Enabling WiFi in Open Access Networks

Shamik Sarkar, Christopher Becker, Josh Kunz, Aarushi Sarbhai, Gurupragaash Annasamymani, Sneha Kumar Kasera, Jacobus Van der Merwe
University of Utah
Shamik.Sarkar@utah.edu,cbecker@cs.utah.edu,josh.kunz@utah.edu,{aarushis,guru,kasera,kobus}@cs.utah.edu

ABSTRACT

The idea of open access networks is gradually becoming a reality with a large number of municipalities and communities deploying their own open network. In the open access network model, the municipality/community can act as the network operator and a multitude of services can be provided to the end users over the deployed infrastructure. In this age of wireless networks and mobile users, WiFi must also be an integral part of the open access network. We discuss the need for designing WiFi from the point of view of open access networks. We identify the aspects of WiFi that need to be modified and the challenges that arise due to these modifications. We address these challenges by presenting a simple, yet novel design for enabling WiFi in open access networks using SDN and access point virtualization. Ours is the first attempt towards integrating open access networks and WiFi. We implement a preliminary prototype of our design in the Emulab test bed and successfully verify its operation.

1 INTRODUCTION

The open access network model deviates from the traditional vertical network model by isolating the service provider from the network operator [1]. The network operator provides the network infrastructure and the service providers (e.g. Internet service providers, smart home service providers, etc) compete in a fair way to get access of this infrastructure for selling their services to the end users. This fair competition among service providers eliminates monopoly by increasing the number of service options from which the users can choose. The idea of open access networks has recently started becoming a reality with an increasing number of municipalities and communities deploying their own network infrastructure [2], [3], [4], [5]. This is generally the case for remote areas where penetration of traditional Internet Service Providers (ISP) is absent or limited due to the cost of reachability. In such scenarios, the municipality/community can act as the network operator and a multitude of services can be provided to the end users over the deployed infrastructure. In open access networks, services can be enabled dynamically. Thus, many innovative services are expected to come into existence. The end users themselves may opt to become service provides for any innovative or cheaper services they may have to offer for short durations. An important challenge in realizing such a model is the overhead associated with enabling and maintaining the services. Efficient network control architectures like *OpenEgde*, powered by Software Defined Networking (SDN) [6], have been designed that automate the management of network resources and allow new services to be plugged in and out of the network dynamically [7].

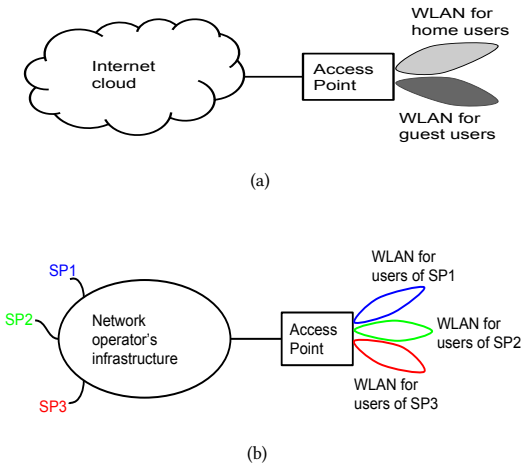

Figure 1: WiFi slicing in (a) traditional home networks and (b) open access networks.

Given the inevitability of open access networks, as observed globally [1], [2], [3], [4], [5], [8], and the indispensability of WiFi, in this paper, we make the first attempt to enable WiFi in open access networks using SDN and access point virtualization. The integration of WiFi with open access networks poses the following challenges. First, the conventional strategy of slicing WiFi (e.g., in home networks), based on the number of different user groups, is not suitable for the open access network model. Here slicing implies virtualizing WiFi by configuring multiple virtual access points (VAP) on the same physical access point, where each VAP has its own network over the wireless medium [9]. Unlike the traditional WiFi model where end users once connected to the WiFi can access all the services provided over the Internet, in the open access network model, a user's association with the WiFi is for the purpose of obtaining only those services that the user subscribes to. Figure 1(a) shows the traditional WiFi slicing with two slices where the number of slices is determined by the number of user groups with different hierarchy of network resource accessibility (e.g. one slice may be for the home users with unlimited Internet access and the other slice may be for the guest users with rate limited Internet

HotWireless'17, October 16, 2017, Snowbird, UT, USA

ACM ISBN 978-1-4503-5140-9/17/10…$15.00
https://doi.org/10.1145/3127882.3127889

access). The wireless LANs (WLAN) created by these two slices can have different authentication mechanisms, rates and quality of service configurations, but within a slice all users are served identically by the wireless network.

In open access networks traffic for different services may have different requirements and priorities (e.g., VoIP services may need low jitter, video traffic services may require higher bandwidth and public safety services must have maximum priority). So, users subscribed to different service providers should be served differentially by WiFi. However, with the WiFi slicing model of Figure 1(a), it is not possible to serve users belonging to different services differentially if they are connected to the same WiFi slice. We need the WiFi users that should be treated similarly to be in a single group and accordingly connected to a single WiFi slice. Since all subscribers of a service can be treated identically, we propose that the number of slices over WiFi, in the open access model, should be determined by the number of services or the number of service classes as in Figure 1(b). In this figure, SP represents a service provider and three WiFi slices have been created corresponding to the three active service providers. The difference in open access networks in terms of slicing the wireless network raises important questions including who will configure the VAPs, how many VAPs should be configured on the access point, how the services will be mapped to the VAPs (one to one or many to one), how will the users know which Service Set Identifier (SSID) to use for a service.

The second challenge is associated with frequent SSID transition from a user's perspective. In traditional WiFi, users manually, or automatically, connect to an SSID. Once connected, the user usually remains associated with that SSID for the rest of his/her session. However, in the open access network model, the user might be hopping on to different services in a short time span. Since different services/service classes are on different SSIDs, as shown in Figure 1(b), the user must manually disassociate from an SSID and associate to another SSID every time he/she hops on from one service to another. This procedure may be burdensome from the user's perspective. So the challenge here is to provide a mechanism for handling this frequent SSID transition in a automated way.

The third challenge is tied to the use of SDN in open access networks. New network control architectures for open access networks are expected to be built upon the flexibility provided by SDN protocols like Openflow [10]. In many scenarios, the Openflow controller configures the switches to forward data traffic based on input and output port numbers rather than MAC addresses. This is usually the case to make the system work for other layer 2 protocols like MPLS. In case of port based forwarding, the edge Openflow switch would simply identify an end user based on the port number to which the user is connected to. However, in case of WiFi, multiple wireless users can connect to the same wireless interface of the access point. If we simply connect the access point's uplink port to the edge Openflow switch, all users connected to the access point would be identified by the same port. The challenge in this case is to devise a way such that each user, connected to a single access point, can be identified uniquely while it uses his/her own choice of service, simultaneously.

In this work, we explore a simple, yet novel design that addresses the aforementioned challenges associated with integration of WiFi to a network control architecture that enables the open access paradigm. Specifically, our contributions can be summarized as follows:

- We identify the need for looking at WiFi from the perspective of an open access network model as compared to the tradition model.
- We present the design of access points, controlled by the network operator, for open access networks. These access points can be configured dynamically to assign different services or service classes to different SSIDs.
- We present the idea of using an open-vswitch (OVS) [11] bridge inside the access point in order to create unique mappings between end users and virtual ports. Using this approach, the Openflow edge switch can identify users by the virtual port numbers in case of port based forwarding. This will enable multiple users, connected to a single access point, to use their own choice of service, simultaneously.
- We implement a simplified version of our design on the Emulab test bed and successfully verify the operation of our proposed idea.

The idea of slicing ISP network and renting the slices to service providers and utility providers, where each provider can manage end-to-end or last mile network resources corresponding to its own slice, has been discussed in [12], [13]. The model considered in our work has some similarity with the models described in [12], [13] in the sense that, in open access networks the network operator may provision different services differently (based on service contract, priority, optimizing overall resource allocation, quality of service (QoS) requirements etc.). Though our work is built upon the existence of such a model, our design is independent of the provisioning policies adopted by the network operator. The primary contribution of our work is the design of WiFi for open access networks where different services, becoming online/offline dynamically (sometimes even for a short duration), are provisioned differently as well as dynamically. Our design of WiFi for open access networks ensures that the service specific provisioning used by the network operator over the wired links can be extended over the wireless links too.

2 OVERVIEW OF THE PROPOSED DESIGN

There are three players in the open access network model - the network operator, the service providers and the end users. The network operator manages the network infrastructure and the end users subscribe to the network operator for obtaining the services provided by the service providers over the network operator's infrastructure. In general, the relation between the end users and the network operator is not revenue driven. E.g., the network infrastructure laid down by the municipalities is primarily for enabling basic services like public safety or smart grid facilities for the end users and not for generating revenue from the users. On the other hand, the relation between the service providers and the network operator is generally revenue driven. The service providers pay the network operator for using the network infrastructure and generate revenue by getting end users subscribe to their services. The end users usually connect to the network operator's infrastructure using an optical network terminal (ONT) [8]. With WiFi support enabled, the wireless users would connect to the network operator's infrastructure using an access point.

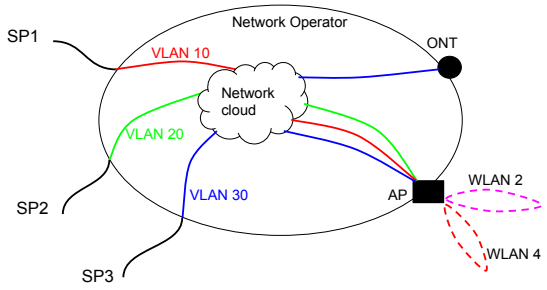

Figure 2: Mapping of service classes to VAPs in WiFi

To address the challenges associated with enabling WiFi in open access networks, we propose a design where the access points are managed by the network operator. This design will enable the network operator to dynamically control the wireless networks. So, the onus of configuring and maintaining service/service class specific VAPs is on the network operator. In the following discussion, we consider each service class to have its own VAP rather than each service having a dedicated VAP. Each service class consists of a number of services with similar traffic characteristics like priorities, QoS requirements etc. The network operator configures each access point with as many VAPs as the number of service classes. Assuming that each service will have a different virtual LAN (VLAN) for traffic isolation on the wired part of the infrastructure, there will be many to one mappings from the VLANs to the VAPs as shown in Figure 2. This figure shows an open access network with three service providers SP1, SP2 and SP3. Traffic for SP1, SP2 and SP3 are carried inside VLAN 10, VLAN 20 and VLAN 30, respectively, inside the network operator's infrastructure. SP2 and SP3 belong to the same class of service, so they map to a single VAP which creates WLAN 2. SP1 belongs to a different class of service, so it maps to another VAP which creates WLAN 4. In open access networks, the number of service providers is not expected to be static. New service providers may come online, while some older ones may go offline. As a result, the network operator must handle the mapping between VLANs and VAPs dynamically. Specifically, when a new service provider, belonging to a certain service class, comes online, the network operator needs to dynamically assign its traffic to the relevant VAP of its service class. For this purpose our design considers an OVS bridge, controlled by a central controller, inside the access point. The central controller dynamically maps the VLAN id of a newly created service to the VAP of its service class as shown in Figure 3. We assume that as soon as a new service provider comes online, it registers itself as a service belonging to a certain service class and the network operator assigns a new VLAN id for this service. Based on this information regarding the service class and the VLAN id of the service, the controller knows which VAP this newly created VLAN should be mapped to and accordingly it creates flows on the OVS bridges of the controlled access points.

The reason for mapping each service class to a VAP rather than mapping each service to a VAP is to limit the number of VAPs on a single physical access point. Since all the VAPs associated with an access point operate on the same physical WiFi channel, a significant amount of available air time might be occupied just for

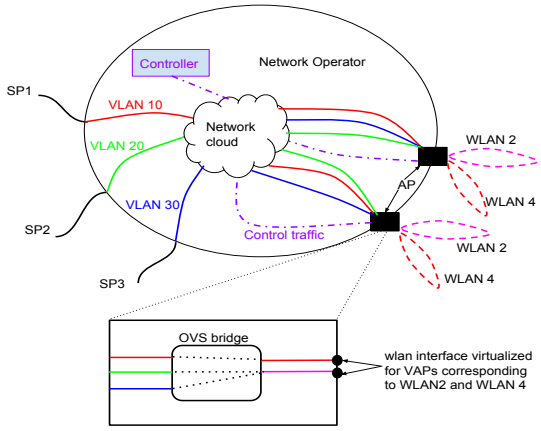

Figure 3: Dynamic configuration of OVS bridges inside access points by a central controller

broadcasting the beacon frames [9] associated with a large number of VAPs resulting in a lower throughput of the overall system.

3 AUTOMATIC TRANSITION OF SSIDS

We now look at our design from the end user's perspective and identify the associated challenges. From a user's perspective, the most convenient option would be to select a service from the set of available services displayed on the mobile device's screen. However, there are few challenges in effectuating such convenience in the model of Figure 3. First, it might be difficult for the user to know which SSID to use for a particular service. Simplistically, each service class' SSID could indicate the type of traffic it carries. However, we would have to rely on the end user's cognizance to be able to identify the right SSID. Second, it would be inconvenient for the end user to disassociate from the current VAP and associate with another VAP whenever the user decides to move on from one service class to another. Third, although we assume that the end user's relation with the network operator is not revenue driven, the user would still need to authenticate against the network operator to prove that the user is a legitimate beneficiary.

In order to address these challenges, we enhance our design to handle additional functions. For authenticating to the network operator, we create a default VAP with SSID 'Bootstrap' on every access point. This VAP does not carry traffic from any service provider. It is dedicated for authentication and communication with the network operator. When an end user turns on his/her wireless device, it manually connects to the 'Bootstrap' SSID and authenticates itself to the network operator. This authentication can be achieved using IEEE 802.11i while connecting to 'Bootstrap' SSID, or any higher layer authentication scheme after connecting to 'Bootstrap' SSID or a combination of these approaches. Once authenticated, the end user's device runs a background script. This script maintains a client server socket connection with the server process running on the authentication server of the network operator. The end user selects a service of his/her choice and the client process informs the server about the choice of this particular service. The server checks whether or not this user has already been authenticated by

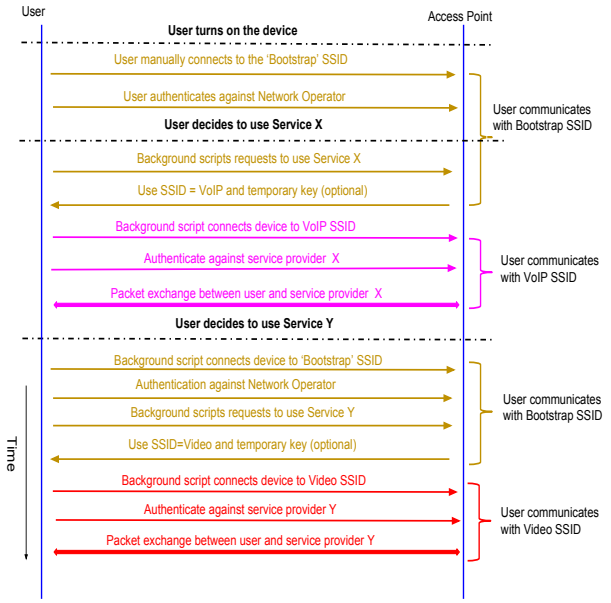

Figure 4: Automatic transition of SSIDs on user's device

the network operator. In case of affirmation, the server provides the client the SSID and an optional temporary key. The background script on the client uses this information to automatically associate the end user's device to the relevant SSID. This procedure of informing the end user of the SSID eliminates the user's burden of selecting the appropriate SSID and allows the SSIDs specific to the service classes to be hidden. Only the 'Bootstrap' SSID is visible. This makes selection of the default SSID ('Bootstrap' SSID) easier for a new user and significantly shortens the list of WiFi networks at the end user's device.

Once the user is moved from the 'Bootstrap' SSID to another SSID, a path is created between the end user and the relevant service provider. The end user then authenticates with the service provider over this newly created path. Once authenticated by the service provider, the user can enjoy the service as long as he/she wishes. When the end user desires to move on to another service, he/she simply selects that service on the display of the mobile device. This action triggers the background script which associates the wireless device to the 'Bootstrap' SSID and informs the server about the desired service. The server again checks if the user is authenticated or not and the whole process is repeated. The use of a one time password or a temporary key is optional. If a service does not use any encryption mechanism, this temporary key may be used by the network operator to ensure that a rogue end user does not hijack the authenticated session. Figure 4 shows the sequence of events when an end user turns on his/her wireless device and uses a Service X for VoIP followed by a Service Y for video streaming.

4 HANDLING PORT BASED FORWARDING FOR WIRELESS END USERS

The access points in our design are controlled by a central controller using a protocol like Openflow. At the same time, we assume

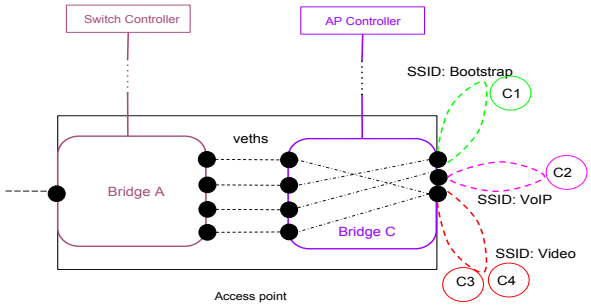

Figure 5: Combining the terminal switch and the access point to a single entity by having two OVS bridges.

that the wired switches in the network operator's infrastructure are controlled by a controller as well. Since the controller for the access points has functions that are different from the controller for switches in the infrastructure, we consider these two controllers to be separate. These may be simply two different processes on the same machine. In many cases, the Openflow controller, for managing the switches in the infrastructure network, uses port based forwarding to make the system work for any choice of layer 2 protocol. In case of port based forwarding, the end user would typically connect to a port on the terminal switch/ONT. The terminal switch would identify the end user based on the port number to which the end user is connected. Based on this identity, the central controller would create a path between this port and the service of choice for this specific end user. So, whenever there is uplink traffic from the end user, based on the port number of the incoming traffic, packets would be forwarded to the destined service provider. In the case of WiFi, the uplink port of the access point would connect to the terminal switch. Here, all users connected to the access point will have to use the same service at a particular time instant because the terminal switch will be able to associate the port number (that connects to access point's uplink port) to the identity of only one user. There is no way of knowing which service provider an uplink packet is for.

In order to resolve this issue, we combine the terminal switch and the access point into a single device as shown in Figure 5. There are two OVS bridges configured inside the access point. Bridge A acts as a terminal switch and bridge C acts as the OVS bridge in Figure 3. The only difference between bridge C and the OVS bridge in Figure 3 is that flows are configured on bridge C based on port numbers and in Figure 3 flows are configured on the OVS bridge based on the VLAN ids. For each user connected to the WiFi, the controller for bridge C creates a virtual link between bridge C and bridge A. Now, there are as many virtual ports on bridge A as the number of wireless end users connected to the access point. Flows created inside bridge C would ensure that each user has a unique virtual port on bridge A. Since the bridge A acts as a terminal switch and there is a virtual port for each end user, connected to the access point, uplink traffic for different services can be classified based on the virtual port number through which the packet comes into bridge A. In Figure 5 there are four end users, C1, C2, C3 and C4, so four virtual links are created between bridge A and bridge C.

5 PRELIMINARY IMPLEMENTATION

We implement a simplified version of our design using four Linux machines on the Emulab test bed [14]. We configure three of these machines as wireless clients and the fourth one as an access point. We virtualize the wireless interface on the access point using hostapd [15]. We create four VAPs that include one for the 'Bootstrap' SSID. While we configure a Linux machine as an access point, our design can be implemented on OpenWRT [16] based access points as well. We create an RYU [17] based Openflow controller for managing the access point by a central controller. We integrate our design with *OpenEdge* [7], a dynamic network control architecture capable of controlling the network operator's switches in an open access network. We virtualize the network operator's switches and service providers using Mininet [18]. OpenEdge uses port based traffic forwarding. This allows us to test the idea discussed in Section 4.

To verify our preliminary implementation, we select three different services on the three wireless nodes. Each of them is first connected to the 'Bootstrap' SSID and upon successful authentication against the network operator, gets automatically shifted to the SSID of their selected service by a background script. We use the *SecureOps* authentication mechanism, a component of OpenEdge [7], for authenticating end users to network operators and service providers. SecureOps uses a virtualized SIM abstraction for authentication. In our implementation, the duration for moving from one SSID to another (which consists of disassociation and association with SSIDs, client server communication for obtaining the required SSID, DHCP request and response for getting IP address in the new SSID) by the background script is $\approx$ 2 - 3 seconds. Whenever an end user moves from one service to another, it must encounter this delay. We also observe that flows get created dynamically on bridge C (Figure 5) for mapping each end user to a unique virtual port on the bridge A. This ensures that the end users connected to the same access point can use their own choice of service, simultaneously. Based on the above observations, we consider the outcome of our initial implementation to be successful and we believe that our design can be integrated in existing open access networks.

6 CONCLUSION AND FUTURE WORK

We identified the need to revisit WiFi from the viewpoint of open access networks. We presented the design for enabling WiFi in open access networks using SDN and access point virtualization. Ours is the first attempt towards integrating open access networks and WiFi. We have built a preliminary prototype of our design and successfully verified its operation in an open access network. Our initial exploration regarding enabling WiFi in open access networks can proceed along the following directions:

Since services can be dynamically enabled in open access networks, a multitude of innovative services shall come into existence. The end users may opt to become service providers (for some innovative service they have to offer) sometimes, even for short durations. As a result, the number of services may grow exponentially. In such scenarios, the service specific SSID approach shall not be sustainable. Isolation of traffic within a single SSID could be a possible direction of investigation for this highly scaled scenario.

As end users tend to become part time service providers, we can expect that service providers will also use WiFi for connecting to the network operator [1]. It might be possible for a service provider and a service consumer to use same access point for connecting to the network operator. While the end users connect to the WiFi, either as a service consumer or as a service provider, WiFi must treat them differently.

In our proposed design, a wireless end user cannot use services belonging to different service classes at the same time. This limitation arises since different service classes are on different VAPs and the end user can be connected to only one SSID at any point in time. As soon as the user selects a new service, he/she would get disconnected from the first service. Since the transition time to move from one SSID to another is not negligible, as observed during our implementation, the user may have a disruptive experience. The possibility of traffic isolation within a single SSID would be able to address this issue too.

As more open access networks get deployed, we envision network operators to cooperate among themselves for bringing in services that are not available locally. In such scenarios, the network operator could have reduced visibility about a remote service provider (in the domain of another network operator) and controlling WiFi based on such complex service specific dynamics would be a challenging task.

REFERENCES

[1] Adrián González, Arturo Vergara, Antolín Moral, and Jorge Pérez. Prospects on ftth/ep2p open access models. In *Federation of Telecommunications Engineers of the European Union (FITCE) 49th Congress*, 2010.

[2] Institute for Local Self-Reliance. Community Networks. https://muninetworks.org/content/ammon-id-experimenting-open-access-ftth-network.

[3] FTTH Council Europe. Fibre broadband flourishes as Switzerland joins the league of FTTH leaders. http://www.ftthcouncil.eu/documents/PressReleases/2014/PR2014_EU_Ranking_Stockholm_FINAL.pdf.

[4] NoaNet, Northwest Open Access Network. http://www.noanet.net/technology/default.aspx.

[5] Marco Forzati, Claus Popp Larsen, and Crister Mattsson. Open access networks, the swedish experience. In *Transparent Optical Networks (ICTON), 2010 12th International Conference on*, pages 1–4. IEEE, 2010.

[6] Bruno Astuto A Nunes, Marc Mendonca, Xuan-Nam Nguyen, Katia Obraczka, and Thierry Turletti. A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications Surveys & Tutorials*, 16(3):1617–1634, 2014.

[7] Josh Kunz, Christopher Becker, Mohamed Jamshidy, Sneha Kasera, Robert Ricci, and Jacobus Van der Merwe. Openedge: A dynamic and secure open service edge network. In *Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP*, pages 257–264. IEEE, 2016.

[8] William Lehr, Marvin Sirbu, and Sharon Gillett. Broadband open access: Lessons from municipal network case studies. In *Proceeding of the TPRC conference*, 2004.

[9] Matthew Gast. *802.11 wireless networks: the definitive guide.* "O'Reilly Media, Inc.", 2005.

[10] Adrian Lara, Anisha Kolasani, and Byrav Ramamurthy. Network innovation using openflow: A survey. *IEEE communications surveys & tutorials*, 16(1):493–512, 2014.

[11] Open vSwitch - an open virtual switch. http://openvswitch.org/.

[12] Yiannis Yiakoumis, Kok-Kiong Yap, Sachin Katti, Guru Parulkar, and Nick McKeown. Slicing home networks. In *Proceedings of the 2nd ACM SIGCOMM workshop on Home networks*, pages 1–6. ACM, 2011.

[13] Vijay Sivaraman, Tim Moors, Hassan Habibi Gharakheili, Dennis Ong, John Matthews, and Craig Russell. Virtualizing the access network via open apis. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*, pages 31–42. ACM, 2013.

[14] Emulab - network emulation testbed. http://emulab.net/.

[15] hostapd. https://w1.fi/hostapd/.

[16] OpenWrt. https://openwrt.org/.

[17] Ryu SDN Framework. https://osrg.github.io/ryu/.

[18] Mininet. http://mininet.org/.

[1] In our work we have considered WiFi capability only for service consumers