Table 6.1.: Storage of received layer 2 messages

Inside this table, the time when a frame was received, the sender address, the receiver ad-
ress and the packet inside the frame are store. To fill the table the node will gather all
incoming frames over the frames port and check whether frame address and message address
match. In case they do, the message will be stored as relevant traffic according to the Lanes
protocol. It will also add the current simulation time as the point in time, when a frame
was received. Accordingly it is possible to retrieve all messages a certain node has sent, e.g. to check whether it has forwarded a ServiceRequest. Therefore all messages from that node would be extracted, which are ServiceRequests into a new list. By going over this list the packets inside the frames will be compared; in case there is an equal one the forwarding was done, in case there is not a case of misbehaviour might be detected.

6.3. Scenario 1 - no receipts are sent

Based on the incentive scheme suggesting a self-billing mechanism, it is of high importance to forward and send receipts for actions that have taken place, because without them the initiating node will not hand out the bills. The ServiceRequest algorithm can be depicted as follows including the self billing mechanism.

Node 1:
1. signs ServiceRequest message
2. sends ServiceRequest to node 2
15. controls signature of node 3 and
   informs reputation system
16. sends signed bill to node 3
17. controls signature of node 2 and
   informs reputation system
18. sends signed bill to node 2

Node 2:
3. controls signature of node 1
4. requests reputation system
5. search for service unsuccessful
6. signs original ServiceRequest with own signature
7. sends ServiceRequest to node 3
14. receives receipt, informs reputation system and sends signed receipt to node 1
13. signs ServiceRequestComplete on the found service and sends it to node 1

Node 3:
8. controls signature of node 1 & 2
9. informs reputation system
10. requests reputation system
11. search for service successful
12. signs receipt and sends it to node 2

In figure [6.7] compared to the simple ServiceRequest [4.7] a higher message outcome can be recognized. This outcome will rise further with the lane eroding, since more nodes will be involved as transmitters for routing on layer 3.

Misbehaviour, in this case, would be node 3 behaving malicious and not sending a receipt to node 2 for taking part in the transaction. Without a receipt node 2 will not be able to send a
6. Conception of context creation

Receipt to node 1 and therewith would not be remunerated with a bill for participating in the transaction. Furthermore, node 1 would lack an incentive to take part in the overlay protocol and share the costs of it, since it is “misused”. Accordingly, this behaviour represents a violation of the MANET principle to equally share the network costs. Since this principle is basic, it is highly important to be able to detect and penalize this behaviour.

Detection by overhearing can be realized, since an observing node knows, which messages need a receipt and tracks messages that need to be answered. Hence, whenever a specific message as a ServiceRequest is forwarded, the observing node will recognize the message type and store the message. After a time period of 10 seconds it will check whether the observed node has answered the request or further forwarded the ServiceRequest message. In case the node has not forwarded the ServiceRequest, the observer will check whether a receipt has been sent. If it detects, there were no receipts sent, it will mark this action as suspicious. However, it is possible the observer missed the frame containing the receipt due to problems in the sending process. Therefore, it is necessary that the final decision about claiming a node unreliable is not done by a single observing node, but by a voting process of several nodes. Eventually, another node has tracked the receipt and can unburden the accused participant.

6.4. Scenario 2 - service descriptions were not saved

Independent from the incentive scheme, it is necessary that each node within a lane saves the service descriptions it receives within ServiceOffers. To verify this necessity, the second scenario considers the case that a participating node does not save service descriptions. By not saving the description, the participating node will not be able to answer a ServiceRequest it receives from a node of another lane. This behaviour infringes the idea of cooperation, and for this reason represents a form of selfish behaviour that is supposed to be punished. Consequently, three aspects are influenced by this behaviour. Firstly, the lane of the misbehaving node does not have the chance to take part in the transaction and will not be able to use the opportunity to gain bills, which are needed for further actions. Secondly, if the requested service is only available in the lane of the misbehaving node, the search will finish unsuccessful and the requesting node will not use the service at all. And finally, the misbehaving node forces more nodes into the ServiceRequest process and therewith charges their resources by producing unnecessary traffic. This happens due to the fact that simple forwarding of a ServiceRequest enforces a contact to another lane, if there are any. In the worst case causes a full horizontal search, whereas if the node would answer correctly the ServiceRequest would
not need to be forwarded.

To resolve this scenario, an observing node must know, which service descriptions reached the observed node. Nevertheless, an observer cannot see the service descriptions a participant saves, but it can track ServiceOffers sent to the participant. In addition, the observer also has to take ServiceRevoke messages into account to form a concrete list of services the node provides. Whenever the observer now realizes that the participant receives a ServiceRequest message and does not answer accordingly, it can search through its tracked messages, extract the service descriptions and recheck whether the participant really does not know the service. Yet, since failure free communication cannot be guaranteed it might occur, that the participant is not answering for several reasons. There is the possibility it left the transmission range of the observer or ran out of battery. Considering the case that the participant ran out of energy, it is possible to estimate the probability with the help of the battery model. Furthermore, when announcing a participant unreliable the observer has to make it as certain as possible that the participant is still within its reach and operable. Besides the battery it can further use contacts made with the participant - if there have not been contacts within a given time period the participant can be considered as out of range.

6.5. Scenario 3 - no message forwarding

The third scenario concentrates on nodes, which pretend to be end of the lane and therefore do not further forward messages within their lane. In addition, this behaviour can also occur for a ServiceRequest, which must be forwarded horizontally across the lanes. In both cases, the misbehaving node is ignoring following nodes and messages received from them. Therewith it can conserve its resources, since each forwarding process lowers charges. This misbehaviour negatively affects the possibilities and actions of the nodes it is receiving from and should forward to, since it forms a barrier in the way of transmission determined by the Lanes protocol. Consequently, the node is isolating groups from each other with regard to the service management, because only unicast messages can cross the node.

This misbehaviour can be easily detected, since the observing node recognizes contacts of further lane members. At least during the process of logging in, the node must have answered further nodes, otherwise they would not have that position in the lane. It can see the messages and whether they are directly destined for the participant or for further nodes. If it detects messages that should be forwarded the observer can immediately decide that
6. Conception of context creation

misbehaviour is given. However, it cannot be guaranteed that the node is still in the operating reach of the node and operating. This insecurity has to be calculated into the observation information.

6.6. Reachability of nodes

With regard to scenario 1 and 3 insecurity about the reachability of the participating devices has to be taken into account. An observation cannot be granted by a hundred percent but with a certain probability, which is based on the time since the last secured contact of a node.

![Comparison of nodes for the availability calculation](image)

Figure 6.8.: Comparison of nodes for the availability calculation

For trust building in the model, following considerations were made. The availability of a node in the model and therewith the probability it has moved on or died away silently, is calculated by using the time elapsed since the last secured contact. Therefore the observing node gets the highest time of a message received from the observed node and subtracts the actual simulation time. The absolute value represents the elapsed time since this last contact. The longer this time the greater the probability that the observing node is no longer in the transmission reach of the observed node. To find a solution several functions over time were considered. Each function should show that the actual availability sinks with the time elapsed. However, the considered time space is smaller than 15 seconds, since this is the time period after which a ping message is sent.
6. Conception of context creation

The first and simplest approach was the potential one. Yet these functions decay rather fast, after ten seconds the availability with the function \((1/t)\) sinks to ten percent. With regard to the campus model, this does not map the reality properly. On a campus participants do not simply pass by all the time, but spend lecture periods of several hours next to each other. With regard to the strategies for simulation of arrival processes, exponential, particularly logarithmic functions are common choice [20]. As a function to model the probability of an observing node to still be within the transmission reach of the observed node the following was chosen: \(1/(\log_{10}(t + 1))\). Its decay is lower than the decay of \((1/t)\) as can be seen in figure [6.8].
7. Realization

Realizing integration of trust values, it was important to equip the partial observer model from figure [6.3] with the necessary functions to work accordingly to the PDEVS formalism. Thus, following classes were created as shown in figure [7.1]. They were combined into the trust package.
Whenever during a simulation run a new node is created to join the MANET model, a coupled model, see figure [6.4], is instantiated for the node. In the node model, all atomic models are instantiated, and therewith the Model class belonging to the observer model implemented
7. Realization

within the trust package. The Model class provides the general framework for the PDEVS formalism, especially in terms of functions of the formalism. It is created for each new node, which is simulated, and creates an instance of the State class for the specific node. This way Model and State class are interrelated, as the Model instance is responsible for the creation of a State instance. In the State class instance, all values describing the actual state of the node are saved. This means, the MessageList mL is filled with TimedFrames, the node receives. All messages, that can cause one of the scenarios mentioned in chapter 6.2, are stored in the fQ FrameQueue. Furthermore, the unique id of the node and the simulation time are saved in the State class instance. The actual processing of the incoming frames within a node is illustrated in figure [7.2]. The activity diagram shows that, as soon as the simulation has started, the incoming frames are received and according to their meaning stored in the MessageList for general evidence and in the FrameQueue, if they remark a critical transaction. Subsequently, each frame, which entered the queue is removed from it after a given period of time as an internal event. In case the frame contains a ServiceRequestMessage it is checked, whether scenario 2 misbehaviour occurred, in every other case the node checks whether scenario 3 misbehaviour was given. With information retrieved from the MessageList mL the node controls, whether the transaction started by the message in the frame was correctly fulfilled. In case the transaction was not fulfilled, a TrustEntry is created. It is stored in the TrustValues list in the State class instance. At the end of the simulation, these TrustEntries are transferred to the general simulation data table. Based on this data the final calculations to gain an overall result are made.

Considering all the scenarios described in chapter 6, only scenario 2 and 3 were implemented and evaluated, since the implementation of the self-bond mechanism was not introduced at the University of Rostock. Yet, the model could be enhanced to include the remaining scenarios, according to the ideas introduced within chapter 6.

The following sections will explain the individual parts of the implementation.

7.1. Frame receiving

The Model class implements the import of the observer model, which receives the frame events. It is attached to the Frames import of the composed node model, figure [6.4]. As long as frames are coming over this port, they will be received in δ_{ext} of the Model class instance. First, the node will check whether the received frame represent a real Lanes protocol transaction and therefore compares the address of the packet with the one of the message within. If the frame event passes this check, the observing node retrieves the message within
the frame and looks for the message type. In case it is a frame that contains a message of a type, requiring forwarding according to table [4.1], it will be inserted into the FrameQueue fQ. Each frame in fQ will be stored with a time it has to wait. Considered as an event this means, they are waiting to become an internal event. At the moment their time to wait is up, they will be covered by $\delta_{int}$.

Moreover, all messages will be stored in the MessageList mL, since they are used for information retrieval. For instance, the last frame sent to or from a can be used to gain the last time a node was active. Moreover, a list of service descriptions known to the observed node, can be retrieved as a summary of all ServiceOffer and ServiceRevoke messages.

7.2. Check for misbehaovour

As the time advance function $ta$ calculates the discrete time steps it causes $\delta_{int}$ to check whether one of the frames in fQ ran from time and arises as an internal event. As soon as the time is up it will need handling according to scenario 2 or 3.

7.2.1. Scenario 2

Only frames containing a ServiceRequest message are relevant for scenario 2. For this reason this is the first aspect checked in $\delta_{int}$. In case the first frame in the queue is a ServiceRequest and its time is up, the node checks first, if the message was forwarded and no ServiceRequestResult was sent. Only if the message was forwarded without producing a result, there is the chance that the node is misbehaving. If this is the case, the observer will retrieve a service list of out mL. While looping over it, the observer will check whether the node knows the requested service or not. In case it does not know the service, no accusation will be made, otherwise a TrustEntry including an accusation will be created.

```
Frame suspect = pop(FrameQueue)
if (suspect.packet.message = ServiceRequest)
    then begin
        boolean forward := false;
        // all messages the observed node sent
        checkList := new list();
        checkList := allMessagesfromnode(suspect.receiveraddress)
        if (checkList.isEmpty = false)
            then begin
                while notendofList(checkList)
                begin
```
7. Realization

```plaintext
compare := next(checkList)
if compare.senderaddress =
    compare.packet.message.address AND
    compare.senderaddress = suspect.receiveraddress AND
    compare.packet.message = suspect.packet.message
then forward := true;
endofwhile

// in case the message was forwardet the node will check for misbehaviour
boolean misbehaviour := false;
if (forward = true)
then begin
    serviceList := new list()
    ServiceList := availableServicesfor(suspect.receiveraddress);
    if (serviceList.isEmpty = false)
    then begin
        while notendofList(serviceList)
        begin
            service := next(ServiceList)
            if service = suspect.packet.message.requestedService
            then misbehaviour := true;
        endofwhile
    end
endif

if misbehaviour = true then createTrustEntry(suspect.receiveraddress, scenario 2)
endif
```

Listing 7.1: Pseudocode for Scenario 2

7.2.2. Scenario 3

If the frame from the FrameQueue retrieved in $\delta_{int}$ is no ServiceRequest, it consequently must be one of the message, which needs forwarding. Hence, all messages the observer has stored from the receiving node are retrieved from the MessageList mL into a check list. While looping over the check list, the node examines, whether one of the frames in the check list includes the same message like the suspicious frame. In case it does, forwarding was done correctly, otherwise it was not. Accordingly a TrustEntry will be created. The pseudocode
for this action is shown in listing [7.2].

```plaintext
else begin
  boolean forward := false;
  checkList := new list();
  get all messages that were sent from the list
  checkList := allFramesfromnode(suspect.receiveraddress)
  if (checkList.isEmpty = false)
    then begin
      while notendofList(checkList)
        begin
          compare = next(checkList)
          if compare.senderaddress =
            compare.packet.message.address AND
          compare.senderadress =
            suspect.receiveraddress AND
          compare.packet.message =
            suspect.packet.message
            then forward=true;
        endofwhile
    endif
  endif
  if geforwardet = true then
    createTrustEntry(suspect.receiveraddress, scenario 3)
```

Listing 7.2: Pseudocode for Scenario 3

### 7.2.3. Trust value creation

The trust value creation differentiates between scenario 2 and 3. As scenario 2 can be clearly identified without doubt and occurs more seldom, it does not need a probability calculation, whereas scenario 3 can only be detected with insecurity and therefore uses probability calculation. Probability calculation is done like described in chapter 6.6. After the probability for misbehaviour was retrieved, the node will weigh the message type it checked. This weight is predetermined, as the effort for some messages is higher than for others. For instance, a ServiceRequest causes workload for the node, which received it, before it can be forwarded. Contrary, a ServiceOffer can be forwarded at once. According to the caused effort the weight was determined for each message type of the Lanes protocol as documented in table [4.1]. This process is necessary, as a node should not be able to upgrade its trust values by not
forwarding specific messages. For instance, a node could decide to forward messages, which cause less effort and leave out the ones implying a higher workload. Therewith, if a misbehaving node would only leave out some messages, it would conserve its resources and would restore its trust values by processing all other. Yet, by adding weight to the calculation, each message is taken into account, according to the caused workload. The total trust value is calculated as follows:

\[ \sum = \frac{\sum (probability \times weight)}{\sum (weight)} \]

This value is calculated for each observed node and stored in the TrustValues list. This list is transferred to the central simulation data table at the end of the simulation, where a final trust value for a node is calculated.

The nodes are accused based on mean values calculated from the individual trust values retrieved from single nodes. For example following TrustEntries might be retrieved at the end of a simulation run:

Node 1 accused with 0.34 by node 2  
Node 5 accused with 0.70 by node 2  
Node 5 accused with 1.00 by node 3  
Node 1 accused with 0.54 by node 3  
Node 1 accused with 0.08 by node 9  
Node 3 accused with 0.34 by node 9  
Node 3 accused with 0.60 by node 6

Listing 7.3: Trust values for individual nodes

Consequently, the central MANET model merges these values to calculate a single trust value for each node. Therefore it calculates the average trust value of every node. Correspondingly, the result for the example shown in listing [7.3] looks like depicted in listing [7.4].

Node 1 : 0.32  
Node 5 : 0.85  
Node 3 : 0.47

Listing 7.4: Mean trust values
8. Evaluation

Next to the provision of a context and a trust building system based on it, the solutions described and implemented should also be evaluated in terms of efficiency. To retrieve analyzable data, several simulation runs with differing parameters had to be done. Section 8.1 describes the settings for the tested scenarios. Afterwards section 8.2 presents the received data of the simulation runs. Furthermore, the evaluation for the individual simulated scenarios will be given.

8.1. Scenario settings

All scenarios that were tested for evaluation are based on the simulation model used in the DIANE project at the University of Rostock. This model uses the campus map as underlying geographical environment. In addition, there is no social environment implemented and routing is realized as described in chapter 6. One simulation run covers the time period of 2 hours (7200 seconds). In the first hour of this period all nodes enter (login) the Manet. Afterwards, in the second hour, all nodes stay available. The number of nodes to be simulated is a flexible parameter. For the simulation runs used in this thesis, runs with 20 and 50 nodes were made. To test the efficiency of the found solution, scenarios had to be created within this presetting.

As described in [17] there are two different approaches for misbehaviour simulation:

- black hole scenarios
- grey hole scenarios

In case a node is misbehaving all the time, it is called black hole scenario. This implies, that the concerned node processes no message. If the node processes only parts of the messages it receives, this is called grey hole scenario.

With regard to the two scenarios of misbehaviour, for which a solution could be implemented, following parameters were introduced to create simulation scenarios:

- number of nodes in the network cheating
8. Evaluation

- probability for cheating in terms of scenario 2
- probability for cheating in terms of scenario 3

So, simulation scenarios could be created, where every fifth node was actually misbehaving. The particular misbehaviour could be according to scenario 2 or 3, or both. Consequently, the final parameters for the simulation runs looked as follows:

1. Black hole scenario for scenario 2: every fifth node in the environment is not saving service descriptions and can therefore not answer service requests

2. Black hole scenario for scenario 3: every fifth node in the environment is not forwarding any of the messages that need forwarding according to table [4.1]

3. Grey hole scenario for scenario 2 and 3: every fifth node in the environment is not saving service descriptions with a probability of 0.5 and is not forwarding any of the messages that need forwarding according to table [4.1] with a probability of 0.5.

4. Grey hole scenario for scenario 2 and 3: every fifth node in the environment is not saving service descriptions with a probability of 0.3 and is not forwarding any of the messages that need forwarding according to table [4.1] with a probability of 0.3.

For each of these scenarios, 6 simulation runs were made with 20, as well as with 50 nodes.

8.2. Results

To be able to evaluate in terms of efficiency and analyzable output was required. To achieve this, for each simulation run following data was determined as output.

1. nodes correctly detected
2. nodes correctly not detected
3. nodes wrongly detected
4. nodes wrongly not detected

With every fifth node in the network cheating, for 20 simulated nodes at most 4, for 50 at most 10 nodes can be detected. Accordingly, the first value shows, how many of the nodes that really were cheating, were correctly detected. Subsequently, the second value represents how many nodes, which did not cheat, were correctly not accused. The third value shows how many nodes were accused falsely, whereas the forth value remarks the number of nodes,
which could possibly be cheating, but were not accused. 
These four values were determined based on the mean trust values and a threshold value of 0.5. Correspondingly, a node was accused, if its mean trust value in the time regarded period of simulation is above 0.5. During first simulation runs, this time period covered the whole simulation time. These runs showed no significant value for nodes, which were cheating according to scenario 2. Even though scenario 2 misbehaviour can be clearly identified, and therefore has a probability of 1.0, as well as a weight of 1000, instead of the 400 used otherwise for a ServiceRequest, it rarely occurs. For this reason, its actual occurrence is flattened out through calculation with the whole amount of messages of other categories. To overcome this restriction, the simulation time of 7200 was divided into 24 intervals of 300 seconds. For each of the time intervals all four values were displayed. Therewith, it could be observed, how the different values develop over the course of time.

![Figure 8.1.](image)

Figure 8.1.: Amount of observed nodes with a)20 and b)50 nodes in total

In figure [8.1] the total amount of observations made in each time interval is illustrated. Each
8. **Evaluation**

graph within a diagram is the average of the 6 runs done for a single simulation scenario. For 50 nodes not until the 12th interval the amount of observations reaches 60 percent (30 nodes) of all nodes in the network. For 20 nodes, this percentage (12 nodes) is hardly ever reached. The first twelve intervals are the ones, before all nodes are logged in. Accordingly, the probability that two nodes are close enough to observe each other is lower than with all nodes available. This fact is also underlined, if the values of the runs with 20 and 50 nodes are compared. In average for 20 nodes 9.7 nodes are observed per interval, whereas for 50 nodes 39.7 observed nodes are available. In percent, this is 48.5 percent for 20 nodes and 79.4 percent for 50 nodes. The variance for the values constitutes 8.6 for 50 nodes (17.2 percent) and 2.4 for 20 nodes (12 percent). Without all nodes being available, the distance between two nodes is likely to be bigger than the transmission reach of 75 meter. Accordingly, even if 30 nodes are already available, they will not be necessarily in the position to do observations. Nevertheless, as the the area is restricted and moreover, the paths of the nodes are predetermined, the transmission reaches are more likely to meet, the more nodes are available on the campus. However, it has to be remarked that for 20, as well as for 50 nodes, never all nodes could be observed. Accordingly, it can be stated that the basic data foundation grows with the number of nodes in a geographically restricted area. An observation for every node in the network indeed, cannot be granted as long as special locations, for instance lecture rooms, where nodes stay for several hours in contrast to pathways, which nodes only pass, are gathering nodes. Such settings circumvent the regular allocation of nodes over the campus, and map the reality, where students carrying devices will gather at certain places.

For further evaluation, only the second 3600 seconds will be taken into account, as they represent the stable state of the network and deliver more observations to calculate trust values from. Following the observations from the individual scenarios will be evaluated.

8.2.1. **Scenario 2, probability: 1.0**

The values for the black hole scenario 2 with 50 nodes can be seen in figure [8.7], for 20 nodes in figure [8.8]. For 50 nodes it includes following information: firstly, the amount of correctly not accused nodes in each reach is the highest; secondly, there were never more than two nodes correctly accused (1.39 nodes); thirdly, the amount of wrongly accused nodes varies between 3 and 5 (on average 4.2), and finally, the number of wrongly not accused nodes varies between 5 and 7. Summarizing observation 1 and 4 it must be recognized, that never all nodes, which should be cheating could be observed.

With a strong probability of 1.0 and a weight of 1000 clearly remarkable values were expected. For interpretation, it has to be remarked as positive, that most nodes were cor-
8. Evaluation

rectly not accused. Nevertheless, it is remarkable that of 10 nodes to be detected, maximal 20 percent were detected. This happen due to several reasons. Firstly, scenario 2 is occurring rather seldom, as service trading with the Lanes protocol aims at using most services within the lane. Moreover, to have a detectable scenario 2 a ServiceRequest must be anycasted to one of the cheating nodes. As anycast is always done to the closest member of the group, the probability that a cheating node receives an anycast ServiceRequest drops further. In addition, the observing node can only draw conclusions on the time it observed a node. In case, during this time, the requested service was not announced, it cannot detect the observed node. These facts lower the detection probability of scenario 2 misbehaviour.

Another point influencing the result, is the actual calculation of the trust value. It is calculated over a time interval of 300 seconds, in which all relevant misbehaving traffic is divided by all relevant traffic. The longer the interval, the more observations flattening the value are made. If for instance, the node receives a ServiceRequest and cheats on it in the first minute, but afterwards is detected several times for not forwarding due to mobility reasons with a small probability, the initial value, equipped with a probability of 1.0 will be flattened. For this reason a second simulation series was done with a time interval of 150 seconds. The results for the second hour are shown in figure [8.2]. These facts can also explain the rather high wrongly not detected number of nodes. Nodes, which should misbehave according to the setting simply did not get the chance during the simulation run. Nevertheless, this does not mean, they were not observed, otherwise they would not have been detected at all, but their low mean trust value results from the calculation, which is done for scenario 3 detection.

Figure 8.2.: Scenario 2 - probability 1.0; simulated: 50 nodes, time interval: 150 s

Analysing these observations, it can be remarked, that even though less nodes per time interval were observed, the number of right accusations stayed the same. Therewith it can be
concluded, that totally more nodes were rightly accused. This is due to the fact, that accused nodes have a high mean value in each time interval. This way misbehaviour is revealed instead of being hidden through calculation. In both figures [8.2] and [8.3], the average values over all runs are shown.

The diagrams for the individual values of correctly detected, correctly not detected, wrongly detected and wrongly not detected amount of nodes can be found in Appendix A. The average values are calculated based on the material shown in those diagrams.

![Diagram](image)

Figure 8.3.: Scenario 2 - probability 1.0; simulated: 50 nodes, time interval: 300 s

With regard to the effectiveness of scenario 2 misbehaviour detection, this implies that misbehaviour can be clearly identified in case the observing node also tracked all ServiceOffers and ServiceRevokes, which were send to the node. What may cause failures is a node, which moves into the transmission reach of an observer after it received a certain ServiceOffer. In case it afterwards is requested for this service and does not answer the request the observer will not accuse it, due to the fact it cannot give evidence for misbehaviour. Moreover this indicates, that the detection mechanism works more reliable for nodes moving together or staying together for a longer time period, like e.g. in a lecture room.
8.2.2. Scenario 3, probability: 1.0

The diagrams [8.7] and [8.8] show the calculated values for the black hole simulation of scenario 3. For 20 nodes, it can be seen from the diagrams in the appendix, that there were nodes detected, yet the average is below 0.5. With regard to the amount of correctly not detected nodes, the average value over all intervals is 7.43 and has a variance of 1.1. Consequently, only 37 percent were correctly not detected. However, the average number of nodes wrongly accused is 0.5, in contrast 1.5 nodes were not detected correctly.

For 50 nodes the values look as follows: First, on average 0.9 nodes were accused correctly. Second, 29.5 (59 percent) were observed and not accused. This value has a variance of 2.8 (5 percent) around the average value. With regard to the correctly detected nodes a rise from interval 13 (6 nodes) to interval 24 (8 nodes) can be seen. Thirdly, the percentage of wrong detections is 8 (4.15 nodes). Finally, 7.25 misbehaving nodes were observed but not detected, that are 14.5 percent of all nodes.

In comparison of both parameter settings it can be positively remarked, that the percentage of correctly not detected nodes rises with the number of nodes simulated in the network, whereas the variance in percent stays the same. The number of not detected misbehaving nodes doubles, and the number of wrongly not accused nodes triples.

Possible influencing conditions are that, opposite to scenario 2 misbehaviour, scenario 3 misbehaviour occurs more often, as messages that need forwarding are mostly used for mecha-
8. Evaluation

![Graph showing number of nodes correctly and incorrectly detected over time]

**Figure 8.5.** Scenario 3 - probability 1.0; simulated: 50 nodes, time interval: 300 s

![Graph showing number of nodes correctly and incorrectly detected over time]

**Figure 8.6.** Scenario 3 - probability 1.0; simulated: 20 nodes, time interval: 300 s

As the Lanes protocol is built mostly on these mechanisms, the message types occur more often. With regard to this fact, the possibility of detection is given more often than for scenario 2.

However, scenario 3 includes a calculation to resolve insecurities, which arise due to the mobility of the network. An observer, as well as the observed node move while transmitting messages and therewith it cannot be guaranteed that an observer can always track all messages needed to fulfill one action. Yet, due to this insecurity more accusations are actually
made than cases of misbehaviour occur in the network. In addition, both diagrams show, that most cases of misbehaviour were not detected reliably. This can have two possible reasons. First, the calculation of the mean trust values, does not take into account, which observer detected misbehaviour with what probability, but only calculates the average trust value. Second, there is no mechanism integrated, that reveals false accusations. For instance, if three participant are sitting in a lecture room and a third is passing by in the hallway. The simulated node, that passes and observes the start of an action, but not its end would, accuses one of the nodes which remain on the location. This trust value is taken into account for calculation, yet an observer inside the location cannot unburden the accused node until now.

8.2.3. Grey hole scenarios

The average diagrams showing the development over the course of time for these scenarios are shown below with figure [8.7] and [8.8], as well as in [8.9] and [8.10]. The average calculation is presented in table [8.1] and [8.2].

Figure 8.7.: Scenario 2 - probability 0.5; Scenario 3 - probability 0.5; simulated: 50 nodes, time interval: 300 s

For these scenario settings, the one with a probability of 0.5 as well as the one with 0.3, cheating nodes were misbehaving in terms of both possible misbehaviour scenarios. However, they do not cheat on every message of the Lanes protocol, but only on 50 or 30 percent.
8. Evaluation

Figure 8.8.: Scenario 2 - probability 0.5; Scenario 3 - probability 0.5; simulated: 20 nodes, time interval: 300 s

Figure 8.9.: Scenario 2 - probability 0.3; Scenario 3 - probability 0.3; simulated: 50 nodes, time interval: 300 s

From the calculated value it can, again be seen, that the number of nodes detected as reliable rises with the number of nodes in the networks. In addition, like in the black hole scenarios the number of wrong accusation rises with the number of nodes in the network. At the same time, the percentage of wrongly not detected nodes in the amount of overall detectable nodes increases about a third from 43-50 to 66-72 percent.
Figure 8.10.: Scenario 2 - probability 0.3; Scenario 3 - probability 0.3; simulated: 20 nodes, time interval: 300 s

<table>
<thead>
<tr>
<th></th>
<th>20 nodes</th>
<th></th>
<th>50 nodes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>variance</td>
<td>Average</td>
<td>variance</td>
</tr>
<tr>
<td>correctly detected</td>
<td>0.195</td>
<td>0.91</td>
<td></td>
<td></td>
</tr>
<tr>
<td>correctly not detected</td>
<td>7.1</td>
<td>1.46</td>
<td>29.15</td>
<td>4.8</td>
</tr>
<tr>
<td>wrongly detected</td>
<td>0.47</td>
<td>3.68</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wrongly not detected</td>
<td>1.93</td>
<td>0.37</td>
<td>6.8</td>
<td>1.6</td>
</tr>
</tbody>
</table>

Table 8.1.: calculated values for the 0.5 grey hole scenario

<table>
<thead>
<tr>
<th></th>
<th>20 nodes</th>
<th></th>
<th>50 nodes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>variance</td>
<td>Average</td>
<td>variance</td>
</tr>
<tr>
<td>correctly detected</td>
<td>0.21</td>
<td>1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>correctly not detected</td>
<td>7.69</td>
<td>0.7</td>
<td>28.64</td>
<td>6.13</td>
</tr>
<tr>
<td>wrongly detected</td>
<td>0.88</td>
<td>3.91</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wrongly not detected</td>
<td>1.6</td>
<td>0.31</td>
<td>7.27</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Table 8.2.: calculated values for the 0.3 grey hole scenario

Even though a difference in terms of accusations made between the scenarios was expected, the observed values look similar. Moreover, for 50 nodes, a slight rise in percentage of accused nodes, from 1.8 percent (probability: 0.5) to 2.2 percent (probability:0.3) could be made. Nevertheless, this may be due to the low amount of overall simulation runs. In terms of
scenario 2 detection this can be explained, as before, with the flattening of the values by the calculation of the mean value. This issue could be resolved with smaller time intervals as discussed in section 8.2.1. For scenario 3 misbehaviour also the same explanations can be given like in section 8.2.2.
8. Evaluation

8.3. Summary

Summing the observations up following points can be remarked:

- Scenario 2 can be definitely revealed if the observer gained all necessary information. Therewith the probability of detection should rise with the time an observer can actually observe a node.

- Scenario 3 detects more misbehaviour, than actually was committed. This results from the calculation of insecurity and needs further treatment.

- The observed values did not differ significantly with different probabilities of misbehaviour according to the two scenarios.

- With smaller time intervals a better scenario 2 detection could be achieved, what results from the fact, that mean trust values are calculated, what flattens the rather seldom occurrences of scenario 2.

To achieve a more sophisticated result for misbehaviour detection these points should be taken into account for further work. Overall it can be stated, that in terms of efficiency, the implemented solution delivers results, on which a trust value can be created. Nevertheless, the found solution leaves room for further improvement for instance in the field of determining an overall trust value. Here a voting on the misbehaviour based on evidence should add more security to the final result.
9. Summary

It was the main intention of this thesis to find a possibility to gain trust from contextual information in a MANET, which should be obtained from lower levels of the OSI reference model. Therefore, the Manet was used for simulation within the existing framework of the DIANE project. Solutions found had to be modelled according to the PDEVS formalism used in the project. Before solving explicit scenarios of misbehaviour, a method was created to produce context information, which was not introduced before in terms of lower level information. Even though it is not a fully implemented routing protocol, all necessary information to integrate overhearing are inserted into the model. A node is now able to receive frames and packets from nodes, in whose transmission range it is located. Therewith a node can overhear its neighbours and draw conclusions on their behaviour.

Following sections will give a concluding overview of the results gained with the context information, as well as some considerations for further research.

9.1. Conclusion

The first conclusion to be drawn, is that observing works better if more nodes are available on the location. For simulation runs with 20 nodes, the amount of observation was far less than with 50 nodes. However, more observation guarantees more information on which a trust value could be calculated. Without enough possibilities of being observed, nodes misbehaves without being recognized.

Furthermore, a solution was found for all three scenarios, which needed a trust value. The solutions for scenario 2 and 3 were implemented and tested. Detection of misbehaviour was realized, but with varying success.

For scenario 2 full detection could be realized without problems in the detection itself. In case misbehaviour occurs and the observing node has full access to information on the services propagated in the lane, it will reveal the misbehaviour. Problems occur with regard to the information on services. A further idea on this problem might be to integrate the ServiceRefresh messages, since they do contain recent service lists of the lane as well.

In contrast the solution for scenario 3 was not satisfying, as the results showed no significant
9. Summary

trust values with regard to misbehaving nodes. Nevertheless, this is not due to the actual information gained, but needs further analysis. In scenario 3 the full dynamic MANET scenario influences the value calculated by each node. One idea to solve this problem would be to establish a voting algorithm between the nodes, so evidences can be exchanged and nodes can be unburdened.

9.2. Further research

A possible next step could be the implementation of a routing protocol. Yet with regard to the variety of protocols and the found solution to retrieve routing information for context, this would only be necessary for the specific testing of routing protocols.

A far more interesting point to be taken into account, is the simulation of energy resources of the nodes. Here the thesis [12] suggests several approaches for a simple simulation of battery behaviour, as well as some advanced techniques, which also take recovery and memory effects into account. Besides, the simulation of a battery per existing node, the observer should integrate the battery resources into its calculation as well. Therewith the simulation of justifiable non-cooperation could be enhanced on nodes dying silently. It could be possible to initiate a kind of battery decreasing with regard to the messages that are processed within a node. For that effect the weight from table [4.1] could be used, as different messages have different energetic resources. The weight should be complemented with values for packets which are routed only. therewith a decreasing counter for battery values should be possible. Nevertheless it would be the question, how an observing node can retrieve the initial value for the battery resource.

Besides all this, an evaluation with regard to the effort observing costs would be necessary, as the process also consumes resources, and the question arises, whether the resource loss is worth the effort.

Furthermore with the introduction of layer 2 and three behaviour, it should also be considered, that misbehaviour occurs on these layers. A node is not restricted to forward with regard to layer 5 messages, but could always decide not to forward no matter what kind of message is currently incoming. An approach to detect layer 2 forwarding issues is already discussed in [17] and possibly could be be integrated in the existing model.
Bibliography


[18] Nimis, Jens; Obreiter, Philip: "A Taxonomy of Incentive Patterns - The Design Space of Incentives for Cooperation”; Second International Workshop on Agents and Peer-to-Peer Computing (AP2PC’03); Melbourne, Australia; 2003


[23] Wendell, Odom: "CCNA intro exam certification guide"; Indianapolis, USA; Cisco Press; 2006


[27] ISO 7498-1: "Information processing systems - open system interconnection - basic reference model"; 1984

[28] Müller, Karsten: "Routing in mobilen Ad-hoc-Netzen mit dynamisch zugewiesenen IP-Adressen"; Diplomarbeit; Universität Karlsruhe; 2004


BIBLIOGRAPHY


List of Figures

3.2. Directed acyclic graph ...................................................................... 17
3.3. Link reversal in the DAG of TORA ................................................. 17
4.1. Atomic model in PDEVS ................................................................... 21
4.2. Composed model in PDEVS .............................................................. 23
4.4. The composed model of a node within Manet - from [24] ................. 24
4.5. DIANE Visualization ....................................................................... 26
4.7. ServiceRequest in the Lanes protocol - according to [3] ................. 32
5.1. Malpractice within the lanes protocol - according to [3] ................. 35
5.2. Incentive patterns ............................................................................. 36
5.3. Extended ServiceOffer - from [3] ................................................... 38
6.1. Nodes in WLAN ............................................................................... 41
6.3. Model of a single node - from [24] .................................................. 43
6.4. Model of a single node extended by an observer component ............ 43
6.5. Nodes in an ad-hoc network ............................................................. 45
6.6. Nodes working and routing in an ad-hoc network .............................. 46
6.7. Extended ServiceRequest - according to [3] .................................. 47
6.8. Comparison of nodes for the availability calculation ....................... 50
7.1. Class diagram .................................................................................. 52
7.2. Activity diagram ............................................................................. 53
8.1. Amount of observed nodes with a)20 and b)50 nodes in total .......... 61
LIST OF FIGURES

8.2. Scenario 2 - probability 1.0; simulated: 50 nodes, time interval: 150 s . . . . . . 63
8.3. Scenario 2 - probability 1.0; simulated: 50 nodes, time interval: 300 s . . . . . . 64
8.4. Scenario 2 - probability 1.0; simulated: 20 nodes, time interval: 300 s . . . . . . 65
8.5. Scenario 3 - probability 1.0; simulated: 50 nodes, time interval: 300 s . . . . . . 66
8.6. Scenario 3 - probability 1.0; simulated: 20 nodes, time interval: 300 s . . . . . . 66
8.7. Scenario 2 - probability 0.5; Scenario 3 - probability 0.5; simulated: 50 nodes,
   time interval: 300 s . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 67
8.8. Scenario 2 - probability 0.5; Scenario 3 - probability 0.5; simulated: 20 nodes,
   time interval: 300 s . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68
8.9. Scenario 2 - probability 0.3; Scenario 3 - probability 0.3; simulated: 50 nodes,
   time interval: 300 s . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68
8.10. Scenario 2 - probability 0.3; Scenario 3 - probability 0.3; simulated: 20 nodes,
      time interval: 300 s . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69
## List of Tables

4.1. Lanes protocol messages .................................................. 29

6.1. Storage of received layer 2 messages ................................. 46

8.1. calculated values for the 0.5 grey hole scenario .................. 69
8.2. calculated values for the 0.3 grey hole scenario .................. 69
Listings

7.1. Pseudocode for Scenario 2 ................................................. 55
7.2. Pseudocode for Scenario 3 ................................................. 57
7.3. Trust values for individual nodes ....................................... 58
7.4. Mean trust values ............................................................ 58
A. Appendix: Diagrams
50 nodes, scenario 2: 1.0, scenario 3: 0.0

Correctly detected

Correctly not detected
Wrongly detected

Wrongly not detected
20 nodes, scenario 2: 1.0, scenario 3: 0.0

Correctly detected

Correctly not detected
Wrongly detected

Wrongly not detected
50 nodes, scenario 2: 0.0, scenario 3: 1.0

Correctly detected

Correctly not detected
A. Appendix: Diagrams

Wrongly detected

Wrongly not detected
A. Appendix: Diagrams

20 nodes, scenario 2: 0.0, scenario 3: 1.0

Correctly detected

Correctly not detected
A. Appendix: Diagrams

Wrongly detected

Wrongly not detected
50 nodes, scenario 2: 0.5, scenario 3: 0.5

A. Appendix: Diagrams

Correctly detected

Correctly not detected
A. Appendix: Diagrams

Wrongly detected

Wrongly not detected
A. Appendix: Diagrams

20 nodes, scenario 2: 0.5, scenario 3: 0.5

Correctly detected

Correctly not detected
Wrongly detected

Wrongly not detected
50 nodes, scenario 2: 0.3, scenario 3: 0.3

Correctly detected

Correctly not detected
A. Appendix: Diagrams

![Graph 1: Wrongly detected](image1)

![Graph 2: Wrongly not detected](image2)
20 nodes, scenario 2: 0.3, scenario 3: 0.3

Correctly detected

Correctly not detected
A. Appendix: Diagrams

Wrongly detected

Wrongly not detected
Theses

1. Routing information can be used as context information, since it indicates whether a neighbouring node is working. With regard to the mobile technology used within MANETs this implies, that within a model the information of layer 1 to 3 must be available for each node within the transmission reach of a sender.

2. The great variety of routing protocols for mobile ad-hoc networks, proves it difficult to choose a particular one according to a given scenario, in which the protocol should be applied. In this field of research a simplification would be desirable.

3. Within the implemented model of a MANET, it was possible to produce trust values based on context information, particularly routing information, without implementing a full routing protocol.

4. By overhearing network traffic it was possible to find solutions for open trust issues, which could not be resolved within the already existing reputation system, that was focussed on pure network traffic.

5. With full information some scenarios of misbehaviour can be detected without insecurities.

6. An integrated detection of several scenarios of misbehaviour is difficult in terms of combining the individual values into a combined trust value, as they should trigger different reactions. Otherwise, if all information is treated equally, the specific event is lost in the mass of data, though it indicates a clear case of misbehaviour.

7. As the trust values of individual nodes might contain insecurities, their should not rely only on their own calculation, but keep contact with other nodes to form a profound decision base - to build this profound trust value for a particular node in the network, evidences should be used to avoid unwanted accusations.
Eidesstattliche Erklärung

Ich erkläre, dass ich die vorliegende Arbeit selbständig und nur unter Vorlage der angegebenen Literatur und Hilfsmittel angefertigt habe.

Ort, Datum
Unterschrift

Einverständniserklärung

Ich erkläre mich damit einverstanden, dass ein Exemplar meiner Masterarbeit in der Universitätsbibliothek der Universität Rostock aufbewahrt und für die allgemeine Nutzung zugänglich gemacht wird. Ich nehme zur Kenntnis, dass dies Wirkung einer Veröffentlichung im urheberrechtlichen Sinne hat.

Ort, Datum
Unterschrift