A Short Review on the Dynamic Slice Management in Software-Defined Network Virtualization

Mohamed Khalafalla Hassan
Faculty of Electrical Engineering, Universiti Technologi Malaysia, Malaysia | Faculty of Telecommunication Engineering, Future University, Sudan
memo1023@hotmail.com (corresponding author)

Sharifah Hafizah Sayed Ariffin
Faculty of Electrical Engineering, Universiti Technologi Malaysia, Malaysia
shafizah@utm.my

Sharifah Kamila Syed-Yusof
Faculty of Electrical Engineering, Universiti Technologi Malaysia, Malaysia
kamilah@fke.utm.my

Nurzal Effiyana Ghazali
Faculty of Electrical Engineering, Universiti Technologi Malaysia, Malaysia
nurzal@utm.my

Kobby Asare Obeng
Faculty of Electrical Engineering, Universiti Technologi Malaysia, Malaysia | Department of Computer Science, Koforidua Technical University, Ghana
obeng@graduate.utm.my

Received: 14 September 2023 | Revised: 30 September 2023 | Accepted: 2 October 2023

Licensed under a CC-BY 4.0 license | Copyright (c) by the authors | DOI: https://doi.org/10.48084/etasr.6394

ABSTRACT

Software-Defined Network (SDN) is a contemporary networking technology that offers enhanced network flexibility and streamlines network management processes. Virtual Software-Defined Networking (vSDN) enables the dynamic allocation and sharing of physical networking resources among several slices, each representing distinct service providers or services. Each tenant is granted autonomous control over their respective services or applications within the Virtual Network (VN). Network virtualization allows providers to deliver novel, advanced services while enhancing efficiency and dependability. Utilizing numerous virtual networks on a specific infrastructure presents difficulties in implementing effective resource allocation mechanisms to prevent congestion and resource scarcity while maintaining the Service Level Agreements (SLAs) in the vSDN. A limited body of research has focused on dynamic slice allocation in the vSDN domain. This article will briefly review dynamic resource management, focusing on slice resource dynamic allocation through SDN hypervisors. The survey outlined that very few studies have tackled the impact of dynamicity slice management in vSDN, and there are research gaps in implementing proactive and intelligent frameworks for slice management in vSDN.

Keywords-software-defined network; slice management; virtualization; hypervisors

I. INTRODUCTION

The networking field has witnessed a surge of interest in virtualization due to the significant achievements in this area within the computer sector [1-4]. Regarding Software-Defined Networks (SDNs), the process of network virtualization has resulted in the creation of several virtual networks, also known as slices. Each virtual network has its abstractions of the underlying network topology [5]. The topic of networking already encompasses concepts such as "slicing." For example, traditional networks' virtual local network (VLAN) and multiprotocol label switching (MPLS) are standard layer two technologies. Nevertheless, the current trend in network virtualization is to generate partitions for the underlying physical network, apart from the conventional layering methodology. Each slice can possess resource abstraction, including forwarding tables, Central Processing Unit (CPU) resources, and network bandwidth. Software-defined virtual network slicing offers similar advantages to conventional network slicing but with enhanced flexibility and agility. The ability to program with ease contributes to the flexibility and agility provided by SDN hypervisors. This allows fast service provisioning, dynamic resource allocation, and network automation [5].

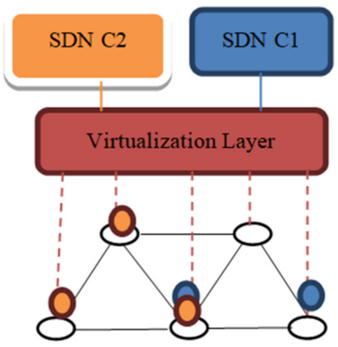

Fig. 1. Virtual software-defined network.

Virtual networks are organized hierarchically within a substrate network, using dashed lines to represent virtual resources (Figure 1). Implementing virtual software-defined networks involves the utilization of virtualization and SDN technologies to facilitate the development of Virtual Network Functions (VNFs) and virtual network services. This enables network administrators to dynamically create, delete, and manage different slices. The virtualization layer serves as a coordinator for the virtual networks of the service providers, namely SDN Controller 1 (C1) and SDN Controller 2 (C2). SDN C1 possesses control over a pair of network components, while SDN C2 exercises authority over a trio of network parts. Due to the importance of resource management in maintaining Quality of Service (QoS) and Service Level Agreement (SLA), this paper will examine the prominent hypervisors and frameworks that have handled resource management with more focus on dynamic slice management in vSDN.

II. REVIEW METHODOLOGY

This research employs the narrative literature review technique to examine the existing literature on slice management in vSDN to summarize and synthesize the available knowledge. The process of gathering and integrating existing literature is being undertaken to illustrate the research gap within the vSDN. The articles were chosen objectively to offer readers a thorough foundation for comprehending the current state of knowledge and to emphasize the importance of the research gap. The review commenced with a comprehensive literature search and subsequent screening process. After this, data extraction and analysis were performed, and the findings were used to compose the literature review.

III. DYNAMIC SLICE MANAGEMENT IN VSDN

FlowVisor (FV) [6] is widely recognized as the pioneering hypervisor for Software-Defined Networking (SDN) networks. The introduction of FV has facilitated the ability to distribute SDN networking resources among multiple controllers. According to [6], FV eventually functioned as an independent program on standard hardware (server). Its purpose was to guarantee that users or service providers had separate flow areas during service usage. When a controller attempts to handle a flow in a switch not within its designated flow areas, it modifies packet headers to indicate this action. Alternatively, the FV function produces an Open flow "OF error" notice. Many (vSDN) hypervisors are considered extensions to the FV hypervisor. Authors in [7] implemented the FV technique within mobile packet core networks and called the proposed framework MobileVisor. This framework encompassed various physical mobile networks, including 3G and 4G. It facilitated communication between diverse vendors' data flow and the Radio Access Network (RAN) [8]. On the other side, authors in [9] introduced the concept of a Network Virtualization Platform (NVP) designed to facilitate the abstraction of data center network resources. This platform is specifically tailored for cloud operation in multi-tenant scenarios. The NVP functioned as a controller for SDN service providers, enabling them to manage their SDN controllers using the Application Protocol Interface (API). This allowed users to exert control over their slices within the data center. NVP logical data pathways, also known as tunnels, establish connections between the source and destination Open Virtual Switches (OVSs). These logical paths are specifically associated with the respective user's slice. However, the EnterpriseVisor is a prominent hypervisor regarding resource distribution, particularly in bandwidth slice allocation and management. A novel software module was implemented to monitor and study the utilization of slices effectively. In this context, a mathematical model was constructed and solved using linear programming techniques to optimize the allocation of bandwidth slices inside the network. Authors in [10] presented AutoVFlow as a distributed hypervisor. They addressed the scenario in which the underlying infrastructure extends over distinct and non-overlapping domains in the context of a wide-area network. The hypervisor manages each part and functions as an intermediary that carries out slice and abstraction mapping. AutoVflow facilitates the delegation of administrative tasks from high- to lower-capacity controllers. Effective update policies are consistently upheld across the centralized and distributed controllers due to the ability of a single slice to encompass several domains. Various identities were employed, including virtual MAC addresses, which may vary across multiple domains. The observed latency was significantly elevated, with an average value of around 5.85 ms. The current control technique needs more automation for the dynamic load distribution. In [11], the ONVisor hypervisor was introduced as an SDN and Network Virtualization (NV) platform. This platform was designed to enhance flexibility by implementing distributed instances of hypervisors, enabling sharing a Virtual Network (VN) state. The system supports various SDN protocols, including Netconf and LISP. Nevertheless, the testing and evaluation of the system were limited to simple scenarios, and the discussion did not encompass dynamic resource allocation. They proposed a management framework to allocate bandwidth by establishing fixed thresholds for individual slices based on prioritization. The framework was introduced and demonstrated as a method for admission control. The proposed management system involves bandwidth distribution by establishing static thresholds for individual slices, which are determined based on slice prioritization. The framework was introduced as an admission control mechanism.

Authors in [12] proposed the DART framework which introduces a dynamic network bandwidth distribution method. The study was confined to the Industrial Internet of Things (IIoT) domain. The DART framework consists of two modules. The first is the communication module, which facilitates the transmission and reception of information. The second is the publishing module, which coordinates the controllers' activities. The centralized component provides recommendations regarding the optimal bandwidth allocation for every SDN controller. Authors in [13] introduced the Resource Manager (RM). The RM serves as a resource management system for virtualized SDN-based networks. It is designed to offer admission control functionality. The proposed methodology imposes limitations on the allocation of resources to virtual slices. The approach utilized dynamic thresholding to react to fluctuations in traffic volume effectively. The storage system effectively monitors and records the present level of bandwidth utilization. The compute module computes the available bandwidth resources before allocating the required amount. This computation is carried out by utilizing a flow rules manager and a threshold manager. These pairs allocate bandwidth depending on flow priority, with the ability to adjust the priority threshold based on predetermined flow priority. The methodology under consideration exhibits similarities to the strategy proposed in the abovementioned research. However, it focused on the present bandwidth use. The Libera hypervisor [14] was introduced to mitigate scalability challenges and enhance the capacity of service providers to offer their services. The scalability issue was primarily addressed by implementing Virtual Machine (VM) migrations and architectural adjustments that reduce slices and flow rules. This reduction in flow rules helps to minimize the bandwidth required between controllers and virtual switches. Libera is widely acknowledged as an extension of OpenVirtx. Nevertheless, the current implementation of resource scalability needs to consider future demands. Furthermore, a comprehensive analysis of the scalability of VM migration was not conducted. Authors in [15] introduced TeaVisor as a mechanism to ensure the provision of bandwidth isolation within vSDNs. The proposed hypervisor effectively mitigated the problem of overloaded links by employing a greedy heuristic method to distribute the traffic of these congested links among multiple channels with lower congestion levels. The findings indicated satisfactory performance. Nevertheless, the algorithm was developed using the most up-to-date traffic assessment techniques, potentially resulting in a depletion of resources, specifically bandwidth.

Authors in [16] proposed a resource management framework to enhance the load scalability of multi-domain SDN controllers. The utilization of a non-SDN hypervisor achieved this. The framework was developed to build a new VM for the controller or move the SDN controller based on load elevation. Software agents were employed to monitor the loads of the controller. In contrast to SDN-based hypervisors, using a stand-alone hypervisor may introduce challenges to the absence of resource segregation for SDN controllers. In addition, creating VMs and performing live migrations of VMs are associated with periods of service and slice unavailability. More recently, authors in [17] introduced the Meteor hypervisor as a proactive solution for predicting virtual SDN control traffic using LSTM. The study's results demonstrated a substantial improvement in control traffic latency, with a reduction of over 73%.

In our recent work [2], the DLVisor hypervisor was created using a dynamic learning framework integrating smooth-aided machine learning algorithms to predict future requests using techniques presented in [17-19]. The system exhibits reactivity and adaptability to substantial changes in traffic patterns by utilizing concept change detectors and significant tests. Previous research has indicated that oscillations in data traffic can have a negative impact on machine learning performance [20]. To address this issue, window-based approaches were utilized to reduce or eliminate these fluctuations. The enhanced dynamic learning framework is subsequently implemented within the resource management framework to improve resource utilization and mitigate the issue of overutilization arising from supply and demand estimates.

Table I presents an overview of the existing literature about resource and slice management in vSDN.

IV. DISCUSSION

Managing resources in vSDNs is typically carried out by network hypervisors or expanded frameworks connected to the network hypervisor. The most important parameters or terminologies in vSDN are hypervisors, flow spaces, open flow protocol, and virtual identifiers. According to the existing literature, most SDN hypervisors are built upon the FlowVisor hypervisor, widely recognized as the initial endeavor in SDN virtualization. The succeeding experiments aimed to address the flaws of FlowVisor, which encompassed the lack of admission control, dynamic resource management, interference in flow space, restriction of virtual topologies to subsets of the physical topology, and the absence of Quality of Service (QoS) features, as observed in the OpenVirteX. In contrast, specific hypervisors have endeavored to handle other facets, such as resource management in diverse controllers, as exemplified by the Compositional Hypervisor, and the facilitation of specialized network infrastructure, as demonstrated by the MobileVisor. On the other hand, hypervisors such as Virtualization Platform (NVP), AutoVflow, DFVisor, and CoVisor made attempts to tackle the operational and scalability challenges associated with vSDN resources in distributed or cluster systems.

TABLE I. SUMMARY TABLE

Hypervisor	**Summary**	**Drawbacks**	**Dynamic resource/Slice management**
FlowVisor [6]	1- General purpose first hypervisor 2-Stressed on network traffic isolation	1-Introduced OF message latency 2-Flow space interference 3-No admission control	No
MobileVisor [7]	1-Special network type 2-Introduced for mobile packet core	Same issues with FlowVisor	No
Network Virtualization Platform (NVP) [9]	1-General purpose 2-Focuses on the virtualization of the SDN switches 3-Applies to distributed controller scenarios only	1-Evaluation performed on tunnelling techniques only 2-OF message latency 3-Flow space interference	No. Uses clustering to scale up higher demands
OpenVirtex [22]	1-General purpose with minor latency added 2-Added resilience by VL mapping	OF message latency (minor)	No
Autoflow [10]	1-General purpose 2-Autoflow delegates configuration role to distributed administrators	Elevated OF message delays	No
Datapath Centric [23]	1-General purpose 2-Addressed the single point of failure in the FlowVisor 3-Support QoS	Added OF Latency 18%	No
AutoSlice [24]	1-General purpose 2-Improved the scalability of logically centralized hypervisor. 3-Differentiates between mice and elephant network flows	1-No performance evaluation on the simulation/ prototype has been published. 2-OF message latency	Yes. Enables SDN node and link migration. Reactive.
DFVisor [25]	1-General purpose 2-Added scalability 3-Introduced the concept of enhanced OpenFlow switches	Needs proper database synchronization between distributed controller databases	No
CoVisor [26]	1-General purpose 2-Policy-based 3-Designed to support heterogeneous controllers 4-More security	Added overheads. OF message latency.	No
Enterprise Visor [27]	1-General +special network type 2-Modify virtual topologies on demand 3-Admission control can be applied.	OF message latency.	Yes. Reactive.
ONVisor [11]	1-General purpose 2-Proposed scalable and flexible architecture	1-OF message latencies. 2-Lack of extensive evaluation.	No
Dynamic Resource Management Framework [28]	1-General purpose 2-Priority-based bandwidth resource allocation	1-Tested in a small scenario 1-No information was provided regarding latency overheads	Yes. Priority based. Reactive, not proactive.
DART Framework [12]	1-Designed for IIoT 2-Provided priority-based bandwidth resource allocation	1-Added an extra layer on top of the virtualization layer, which added more OF latency overhead 2-Limited scenarios	Yes. Priority based. Reactive, not proactive.
PrioSDN [13]	1-General purpose 2-Priority-based bandwidth resource allocation extension 3-Better results than DART	1-No OF latency provided 2-Some of the bandwidth resources are wasted 3-Requires higher computation	Yes. Priority based. Reactive, not proactive.
Libera [29]	1-General purpose 2-The scalability issue was addressed and enhanced (VM migration) 3-Integrated host VM migration and reconfiguration 4-Simplified and added more programmability feature enhancement	1-Added OF latency 2-VM Migration may contain downtime	Yes. Reactive.
TeaVisor [15]	1-General purpose 2-Mechanism for bandwidth guarantee 3-Algorithm to offload the overloaded links	1-More control overheads have been added	Yes. Reactive.
[16]	1-General purpose 2-Architecture for controller scalability	1-More control overheads have been added 2-VM creation and migration include downtime	Yes. Reactive.
Meteor [21]	1-General purpose 2-ML-based control plan prediction framework	1-Focused on control plan only	Yes (on the control plan).
DLVisor [2]	1-General purpose 2-Based on Enterprise visor and Libera 3-ML-based bandwidth slice allocation.	1-Exhibits same OF overheads with EnterpriseVisor	Yes. Proactive.

Nevertheless, few attempts have been made to address the challenge of dynamic slice and resource management. These include resource offloading/migration in Libera, AutoSlice, and FlowVirt. Furthermore, the concept of resource management through admission control has been introduced in the Dart and PrioSDN frameworks. These approaches offer a method of resource distribution based on static priorities. In addition, EnterpriseVisor provides a resource allocation framework that is highly promising. This framework seamlessly connects with the OpenVirtex hypervisor, enabling allocating resources, specifically bandwidth slices, through a mathematical model. The allocation process is resolved using linear programming techniques. In [15, 16], efforts were made to solve link and load scalability difficulties. However, it is worth noting that the proposed solutions in these studies were reactive and needed to examine resource allocation and future demand considerations adequately. The existing literature on the subject reveals that the available methodologies and frameworks have limitations in their adaptability to traffic and network changes. These approaches are either static, unable to adjust to such changes, or reactive, as they operate only based on current conditions without considering resource forecasting. Consequently, this might result in resource scarcity or depletion. Only [2] and [21] provided proactive bandwidth slice management using resource forecast and machine learning.

V. CONCLUSION

The current paper briefly presented an overview of the strategies and techniques employed in managing slice (bandwidth) resources within a virtual Software-Defined Networking (vSDN) environment. Based on the extant literature and relevant studies, a research gap has been identified concerning enhancing resource management in vSDN technologies. Based on our analysis and some of the discussed related studies, managing resources and allocating bandwidth resources, particularly in real-time scenarios, can result in resource scarcity due to excessive resource utilization. This is especially true considering that most existing resource management solutions should be regarded as future demands and sudden fluctuations in traffic patterns when calculating resource supply and demand. Hence, there is a requirement to implement proactive and intelligent frameworks for resource management. Therefore, it is imperative to have precise and resilient algorithms for estimating resources, particularly traffic. Consequently, a dynamic learning framework that integrates machine learning algorithms to acquire knowledge and predict forthcoming requests was devised. The system exhibits reactivity and adaptability to substantial alterations in traffic patterns by utilizing concept change detectors and considerable tests. Previous research has indicated that ML-based approaches can effectively mitigate the adverse effects of data traffic fluctuations and improve performance, such as in [2, 17].

REFERENCES

[1] M. Berman *et al.*, "GENI: A federated testbed for innovative network experiments," *Computer Networks*, vol. 61, pp. 5–23, Mar. 2014, https://doi.org/10.1016/j.bjp.2013.12.037.

[2] M. K. Hassan *et al.*, "DLVisor: Dynamic Learning Hypervisor for Software Defined Network," *IEEE Access*, vol. 11, pp. 84144–84167, 2023, https://doi.org/10.1109/ACCESS.2023.3302266.

[3] M. K. Hassan, A. Babiker, M. Baker, and M. Hamad, "SLA Management For Virtual Machine Live Migration Using Machine Learning with Modified Kernel and Statistical Approach," *Engineering, Technology & Applied Science Research*, vol. 8, no. 1, pp. 2459–2463, Feb. 2018, https://doi.org/10.48084/etasr.1692.

[4] M. H. H. Khairi, S. H. S. Ariffin, N. M. A. Latiff, A. S. Abdullah, and M. K. Hassan, "A Review of Anomaly Detection Techniques and Distributed Denial of Service (DDoS) on Software Defined Network (SDN)," *Engineering, Technology & Applied Science Research*, vol. 8, no. 2, pp. 2724–2730, Apr. 2018, https://doi.org/10.48084/etasr.1840.

[5] A. A. Blenk, "Towards Virtualization of Software-Defined Networks: Analysis, Modeling, and Optimization," Ph.D. dissertation, Technical University of Munich, Munich, Germany, 2018.

[6] R. Sherwood *et al.*, "FlowVisor: A Network Virtualization Layer," OpenFlow Switch Consortium, Technical Report 1: 132, Jan. 2009.

[7] N. Van Giang and Y. H. Kim, "Slicing the next mobile packet core network," in *11th International Symposium on Wireless Communications Systems*, Barcelona, Spain, Aug. 2014, pp. 901–904, https://doi.org/10.1109/ISWCS.2014.6933481.

[8] A. A. Abuelgasim, M. K. Hassan, M. H. Khairi, M. N. Marsono, and K. M. Yusof, "Real-time high-speed mobility management," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 24, no. 3, pp. 1534–1541, Dec. 2021, https://doi.org/10.11591/ijeecs.v24.i3.

[9] T. Koponen *et al.*, "Network virtualization in multi-tenant datacenters," in *11th USENIX Conference on Networked Systems Design and Implementation*, Seattle, WA, USA, Apr. 2014, pp. 203–216.

[10] H. Yamanaka, E. Kawai, and S. Shimojo, "AutoVFlow: Virtualization of large-scale wide-area OpenFlow networks," *Computer Communications*, vol. 102, pp. 28–46, Apr. 2017, https://doi.org/10.1016/j.comcom.2016.12.006.

[11] Y. Han *et al.*, "ONVisor: Towards a scalable and flexible SDN-based network virtualization platform on ONOS," *International Journal of Network Management*, vol. 28, no. 2, Mar. 2018, Art. no. e2012, https://doi.org/10.1002/nem.2012.

[12] V. Struhar, M. Ashjaei, M. Behnam, S. S. Craciunas, and A. V. Papadopoulos, "DART: Dynamic Bandwidth Distribution Framework for Virtualized Software Defined Networks," in *45th Annual Conference of the IEEE Industrial Electronics Society*, Lisbon, Portugal, Oct. 2019, vol. 1, pp. 2934–2939, https://doi.org/10.1109/IECON.2019.8927780.

[13] L. Leonardi, L. Lo Bello, and S. Agliano, "Priority-Based Bandwidth Management in Virtualized Software-Defined Networks," *Electronics*, vol. 9, no. 6, Jun. 2020, Art. no. 100, https://doi.org/10.3390/electronics9061009.

[14] W. Zhao, H. Yang, J. Li, L. Shang, L. Hu, and Q. Fu, "Network Traffic Prediction in Network Security Based on EMD and LSTM," in *9th International Conference on Computer Engineering and Networks*, Singapore, Asia, Jul. 2021, pp. 509–518, https://doi.org/10.1007/978-981-15-3753-0_50.

[15] G. Yang, Y. Yoo, M. Kang, H. Jin, and C. Yoo, "Bandwidth Isolation Guarantee for SDN Virtual Networks," in *IEEE Conference on Computer Communications*, Vancouver, BC, Canada, Dec. 2021, pp. 1–10, https://doi.org/10.1109/INFOCOM42981.2021.9488797.

[16] A. Ahmadian and M. Ahmadi, "DC-CAMP: Dynamic Controller Creation, Allocation and Management Protocol in SDN," *Wireless Personal Communications*, vol. 125, no. 1, pp. 531–558, Jul. 2022, https://doi.org/10.1007/s11277-022-09563-8.

[17] M. K. Hassan, S. H. S. Ariffin, S. K. S.- Yusof, N. E. Ghazali, and M. E. Kanona, "Analysis of hybrid non-linear autoregressive neural network and local smoothing technique for bandwidth slice forecast," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 19, no. 4, pp. 1078–1089, Aug. 2021, https://doi.org/10.12928/telkomnika.v19i4.17024.

[18] M. K. Hassan *et al.*, "Dynamic Learning Framework for Smooth-Aided Machine-Learning-Based Backbone Traffic Forecasts," *Sensors*, vol. 22, no. 9, Jan. 2022, Art. no. 3592, https://doi.org/10.3390/s22093592.

[19] M. K. Hassan, S. H. Sayed Ariffin, S. K. Syed-Yusof, N. E. Ghazali, M. E. Ahmed Kanona, and M. Rava, "Smoothing-aided long-short term memory neural network-based LTE network traffic forecasting," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 6, pp. 6859–6868, Dec. 2022, https://doi.org/10.11591/ijece.v12i6.pp6859-6868.

[20] M. E. A. Kanona, M. G. Hamza, A. G. Abdalla, and M. K. Hassan, "A Review of Ground Target Detection and Classification Techniques in Forward Scattering Radars," *Engineering, Technology & Applied Science Research*, vol. 8, no. 3, pp. 3018–3022, Jun. 2018, https://doi.org/10.48084/etasr.2026.

[21] Y. Yoo, G. Yang, C. Shin, J. Lee, and C. Yoo, "Control Channel Isolation in SDN Virtualization: A Machine Learning Approach," in *23rd International Symposium on Cluster, Cloud and Internet Computing*, Bangalore, India, Dec. 2023, pp. 273–285, https://doi.org/10.1109/CCGrid57682.2023.00034.

[22] A. Al-Shabibi, M. D. Leenheer, M. Gerola, A. Koshibe, W. Snow, and G. Parulkar, "OpenVirteX: A Network Hypervisor," presented at the Open Networking Summit 2014, 2014.

[23] R. Doriguzzi-Corin, E. Salvadori, M. Gerola, M. Sune, and H. Woesner, "A Datapath-Centric Virtualization Mechanism for OpenFlow Networks," in *Third European Workshop on Software Defined Networks*, Budapest, Hungary, Sep. 2014, pp. 19–24, https://doi.org/10.1109/EWSDN.2014.19.

[24] Z. Bozakov and P. Papadimitriou, "AutoSlice: automated and scalable slicing for software-defined networks," in *ACM conference on CoNEXT student workshop*, Nice, France, Dec. 2012, pp. 3–4, https://doi.org/10.1145/2413247.2413251.

[25] L. Liao, A. Shami, and V. C. M. Leung, "Distributed FlowVisor: a distributed FlowVisor platform for quality of service aware cloud network virtualisation," *IET Networks*, vol. 4, no. 5, pp. 270–277, 2015, https://doi.org/10.1049/iet-net.2014.0107.

[26] X. Jin, J. Gossels, J. Rexford, and D. Walker, "CoVisor: a compositional hypervisor for software-defined networks," in *12th USENIX Conference on Networked Systems Design and Implementation*, Oakland, CA, USA, Dec. 2015, pp. 87–101.

[27] J.-L. Chen, Y.-W. Ma, H.-Y. Kuo, C.-S. Yang, and W.-C. Hung, "Software-Defined Network Virtualization Platform for Enterprise Network Resource Management," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 179–186, Apr. 2016, https://doi.org/10.1109/TETC.2015.2478757.

[28] S. Agliano, M. Ashjaei, M. Behnam, and L. L. Bello, "Resource management and control in virtualized SDN networks," in *Real-Time and Embedded Systems and Technologies*, Tehran, Iran, Dec. 2018, pp. 47–53, https://doi.org/10.1109/RTEST.2018.8397078.

[29] G. Yang, B. Yu, H. Jin, and C. Yoo, "Libera for Programmable Network Virtualization," *IEEE Communications Magazine*, vol. 58, no. 4, pp. 38–44, Apr. 2020, https://doi.org/10.1109/MCOM.001.1900290.