Hindawi Publishing Corporation
Scientific Programming
Volume 2016, Article ID 9315672, 6 pages
http://dx.doi.org/10.1155/2016/9315672

Research Article

A User-Customized Virtual Network Platform for NaaS Cloud

Lei Xiao,[1] Yu Sheng,[1] Guanlan Tan,[1] Jianxin Wang,[1] and Yi Pan[2]

[1]*School of Information Science and Engineering, Central South University, Changsha 401083, China*
[2]*Department of Computer Science, Georgia State University, Atlanta, GA 30302-4110, USA*

Correspondence should be addressed to Yu Sheng; shengyu@csu.edu.cn

Received 25 February 2016; Accepted 18 May 2016

Academic Editor: Ligang He

Now all kinds of public cloud providers take computing and storage resources as the user's main demand, making it difficult for users to deploy complex network in the public cloud. This paper proposes a virtual cloud platform with network as the core demand of the user, which can provide the user with the capacity of free network architecture as well as all kinds of virtual resources. The network is isolated by port groups of the virtual distributed switch and the data forwarding and access control between different network segments are implemented by virtual machines loading a soft-routing system. This paper also studies the management interface of network architecture and the uniform way to connect the remote desktop of virtual resources on the web, hoping to provide some new ideas for the Network as a Service model.

1. Introduction

With the recent rapid development of cloud computing, more and more personal and enterprise users enjoy the various and convenient services brought by cloud computing. The demand based access model not only saves a lot of expenses for the user, but also avoids the waste of a large number of idle resources for the society as a whole [1]. Being necessary for the development of the cloud computing, virtualization technology abstracts physical resources into logical resources, makes the diversity and compatibility of equipment transparent to the upper structure, and realizes more granular utilization of resources, including calculation, memory, disk, network, and operating system.

Along with the advance of virtualization technology, cloud providers are able to meet more requirements of computing and storage capacity. Meanwhile, great efforts have also been made to do the research in this respect [2]. Data networks, considered a necessary component of the infrastructure that links the consumable resources together, have not been utilized sufficiently compared to other resources. While the computing and storage resources have their flexible consumption model [3] in most IaaS (Infrastructure as a Service), the network resource has not yet been turned into a consumable service from the user perspective.

It is necessary to start with the reference to the concept of network virtualization, which is the focus of the paper [4]. The network virtualization generally refers to the technology used to abstract the physical network resources. It makes the physical network resources pooled, which allocates resources to users according to their requests in a logical way, which can achieve the goal of flexible resources division or merger. The users can manage their allocated virtual networks independently without the concerns about the physical implementation detail and the network resource conflict with other users. The technologies commonly used in the implementation of network virtualization are VLAN (Virtual Local Area Network), VPN (Virtual Private Network), Overlay Network, programmable network, and the currently hot SDN (software-defined networking) [5] and DCN (data center network) [6, 7]. Unlike the above, our research objective is to connect the virtual network to the virtual machine inside the cloud.

The cloud computing service model has three basic services: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Similarly, in [8], the authors proposed a new concept on the network, Network as a Service (NaaS). According to the authors, NaaS should allow a user to request a network by specifying precisely

the network topology, the router computing capacities, the routing protocols, and possible other features.

Costa et al. [9] think that users now have little control over the network, so they suggest that NaaS should be network visibility and custom forwarding. As for the network visibility, users are provided with an abstract view of their allocated VMs without the need for reverse-engineering, but with easier deployment. As for custom forwarding, users can implement custom routing protocols, design firewall [10], and develop gateway, such as acceleration gateway [11].

In this paper, based on the above concepts, we present our design and implementation of VNetCloud, a platform that allows users to set up their own network infrastructure in a manageable way and makes the network a consumable service for users. VNetCloud is a virtual cloud platform with the network as the core requirement. The user platform will get an exclusive, isolated cloud environment. In VNetCloud, users can organize their computing, storage, and network resources, especially their network resources in their exclusive space. VNetCloud is not a local area network (LAN) but a collection of networks.

The contribution of our work can be summarized as follows:

(1) We designed and implemented a NaaS oriented platform prototype, which provides users with not only an isolated virtual resource environment, but also a flexible network topology, which enables a user to freely combine and dynamically dispatch his virtual resources according to the requirement.

(2) We have used the HTML5 based drawing technology to provide users with a more visual and convenient interaction environment, in which the users can drag and drop to add or manage virtual resources and topological relations between virtual resources, thus bringing about the implementation of high autonomy in operations management.

(3) We have studied and implemented a remote desktop method based on HTML5, which is used to push all kinds of the system desktop environments to a unified web interface through the protocol conversion middleware and therefore provides a strong support for the platform.

The rest of this paper is organized as follows. Section 2 summarizes the related work. Section 3 illustrates the architecture and implementation of VNetCloud. In Section 4, we describe application scenarios of our platform. We conclude the paper in Section 5.

2. Related Work

VPC (virtual private cloud), an on-demand configurable pool of shared computing resources allocated within a public cloud environment, was first put forward as a commercial solution by AWS [12]. It can divide for each user on public cloud an exclusive VLAN, and the user can add cloud services to this private network space, create a subnet, and establish connection with enterprise internal private cloud through VPN. Based on the idea of VPC, Wood et al. proposed the idea of CloudNet architecture [13], which joins VPNs with cloud computing, with VPNs providing secure communication channels and allowing customer's control over network provisioning and configuration. CloudNet can provide secure and seamless cloud resources to enterprises.

In [8], the authors proposed an extension of the virtual private cloud concept, that is, Cloud Networking. In this paper, the authors generalize the ideas in [13] and allow cloud and network resources to be handled as a single set. Network here refers to the connection between clouds, which are the nodes in the net. NCSS (Network-Aware Cloud System Suite) proposed by the authors has an integral control over clouds and the network between clouds, providing functionalities such as distributed network and cloud resource discovery, network and cloud mapping and creation, network and computing monitoring, and network and cloud resource management. The authors' research is based on the proposal that the whole routing topology information and physical routing equipment (such as PE and CE) information be obtained or set up on the same control platform.

CloudNaaS is a network service platform that allows users to utilize many network functions [14]. The authors present a design of an integrated provisioning system for cloud applications and network services with interfaces for customers to specify network requirements. CloudNaaS leverages the OpenNebula [15] cloud framework to implement the cloud controller component and utilize OpenFlow [16] enabled switches within lab-base setup. CloudNaaS primitives are implemented within the cloud infrastructure itself using high-speed programmable network elements.

In [17], the authors put forward some challenges and applicable scenes concerning Network as a Service and design and implement a cloud platform with NaaS through exploiting the functionality provided by software-defined networks, hoping that the cloud provider can integrate the cloud controller into an integrated network manager module to have a centralized view of the network, which implements the NaaS paradigm and is responsible for serving all the network related requests including queries.

In recent years, many researches are about how to enable the user to have control over different data centers or all kinds of transmission equipment between cloud networks or have implemented the flexible control of Network traffic based on software-defined network, making the network more intelligent. But it is difficult to make common users be able to fully control all kinds of transmission equipment now. To most of the requirements of the enterprises, all their resources are not distributed on different clouds. The implementation of their requirements for access control, security isolation, load balancing, and so forth does not necessarily rely on the control of transmission equipment like PE route, for soft-routing can be provided to users as a virtual machine in the public network. In this paper, what we are going to solve is aiming at the solutions to the free organization of the logical relations between their virtual resources, even the construction of complex network architecture, and the deployment of all kinds of security strategies for large users on public clouds.

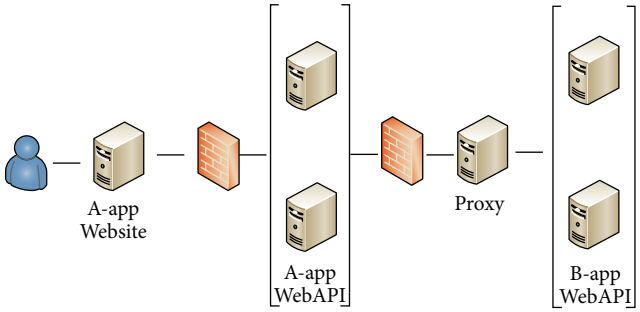

Figure 1: A network topology of a commercial system.

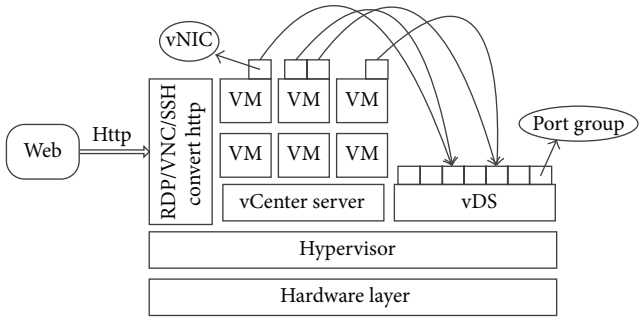

Figure 2: The infrastructure of VNetCloud.

3. The Architecture and Its Implementation

In our platform, the user can not only use the virtual machines resource in cloud, but also build their network topology for their business. As shown in Figure 1, it is a network topology of a commercial system.

Let us call this mobile application A. When it is in use, it first visits the website server on the outer network. The authentication and authorization servers, along with the WebAPI server of A-app, are set up in a security interception area surrounded by two layers of firewall. Passing through the security zone, mobile application A also needs to get access to the load balance module of B-app and the gateway before getting access to basic services, which is the production environment architecture of the product, and the test environment of the product must have same property. Here is just one product of the company. When the production environment is a complicated environment with a high cost, the test environment needs an equivalent investment.

As we know, test environment is different from the production environment, which is a transient operating environment, instead of being used in software iterative cycle. This company may require a test environment charged by demand and time. While the mainstream cloud service providers provide users with services focusing on computing and storage, to build such a complex test environment on a public cloud platform will be something very painful and difficult.

In this paper, we aim to provide users with a virtual cloud platform which is made by virtual machines and virtual network devices. With the platform, users can build a test environment for the commercial system.

The types of customers the public cloud platform faces are various, and their demands for network are varied. Considering that the existing cloud providers have quantitative billing on resources such as computing, memory, and hard disk, it is natural to list the network as one of the quantifiable resources.

3.1. The Platform Infrastructure. As a prototype, we choose VMware vSphere as our virtualization solution. The vSphere, as a mature commercial product, not only features the most advanced and complete server virtualization technology, but also offers the API and the SDK for these products and components. This platform customization management of vSphere can be implemented with the use of VMware vSphere Web Services SDK [18], which provides management interfaces for ESXi and vCenter and contains a Java sample code and very detailed API documentation. As shown in Figure 2, it is the architecture diagram for VNetCloud.

In addition, the VDS (vSphere distributed switch) [19] is deployed for the management of virtual network, and the collection of network segments is differentiated through the VDS port group. The distributed port groups group many ports under a common zone and provide locating point for a virtual machine connected to the marked network [20]. The virtual machine does not connect its virtual network interface card (vNIC) to a specific port on the vSwitch but to a port group. All the virtual machines connected to the same port group belong to the same network in the virtual environment. The port group can be used to break up the broadcast domains in the virtual environment. The vNIC of different port groups will not be broadcast by each other. As a result, the package transmission in the same port group is more secure.

Apparently, this virtual switch system is based on the Forwarding and Control Element Separation (ForCES) [21] framework, which can be divided into two logical parts, data plane and control plane. Data plane performs the actual operations such as packet switching, filtering, and marking. Control plane is a central controller of virtual network management, allowing administrators to manage network services through the higher-level functionality.

3.2. Web Interface. Our platform highlights that it is free for users to organize the isolation network relations between the virtual resources they have. Therefore we not only give the infrastructure support, but also make users feel intuitively that this network is a user-customized platform.

KineticJS is an open source JavaScript framework based on HTML5 [22, 23]. It encapsulates HTML5 Canvas [24], allowing easy and convenient operations of HTML5 Canvas. In KineticJS, users can draw, operate, and modify all graphics or pictures on the canvas and add event monitors on them.

We design an HTML5 based drawing panel as the main interface that allows users to deploy their customized network topology. We can drag and drop the preset virtual device icon from one division to the center panel through KineticJS. The user, through a drag-and-drop operation, can freely combine his resources and connect a virtual device

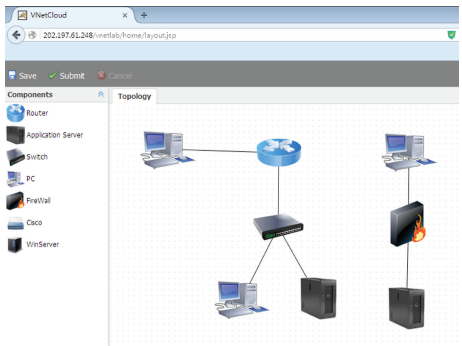

Figure 3: Web interface of the platform.

icon to another, which means that the two devices can be put in the same port group. By clicking on the linked line icon, the corresponding network interface relation on both ends of the device can be set. And by clicking on the device, the configuration of virtual resources such as CPU, memory, hard disks, and network interfaces can be viewed and set up. We also place the entrance to the remote desktop in them. Figure 3 shows the result of the implementation of the platform interface in a user-customized way.

The virtual devices on the canvas and their dragging, drawing, and connection are all based on KineticJS framework, implementing instances such as Kinetic.Topology, Kinetic.Topology.Background, Kinetic.Topology.Device, and Kinetic.Topology.Line. Kinetic.Topology is the main class of the topological graphs, containing the device list and the line list and the connectors between them and the get and set methods of these objects. When a device is dragged into the canvas, the set method of the device object is invoked, and the canvas will be updated. There are two layers in Kinetic.Topology, used to display, respectively, graphics painted and message data. Kinetic.Topology.Device is a device class, and it also has a line list, different from the line list in Kinetic.Topology, which is the relevant line to the device. In addition, it also contains removing, checking range, binding event, drawing, and so on. Such a visual usage scenario can strengthen the user's global mastery of the virtual environment.

3.3. Logic of the Customized Network Topology. After the topology design in the web interface is finished, the data of connecting relation can be sent to background. With toJSON method of Kinetic Topology, we can get the JSON data of our customized network topology, which will be a collection of Device and Line. Each subset in Device includes all kinds of configuration information of the virtual devices and their coordinates on the canvas. Each subset in Line contains the network interface information of virtual machines on both ends of all lines; those JSON data are submitted to the background, through the Java reflection mechanism and the deserialization of JSON, and a collection of Device and Line is aggregated to become the object of network topology.

In the customized network topology, normally we put the network interface of the virtual devices attached to the same line into the same port group. Since a device can have multiple virtual networks, the different network interfaces can be placed in different virtual networks, and network interfaces of virtual machines in the same port group can communicate with each other in the same subnet. Through VDS, we can connect the network interfaces, which are necessarily to be put in the same LAN, to the same port group at the same time.

As for network isolation and access control, we need a soft-routing system as a customizable gateway device, which can be loaded to a virtual machine. So we choose openWRT [25], to be the core device of our network partitioning that can completely meet the complex requirements of network architecture for users. By joining different network interfaces of openWRT to different port groups, the network data between the port groups can be forwarded by openWRT. Like using other common Linux systems, using openWRT allows more flexible and diversified connections of the configuration between different subnets for the user.

3.4. Cloud Desktop. VNetCloud, as a one-stop virtual cloud platform, is expected not only to implement the overall control over the network architecture of the resources the users have, but also to get easy access to virtual resources. A click of the device icon on web interface will implement the access to the virtual machines of different systems on the browser [26, 27].

Guacamole is an open source web application based on HTML5 [28]. Guacamole pushes various types of desktop environment by proxy to the unified web page through the protocol conversion middleware. It is fused with our platform well. The user can use VNetCloud in any place where he can get access to the Internet with a computer or mobile phone via a browser, without the need to install any client and plug-in.

Guaca mainly consists of three parts: Guacamole protocol, guacad, and web application. Guacamole protocol contains remote display rendering and event transmission. Guacd is a daemon process responsible for protocol conversion, the core of Guacamole, which dynamically loads various remote desktop protocols. Web application is presented at the front. Through the integration of Guaca on the platform, we simply click on the remote desktop interface of a device on the topology with the browser, and we can view the remote desktop in a new window or embedded in iframe and have access to it.

When a user has dozens of virtual devices, it may be difficult for him to locate the machine he wants to visit, so it costs time to check back. The combination of network topology and cloud desktop makes users' access target clearer.

4. Application

An important application of the NaaS cloud is the university network experiment. We first build an experimental environment for the computer network course in our university by the VNetCloud. Computer network, as one of the core courses of the Information Specialty, is abstract and difficult to understand. At many times, it requires not only theoretical teaching but also lab practicing. On one hand, the computer network lab currently is taught in laboratory and requires

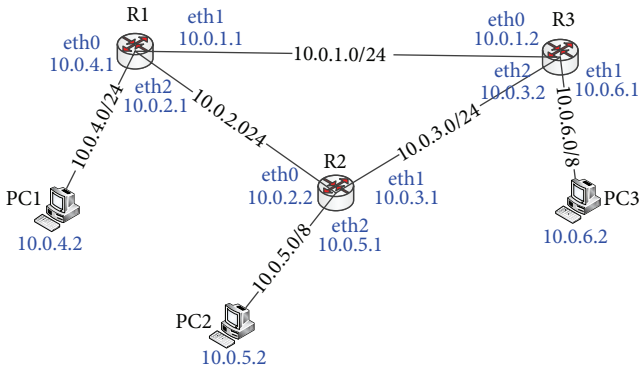

Figure 4: A topology of RIP experiment.

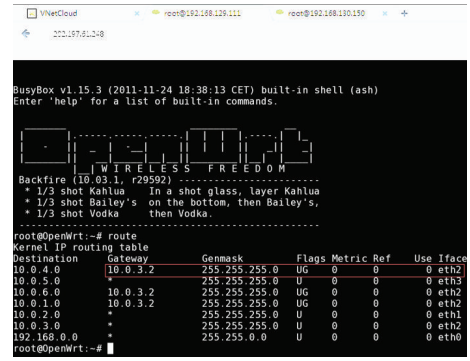

Figure 5: R2 routing table.

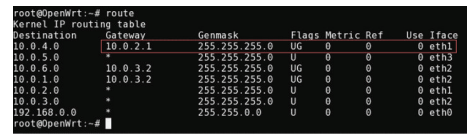

Figure 6: The change of R2 routing table after we start up the eth2 of R1.

a lot of resources. On the other hand, the simulation based experiments cannot show the real data package transmitting in the physical networks [29]. VNetCloud provides students with a customized virtual network environment for computer network experiments, where students can design a certain network topology, deploy a new routing protocol, and reconfigure the routing rules. Then VNetCloud generates the corresponding real network by connecting virtual machines. From the generated network, students can learn the flows of data packets and analyze the working process and performance of protocols by using packets capturing tool.

In this section, the Routing Information Protocol (RIP) experiment is used as an example to illustrate the usage of VNetCloud. In Figure 4, the network topology contains three routers and three PCs. The design of the network topology as shown in Figure 4 can be simplified, done by dragging the device icons from the web interface of VNetCloud and connecting them. Then, we can click the "Save" button. The background of VNetCloud will assign the demanded resources (three routers and three PCs) for the user. In this scenario, the resources will be released if the students exit. The user can specify a pair of network cards which are responsible for the connection between two devices, by clicking on the "Connection" icon. For example, in Figure 4, eth1 and eth3 are the pair of network cards responsible for the linking between PC1 and PC2.

When the network environment is ready, RIP experiment can be carried out. In the initial state, PC2 (10.0.5.2) is unable to access PC1 (10.0.4.2). By clicking the icon of router R2 and the access button of the pop-up device configuration window, we can remotely access R2 through cloud desktop. Through the access of R2, we can learn the routing table of R2. The routing table displays the network information that can only be directly accessed by R2, without knowing the network segments 10.0.1.0, 10.0.4.0, and 10.0.6.0. Thus, the students can install and configure Zebra and RIP for three routers. To learn the response of RIP to the network topology change, we can shut down eth2 of R1 and open the quagga service of three routers. Then, we can observe that the routing table of R2 will change soon. As shown in Figure 5, three pieces of new routing information are added in the routing table of R2 with "R" flag. The routing information with "R" flag denotes that it is produced by the RIP dynamic routing. Due to the shutdown of eth1, R2 cannot access R1 directly. It can access the network segment 10.0.4.0/24 through the route from R3.

In the next operation, we restart eth2 of R1. It is observed that the routing table of R2 changes accordingly. As we can learn from Figure 6, R2 can access R1 directly. It can access the network segment 10.0.4.0/24 through routing to R1.

Besides the computing network experiment, the platform can also be used to simulate a commercial system, even to deploy a complex commercial system.

5. Conclusion

In this paper, we design a user-customizable virtual network platform that enables users to simulate real network environment to build topology for project testing according to the production environment, to undertake penetration testing and network protocol testing or design a more secure cloud for one's own application. In VNetCloud, physical infrastructure is built by the system, and the user only needs to focus on the network architecture, which is very convenient for deploying a complex network architecture on the public cloud.

To simplify the platform interactivity, we provide users with a more intuitive and convenient operating environment in which many operations can be easily done by dragging and dropping the icons, so that the user network resources, along with their dynamic relationships and operations, can be clearly displayed. In addition, concerning the method used to connect to the virtual resources, an open source project Guacamole integrated into this platform. Thus the users can access the remote desktop of the corresponding device in the customized topology with a web browser.

In the future, we will also add such functionalities to the platform as user-customizable device icons, uploading image files, and deployment of OVF (Open Virtualization

Format) template, allowing the user to independently add virtual devices to the customized web space, making the user really feel that this is his unique, personalized cloud when enjoying public cloud services.

Competing Interests

The authors declare that there is no conflict of interests.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China under Grant nos. 61202494, 61402541, 61402542, and 61572530.

References

[1] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Proceedings of the Grid Computing Environments Workshop (GCE '08)*, pp. 1–10, November 2008.

[2] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, "CloudScale: elastic resource scaling for multi-tenant cloud systems," in *Proceedings of the 2nd ACM Symposium on Cloud Computing (SOCC '11)*, October 2011.

[3] S. S. Manvi and G. K. Shyam, "Resource management for Infrastructure as a Service (IaaS) in cloud computing: a survey," *Journal of Network & Computer Applications*, vol. 41, no. 1, pp. 424–440, 2014.

[4] M. F. Bari, R. Boutaba, R. Esteves et al., "Data center network virtualization: a survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 909–928, 2013.

[5] M. Banikazemi, D. Olshefski, A. Shaikh, J. Tracey, and G. Wang, "Meridian: an SDN platform for cloud network services," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 120–127, 2013.

[6] T. Zhang, J. Wang, J. Huang, Y. Huang, J. Chen, and Y. Pan, "Adaptive-acceleration data center TCP," *IEEE Transactions on Computers*, vol. 64, no. 6, pp. 1522–1533, 2015.

[7] T. Zhang, J. Wang, J. Huang, Y. Huang, J. Chen, and Y. Pan, "Adaptive marking threshold method for delay-sensitive TCP in data center network," *Journal of Network and Computer Applications*, vol. 61, pp. 222–234, 2016.

[8] J. C. Soares, M. Melo, R. Monteiro, and S. Sargento, "Building virtual private clouds with network-aware cloud," in *Proceedings of the 5th International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP '11)*, Lisbon, Portugal, November 2011.

[9] P. Costa, M. Migliavacca, P. Pietzuch, and A. L. Wolf, *NaaS: Network-as-a-Service in the Cloud*, USENIX, 2012.

[10] Y. Cheng, W. Wang, G. Min, and J. Wang, "A new approach to designing firewall based on multidimensional matrix," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 12, pp. 3075–3088, 2015.

[11] P. Dong, J. Wang, J. Huang, and H. Wang, "Split-TCP based acceleration gateway over packet lossy networks," *China Communications*, vol. 12, no. 5, Article ID 7112033, pp. 100–112, 2015.

[12] L. M. Surhone, M. T. Tennoe, and S. F. Henssonow, *Amazon Virtual Private Cloud*, Betascript Publishing, 2010.

[13] T. Wood, A. Gerber, K. K. Ramakrishnan, P. Shenoy, and J. V. D. Merwe, "The case for enterprise-ready virtual private clouds," in *Proceedings of the Conference on Hot Topics in Cloud Computing (HotCloud '09)*, article 4, USENIX Association, 2009.

[14] T. Benson, A. Akella, A. Shaikh, and S. Sahu, "CloudNaaS: a cloud networking platform for enterprise applications," in *Proceedings of the 2nd ACM Symposium on Cloud Computing (SOCC '11)*, article 8, Cascais, Portugal, October 2011.

[15] D. Milojičić, I. M. Llorente, and R. S. Montero, "OpenNebula: a cloud management tool," *IEEE Internet Computing*, vol. 15, no. 2, pp. 11–14, 2011.

[16] E. Watanabe, K. Ando, I. Karube, H. Matsuoka, and S. Suzuki, "Network innovation using openflow: a survey," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 16, no. 1, pp. 493–512, 2014.

[17] D. Kakadia and V. Varma, "Network virtualization platform for hybrid cloud," in *Proceedings of the 5th IEEE International Conference on Cloud Computing Technology and Science (CloudCom '13)*, vol. 2, pp. 69–74, December 2013.

[18] VMware vSphere Web Services SDK, https://www.vmware.com/support/developer/vc-sdk/.

[19] VMware vSphere Distributed Switch (VDS), http://www.vmware.com/products/vsphere/features/distributed-switch.

[20] S. Zhou, "Virtual networking," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 4, pp. 80–85, 2010.

[21] L. Yang, R. Dantu, T. Anderson, and R. Gopal, "Forwarding and Control Element Separation (ForCES) Framework," 2004, https://www.ietf.org/rfc/rfc3746.txt.

[22] Z. Ma, C.-Y. Yeh, H. He, and H. Chen, "A web based UML modeling tool with touch screens," in *Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering (ASE '14)*, pp. 835–838, Västerås, Sweden, September 2014.

[23] KineticJS, http://www.kineticjs.com/.

[24] M. Grady, "Functional programming using JavaScript and the HTML5 canvas element," *Journal of Computing Sciences in Colleges*, vol. 26, pp. 97–105, 2010.

[25] M. Chmielewski, P. Szyszkowski, and M. Piechowiak, "Application of IP multicast in embedded systems with openWRT," in *Proceedings of the 8th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP '12)*, pp. 1–5, Poznan, Poland, July 2012.

[26] L. Deboosere, B. Vankeirsbilck, P. Simoens, F. De Turck, B. Dhoedt, and P. Demeester, "Cloud-based desktop services for thin clients," *IEEE Internet Computing*, vol. 16, no. 6, pp. 60–67, 2012.

[27] I. Alimirza, A. K. Tripathy, V. Sarang, A. Joshi, and S. Shah, "ProtoCloud: a cloud based desktop," *Computer Science & Information Technology*, vol. 1, no. 1, pp. 57–64, 2013.

[28] D. Mulfari, A. Celesti, M. Villari, and A. Puliafito, "Providing assistive technology applications as a service through cloud computing," *Assistive Technology*, vol. 27, no. 1, pp. 44–51, 2015.

[29] N. Yalcin, Y. Altun, and U. Kose, "Educational material development model for teaching computer network and system management," *Computer Applications in Engineering Education*, vol. 23, no. 4, pp. 621–629, 2015.