Filling the Gap Between Software Defined Networking and Wireless Mesh Networks

Vagner Nascimento*, Marcelo Moraes†,Rosivaldo Gomes‡,Billy Pinheiro*, Antônio Abelém†‡,
Vinicius C. M. Borges§, Kleber V. Cardoso§ and Eduardo Cerqueira*
*Program Postgraduate in Electrical Engineering (PPGEE), Federal University of Pará (UFPA), Belém – PA – Brasil
†Program Postgraduate in Science Computer (PPGCC), Federal University of Pará (UFPA), Belém – PA – Brasil
‡Faculdade de Computação (FACOMP), Federal University of Pará (UFPA), Belém – PA – Brasil
{vagner,marcelo,billy,abelem,cerqueira}@ufpa.br
and rfgomesjr@gmail.com
§Institute of Informatics (INF), Federal University of Goiás (UFG), Goiânia – GO – Brasil
{vinicius,kleber}@inf.ufg.br

***Abstract*—Software Defined Networking (SDN) has emerged as a new paradigm that highly increase the network management flexibility through simple but powerful abstractions. The key idea is decoupling the control plane, which makes the forward decisions, from the data plane, which effectively makes the forward. However, the OpenFlow, the main SDN enabler, is designed mainly by wired networks characteristics. As consequence, Wireless Mesh Networks (WMNs) is not suitable for operating as control plane and many wireless networks features are neglected in the OpenFlow, e.g.: power control and network ID. In addition, there are few effort research to extend SDN to wireless networks and these existing works focus on very specific issues of this integration. In this paper, we propose an architecture to extent the OpenFlow functionalities in order to proper deal with wireless networks, including an approach for transporting the control plane over wireless multihop networks. The extensions include new rules, actions, and commands, which bring the network management flexibility to the wireless context. We validated our proposal by implementing and testing some extensions in a small real world testbed. As a proof of concept, we illustrate the OpenFlow capability of isolation between research and production traffics in a wireless backhaul.**

I. Introduction

The Internet became an important tool for significant fraction of the population. In this context, Wireless Mesh Networks (WMN) are wireless access networks that emerge as the first mile wireless technology widely used in the public and private environments, since it is a low-cost solution allowing easy deployment, various topologies and technologies.

On the other hand, there is a research effort to improve the infrastructure as well as necessity of new technologies applied to the core network. However, there is a limitation on the speed of these improvements that is caused by it's strong relation between hardware and software, where the first one leaves the factory with installed software and pre-defined resources.

Any extension of hardware functions can only be performed by the manufacturer or by acquiring use licenses. To fill this gap, the Software Defined Network (SDN) paradigm has emerged, enabling different protocols in different topologies can be tested without infrastructure replacement.

OpenFlow [1] is the main SDN technology. The resources of the network switches are allocated and shared as demanded for each class of users or experiment at the OpenFlow. Thus, enabling the network configuration through remote applications hosted on servers operating a network system. Nonetheless, SDN was initially designed for wired networks.

There are some related work which seek to adapt the characteristics of the WMN in the SDN paradigm. Yap et al [2] employs OpenFlow to provide mechanisms for authentication, authorization and monitoring in a wireless network for visitors. However, the use of OpenFlow is limited to the flow control interfaces between the wireless network and the wired network, neither interacting with multiple interfaces nor with intrinsic features of wireless networks.

Dely et al [3] shows an approach that introduces OpenFlow in WMN which focus on a handoff solution for mobility of users, but it is still limited to its specific scenario. NGUYEN et al [4] proposes approaches that deal with inherent problems of WMN, such as deployment of agents in the radios to reduce problems with the intermittent nature of the wireless links, to offer alternative routing, to use the backup controllers and application of pre-defined rules for situations where there is no connectivity with the controller. Nevertheless, all these approaches has a disadvantage, i.e. adaptations for an environment not always gather the requirements of another and thus, it is difficult to provide a generic approach.

In common wired networks, the connection between switch and controller or between switch and users is made by a physical link (cable), which does not suffer from external interference, however, in the wireless networks scenario the connections status and the quality of the links can often vary.

This work presents a proposal to extend the SDN based on OpenFlow, in order to adapt it to WMN and enables better integration of these environments with wired networks. Openflow extensions are proposed, involving the inclusion of new messages on the OpenFlow protocol, new actions to the flow table and inclusion of IEEE 802.11 MAC headers to the list of headers of the OpenFlow research. In addition, minimum specification of the hardware architecture is also included. As a result, WMN can take advantage of the benefits that SDN offers in order to gather the demands of new services and users.

II. Software Defined Multihop Wireless Networks

The overall objective of this paper is to propose a generic architecture, in order to extent the network management flexibility of SDN to the WMN in a more complete way. This architecture proposes adapting the OpenFlow protocol to allow control flows from wireless interfaces enabling a new form of packet forwarding in wireless networks.

The research on sdn are focused on wired networks and are at an advanced stage, including membership of several hardware manufacturers such as Extreme Network, HP, Juniper, IBM, Cisco, Brocade, and others [5]. In the context of wireless networks this scenario has not yet been reached, but the equipment is no longer "black boxes" which are only proprietary software. Currently, some wireless devices come with Open Source firmware that allow modification of all protocols of TCP/IP model and the development of new technologies, such as OpenWRT, DD-WRT, among others [6].

This new feature of wireless switch allowed us to elaborate a proposal for Software Defined Multihop Wireless Networks - SDMWN. In this network you can leverage the native flexibility of wireless networks to enjoy the benefits of SDN also in these environments, and you can schedule them to meet the demands of new services and users.

A. Definition of Components

The main components needed to implement the SDN paradigm over wireless multihop are similar to those used in wired networks, but it is necessary to redefine some of these components, with the main differences with its equivalents in the wired network and the necessary adjustments to ensure the feasibility of the proposal.

B. Wireless Switch

As well as switches and routers at wired networks, the wireless switch also is compatible with the OpenFlow protocol and will be responsible for connecting the wireless user for other equipment.

The wireless switch should enable the creation of virtual interfaces and ensuring different operating modes (AP, ad hoc and mesh.) on each virtual interface created. It is desirable that each interface can operate in independent channels, with independent powers.

The wireless switch used in this work meets only part of these requirements, it is not possible to define channels and different strengths for different interfaces at the same time. The hardware used is the router TL-WR1043ND with AR9132 400 MHz processor with 8 MB of flash memory and 32 MB of RAM. This radio has only one network interface and wireless to meet the requirements of the architecture SDMWN a D-Link DWA-140 Atheros AR9271 chipset interface, compliant with IEEE 802.11bgn standard was added.

The original software that wireless switch does not have native support for 802.11s protocol [7] and OpenFlow and was replaced by firmware OpenWRT Backfire 10.03.1 [8], which is a Linux distribution with kernel 2.6.32.27 customized to be embedded these devices. With OpenWRT was possible to enable 802.11s and OpenFlow.

C. Control Plan

There are two approaches to reach the datapath to each of the wireless switch. In In-band approach, the flow of control plane is routed using the same interface where the data flow is routed and the out of band approach using one interface dedicated to control traffic and other data traffic, guaranteed best performance in packet forwarding and physical isolation between these flows.

The separation between control flow and data flow on the wired network can be implemented by separating these flows in different Virtual Local Area Network - VLANs.

In this work we chose the out of band approach, however as you cannot extend the VLANs of the wired network over the wireless network, we chose to use a dedicated flow of control plane interface. The interface used was configured to operate in IEEE 802.11s standard that uses the Hybrid Wireless Mesh Protocol Protocol - HWMP for packet forwarding.

Thus it is not necessary to have any kind of physical link (wired or point-to-point) between wireless switch and wireless controller, allowing greater flexibility in the choice of physical topologies.

D. Backbone and Access Interfaces

In the wired network the access (that connect the user) and backbone interfaces (that connect two switches) are generally equal and only the ability of these interfaces are different, so receiving similar treatments for OpenFlow. Already in wireless networks, user interfaces, access and backbone interfaces have different operation modes and features are subject to various interferences that do not exist and need to be considered in wired networks.

- Access Interface: It is the interface that the user will connect to the network. This interface must operate in Access Point - AP mode and should have an associated SSID. Being an interface type AP, it can connect multiple users at the same time. The technique of encryption and security is not defined at the moment but should be considered in real environments.
- Backbone Interface: It is the interface that the data are sent from a wireless router to another and can connect a wireless router to several others at the same time. It must to operate in ad hoc mode. In addition to the 802.11s interface used to flow control, you must to configure other interface in ad hoc mode (for data traffic), forming a backbone data backbone isolated from control. To provide access to mobile users, at least one interface must be configured in AP mode. In this work we chose to use two access interfaces for users, one for production network and one for SDMWN network.

As describe before, will be needed four interfaces (one 802.11s, one ad hoc and two AP), but the hardware architecture presented only two physical interfaces, being necessary to divide them into virtual interface. Thus, we chose to divide the wlan0 interface in three virtual interfaces, as can be seen on the right side of Figure 1.

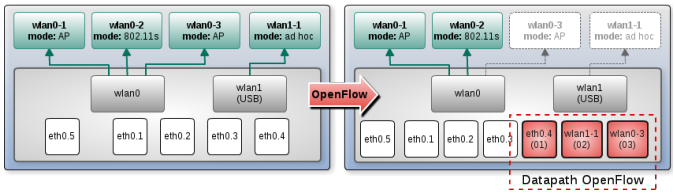

Fig. 1: OpenFlow hardware architecture.

TABLE I: Summary of Interfaces Used

Interfaces	Mode	Function	OpenFlow
wlan0-1	AP	Users access from production network	-
wlan0-2	*mesh*	Backbone of the production/control network	-
wlan0-3	AP	Users access from SDMWN network	Yes
wlan1-1	*ad hoc*	Backbone for Data network "SDMWN"	Yes
eth0.4	*ethernet*	Connect to wired network (when necessary)	Yes
eth0.(1,2,3,5)	*ethernet*	Not Used	-

In Figure 1 it can be seen on the right side a simplified architecture of the hardware used and on the left side a OpenFlow enabled architecture. In view of OpenFlow, wireless interface wlan1, wlan0-1 and 3 are handled as commons ethernet ports and referenced in the wireless router as OpenFlow ports 2 and 3, respectively. The other two wireless interfaces (wlan0 and wlan0-1-2) are used for traffic control and production and should not be handled by the OpenFlow flows.

E. Extensions to OpenFlow

In addition to the definitions presented, the inclusion of new messages on the OpenFlow protocol, extension headers and a new set of actions for flows tables should be inserted into OpenFlow framework to consider the particularities of wireless network environments.

The main change is related to the layers where the OpenFlow acts. Only the headers of the data link layer, network and transport are handled. However, should be considered that for wireless networks features geared to the physical environment equipment may be even more important, for example, signal strength, operating frequency of the channel noise, the distance between the devices and especially amount of forwarding addresses.

1) New Messaging Protocol: Adding new messages to the protocol ensures that the controller knows the existence of wireless interfaces, the number of interfaces, the mode and channel operating, wireless neighbors (next hops), noise levels, among other information associated only this type of interface. If this information is available on the controller, make it easier for developers of applications and protocols for wireless environments.

In [3] already see the need for these changes, since in the case study described, OpenFlow has been used as part of a mechanism to provide the handover in mesh network. The necessity of using agents installed on the routers in the form of Nagios plugins which are often asked about the status of radios and connections.

Figure 2 shows a example of dpctl command (program used to create, modify and delete datapaths) where it can be seen that in line 2, the in-port rule has value as parameter "1w". In this case, the character "w" indicates that this is a wireless

```
1: dpctl add-flow tcp:RADIO_01,
2: in_port=1w, dl_src=$MAC_SRC,
3: actions=change_power:4dbm,CONTROLLER
```

Fig. 2: Command dpctl indicating wireless interface.

interface and therefore may have different treatment of wired interfaces, with actions that can be performed only on these interfaces, changing the power radiated signal, for example.

2) New Actions in Flows Table: The forwarding packets over wireless multihop is only possible by manipulating the source address and destination (data link or network layer) packets. In line 3 of Figure 3 the action mod-dl-dst makes changing the original destination address of the packet to the address of the next hop switch (RADIO-02) then the packet is routed back to the network share IN-PORT. This procedure should be repeated at the next radio, again changing the destination address of the next hop address and so on until the packet reaches the final destination.

```
1: dpctl add-flow tcp:RADIO_01,
2: dl_src=$MAC_SRC, dl_dst=$MAC_DST,
3: actions=mod_dl_dst:MAC_RADIO_02,output:IN_PORT
```

Fig. 3: Dpctl command used to include register on radio-01.

This procedure is valid for testing environments as presented in this work, but is not scalable in an environment where the amount of different flows follow different paths is larger and the network has a varying amount of radio constantly.

One solution to this problem would be the inclusion of an action in the flows table to indicate the radio of the next hop, based on an identifier as can be seen in line 3 of Figure 4.

```
1: dpctl add-flow tcp:RADIO_01,
2: dl_src=$MAC_SRC, dl_dst=$MAC_DST,
3: actions=FORWARD:2
```

Fig. 4: New dpctl command showing action FORWARD.

In this example, a new FORWARD action is presented and determines that packets that satisfy the corresponding rule should be forwarded to the switch with identifier "2". Other actions should be defined for wireless switch, among which we can mention: change the transmission power, change the operating channel, delete virtual interfaces, add/delete SSIDs, change intervals between beacons, change the fragmentation threshold values and among others.

3) New Header Fields: OpenFlow acts only on the address fields of the Ethernet frame (source and destination) and the IEEE 802.11 header, presented in Figure 5, has four address fields that are used depending on the direction of the flow of data and devices involved.

In a wireless network a packet going to a destination may have to go through intermediaries (such as access points). These intermediates are the immediate destination of the package, but not their final destination. Thus, it is necessary

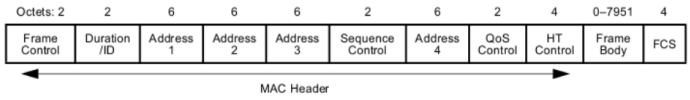

Fig. 5: Format Header IEEE 802.11.

to identify them, as well as identify the final destination for the package to arrive at the same [9].

III. Experimental Environment

A test environment was set up with four radios as shown in Figure 6 which is observed in the area defined as the "source" that there are two users: "Production" and "SDMWN-A" . In the area defined as "destination" there is a Internet access link, a "Controller" and a "SDMWN-B" computer. In the center, in the area called "Backbone", is wireless multihop formed by the four radios.

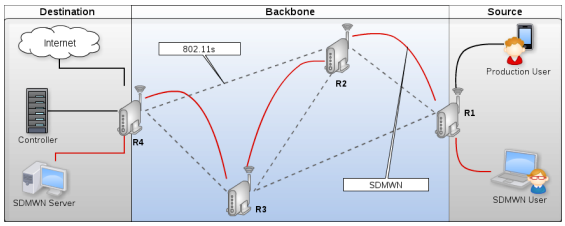

Fig. 6: Test environment.

The user "Production" will access the network through the AP (wlan0-1) production interface and make use of traditional network access web, email, etc. Their packages are sent from the AP interface for 802.11s interface (wlan0-2) through Layer 3 routing. The packet forwarding R1 to R4 is defined by HWMP. Finally, in R4, packets are routed to the Internet .

This process describes the existence of a wireless network production that uses an mesh backbone without any interference on the OpenFlow packet forwarding.

The user "SDMWN-A" access the network through the AP-SDMWN interface (wlan0-3). From the time that the packets arrive at the radio wlan0-3 OpenFlow manipulates the flow table adding the appropriate rules in each radio to forward packets from R1 to R2, R2 to R3 and R3 to R4, where using their respective ad hoc interfaces wlan1-1. Upon arriving at the R4 radio packet is forwarded to the wired interface eth0.4 "SDMWN-B".

The communication between the controller and the radios will be made by 802.11s backbone, the same way as with the production network.

Network performance was determined by evaluating the key performance metrics for Quality of Service (QoS) used for this type of analysis. To determine throughput, jitter and packet loss used the iperf software [10] in sessions lasting 100 seconds. The latency was also analyzed and was used in the ping command, sessions of 100 pings at intervals of 0.2 ms between them. The data collection sessions were repeated sixty (60) times at four (4) different daily times.

IV. Conclusion

In this work, we propose an extension to OpenFlow based Software Defined Networks, adding new messages to this protocol, new actions for its flow table, new headers based in the IEEE 802.11 MAC frame, and a minimum architectures hardware specification providing, thereby, a major compatibility between the environments that will be reached.

The experiments show that a SDMWN can be used in environments where an expansion of the wired network is necessary, through a wireless network infrastructure keeping the premisses and originals configurations applied to the wired SDN, without bigger changes in existing topology or performance losses.

As show before, the extension SDMWN fill gap between the SDN and the WMN, providing the necessary definitions and implementations to handle correctly IEEE 802.11 frame. Also, the results presented show the propose viability keeping the quality of a voip call over a WMN using SDN.

As future work, we intended validate this proposal in larger scale testbeds like FIBRE [11].

Acknowledgment

This work has been partially supported by the CAPES, CNPq, FAPEG, FAPESPA, FADESP and RNP.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746

[2] K.-K. Yap, Y. Yiakoumis, M. Kobayashi, S. Katti, G. Parulkar, and N. McKeown, "Separating Authentication, Access and Accounting: A Case Study with OpenWiFi," OpenFlow Technical Report 2011-1, 8 2011, made available from July 2011.

[3] P. Dely, A. Kassler, and N. Bayer, "Openflow for wireless mesh networks," in *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*, 31 2011-aug. 4 2011, pp. 1 –6.

[4] X. N. Nguyen, G. Neglia, and T. Turletti, "Software Defined Networking in Wireless Mesh Network," *MSc UBINET Thesis.*, 2012.

[5] ONF. (2013) ONF - Open Networking Foundation. https://www.opennetworking.org/.

[6] C. E. Palazzi, M. Brunati, and M. Roccetti, "An OpenWRT solution for future wireless homes," in *Proceedings of the 2010 IEEE International Conference on Multimedia and Expo, ICME 2010, 19-23 July 2010, Singapore*. IEEE, 2010, pp. 1701–1706.

[7] G. Hiertz, D. Denteneer, S. Max, R. Taori, J. Cardona, L. Berlemann, and B. Walke, "IEEE 802.11s: The WLAN Mesh Standard," *IEEE Wireless Communications*, pp. 104–111, Feb 2010. [Online]. Available: http://www.comnets.rwth-aachen.de

[8] OpenWRT. (2013) OpenWRT: Wireless Freedom. https://openwrt.org/.

[9] R. C. Carrano, M. Saade, M. Elias, M. Campista, I. M. Moraes, V. N. de Albuquerque, L. Claudio, M. G. Rubinstein, H. M. K. Costa, O. Carlos, and M. B. Duarte, "Multihop MAC: Desvendando o padrÃ£o 802.11s," in *26Âº Simposio Brasileiro de Redes de Computadores e Sistemas Distribuidos*, Rio de Janeiro-RJ, Brasil, 2008, pp. 13–59.

[10] Iperf. (2013) Iperf - The TCP/UDP Bandwidth Measurement Tool. www.iperf.fr. [Online]. Available: iperf.fr

[11] FIBRE. (2013) FIBRE - Future Internet Testbeds Experimentation Between Brazil and Europe. http://www.fibre-ict.eu/.