IJCSN - International Journal of Computer Science and Network, Volume 3, Issue 6, December 2014
ISSN (Online): 2277-5420 **www.IJCSN.org**
Impact Factor: 0.274



# An Intelligent Framework for Vehicular Ad-hoc Networks using SDN Architecture

**Balamurugan .V**

School of Computing Science and Engineering, VIT University
Chennai Campus, 600127, Tamilnadu, India.

**Abstract -** Vehicular Ad Hoc Network (VANET) is one of the rapid growing areas in the domain of networking technologies that offer various facilities like road safety, providing information about the traffic conditions, weather forecast conditions and events like accident happened on roadside. At present, VANET architecture has not been built with vendor independence and flexibility to be programmable directly without affecting the forwarding operations of data plane. In this paper, the importance of Software Defined Networking (SDN) in VANETs and its deployment using OpenFlow protocol in VANET architecture are discussed. A novel framework is provided for SDN to be employed over VANETs for the future enhancement of vehicular ad-hoc networks.

**Keywords -** *VANETs, Software Defined Networking, OpenFlow, SDN.*

## 1. Introduction

Intelligent Transportation System (ITS) is one of the key implementation areas which is been targeted by every country. It is designed to enhance the safety of the passengers who travel with four wheelers, as well as the quality of the driving experience for them, while operating in a dynamic, constantly changing environment [1]. This environment is fabricated by the high mobility of vehicles, making it very difficult to establish communication between them. Every vehicle will be equipped with networking mechanisms. Every node in the network act as a router, receives and forwards the packet. However, this is not easily achieved and many research efforts have been focused in this area, with significant contributions between government and industrial giants, as well as the academic community. There has been no general consensus yet with respect to the main services to be provided and the architecture of the VANET platform to be installed on-board.

Researchers try to define architectures that are capable of supporting the existing radio technologies. Although there have been several proposals, there is still no clear definition of an architecture capable of dealing with incremental improvements or upgrades of the existing mechanisms. Even though VANETs have a huge protocol stack, there are still many challenges in deploying it, such as unbalanced flow traffic among multi-path topology, and inefficient network utilization, unable to program the network plane directly. To address these challenges, Software-Defined Networking (SDN) is applied. Nowadays, SDN has emerged as a flexible way to control the network in a systematic way with OpenFlow as the most commonly used SDN protocol [2].

While most SDN/OpenFlow based innovations has been focused on wired networks, especially for datacenters, increased attraction is made incorporating SDN/OpenFlow into the wireless domain. The scope of SDN is primarily aimed on carrier backbones and access networks, the usage of SDN in other wireless scenarios has been gaining attention from both academia and industry. Other works on wireless SDN include OpenFlow in wireless mesh environments, OpenFlow in Smartphone as an application, SDN in heterogeneous networked environments. However, it is necessary to understand and extend the usage of SDN/OpenFlow in VANETs. In specific, the paper looks at the benefits of Software-Defined VANET services and new functionalities to support them. By decoupling the control and data planes in VANETs, network intelligence and state can be logically centralized and the underlying network infrastructure is abstracted from the applications. Thus, it will be possible to have highly adaptive, flexible, programmable, and scalable VANETs environments.

## 2. Background and Terminologies

In this section, the terminologies and the background information for VANETs are described, which are used in the remainder of this paper.

### 2.1 Vehicular Ad-hoc Networks (VANETs)

VANETs comprise of mobile nodes that are nothing but the vehicles that move in traffic. Every participating vehicle in the network acts as a node and performs the job of a router in a typical networking scenario [3].
VANETs give rise to many services like traffic information, roadside scenario, weather forecasts,

emergency help, etc., i.e., Intelligent Transporting System, and that is why VANETs gain more attention towards it. Automotive companies like General Motors, Toyota, Nissan, BMW and Ford promote this term. In order to implement ITS (Intelligent Transport System) the VANET architecture is reviewed and many proposals are given to yield more attractive services from VANETs. Once the ITS is implemented in full swing, vehicle users may gain enormous number of benefits out of it, which could possibly save as many lives as possible.

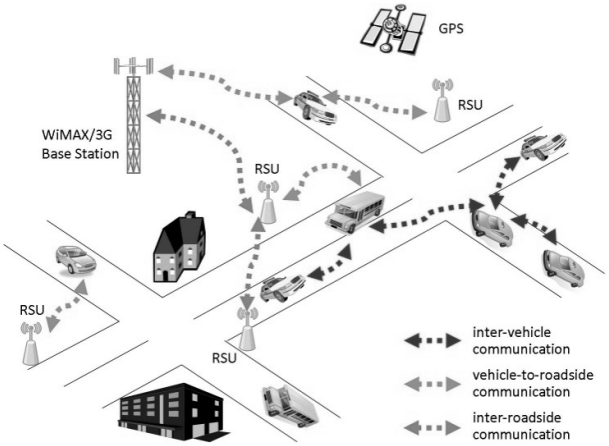


Fig. 1. Components of VANET and its communications

## 2.2 SDN / OpenFlow

The ultimate focus of the implementation of SDN is to separate the control plane and the data plane in the network. The latter is used for the data forwarding while the former is the driver to have complete control on network traffic. OpenFlow is the most commonly used protocol for communication between the SDN control plane and data plane [4].OpenFlow is a communications protocol that gives access to the forwarding plane of a network switch or router over the network. It is managed by ONF – Open Networking Foundation which is a user-led organization to promote SDN.Fig.2 depicts the operation of SDN.

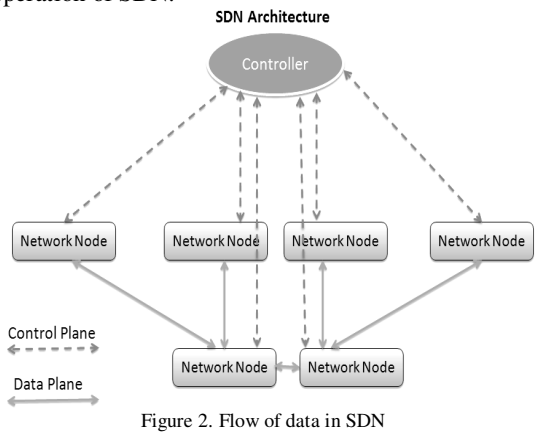


Figure 2. Flow of data in SDN

The operating system should be compliant with OpenFlow protocol in order to implement it in SDN. However SDN concept can be implemented using OpenFlow protocol which has been developed by Open Network Foundation. In every node the control plane and data plane are decoupled as shown in the figure. SDN controller has the control over all the nodes in its networking range. The programming activities and operations done at the control plane should not be transparent to the data plane which is the central idea of SDN. OpenFlow [5] allows direct access and manipulation of forwarding plane of networking devices such as switches and routers. Fig.3 depicts where the Application Program Interface and the SDN controller reside in the architecture of the SDN.

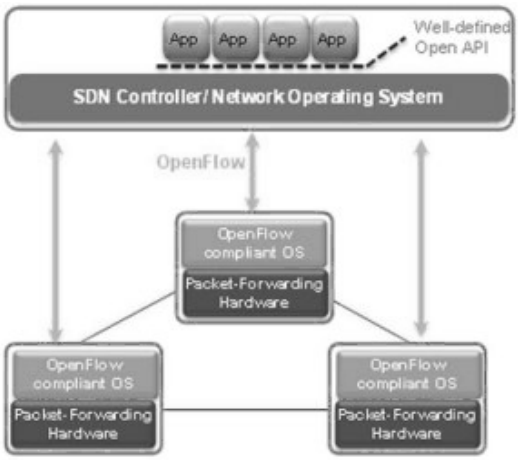


Fig. 3. Openflow

The operating system should be compliant with the Open Flow protocol specifications so that it could be implemented over it. The packet forwarding hardware is sandwiched underneath the operating system.

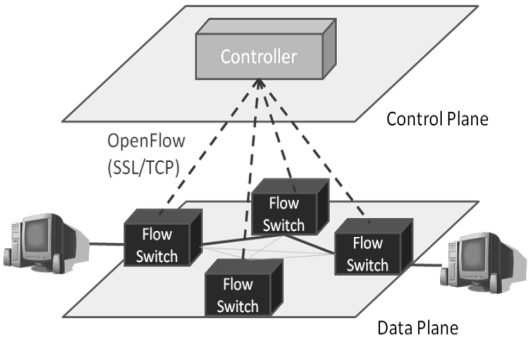


Fig. 4. Components of OpenFlow protocol

As shown in the figure 4, Openflow protocol consists of two main components: OpenFlow controller and several OpenFlow compliant switches that communicate using a secure SSL channel. Many nodes are connected physically via hardware. Every Openflow switch has a

flow table to maintain node information and secure channel for communication.

The SDN controller is literally a software program capsule that is used to modify the content of flow tables, which are located in each OpenFlow-enabled switch [6]. Each flow table contains a list of flow entries, which consist of Match Fields, Priority, Counters, Instructions, Timeouts, Cookie, and actions associated with it. When a packet comes to the switch it is looked up in the matching field of each entry. If there is a match, the packet will be processed according to the actions of the matching flow entry. On the other hand, if a packet does not match, a table-miss occurs. In this case, different actions can be taken as specified in the table-miss flow entry. The OpenFlow compliant switch can encapsulate the packet and send it to the SDN controller through the secure channel or directly drop the packet.

OpenFlow controller controls the behavior of the network by sending *flowmods* packets to modify the content of flow tables. The controller has two ways to add rules in the switch: proactively, where the controller takes responsibility and adds semantics before the packet getting arrived into the network; or reactively, where the controller reacts because of congestion in the network. For example, an Access Point (AP) or a switch that sends a data packet to the controller (by encapsulating the packet in an OpenFlow control packet called *packet-in*) because it does not know how to deal with it. Then, the controller sends the *flowmods* packet to the APs/switches with instructions.

## 3. Software Defined VANET

In this section, the paper explains the architecture of a Software-Defined VANET and its functionalities. VANETs take advantage of SDN concepts and functionalities to improve resource utilization, select best routes, and facilitate network programmability.

### 3.1 Components:-

- **SDN controller**: The central control of the SDN based VANET system. The SDN controller controls the network behavior of the entire system.
- **SDN wireless node**: The data plane components that are controllable by the SDN controller. They are the typically the vehicles that receive control message from the SDN controller to perform some actions as prompted.
- **SDN Road side Unit**: Stationary data plane elements that are controllable by the SDN controller. They are installed on roadside.

One distinct characteristic of Ad hoc networks is that the nodes act both as Hosts (sending/receiving traffic) and Routers (forwarding traffic on behalf of other nodes). An SDN wireless node is representing both an SDN data plane forwarding element and an end-point for data. Traffic from any wireless node will run through its own SDN module before being sent, which allows the SDN controller to determine the access of user traffic into the network.

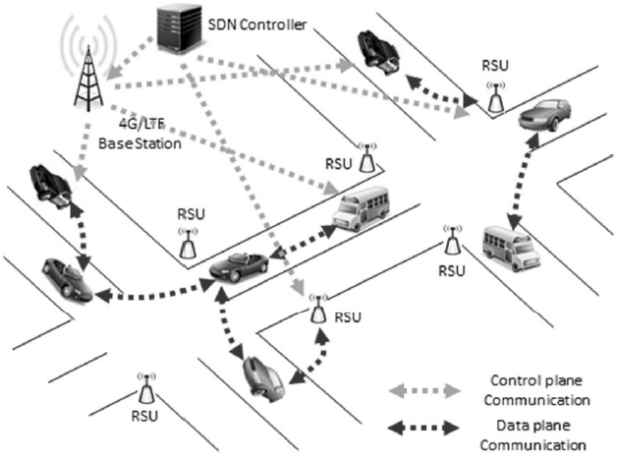


Fig. 5. Data communications in Software Defined VANET

### 3.2 Operations of Software Defined VANET

The primary goal of the architecture is to de-couple the data plane and forward plane. But there exist differences in how the architecture exactly deployed based on the degree of control of SDN controller.

- **Centralized Control Mode:** This is the mode where the SDN controller controls all the actions of underlying SDN wireless nodes and RSUs. In specific, all the actions that the SDN data element performs are explicitly defined by the SDN controller.

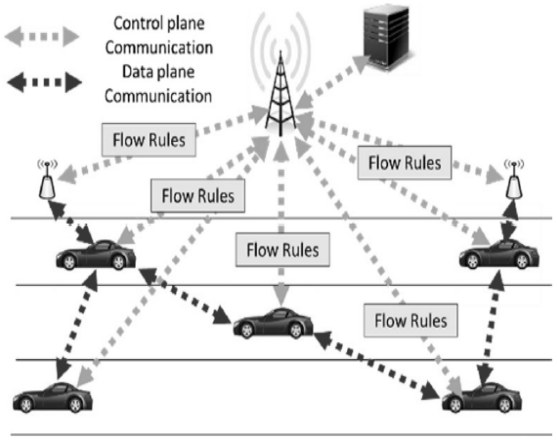


Fig. 6. Centralized Control Mode

- **Non-centralized Control Mode:** SDN wireless nodes and RSUs do not operate under any guidance from the SDN controller during data packet delivery. This control mode is very similar to the original self-organizing distributed network without

any SDN features, except that the local agent on each SDN wireless node controls the behavior of each individual node.

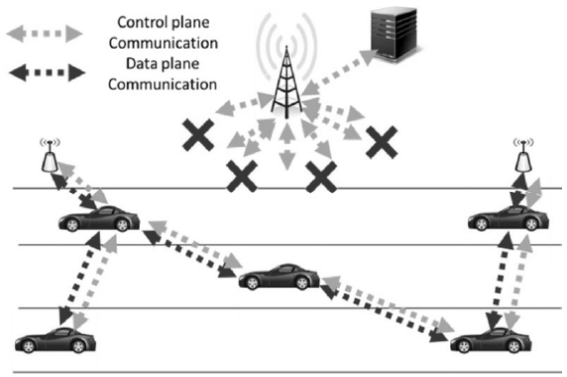


Fig. 7. Non-centralized Control Mode

## 4. Problem Definition

There are always possibilities of communication losses between mobile nodes with their respective controller. Beacon messages are utilized, a common feature in VANET systems which are used to test the link stability of the wireless nodes. The SDN controller needs to be vendor independent. The overall network needs infrastructure support for the proper functioning. Each SDN wireless node will exchange beacon messages to learn information about immediate neighbors. Message priority scheme is lacking in software defined VANETs [7] as it is very useful to have a provision to send prioritized messages first, i.e., they should be given the priority such as emergency, low and high.

## 5. Proposed Framework

### 5.1 Procedure in Steps

#### *5.1.1 Packet flow through a SDN switch:*

- Flow through all the tables 1 to n
- Execute actions as prompted.

#### *5.1.2 Steps in flow table*

- Find highest priority matching entry ‘e’
- Apply instructions : update metadata
- Go to next table if defined in action set.

### 5.2 System Architecture

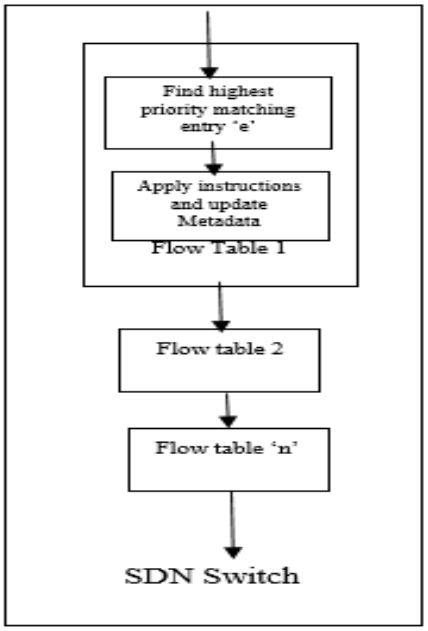


Fig. 8 System Architecture

## 6. Algorithm for Message Prioritization

```
/*****To send only emergency message*****/
If (emergency_event)
        {
        Freeze all message in queue except for the
        emergency message
        }
Else {
        If (packet queue > threshold value)
        //set a threshold value for queue size
                {
                        Discard CCH channel for
                        beacon safety messages
                }
        Else {
                If (event_message_detected>1)
                        {
                        Reject all messages in queue
                        except for the emergency
                        event message queue
                        }
                }
        }
/*******Handling the messages*******/
Vi: vehicle which sends the data to RSU
Di: data which is transmitted between vehicle and RSU
T: time period in which the Di will be fetched
Ds: data size
Dp: processed data
Mrsu: Memory of RSU
Step1:  for( each vehicle Vi that reaches RSU)
Step2:  Transmitted data will be received by
        transceivers of RSU and data is saved in Mrsu.
```

*Step 3:* If(RSU has 1 request)
then
process(requested Di);
send(requested Di)->Vi;
else
if(more than one vehicle to request)
then
fetch(Di)->$M_{rsu}$;
end if
*Step 4:* Arrange the Di according to Ds in an increasing order in $M_{rsu}$.
If(Ds are same for two or more request)
then
process(first_request);
send(Dp)-> Vi;
End If

## 7. Conclusion and Future Work

The future work has to be done in fallback mechanism which would be a key feature that must be provided to apply the SDN concept into mobile wireless scenarios since the nodes are always unstable. Transmission power control should be made as a provision in the services which are offered by Software-Defined VANET. In case of partial SDN controller connectivity loss, where only a subset of mobile nodes loses communication to the controller, remaining nodes should be able to continue functioning.

## References

[1] M.Gerla, "Vehicular Cloud Computing", VCA 2012 Proceedings, Cyprus, June 2012.
[2] Ian Ku, Francesco Ongaro, Rafael, Eduardo, "Towards Software-Defined VANET: Architecture and Services", 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET) 2014.
[3] http://en.wikipedia.org/wiki/Vehicular_ad_hoc_network
[4] N. Mckeown, "Software-defined networking." INFOCOM keynote talk, April 2009.
[5] ONF Solution Brief, "OpenFlow-Enabled Mobile and Wireless Networks", September 2013.
[6] Adrian Matei, Wolny, Slawomir, "Thire International conference on Communication Theory, Reliability and QoS", ISBN 978-0-7695-4070-2/10, 2010.
[7] Neha Verma and Rakesh Kumar, "A Method for Improving Data Delivery Efficiency in Vehicular Adhoc Networks" International Journal of Advanced Science and Technology Vol. 44, July, 2012.