On Software-Defined Wireless Network (SDWN) Network Virtualization: Challenges and Open Issues

AHMED ABDELAZIZ1*, ANG TAN FONG1, ABDULLAH GANI1, SULEMAN KHAN2, FAIZ ALOTAIBI3 AND MUHAMMAD KHURRAM KHAN4

1Faculty of Computer Science and Information Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia
2School of Information Technology, Monash University, Malaysia
3Faculty of Computer Science and Information Technology, University Putra Malaysia, 43400 Serdang, Selangor, Malaysia
4Center of Excellence in Information Assurance (CoEIA), King Saud University, Saudi Arabia

*Corresponding author: ahmedaziz@siswa.um.edu.my

Software-defined networking (SDN) is a new network architecture that emerges as to implement network virtualization (NV) with vast features, especially when it is applied in multi-tenant scenarios. The rapid growth of wireless network applications and services, let to adopt NVs into software-defined wireless network (SDWN). This is because wireless networks require specific features that can be hindered of implementing NVs such as updated location information, dynamic channel configuration, and rapid client re-association. This paper presents state-of-the-art NV methods for SDWN with the aim of highlighting issues and challenges of applying NVs techniques of SDN into SDWN. We discuss three SDN techniques that facilitate NV in the cloud, namely proxy-based virtualization, layer two prefixes-based virtualizations and programing language-based virtualization. Moreover, the paper points out the possibility of providing effective VNIs in the SDWN architecture. We also taxonomy the SDWN proposed virtualization methods based on hypervisor controller in the different networks. Finally, the potential requirements and challenges and open issues of SDWN NVs are also identified and presented as the future directions in SDWN research.

Keywords: software-defined wireless network; software-defined network; OpenFlow; network virtualization

Received 8 December 2016; revised 7 June 2017; editorial decision 8 June 2017
Handling editor: Linghe Kong

1. INTRODUCTION

Software-defined networking (SDN) [1] is a new architecture that is built and designed based on two significant concepts: (i) decouple data plane from the control plane [2] and (ii) centralize control of network [3]. Obviously, these remarkable features of SDN provide a more flexibility, programmability [4] and well-managed virtual networks. SDN is considered as promising network architecture to facilitate network virtualization (NV) that can resolve the existing issues of the traditional network. Recently, NV [5] is commonly used in different types of the networks such as wireless and mobile network to facilitate higher resources utilization and lower cost. Moreover, NVs are a core technology deployed in many test-beds such as GENI [6], Planet Lab [7] and TUNIE [8] to manage broadly the virtual slices and to isolate network traffic between user experiments. NVs have been started in computer networks for decades. For instance, VLANs [9] and VXLAN [10] are widely used in the traditional virtual networks. VLANs use tags for virtualization of the network and have the only capability to accommodate 4,096 VLANs while experiencing the scalability
problem. To overcome this limitation, IEEE introduced VXLAN [11] to provide a large scale of virtual domains by extending the tags attached. However, VLAN and VXLAN still have limitations such as Layer two boundaries, which constrain virtual machine (VM) [12] mobility.

The capacity of the existing wireless architecture is becoming insufficient to cater a lot of services demand and provide effective NVs [13]. It is suffering from several issues such as assurance the isolation property, node virtualization and virtual resource allocation [14]. The NV is not a new concept in the wireless network domain because several wireless NVs (WNVs) [15] conceptual schematic architectures are proposed. Nevertheless, WNVs encounters some difficulties to evolve and deploy new technologies and functions because of tight coupling between hardware and wireless protocols. Therefore, WNVs can hardly meet such great expectations without fundamental architectural changes. To overcome the limitations of traditional wireless NV, software-defined wireless networks (SDWNs) [16] have emerged based on the SDN paradigm. SDWN aims to study the network architecture and a series of relative crucial technologies for the future mobile and wireless network. SDWN is still in its infancy; however, the focus now is more on the SDWN architecture.

This paper provides a comprehensive review of the works that have already been done to achieve wireless NV based on SDWN architecture. We discussed existing NVs approaches for SDNs [17] and how to extend these approaches into SDWN network. The core contributions of the paper are manifolds: (i) comparing the traditional network and SDN architecture network in terms of virtualization layer and show three approaches used by SDN to provide NVs, (ii) classifying the survey paper and illustrated thematic taxonomy based on proxy and layer two virtualization, and (iii) finally, we highlight various challenges and open issues of SDWN NV.

The rest of this paper organized, as follows. Section 2 provides a brief overview SDN virtualization layer and three approaches that used to provide NVs. In Section 3, we discuss the main components of SDWN architecture. Section 4 presents a classification of the hypervisor controller-based type of the network. Finally, challenges and open issues to implement NVs in SDWN are analysed in Section 4.

2. SDN NV

SDN is a revolutionary idea in computer networking [18], which allows centralized management functionality, separates the control plane from the data plane and enables innovation through network programmability. Subsequently, SDN [19] facilitates the implementation of NV, which tolerates the virtualization of network infrastructure with optimization, performance isolation and minimal cost of hardware resources. The main disadvantage of conventional network architecture [20] is to support individual control plane and individual data plane. For example, in the router, the control plane is responsible for running network controls, such as routing algorithms (e.g. RIP [21], OSPF [22] and BGP [23]). At the same time, the router manages network control protocols (e.g. ICMP [24]), in which the forwarding tables and hardware data paths are implemented. However, the data plane in SDN is decoupled from the control plane. This separation provides a more flexible NV with strong isolation between VNs or slices. In addition, each virtual control plane has access to a portion of the data plane and cannot interfere with the other parts. Figure 1 illustrates virtualization resource abstraction in SDN that allows for the division of a resource into several slices [25]. Similar to the actual resource interface, this abstraction creates a software layer that yields a virtually sliced interface. The virtualization layer is placed between controller plane and data plane to create slices. These slices are isolated in terms of the resource and traffic.

Basically, SDN offers the following three approaches to implementing NV: (i) proxy virtualization [26]: referring to a special type of OpenFlow controller that can create slices of network resources in which each slice is managed by a different controller, (ii) layer two prefixes-based virtualizations: integrated VLAN Tags, MPLS [27] or GMPLS into OpenFlow to achieve NV, and (iii) programming virtualization [28]: using a specific type of high-level network programming languages that facilitate traffic isolation.

Proxy virtualization mainly refers to a transparent proxy virtualization layer, which is located between the controllers and the switch, to create slices between virtual networks. FlowVisor [29], OpenVirtex [30] and CoVisor [31] are the special types of the controller placed between SDN controller and open enabled switch. These controllers have the capability to create slices and define a set of flows that can manage specific topology to ensure the operations of the guest controller. FlowVisor and OpenVirtex focus on slicing to address apace isolation, while CoVisor allows numbers of different controllers to manage and share traffic. For example, CoVisor can apply different network applications such as firewall, load balancer, gateway, router and traffic monitor in combination. Moreover, CoVisor abstracts

![FIGURE 1. OpenFlow NV model.](https://academic.oup.com/comjnl/article-abstract/60/10/1510/4321712)
concrete topologies, providing custom virtual topologies. Figure 2 shows a particular scenario of using hypervisor controller wherein a special controller is used for creating virtual networks, to provide user slices. In the top, each user has its own controller that allows to manage and control his own network. The physical layer illustrates the forwarding plane that can be customized with the user’s topology. The hypervisor layer provides multiple isolated logical networks by slicing the flow Table for each forwarding plane and identifies the flow entries belong to which user controller. Various hypervisor controllers are proposed with different capabilities [32]. For instance, OpenVirtex supports full network header space and provides many features such as address space, customize topology. These features allow two tenants to use the same IP subnets and TCP/UDP ports, in two different slices [33]. In addition, each virtual network can run its own network operating system (NOS) to program the virtual switches. FlowN is a container based on application virtualization for OpenFlow network. Each container includes the tenant VNs with its own controller. It provides virtual network topologies to the tenants by entire abstracts the physical. HyperFlex introduced as new OpenFlow proxy controller to tackle the problem of virtualization of the control plane, whereas the existing solution focused on resource isolation of coexisting virtual networks on the physical data-plane infrastructure. Table 1 shows three proxy controllers with their capabilities and features to create NVs.

Several studies introduced a virtualized architecture based on layer two prefixes to extend the control ability of VLAN tags, MPLS and GMPLS for achieving NV through an implemented control module in the OpenFlow controller (Table 2).

Sharafat et al. [39] introduced MPLS-TE and VPN services managed by an application layer of the NOX controller. NOX handles all MPLS features [40], whereas the OpenFlow switch manages push, swap and pop actions. NOX modifies flow Tables according to their respective switches subject to any changes required on the data plane.

Several high-level network languages, such as Frentic [41], Netcore and ProCera, have been developed based on comprehensive API. Due to the limitations for isolating and slicing of the virtualization layer that causes flow entry conflict, several studies have introduced network programming languages and modules to implement NV without involved a middle layer. It can facilitate a complete bandwidth guarantee, isolation of the topology, flows, and controls.

### 3. SDWN ARCHITECTURE

Recently, several studies [42, 43] have proposed to adopt the SDN network architecture features to be sued in the wireless network and come with new terms namely SDWN. As a result of heterogeneous devices, various data formats and diverse protocols used in the wireless network, different architectures are introduced for SDWNs with distinctive

**TABLE 1. Proxy controller and their features.**

<table>
<thead>
<tr>
<th>Proxy controller</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenVirtex [34]</td>
<td>Provide address space, customize topology, users can deploy their own NOS</td>
</tr>
<tr>
<td>FlowVisor [35]</td>
<td>Creates rich slices, manage the isolation between each virtual network</td>
</tr>
<tr>
<td>CoVisor [36]</td>
<td>different controllers to manage and share the same traffic</td>
</tr>
<tr>
<td>FlowN [37]</td>
<td>Fully tenant VNs with its own controller</td>
</tr>
<tr>
<td>HyperFlex [38]</td>
<td>Virtualization of the control plane</td>
</tr>
</tbody>
</table>

**TABLE 2. Seminaries and differences between SDN and SDWN.**

<table>
<thead>
<tr>
<th>SDWN</th>
<th>SDN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Similarity</strong></td>
<td><strong>Programmability</strong></td>
</tr>
<tr>
<td>– Layered framework with the separation of the control logic and underlying devices</td>
<td></td>
</tr>
<tr>
<td>– Centralized control logic</td>
<td></td>
</tr>
<tr>
<td>– NV</td>
<td></td>
</tr>
<tr>
<td><strong>Difference</strong></td>
<td><strong>Wireless networks</strong></td>
</tr>
<tr>
<td>– Base station in the infrastructure layer</td>
<td>Wired networks Router and switch in the infrastructure layer</td>
</tr>
<tr>
<td>– Topology discovery is based on geographic physical links condition</td>
<td>Topology discovery is based on physical links</td>
</tr>
<tr>
<td>– Protocol-based wireless virtualization</td>
<td>Flow-based wireless virtualization</td>
</tr>
<tr>
<td>– Resource-based virtualization</td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE 2. Hypervisor NVs layer.**
capabilities. In this section, we present an architecture of SDWN designed for the mobile network [44] with the aims of enabling standard interfaces of the control layer to manage traffic handling and facilities for the flexible application layer. Typically, the mobile network consists of diverse radio access networks (RANs) [45] communicated with transport core network. The communication between RAN and core network can be done through a wire or wireless media, which involves many technologies such as fiber optic, microwave with different network topologies.

The architecture of SDWN [46] includes RANs, which provides programmability capabilities with multiple functionality levels to enable incremental deployments. In this architecture, two models can be adopted to implement: (i) evolutionary model allows the legacy control plane to communicate with transport core network without changing the existing interfaces. In this case, SDN controller can provide standardized APIs to allow for communicating with legacy entities, in the physical and virtual environment. (ii) In the clean slate model, the application layer placed on top of SDN controller [47] with software API that gives advantages of the programmable network, resulting for, easy and fast develop new functions and services in the network. The architecture is also able to provide different northbound and southbound interfaces, as follows:

1. A northbound interface [48] to the virtual operators to share physical network resources and dynamically change the share of resources. A northbound interface for third parties and providers, to allow them to handle and manage the traffic through cartelized controller how its traffic is handled. A southbound interface to physical core transport backbone. This interface permits SDN controller to implement the different behavior policies according to the requests from parties.

2. A southbound interface [49] RAN to provide effective virtualization by sharing physical resources among different operators. A southbound interface to the mobile provides programmability capabilities on the mobile experience.

4. SDWN ARCHITECTURE REQUIREMENTS

SDWN NV must be designed to address specific requirements these requirements differs from the requirements that apply in SDN NVs. In this section, we will heighted some of these requirements.

4.1. Data aggregation

Data aggregation is a process to show how the packets can be sent and received by the node in the wireless network and it is mainly used in WSNs. Due to the redundant data in WSNs data aggregation is used to decrease the number of transmissions in the network. In SDWN NVs, the method of data aggregation should be considered between the network slices. In addition, how SDN controller can handle the redundant data in each slice. Data aggregation in SDWN NVs may affect other performance metrics such as delay, accuracy and security.

4.2. Support legacy protocol stacks

Legacy wireless networks protocols are going to excite for many more years before the SDWN revolution takes place. Legacy wireless protocols are an absolute requirement for SDW architecture. The current SDWN solutions are designed to use OpenFlow protocol, the first implementation of the SDN, which is used for switch-controller communication. In SDWN, various protocols are used based on the type of the wireless network.

4.3. Flexible flow rules definition

OpenFlow allows easily accessing and managing individual flows. This feature allows the controller to detect elephant flows, and route them separately for better throughput. Typically in SDN network OpenFlow switches usually buffer received packets but the default size of this buffer is only 256 packets. Therefore, wildcard rules are introduced in SDN network to overcome the above mentioned problem. Although wildcard rules approach is complex to be used in SDN, but in SDWN architecture it should be implemented because of a large number of the header filed that support in last OF specification 1.5 (44 different types are header field) (Figure 3).

5. SDWN NV

NV regardless of whether it is in wired or wireless can be viewed as a hard task. Recently, several studies have adapted the same tenet of SDN to achieve NV in wireless environment technologies. This section presents the state-of-the-art wireless network based on SDN NVs that discussed in Section 2. We focused on the approaches that used in the SDN such as hypervisor proxy virtualization. In Fig. 4 we illustrated the classification of SDWN NVs based on two approaches.

5.1. Type of the network

In this section, the classification of SDWN NWs is discussed. The classification focused on the type of the network and hypervisor controller that adopted from the SDN.
5.1.1. Wireless LANs

Wireless LANs is built based on distributed access points that connected to WLAN controllers. Therefore, the concept of centralized control of the network is not new in WLANs but separating data plane from the control plane is not implemented. This section discusses the proposed solutions to deploy NVs in WLANs.

5.1.2. CloudMAC

The CloudMAC [50] is SDN-based architecture design for 802.11 WLAN to enable access points (APs) forward MAC frames to servers for processing purpose [51]. The components of CloudMAC consists of virtual APs (VAPs), wireless termination points (WTPs), OpenFlow controller and tunnels to connect the entities. VAPs represent operating system (OS) of the network deploy on a virtualization host, such as Xen [52] or VSphere Centre [53]. Each VAP manages one or more virtual WLAN cards that appear to the user as a normal physical WLAN card. Therefore, by using the bound physical card, multiple virtual WLAN cards CloudMAC inherently supports NV. CloudMAC adopted OpenFlow technology that increases the flexibility to manage the virtual network and dynamically reconfigure the WTPs and VAPs and allow on-demand allocation of new WTPs to a VAP. The CloudMAC used OpenFlow technology to increase the flexibility and enhance the WLAN [54]. OpenFlow provides the interfaces between the WTPs and VAPs and can be dynamically reconfigured. CloudMAC hypervisor controllers manage and control frames and stores network slicing policies. In addition,
it is intercepted and overwriting control headers to exchange between the WTPs and the VAPs.

5.1.3. OpenRoads

The OpenRoads architecture [55] comprises of three main layers that include flow layer, slicing layer and controller layer. Flow layer uses OpenFlow protocol to manipulate data path flow Table and SNMP protocol for controlling and monitoring SSID of the slice and controller layers. The slicing layer of OpenRoads utilizes FlowVisor to provide a slice and maintain for isolation managing the encapsulation of flow entry into flow Tables. A controller layerbuilt on top of the NOX controller that has a complete view of the network and manage flow Table by inserting flow enters in the switch. By implementing slices OpenRoad facilitate to manage virtual networks

5.1.4. Odin

Odin [56] is an SDN framework for WLANs that built on a light virtual AP (LVAP) to greatly simplify client management. The components of Odin consist of one master, multiple agents and a set of programme applications. The master runs on top of the OpenFlow control and has the complete view of the network. LVAPs have virtual interfaces mapping physical radio interface to provide virtualization. Odin also creates a unique BSSID for each client to isolate client traffic and give the clients illusion of owning its own AP. Therefore, the LVAP abstraction preserves the interfaces between the client and the APs. The design can be easily integrated with Wi-Fi security protocols such as WPA2 enterprise.

5.2. Cellular (mobile network)

The ability of mobile networks [57] to provide and delivery NVs in an efficient way and ensure highest service experience at the same time are key points of mobile operators in the future. Therefore, several studies introduced NVs in mobile network. This section presents state-of-the-art NVs solutions for SDWN in a cellular network.

5.2.1. V-core

V-core is an SDN-based network architecture [58] that used in mobile core networks and used the properties of the SDN NV by adopting slice technology. V-core used MobileVisor, to provide a multi virtual network that uses slicing for the mobile network. This architecture can be used with different control platforms according to various mobile virtual network operators. By adopting SDN NV, the controller and the data plane of the current mobile network are divided from each other. As a result, the control element MME and the whole intelligence in SGW (SGW-C) and PGW (PGW-C) for the LTE packet core network. The key element of V-core architecture is MobileVisor in which mobile controller acts as a transparent proxy between the control plane and data plane cluster (behaves like FlowVisor in SDN architecture). Moreover, all messages, both from GWs to control platform and vice versa, are sent through MobileVisor. It uses an open API to communicate with both controllers in control platform and GWs. MobileVisor manages a slice as a set of traffic flow corresponding to a specific user. Therefore, users are able to apply two different services from the network in slices. On the other hand, using MobileVisor in the mobile network can bring several advantages for mobile operator such as (i) creating slices to share mobile infrastructure between different network operators, (ii) hiring a mobile infrastructure by mobile operator and creating the own slice to deploy the policies and new services without building the own mobile network infrastructure, (iii) managing Quality of Service (QoS) to bill and charge policies in a flexible and efficient manner by the mobile operators. Figure 5 shows MobileVisor controller that place between the guest controller and forward devices. The administrator configures the slices based on policies and rules in which check the packet and forward it to the mobile data plane.

5.2.2. OpenRAN

OpenRAN [59] is build based on SDN architecture to facilitate virtualization and programmability through centralized system control. It comprises three main components: cloud computing resource pool (CCRP), wireless spectrum resource pool (WSRP) and SDN controller. OpenRAN has four levels of virtualization: application level, cloud level, spectrum level and cooperation level. (i) Application level of virtualization handles flow spaces and divides each virtual space operation to manage and control strategies. In this case, virtual spaces correspond to several network operators or services. (ii) In the cloud level of virtualization, SDN controller creates

![FIGURE 5. V-core system architecture.](image-url)
vBBUs and vBSCs by the virtualization of the physical processors and allocating appropriate computing and storage resources. (iii) Spectrum level of virtualization refers to virtualization of the spectrum by RF virtualization technology, which enables several vRRUs with different wireless protocols to coexisting one shared pRRU. (iv) Cooperation levels of virtualization construct several virtual networks, including virtual nodes and virtual links.

5.3. SDWN virtualization controller

Wireless hypervisor controller is designed for realizing custom ability, manageability and programmability of virtual slices. Through wireless virtualization controller, the control plane is decoupled from data plane and SPs can customize the virtual resource within their own virtual slices.

5.3.1. Cisor controller

SDWN NV controller hypervisor act as an intermediate layer [15] between SDWN network infrastructures and virtual controllers. Thus, the virtual controls are connected via the hypervisors to their SDWN infrastructure. The hypervisor layer resides between virtual networks to provide strong isolation. Therefore, resources such as bandwidth, topology, traffic, device CPU and forwarding Tables [60] can be sliced. Several SDWN hypervisors are proposed to facilitate VN in a different network. In this section, we discuss the features of this hypervisor to provide SDWN NVs.

5.3.2. MobileVisor

MobileVisor [61] is build based on the FlowVisor concept to provide NV for the mobile operator. In addition to OpenFlow API, MobileVisor supports GTP to allow a communication between the virtual controller and GWs. It defines a slice as a set of traffic flow corresponding to a specific mobile operator. By implementing MobileVisor, the mobile operator can share underlying data plane (MDPC) so that the network operators would not need to deploy their own mobile infrastructure in the same geographical location. Moreover, MobileVisor facilitates to manage different technologies such as 3G or 4G to save CAPEX and OPEX.

In addition, the slice of the MobileVisor can be defined by bandwidth or other parameters related to QoS so the mobile operator can manage their billing and charge policies in a flexible and efficient manner. Based on the qualitative analysis that carried out in using different scenarios, MobileVisor demonstrates a saving in CAPEX and OPEX specially in Telco specific cost of infrastructure, maintenance and service provisioning.

5.3.3. CellVisor

It is hypervisor controller that implemented in LTE network to provide NVs. CellVisor hypervisor [62] extended the FlowVisor functionality by providing flexibly slices that can be used MPLS or VLAN tags. These slices are base stations and radio resources. So individual mobile operator can manage and control the radio resources for the various slices according to subscriber demands. In addition, ‘slicing of the semantic space’ was used to allow grouping subscribers whose packets belong to the same classification. However, LTE can provide the isolation for the different mobile operator by using traditional BGP/MPLS VPN technologies. But LTE does not allow the different mobile operator to share the infrastructure and enable a complete virtual LTE network. CellVisor can make it relatively easy to support NV by partitioning the ‘flow space’ of packet headers. Thus, CellVisor can virtualize the stations by slicing resources at the physical layer (physical channels), the link layer (scheduling) or network layer (traffic shaping).

6. CHALLENGES AND OPEN ISSUES

NV in SDN has received a great support in the wired and numbers of studies show the flexibility and reliability of it. Moreover, numbers of Testbed used it to isolate the traffic between experiments. However, using same concepts of SDN NV into SWDN brings some challenges, which are discussed throughout this section.

6.1. SDWN and heterogeneous networks

The diversity of underlying infrastructure of wireless networks, for example, e LTE, WiMAX and WLANs a crucial challenge for NV in SDWN. Each of the wireless network infrastructures requires specific features and provides the distinctive type of services, can be hardly interconnected to manage the virtual network. In SDWN, Cvisor controllers manage an operation between different controller platforms, and multiple controllers can cooperate and shared same traffic. Therefore, using same technology can be suitable for heterogeneous SDWN networks. The communication between heterogeneous networks can be implemented through hypervisor technology.

6.2. Multidimensional virtualization in virtual sdwn

The complexity to manage the networking SDWN, for example, mobility management, resource allocation, and resource scheduling are challenges to achieving NV. Therefore, relying on one hypervisor controller to implement all the functions is impracticable. Allocation the resources and scheduling become much more complicated in SDWN due to the variability of radio channels, frequency reuse, power control, interference, coverage, roaming. Mobility management is an important issue in wireless networks that ensures successful delivery of new communications to users and maintains ongoing communication with minimal disruptions, while users move freely and independently.
6.3. Open interfaces (API) of sdwn

NV in SDN supports and abstracts numerous tenants with different topologies and controller applications with open APIs which facilitate network as a service in a cloud environment. For instance, OpenDaylight controller can provide virtual tenet network (VTN) with full Rest API, and user can perform the operations like GET/PUT/POST/DELETE easily. Manage these operations in SDWN raises challenges due to a natural platform for the wireless network. Each tenant has the ability to manage its own address space and customize the topology with clear API.

6.4. SDWN NVS security

Security represents a vital role in SDWN and NV brought many security challenges. In the case of using the hypervisor controller, such as OpenVirtex, to virtualized network, some guest controllers are placed on the end user side. Different types of attack are prone such as a DoS by generating a huge number of flows to down and stop OpenVirtex or make it work improperly. Moreover, it is easy to launch spoofing attack because of the guest controller now the IP of OpenVirtex. The ability for the user to be modified and rewritten header field of the package can create a possibility of injecting attack into another slice.

6.5. Lack of the SDWN NV application modules

In SDWN architecture, the controller should provide flexibility to support various types of the applications such as mobility and topology discovery in the virtual network. However, such applications are widely used in SDN architecture and offered by third parties; for example, HP network application store provides numbers of the SDN application that can be integrated with HP VAN Controller or OpenDaylight controller. Thus, designing and implementing network application for VNs must be considered. For example, forwarding modules that used in SDN application do not support the node mobility that needs rapid client re-association in the virtual environment. In addition, numbers of the proposed SDWN application modules [63] have not been evaluated in detail due to the lack of the simulation and tools that can be used for SDWN NV.

6.6. The placement of the visor controller

Typically, in SDWN NVs using a proxy controller that manages the network slices, the placement of the visor controller represents one of the issues in SDWN NVs. The location of CellVisor hypervisor controller is playing significant roller in terms of the performance of the SDWN. The problem of network hypervisor placement problem (HPP) is raised in SDN NV and several solutions are proposed to address this issue. Adopting these solutions in SDWN NVs is more complicate, because of a specific characteristic of the wireless network such as mobility reach ability especially in wireless wide area network (WWAN). To find the most reliable controller placements for SDWN NV is NP hard.

7. CONCLUSION

The emergence of SDNs is imposing new way to deploy NVs. In this paper, we have illustrated the differences between conventional network architecture and SDN architecture in terms of the virtualization layer. In particular, we identified three approaches that SDN can bring to facilitate NVs. We also discussed the architecture of SDWN, which has two models evolutionary and cleans the slate with different northbound and southbound interfaces that enhance programmability, manage traffic and virtualization. In addition, ongoing research efforts around SDWN NV are classified on the basis of the target network. Finally, key challenges and open issues for deploying SDWN NVs are discussed.

FUNDING

This work was carried out as part of the Mobile Cloud Computing research project funded by the Malaysian Ministry of Higher Education as the University of Malaya High Impact Research Grant (UM.C/625/1/HIR/MOHE/ FCSIT/03). The authors also extend their sincere appreciations to the Deanship of Scientific Research at King Saud University for its funding this Prolific Research Group (PRG-1436-16).

REFERENCES


---

**SECTION B: COMPUTER AND COMMUNICATIONS NETWORKS AND SYSTEMS**

**THE COMPUTER JOURNAL, VOL. 60 NO. 10, 2017**