Global Journal of Engineering and Technology Advances
eISSN: 2582-5003
Cross Ref DOI: 10.30574/gjeta
Journal homepage: https://gjeta.com/

(RESEARCH ARTICLE)

Check for updates

OLSR Protocol based on Fog Computing and SDN in VANet

Intisar Mohsin Saadoon *

Department of Computer Science, Mustansiriyah University-Iraq, Baghdad.

Global Journal of Engineering and Technology Advances, 2022, 10(02), 060–070

Publication history: Received on 08 January 2022; revised on 16 February 2022; accepted on 18 February 2022

Article DOI: https://doi.org/10.30574/gjeta.2022.10.2.0037

Abstract

Wireless technology has been the subject of a lot of study in recent years. VANET is the fastest-growing area in wireless communications. The Vehicle Ad-Hoc Network (VANet) is a subtype of the Mobile Ad-Hoc Network (MANet) that is used to enhance road safety and passenger experience, giving a unique perspective on intelligent transportation systems. This wireless invention is supposed to improve road safety and efficiency as part of the Intelligent Transportation System. Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) techniques that enable IEEE 802.11p wireless access technologies deliver applications (secure/unsecure) while exchanging information to avert an accident and provide travelers with trustworthy information make up the VANET. As an end result of investments in independent cars, new network technology including SDN, side computing, and VANET studies are being introduced, causing VANET simulators to review their assist for those new capabilities. In this paper, we offer an SDN-compliant solution for managing wireless fog networks by combining open Flow and IP transfer protocols in a way that provides a flexible and configurable wireless data plane for fog networks in addition intelligent traffic engineering for network offloading. The proposal includes an efficiency rating for connections with decreased latency, flexible load balancing to select the quickest way, and lower network cost.

Keywords: Vehicle Ad-Hoc Network; OLSR; SDN; Fog; Virtual machine

1. Introduction

The wireless communication in the proposed system is categorized into two kinds of communication: control plane and data plane. A data plane link connects the control plane to the flow policy rules during data transmission. As indicated in Figure (1), controller activities are divided into three types:

- Centralized Control Mode
- Distributed Control Mode
- Hybrid Control Mode.

The SDN controller does not have complete control over the fog system, which is worked in hybrid control mode by the SDN controller sharing work with the BS and RSUC, as indicated in Figure (2). An abstract policy labeled Policy Rules will be communicated by the SDN Controller, in which the RSUCs or BSs will use their local expertise to decide specific behavior. The SDN controller sends data from RSUCs and BSs to the data center for long-term global purposes [1, 2, 3].

A link layer mechanism in every automobile can be used to frequently broadcast warning messages to gather information about neighbors in order to maintain an up-to-date vehicle network topology. When available, this information is delivered to RSU or BS, along with vehicle traffic data including a road map, position, speed, and sensor data [4, 5].

*Corresponding author: Intisar Mohsin Saadoon
Department of Computer Science, Mustansiriyah University-Iraq, Baghdad.

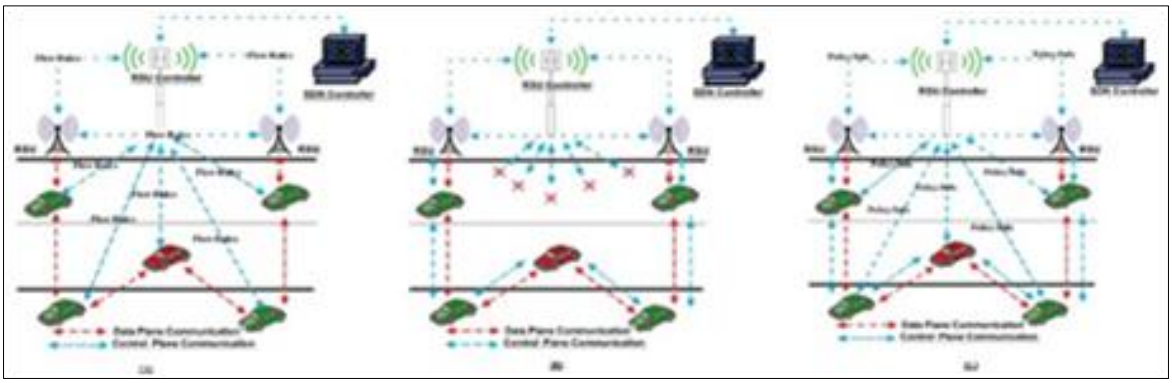


Figure 1 Controller operation modes:
(A) Centre control mode, (B) Distributed control mode and (C) hybrid control mode

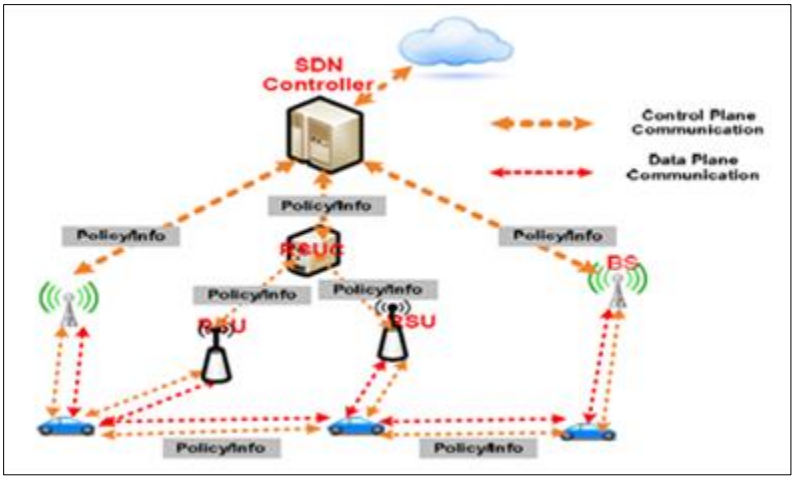


Figure 2 Hybrid Control Modes in SDN VANet

The SDN controller gets transmission data from the RSUs in addition to vehicle information from the BS. As a result, it may build a graph depicting the entire connectivity of SDN wireless nodes, as well as other information about various services. As a fog coordinator and resource manager, the SDN console is used [6].

The RSUC and BS additionally receive vehicle and/or traffic data from a collection of RSUs, allowing them to conduct supervisory control services without complete knowledge, similar to an SDN controller. RSUC and BS provide low-latency and geolocation intelligence for local and dispersed intelligence. The fog synchronization layer in the SDN controller coordinates RSUCs and BSs [7]. Fog computing is a concept in which central data centers and scattered high-end devices exchange heterogeneous resources and services. In our example, the RSUCs and BSs, as Fog devices, not only connect to the cloud via the SDN controller, but also exchange resources in order to operate the cars [8].

The SDN Console, RSUCs, and BSs are equipped with SDN functionality as well as virtualization to provide cloud (Fog) services to enable the SDN-based Fog Computing Framework for VANET. To allow virtual machine (VM) abstraction on those physical computers, a hypervisor must be installed, which is a low-level middleware for operating and maintaining one or more virtual machines on a computer [9].

These virtual machines host the services, allowing for migration and replication. In data centers, this technique is commonly utilized to increase portability, resource usage, and fault tolerance [10]. As indicated in Figure, several software components must be incorporated in the SDN controller (3).

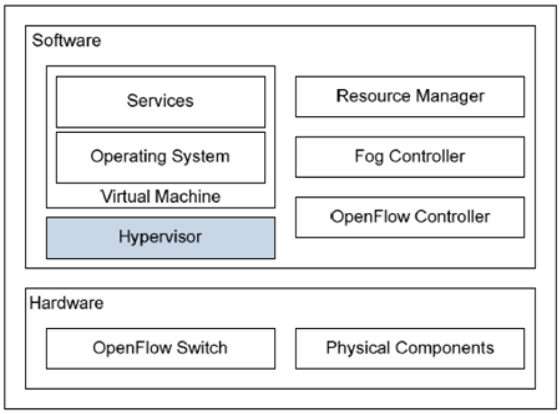


Figure 3 SDN Controller hardware and software components

Based on the suggested paradigm, a service-orientated useful resource sharing structure and mathematical version for SDN, BS, and RSUC controllers has been developed. The paradigm is built on the fundamental notion of service-oriented utility functions that enhance SDN, BS, and RSUC controller utilities for services rather than a task-oriented approach. Assume that SDN console, BS, and RSUC are host services that include the application. They just ask for a single service resource per service [11, 12].

The service consists of various activities, some of which may be completed using the service's resources, while others, known as externalization tasks, must rely on resources from other nodes. RSUCs and BSs seek to finish the service's responsibilities as fast as feasible. The RSUC must employ both its own resources and the resources of other RSUCs to complete the duties, as indicated in Figure (4).

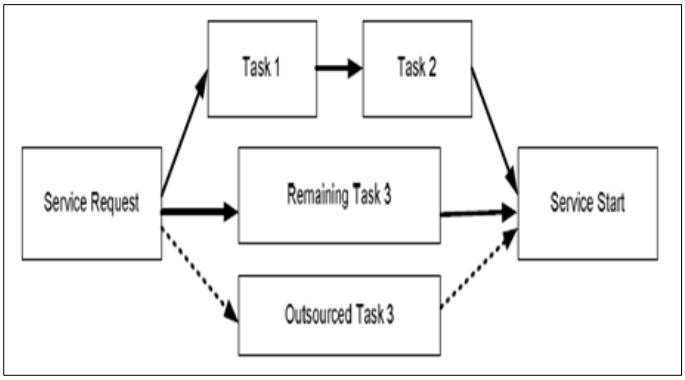


Figure 4 A flow task of a service containing outsourced task

The SDN controller acts as a resource coordinator, coordinating tasks and resources between BSs and RSUCs on a service-by-service basis to maximize service utilization. The advantage of utilizing SDN for Fog is that the SDN controller is aware of all the data about BS and RSUC, including resources and tasks, and can thus assign tasks to others to optimize resource sharing, such as latency, power and performance. Convex optimization techniques are used to reduce consumption [13, 14].

In SDNs, BSs, and RSUCs, services deployed in console virtual machines require a coordination method to disseminate information about data transfer policy changes and service hosting. In the process of creating, replicating, and moving services, you'll also require a coordination mechanism [15].

A routing system-based service may require a different number of BSs or RSUCs to host the service, necessitating the physical migration of VMs to RSUCs or BSs. The fog console, which is incorporated into the SDN panel, may be used to change the service's hosting rules and automatically retransmit data.

The cost of reconfiguring a hosting service, Instantiation, migration, replication, and statistics flow guidelines is high, ensuing in better latency and a worse great of experience (QoE) [16, 17].

2. Related Work

SDN has attracted a lot of attention from the academic community in the last several years as a way to enhance support for cloud and edge networks. [18] Proposes the use of SDN to enhance data transfers among mobile and the cloud by scaling and sending data across cloud computing support infrastructure on a temporary basis. Liang et al., likewise, [19] The Open Pipe framework is introduced, which aids in the virtualization of radio access through the use of fog computing. Sun et al. are a group of people who work together to solve problems. [20] The edgeIoT framework is introduced, which makes use of a central SDN controller to help with packet forwarding across fog nodes. Others like him. [21] For vehicle-to-vehicle (V2V) performance, vehicle-to-vehicle (V2I) infrastructure, and vehicle-based station communication, SDNs are utilized. SDN and fog computing, according to Lazar et al [22] and Truong [23], operate together to handle a wide range of vehicle networks, as well as the Internet of Vehicles (IoV). The developers of [24] also tweaked the SDN switch code to include a prototype fog compute and an embedded controller that uses MQTT middleware to connect scattered fog nodes. Similarly, Bruschi et al. proposed Open Volcano, an SDN-compatible virtual platform that uses network programming skills to function in foggy conditions near end users, in [25]. Infrastructure automation and calculation offloading, scalability and data transfer, in addition to QoS/QoE management and energy efficiency are all possible with Open Volcano. Betzler et al [26] offer a novel SDN-based path redirection strategy that achieves a weak balance and mitigates external interference by combining wireless back-computing and cloud capabilities. Huang et al. I looked at delivering QoS to wireless sensor fog devices. The fog layer is content aware due to the fact it could acquire data from packet headers in addition to content sensor data, allowing the central SDN controller to impose precise QoS regulations. The SDN controller is used in all of these ways to send signaling messages and also data packets via the wireless network. Unlike prior initiatives, our method employs an SDN controller to govern IP-based traffic and convey data in a hop-off way.

3. The Control Layer in Hybrid SDN Control Plane

Figure (5) shows a hybrid SDN solution that aims to make SDN more flexible, dependable, and fault-tolerant without adding complexity to controllers. A single network is divided into two logical slices using the hybrid SDN approach: a control slice having one or more distributed SDN controllers, and a data slice comprising clients and other routers. Mechanisms that involve an electoral method, in instance, permit for the choice of the nearest observer as master. Furthermore, the size of the control plane may be increased or decreased, and the coordination method between the controllers can be changed with this hybrid technique.

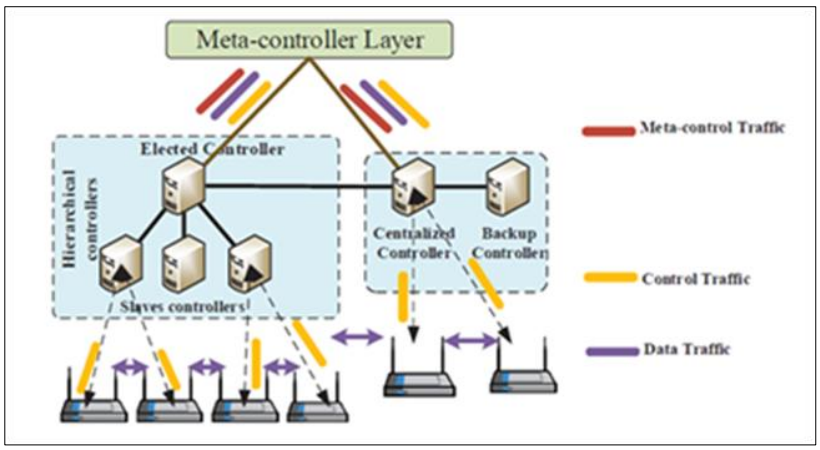


Figure 5 Hybrid control plane

SDN will be required to deliver services to establish virtual SDNs in a large-scale wireless fog infrastructure (vSDNs). In such scenarios, a single vSDN will be governed by many separate controllers, which may need a distinct control level structure. Customers A and B, for example, each lease a vSDN from SDN service provider C. Both vSDNs are hosted by C on the same physical network devices. Customer A prefers hierarchical control using POX controllers, whereas Customer B prefers centralized control with a RYU controller.

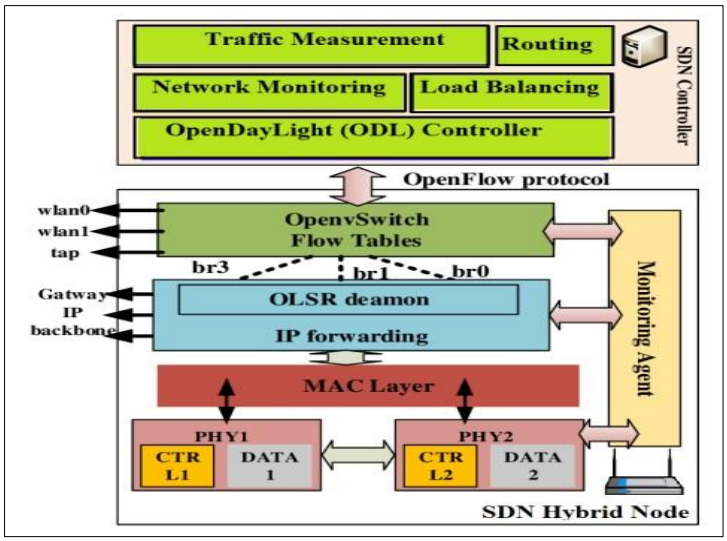
These requirements are simple to implement:

- The status of the vSDN before it is launched, that is, statically. On the data slice, Network Monitor constructs two vSDNs initially. The metadata control layer will then be informed of the keys and hosts that were used to construct these two vSDNs. For these two vSDNs, it will also offer the control plane topology needs. The meta-control layer can figure out how many controllers are needed to fulfil each vSDN's criteria. The keys used by each of the vSDNs are utilized to do this computation. It can reduce the number of control plane hosts necessary based on switch sharing between two networks, such as if two vSDN networks share a switch. Port one connects switch S1 to the first vSDN, whereas port two connects the identical switch to the second vSDN, and soon.
- The status of the vSDN after it has been dynamically formed, that is, after it has been launched. The meta control layer, on the other hand, should, if necessary, employ the current control level or scale. If the present control level fulfills the requirements of the new vSDN control level, this choice will be made. The meta-control layer gives the SDN control level additional flexibility in this approach.

4. Multi-Points Relays (MPRs)

Figure 6 illustrates the routing strategy, which is separated into two logical sub layers: standard SDN data transmission with Open Flow enabled in the top layer, and OLSR routing protocol in the lower layer. The first is in charge of linking Open Flow policies to the top-level console. The latter is in charge of IP routing among wireless interfaces. To transfer signal packets over multiple fog routers, we adopt an in-band technique. The goal to offer long-range wireless communication between wireless connections prompted this design decision. The controller can also use its own routing algorithms to find the optimum pathways for packets to take.

This composite construction has two advantages: i) IP forwarding and the use of OLSR allow all changes in the wireless routers topology graph to be reported, including the addition/removal of a router from fog and/or a wireless link; and ii) OLSR routing will improve fog network provisioning and reliability, keeping in mind that even if the SDN controller goes down or becomes unavailable, IP routing retains control of the network. Using Open Flow, packets can also be routed according to OLSR routing tables as directed by the SDN controller.



Where; Wlan-‒wireless interface; Tap-- network monitoring interface; PHY--physical interface; CTRL--control interface; DATA--data interface; Br--bridged network communications

Figure 6 the hybrid node Architecture

The console configures wireless fog routers by creating, deleting, and/or updating Open Flow rules, as well as retrieving current network status information from the closest wireless router. Because OLSR is a proactive routing system, each fog router uses Multipoint Relay to collect topology information from these other nodes (MPR). MPRs are utilized to minimize network traffic and offer the shortest pathways for all fog routers that have been identified as network destinations. Each router keeps track of its neighbors, which the MPR may identify using the MPR location list. After

exchanging HELLO messages with some other neighbors on a regular basis, each node can update its routing database and design a new shortest path for all endpoints.

5. Subsequent packet paths

As illustrated in Figure, the client submits a service request to the distant fog server after the initialization phase, however there is no end-to-end interconnection linking them (7). The PACKET IN message is used by the fog router to send a request to the console, as seen in Figure 8. The controller then replies to this request by determining the client-server data transfer path. The console evaluates the packets' headers to see if more access to the channel is desired, and if so, what actions should be taken with these packets. Each pipeline is made up of many Open Flow tables, each of which has multiple flow entries. The table below indicates where "go to" statements in the pipeline detect the search item, as well as the actions that are done at each level.

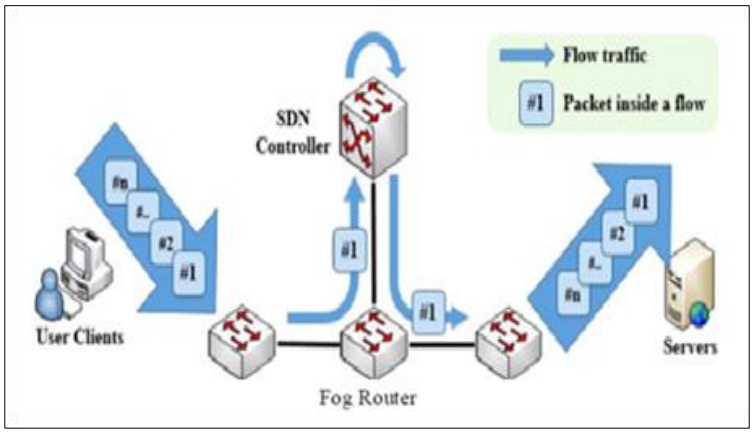


Figure 7 Flow traffic

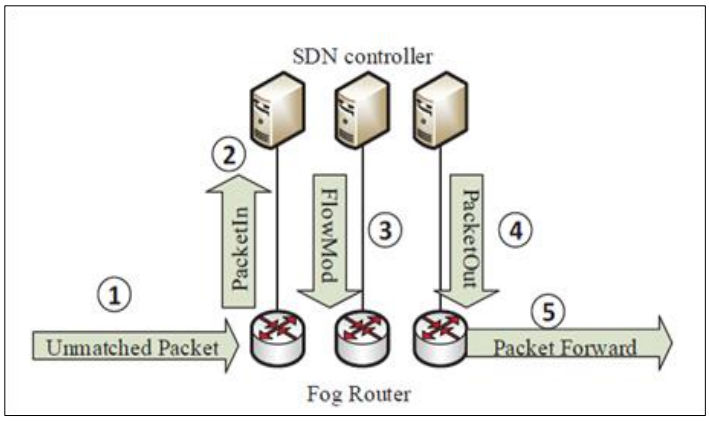


Figure 8 Installation Procedure of a new flow

In the meanwhile, the controller sends PACKET OUT command messages to the fog data structure plane (4 in Figure 8). PACKET OUT messages are handled differently than packets arriving on normal ports. These packages are loaded into the action set program, where the instructions are analyzed to see how the package and its related data must be handled. The server and client on the Mac Address port match all future packets. The server's records are match to the source IP destination address and port on the client side.

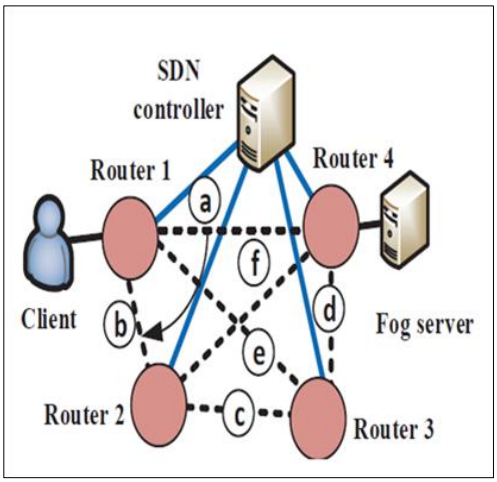


Figure 9 Load Balancing Scenario

6. Traffic Communication Paths

Consider the case of four wireless fog routers coupled in network architecture to demonstrate the pathways of traffic communications. Figure (9) shows the communication channels established by edges a, b, c, d, e and f to all of the fog routers.

In Figure (10), Method 1 depicts the load balancing algorithm for selecting the best path between the client and the fog server (9). Assume the first optimum path between Router one and Router four is on wireless connection a. The controller's default settings for specifying path a and regularly detecting link statuses have already been installed. When the bandwidth utilization approaches Threshold "Th" and the client suffers a traffic jam, the load balancing algorithm recognizes the network bottleneck and begins computing new Open Flow rules to reroute traffic over new connections, such as links b, f or e, d. The controller then floods all ports (using Open Flow Flow Mod packets) to fog routers on the server's path. The controller determines the new best path based on the graph's structure, which comprises all of the available R routers as well as the N connections that connect them. The software pipeline in each router is then updated with new Open Flow rules to plan flow inputs.

```
Algorithm 1: Load Balancing Algorithm
Data: Th, R; N
Result: Rerouting traffic to the optimal path
1 installDefaultFlowRules(R);
2 while Listening to LLDP packets do
3     if TrafficCongestion(Th) then
4         calculateNewOFRules(R; N);
5         FloodPackets(R);
6             calculateOptimalPath(source, destination, R; N);
7         if isBestPATH then
8             InstallnewOFRules(R);
9             else
10                goto:
11                calculateNewOFRules(R; N);
12            end
13      else
14            monitoring();
15        end
16 end
```

Figure 10 Load Balancing Algorithm

In addition, the MAC layer of wireless devices has been updated to allow them to communicate interference data to the controller. In Figure (11) ,Algorithm 2 shows how to compute the optimum path with the maximum SNR using the SNR-

based path optimization technique. We changed the "Maclow" and "MacRxMiddle" modules to allow them to pass SDN radios to each wireless fog router's "StaWifiMac" module instead of holding SNR values in the MAC layer. By altering the "WifiNetDevice" and "Tab Bridge" modules in the SDN emulator, this solution allows for the centralization of information about interferences on the SDN controller side.

```
Algorithm 2: SNR-based routing optimization algorithm
Data: SNR
Result: Best_SNR_Path(R; N);
1 if path (SNR) then
2        path ← Find_Best_SNR (rules)
3        return path
4 else
5       rules  ← calc_new_rules(SNR);
6       FlowMod_router();
7      best_path(rules);
8 end
```

Figure 11 SNR-based path optimization

The first table illustrates the flow inputs that the controller can arrange before and after the traffic jam. The controller has already set up the data channel between Router 1 with DpID1 and Router 4 with DpID4 when it first boots up. When Router 1 receives incoming packets on its default port, such as Input Port: Virtual Port 1, the packet headers are examined to see if they fit the Open Flow rules in the flow entries. The action sets are provided by Router 1's physical port, i.e. egress: to Router port 4, and then Router 1's packet destination, i.e. Router 4. As a result, packets from Router 1 must have their headers wrapped. Router 4's destination IP and MAC addresses are As a result, the controller creates stream entries to allow data to be sent to Router four utilizing both of its IP addresses, such as “Set_IP:router” IP 4, and its MAC address, such as “Set_MAC”: Router MAC 4, as destination addresse.

Table 1 Open flow entries the controller instants the fog routers

OpenFlow rules before controller instants fog routers		**OpenFlow rules After controller instants fog routers**	
Router1	Dp_ID1	Router1	Dp_ID1
Router2	-	Router2	Dp_ID2
Router3	-	Router3	-
Router4	Dp_ID4	Router4	Dp_ID4
Ingress Port	V_port1	Ingress Port	V_port1 & V_port2
OpenFlow entries before		**OpenFlow entries After**	
Set_IP	Router 4	Set_IP	Router 2 & Router 4
Set_MAC	Router 4	Set_MAC	Router 2 & Router 4
Output to port	Router4	Output to port	Router 2 & Router 4

When radio connection a breaks, the controller sets up new Open Flow rules to route traffic from Router one to Router four through Router two with dpID2. Because the new forwarding route must bypass Router two, the controller must program new flow entries into Routers two and four, as indicated in Table's after column (1). Controllers can instantly identify bottlenecks and reroute data to the excellent new path available since they have a consolidated view of the network. To do this, the console generates new rules, such as the MAC and IP addresses of new fog routers in the new path, and sends Open Flow Flow Mod messages to specify the new end-to-end path.

7. Results and discussion

Open Flow traffic is around 2 KB/sec when it first starts. This is due to the fact that discovery messages are included in the transferred data. The controller and the fog router exchange fresh Open Flow messages every 32 seconds. The new Open Flow rules that the controller must install in order to schedule flow entries in the fog prompt are included in these messages. As a result, Open Flow traffic spikes during this time. Because there is no communication with the controller at this period, OLSR traffic stays steady. Routing tables and neighbor tables are exchanged by all fog routers. The console continues to listen for LLDP packets received by fog routers after scheduling flow entries that expired out after 46 seconds. This implies that the hybrid routing strategy we employ to transfer data with OLSR uses less bandwidth than standard Open Flow-only routing, which is utilized to share data between the SDN controller and suitable fog router. As a result, as demonstrated in Figure 12, Which depicts the traffic rates generated through OLSR and Open Flow, our hybrid routing approach does now no longer upload network strain to the fog routers.

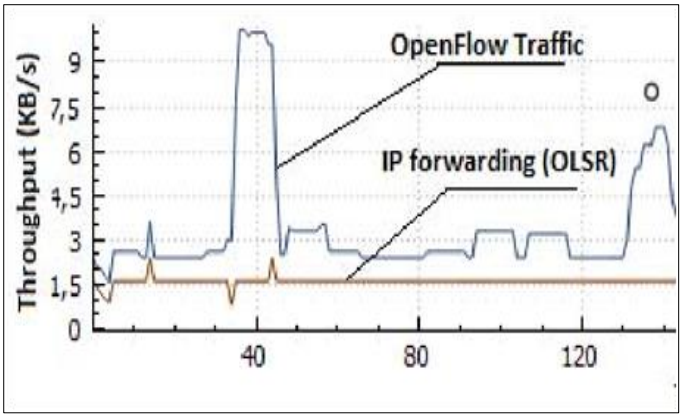


Figure 12 The traffic rates generated by OLSR and open flow

Router 4 begins discarding packets as a result of the buffer overflow, and throughput reduces from 850 to 200 kbps, resulting in substantial packet loss, as indicated by our tests. The control unit's load balancing mechanism, which is set to activate when the bandwidth hits 200 kbps, kicks in after 50 seconds to divert traffic from radio link A to radio link B, as shown in Figure (9). The console's topology detection module detects the wireless radio separation between Routers one and two, examines the new available route depend on its graph, and chooses Router two as the new shortest path to the target. The new route is derived from the OLSR protocol's routing database, which is continually updatedthose that send traffic across Link A, push down, and install new forwarding rules, for example. Router IP and MAC addresses are now restricted. After the new data channel is established, traffic is balanced across new wireless links. It takes the controller close to 10 milliseconds to detect the new available path and transfer the data. The time it takes to remove old rules from routers and put new rules into each router's flow tables is known as redirection delay. Our findings indicate that when many wireless hops are available in a fog network, our traffic engineering strategy is successful in diverting packets to the new chosen path. Figure 13 depicts all Router 4 observations.

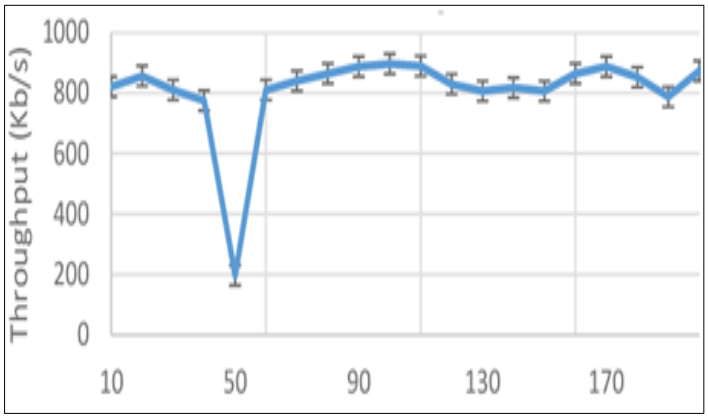


Figure 13 TCP Throughput

8. Conclusion

We provide an SDN-compatible approach for managing wireless fog networks by combining Open Flow and IP forwarding protocols in this paper. For fog networks, our method offers a flexible and adaptable wireless data plan as well as intelligent traffic engineering for network offloading. The results show that the suggested method is effective at creating connections with decreased latency Adaptive balancing to discover the perfect shortest path, as well as save network costs.

Compliance with ethical standards

Acknowledgments

The authors gratefully thank the Mustansiriyah University-Department of Computer Science-Iraq-Baghdad, for their cooperation with this paper.

References

[1] Flavio Bonomi, Rodolfo Milito," Fog computing and its role in the internet of things". Aug 2012. DOI:10.1145/2342509.2342513. https://www.researchgate.net/publication/235409978

[2] L Huang, G Li, J Wu, L Li, J Li, R Morello. Software-defined qos provisioning for fog computing advanced wireless sensor networks," in 2016 IEEE SENSORS. 2016; 1–3.

[3] Tarandeep Kaur Bhatia, Ramkumar Ketti Ramachandran, Robin Doss and Lei Pan. A Comprehensive Review on the Vehicular Ad-hoc Networks. International Conference on Reliability, Infocom Technologies and Optimization. 2020; 515-520.

[4] Ivan Stojmenovic, Sheng Wen, "The Fog Computing Paradigm: Scenarios and Security Issues". Federated Conference on Computer Science and Information Systems (FedCSIS).2014.ACSIS, Vol. 2. DOI: 10.15439/2014F503. https://www.academia.edu/22705803.

[5] Taoufik Yeferny and Sofian Hamad, Vehicular Ad-hoc Networks: Architecture, Applications and Challenges, International Journal of Computer Science and Network Security. 2020; 20(2).

[6] Sakshi Sharma, Nidhi. Vehicular Ad-Hoc Network: An Overview International Conference on Computing, Communication, and Intelligent Systems. 2019; 131-134.

[7] H Moustafa, Y Zhang. Vehicular networks: techniques, standards, and applications. AUERBACH/CRC Press. September 2019; 23-35.

[8] RadhaKrishna Karne , Dr. T.K.Sreeja. " REVIEW ON VANET ARCHITECTURE AND APPLICATIONS". Turkish Journal of Computer and Mathematics Education. Vol.12 No.04 (2021), 1745 – 1749.

[9] Sommer C, Eckhoff D, Brummer A, Buse DS, Hagenauer F, Joerer S, Segata M. Veins: The open source vehicular network simulation framework. In: Recent Advances in Network Simulation. Springer. 2019; 215–252.

[10] Anja Strunk, Waltenegus Dargie. "Does Live Migration of Virtual Machines Cost Energy?". March 2013. DOI:10.1109/AINA.2013.137. https://www.researchgate.net/publication/261089760.

[11] DCAITI. "Eclipse MOSAIC - Smart Mobility Simulation". 2006.https://www.dcaiti.tu-berlin.de/research/simulation.

[12] Ullah K. "On the use of opportunistic vehicular communication for roadside services advertisement and discovery" . August 2016..DOI:10.11606/T.55.2016.TDE-27102016-142325. Corpus ID: 168293717. https://www.semanticscholar.org/

[13] Margherita Grossi, Stefano Bortoli, el. "Trustworthy AI for Intelligent Traffic Systems (ITS)". Jiru J. Fraunhofer-Instituts für Kognitive Systeme IKS. 2021. https://www.iks.fraunhofer.de/content/dam/iks/documents/whitepaper-intelligent-traffic-systems.pdf.

[14] M Bari, A Roy, S Chowdhury, Q Zhang, M Zhani, R Ahmed, R Boutaba. "Dynamic Controller Provisioning in Software Defined Networks", October 2013. DOI:10.1109/CNSM.2013.6727805. https://www.researchgate.net/publication/257132495

[15] Amoozadeh M, Ching B, Chuah C-N, Ghosal D, Zhang HM. VENTOS: Vehicular network open simulator with hardware-in-the-loop support. Procedia Comput Sci. 2019; 151: 61–68.

[16] Sommer C. Veins User Manual documentation. 2017. https://veins.car2x.org/documentation/modules.

[17] Sliman A, Madi K, Khadour A, Maala B, Ahmad AS. Fabrication attack effect on medical applications based on Q4 VANETs. Int J Comput Sci Trends Technol (IJCST). 2017; 5(2): 1–4.

[18] I Ku, Y Lu, M Gerla. Software-defined mobile cloud: Architecture, services and use cases. in International Wireless Communications and Mobile Computing Conference, IWCMC 2014, Nicosia, Cyprus, August 2014. DOI:10.1109/IWCMC.2014.6906323. https://www.researchgate.net/publication/261026279.

[19] K Liang, L Zhao, X Chu, HH Chen. An integrated architecture for software defined and virtualized radio access networks with fog computing. IEEE Network. 2017; 31(1): 80–87.

[20] X Sun, N Ansari. Edgeiot: Mobile edge computing for the internet of things. IEEE Communications Magazine. 2016; 54(12): 22–29.

[21] X. He, Z Ren, C Shi, J Fang. A novel load balancing strategy of software-defined cloud/fog networking in the internet of vehicles," China Communications. 2016; 13(2): 140–149.

[22] SA Lazar, CE Stefan. "Future vehicular networks: What control technologies?" in 2016 International Conference on Communications (COMM). 2016; 337–340.

[23] NB Truong, GM Lee, Y Ghamri-Doudane. Software defined networking-based vehicular adhoc network with fog computing. In 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). 2015; 1202–1207.

[24] Y Xu, V Mahendran, S Radhakrishnan. Towards sdn-based fog computing: Mqtt broker virtualization for effective and reliable delivery in 2016 8th International Conference on Communication Systems and Networks (COMSNETS). 2016; 1–6.

[25] R Bruschi, P Lago, G Lamanna, C Lombardo, S Mangialardi. Openvolcano: An open-source software platform for fog computing in 2016 28th International Teletraffic Congress (ITC 28). 2016; 2: 22–27.

[26] A Betzler, F Quer, D Camps-Mur, I Demirkol, E Garcia-Villegas. On the benefits of wireless sdn in networks of constrained edge devices in 2016 European Conference on Networks and Communications (EuCNC). 2016; 37–41.