Network Virtualization for QoS-Aware Resource Management in Cloud Data Centers: A Survey

Piotr Rygielski
Institute for Program Structures and Data Organization,
Karlsruhe Institute of Technology (KIT)
76131 Karlsruhe, Germany
Email: piotr.rygielski@kit.edu

Samuel Kounev
Institute for Program Structures and Data Organization,
Karlsruhe Institute of Technology (KIT)
76131 Karlsruhe, Germany
Email: kounev@kit.edu

***Abstract*— The increasing popularity of Cloud Computing is leading to the emergence of large virtualized data centers hosting increasingly complex and dynamic IT systems and services. Over the past decade, the efficient sharing of computational resources through virtualization has been subject to intensive research, while network management in cloud data centers has received less attention. A variety of network-intensive applications require QoS (Quality-of-Service) provisioning, performance isolation and support for flexible and efficient migration of virtual machines. In this paper, we survey existing network virtualization approaches and evaluate the extent to which they can be used as a basis for realizing the mentioned requirements in a cloud data center. More specifically, we identify generic network virtualization techniques, characterize them according to their features related to QoS management and performance isolation, and show how they can be composed together and used as building blocks for complex network virtualization solutions. We then present an overview of selected representative cloud platforms and show how they leverage the generic techniques as a basis for network resource management. Finally, we outline open issues and research challenges in the area of performance modeling and proactive resource management of virtualized data center infrastructures.**

I. Introduction

Despite the hype around cloud computing, it is well established that if this new computing model ends up being widely adopted, it will transform a large part of the IT industry [1], [2]. However, the inability of today's cloud technologies to provide dependable and predictable services is a major showstopper for the widespread adoption of the cloud paradigm, especially for mission-critical applications [3], [4]. According to [1], [3], the concerns of organizations about service availability is the number one obstacle to the adoption of cloud computing. Service overload, hardware failures, and software errors are among the most common causes of service unavailability as experience with Google' and Amazon's cloud services shows [3], [5], [6].

Cloud computing is commonly considered at three different levels: Infrastructure-, Platform-, and Software-as-a-Service. In each case, the amount of provisioned resources (computing, storage and networking) is managed by data center operators according to internal policies and Service-Level Agreements (SLAs) established with the users. The use of virtualization techniques allows flexible assignment of resources to virtual machines (VMs) enabling elastic, on-demand resource provisioning. Furthermore, virtualization allows to consolidate multiple applications on a smaller number of physical servers promising significant cost savings resulting from higher energy efficiency and lower system management costs. However, virtualization comes at the cost of increased system complexity and dynamicity due to the introduction of an additional level of indirection in resource allocations and the resulting complex interactions between the applications and workloads sharing the physical infrastructure. Moreover, the consolidation of workloads translates into higher utilization of physical resources making applications more vulnerable to threats resulting from unforeseen load fluctuations or network attacks.

While sharing computational resources and main memory works relatively well in cloud computing, sharing of network resources is more problematic [1]. There are established mature solutions for system virtualization enabling efficient and fair sharing of computational and storage resources like, e.g., the virtualization platforms based on Xen [7] or VMware [8]. However, currently no such widely adopted standard approach exists for network virtualization and QoS-aware bandwidth management in cloud data centers. If workloads are not reliably isolated at the network level, an unexpected load fluctuation experienced by one customer service (e.g., caused by a load spike or a DoS attack) might easily spread to services of other customers leading to SLA violations. Thus, the ability to manage the bandwidth of networks in data centers, both in terms of flexible resource allocation and performance isolation, is equally important to the ability to control the consumption of computing resources. Furthermore, due to the consolidation of workloads, cloud platforms must be able to deal with increased amount of network traffic at the physical machine level.

In this paper, we present a survey of existing network virtualization approaches that can be used as a basis for enforcing QoS policies and performance isolation in cloud data centers. The survey is motivated by our ongoing effort on developing a generic modeling approach for virtualized data center networks providing abstractions to model the performance-influencing factors of the network infrastructure, as well as its degrees-of-freedom and run-time reconfiguration strategies. This effort itself is part of our broader work in

the context of the Descartes Meta-Model (DMM) [9], an architecture-level modeling language for dynamic IT systems and services intended to serve as a basis for proactive QoS-aware resource management during operation [10], [11].

The modeling of the network infrastructure in a virtualized data center is a challenging task because of the wide variety of different network architectures and virtualization techniques with different features and characteristics. Our ultimate goal is to provide a generic and flexible set of modeling abstractions that can be applied in different scenarios without being limited to a particular network architecture. To this end, we survey existing network virtualization approaches that are used as a basis for building network infrastructures in modern data centers. Due to the large amount of specific approaches, we focus on generic virtualization techniques that are typically used as building blocks for implementing concrete network virtualization solutions. We categorize the various techniques and discuss their common aspects and different characteristics with special attention on their features related to QoS management and performance isolation. We then present an overview of selected representative cloud platforms and show how they leverage the presented generic approaches as a basis for network virtualization. Finally, we outline open issues and research challenges in the area of network virtualization, performance modeling, and proactive QoS-aware resource management in the context of virtualized data center networks.

In summary, the contributions of this paper are: i) A survey of existing generic network virtualization techniques typically used as a basis for building virtualized network infrastructures in modern cloud data centers, ii) A categorization and comparison of the presented techniques focusing on their features related to QoS management and performance isolation, iii) An overview of several representative cloud platforms showing how they employ the generic techniques as a basis for building a concrete network virtualization approach, iv) A discussion of open issues and research challenges in the area of data center network virtualization and proactive QoS-aware resource management.

The rest of this paper is organized as follows. In Section II, we review generic network virtualization approaches and evaluate the features they provide. In Section III, we present how the generic approaches are used in current software cloud platforms, hypervisors, and network virtualization architectures. We discuss open research issues and challenges in Section IV. Finally, we conclude the paper in Section V.

II. Generic Approaches to Network Virtualization in Data Centers

We start by reviewing generic virtualization techniques typically used as building blocks for implementing concrete network virtualization solutions. We characterize and categorize these techniques providing examples of their application and use in practical scenarios.

A. Link Virtualization

Under the term 'link virtualization' we understand a way to transfer multiple separate traffic flows over a shared link (physical or emulated), in such a way that each traffic flow appears to be using a dedicated link referred to as virtual link.

The IETF developed the Integrated Services approach (IntServ) [12] to provide guaranteed bandwidth to individual flows. The guarantee is provided by a reservation of resources over an entire communication path using the Resource Reservation Protocol (RSVP). In modern packet-switched networks, traffic classification combined with packet scheduling algorithms is used to differentiate QoS levels. Based on the Differentiated Services (DiffServ) approach [13], probabilistic QoS guarantees are provided by classifying enqueued packets and dequeuing them according to predefined policies or advanced scheduling algorithms [14], [15]. In addition, a traffic profiling technique allows to limit the packet sending rate. In contrast to IntServ, the deployment of traffic profiling at the end hosts allows to avoid bandwidth reservation in switches if the cross traffic is not exceeding the capacity of a given path. Admission control techniques allow to drop incoming flows if the admission would exceed capacity or cause QoS degradation. Finally, load balancing mechanisms utilize multiple paths leading to a given destination by spreading the traffic over separate routes and providing an illusion of a single link with increased capacity.

B. Virtual Network Appliances

A 'virtual network appliance' is any networking device that does not exist in a pure physical form but acts like an analogous physical equivalent. We distinguish two types of virtual network appliances: i) device aggregation where multiple networking devices act as a single logical entity, and ii) device emulation where an equivalent of a physical device is emulated by software. Emulation can apply to selected fragments or to a whole device.

1) Device Aggregation: VMware offers the vNetwork Distributed Switch that combines all hypervisors' virtual switches into one logical centrally managed unit [16]. Another example of device aggregation is the Juniper Virtual Chassis technology [17]. This feature allows up to ten switches to be interconnected and managed as a single virtual switch. Similar approaches are usually applied to provide a single point of management over the devices. However, the authors of [18] observe difficulties with determining the source of failures and errors (virtualization software vs. physical port) while using Virtual Chassis. Other examples of device aggregation are Open vSwitch [19] and OpenFlow [20]. In fact, these approaches are software implementations of physical devices, however, they provide a unified API to manage multiple emulated devices. Open vSwitch is an alternative to the default network bridge used in hypervisors nowadays. It can operate within a hypervisor or as a separate emulated node. OpenFlow assumes a software implementation of the control plane while the data plane is still realized in hardware. The implementation of the control plane as software provides more flexibility to

define the behavior of a device, e.g., by implementing new protocols.

2) Device Emulation: Multiple VMs running on a single physical machine are normally communicating using a software switch (or a bridge) provided by a hypervisor. In this case, the functionality of a networking device is emulated by the virtualization software [8]. Moreover, a physical server can be turned into a networking device using software emulation. There are several software solutions that provide such functionality, e.g., Quagga [21] or Open vSwitch [19].

Emulation of a physical device usually introduces additional performance overhead [22]. The author of [23] shows that the degradation of network performance caused by virtualization overheads can go up to 29% and 55% for the outgoing and incoming bandwidth, respectively. On the other hand, according to [24], it is possible to optimize an emulated switch to achieve the bandwidth of 5Gbps between co-hosted VMs compared to about 500Mbps using default solutions.

The performance degradation can be reduced when an appropriate hardware equipment is used. For example, in case of OpenFlow, there exist compatible devices that support the OpenFlow protocol so that forwarding is realized in a dedicated hardware while only a control plane is emulated. Such deployments are successfully used in production environments, e.g., by Google or as part of the Internet2 project [25].

C. Overlaying

An 'overlay network' is a network resulting from a modification or expansion of a layer belonging to the ISO/OSI stack; in short, it is a method for building a network on top of another network [26]. The major advantage of overlay networks is their separation from the underlying infrastructure. Overlaying in networks consists mainly of *adding a new layer* to the existing stack of protocols by defining tunnels, or *modifying a layer*, e.g., by introducing a new addressing scheme. Some virtualization techniques may use both approaches (adding and modifying a layer) simultaneously.

1) Adding a Layer (Tunneling): Tunneling consists of using one layer in order to transport data units of another layer — data units of one protocol are encapsulated in the data units of another protocol. The result of the tunneling is adding an additional layer to the default networking stack. For example, tunneling of layer 2 (L2) frames over L3 IP protocol creates a new layer between the L3 and L4 layers.

One of the most popular techniques that use tunneling in practice is VPN (Virtual Private Network). VPNs carry private traffic over a public network using encrypted tunnels. VPNs focus on security issues and do not provide QoS or performance isolation. Generic Routing Encapsulation (GRE) is a tunneling protocol that can encapsulate a protocol in an L3 protocol, e.g., IPv6 over IPv4. Another example of an overlay network is Multi-Protocol Label Switching (MPLS) [27]. It operates between L2 and L3 of the ISO/OSI stack and is often referred to as layer 2.5 protocol. MPLS provides IP packet switching based on a short label instead of a long IP address. The subsequent classification and forwarding are based only on the label, accelerating the forwarding process. MPLS assures QoS similarly to the DiffServ approach. Due to mature traffic engineering functions of MPLS, it is mainly deployed by the Internet service providers as a replacement of ATM or Frame Relay protocols. However, deployment of MPLS in a data center requires compatible hardware.

The issues of VM mobility and resource isolation have been addressed in the VIOLIN virtual network architecture [28]. In the latter, the emulated network is realized by tunneling over the UDP. The AGAVE project [29] aims at providing end-to-end QoS-aware service provisioning over IP networks using IP-in-IP tunneling. Another project called Virtuoso [30] provides an L2 overlay network supporting VM migration while maintaining active connections open after the migration. Virtuoso provides L2 connectivity using tunneling over TCP/SSL connections. The Reservoir project [26] defines Virtual Application Networks (VANs) to provide connectivity between distributed data centers. VANs are realized using L2-in-L3 tunneling and emulated switches based on the KVM hypervisor.

Overlay networks are often used to address limitations of the Internet or to provide new functionalities like, e.g., provide local connectivity in distributed computing environments [31]. The VIOLIN, AGAVE, Virtuoso and Reservoir projects are targeted mainly at Grid computing infrastructures or data center federations and thus their scope is not limited to a single data center.

2) Modifying a Layer: Modifying a specific layer consists mainly of providing a new protocol for that layer or changing the behavior of an existing one. The main goal of such a modification is to mend certain drawbacks of an existing technique or to provide a new functionality.

The VLAN technique [32] provides logical isolation between broadcast domains in L2 by creating virtual subnets on top of a single physical subnet. It is a modification of the Ethernet consisting of adding additional fields to the Ethernet frame headers. In VLANs (802.1Q), the new addressing scheme consists of expanding traditional Ethernet frames by a VLAN ID (VID) which allows to create 4096 VLANs. The main limitation of the VLAN technique, namely a limit of 4096 VLANs in a network, is about to be eliminated by VXLAN which assumes 24-bit VLAN network identifier [33]. Unfortunately, VLANs do not provide performance isolation and their QoS capabilities are limited to traffic prioritization (802.1p). Moreover, the Spanning Tree Protocol typically used in VLANs cannot utilize the high network capacity of modern data center network architectures like, e.g., fat-tree [34], DCell [35], BCube [36]. Another example of a layer modification is the networking infrastructure used at Facebook where the default L4 TCP has been replaced with a custom UDP transport layer to obtain lower latencies in their data centers [37]. The main drawback of network virtualization based on overlaying is the performance overhead caused by processing packets in the additional layer or the lack of hardware support for modifications of layers.

D. Summary

We distinguish three categories of generic network virtualization techniques. Each category contains generic building blocks that enable the implementation of specific features as part of a virtualization solution based on these techniques. For the sake of completeness, we include the *"Other Uncommon"* category to cover uncommon, highly specialized approaches as well. The categorization is depicted in Figure 1.

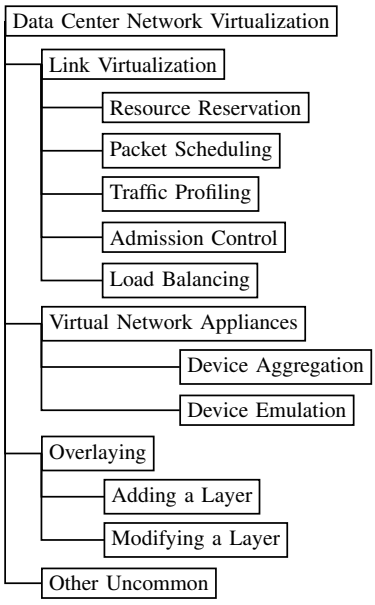

Fig. 1. Categorization of generic data center network virtualization techniques.

We now evaluate the support of selected features that are usually enabled by the considered generic virtualization techniques. The evaluation is presented in Table I.

Link virtualization techniques provide mainly QoS capabilities and performance isolation. Approaches based on IntServ (i.e., resource reservation, admission control) provide QoS with strict bandwidth guarantees, however, they are not scalable, while those based on DiffServ (i.e., packet scheduling, traffic profiling, admission control) provide only probabilistic QoS guarantees, however, in a scalable manner. Packet scheduling algorithms are used for traffic prioritization while traffic profiling techniques are used for limiting the bandwidth of a virtual link. Admission control techniques can drop excessive traffic when a link is overloaded; additionally, they can deny communication between specific tenants by dropping foreign flows. Finally, load balancing mechanisms distribute traffic flows over multiple paths to balance the bendwidth utilization and hereby support other QoS provisioning mechanisms.

Virtual Network Appliances usually define a centralized point of management of the infrastructure by defining a common management interface for multiple devices. Providing a single point of management is the main incentive for aggregation of devices into a single logical unit. Device emulation techniques are mainly used as a substitute for unavailable hardware equivalent or to enrich default capabilities of the equivalent. Additionally, software used for emulation can enable programmability capabilities (e.g., OpenFlow) or provide a common management interface across multiple emulated devices (e.g., Open vSwitch).

A wide variety of capabilities are enabled by *Overlaying* which depend on the particular implementation of a given technique. Logical isolation is usually the incentive for adding a new layer to the traditional networking stack[1]. Modifying the default capabilities of standardized protocols — e.g., by introducing custom addressing schemes or customizing headers of data units — can yield a wide variety of features depending on the particular implementation, however, an overlay usually requires the presence of additional mechanisms in order to provide QoS capabilities. Moreover, due to the illusion of local connectivity enabled, e.g., by the tunneling of protocols, VMs can be flexibly migrated between physical machines without being bound to a single subnet or VLAN.

Finally, with the *Other Uncommon* category, we account for highly specialized approaches that can provide any specific functionality based on a given underlying hardware infrastructure. An example of an approach that is based on the features of a specific hardware is NetShare [38]. Its functions rely on the hardware implementation of the Deficit Round Robin algorithm which is a specific feature provided by Fulcrum switches. Moreover, some approaches require specific configuration of the physical topology, e.g., VL2 [39] works on Clos topologies while Portland [40] works only with fat-tree multi-rooted topologies.

III. Leveraging the Generic Virtualization Approaches in Cloud Data Centers

In this section, we describe how the presented generic virtualization approaches are used as building blocks for complex network virtualization solutions in current software cloud platforms, hypervisors, and network virtualization architectures. Due to space constraints, we review only selected representative software cloud platforms. Additionally, we outline the features enabled by solutions designed for building Software Defined Networks (SDN). In the end, we briefly characterize generic virtualization approaches used in current architectures for network virtualization.

A. Network Virtualization in IaaS Software Cloud Platforms

IaaS software cloud platforms are complete software solutions for building an IaaS cloud infrastructure. They support the management of a physical infrastructure by providing tools for management and virtualization of resources (computation, storage and networking). Most software Cloud platforms implement APIs for controlling the virtualized resources providing a centralized management interface.

[1]For the needs of this paper, we assume that the default networking stack consists of Ethernet, IP, and TCP/UDP protocols.

TABLE I
PROPERTIES OF GENERIC NETWORK VIRTUALIZATION TECHNIQUES.
(+ YES, − NO, ⋆ DEPENDS ON IMPLEMENTATION)

Generic Approach	Provided property					
	QoS Provisioning	Performance Isolation	Logical Isolation	Boundless VM migration	Scalability	Centralized Management
Link Virtualization						
IntServ	+	+	−	−	−	−
DiffServ	+	⋆	−	−	+	−
Resource Reservation	+	+	−	−	−	−
Packet Scheduling	+	−	−	−	+	−
Traffic Profiling	+	⋆	−	−	⋆	−
Admission Control	⋆	⋆	⋆	−	−	−
Load Balancing	⋆	⋆	−	−	−	−
Virtual Network Appliance						
Devices Aggregation	−	−	−	−	−	+
Devices Emulation	−	−	−	−	−	⋆
Overlaying						
Adding a Layer	−	−	+	⋆	⋆	⋆
Modifying a Layer	−	−	⋆	⋆	⋆	⋆
Other Uncommon	⋆	⋆	⋆	⋆	⋆	⋆

A module of OpenStack called Quantum [41] manages the virtual networks in OpenStack-based software Cloud platforms. Although Quantum is not a mature product yet, it provides an API to manage the connectivity between the interfaces of devices managed by other OpenStack services in a data center. The basic offerings of Quantum provide limited functionalities, however, Quantum supports extensions in the form of plugins which enable additional features. Quantum with proper plugins allows among other things to provide end-to-end QoS guarantees as well as support for Open vSwitch or OpenFlow. In addition, new plugins can be developed to extend the set of supported capabilities.

OpenNebula [42] supports creating virtual networks based on VLANs. It defines fixed and ranged virtual networks. The former connects two predefined VMs using MAC-IP pairs, while the latter provides connectivity to a group of VMs selected by wildcarding of IP addresses. Based on the VLAN technique, OpenNebula provides only logical isolation in its network infrastructures.

CloudStack [43] supports VLAN based logical isolation between tenants in L2 and L3. CloudStack allows the MPLS technique to be used over a designated VLAN. In addition, CloudStack supports hardware load balancers and implements a virtual router.

In the following subsection, we describe selected virtual network appliances that cooperate with software cloud platforms.

B. Virtual Network Appliances in Software Cloud Platforms

Software Defined Networking (SDN) is an architecture which aims to separate the control plane from the data plane in network switches and routers. In SDN, the control plane is implemented in software and is located on separate servers while the data plane is realized in networking hardware. OpenFlow [20] is an example of an SDN architecture.

OpenFlow is an implementation of a switching fabric where the control plane and data plane are clearly separated. OpenFlow can be deployed on a server (emulation) as well as in firmware of a compatible networking switch. The software realization of a control plane allows to make decisions about forwarding of flows based on L2–L4 data headers, VLAN headers, and type-of-service (ToS) fields. Moreover, the control plane controller provides an API interface that can be utilized by software Cloud platforms to centralize the management. OpenFlow is supported by OVS such that every switch located within a hypervisor can be managed in the same way as other OpenFlow switches. An OpenFlow switch can classify a flow and assign it to a dedicated queue for processing. This allows using scheduling algorithms along with traffic profiling to provide QoS based on the DiffServ approach. Moreover, the flow matching mechanism can drop a given flow as an action to a matching.

FlowVisor [44] is a special purpose OpenFlow controller that acts as a transparent proxy between OpenFlow switches and multiple OpenFlow controllers. It allows to slice a network into parts that are managed by separate OpenFlow controllers while enforcing performance isolation between slices. As OpenFlow drawbacks, we account mainly the possibility of performance degradation due to the separated control plane controller. In addition, the size of a custom forwarding table is constrained by the amount of available memory in the data plane's hardware.

A default networking software bridge located in a hypervisor can be controlled by OpenFlow as well. The ability to use OpenFlow in a hypervisor is enabled by Open vSwitch (OVS) [19]. OVS enables automation through programmatic extensions and supports standard management interfaces and protocols. Additionally, OVS supports migration of VMs by migrating the network state associated with a migrated VM. Regarding QoS provisioning, OVS supports traffic shaping by HTB (Hierarchical Token Bucket), packet scheduling by HFSC (Hierarchical Fair Service Curve), and is ready for IPv6.

C. Data Center Network Virtualization Architectures

Several virtualization architectures for data center network virtualization have been proposed in the literature. Due to space limitations, we provide brief descriptions of selected representative approaches.

VL2 [39] provides a network abstraction of a single subnet connecting a service to all servers associated with it. Experiments showed that using the Clos topology, VL2 reaches average goodput of 60Gbps. Moreover, VL2 provides performance isolation by using Valiant Load Balancing. Unfortunately, VL2 is not able to provide bandwidth guarantees and its architecture strongly relies on the specific physical topology.

The authors of [45] propose SecondNet, an architecture that provides bandwidth guarantees without the need to maintain

bandwidth reservations in switches. The guarantees are provided under the assumption that a data center's structure is known in advance and is managed by a single entity. SecondNet provides isolation between data center network slices and is scalable by keeping all switches stateless, however, its performance strongly depends on the topology of the physical network.

The PortLand approach [40] addresses the issues of VMs' population scalability, migration and management. Its main limitations are a requirement of fat-tree multi-rooted physical topology and lack of bandwidth guarantees.

The Gatekeeper approach [46] aims at providing QoS guarantees and high bandwidth utilization. The QoS provisioning in Gatekeeper consists of defining a minimum and maximum transmission rate for each flow. This addresses the overprovisioning problem incurred by providing strict bandwidth guarantees. Unfortunately, Gatekeeper does not consider other QoS properties apart from bandwidth and lacks maturity.

The CloudNaaS architecture [47] is built based on OpenFlow and aims to provide application-specific addressing scheme, grouping of VMs and bandwidth reservations. The authors of CloudNaaS claim that the approach does not use overlays, however, they provide custom addressing schemes and use a VLAN-based isolation.

The Oktopus approach [48] provides performance guarantees and separate virtual network abstractions for tenants in tree-like physical topologies. An abstraction called 'virtual cluster' supports MapReduce applications, while a 'virtual oversubscribed cluster' supports applications using local communication patterns.

The NetLord architecture [49] relies on L2-in-L3 encapsulation and provides good scalability, logical isolation between tenants, low performance overheads, and centralized management. Unfortunately, NetLord does not provide any bandwidth or QoS guarantees.

The Seawall approach [50] defines a bandwidth allocation scheme and shares the bandwidth proportionally based on weights assigned to VMs. It provides bandwidth isolation using congestion-controlled tunnels implemented in hosts.

The NetShare approach [38] proposes statistical multiplexing for fair bandwidth allocation. Unfortunately, the approach has limited deployability due to the requirement of special features enabled by Fulcrum switches. Moreover, using the fairness mechanism excludes providing bandwidth guarantees.

D. Summary

In Table II, we summarize how the surveyed network virtualization architectures, software cloud platforms, and cloud virtual network appliances leverage the generic techniques identified in Section II. The presented mapping is based on the documentation of particular approaches and the literature review.

The surveyed IaaS cloud software platforms provide a unified, centralized point of management over a cloud infrastructure. All of the surveyed platforms provide an API to control virtual network appliances and add or modify a layer to create logically isolated subnets. The CloudStack and OpenStack/Quantum platforms provide basic QoS. Noteworthy is the fact that the offerings of Quantum can be further extended when new plugins emerge.

TABLE II
COMPOSITION OF BUILDING BLOCKS IN DATA CENTER NETWORK VIRTUALIZATION APPROACHES (+ YES, − NO)

Approach	Virtualization Building Block									
	Resource Reservation	Packet Scheduling	Traffic Profiling	Admission Control	Load Balancing	Device Aggregation	Device Emulation	Adding a Layer	Modifying a Layer	Other Uncommon
Cloud Software Platforms										
OpenNebula	−	−	−	−	−	−	+	−	+	−
OpenStack	−	−	−	−	−	−	+	+	−	−
OpenStack w. Quantum [a]	−	+	+	−	−	−	+	+	−	−
CloudStack	−	−	+	+	+	−	+	+	+	−
Cloud Virtual Network Appliances										
Linux Bridge	−	+	+	−	−	−	+	−	−	−
Open vSwitch	−	+	+	−	−	+	+	−	−	−
OpenFlow	−	+	+	+	−	+	+	−	−	−
Data Center Network Virtualization Architectures										
VL2	−	−	−	−	+	−	−	+	−	+ [b]
SecondNet	+	−	+	+	−	−	−	−	+	−
Portland	−	−	−	−	+	+	−	−	+	+[c]
Gatekeeper	+	+	+	+	−	−	+	−	−	−
Oktopus	−	+	+	+	+	+	−	−	−	−
CloudNaaS	+	+	−	+	−	+	+	−	+	−
NetLord	−	−	−	−	+	+	−	+	+	−
Seawall	−	+	+	−	−	−	+	+	−	−
NetShare	−	+	+	−	−	−	−	+	−	+[d]

[a] Expandable via plugins.
[b] Requires Clos physical topology.
[c] Requires fat-tree, multi-rooted physical topology.
[d] Requires Fulcrum switches.

The virtualization of the computing infrastructure implies the presence of hypervisors and the emulation of virtual network appliances located inside them. The limited set of features of default network bridges (we use Linux bridge as a representative example) can be further expanded by providing QoS and management APIs using Open vSwitch and OpenFlow.

Finally, complex network virtualization architectures employ various generic virtualization techniques. Most of the reviewed virtualization approaches provide QoS, however, only three of them use resource reservation mechanisms. Similarly, all but two solutions rely on overlaying techniques. Some approaches (e.g., VL2, Portland, NetShare) require specialized underlying physical infrastructure which can limit their deployability in practice.

We stress that there is currently no single approach that

provides all features which data center operators expect nowadays. However, combining particular approaches together and deploying them simultaneously in a data center may enable additional features that are not available by default. Such composition requires that the combined approaches are well understood in order to anticipate possible interferences and bottlenecks.

IV. Research Challenges

In this section, we outline open issues and research challenges in the area of network virtualization, performance modeling, and proactive QoS-aware resource management in the context of virtualized data center networks. The authors of [51] present research directions in the data center networking domain. They outline challenges regarding virtualized edge data centers, data center embedding, programmability, network performance guarantees, data center management, security, and pricing. We extend the set of challenging research directions by considering performance modeling and proactive model-based QoS-aware resource management which is part of our research agenda. We identify the following challenges and future research directions:

- The state-of-the-art is missing a mature widely used and standardized approach to network virtualization that would provide performance isolation and QoS guarantees. Based on the surveyed approaches, we observe deployments utilizing various diverse approaches to provide commonly demanded features. Although most approaches use the identified generic network virtualization building blocks as a basis, each of the solutions is based on a specific implementation and has different design assumptions. Therefore, one solution is usually incompatible with other solutions. The standardization of selected virtualization approaches that provide performance isolation and QoS management would enable new cloud offerings providing QoS guarantees.
- Another challenge is the lack of a generic modeling approach providing abstractions to model the performance-influencing factors of the network infrastructure as well as its degrees-of-freedom and run-time reconfiguration strategies. Existing modeling techniques are mainly focused on simulating specific architectures at the protocol level and do not provide generic modeling abstractions.
- Additionally, currently no holistic end-to-end performance prediction approach exists taking into account all layers of the system architecture including the software architecture, the deployment environment (server virtualization, middleware) and the networking infrastructure. There is generally a gap between performance models at the software architecture level (e.g., based on approaches such as UML MARTE [52], KLAPER [53], or PCM [54]) and low level network simulation models (e.g., based on NS-3 or OMNeT++).
- Finally, there is a lack of proactive network management techniques that automatically reconfigure the network during operation exploiting online performance prediction techniques to ensure that application SLAs are continuously satisfied while resource efficiency is optimized.

V. Conclusions

In this survey paper, we surveyed existing network virtualization approaches evaluating the extent to which they can be used as a basis for virtualization and resource management in modern cloud data centers. These generic approaches can be treated as network virtualization building blocks and further composed together to construct complex network virtualization architectures. We focused mainly on QoS and performance isolation aspects of data center networks, as this domain still lacks a mature and widely adopted virtualization approach. We surveyed representative state-of-the-art network virtualization solutions and showed how they employ the distinguished generic techniques. In addition, we categorized the support of software cloud platforms and software defined networks for network virtualization in a data center environment. Finally, we outlined open issues and research challenges in the area of performance modeling and proactive resource management of virtualized data center infrastructures.

Acknowledgment

This work is a part of RELATE project supported by the European Commission under the Seventh Framework Programme FP7 with Grant agreement no.: 264840ITN.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] "Gartner's data on energy consumption, virtualization, cloud," IT world, The IDG Network, 2008. [Online]. Available: http://www.itworld.com/green-it/59328/gartners-data-energy-consumption-virtualization-cloud

[3] S. Lohr, "Amazon's trouble raises cloud computing doubts," The New York Times, April 2011, http://www.nytimes.com/2011/04/23/technology/23cloud.html?_r=1.

[4] D. Durkee, "Why cloud computing will never be free," *Commun. ACM*, vol. 53, no. 5, pp. 62–69, 2010.

[5] B. Winterford, "Stress tests rain on amazon's cloud," IT News, April 2009, http://www.itnews.com.au/News/153451,stress-tests-rain-on-amazons-cloud.aspx.

[6] S. Wilson, "Appengine outage," CIO Weblog, April 2008, http://www.cioweblog.com/50226711/appengineoutage.php.

[7] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," *SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 164–177, 2003.

[8] J. Sugerman, G. Venkitachalam, and B.-H. Lim, "Virtualizing i/o devices on vmware workstation's hosted virtual machine monitor," in *Proceedings of the General Track: 2002 USENIX Annual Technical Conference*. USENIX Association, 2001, pp. 1–14.

[9] S. Kounev, F. Brosig, and N. Huber, "Descartes Meta-Model (DMM)," Karlsruhe Institute of Technology, Technical Report, 2012.

[10] S. Kounev, F. Brosig, N. Huber, and R. Reussner, "Towards self-aware performance and resource management in modern service-oriented systems," in *Proceedings of the 7th IEEE International Conference on Services Computing (SCC 2010)*. IEEE Computer Society, 2010.

[11] N. Huber, F. Brosig, and S. Kounev, "Model-based Self-Adaptive Resource Allocation in Virtualized Environments," in *6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2011)*, May 2011.

[12] S. S. R. Braden, D. Clark, "Integrated services in the internet architecture: an overview," RFC 1633, 1994.

[13] S. B. et al., "An architecture for differentiated services," IETF RFC 2475, 1998.

[14] A. Grzech and P. Świątek, "Parallel processing of connection streams in nodes of packet-switched computer communication systems," *Cybernetics and Systems*, vol. 39, no. 2, 2008.

[15] P. Świątek, A. Grzech, and P. Rygielski, "Adaptive packet scheduling for requests delay guaranties in packet-switched computer communication network," *Systems Science*, vol. 36, no. 1, pp. 7–12, 2010.

[16] S. Zhou, "Virtual networking," *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 4, 2010.

[17] J. Networks, "Network simplification with juniper networks virtual chassis technology," Whitepaper, 2011.

[18] H. Masuda, K. Murata, and Y. Shibuya, "Low tco and high-speed network infrastructure with virtual technology," in *Proceedings of the 37th annual ACM SIGUCCS fall conference*, ser. SIGUCCS '09. ACM, 2009, pp. 321–324.

[19] "Open vswitch," Online, 2011. [Online]. Available: openvswitch.org

[20] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

[21] G. Schmied, *Integrated Cisco, UNIX Network Architectures*. Cisco Press, 2009.

[22] L. Cheng, C.-L. Wang, and S. Di, "Defeating network jitter for virtual machines," in *Proc. of the Fourth IEEE International Conference on Utility, Cloud Computing*, 2011, pp. 65–72.

[23] J. Shafer, "I/o virtualization bottlenecks in cloud computing today," in *Proceedings of the 2nd conference on I/O virtualization*, ser. WIOV'10, 2010.

[24] A. Landau, D. Hadas, and M. Ben-Yehuda, "Plugging the hypervisor abstraction leaks caused by virtual networking," in *Proceedings of the 3rd Annual Haifa Experimental Systems Conference*, ser. SYSTOR '10. ACM, 2010, pp. 16:1–16:9.

[25] U. Hölzle, "Openflow at google," Keynote talk at Open Networking Summit 2012, 2012.

[26] K. Barabash, R. Cohen, D. Hadas, V. Jain, R. Recio, and B. Rochwerger, "A case for overlays in dcn virtualization," in *Proceedings of the 3rd Workshop on Data Center - Converged and Virtual Ethernet Switching*, ser. DC-CaVES '11. ITCP, 2011, pp. 30–37.

[27] R. C. E. Rosen, A. Viswanathan, "Rfc 3031: Multiprotocol label switching architecture," IETF RFC 3031, 2001.

[28] X. Jiang and D. Xu, "Violin: virtual internetworking on overlay infrastructure," in *Proceedings of the Second international conference on Parallel and Distributed Processing and Applications*, ser. ISPA'04. Springer-Verlag, 2004, pp. 937–946.

[29] M. Boucadair, P. Levis, D. Griffin, N. Wang, M. Howarth, G. Pavlou, E. Mykoniati, P. Georgatsos, B. Quoitin, J. Rodriguez Sanchez, and M. L. Garcia-Osma, "A framework for end-to-end service differentiation: Network planes and parallel internets," *Comm. Mag.*, vol. 45, no. 9, pp. 134–143, 2007.

[30] A. I. Sundararaj and P. A. Dinda, "Towards virtual networks for virtual machine grid computing," in *Proceedings of the 3rd conference on Virtual Machine Research And Technology Symposium - Volume 3*, ser. VM'04. USENIX Association, 2004, pp. 14–14.

[31] A. Ganguly, A. Agrawal, P. O. Boykin, and R. Figueiredo, "Wow: Self-organizing wide area overlay networks of virtual workstations," in *In Proc. of the 15th International Symposium on High-Performance Distributed Computing (HPDC-15*, 2006, pp. 30–41.

[32] I. C. Society, "IEEE 802.1Q virtual bridged local area networks," IEEE, Tech. Rep., 2005.

[33] "Vxlan: A framework for overlaying virtualized layer 2 networks over layer 3 networks," Online, 2011. [Online]. Available: tools.ietf.org/html/draft-mahalingam-dutt-dcops-vxlan-00/

[34] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 63–74, 2008.

[35] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: a scalable and fault-tolerant network structure for data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 75–86, 2008.

[36] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "Bcube: a high performance, server-centric network architecture for modular data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 63–74, 2009.

[37] J. Rothschild, "High performance at massive scale lessons learned at facebook," Onine, 2009, accessed June 2012. [Online]. Available: http://video-jsoe.ucsd.edu/asx/JeffRothschildFacebook.asx

[38] A. V. T. Lam, S. Radhakrishnan and G. Varghese., "Netshare: Virtualizing data center networks across services," University of California, San Diego, Technical Report CS2010-0957, 2010.

[39] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "Vl2: a scalable and flexible data center network," *Commun. ACM*, vol. 54, no. 3, pp. 95–104, 2011.

[40] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "Portland: a scalable fault-tolerant layer 2 data center network fabric," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 39–50, 2009.

[41] "Openstack networking ("quantum")," Online, 2011. [Online]. Available: http://wiki.openstack.org/Quantum

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

[43] "Cloudstack - open source cloud computing," Online, 2012. [Online]. Available: http://www.cloudstack.org/

[44] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T.-Y. Huang, P. Kazemian, M. Kobayashi, J. Naous, S. Seetharaman, D. Underhill, T. Yabe, K.-K. Yap, Y. Yiakoumis, H. Zeng, G. Appenzeller, R. Johari, N. McKeown, and G. Parulkar, "Carving research slices out of your production networks with openflow," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 129–130, 2010.

[45] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, "Secondnet: a data center network virtualization architecture with bandwidth guarantees," in *Proceedings of the 6th International COnference*, ser. Co-NEXT '10. New York, NY, USA: ACM, 2010, pp. 15:1–15:12.

[46] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. Guedes, "Gatekeeper: supporting bandwidth guarantees for multi-tenant datacenter networks," in *Proceedings of the 3rd conference on I/O virtualization*, ser. WIOV'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 6–6.

[47] 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*, ser. SOCC '11. New York, NY, USA: ACM, 2011, pp. 8:1–8:13.

[48] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 242–253, 2011.

[49] J. Mudigonda, P. Yalagandula, J. Mogul, B. Stiekes, and Y. Pouffary, "Netlord: a scalable multi-tenant network architecture for virtualized datacenters," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 62–73, 2011.

[50] A. Shieh, S. Kandula, A. Greenberg, C. Kim, and B. Saha, "Sharing the data center network," in *Proceedings of the 8th USENIX conference on Networked systems design and implementation*, ser. NSDI'11. Berkeley, CA, USA: USENIX Association, 2011.

[51] M. Bari, R. Boutaba, R. Esteves, L. Granville, M. Podlesny, M. Rabbani, Q. Zhang, and M. Zhani, "Data center network virtualization: A survey," *IEEE Communications Surveys and Tutorials*, September 2012.

[52] Object Management Group (OMG), "UML Profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE)," May 2006.

[53] V. Grassi, R. Mirandola, and A. Sabetta, "Filling the gap between design and performance/reliability models of component-based systems: A model-driven approach," *Journal of Systems and Software*, vol. 80, no. 4, pp. 528–558, April 2007.

[54] S. Becker, H. Koziolek, and R. Reussner, "The palladio component model for model-driven performance prediction," *Journal of Systems and Software*, vol. 82, no. 1, pp. 3–22, 2009.