Software Defined Architecture for VANET: A Testbed Implementation with Wireless Access Management

Gökhan Seçinti, Berk Canberk, Trung Q. Duong, and Lei Shu

ABSTRACT

Toward ITS, academia and industry aim to utilize all possible radio access technologies in order to support reliable services and applications in VANETs. Thus, the inclusion of already deployed Wi-Fi networks in VANET topology is a crucial step for the next generation vehicular networks. However, the VANET topology also requires preservation of the features already offered by DSRC and the core cellular network. As a result, the coexistence of multiple different access technologies results in high complexity in terms of the control and management of the network infrastructure. To this end, software defined networking provides a promising opportunity to simplify the management and control of clumsy network infrastructures by decoupling the data and control planes in order to provide elasticity for current networks. In this article, we propose an architectural model that exploits this opportunity in order to enhance VANET with Wi-Fi access capability. Moreover, we offer a novel software defined VANET architecture that consists of soft OpenFlow switches with Wi-Fi capabilities as both roadside units and vehicles. In particular, we first investigate existing test tools and environments for software defined wireless networks and also supply a novel testbed architecture in order to provide a feasible test environment for evaluating the proposed architecture. Additionally, we propose a Wireless Access Management (WAM) protocol that provides wireless host management and basic flow admission with respect to the available bandwidth to validate the capability of the offered architecture. The observation results of the deployed testbed prove the conformity of the offered 802.11 architecture to the VANET.

INTRODUCTION

Intelligent transportation systems (ITS) are supported by governments, industry, and academia in order to provide safer and more efficient transportation environments [1]. However, the high mobility of traffic and the dynamic nature of the communication environment has stood as a great challenge to reliable vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communication to improve efficiency of transportation systems. For this reason, vehicular ad hoc networks (VANETs) aim to leverage various radio access technologies cooperatively to meet these demands while overcoming the challenges.

The already widely deployed 802.11 (Wi-Fi) networks are perfect candidates for VANETs. However, the growth of networks and simultaneous use of different radio access technologies present severe difficulties in terms of infrastructure control and management. Specifically, in large-scale deployment of legacy wireless networks, a significant amount of the operational expense (OpEx) is spent on the management of infrastructure devices. Major challenges encountered in mobile wireless access networks include the configuration of the wireless access points, the management of wireless hosts, forwarding the traffic generated by wireless hosts, and management of the radio resources, such as dynamic channel assignment and power planning. Several approaches have been proposed to overcome the aforementioned problems; however, they are not without disadvantages. The existing wireless management solution, the Control and Provisioning of Wireless Access Points (CAPWAP) protocol [2], lacks a traffic offload mechanism. Furthermore, bandwidth and the processing effort spent ON the control plane traffic in CAPWAP becomes underutilized because of traffic overhead.

As an alternative approach, software defined networking (SDN) has been adopted. SDN separates the network control function (the control plane) from the forwarding functions (the data plane), allowing us to control all network functionalities by a controller in a centralized manner. This approach enables an opportunity for a manageable network where it is feasible to implement new protocols. Additionally, SDNs have the ability to install rules to the end devices once the forwarding path for a specific traffic type has been discovered by the controller and installed as a flow to the flow table of the OpenFlow Switch. The data traffic can then be offloaded from the controller. With all the advantages, SDN is considered as a strong solution candidate for wireless access management.

A software defined VANET (SD-VANET) testbed includes soft-switches at the data plane, which are composed of Raspberry Pi as hard-

Software defined networking provides a promising opportunity to simplify the management and control of clumsy network infrastructures by decoupling the data and control planes in order to provide elasticity for current networks. The authors propose an architectural model that exploits this opportunity in order to enhance VANET with Wi-Fi access capability.

Gökhan Seçinti and Berk Canberk are with Istanbul Technical University; Trung Q. Duong is with Queen's University Belfast; Lei Shu is with Guangdong University of Petrochemical Technology.

Digital Object Identifier: 10.1109/MCOM.2017.1601186

0163-6804/17/$25.00

SD-Wireless Access Management simply benefits this virtualization capability of the soft-switches and provides wireless access virtualization in order to realize control and management mechanisms without increasing signalling overhead or introducing any middle-layers to existing OpenFlow protocol.

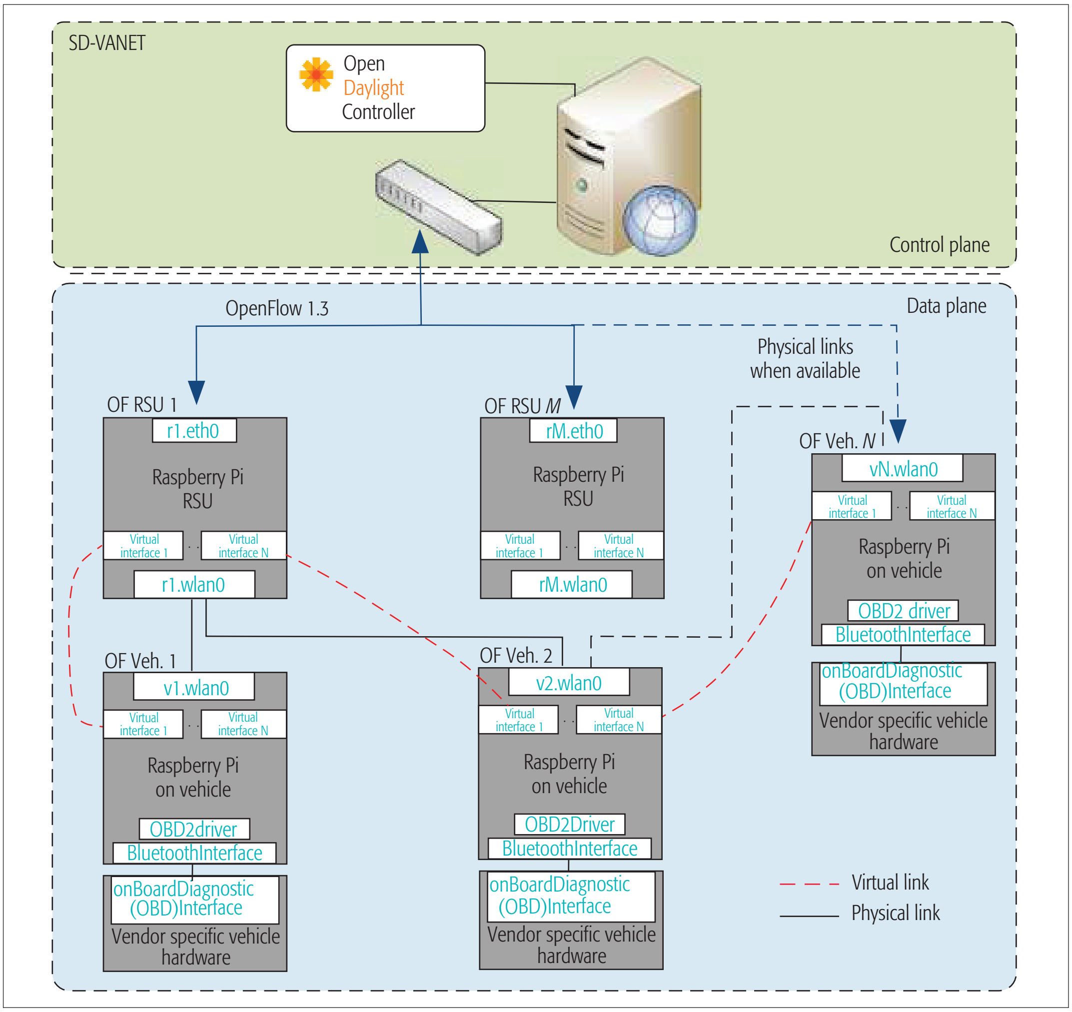

Figure 1. The proposed software defined VANET architecture.

ware and OpenvSwitch (OVS) as the main software component to work as both a roadside unit (RSU) and a vehicle access point. Moreover, these soft-switches are enhanced with virtual port capability, which defines a virtual wireless access port for each client (vehicle and RSU). This virtual port capability provides improved interface to the controller plane, which enables control and management commands to be implemented. Consequently, SD-Wireless Access Management (SD-WAM) simply enhances this virtualization capability of the soft-switches and provides wireless access virtualization in order to realize control and management mechanisms without increasing signaling overhead or introducing any middle layers to the existing OpenFlow protocol.

Utilizing SDN architecture in vehicular and mobile networks in order to benefit currently deployed network infrastructure has already drawn the attention of academia in recent years. In [3], the authors provided an overview for software defined mobile networks and identified current research approaches and challenges. In [4], the authors proposed a framework based on the cloud radio access network (Cloud-RAN) in order to utilize multiple RAN technologies through virtulization in vehicular networks. Moreover, in [5], the authors proposed a software defined mechanism named SERVICE to propose delay optimality. In this manner, they formulated the delay optimality through virtual resource scheduling by using a partially observed Markov decision process. Also, the authors defined a novel dissatisfaction parameter and proposed an offloading mechanism in order to satisfy quality of service (QoS) demands of mobile users in software defined heterogeneous networks in [6]. In addition, the authors in [7] proposed to utilize a slicing mechanism simply based on vehicle driving directions in order to achieve multi-tenant isolation in vehicular networks. However, none of the above studies focused on building an SD-VANET to enable rapid deployment and testing of high-level applications and services.

Although these approaches seem beneficial, they also require huge changes and definitions of new protocols in order to be implemented.

Additionally, there are various studies which focus on the network virtualization approach in SDN in order to tackle management challenges in the wireless environment. In [8], virtualization strategies for the wireless environment were investigated and a novel resource description methodology was proposed to better address challenges in software defined wireless networks. Moreover, in [9], the authors defined and implemented a network virtualization mechanism called "FlowVisor," which introduces a middleware between the data and control planes and creates an opportunity to orchestrate different underlying networks in a single physical data plane. However, this middleware requires computational capabilities to reshape the control packets among the two planes. Thus, it creates another bottleneck problem for SDN architecture where there is already a troublesome bottleneck problem due to the singularity of the controller. To overcome this problem, the authors in [10] proposed a distributed version of FlowVisor. However, their method significantly increases the latency of the control packets when distributing the load of FlowVisor since a database is introduced in order to distribute the knowledge among middle entities.

As new technologies emerge, it is getting complicated for researchers to implement and test their novel protocols and approaches on current physical infrastructures, whose functions are restricted by their vendor-specific nature. The gap between academia and industry has been widened with the difficulty of testing and analyzing new protocols for every new technology, there are a limited number of studies that focus on filling this gap by introducing testbeds and analytical tools for various environmental settings. In [11], a West-East-bridge-based testbed for SDN is introduced in order to manage different domains and inter-domain communication. Two different inter-domain routing protocols to validate their testbed were also proposed. Similarly, the authors designed a testbed to analyze mobile crowd sensing by building a test network in the University of Bologna comprising 300 students as participants in [12]. Furthermore, in [13], the authors investigated existing experimental platforms and testbeds for wireless ad hoc networks and provided insights about the current limitations and challenges for future studies. But none of these testbeds specifically focused on how to provide a testing environment for an SDN that consists of wireless channels as the access network.

Motivated by the above discussion, in this article, we propose an SD-VANET testbed architecture that utilizes already deployed WiFi networks in the Istanbut Technical University (ITU) campus. Moreover, we implement software defined network-WAM (SDN-WAM) to validate our testbed. Our proposed architecture aims to minimize the necessity of changes in current network infrastructure; that is, the conjunction of mobile networks and WiFi networks could be handled with the software defined controller without changing any of the underlying infrastructure. More importantly, our SDN-WAM scheme does not necessitate any change at the control-data plane interface (CDPI), and thus does not increase the latency of the control packets.

The rest of the article is organized as follows. The architecture and the implementation details of the D-VANET testbed are given. Then the system model of SDN-WAM is explained, and the performance of DN-WAM is evaluated. Finally, we conclude the article by summarizing the contributions and emphasizing future work.

SD-VANET Testbed

Architecture

The SD-VANET topology depicted in Fig. 1 consists of two components responsible for handling separate domains of the communication network: the control plane and data plane. The control plane is responsible for supplying the network services required by the access networks, while the data plane is responsible for the forwarding of data packets that match the rules installed by the controller.

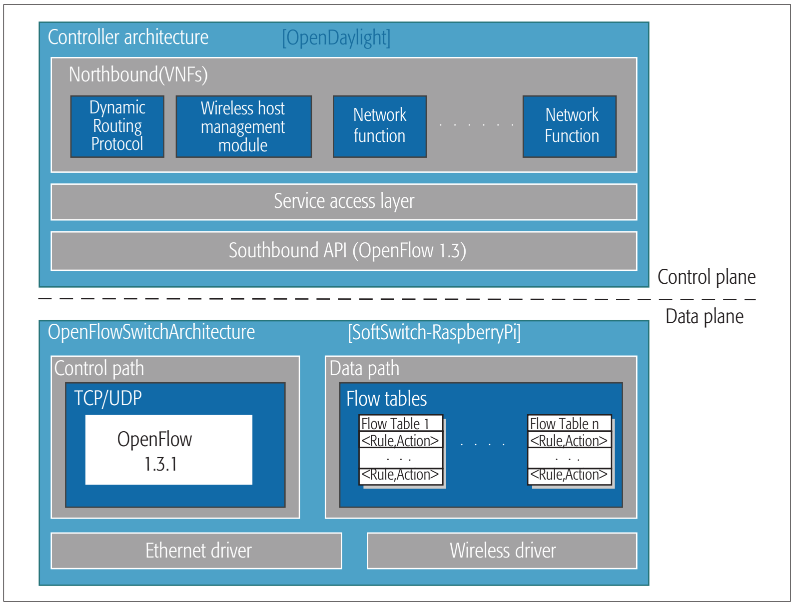

Figure 2. SD-VANET software architecture.

The basic components of an SD-VANET network are the controller and OpenFlow switches. Furthermore, the controller is a software application running on a powerful server computer that has a standard southbound interface such as OpenFlow to communicate with access devices, as depicted in Fig. 1. OpenFlow switches are the access devices of the SD-VANET placed in the backbone of the data plane. As seen in the figure, both RSU and vehicles are modeled by using Raspberry Pi with WiFi capability. There are two types of links defined between vehicles. While physical links represent the actual connection between network interface cards, virtual links provide an opportunity to define overlay networks on which different services and applications are utilized. To this end, SDN-WAM is implemented to validate use of virtual links. In addition, Raspberry Pi on a vehicle is connected through an onboard diagnostic (OBD) interface to the hardware of the vehicle. In this manner, information on the dashboard of the vehicle could be fetched with vendor-specific software.

Our testbed includes both of the aforementioned components of SD-VANET. Moreover, the testbed enables researchers to implement and measure their work, which utilizes either the controller or the OpenFlow Switch.

Implementation

Controller–OpenDaylight: Fundamental requirements of the controller are.

- A southbound application programming interface (API) as seen in Fig. 2 that supports OpenFlow standard to communicate with OpenFlow switches.
- A northbound API that is easy to use and enables rapid development of new network functions.
- Reconfiguration capability

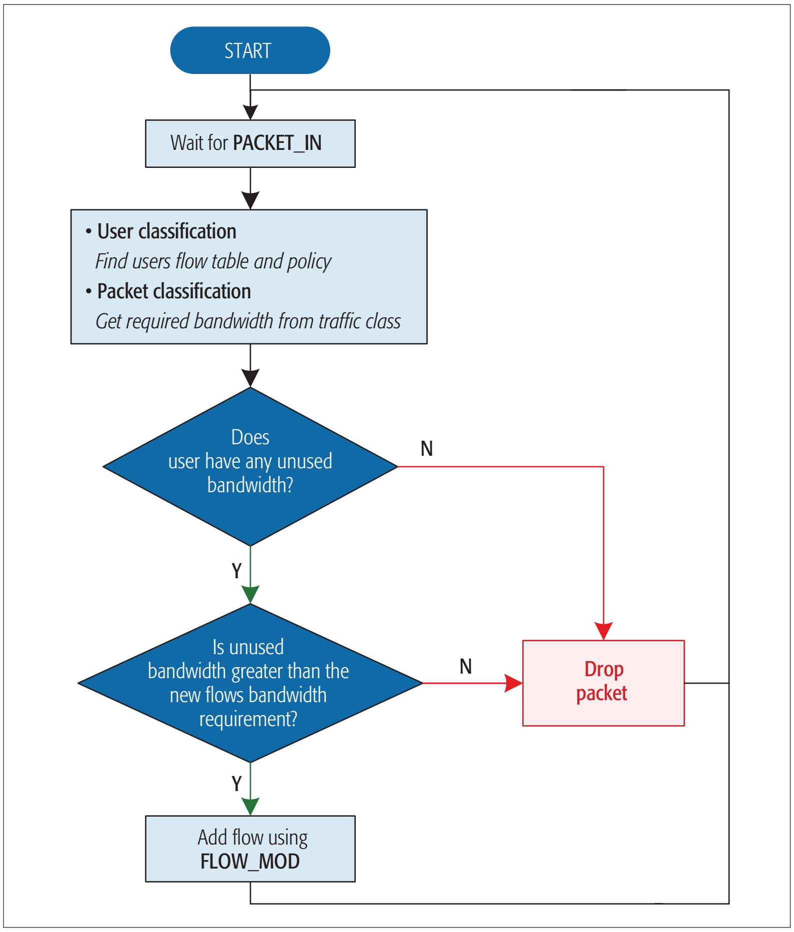

Figure 3. Flow admission implemented on proposed SD-VANET testbed.

The OpenDaylight [14] controller's southbound API supports the OpenFlow protocol. In addition, it is also compatible with the OpenFlow 1.3.1 specification. Moreover, the OpenDaylight controller also provides easy-to-use JAVA APIs to implement network functions on the northbound interface, as seen in Fig. 2. The OpenDaylight controller community consists of technology pioneers of communication networks [14], including Cisco, HP, Extreme Networks, and so on. Having a huge number of high-quality members in the community provides a native support chain for the OpenDaylight controller, which reduces the time required to overcome the obstacles encountered during development. Finally, the OpenDaylight controller's full source code is available free of charge, which allows researchers to implement any kind of customization required during development.

The OpenDaylight controller is installed on a server computer and used as the controller of the testbed because of the aforementioned features.

OpenFlow Switch–OpenvSwitch-Based Raspberry Pi: OpenFlow switches are the access components of the SD-VANET. In addition, the main functionalities of this component are as follows:
- To provide the service chaining infrastructure by supplying flow tables for the controller to install the required <Rule,Action> pairs
- To relay the packets received from the data plane with no matching service chaining rule to the controller using PACKET IN messages defined in OpenFlow to be assessed by the flow admission control algorithm, as seen in Fig. 3

Soft OpenFlow switch implementations are designed to cope with general-purpose inexpensive hardware. As the flow matching algorithm works on a traditional CPU rather than a parallel hardware such as a field programmable gate array (FPGA), a delay is added to the packet forwarding in soft-switch-based SD-VANET. This may lead to a tendency to increase the memory requirements of the general-purpose hardware to handle heavy traffic. For this reason, soft-switches are more likely to be deployed in lightly loaded networks. However, these devices are suitable for gathering useful measurement information for researchers and to prove the applicability of the offered solution to real-time environment even with the software reconfiguration capabilities.

We use Raspberry Pi as the hardware component of the OpenFlow soft-switch implementation. Furthermore, the expansion capability of the device also affected the final decision. The inexpensive hardware makes the device suitable to be used as an OpenFlow switch.

There are two major soft-switch implementations:
- OpenvSwitch(OVS): [1] An open source soft-switch implementation that is natively supported by the Linux kernel. In addition, as the software has integrated components running at the operating system level, this software facilitates the resources in a better way by augmenting the flow tables in Fig. 2 into the existing network device driver.
- CPQD softswitch:[2] An open source user space application that uses tunnels to obtain data from the operating system. This software is useful for inspecting the control plane traffic when there is no need to analyze and orchestrate the traffic.

OpenvSwitch v2.3.90 is ported for Raspberry Pi as the soft-switch implementation of the testbed. OVS provides both the flow table implementation and the OpenFlow communication structure. The Linux kernel is used as the network operating system of the OpenFlow switch. Moreover, the most common Linux distribution used on network devices, OpenWRT [15], is used as the file system of the Raspberry Pi OpenFlow switch. Furthermore, the existence of Raspberry Pi support in OpenWRT makes the file system an even better solution for the testbed. The head version of OpenWRT is used alongside Linux Kernel 4.1. The file system is configured to be compiled with eglibc 2.22 to support the functionalities required by the OVS. The distribution is modified to support a Realtek 5370 Wi-Fi dongle driver. Finally, the Realtek 5370 Wi-Fi dongle is physically attached the device as depicted in Fig. 4, which allows the wireless capability to be attached to the OpenFlow Switch.

Software-Defined-Networking-Based Wireless Access Management

SDN-WAM creates a virtual topology that facilitates the management of the wireless hosts using existing standardized OpenFlow messages. The solution virtualizes each wireless client as an inter-

[1] http://openvswitch.org, accessed on Jan 20, 2106

[2] http://cpqd.github.io/ofsoftswitch13/, accessed on Jan 20, 2106

face of an OpenFlow Switch, as seen in Fig. 1. In addition, management of connected clients are bound to the PORT MOD messages defined in OpenFlow. A PORT MOD message is first introduced in OpenFlow Spec. 1.3.0 and gives the controller the ability to shut the ports of an OpenFlow switch. The offered scheme uses port shutdown messages to provide the controller with the ability to disconnect wireless clients from the OpenFlow switch. In this way, the southbound interface of the controller is preserved as defined in the OpenFlow standard. Furthermore, wireless client virtualization also allows the SDN controller to apply different management policies easily to each client even though clients are connected to the same physical wireless interface.

Implementation

SDN-WAM is implemented on the offered testbed by developing code for both the controller and the OpenFlow switch. A new network function, wireless host control, is developed and added to the OpenDaylight controller using northbound APIs provided by the controller. Wireless host control keeps track of the bandwidth used by the wireless hosts and implements a basic flow admission control depending on the available bandwidth assigned to the user. In addition, the module also relays the user connect/disconnect events to the wireless access management module for evaluation. The flow admission control algorithm depicted in Fig. 3 is triggered whenever a PACKET IN message is received by the controller. PACKET IN messages are generated by the OpenFlow switches when a packet received from the data plane has no matching service chaining rule in flow tables. The algorithm inserts a new service chaining rule using FLOW MOD messages defined in OpenFlow if the flow is admitted. Otherwise, a packet received through PACKET IN is dropped.

The wireless driver of the Raspberry-Pi-based access point has been altered to create a Linux Netdevice for each connected wireless client. In addition, a soft-switch daemon is triggered by connect/disconnect events of wireless clients to generate a PORT MOD message destined to the controller. This message intends to notify the controller about the status of the current network topology. The shutdown handler of the PORT MOD message generated by the controller is bound to the disconnect action for the interfaces virtualizing wireless clients. Furthermore, each packet received from the physical wireless interface is redirected to the relevant Linux Netdevice interface to satisfy the SDN soft switch forwarding structure that runs on Raspberry Pi.

Deployment

The ITU campus was selected as the test environment to deploy the implemented SDN-WAM testbed. The testbed consists of an OpenDaylight SDN controller and 10 Raspberry Pi OpenFlow switches. Furthermore, the controller was placed in the Computer Engineering Department's Communication Laboratory, and the OpenFlow switches were distributed throughout the campus to several locations. The locations of the controller and OpenFlow switches are depicted in Fig. 5. The green marker points to the location of the

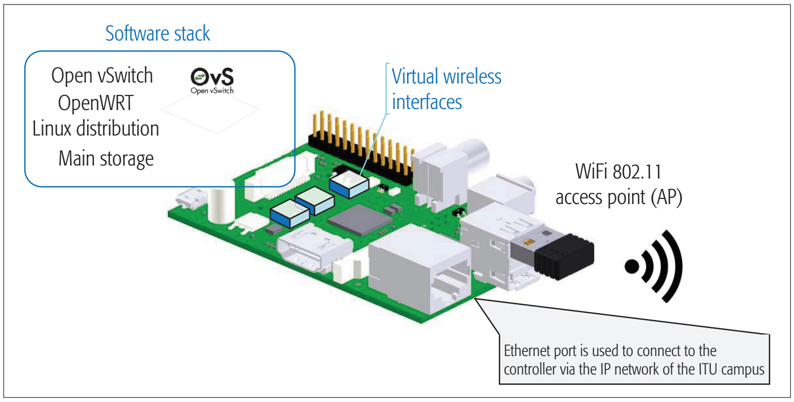

Figure 4. Raspberry Pi — OpenFlow RSU.

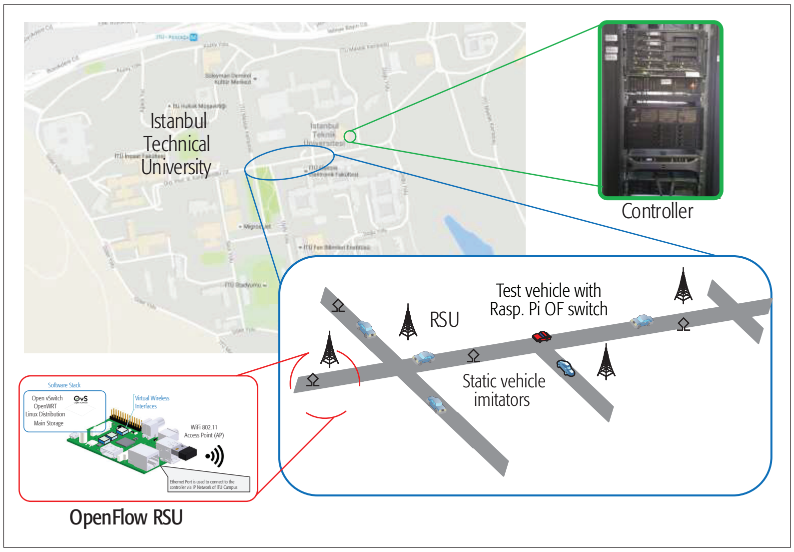

Figure 5. SD-VANET ITU campus testbed.

controller, while the red circle pinpoints locations of OpenFlow switches. The management ports of the OpenFlow switches were connected to the wired LAN of the university campus. The controller was also connected to the wired LAN of the university campus via Gigabit Ethernet connection. Furthermore, the control plane of the SD-VANET communicated through TCP connection, as seen in the control path of the OpenFlow switch in Fig. 2, over the wired LAN of the university campus. The control plane communication was isolated using the private virtual LAN (VLAN) allocated for the SD-VANET control plane communication within the ITU network.

One dedicated laptop computer with wireless connection was placed within the coverage of each OpenFlow switch. In addition, students of the Computer Engineering Department were notified about the existence of the OpenFlow wireless access points. Students were requested to join the real-time test environment and connect to the Internet using OpenFlow switches over their smartphones.

Wireless hosts connected to the OpenFlow switches were orchestrated by the controller. In addition, the number of users allowed within an OpenFlow switch was also organized by the implemented wireless host management module on the controller. A test that contains several users was run

One of the major future challenges of the proposed architecture is providing scalable software-defined controller for large VANETs which contain highly dynamic topologies. Network virtualization is a promising technique to handle the scalability problem, but it is critical to decide how to create or partition the network into overlay networks or slices based on various applications and service demands.

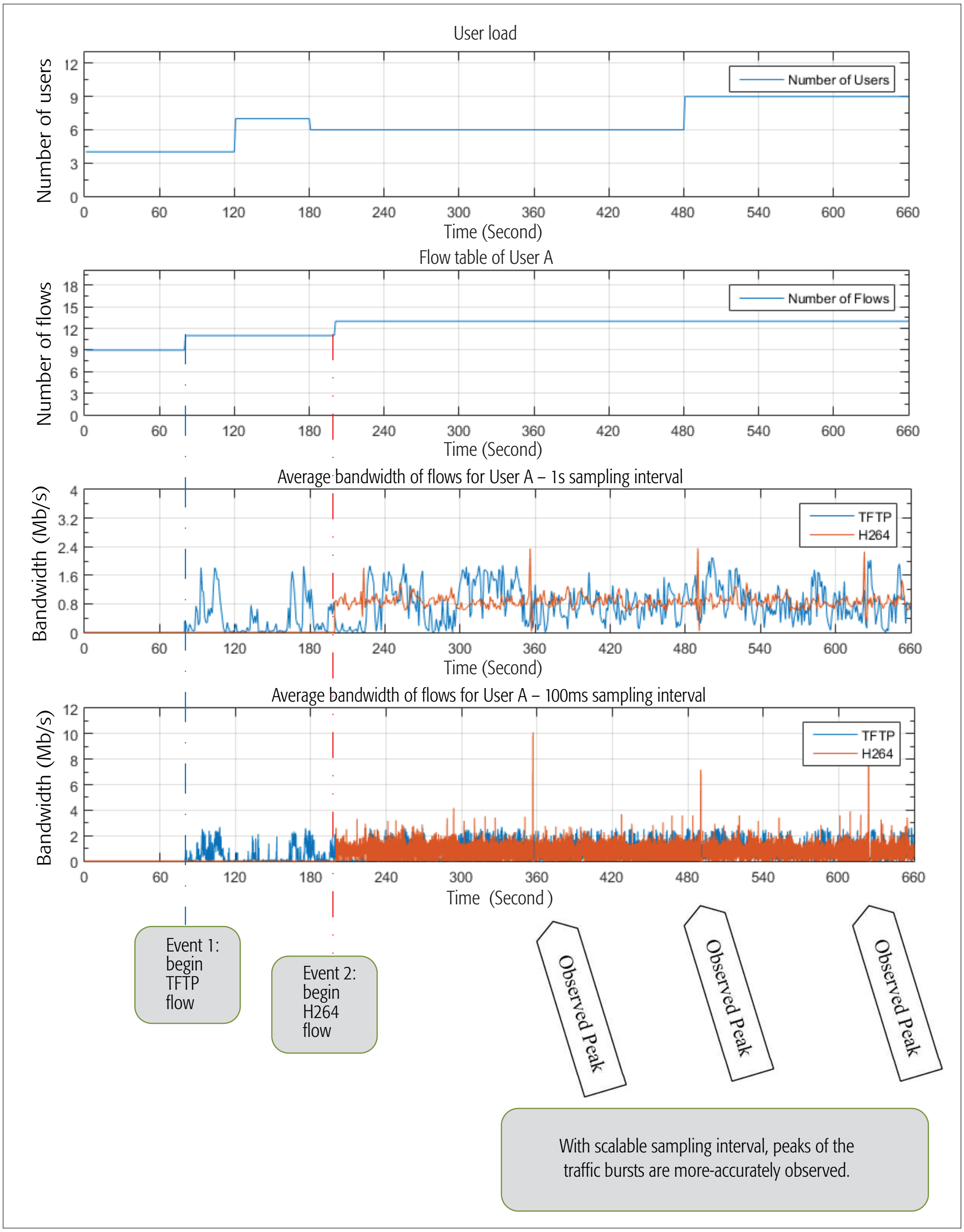

Figure 6. SDN-WAM monitoring results.

to validate the implemented architecture. The two distinct events started by **User A** were observed during the test with two different sampling interval such as 1 s and 100 ms. Additionally, the number of users within the observation period was also tracked by the controller and partially plotted in Fig. 6. Moreover, the basic flow admission control algorithm expressed in Fig. 3 optimized the available bandwidth of the user. The number of flows per user is also depicted in Fig. 6. The bandwidth profile of the H.264 stream and TFTP flows are also depicted in Fig. 6. By changing the sampling period of the bandwidth profiling, the controller also managed to capture the traffic bursts generated by the flows in small periods such as the one generated by H.264 in Fig. 6.

Conclusion

In this article, the SD-VANET architecture proposed for enhancing legacy Wi-Fi network to VANET has been explained in detail. Furthermore, a novel SD-VANET testbed has been explained and implemented. The SDN-WAM protocol, proposed for user management and flow admission control, was implemented on a testbed. Moreover, monitoring and management capabilities of the proposed architecture was shown using the deployed SDN-WAM implementation. Using the aforementioned capabilities, the proposed architecture is nominated as a promising augmentation candidate for fifth generation (5G) network design.

One of the major future challenges of the proposed architecture is providing a scalable software-defined controller for large VANETs that contain highly dynamic topologies. To this end, network virtualization is a promising technique to handle the scalability problem, but it is critical to decide how to create or partition the network into overlay networks or slices based on various applications and service demands. Furthermore, the future architecture should provide an interface to support critical use cases of 5G networks such as smart transportation, vehicles, and infrastructure. With this integration, orchestration of the whole network could be realized more efficiently, and even the network slices could be defined intelligently to address the major scalability problem.

In the near future, we will focus on defining the structure of the SD-VANETs and cellular core network interaction. In addition, we will also expand the offered architecture to address scalability issues that have not been mentioned within this work.

Acknowledgment

This work was supported by the Newton Research Collaboration Program under Grant ID NRCP1516/1/131.

References

[1] A. Al-Fuqaha *et al.*, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Commun. Surveys & Tutorials*, vol. 17, no. 4, 2015, pp. 2347–76.

[2] C. Bernardos *et al.*, "An Architecture for Software Defined Wireless Networking," *IEEE Wireless Commun.*, vol. 21, no. 3, June 2014, pp. 52–61.

[3] T. Chen *et al.*, "Software Defined Mobile Networks: Concept, Survey, and Research Directions," *IEEE Commun. Mag.*, vol. 53, no. 11, Nov. 2015, pp. 126–33.

[4] K. Zheng *et al.*, "Software-Defined Heterogeneous Vehicular Network: Architecture and Challenges," *IEEE Network*, vol. 30, no. 4, July 2016, pp. 72–80.

[5] Q. Zheng *et al.*, "Delay-Optimal Virtualized Radio Resource Scheduling in Software-Defined Vehicular Networks via Stochastic Learning," *IEEE Trans. Vehic. Tech.*, vol. 65, no. 10, Oct. 2016, pp. 7857–67.

[6] Z. Arslan *et al.*, "Sdoff: A Software-Defined Offloading Controller for Heterogeneous Networks," *2014 IEEE Wireless Commun. and Networking Conf.* Apr. 2014, pp. 2827–32.

[7] Z. He, J. Cao, and X. Liu, "SDVN: Enabling Rapid Network Innovation for Heterogeneous Vehicular Communication," *IEEE Network*, vol. 30, no. 4, July 2016, pp. 10–15.

[8] Q. Zhou *et al.*, "Network Virtualization and Resource Description in Software-Defined Wireless Networks," *IEEE Commun. Mag.*, vol. 53, no. 11, Nov. 2015, pp. 110–17.

[9] R. Sherwood *et al.*, "Flowvisor: A Network Virtualization Layer," ONF tech. rep., 2009.

[10] L. Liao, A. Shami, and V. Leung, "Distributed Flowvisor: A Distributed Flowvisor Platform for Quality of Service Aware Cloud Network Virtualisation," *IET Networks*, vol. 4, no. 5, 2015, pp. 270–77.

[11] P. Lin *et al.*, "A West-East Bridge Based SDN Inter-Domain Testbed," *IEEE Commun. Mag.*, vol. 53, no. 2, Feb. 2015, pp. 190–97.

[12] G. Cardone *et al.*, "The Participact Mobile Crowd Sensing Living Lab: The Testbed for Smart Cities," *IEEE Commun. Mag.*, vol. 52, no. 10, Oct. 2014, pp. 78–85.

[13] K. Chowdhury and T. Melodia, "Platforms and Testbeds for Experimental Evaluation of Cognitive Ad Hoc Networks," *IEEE Commun. Mag.*, vol. 48, no. 9, Sept. 2010, pp. 96–104.

[14] OpenDaylight — An Open Source Community and Meritocracy for Software Defined Networking, https://www.opendaylight.org/, accessed Dec. 15, 2015.

[15] Developer's Guide — Basic Approach to OpenWrt, https://wiki.openwrt.org/doc/guide-developer, accessed Dec. 15, 2015.

Biographies

Gökhan Seçinti [S'13] (secinti@itu.edu.tr) is a Ph.D. candidate in the Computer Engineering Programme, Istanbul Technical University (ITU). He serves as a reviewer for *IEEE Communications Magazine*, *IEEE Transactions on Vehicular Technology*, *IEEE Transactions on Wireless Communications*, the *International Journal of Communication Systems*, and the *International Journal of Computer and Telecommunications Networking*. He is a recipient of the IEEE INFOCOM Best Poster Paper Award (2015) and IEEE CAMAD Best Paper Award (2016). His current research includes software-defined networking and performance analysis of 5G networks.

Berk Canberk [S'03, M'11, SM'16] is an associate professor in the Department of Computer Engineering at ITU. Since 2016, he is also a visiting associate professor with the Department of Electrical and Computer Engineering at Northeastern University. He serves as an Editor for *IEEE Transactions in Vehicular Technology*, *Elsevier Computer Networks*, *Elsevier Computer Communications*, and the *Wiley International Journal of Communication Systems*. He is a recipient of the IEEE CAMAD Best Paper Award (2016), the Royal Academy of Engineering (UK) NEWTON Research Collaboration Award (2015), and the IEEE INFOCOM Best Poster Paper Award (2015). His current research areas include software-defined networking, 5G network performance analysis and modeling, and cognitive radio networks.

Trung Q. Duong [S'05, M'12, SM'13] (trung.q.duong@qub.ac.uk) is currently an assistant professor (lecturer) at Queen's University Belfast, United Kingdom. He has been awarded the Best Paper Award at IEEE VTC 2013, IEEE ICC 2014,and IEEE GLOBECOM 2016. His current research interests include 5G networks. He has published more than 250 technical papers (including 138 journals). He is serving as an Editor for *IEEE Transactions on Wireless Communications* and *IEEE Transactions on Communications*.

Lei Shu [M'07, SM'15] (lei.shu@ieee.org) is a professor at Guangdong University of Petrochemical Technology. He is also the executive director of the Guangdong Provincial Key Laboratory of Petrochemical Equipment Fault Diagnosis, China. His main research field is wireless sensor networks. He has published over 300 papers. He received GLOBECOM 2010 and ICC 2013 Best Paper awards. He serves as Editor-in-Chief of *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*.

In near future, we will focus on defining the structure of the SD-VANETs and Cellular Core Network interaction. In addition, we will also expand the offered architecture to address scalability issues that have not been mentioned within this work.