


HE-MAN: Hierarchical Management for Vehicular Delay-Tolerant Networks

Ewerton M. Salvador¹  · Daniel F. Macedo¹ · José Marcos S. Nogueira¹

Received: 25 September 2016 / Accepted: 6 November 2017 / Published online: 13 November 2017
© Springer Science+Business Media, LLC, part of Springer Nature 2017

Abstract Vehicular Ad Hoc Networks (VANETs) are mobile networks that extend over vast areas and have intense node mobility. These characteristics lead to frequent delays and disruptions. A solution is to employ the Delay Tolerant Network (DTN) paradigm. However, the frequent disruptions as well as the delay and reliability constraints of certain VANET applications hinder the employment of both conventional and DTN-based management architectures. We present the HiE-rarchical MANagement (HE-MAN) architecture, which considers the specificities of Vehicular Delay-Tolerant Network (VDTN) management. The HE-MAN architecture implements a hierarchical management topology in order to take advantage of local communication opportunities for monitoring and configuration tasks. The proposed techniques for network clustering, monitoring, and configuration are evaluated using simulations, and results show that the proposed architecture successfully organizes the VDTN in relatively stable clusters, leading to more intelligent and efficient management of VDTN nodes through the usage of middle-level managers.

The authors would like to thank CNPq, CAPES, FAPEMIG, and CEMIG/ANEEL for their financial support in this work.

✉ Ewerton M. Salvador
ewerton.salvador@unipe.br

Daniel F. Macedo
damacedo@dcc.ufmg.br

José Marcos S. Nogueira
jmarcos@dcc.ufmg.br

¹ Computer Science Department, Universidade Federal de Minas Gerais, Belo Horizonte, Minas Gerais, Brazil

Keywords Vehicular Ad Hoc Networks · Delay-Tolerant Networks · Opportunistic networks · Network management

Mathematics Subject Classification 00-01 · 99-00

1 Introduction

VANET is becoming increasingly popular. Some countries are already conducting efforts to develop their own VANET systems, aiming to approach their specific needs. For instance, in the United States, the National Highway Traffic Safety Administration proposed a Federal Motor Vehicle Safety Standard (FMVSS). The proposed standard requires vehicle-to-vehicle (V2V) communication capability for light vehicles and the creation of minimum performance requirements for V2V devices and messages, in order to primarily support advanced safety applications [1].

Traditional wireless networking approaches should not be effective for implementing VANETs, since vehicular networks suffer from frequent topology changes and constant node disconnections. This occurs because of the high speed and mobility of vehicles, together with the fact that VANETs commonly extend themselves over vast areas, which may lead to partitioning of the network. While the usage of mobile Internet services (e.g., 3G/4G, GPRS, etc.) is increasingly growing and is an alternative for dealing with the mobility and the size of VANETs, service outages will still happen from time to time, and it is unrealistic to assume universally consistent coverage of such services in large and/or underdeveloped countries, as well as in challenging areas (e.g., mountains, deserts, forests, etc.). A Delay Tolerant Network (DTN) is a more suitable approach for the connectivity problem in VANETs, since it enables communication in environments with frequent disconnections and long transmission delays through store-carry-forward message switching. A VANET employing DTN protocols is usually referred to as a Vehicular Delay-Tolerant Network (VDTN) [2].

Just like any production network, VDTN needs to be managed. However, it is unlikely that traditional management solutions will work properly in VDTNs, since these solutions assume the permanent existence of end-to-end paths between any pair of nodes, and that the average communication delay is relatively low. These assumptions are not always met in DTNs. Architectures conceived for other types of DTNs (e.g., a deep-space DTN) are also not suitable for many monitoring needs of VDTNs, since a number of applications, such as the safety-related applications, require management operations with particular characteristics, such as very strict delay constraints to ensure a timely response for detected problems. For instance, the intersection collision warning application deals with messages being issued by neighboring vehicles every 100 ms. An eventual fault in such an application must be detected and handled within a matter of milliseconds, or a few seconds at most. Not coping with this delay constraint may lead to road accidents caused, among other things, by an untreated fault in the intersection collision warning application.

This article presents a new DTN management architecture devised for vehicular scenarios. The presented research is an extension of a previous work [3], as it provides an improved version of the clustering technique presented then, and it adds techniques for enabling local and remote configuration operations in the VDTN.

The remainder of this article is organized as follows. Section 2 presents the related work. The proposed architecture, as well as its algorithms for clustering, monitoring, and configuration are introduced in Sect. 3. Section 4 details the evaluation methodology and simulation scenarios, and the results are discussed in Sect. 5. Conclusions and future work are presented in Sect. 6.

2 Related Work

Birrane and Cole investigated DTN management by employing a system engineering approach [4]. The authors present implementation recommendations and current threads of research on the following: system architecture, data definition/usage, application architecture, and tool implementation. However, the authors do not provide any new concrete protocols or architectures for DTN management.

Peoples et al. [5] proposed a network middleware focused on providing context-aware policies in DTNs for deep-space communication. The impact of the policy-driven model on transmissions, its ability to achieve autonomy and self-management, and the relationship between an offered and received load were the aspects evaluated for the proposed middleware. However, the authors do not show how their solution could be extended to other scenarios.

Sachin et al. [6] extended the NETCONF to DTNs to install, manipulate, and delete configurations in network devices. Their proposal is called the Configuration Network Management Protocol (CNMP), and an initial prototype of this protocol was tested on a GNU Radio testbed. The authors discussed design on trade-offs and their impact on the configuration of nodes and services in a DTN. However, the definitions presented by Sachin et al. are rather preliminary. The research still lacks further design details, such as the specifics of the local policy rules checker, as well as a more extensive evaluation concerning how well their NETCONF extension scales in larger DTNs.

Papalambrou et al. [7] proposed a method for monitoring DTNs based on the publisher–subscriber model. The proposed protocol is called the Diagnostic Interplanetary Network Gateway (DING), and it was evaluated in a small network of three nodes based on the DTN2 implementation of the DTN Bundle Protocol. Despite the authors claiming their solution is reliable, it lacks mechanisms for properly dealing with frequent disconnections and long delays. Moreover, it is not clear if the authors' solution scales well with the size of the DTN, since their evaluation was limited to three nodes.

The Delay-Tolerant Network Management Protocol is proposed by the NASA DTN program, and it is designed to support the management functions of configuration, performance reporting, control, and administration [8]. Nonetheless, the document's primary focuses are to present an overview of the DTNMP and to

specify message flows and formats, without providing details on how the inner components of agents and managers should be implemented. Also, the DTNMP solution does not approach the particularities of VDTN management.

Nobre et al. [9] proposed the Opportunistic Management Contact Estimator (OMCE), which was designed for estimating appropriate times to execute future management tasks. Simulation results showed that the authors' proposal is able to improve the monitoring tasks in a DTN management system. However, this work relies on a P2P overlay network on top of the DTN. The computational costs of this set of overlays in a network were not evaluated.

Ferreira et al. [10] developed an SNMP-based management solution for VDTNs. They claim that their solution supports a load-related information collection, and provides a demonstration of this management application in a testbed. Their work, however, does not control the delays involved in such transmissions but instead, only encapsulates SNMP messages in DTN bundles.

Dias et al. [11] designed a monitoring and configuration tool for VDTNs, called MoM. This tool uses out-of-band signaling to collect statistical information about the performance of the nodes and spreads configuration messages across the network. Monitored statistics are addressed to terminal nodes, and whenever a problematic node is detected, a terminal node disseminates “advertisements” to other nodes of the VDTN in order to try to solve the problem. In this work, the authors rely on fixed infra-structure in the VDTN for receiving monitoring data and sending configurations to other nodes, which increases the cost of their solution and reduces its flexibility.

3 HE-MAN Management Architecture

In order to approach the open challenges regarding the management of VDTNs, a new architecture is proposed. This architecture is composed of a set of algorithms and definitions that (a) organizes the topology of managers and agents; (b) monitors devices and services in a way that is suited for VDTNs; and (c) implements configuration techniques that deal with control commands being issued to vehicular nodes not only in situations where long delays are involved, but also in situations where the configuration must occur with strict time restraints (such as safety applications).

We propose the HiErarchical MANagement (HE-MAN) Architecture for Vehicular Delay-Tolerant Networks, which is an architecture designed for managing VDTNs through the usage of a hierarchical organization of its components. The hierarchy allows the detection of a connected group of nodes to establish local managers and to take advantage of local communication opportunities for achieving near real-time management. It is important to notice that the usage of local communication in urban scenarios for improving networking aspects is also studied in other research [12]. The architecture is composed of a set of managers, one Top-Level Manager for the entire VDTN and a set of Mid-Level Managers, one per group of connected nodes. The other components found in the

HE-MAN architecture are agents, and protocols for organizing message exchanges and the logical organization of the VDTN's topology (Fig. 1).

HE-MAN takes advantage of local communication opportunities, which are faster and more reliable, while avoiding remote communication whenever possible (slower and less reliable). This strategy allows the management of applications and devices with little tolerance for communication delays. On the other hand, applications that are not sensitive to delays can be managed via remote communication. In VDTN, the division of the management tasks in local and remote tasks can be accomplished with a hierarchical architecture, where Mid-Level Managers (MLMs) manage locally connected groups. In the scope of this work, a locally connected group exists when at least two nodes are within transmission range of each other. Vehicles that are not within the transmission range of any other node are considered isolated. MLMs can perform tasks delegated by the VDTN's Top-Level Manager (TLM), which is a unique central entity that manages a vehicular network (e.g., the operational center of an enterprise that runs a fleet of vehicles). All MLMs established along the VDTN are directly subordinated to the TLM, which results in a 2-levels hierarchy of managers.

Figure 2 illustrates the software modules of the HE-MAN architecture: the TLM's software components on the left-hand side, and the agents and MLMs' software components on the right-hand side. The TLM must keep record of the status of network through time, in order to support the decision making process, either via a human network administrator or autonomic procedures. The main

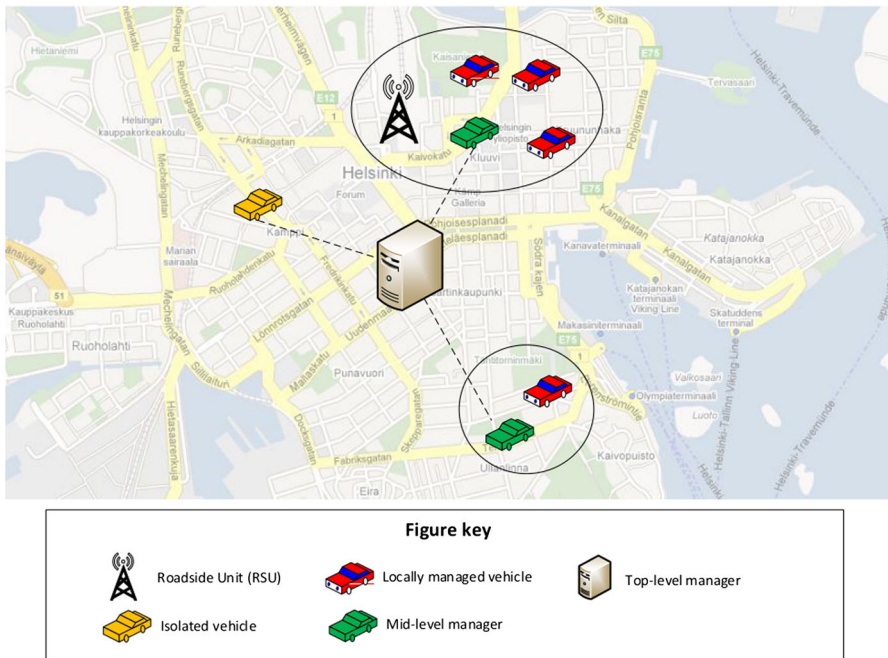


Fig. 1 Hierarchical monitoring in a VDTN

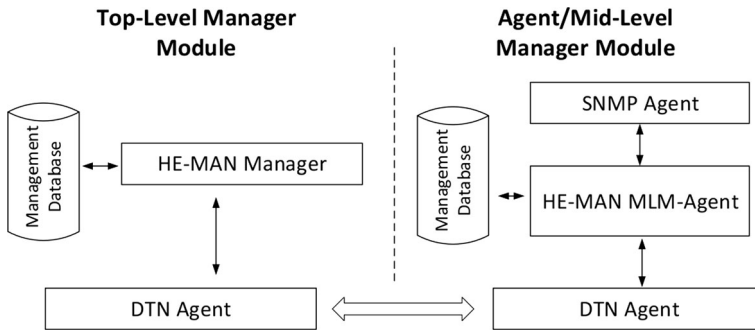


Fig. 2 Main components of the HE-MAN architecture

component of the TLM software module is the HE-MAN Manager, which requests information from the managed nodes and stores the received data in a local database. Different from traditional SNMP usage, which is polling-based, the HE-MAN Manager adopts the publisher–subscriber model: managed nodes are publishers while manager nodes act as subscribers. It is important to notice that the communication between publisher and subscribers are supported by the Bundle Protocol whenever the involved nodes are not in the transmission radius of one another. The HE-MAN Manager issues commands to the VDTN nodes, in order to control the behavior of the network as a whole.

The *MLM* nodes and *Agent* nodes share the same internal architecture, as shown in Fig. 2. This allows HE-MAN to promote an Agent into an MLM Node. In the HE-MAN MLM-Agent Module, the publisher–subscriber scheme is also implemented. Management data is accessed via an SNMP Agent, and the acquired data are sent to managers and stored in a local *Monitoring Database*, for redundancy. Whenever Agents/Local Managers need to contact nodes that are not within their communication range, the Bundle Protocol is used.

The materialization of a hierarchical architecture depends on the formation of stable groups of nodes, and the node with the best connection to all others must be selected as an MLM. These tasks are described in the next subsection.

3.1 VDTN Clustering Algorithm

HE-MAN clustering algorithm maximizes the amount of time that a node is connected to an MLM, so this node can be locally monitored/configured as much as possible. Unfortunately, clustering algorithms designed for vehicular networks sometimes implement filters to prevent nodes from joining a group, in order to increase cluster stability, thus reducing the connection time. For instance, some algorithms do not allow vehicles travelling in opposing lanes to be members of a same cluster, due to the short amount of time they would be in contact with each other.

We propose a new VDTN clustering algorithm that relies on information of the connectivity history of nodes to create relatively stable clusters. Our algorithm organizes clusters without preventing nodes from joining said clusters. Further, our

algorithm attempts to maximize the amount of time that a node spends connected to one of the VDTN clusters. To the best of our knowledge, an algorithm with such characteristics is not available in the literature.

A node can assume one of three roles: **isolated**, **agent**, or **MLM**:

- *Isolated* Nodes that have no active contact with others. Their objective is to identify opportunities of joining existing clusters or creating new ones;
- *MLM* The cluster-heads. Their main objectives are to maintain updated data about the current state of the cluster and to initiate cluster-head transference operations whenever needed;
- *Agent* It is a member of a cluster, but not its cluster-head. Nodes performing this role are constantly feeding the MLM (cluster-head) with monitoring data.

In our proposed clustering algorithm, the detection of neighboring nodes is based on the usage of periodic beacon messages. Only neighbors within a 1-hop distance are detected, so that nodes are 1-hop away from the CH. This is to avoid beacon messages needing to be retransmitted upon being received by a node in order to reach other nodes that are more than 1-hop distance away, which decreases the overall performance of the algorithm.

Beacon messages transport information concerning basic properties of a node (e.g., network address, vehicle identification, vehicle speed, etc.), as well as its **Connection Score** (connScore). The connScore is a metric that represents the history of connection opportunities that a node had, and it is used to determine if such a node is the fittest for the MLM role. This metric gives priority to nodes that are consistently establishing contacts with others. Each node calculates its own connScore by sampling the number of active contacts at a given time, and then calculating the samples' Exponentially Weighted Moving Average (EWMA).

The procedure *update* is periodically executed (e.g., every second) by each node of the VDTN. The first thing this procedure does is to update the value of the node's connScore. If the node is a member of a group (either as an agent or MLM), it checks if it can still be considered connected to the group. In order to do so, agents check if the last received beacon from the MLM node is within an acceptable time window, and MLMs check if beacons from the managed agents arrived in an acceptable time window. These time windows are configurable parameters. If this check fails, the node changes its status to **isolated**. At the end of the procedure, the node broadcasts the beacon with its current up-to-date role and other relevant information to its vicinity.

Each type of node reacts differently to the reception of a beacon. Because of this, different procedures for handling received beacons are specified, one for each type of node (isolated, MLM, and agent).

Algorithm 1 details how isolated nodes react when receiving a beacon. First of all, the recipient reads the information contained in the beacon to update the data about neighboring nodes, through the *updateNeighborhood()* call (line 2). Then, it checks if the received beacon originated from an MLM or another isolated node (lines 3 and 7). If the received beacon originated from an MLM, the node joins the MLM's cluster, changing its status to agent (line 5) and requesting that the MLM

subscribes to it in order to receive management data (line 4). When the MLM receives the *subscriptionRequestTo* from the isolated node, it issues a *subscribe* message to indicate the inclusion of the node in the cluster, which is then replied to with a *subscription ack* message. If the received beacon originated from another isolated node, the recipient node creates a new cluster and runs the election function to determine which node will perform the MLM role (line 8).

Algorithm 1 isolatedBeaconHandler(beacon) procedure

```

1: procedure ISOLATEDBEACONHANDLER(BEACON)
2:   updateNeighborhood();
3:   if isMLMBeacon(beacon)=true then
4:     subscriptionRequestTo(getFrom(beacon));
5:     setStatus(agent);
6:   else
7:     if isIsolatedBeacon(beacon)=true then
8:       winner ← runElection(beacon);
9:       if winner=me then
10:        subscribeTo(getFrom(beacon));
11:        setStatus(MLM);
12: end procedure

```

Algorithm 2 dictates how MLMs react to beacons issued by other MLMs, which happens whenever the involved MLMs attempt to merge their respective clusters into a single larger cluster. Upon the reception of a beacon sent by another MLM, the recipient calls the *contTime* function to verify if the two MLMs are in contact for an amount of time higher than a pre-configured threshold (line 4). The threshold is a configurable parameter in this algorithm, and can be defined according to the needs of each VDTN using the proposed clustering algorithm. If the minimum threshold is crossed, then the procedure checks whether the *connScore* of the MLM that issued the beacon is higher than the *connScore* of the MLM that received the beacon (line 5). In case the last condition is true, then the MLM with the smaller *connScore* issues a *requestGroupMergeTo* message, which starts the group's merging procedure (line 6). The *requestGroupMergeTo* message contains a list of the agents being managed by the MLM with the smaller *connScore*. When the other MLM receives this message, it issues a subscription request to the nodes in the received agents list, and sends a *Group Merge Ack* message back to the MLM with the smaller *connScore*, in order to release it from the MLM role. On the other hand, if the received beacon was issued by an agent, the MLM verifies if the connection to said agent is above a pre-configured threshold (line 8). If this verification is evaluated as true, the MLM checks if the *connScore* of the agent that issued the beacon is higher than its own *connScore* (line 9). If the agent has a higher

connScore, than the MLM issues a *requestMLMTransferTo* message, which starts the transference of the MLM role to the agent that issued the beacon (line 10).

Algorithm 2 MLMBeaconHandler(beacon) procedure

```

1: procedure MLMBEACONHANDLER(BEACON)
2:   updateNeighborhood();
3:   if isMLMBeacon(beacon)=true then
4:     if contTime(getFrom(beacon))>=THRSHLD then
5:       if getConnScore(beacon)>connScore then
6:         requestGroupMergeTo(getFrom(beacon));
7:   if isAgentBeacon(beacon)=true then
8:     if contTime(getFrom(beacon))>=THRSHLD then
9:       if getConnScore(beacon)>connScore then
10:        requestMLMTransferTo(getFrom(beacon));
11: end procedure

```

Finally, agent nodes do not have any special reaction to other beacons other than updating their data about the vicinity with the *updateNeighborhood* procedure. Agents respond to *subscription requests* and *subscription transfers* issued by the MLM that are managing them, in order to realize subscriptions and, ultimately, to become manageable by the cluster’s MLM.

3.2 Monitoring Support

A publisher/subscriber based on DING [7] is used in the HE-MAN architecture for monitoring tasks. In this model, presented in Fig. 3, the subscription message issued by the MLM is composed of two parts: a **schema** and a **schedule**. The **schema** is a static data model definition that describes what the manager wants to receive. It is a list of Object Identifiers (OIDs) that is verified by the publisher upon reception. After the publisher verifies what OIDs from the list it can provide to the subscriber, it will start to send the requested information in accordance to what is defined in the

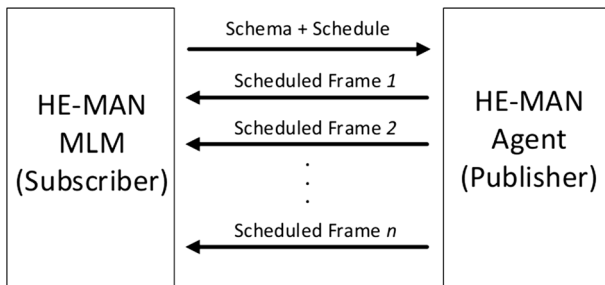


Fig. 3 Messages exchange in the publisher–subscriber monitoring

schedule. In this case, a **schedule** is a list of time intervals where agents are supposed to send management information back to managers. This **schedule** can have a single time interval that is applied to all OIDs requested in the schema, or one time interval for each OID in the schema. The time interval is specified in seconds.

Local management allows MLMs to subscribe to the agents in their respective clusters using subscription schedules with relatively short intervals, thanks to the low delay experienced in local communication. As a result, monitored objects can be verified more frequently, which is useful for the surveillance of critical network services and hardware components by a network manager. On the other hand, when **remote monitoring** is employed, the subscription schedule interval must be considerably higher than the schedule interval used in local monitoring. This is due to the limitations of remote communication in VDTNs: if a small time interval is specified when a TLM is subscribing to remote agents, then the VDTN is likely to be flooded by monitoring bundles issued by those agents in small intervals, which can be very taxing to the VDTN as a whole. MLMs can also aggregate monitoring data from vehicles for sending it in a single monitoring bundle to the TLM, in order to reduce the amount of bundles being transmitted via remote communication in the VDTN.

3.3 Configuration

Similarly to how monitoring is designed in HE-MAN, configuration operations can happen within clusters (local configuration) or between the TLM and remote located nodes (remote configuration). **Local configuration** is suited for scenarios where delay-sensitive devices and/or services must be configured, requiring new configurations to be transmitted in near real-time.

Local configuration is suited for scenarios where delay-sensitive devices and/or services must be configured, requiring new configurations to be transmitted in near real-time. The significant short delays and higher delivery ratio of local communication allows the MLM to detect issues almost instantly, similarly to what happens in regular connected networks. Because of this, it is possible to implement complex interactive configuration processes involving successive monitoring operations followed by configuration commands being issued.

Remote configuration occurs when the VDTN's TLM is the entity issuing configuration commands. Remote configuration is performed in a central point of the network where monitoring data concerning the entire network is stored, which usually leads to more well-founded management decisions. In HE-MAN, a TLM is able to create configuration scripts containing a set of configuration commands that are executed only if certain conditions are met. These conditions are meant to assess if the issue being approached is still valid when the configuration message reaches its final destination. The structure of the remote configuration message can be described as follows:

- *Conditional expression* Configuration messages carry a conditional expression, primarily to check if the state of the targeted node is still the same. Each

expression is composed of a reference to an Object Identifier (OID), a relational operator ($<$, $>$, $==$, $<=$, $>=$, or $!=$), and a constant. Relation expressions can be connected by **AND** and **OR** operators to enable the evaluation of multiple conditions in the destination node;

- *Cluster configuration flag* When the TLM issues a configuration message to a known MLM, a special flag can be set to inform the MLM that the configuration must be relayed to all cluster members. The flag is ignored in case the targeted node is not the MLM anymore;
- *Configuration commands list* The configuration commands are executed when the conditions within the configuration message are met. If the cluster configuration flag is active, the MLM will locally retransmit the configuration commands to all nodes in its cluster.

In order to illustrate how the configuration scripts described above could be applied in a vehicular network environment, consider the case where the VDTN vehicular nodes run a set of applications that trigger message exchanges whenever the vehicles get in communication range of one another. Now imagine that a large number of these vehicles got clustered together, and due to the high amount of exchanged messages, the nodes' buffers run out of space. This problem is reported by the nodes to the cluster MLM, which then notifies the problem to the TLM through a remote report message. A network administrator decides that some of the applications running in the vehicular nodes should be closed in an attempt to solve the buffer space problem, so the TLM issues the following message to the MLM of the group where the problem originated.

Conditional expression	<code>vdtm.mlm.average_cluster_buffer_usage (1.3.6.1.3.40.3.5) > 0.7</code>
Cluster configuration flag	<code>true</code>
Configuration commands list	<code>vdtm.applications.social_chat.active (1.3.6.1.3.40.5.82.1) = false</code> <code>vdtm.applications.advertisements.active (1.3.6.1.3.40.5.91.1) = false</code>

The expression of the message above instructs the MLM to verify if the managed object `vdtm.mlm.average_cluster_buffer_usage`, with OID 1.3.6.1.3.40.3.5, is still above 70% of usage, in order to avoid an unnecessary termination of applications. If the condition is true, the node checks the cluster configuration flag, which is marked as true, meaning the configuration commands should be relayed to all cluster members. Finally, the configuration commands list deactivates two applications that are represented by the objects `vdtm.applications.social_chat` (OID 1.3.6.1.3.40.5.82) and `vdtm.applications.advertisements` (OID 1.3.6.1.3.40.5.91).

3.4 Case Studies

This subsection presents two case studies for network management in order to illustrate the applications of the techniques presented in this chapter. Each case

study includes a description of the applications considered in the scenario, the issues that the management operations face in such a scenario, and how the HE-MAN architecture tries to solve those issues.

3.4.1 Case Study One: Platoon of Autonomous Vehicles

Consider the case of an army that transports equipment and troops using autonomous terrestrial vehicles. Moreover, these vehicles are grouped into platoons that move in a standardized way, where all vehicles of the group move at consensual speed while maintaining a desired space between adjacent vehicles. Coordination between the vehicles is necessary in order for the platoon to move as a cohesive unit. This coordination is done through a series of adjustments over their travel direction and speed, in order to respect the accorded platoon movement speed and direction, as well as the spacing between vehicles. It is not hard to notice that this coordination needs to be carried out with near real-time communication, otherwise the vehicles in the platoon would break the platoon formation because of laggy coordination commands.

The autonomous vehicles platoon scenario can benefit from the HE-MAN architecture in a few ways. First, the establishment of a coordinator in the platoon would be transparent, since the proposed clustering technique used in HE-MAN elects an MLM (cluster-head) whenever two or more nodes are in communication range of each other. Then, the MLM can assume the coordination of the platoon by subscribing itself for receiving monitoring data from the other vehicles of the group, using the local monitoring support. By having the required amount of monitoring data in a timely fashion, the MLM will then be able to issue configuration commands to the platoon's vehicles, such as speed and direction changes, so the platoon's formation can be maintained during the entirety of the operation. Finally, the VDTN's TLM can be installed in an operational command center, that will disseminate new configuration scripts (i.e., a set of configuration commands) to specific platoons with reduced overhead (in comparison to a pure flooding strategy).

3.4.2 Case Study Two: Bus Fleet Monitoring

For this second case study, consider a company that owns a bus fleet that transports passengers from one city to another. Each bus of the fleet is equipped with an OBU and a set of sensors placed on the vehicles' parts that the company wishes to monitor (e.g., fuel tank, tires, etc.). Opportunistic bus networks such as the one described in this case study are explored in other works in the literature [13]. Let us also assume that some of the cities the busses travel to, as well as some of the traveled roads, do not have access to any sort of mobile connection to the Internet. The company wishes to use a fleet management system that aims to collect data from the vehicles of the fleet. The company wants to monitor the trips carried out by the busses (e.g., number of passengers, number of unscheduled stops, etc.), and the "health" of the vehicles themselves (e.g., fuel consumption, tire pressure, etc.). Because of the nature of the data, the company does not have to receive the

monitoring data in real-time - there is room for some delay being tolerated, although it is preferable that the data arrives in the shortest time.

The main benefit the HE-MAN architecture can offer to this scenario is the remote monitoring feature, which refers to the MLM's ability to aggregate monitoring information regarding the nodes in its cluster and send this aggregate back to the TLM in a bundle. By doing so, it is possible to monitor a large number of vehicles without flooding the network with individual monitoring bundles. It is important to notice that this form of monitoring can deliver data more quickly if compared to a solution where the monitoring data is delivered to the TLM only when a monitored vehicle gets in range with fixed structures, such as an RSU inside one of the company's garages.

4 Evaluation Setup

Our proposals for VDTN management were evaluated through simulations, using The Opportunistic Network Environment (ONE) simulator [14], which is a well established simulator in the VDTN community [15]. The main objective was to compare our proposals to similar existing alternatives, in order to check if our proposals can successfully improve VDTN management tasks.¹ The original trace files went through a data sanitization process, where a set of nodes with a sample rate inferior to 30 s was selected. At the end of this process, 410 nodes were chosen, and the resulting trace file contained over 4 h of mobility data. Finally, this trace file was divided into 4 traces with approximately 1 h (3708 s) of mobility data, each for the experiments' repetitions. The number of vehicles in the simulations ranged from 100 to 400, and the total simulation area was approximately 30km × 45km (1350 km²). The obtained results were plotted with a confidence interval of 95%.

All simulated scenarios shared some common configurations. Vehicles communicated with the network using the Dedicated Short Range Communications (DSRC) technology, which has a typical communication range of 300 m. The radios were configured to transmit data at 10 Mbps, which is within the typical data rate range of DSRC. All communications with a bandwidth of 10 Mbps were half-duplex with a constant bit-rate. The buffer size of the nodes were 512 MB. Beacons were issued by the HE-MAN's clustering algorithm every 1 s. Finally, the TTL used in the DTN bundles was 60 min, in order to approximately match the total simulation time of each scenario (3708 s).

As for the specifics of each group of experiments, they are presented in the next subsections.

4.1 Clustering Algorithms

Simulations were carried out with different clustering algorithms in order to evaluate the performance of the HE-MAN clustering algorithm. The chosen base of

¹ As of October 2017, the mobility traces can be found at <https://www.ibr.cs.tu-bs.de/users/mdoering/bustraces/>.

comparison was the Modified DMAC Clustering Algorithm for VANETs (or simply MDMAC) [16] because it is one of the most recent algorithms that shares an important design principle with the clustering technique designed with HE-MAN: they both use information on the movement of the vehicles as the most important source to decide how to cluster network nodes. Both HE-MAN's clustering algorithm and MDMAC are based on periodical broadcasting of beacons to detect topology changes. Moreover, MDMAC is a relatively simple algorithm that is described with plenty of details in the literature and allows an easy implementation. The decision making of both MDMAC and HE-MAN's clustering algorithms relies only on the vehicles' movement speed and direction, and it does not require other sources of information, such as GPS or information about the roads' lanes.

MDMAC is a representative of a large group of clustering algorithms that employs some criteria for determining if a node will be able to join a cluster or not, aiming to minimize changes to the cluster due to nodes constantly joining and leaving just a few moments later. This design choice is different from the HE-MAN clustering algorithm that aims to maximize the time where vehicles are members of clusters, without preventing vehicles to join clusters. Other recent clustering algorithms for vehicular networks tend to employ more complex clustering metrics (e.g., quality of the link between a pair of nodes, similarity functions, etc.) that are usually not viable to be implemented in a simulator such as The ONE Simulator. Finally, it is important to notice that the HE-MAN clustering algorithm was proposed as an example of a clustering algorithm that have the properties needed by HE-MAN's management modules. HE-MAN's overall architecture is not dependent on this specific algorithm, and it can work with other clustering algorithms that has the property of not preventing nodes from joining clusters that are within the range of the nodes' transmission radius.

The value of λ in the EWMA that was used for calculating the connection score metric was set to 0.2. The time threshold, where agent nodes tolerate the absence of received MLM beacons without abandoning their current cluster, is set to 3 s. There is also a minimum time threshold of 80 s of continuous contact for operations of MLM transferences inside groups and groups merging to be started. As for the MDMAC algorithm, we used the same parameters found in the original work [16].

4.2 Monitoring

The efficiency of our proposed algorithms for monitoring and configuration are evaluated through two metrics: transmission delay, which is the time a management messages takes to travel from a source to a destination; and delivery ratio, which is the relation between delivered and total sent messages. Two applications are present in the vehicular nodes of our simulated VDTN.

Collision avoidance is a safety application for vehicular networks. It uses sensors in vehicles to monitor the surroundings. Beacon messages are sent every 100 ms, containing data such as the vehicles' position, speed, and direction that are used for the prevention of accidents. It is important to notice that 100 ms is a typical beacon

interval for vehicular safety applications [17, 18]. The application demands high data rates and low response times (hundreds of milliseconds).

Event logging system is an application designed for the management of fleets of vehicles. This is a typical application used to ensure safety and operational efficiency, emitting alerts of broken parts, unauthorized stops, or vehicles exceeding the maximum speed. This is a non real-time and low data rate application that accepts delays in the order of hours or a few days. The application reports data concerning distances traveled and registered speeds for each vehicle, which is sent to a central management system in remote reports.

The size of the beacon messages issued by the Collision Avoidance system was 64 bytes, while the size of the fault notification messages sent by the VDTN monitoring system whenever a faulty beacon message was detected was 1024 bytes [18, 19]. The probability of a beacon from the Collision Avoidance system being corrupted is 10%. This intentionally high probability was chosen in order to test the ability of the evaluated monitoring systems to cope with a relatively demanding situation. The remote reports issued by the Event Logging system have 512 KB of size (hypothetical size relative to 30 min of monitoring data of the cluster's members), and are sent by the vehicular nodes to a network manager every 30 min. The considered routing algorithm are the following: Epidemic with Oracle, Epidemic, Spray and Wait, and Prophet.

Vehicles also run a monitoring software module, called the *Agent/Mid-Level Manager Module*. When the **Agent Module** is employed, the application detects faulty messages issued by other vehicles near them running the Collision Avoidance application described above, in order to notify such faults to one of the network managers using monitoring notification messages. If the vehicle is an MLM, then it also runs the functions described in Sect. 3.

The proposed monitored solution, as well as the baseline, are described below.

HE-MAN's monitoring solution In this scenario, the VDTN is organized with a TLM placed in a fixed position at the center of the simulation area, and the vehicular nodes moving around the TLM, either within the established clusters (as MLMs or managed agents) or travelling in an isolated manner. The MLMs are elected using the HE-MAN's clustering algorithms. The publisher–subscriber monitoring model is employed in the communications between managers and agents, with agents issuing monitoring messages to subscribed MLMs whenever an issue is detected. Also, MLMs aggregate the remote reports issued by the event logging system of the vehicles within the local group in order to forward a single remote report bundle to the TLM, in order to reduce the overhead of messages travelling through the VDTN.

Simple VDTN monitoring This scenario, inspired in traditional network management approaches, employs a single manager at the center of the simulation area monitoring the VDTN. For the sake of fairness, the publisher–subscriber approach will also be employed in this scenario.

4.3 Configuration

The communication pattern found in the local configuration category is practically the same of the HE-MAN's local monitoring, especially when some sort of

interactive configuration is being employed. Because of this, only the remote configuration support will be evaluated in this subsection. Therefore, the main objective of this group of experiments is to verify the feasibility of the distribution of remote configurations to agents and MLMs.

The configuration message size generated by the main manager is 10 KB (hypothetical size of a configuration script containing a set of conditions and configuration commands). After the warmup phase, the simulation time is divided in configuration generation ticks, with each tick occurring every 5 min. Immediately after a configuration generation tick, the TLM generates 10 configuration messages to 10 different vehicles. The vehicles are randomized in every configuration generation tick. The Epidemic with Oracle routing was used, in order to evaluate the performance of the distribution of configuration messages in a best-case scenario routing wise.

5 Results

This section presents the results of the evaluations concerning our clustering algorithm and the proposed algorithms for monitoring and configuration.

5.1 Proposed Clustering Algorithm

Figure 4 shows the average amount of time that a node spends connected to one of the network clusters. Since HE-MAN's clustering algorithm does not employ any sort of filter for a node to join a cluster, it performs better in this metric than the MDMAC algorithm. Moreover, as the density of the network increases, the difference of performance between HE-MAN's clustering and MDMAC also increases.

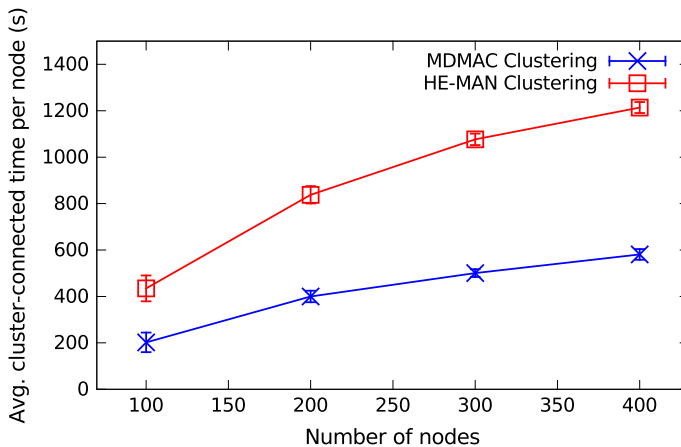


Fig. 4 Average time that nodes spend connected to clusters

The cluster-head transferences per cluster are an indicator of how stable a cluster is, since well picked cluster-heads usually do not need to be changed in short periods of time. Figure 5 shows that HE-MAN’s clustering algorithm needs fewer cluster-head changes per cluster in comparison to MDMAC. HE-MAN’s algorithm registered from approximately 0.03–0.04 cluster-head transferences per cluster in all simulated scenarios, while MDMAC registered from 1.04 cluster-head changes per cluster (a 100 node scenario) to 1.44 cluster-head changes per cluster (a 400 node scenario). The probable explanation for this difference is the usage of a minimum connection time threshold in HE-MAN. This threshold restricts clusterhead transferences only to the cases where the clusterhead and clusterhead candidate are connected for a significant amount of time, which drastically reduces the overall number of cluster-head transferences.

Figure 6 shows the average delay achieved through the clustering algorithms during intra-cluster communication. For both algorithms, the average delay was typically inferior to 2 s, yet HE-MAN’s clustering managed to perform better in this metric than MDMAC. While the local communication delay with MDMAC ranged from 0.3 s (100 nodes) to 1.9 s (400 nodes), in HE-MAN, this delay ranged from 0.3 s (100 nodes) to 1.1 s (400 nodes). The reason for the better performance of HE-MAN over MDMAC resides in the fact that MDMAC takes longer to realize when a node is not connected to a cluster anymore, eventually being able to deliver messages to nodes that spent a few seconds disconnected from the cluster, which increases the average delay.

The results presented in this subsection show that the proposed HE-MAN’s clustering algorithm increases the amount of time that nodes spend being members of clusters in comparison to MDMAC. Moreover, the clusters formed with HE-MAN’s clustering algorithm were stable enough to keep the number of clusterhead transferences and the local communication delay lower than the MDMAC, while being able to provide a higher delivery ratio. On the other hand, the number of created clusters is considerably higher in the HE-MAN’s clustering algorithm in

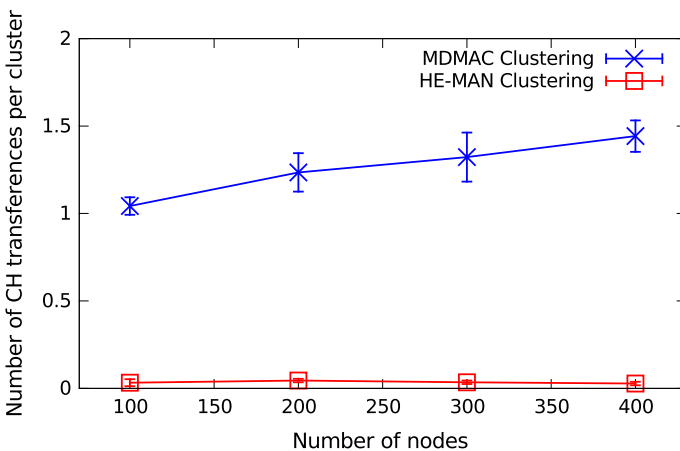


Fig. 5 Number of CH transferences per cluster

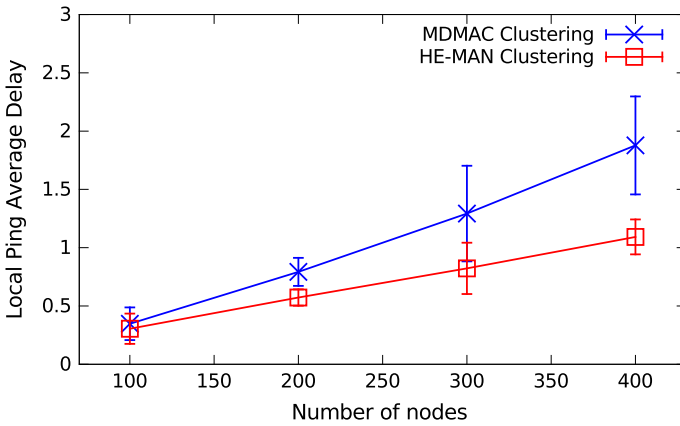


Fig. 6 Notification delay in local communication

comparison to MDMAC, which can lead to a higher overhead related to the messages issued during clusters creation.

5.2 Monitoring

Figure 7 shows the notification delay values obtained with the Collision Avoidance Application. In this scenario, a monitoring notification message takes from 14 min up to 20 min (on average) to reach its destination in the Simple VDTN Monitoring scenario. On the other hand, the average notification delay is between 0 and 1.46 s in the monitoring solution proposed with the HE-MAN’s architecture. This is an

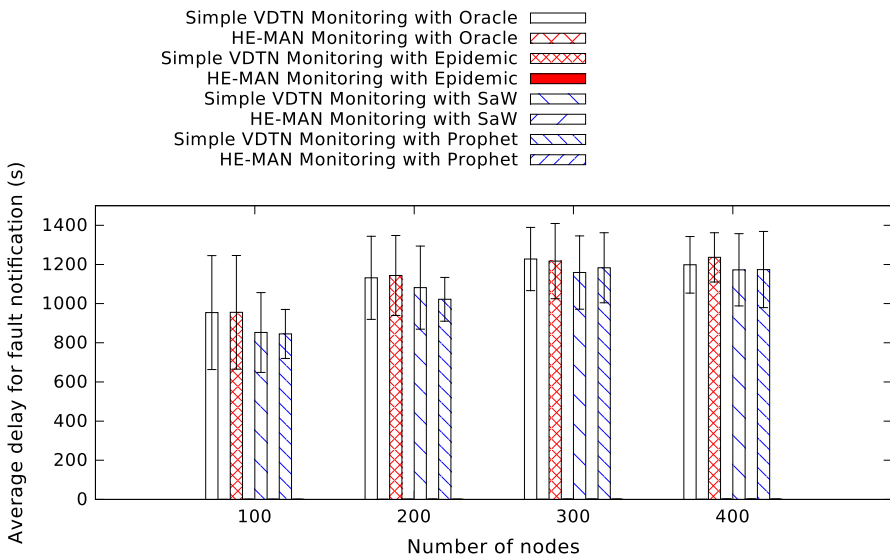


Fig. 7 Fault notification average delay

expected behavior since the communication occurs within a connected group due to the hierarchical organization of the network. Although there are some differences in the averages obtained through the used routing algorithms, these differences are subtle. The differences are more clear in the sparser simulated scenarios.

Figure 8 presents the notification delay values achieved with the Event Logging application. A remote report generated within this application takes from 13 to 15 min on average to reach its destination in the Simple VDTN Monitoring solution. The notification delay increases as the VDTN gets denser, as one would expect since denser networks provide more opportunities for messages getting to their final destination using more hops, which results in longer delays and higher delivery ratios. In the HE-MAN’s monitoring solution this delay is between 6 and 16% higher than the ones observed in the Simple VDTN Monitoring scenario. This difference happens because of the aggregation process that occurs in the MLMs, which reduces the number of remote report messages in the VDTN. Different routing algorithms had a little impact over the performance of the evaluated monitoring approaches.

The message delivery ratio was the second metric studied. Figure 9 shows the delivery ratio obtained with local communication. For all tested DTN routing algorithms, there is a clear difference between the HE-MAN’s Hierarchical Monitoring and the Simple VDTN Monitoring. HE-MAN’s monitoring solution always delivered more than 98.5% of the notifications, while the Simple Monitoring Solution always delivered less than 40%. The increase in the density of the VDTN benefits the fault notification delivery ratio of the Simple Monitoring solution due to higher availability of forwarding opportunities.

As for the delivery ratio achieved by the Event Logging application, shown in Fig. 10, the observed average in the Simple VDTN Monitoring solution was always

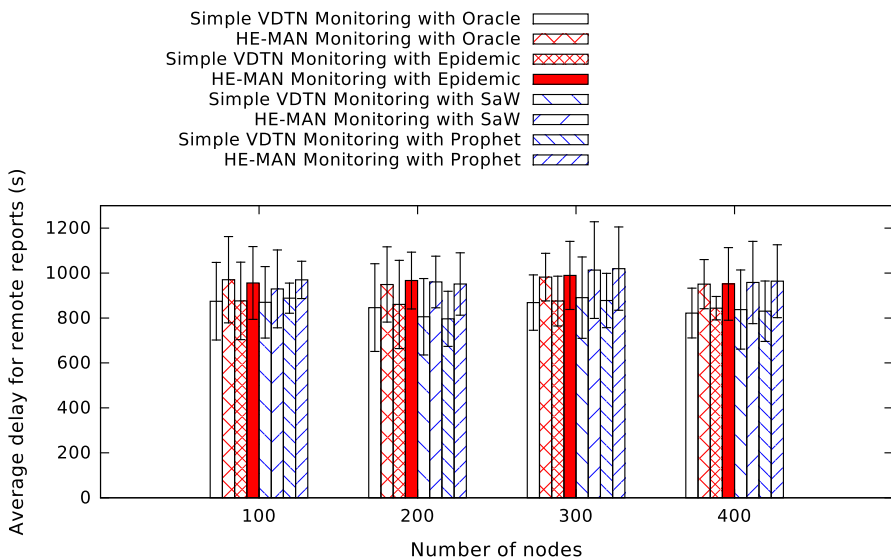


Fig. 8 Remote reports average delay

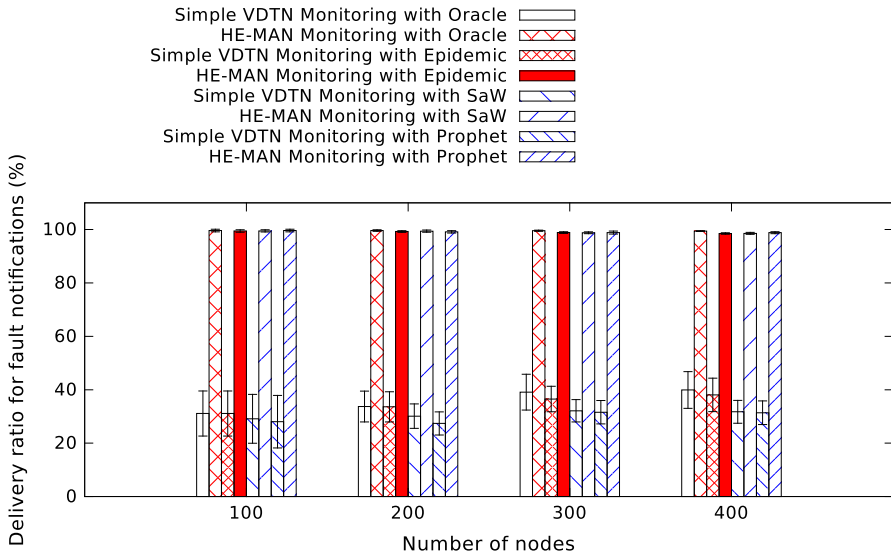


Fig. 9 Fault notification delivery

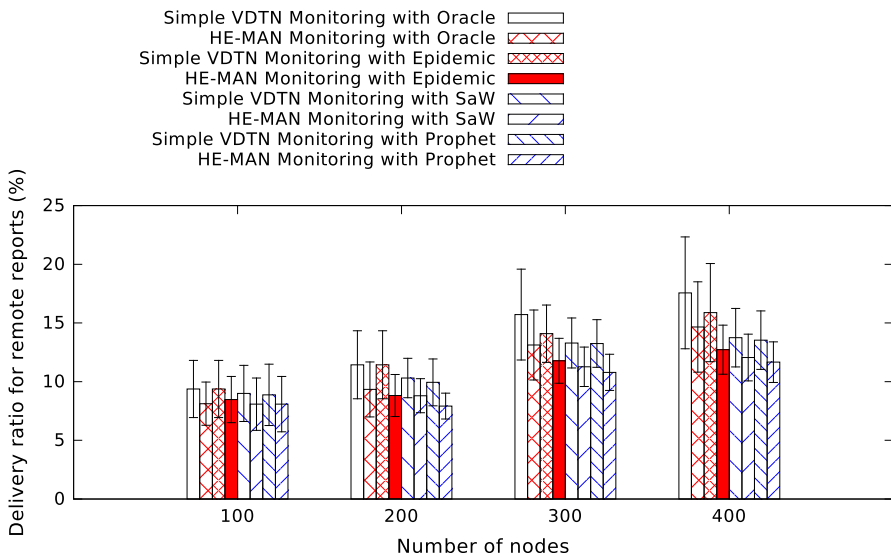


Fig. 10 Remote reports delivery ratio

inferior to 18%. With the HE-MAN’s monitoring solution, the average delivery ratio remained inferior to 15%. Different routing algorithms had a limited impact on the results. The difference between different routing techniques tends to become more perceptible as the VDTN gets denser. Again, the inferior delivery rate observed in the HE-MAN’s approach can be explained by the aggregation of reports in the MLMs.

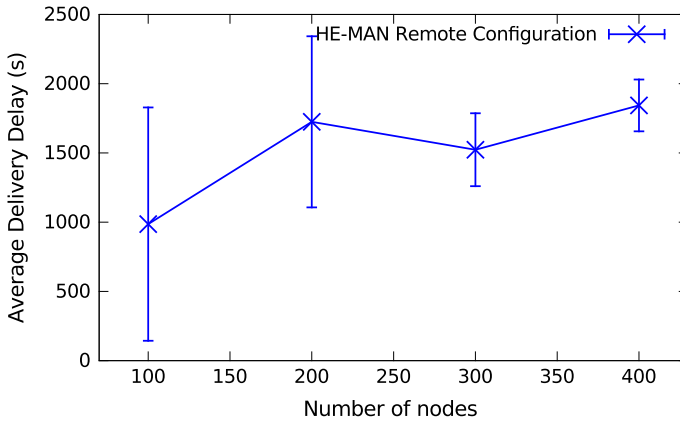


Fig. 11 Delivery delay for configuration messages

5.3 Configuration

Figure 11 shows that the delivery delay tends to increase as the network becomes denser. The explanation to this is the increase of forwarding opportunities that comes with a denser network—a higher number of forwards frequently results in a higher communication delay. The messages that are delivered after a high number of forwarding have a high associated delay, which results in the total average delivery delay being increased.

The second metric is the delivery ratio, aiming to assess the probability of a configuration being actually delivered to its destination. Simulations showed that the delivery ratio of remote configuration messages can be as low as 10.25% in a VDTN with 100 nodes, and as high as 30.5% in a VDTN with 400 nodes. The increase of the delivery ratio in a denser VDTN is expected due to the increase of forwarding opportunities for the configuration messages.

It is important to highlight that the achieved results in the experiments with HE-MAN's remote configuration were obtained with the usage of the Epidemic with Oracle routing protocol, which provides the highest possible delivery ratio and infinite buffer size, with each message being delivered as soon as theoretically possible. This fact emphasizes the existence of a hard limit for remote configuration viable. That also reinforces the necessity for local configuration techniques whenever possible, leaving the remote configuration as a last resource.

6 Conclusions and Future Work

This article proposed an architecture designed to establish a hierarchical management of the network. The proposed architecture, called HE-MAN, provides the means for connected group detection and clusters formation, establishing a Mid-Level Manager (MLM) for each connected group of vehicular nodes. The group detection and MLM election processes are made possible through a new clustering

algorithm that was designed specifically for the HE-MAN architecture. Once the management topology is hierarchically organized, our proposed techniques for monitoring and configuration take advantage of local communication for optimizing the management tasks, aiming to improve the overall responsiveness and reliability of the management systems.

Simulations showed that our proposed clustering algorithm can organize connected groups successfully, having as one of the main objectives the maximization of the amount of time a node spends connected to VDTN clusters. By doing so, nodes can be locally monitored and configured by MLMs for a maximized amount of time, making these management tasks more reliable and efficient. MLMs were able to gather monitoring data from the agents in their clusters for forwarding it to the TLM, achieving a delivery ratio and notification delay only slightly inferior to the considered baseline, which consisted of every agent sending their monitoring data directly to the central manager of the VDTN. Finally, the proposed remote configuration scheme allows the TLM to issue commands to remote nodes of the VDTN while having the means for verifying whether the situation of the node upon the reception of the new configuration remains consistent to the knowledge the TLM has about it or not.

In future work we intend to expand our proposed architecture to explore other management functions (e.g., accounting, performance, and security). We also want to prototype an implementation of the HE-MAN architecture in order to carry evaluations with real devices and applications.

References

1. National Highway Traffic Safety Administration, Federal motor vehicle safety standards: vehicle-to-vehicle (V2V) communications, http://www.nhtsa.gov/staticfiles/rulemaking/pdf/V2V/V2V-ANPRM_081514.pdf (2014)
2. Pereira, P.R., Casaca, A., Rodrigues, J.J.P.C., Soares, V.N.G.J., Triay, J., Cervello-Pastor, C.: From Delay-Tolerant Networks to vehicular Delay-Tolerant Networks. *IEEE Commun. Surv. Tutor.* **14**(4), 1166–1182 (2012)
3. Salvador, E.M., Macedo, D.F., Nogueira, J.M., Almeida, V.D.D., Granville, L.Z.: Hierarchy-based monitoring of vehicular Delay-Tolerant Networks. In: *Consumer Communications and Networking Conference (CCNC)*, 2016 13th Annual IEEE, (2016)
4. Birrane, E., Cole, R.G.: Management of disruption-tolerant networks: a systems engineering approach. In: *SpaceOps Conference*, (2010)
5. C. Peoples, G. Parr, B. Scotney, A. Moore: Context-aware policy-based framework for self-management in Delay-Tolerant Networks: a case study for deep space exploration. *IEEE Commun. Mag.* **48**(7), 102–109 (2010)
6. Sachin, K., Mishra, A., Cole, R.G.: Configuration management for DTNs. In: *IEEE Globecom Workshops*, pp. 589–594 (2010)
7. Papalambrou, A., Voyiatzis, A., Serpanos, D., Soufrilas, P.: Monitoring of a DTN2 network. In: *Baltic Congress on Future Internet Communications (BCFIC Riga)*, pp. 116–119 (2011)
8. Birrane, E., Ramachandran, V.: Delay Tolerant Network management protocol. draft-irtf-dtnrg-dt-nmp-01 (2014)
9. Nobre, J., Duarte, P., Granville, L., Tarouco, L., Bertinatto, F.: On using P2P technology to enable opportunistic management in DTNs through statistical estimation. In: *IEEE International Conference on Communications (ICC)*, pp. 3124–3129 (2014)

10. Ferreira, B .F., Rodrigues, J .J., Dias, J .A., Isento, J .N.: Man4VDTN—a network management solution for vehicular Delay-Tolerant Networks. *Comput. Commun.* **39**, 3–10 (2014)
11. Dias, J.A.F.F., Rodrigues, J.J.P.C., de Paz, J.F., Corchado, J.M.: MoM—a real time monitoring and management tool to improve the performance of vehicular Delay Tolerant Networks. In: Eighth International Conference on Ubiquitous and Future Networks (ICUFN) **2016**, pp. 1071–1076 (2016)
12. Palazzi, C.E., Bujari, A., Marfia, G., Rocchetti, M.: An overview of opportunistic ad hoc communication in urban scenarios. In: 2014 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET), pp. 146–149 (2014). <https://doi.org/10.1109/MedHocNet.2014.6849117>
13. Gaito, S., Maggiorini, D., Rossi, G., Sala, A.: Bus switched networks: an ad hoc mobile platform enabling urban-wide communications. *Ad Hoc Netw.* **10**(6), 931–945 (2012). <https://doi.org/10.1016/j.adhoc.2011.12.005>
14. Keränen, A., Ott, J., Kärkkäinen, T.: The ONE simulator for DTN protocol evaluation. In: 2nd International Conference on Simulation Tools and Techniques (2009)
15. Wang, F., Xu, Y., Zhang, H., Zhang, Y., Zhu, L.: 2FLIP: a two-factor lightweight privacy preserving authentication scheme for VANET. *IEEE Trans. Veh. Technol.* **65**(2), 896–911 (2016)
16. Wolny, G.: Modified DMAC clustering algorithm for VANETs. In: 3rd International Conference on Systems and Networks Communications, 2008. ICSNC '08. (2008) pp. 268–273
17. Kargl, F., Papadimitratos, P., Buttyan, L., Mter, M., Schoch, E., Wiedersheim, B., Thong, T.V., Calandriello, G., Held, A., Kung, A., Hubaux, J.P.: Secure vehicular communication systems: implementation, performance, and research challenges. *IEEE Commun. Mag.* **46**(11), 110–118 (2008)
18. Biswas, S., Tatchikou, R., Dion, F.: Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety. *IEEE Commun. Mag.* **44**(1), 74–82 (2006)
19. Kato, S., Hiltunen, M., Joshi, K., Schlichting, R.: Enabling vehicular safety applications over LTE networks. In: International Conference on Connected Vehicles and Expo (ICCVE) **2013**, pp. 747–752 (2013)

Ewerton M. Salvador is an Assistant Professor at Centro Universitário de João Pessoa (UNIPÊ), Brazil. He holds a Ph.D. degree in Computer Science from Universidade Federal de Minas Gerais (2016). He also holds a M.Sc. in Computer Science from Universidade Federal do Rio Grande do Sul and a B.Sc. in Computer Science from Universidade Federal da Paraíba. His areas of interest include computer networks management, vehicular networks and Delay-Tolerant Networks.

Daniel F. Macedo is an Assistant Professor in the Computing Department at UFMG, Brazil. He has productivity in research scholarship (Bolsa PQ CNPq) level 2. He holds a Ph.D. in computer science from Université Pierre et Marie Curie-Paris VI (2009). He also holds a M.Sc. and a B.Sc. in Computer Science from Universidade Federal de Minas Gerais (2006).

José Marcos S. Nogueira is a full professor of Computer Science at the Federal University of Minas Gerais (UFMG), Brazil. He received a B.S. degree in Electrical Engineering and a M.S. degree in Computer Science from UFMG (1979), and a Ph.D. degree in Electrical Engineering from University of Campinas, Brazil (1985). His areas of interest and research include computer networks, computer networks management, mobile and vehicular networks, and opportunistic networks.

Reproduced with permission of copyright owner.
Further reproduction prohibited without permission.