Introducing Wireless Access Programmability using Software-Defined Infrastructure

Thomas Lin, Hadi Bannazadeh, and Alberto Leon-Garcia
The Edward S. Rogers Sr. Department of Electrical and Computer Engineering
University of Toronto
Toronto, ON, Canada
Email: {t.lin, hadi.bannazadeh, alberto.leongarcia}@utoronto.ca

***Abstract*—Programmability in wireless access networks can provide unprecedented flexibility in meeting the communications needs of a diverse set of wireless devices under changing demand and network conditions. Programmability also holds the promise of enabling multiple simultaneous virtual operators providing a variety of access networks that offer different services using different business models. In this paper we present our work on enabling wireless access network programmability in the SAVI testbed. We introduce an architecture for Software-Defined Infrastructure that offers virtualized heterogeneous resources in support of services and applications. Central to this architecture is the *Janus* SDI resource manager that can coordinate the actions of a set of controllers, e.g. OpenStack, OpenFlow, and other controllers such as a wireless access controller. We describe a design where Janus is used to integrate the wireless access network into a Smart Edge node. We introduce use cases that exploit the flexibility offered by this design, and we present experimental results from our implementation.**

Keywords—Wireless Access; Virtualization; Software-Defined Infrastructure (SDI); Software-Defined Networking (SDN)

I. INTRODUCTION

The flexibility and economies of scales that derive from cloud computing are driving a revolution in the architecture and operation of communication networks. Software-Defined Networking (SDN) and OpenFlow [1] promise to reduce capital expenses by exploiting the network resources offered by commodity switches. SDN controllers promise a rich set of network services that result from the programmability of flows in the network. The network services, however, ultimately exist to support applications, and so the network controller must ultimately respond to the needs of applications. Thus, the international activities on Network Function Virtualization (NFV) are focused on the creation of network functions that can be chained to provide network services.

With NFV it becomes immediately clear that the set of resources under consideration must include computing as well as networking, and this introduces the problem of how to manage heterogeneous resources. While the initial NFV activities focus on the exploitation of virtualized computing resources to reduce cost and increase the flexibility of networks, a case can be made that the set of resources should include unconventional resources such as FPGA programmable hardware, GPUs, and Software-Defined Radio (SDR) systems and components. The virtualization and integrated control and management of these diverse resources can provide high performance and reduce costs in settings where cloud computing approaches do not. To meet these challenges, the SAVI research network has been exploring the role of a Software-Defined Infrastructure [2] (SDI) in support of future application platforms.

SAVI is a Canadian research network investigating future application platforms [3]. A key goal of SAVI was to develop an experimental testbed to support experimentation with future network architectures, services, and applications [4]. SAVI's focus on future application enablement dictated that cloud computing and virtual networking had to be investigated jointly. This led SAVI to introduce the notion of SDI and to design resource clusters with virtualized heterogeneous resources under integrated control.

The SAVI testbed, first deployed in 2013, features a rich array of available computational resources beyond that of traditional virtual machines (VMs), all connected via an OpenFlow-enabled network substrate. As the testbed contains various types of resources, a key aim in the design was to virtualize all aspects of the infrastructure. To meet the management needs of the testbed, SAVI proposed SDI and developed an SDI resource management system [5] to provide converged control and management over heterogeneous resources. SAVI's SDI manager, codenamed *Janus*, offers a top-level management system which has a global view over the entire infrastructure and the state of all resources, both physical and virtualized.

In this paper we focus on SAVI's efforts to introduce programmability into the wireless access network. While other experimentation testbeds use OpenFlow to expand the research potential for their users, what is still lacking in many is the ability to tightly integrate and experiment with wireless devices. Unlike OpenFlow, which was quickly and widely adopted as a way to virtualize the network forwarding elements, there is a lack of consensus on the best way to virtualize, and provide an abstraction for, wireless networks and devices. The proliferation of wireless-enabled devices, vehicles, sensors, and etc. presents a potentially growing research field relating to how to best manage the wireless spectrum and share the access devices that bridge the wireless to the wired. To empower these researchers on the SAVI testbed, the first steps are to extend SDN and OpenFlow to the access networks, virtualize the wireless access points, and integrate them with the Janus management system. This would enable the connection of

wireless devices and open opportunities for the experimentation and validation of novel applications.

This paper will present our ongoing work to introduce programmability by integrating virtualizable wireless access points into the SAVI testbed, managed within the Janus SDI framework. The organization of this paper is as follows: Section II presents background and related works regarding wireless virtualization initiatives. A brief high level overview of the SAVI testbed and SDI are then presented in Section III. This is followed by Section IV, which describes our architectural design to enable virtualized wireless access in the SAVI testbed, as well as a discussion regarding use-cases. A summary of our work to enable Wi-Fi access points in the SAVI testbed is then documented in Section V. Section VI presents a preliminary evaluation of our working access points, as well as a proof-of-concept demonstration that ties in quality of service (QoS). Future work is discussed in Section VII, which is followed by the conclusion in Section VIII.

II. Background & Related Works

Network virtualization is an active research topic that has received much attention for at least the last twenty years [6]. A subset of this topic is the area of wireless network virtualization, which has attracted attention as wireless access has become the default approach to accessing applications. The number of wireless devices and sensors, as well as the traffic that they generate, will grow at an incredible pace over the next several years [7]. To meet this future demand, researchers have been exploring ways to share the wireless space via virtualization. As this concept of sharing is one of the core tenets of cloud computing, it makes sense for a cloud-based experimental testbed to include wireless technologies to facilitate the exploration and testing of future wireless virtualization concepts.

As early as 2006, the Global Environment for Network Innovations (GENI) [8] initiative has explored the possibility of utilization virtualization to support the sharing of wireless resources amongst multiple concurrent experiments. The authors of [9] studied various types of wireless networks, as well as different potential techniques to achieving wireless virtualization. The study included discussions on ways to map those techniques to applications and experiments that may vary in length from a few hours to several months, and ended with a list of recommendations for different wireless network types on how best to achieve virtualization and slicing.

In recognizing that future wireless network infrastructures will be composed of different technologies working in tandem, Nakauchi et al. proposed AMPHIBIA [10], a Cognitive Virtualization Platform designed to virtualize wired and heterogeneous wireless networks. The design of AMPHIBIA was motivated by the desire to achieve end-to-end slicing from the wired to the wireless domain. The authors suggest that this platform will enable service providers the ability to dynamically provision virtual network slices on-demand.

To empower wireless researchers, Yap et al. introduced OpenRoads [11], a platform which enables SDN control over wireless access points (WAPs). OpenRoads, also known as OpenFlow Wireless, leverages the capabilities of the OpenFlow protocol to enable control over the datapath while using SNMP to configure the WAPs. Both OpenFlow and SNMP are then exposed to applications through an OpenRoads management interface, which provides users the ease of managing and controlling various types of wireless technologies. The initial deployment of OpenRoads into a production network, the Stanford campus, was later reported in [12].

The research plan within SAVI includes an exploration of wireless virtualization and its role in application platforms. In its first phase, a SAVI team at McGill University has completed the design of a wireless virtualization platform named Aurora [13] [14]. The goal of Aurora is to become a unifying framework to ease the deployment and implementation of virtualized wireless networks in heterogeneous wireless ecosystems. The current implementation of Aurora is able to provide fully functional basic services for virtualizing IEEE 802.11/Wi-Fi access networks. Preliminary work exploring the integration of Aurora into SAVI's SDI management system has been prototyped as an experiment on the SAVI testbed [14].

III. The SAVI Testbed and SDI

The Canadian SAVI testbed is a multi-tier cloud application platform. As seen in Fig. 1, the tiers of the infrastructure comprise massive scale core datacentres, connected to various smaller "Smart Edge" datacentres, which may additionally be connected to access nodes that allows the inclusion of end-user clients into the SAVI testbed. The core nodes primarily consist of large numbers of traditional compute resources (i.e. virtual machines) as well as large storage capacities. Conversely, the Smart Edge nodes are smaller in scale and located closer to the end-users for hosting services that may require low-latency. In addition, the Smart Edges are able to offer non-conventional computing resources including programmable hardware (FPGAs), general purpose GPUs (GP-GPUs), and etc. The access nodes are to provide end-user clients and personal devices direct connectivity to the services in the testbed via wireless access. As the SAVI testbed is built on a virtualized infrastructure, it will enable experimenters and service-application developers the ability to quickly deploy, maintain, and retire their resources. This virtualized infrastructure thus provides rapid repurposing of the infrastructure resources in an elastic manner.

To manage and control such a diverse testbed, the SAVI project conceived of Software-Defined Infrastructure as a way to perform converged management of heterogeneous infrastructures. An SDI resource management system has a *global view* of the infrastructure and all its resources, both physical and virtual. This global view is stored within a topology manager (see Fig. 2), which stores the configuration and topology information related to all the resources. The SDI resource management system's ability to control the

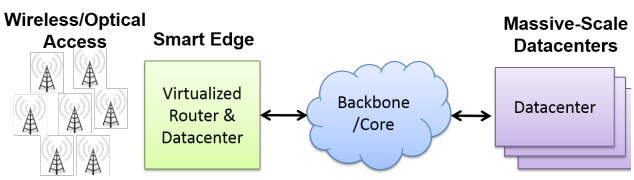

Fig. 1. SAVI multi-tier cloud

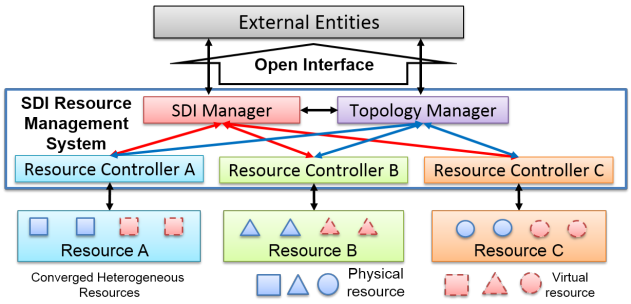

Fig. 2. High-level architecture of SDI resource management system

heterogeneous resources in the infrastructure relies on a set of proxy resource controllers, one for each type of resource, that act as actuators to affect the related resource type (e.g. network, VMs, FPGAs, GPUs, sensors, etc.). Similar to how an SDN controller provides an abstraction for network control applications to run on top and define global network control policies, the SDI resource management system offers a set of open interfaces (i.e. a set of APIs) enabling external entities to query information, provision resources, and manage their share of the infrastructure. It is anticipated that applications built on top of this set of open interfaces will enable novel integrated resource management schemes that leverage the global view of the infrastructure's heterogeneous resources.

The SAVI testbed's resource management system involves an SDI manager codenamed *Janus*. For the compute and storage resources of the testbed, Janus leverages OpenStack [15] as the proxy resource controller. For network control, Janus uses the Ryu OpenFlow controller [16] as its proxy. The use of proxy controllers grants the SDI resource management system two major benefits:

1. Specific resource controllers can be easily swapped, while maintaining the same abstraction to the external entities relying on the SDI's northbound abstraction. This may open up control and management capabilities formerly inaccessible with the old controller, and in turn allows the SDI manager to expose them in its northbound abstraction interface;
2. New resource types can be more quickly integrated into the existing testbed by finding an adequate controller for that type of resource, and developing a simple RPC method (e.g. RESTful APIs) to facilitate communication between the controller and the SDI manager.

In summary, new resource controllers enable new capabilities and new resource types to be added into the SAVI testbed while Janus maintains its global view over the infrastructure. This enables Janus to keep its place as a converged control and management framework over all sets of resources. In turn, this benefits the owners of experiments and applications that build upon Janus' northbound interfaces, and supports the ongoing innovation of future technologies.

IV. Wireless Access on the SAVI Testbed

The enablement of wireless access in the SAVI testbed will allow users of the testbed to engage in novel end-to-end experiments involving mobile devices. This will open many possibilities in regards to applications and experiments related to mobility and handover, as well as support other innovations related to wireless technologies. In order for Janus, the SDI manager, to control and manage the access points themselves, it requires a resource controller capable of interfacing with various access point technologies. It is our intention to integrate the Aurora wireless virtualization framework as a new resource controller working under Janus. This integration would expose wireless-related management options, and offer application developers and experimenters the ability to dynamically configure virtual wireless networks, create virtual bridges, create virtual interfaces, and etc. in a programmatic fashion.

This section will describe the high-level architectural design for integrating wireless access points into the SAVI testbed. The first subsection will provide a description of how we extend SDI control over the access points of the testbed, followed by the second subsection which contains a brief discussion regarding the potential use-cases of virtualized wireless access nodes.

A. SDI Control of Access Points

An access point's role is not simply limited to bridging the wired and wireless networks. As we envision all resources to be virtualizable and sharable amongst many tenants, the access points used within the SAVI testbed must enable dynamic reconfiguration of its wireless networks and their related parameters. In addition, they must support the ability for the SDI manager to control their traffic via SDN principles. Thus, we require that the access points be OpenFlow-enabled. This allows the SDI manager to configure the flow table entries within the access points via its proxy SDN controller, and thus manage the traffic entering and leaving the testbed through the wireless network. Managing the traffic directly at the access points opens the door for introducing NFV capabilities at the edge of the network, prior to the traffic entering the wired portion of the testbed. The configuration of the access point devices themselves require a new access point (AP) controller capable of interfacing with them in order to adjust their wireless settings and parameters on-the-fly. This high-level architecture, presented in Fig. 3, will enable the ability to create new virtual wireless networks, each with different parameters (e.g. channel, quality of service level, signal strength, etc.).

As the SAVI testbed supports multiple tenants (projects that include at least one user), this allows us the ability to create a unique virtual wireless network per tenant. Users with a wireless-enabled device will be able to choose which tenant they would like to connect their mobile device to. To secure the

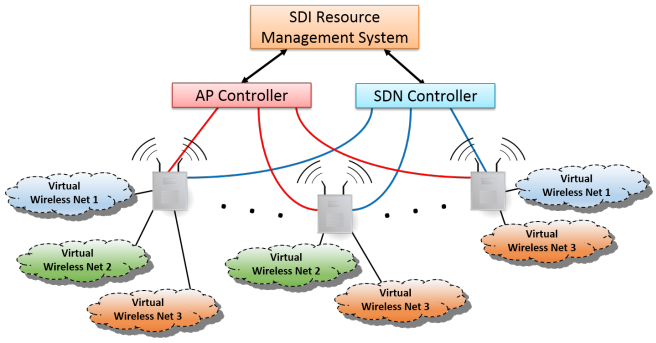

Fig. 3. High-level architecture for wireless access integration w/ SDI

wireless access, the mobile clients who connect to a specific virtual wireless network must be registered with Janus before their traffic is allowed to enter the testbed. Once registered, a user's mobile device will essentially become another end-host resource in the testbed. We note that the flexibility afforded by the ability to dynamically spawn new virtual wireless networks enables the possibility of virtualizing at granularities finer than a per-tenant basis. A tenant running multiple wireless services may create several virtual wireless networks, each associated with their own unique set of parameters.

B. Mobile Wireless Access Points

For the access points to be integrated into the SAVI testbed and controlled by Janus, they need not require a physical connection. It is possible to interface an access point device with the testbed by creating a network tunnel between the two. This opens the door for the possibility to have an SDN-enabled access network which is remotely managed and controlled by Janus. For example, if an access point device was connected to a publicly accessible IP address in a public setting (e.g. free internet at a café, satellite connection, etc.), then it can set up a tunnel over the internet to connect to the SAVI testbed. This remote connection of the access point would enable mobile clients within its proximity to access existing services, applications, and experiments hosted on the testbed.

A mobile SAVI access point opens the door for many use-cases and applications. We list a few potential use-cases and elaborate on them:

- **Disaster Areas**: In regions where the civil infrastructure has been heavily affected by natural, social, or technological catastrophes, the regular communication infrastructure that services the area may be damaged. Emergency response units who are sent into the area may need access to applications which are hosted on Smart Edges. Dynamically creating access nodes that enable mobile devices to access these Smart Edge resources would benefit and aid first responders in their tasks. Such a system would similarly be beneficial for people working in remote locations who require access to data and services within the Smart Edge.
- **Large Social Congregations**: Events involving large gatherings of people in a social setting such as concerts or sport events often involve many individuals using smartphones to communicate with others in real time. The assembly of many mobile devices into a relatively small geographical area may strain the existing wireless access infrastructure in place. The creation of dynamic wireless access networks, coupled with an access point coordination strategy to control associations [17], would alleviate the burden created by the sudden increase in wireless traffic.
- **Differentiated Service Levels**: A single tenant may host several services, each of which may involve their own unique virtual access network. Each virtual access network can also be associated with different wireless attributes (e.g. guaranteed bandwidth, traffic priorities, etc.). A service provider may further associate these wireless attributes with business metrics such as cost for the purpose of charging users. The dynamic creation of differentiated wireless access networks within a single physical infrastructure allows users to receive their preferred service irrespective of location.

V. Enabling Wi-Fi on the SAVI Testbed

In this section we present our work to integrate Wi-Fi-based access points into the SAVI testbed. While this integration effort is related to [18] [14], the efforts presented in this paper concentrate on the full utilization of OpenFlow to control the traffic and provide tenant isolation. The current WAPs used are PC Engine Alix3d2 boards with support for IEEE 802.11b/g. For customizability of the WAPs, we utilize OpenWrt [19], an open source Linux-based embedded operating system for wireless routers. OpenWrt provides the ability to broadcast multiple SSIDs, with each SSID mapped to a different virtual interface within the operating system. This ability is a simple virtualization technique in that end-users will get the illusion of multiple WAPs when in fact there is only one physical WAP. As we create a unique SSID per tenant, we are able to map the wireless traffic from each tenant to a single virtual interface.

The use of a Linux-based operating system allows us to compile and employ Open vSwitch [20] (OVS), a software-based OpenFlow switch, for use within the WAPs. Inside the OpenWrt operating system, each of the virtual interfaces may then be connected to an OVS. Thus, the traffic traversing through the access points can be controlled by the SAVI testbed's central SDN application, which runs on the Janus SDI manager [21]. When the virtual interfaces (which are each mapped to a unique SSID/tenant) are connected to the OVS, the port number associated with the interface must be registered with Janus to indicate which tenant it belongs to. Similarly, mobile clients who connect to a specific tenant SSID must have their MACs registered with Janus before their traffic is permitted to enter the testbed. Fig. 4 shows a sequence diagram indicating the steps required for a mobile client to successfully connect to the SAVI testbed and acquire a useable IP address.

When a mobile client attempts to negotiate a connection with the WAP, this association process is handled by a local software process running within OpenWrt. Upon the successful association of a new client device, a local script is responsible for automatically registering the MAC address of the mobile

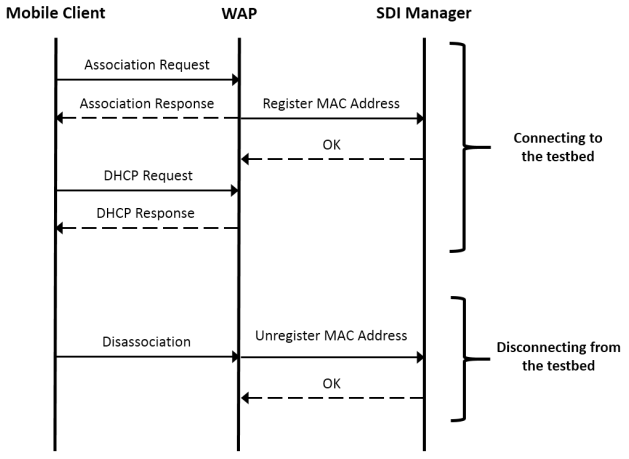

Fig. 4. Mobile client connection & disconnection sequence diagram

client with the SDI manager. Without this explicit registration of the MAC address, the client will still be able to connect to the WAP, but its DHCP requests will be dropped by the SDI manager, thus leaving it with no IP address. Once this step has been completed, the mobile client is able to communicate with the rest of the resources within the same tenant.

Disconnecting from the testbed follows a similar process. When a mobile devices disassociates from the WAP, a local script is executed which will notify the SDI manager to un-register the client's MAC address from the tenant and the testbed. Optionally, we can also choose to send an explicit DHCP release message. However, we currently avoid doing so in the event that the mobile client re-connects a short time later (i.e. within the DHCP lease time), it may re-acquire its old IP address.

As the WAPs are running a Linux-based operating system, they come with the Linux *tc* traffic control tool. In addition, OVS itself comes with support for configuring multiple queues per port with different QoS parameters associated with each queue. Thus, OVS is able to provide basic traffic policing, classification, queueing, and bandwidth guarantees by leveraging tc under-the-hood. The OpenFlow protocol also supports an action to enqueue an outgoing packet on a certain queue of an output port. Thus, with each SSID represented as a port on the OVS, the SDI manager has the option to enforce QoS on either a coarse grain scale (i.e. individualized traffic parameters per port/SSID) or on a fine grain scale (i.e. creating multiple queues per port and using OpenFlow to map specific flows to each queue).

VI. Evaluation and Use-Case

At the time of writing this paper, we have successfully enabled wireless access within a SAVI node located at the University of Toronto wherein the SDI manager is responsible for controlling and isolating the traffic entering and leaving the WAP. We first present a preliminary evaluation on the bandwidth of the WAP. Afterwards, we validate its practicality by presenting a simple experiment designed as a proof-of-concept to demonstrate the ability to use the SDI manager to enforce quality of service for wireless clients.

A. Preliminary Evaluation

In order to determine the throughput provided by the WAPs, we conducted a series of measurements to find the uplink and downlink bandwidths. We employed *iperf*, a bandwidth measurement tool, to run a series of measurements and collected data for both uplink and downlink. The measurements were conducted for both UDP and TCP traffic, and the averages of the sessions were then calculated. TABLE I summarizes our findings.

Our first observation is that the UDP throughput is generally higher for all three columns, but the range between minimum and maximum is greater as well. This was expected as UDP lacks the flow control mechanisms of TCP. We note that the average throughput of both UDP and TCP appears to be slightly higher than those observed by Yap et al. in [12] for the single SSID case, while running the same type of hardware access point. We believe that this is likely due to the lack of packet encapsulation in our setup. These results show that our current

TABLE I. Wireless Access Point Bandwidth Measurements

	Min. (Mbps)	Avg. (Mbps)	Max. (Mbps)
UDP Uplink	15.5	23.15	27.2
UDP Downlink	13.2	17.59	22
TCP Uplink	10.4	16.538	19.6
TCP Downlink	13.3	16.01	17.2

wireless bandwidth is a very limited resource. Thus, even simple applications can easily act as a contender for bandwidth and degrade the quality of concurrently running applications and experiments. The next section showcases our work to implement quality of service within the wireless access points.

B. Video Streaming Use-Case

Envision a scenario in which a user's experiment, which requires some level of guaranteed bandwidth, finds itself short of the bandwidth it requires. Using the knowledge provided by the SAVI topology manager on the SDI resource management system, as well as the SAVI monitoring and measurement system [22], the user can either route their traffic around the bottleneck region(s) of the network, or utilize traffic priority queueing. While both are feasible solutions, for the purpose of observing the traffic control capabilities of the integrated WAPs, we opt to run a simple experiment involving the latter.

An experiment was set up wherein a single VM running within the testbed serves as a streaming video source. A mobile device in the form of a laptop was connected to the testbed via a WAP, and serves as a client for the streaming video. Simultaneous with the video streaming, another server within the testbed will be conducting a "separate experiment", called Experiment X, with a secondary laptop, also connected to the same WAP as the first laptop. This setup is illustrated in Fig. 5. Experiment X will involve large amounts of traffic, with the goal of consuming bandwidth on a best-effort basis. This will be done in the form of maximum-size ICMP packets (i.e. 65507 bytes) sent at a rate of up to 40 packets per second, resulting in traffic that consumes roughly 20 Mbps of bandwidth. Based on the throughput measurements of the previous subsection, it is

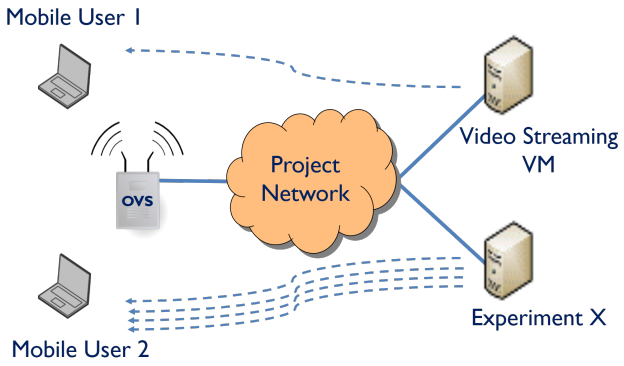

Fig. 5. Setup of video streaming w/ QoS demonstration

expected that the bottleneck for the experiment will be the WAP itself as the SAVI wired network provides a minimum of 1 GE bandwidth. Furthermore, the experiment was conducted on an isolated region of the testbed, ensuring that no extraneous traffic from other experiments will contest the bandwidth.

For the OVS running within OpenWrt on the WAP, we created two queues for the egress traffic: one high priority queue that enforces a minimum bandwidth guarantee and a default queue for everything else. This configuration was enforced via the Linux Hierarchical Token Bucket [23] queuing discipline. Upon detection of traffic related to the video streaming experiment, the network controller installs a rule into the flow table of the OVS to enqueue packets of the video flow into the higher priority queue. Fig. 6 shows the video's unique profile under optimal operating conditions (i.e. without Experiment X running). When the video is streamed alongside Experiment X, the ICMP packets congest the bandwidth of the WAP thus resulting in a degradation of the video. This can be visualized when observing the bandwidth usage for both experiments in Fig. 7. When queueing is utilized to provide the video stream a minimal bandwidth guarantee, we observe in Fig. 8 is that the bandwidth utilization of the video returns to its normal state, similar to Fig. 6.

These results show that a WAP running the OpenWrt system coupled with OVS is able to correctly enforce QoS parameters specified by the controller. Users and applications building upon the infrastructure abstraction provided by the SDI manager will greatly benefit from its ability to dynamically specify QoS levels for specific flows and leverage OVS to enforce them.

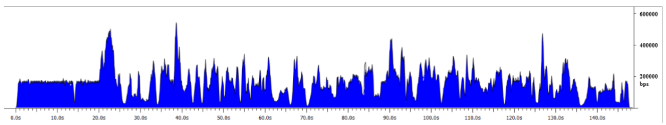

Fig. 6. Video clip profile

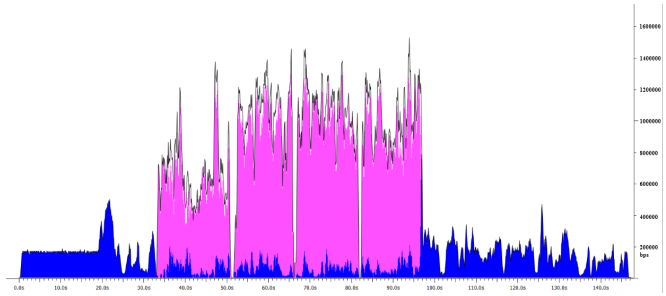

Fig. 7. Video clip profile (No traffic control)

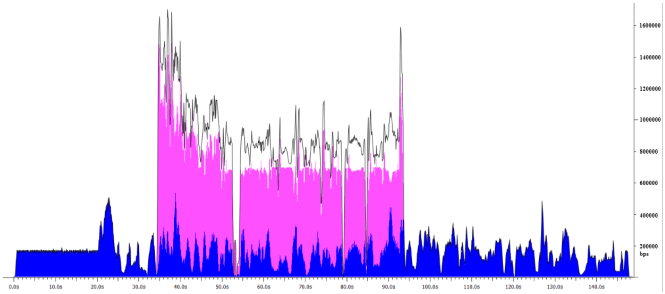

Fig. 8. Video clip profile (With OpenFlow/OVS traffic control)

VII. Future Work

The work presented in this paper represents the first step towards realizing a fully virtualized, programmable wireless access infrastructure. Moving forward, we wish to work towards the full integration of the Aurora wireless virtualization framework into the SAVI SDI resource management system. In regards to the throughput of our existing WAPs, we believe that careful tuning of the wireless parameters in OpenWrt will result in better performance. As the SAVI testbed currently includes other wireless resources such as software-defined radios and cellular base stations, research into how to best virtualize and share these resources will be required. Further coordination between the Toronto and McGill SAVI teams are expected in order to ensure that Aurora continues to evolve such that it can virtualize these access technologies.

VIII. Concluion

The SAVI testbed currently offers a mix of virtualizable computing and networking resources which are controlled in an integrated fashion by an SDI resource management system. Janus, the SAVI SDI manager, utilizes proxy controllers to affect the state of infrastructure resources, while exposing a programmatic interface for external users to provision and manage their share of the resources. This paper reported on our work towards introducing virtualizable wireless access resources into the SAVI testbed. We proposed a design which interfaces Janus with a wireless access controller capable of virtualizing access points, thus opening further opportunities for application developers and experimenters on the testbed. The programmatic interface on Janus enables users to dynamically create multiple virtual wireless networks on access points, in which each network may be associated with individualized parameters. We included descriptions of use-cases where virtualized wireless access points may be mobilized while remaining connected to the SAVI testbed, and controlled by Janus. We then presented our implementation which integrated Wi-Fi access points into the SAVI testbed, wherein a different virtual wireless access network was created for each tenant. This integration work enabled Janus to control the incoming and outgoing wireless traffic. An experiment was shown as a proof-of-concept showcasing the ability of the SDI manager to enforce different QoS parameters onto flows within the virtualized wireless access points.

Acknowledgment

The authors would like to acknowledge the contributions of Michael Smith, Kevin Han, Heming Wen, and the rest of the McGill SAVI team for their support bringing up the wireless access points.

This work and all the published papers which follow are funded in part or completely by the Smart Applications on Virtual Infrastructure (SAVI) project, funded under the National Sciences and Engineering Research Council of Canada (NSERC) Strategic Networks grant number NETGP394424-10.

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," *ACM SIGCOMM Computer Communication Review,* vol. 38, no. 2, pp. 69-74, 2008.

[2] J.-M. Kang, H. Bannazadeh, H. Rahimi, T. Lin, M. Faraji and A. Leon-Garcia, "Software-Defined Infrastructure and the Future Central Office," in *2013 IEEE International Conference on Communications Workshops (ICC)*, Budapest, Hungary, 2013.

[3] Smart Applications on Virtual Infrastructure, "Smart Applications on Virtual Infrastructure," [Online]. Available: http://www.savinetwork.ca/. [Accessed 15 August 2014].

[4] J.-M. Kang, H. Bannazadeh and A. Leon-Garcia, "SAVI Testbed: Control and Management of Converged Virtual ICT Resources," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, Ghent, Belgium, 2013.

[5] J.-M. Kang, T. Lin, H. Bannazadeh and A. Leon-Garcia, "Software-Defined Infrastructure and the SAVI Testbed," in *9th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TRIDENTCOM 2014)*, Guangzhou, People's Republic of China, 2014.

[6] G. Woodruff, N. Perinpanathan, F. Chang, P. Appanna and A. Leon-Garcia, "ATM Network Resources Management using Layer and Virtual Network Concepts,," in *IEEE Symposium on Integrated Network Management*, 1997.

[7] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013-2018," 2014. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.pdf. [Accessed 25 September 2014].

[8] Global Environment for Network Innovations, "Global Environment for Network Innovations," [Online]. Available: http://www.geni.net/. [Accessed 15 August 2014].

[9] S. Paul and S. Seshan, "Technical Document on Wireless Virtualization," 15 September 2006. [Online]. Available: http://groups.geni.net/geni/raw-attachment/wiki/OldGPGDesignDocuments/GDD-06-17.pdf. [Accessed 28 September 2014].

[10] K. Nakauchi, K. Ishizu, H. Murakami, A. Nakao and H. Harada, "AMPHIBIA: A Cognitive Virtualization Platform for End-to-End Slicing," in *IEEE International Conference on Communications (ICC)*, Kyoto, 2011.

[11] K.-K. Yap, M. Kobayashi, R. Sherwood, N. Handigol, T.-Y. Huang, M. Chan and N. McKeown, "OpenRoads: Empowering Research in Mobile Networks," *ACM SIGCOMM Computer Communication Review,* vol. 40, no. 1, pp. 125-126, 2010.

[12] K.-K. Yap, M. Kobayashi, D. Underhill, S. Seetharaman, P. Kazemian and N. McKeown, "The Stanford OpenRoads Deployment," in *4th ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH)*, Beijing, 2009.

[13] P.-K. Tiwary, K. Han, H.-P. Truong, Q.-H. Ho and T. Le-Ngoc, "Aurora: A Virtualization and Software-Defined Infrastructure Framework for Wireless Networks," in *SAVI 2014 Annual General Meeting (AGM) Student Posters*, Toronto, SAVI Project, 2014, pp. 73-74.

[14] H. Wen, "Virtualization and Software-Defined Infrastructure Framework for Wireless Access Networks," Montreal, Canada, 2014.

[15] OpenStack, "OpenStack," [Online]. Available: http://www.openstack.org/. [Accessed 17 August 2014].

[16] NTT DoCoMo, "Ryu SDN Framework," [Online]. Available: http://osrg.github.io/ryu/. [Accessed 17 August 2014].

[17] M. Derakhshani, X. Wang, Le-Ngoc, Tho, Leon-Garcia and Alberto, "Improving Throughput and Fairness in Virtualized 802.11 Networks through Association and Airtime Control," *IEEE Transactions on Wireless Communications,* Submitted for publication, 2015.

[18] H. Wen, K. Han, M. Smith, P. K. Tiwary and T. Le-Ngoc, "802.11 Wireless Access Point Virtualization Testbed," in *SAVI 2013 Annual General Meeting (AGM) Student Posters*, Toronto, SAVI Project, 2013, pp. 56-57.

[19] OpenWrt, "OpenWrt: Wireless Freedom," [Online]. Available: https://openwrt.org/. [Accessed 21 August 2014].

[20] Open vSwitch, "Open vSwitch," [Online]. Available: http://openvswitch.org/. [Accessed 17 August 2014].

[21] T. Lin, J.-M. Kang, H. Bannazadeh and A. Leon-Garcia, "Enabling SDN Applications on Software-Defined Infrastructure," in *IEEE Network Operations and Management Symposium (NOMS)*, Krakow, Poland, 2014.

[22] J. Lin, R. Ravichandiran, H. Bannazadeh and A. Leon-Garcia, "Monitoring and Measurement as a Service in SDI Deployed on SAVI Testbed," in *SAVI 2014 Annual General Meeting (AGM) Student Posters*, Toronto, SAVI Project, 2014, pp. 91-92.

[23] M. D. (devik), "HTB Linux queuing discipline manual - user guide," [Online]. Available: http://luxik.cdi.cz/~devik/qos/htb/manual/userg.htm. [Accessed 21 August 2014].