



OpenFlow based Topology Discovery Service in Software Defined Vehicular Networks: limitations and future approaches

Soufian Toufga, Slim Abdellatif, Philippe Owezarski, Thierry Villemur

► To cite this version:

Soufian Toufga, Slim Abdellatif, Philippe Owezarski, Thierry Villemur. OpenFlow based Topology Discovery Service in Software Defined Vehicular Networks: limitations and future approaches. IEEE Vehicular Networking Conference (VNC), Dec 2018, Taipei, Taiwan. hal-01951191

HAL Id: hal-01951191

<https://laas.hal.science/hal-01951191>

Submitted on 11 Dec 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

OpenFlow based Topology Discovery Service in Software Defined Vehicular Networks : limitations and future approaches

Soufian Toufga*, Slim Abdellatif†, Philippe Owezarski*, Thierry Villemur‡

* LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France

† LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse, France

‡ LAAS-CNRS, Université de Toulouse, CNRS, UT2J, Toulouse, France
{toufga, slim, owe, villemur}@laas.fr

Abstract—This paper presents our ongoing work towards the definition of an effective topology discovery service for SDN based hybrid vehicular networks. The main contributions of this paper are threefold: (1) an analysis of the main limitations of the “de facto” topology discovery service when applied to vehicular networks; (2) a quantitative assessment of its traffic overhead and the required computing resources at the controller to show the scalability issues that it raises; and (3) an insight into our design principles for an effective topology discovery service for SDN-based hybrid vehicular networks.

Index Terms—*Software Defined Vehicular Network, Topology discovery service, OFDP, OpenFlow.*

I. INTRODUCTION

As part of an ITS (Intelligent Transport System), next generation vehicles will be equipped with multiple interfaces of different technologies to communicate with other ITS entities, i.e. vehicles, pedestrians, infrastructure - RSU (Road Side Units), BS (Base Stations), Cloud. DSRC (Dedicated Short Range Communication) and LTE (Long Term Evolution) are considered as the two major complementary networking vehicular technologies to support effectively the various Quality of Service (QoS) requirements of ITS services. Hybrid architectures combining these technologies that follow the SDN paradigm are gaining momentum and are being developed with the intention of achieving integrated network control and improved along with “on the fly” and dynamic network programmability [1], [2], [3].

A crucial piece of any SDN system is the topology discovery service, implemented by the controller, whose objective is to build and follow-up the network state and topology, from which, customized representations of the underlying network are exposed to network control functions. Recent studies have highlighted many limitations of the “de facto” discovery protocol (Openflow and OFDP : Openflow Discovery Protocol) when applied to wireless networks in terms of security, overhead and capabilities [4], [5], [6]. The objective of this work is to further analyze the limitations of Openflow and OFDP with respect to some of the challenging specificities of vehicular network, namely the high density of wireless nodes and their potential high mobility. From these limitations, we outline some of the design guidelines for an efficient topology discovery service for

vehicular networks.

This paper is organized as follows. Section II presents the Software Defined Vehicular Network architecture and the assumptions considered in our study. Section III analyzes the limitations of the Openflow based topology discovery approach for SDVN (Software Defined Vehicular Network). Section IV presents the experimental results. Section V highlights our future directions in order to design an efficient topology discovery mechanisms in SDVN based on the limits analysis results.

II. SDN BASED HYBRID VEHICULAR NETWORK : OUR REFERENCE ARCHITECTURE & ASSUMPTIONS

Figure 1 depicts a typical SDN based hybrid vehicular network composed of heterogeneous routing nodes: mobile nodes (Vehicles), static nodes - BSs (communicating with LTE) and RSUs (communicating with DSRC). We assume that, vehicles are equipped with two main DSRC and LTE interfaces. RSU entities are interconnected via wired links.

We consider a two-level hierarchical control plane, composed of controllers that manage BSs and RSUs (level 1 controllers), and an orchestrator controller (level 2 controller) used to designate the global logical controller that coordinates lower level controllers. Based on the information exposed by level 1 controllers, the orchestrator is able to jointly control both the LTE and DSRC networks. SDN controllers are connected via wired links to BS / RSU entities and are located in data centers, in clouds or on fog nodes. For scalability reasons, a controller can only control a limited number of routing nodes (RSUs / BSs / Vehicles) [7]. As a result, a vehicle may typically be successively associated to different controllers.

III. WHY OPENFLOW BASED TOPOLOGY DISCOVERY IS NOT ADAPTED FOR SDVN ?

A. Topology discovery in SDN/Openflow networks

Network topology discovery covers node discovery and link discovery. In Openflow networks, node discovery is performed by the Openflow protocol while link discovery is provided by OFDP.

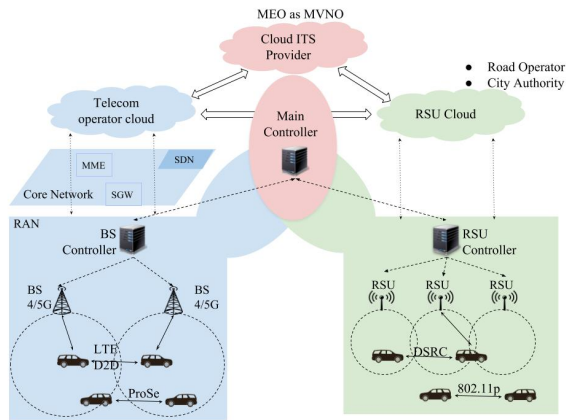


Fig. 1. SDN based hybrid vehicular network architecture.

- **Node Discovery** : During its boot process, an Openflow switch establishes an Openflow session with its preconfigured SDN controller. During the session initiation, the switch transmits to the controller information about its identity (Datapath ID), its characteristics (e.g. capacity of the flow tables), its active ports and the functionalities supported through the exchange of "Features Request/Reply" messages. From there, a periodic Echo (Request / Reply message) procedure initiated by the controller is used to maintain the state of the node. Port-Status messages may be initiated by an OF switch to inform the controller in case a state port change.
- **Link Discovery** : From the information obtained during session initiation, the SDN controller is aware of the different active ports of each active switch. Following the OFDP procedure, the controller sends regularly LLDP (Link Layer Discovery Protocol) packets encapsulated in Openflow messages on a per active port basis. The Openflow message that encapsulates the LLDP packet (that conveys the switch ID and the concerned port ID) instructs the switch, via an Openflow rule, to relay the LLDP packet via the chosen port. When received by the adjacent switch, the LLDP is encapsulated in an Openflow message and sent back to the controller with its switch ID and the receiving port ID. By doing so, a link connecting both switches via the sending and receiving ports is discovered.

- Link Discovery :** From the information obtained during session initiation, the SDN controller is aware of the different active ports of each active switch. Following the OFDP procedure, the controller sends regularly LLDP (Link Layer Discovery Protocol) packets encapsulated in Openflow messages on a per active port basis. The Openflow message that encapsulates the LLDP packet (that conveys the switch ID and the concerned port ID) instructs the switch, via an Openflow rule, to relay the LLDP packet via the chosen port. When received by the adjacent switch, the LLDP is encapsulated in an Openflow message and sent back to the controller with its switch ID and the receiving port ID. By doing so, a link connecting both switches via the sending and receiving ports is discovered.

B. Main limitations of Openflow/OFDP based topology discovery service for SDVN

The Openflow/OFDP based topology discovery service was originally proposed with wired networks in mind. It clearly suffers from some limitations when it comes to wireless networks with mobile nodes. These are presented below.

1) *Node Discovery:*

- Preconfigured SDN controller ID: The Openflow/OFDP discovery service relies on a continuous living Openflow

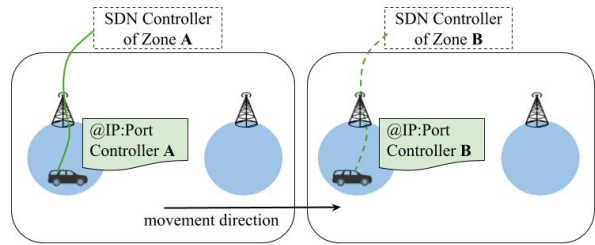


Fig. 2. SDN controller change caused by car movement.

session between each active switch and its associated controller(s). As shown in figure 2, in a vehicular network, a moving vehicle can leave the area where its initial SDN controller operates (Area A in figure 2) to get into a second area under the responsibility of another controller (e.g. zone B, in Figure 2), jeopardizing its Openflow session and consequently the topology discovery procedures described above. An SDN controller discovery mechanism and a handover mechanism are needed to ensure Openflow session survival and the exchange between controllers of vehicles related topology discovery attributes.

- Presence of non-openflow nodes or loss of connectivity with the controller: In both cases, the proposed service, does not report such nodes into the network topology built on the controller despite their presence and their ability to support and forward network traffic.

2) *Link Discovery:*

- Presence of Heterogeneous links: In a vehicular network architecture, some Routing Nodes (RSUs and BSs) have wired as well as wireless links. Referring to [8], the wired link discovery mechanism is a bit different from that used for wireless links. The Link discovery procedure must take into account the presence of different types of links in the same node and compute them differently.
- Link discovery with the presence of Openflow and non-Openflow nodes: The link discovery procedure of OFDP is operational when nodes are OpenFlow enabled. When the network includes non-openflow enabled nodes, [9] proposes that the SDN controllers use the Broadcast Domain Discovery Protocol (BDDP) instead of LLDP. The specificity of these messages is that their destination address is set to broadcast instructing the non-Openflow nodes to flood the packet (received from the transmitting adjacent node) until it is received by an openflow-compatible node where it is treated as a normal LLDP packet. Broadcasting topology discovery messages is prohibitive in terms of resource wastage in a vehicular network context.
- Scalability: Link discovery but also node discovery in Openflow/OFDP require respectively a periodic transmission of Openflow messages on a per active port and a per node basis. This periodicity should be adapted to the degree of node mobility. The higher the mobility, the lower the periodicity, and the more significant the discovery traffic. Even if a modification of the OFDP mechanism has

- **Link discovery with the presence of Openflow and non-Openflow nodes:** The link discovery procedure of OFDP is operational when nodes are OpenFlow enabled. When the network includes non-openflow enabled nodes, [9] proposes that the SDN controllers use the Broadcast Domain Discovery Protocol (BDDP) instead of LLDP. The specificity of these messages is that their destination address is set to broadcast instructing the non-Openflow nodes to flood the packet (received from the transmitting adjacent node) until it is received by an openflow-compatible node where it is treated as a normal LLDP packet. Broadcasting topology discovery messages is prohibitive in terms of resource wastage in a vehicular network context.

- **Scalability:** Link discovery but also node discovery in Openflow/OFDP require respectively a periodic transmission of Openflow messages on a per active port and a per node basis. This periodicity should be adapted to the degree of node mobility. The higher the mobility, the lower the periodicity, and the more significant the discovery traffic. Even if a modification of the OFDP mechanism has

been proposed in the literature in order to send a single packet to each switch [6], the generated traffic remains significant and, also, useless when the network does not experience any change.

3) *Lack of useful node and link attributes:* Some vehicular network control functions (routing, mobility management, load balancing, network technology selection, etc.) may benefit from information related to nodes mobility (speed, position, direction) or wireless port characteristics (operating channel, transmission power, receiver sensitivity, etc.) as well as wireless link attributes (latency, reliability, etc.) to make informed control decisions. The Openflow/ODFP discovery service does not collect any of such attributes.

IV. ASSESSING THE OVERHEAD OF OPENFLOW/ODFP NETWORK DISCOVERY SERVICE IN A VEHICULAR NETWORK CONTEXT

The goal is to analyze the impact of node mobility and node density, peculiar to vehicular networks, on the network traffic overhead generated by the Openflow/ODFP discovery service as well as the required computing resources at the Openflow controller. The rationale of these experimentations is to demonstrate that the increase of the number of vehicles and their velocity induces scalability issues.

A. Simulation Settings

The simulated network consists of 13 RSU entities interconnected via wired links, each with a communication range of 500 m covering a map area of 2000 x 2000 m², all under the control of an SDN controller. The road network is a grid type network with each road segment of 500 m length (see Fig. 3). We use 10-30 vehicles (step of 5) with varied velocity 5-25 m/s (step of 5). Each vehicle starts at a random location and then moves with the configured velocity following the road network by randomly choosing a direction at each intersection.

To the best of our knowledge and at the time of writing this paper, the only emulator that integrates real SDN/ OpenFlow Controller with programmable wireless network is Mininet-Wifi [10], and given that the current ODFP implementations are not compatible with the wireless link extensions added in MiniNet-Wifi, we used MiniNet considering the following assumptions: Each vehicle / RSU entity represents an OpenFlow switch, and the vehicle movement from one RSU to another results in the removal of the link established with the first RSU, and the creation of a new link with the second RSU. In addition, the existence of a vehicle in another's communication range enforces the creation of a V2V link between them.

In our scenario, we are interested in the dynamicity of the topology (link addition and removal caused by node mobility as explained above). This network model was implemented, using the MiniNet API and we used Floodlight (v1.2) [11] SDN Controller with its implementation of the Openflow/ODFP topology discovery service. The SDN controller and the mininet emulator (v 2.2.2) [12] are running on separated Ubuntu 16.04 VMs with 2 Go RAM and a dual core CPU (Intel ® Core (TM) i5-4310U 2 Ghz). We measure the CPU utilization and

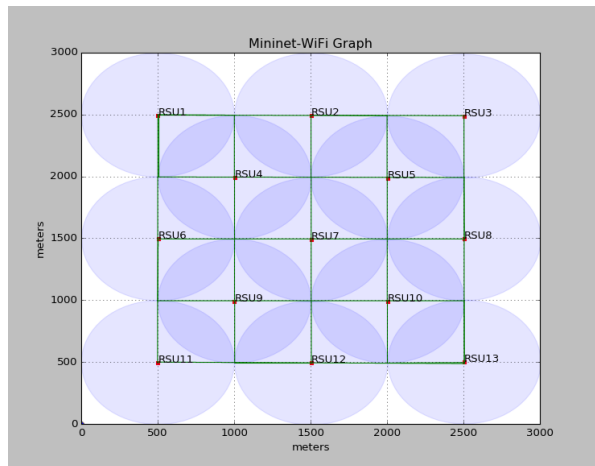


Fig. 3. Simulated vehicular network infrastructure.

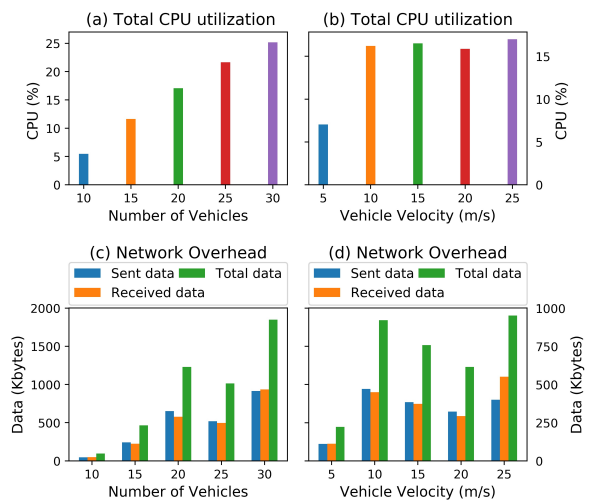


Fig. 4. Simulation results.

the amount of control data exchanged between the controller and all active nodes. Each set of simulations is averaged over 3 runs, each running for 5 minutes.

B. Simulation Results

Two kind of simulations were first realized. Firstly, all the vehicles move with the same speed (10 m/s) and their number varies from 10 to 30 with a step of 5. In a second phase, we keep the same number of vehicles (20) and their speed varies from 5 to 25 m/s with a step of 5. Figures 4(a,b) and 4(c,d) show respectively the variation of the CPU utilization and network overhead as a function of the number and the speed of vehicles.

We notice that the CPU utilization increases in a linear way and proportionally to the number of vehicles. This is explained by the fact that ODFP operates in "one-to-one communication" mode (controller to each node) which multiplies by the number of controlled nodes the number of actions performed by the SDN controller to discover and maintain the topology. This

also holds for network overhead that depends mainly on the number of messages exchanged with each node and the number of created links.

As expected, when increasing the speed of vehicles, the CPU utilization and the network overhead increase as well. They do not grow linearly as a function of vehicle velocity. This is explained by the fact that the speed variation induces a longer variation in the number of added or removed links. In fact, this number increases according to the vehicle speed for V2I links but remains random for the V2V links that depend mainly on the location of each vehicle at a given time. We also notice that the variation in the number of vehicles (density) has a more significant impact in terms of scalability than their mobility.

V. DISCUSSION AND FUTURE APPROACHES

With the recent interest in applying an SDN approach to vehicular networks, the design of an effective topology discovery service is a crucial step towards programmable vehicular communications.

In order to avoid the node discovery limitations mentioned in the previous section, a dynamic discovery of SDN controllers appears more suited in a vehicular network context. Another approach is the use of a paradigm that separates network node identification and localization, such as the LISP protocol.

In addition, to avoid the scalability issue of existing discovery approaches, we need to rethink the link discovery process. Indeed, the use of a "one-to-one communication" approach presents scalability issues (as demonstrated in the previous section). One research direction could be the adoption of a "one-to-many" communication approach, by defining a vehicle cluster with a Cluster Head that handles the communication with the controller, instead of communicating with each node separately. By doing so, the network overhead between the controller and the network nodes is reduced. However, additional traffic will be added between the vehicles. Hence, an efficient mechanism is needed for aggregating the information to be sent to the controller while reducing the load and maintaining a consistent discovery.

Another point to consider in the design of a topology service is the LLDP message sending interval. This frequency is statically configured according to the dynamicity of the network: the more dynamic the network, the higher its frequency. However in a vehicular network, the dynamicity of network varies depending on several parameters (the density and speed of vehicles (vary according to the time of day (day, night, peak hours)) and also as a function of the environment (urban/suburban areas)). An interesting direction will be to have an adaptable frequency depending on the environment. On the other hand, considering the use of an event-based approach instead of a periodic approach can be more interesting, as far as only information about added/removed links will be notified to the controller.

Finally, some mobility-related data (velocity, trajectory, etc.) required by the network control functions can be recovered from the ITS service provider clouds instead of being sent through the SDN control plane. Indeed, these data are already collected by these service providers on behalf of the development of their proposed services.

VI. CONCLUSION AND FUTURE WORK

With the purpose of effectively supporting the various ITS services, this paper aims at presenting our ongoing work towards the definition of an effective topology discovery service which represents a crucial component in Software Defined Vehicular Network system.

We have demonstrated that the "de facto" OpenFlow based topology discovery service is not adapted for Software Defined Vehicular Networks and we proposed some directions to follow in order to address the highlighted limits and to design an efficient topology discovery service more suited to vehicular networks.

ACKNOWLEDGMENT

This work is funded by Continental Digital Service France (CDSF) in the framework of the eHorizon project.

REFERENCES

- [1] X. Ge, Z. Li, and S. Li, "5g software defined vehicular networks," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 87–93, 2017.
- [2] I. Ku, Y. Lu, M. Gerla, R. L. Gomes, F. Ongaro, and E. Cerqueira, "Towards software-defined vanet: Architecture and services," in *2014 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET)*, pp. 103–110, June 2014.
- [3] S. TOUFGA, P. Owezarski, S. Abdellatif, and T. Villemur, "An SDN hybrid architecture for vehicular networks: Application to Intelligent Transport System," in *9th European Congress on Embedded Real Time Software And Systems (ERTS)*, (Toulouse, France), p. 8p., Jan. 2018.
- [4] S. Khan, A. Gani, A. W. A. Wahab, M. Guizani, and M. K. Khan, "Topology discovery in software defined networks: Threats, taxonomy, and state-of-the-art," *IEEE Communications Surveys Tutorials*, vol. 19, pp. 303–324, Firstquarter 2017.
- [5] A. Azzouni, R. Boutaba, T. M. T. Nguyen, and G. Pujolle, "softdp: Secure and efficient topology discovery protocol for SDN," *CoRR*, vol. abs/1705.04527, 2017.
- [6] F. Pakzad, M. Portmann, W. L. Tan, and J. Indulska, "Efficient topology discovery in OpenFlow-based software defined networks," *Computer Communications*, vol. 77, pp. 52–61, mar 2016.
- [7] M. Karakus and A. Durrezi, "A survey: Control plane scalability issues and approaches in software-defined networking (SDN)," *Computer Networks*, vol. 112, pp. 279–293, jan 2017.
- [8] L. Chen, S. Abdellatif, P. Berthou, B. Nougancke, and T. Gayraud, "A Generic And Configurable Topology Discovery Service For Software Defined Wireless Multi-Hop Network," in *15th ACM International Symposium on Mobility Management and Wireless Access*, (Miami, United States), p. 4p., Nov. 2017.
- [9] C.-P. C. F.-F. A. Ochoa-Aday, L., "Current trends of topology discovery in openflow-based software defined networks," 2015.
- [10] R. R. Fontes, S. Afzal, S. H. B. Brito, M. A. S. Santos, and C. E. Rothenberg, "Mininet-wifi: Emulating software-defined wireless networks," in *2015 11th International Conference on Network and Service Management (CNSM)*, pp. 384–389, Nov 2015.
- [11] "Floodlight project." <http://www.projectfloodlight.org/floodlight/>. Accessed: 2018-08-01.
- [12] "Mininet emulator." <http://mininet.org/>. Accessed: 2018-08-01.