Data Center Network Virtualization: A Survey

Md. Faizul Bari, Raouf Boutaba, Rafael Esteves, Lisandro Zambenedetti Granville, Maxim Podlesny, Md Golam Rabbani, Qi Zhang, and Mohamed Faten Zhani

***Abstract*—With the growth of data volumes and variety of Internet applications, data centers (DCs) have become an efficient and promising infrastructure for supporting data storage, and providing the platform for the deployment of diversified network services and applications (*e.g.*, video streaming, cloud computing). These applications and services often impose multifarious resource demands (storage, compute power, bandwidth, latency) on the underlying infrastructure. Existing data center architectures lack the flexibility to effectively support these applications, which results in poor support of QoS, deployability, manageability, and defence against security attacks. Data center network virtualization is a promising solution to address these problems. Virtualized data centers are envisioned to provide better management flexibility, lower cost, scalability, better resources utilization, and energy efficiency. In this paper, we present a survey of the current state-of-the-art in data center networks virtualization, and provide a detailed comparison of the surveyed proposals. We discuss the key research challenges for future research and point out some potential directions for tackling the problems related to data center design.**

***Index Terms*—Data Center Architecture, Virtualization, Virtualized Data Center**

I. Introduction

DATA centers have recently received significant attention as a cost-effective infrastructure for storing large volumes of data and hosting large-scale service applications. Today, large companies like Amazon, Google, Facebook, and Yahoo! routinely use data centers for storage, Web search, and large-scale computations [1]–[3]. With the rise of cloud computing, service hosting in data centers has become a multi-billion dollar business that plays a crucial role in the future Information Technology (IT) industry.

Despite their importance, the architectures of todays data centers are still far from being ideal. Traditionally, data centers use dedicated servers to run applications, resulting in poor server utilization and high operational cost. The situation improved with the emergence of server virtualization technologies (*e.g.*, VMware [4], Xen [5]), which allow multiple virtual machines (VMs) to be co-located on a single physical machine. These technologies can provide performance isolation between collocated VMs to improve application performance and prevent interference attacks. However, server virtualization alone is insufficient to address all limitations of todays data center architectures. In particular, data center networks are still largely relying on traditional TCP/IP protocol stack, resulting in a number of limitations:

- *No performance isolation*: Many of todays cloud applications, like search engines and web services have strict requirements on network performance in terms of latency and throughput. However, traditional networking technologies only provide best-effort delivery service with no performance isolation. Thus, it is difficult to provide predictable quality of service (QoS) for these applications.
- *Increased security risks*: Traditional data center networks do not restrict the communication pattern and bandwidth usage of each application. As a result, the network is vulnerable to insider attacks such as performance interference and Denial of Service (DoS) attacks [6].
- *Poor application deployability*: Today many enterprise applications use application-specific protocols and address spaces [7]. Migrating these applications to data center environments is a major hurdle because it often requires cumbersome modifications to these protocols and the application source code.
- *Limited management flexibility*: In a data center environment where both servers and networks are shared among multiple applications, application owners often wish to control and manage the network fabric for a variety of purposes such as load balancing, fault diagnosis, and security protection. However, traditional data center network architectures do not provide the flexibility for tenants to manage their communication fabric in a data center.
- *No support for network innovation*: Inflexibility of the traditional data center architecture prohibits network innovation. As a result, it is difficult to introduce changes in traditional data center networks such as upgrading network protocols or introducing new network services. In the long run, it will reduce the effectiveness of the initial capital investment in data center networks.

Motivated by these limitations, there is an emerging trend towards virtualizing data center networks in addition to server virtualization. Similar to server virtualization, network virtual-

Manuscript received 7 March 2012; revised 2 August 2012. This work was supported in part by the Natural Science and Engineering Council of Canada (NSERC) under the Smart Applications on Virtual Infrastructure (SAVI) Research Network, and in part by the World Class University (WCU) Program under the Korea Science and Engineering Foundation funded by the Ministry of Education, Science and Technology (Project No. R31-2008-000-10100-0).

Md. F. Bari, M. Podlesny, Md. Golam Rabbani, Q. Zhang, and M. F. Zhani are with the D.R. Cheriton School of Computer Science, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, Canada, N2L 3G1 (e-mail: faizulbari@gmail.com).

R. Boutaba is with the David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail:rboutaba@cs.uwaterloo.ca). He is also with the Division of IT Convergence Engineering, Pohang University of Science and Technology (POSTECH), Pohang 790-784, Korea.

R. Esteves and L. Z. Granville are with the Institute of Informatics — Federal University of Rio Grande do Sul, Av. Bento Gonçalves, 9500 — Porto Alegre, Brazil.

Digital Object Identifier 10.1109/SURV.2012.090512.00043

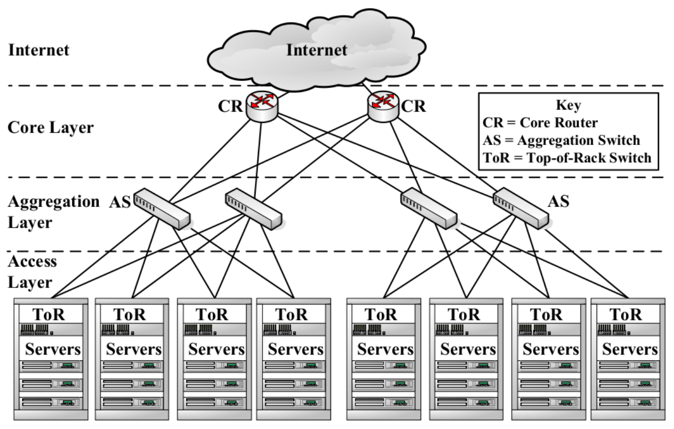


Fig. 1. Conventional data center network topology

ization aims at creating multiple virtual networks (VNs) on top of a shared physical network substrate [8] allowing each VN to be implemented and managed independently. By separating logical networks from the underlying physical network, it is possible to introduce customized network protocols and management policies. Also, since VNs are logically separated from one another, implementing performance isolation and application QoS is facilitated. Management procedures of VNs will be more flexible because every VN has its own control and management system. Furthermore, isolation offered in network virtualization environments can also minimize the impact of security threats. Finally, the deployment of new applications and services in virtualized data center environments is facilitated by customized protocols and address spaces, which expedites network innovation. So far, most of the existing work on network virtualization has been focused on virtualizing traditional Internet Service Provider (ISP) networks. Thus, virtualizing data center networks is a relatively new research direction, and a key step towards fully virtualized data center architectures.

While virtualizing data center networks addresses all of the aforementioned issues, it also opens a variety of new research challenges including virtualization techniques, addressing schemes, performance isolation, scalability, failure tolerance, monitoring, interfacing, security, pricing, and resource management. In this paper, we present a survey of recent research on virtualizing data center networks. Our contributions are three-fold: first, we provide a summary of the recent work on data center network virtualization. Second, we compare these architectures and highlight their design trade-offs. Finally, we point out the key future research directions for data center network virtualization. To the best of our knowledge, this work is the first to survey the literature on virtualizing data center networks.

The remainder of the survey is organized as follows. After introducing the terminology and definitions pertaining to data center virtualization (Section II), we summarize the proposals (Section III) related to data center network virtualization and compare them from different perspectives (Section IV). We then discuss the key research challenges for future explorations (Section V) and, finally, conclude our paper (Section VI).

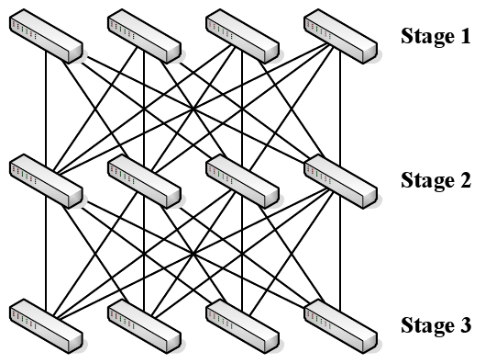


Fig. 2. Clos topology

II. Background

In this section, we present the terminology relevant to data center network virtualization that we will be using in this paper. Table I provides a list of abbreviations used throughout the paper.

A. Data Center

A data center (DC) is a facility consisting of servers (physical machines), storage and network devices (*e.g.*, switches, routers, and cables), power distribution systems, cooling systems.

A *data center network* is the communication infrastructure used in a data center, and is described by the network topology, routing/switching equipment, and the used protocols (*e.g.*, Ethernet and IP). In what follows, we present the conventional topology used in data centers and some other topologies that have been recently proposed.

Figure 1 shows a conventional data center network topology [9]. In this topology, the Top-of-Rack (ToR) switch in the access layer provides connectivity to the servers mounted on every rack. Each *aggregation switch* (AS) in the aggregation layer (sometimes referred to as distribution layer) forwards traffic from multiple access layer (ToR) switches to the core layer. Every ToR switch is connected to multiple aggregation switches for redundancy. The core layer provides secure connectivity between aggregation switches and *core routers* (CR) connected to the Internet. A particular case of the conventional topology is the *flat layer 2 topology*, which uses only layer 2 switches.

Clos topology is a topology built up from multiple stages of switches [10]. Each switch in a stage is connected to all switches in the next stage, which provides extensive path diversity. Figure 2 shows an example of a three-stage Clos topology.

Fat-tree topology [11] is a special type of Clos topology that is organized in a tree-like structure, as shown in Figure 3. The topology built of k-port switches contains k $pods$; each of them has two layers ($aggregation$ and $edge$) of $k/2$ switches. Each of $(k/2)^2$ core switches has one port connected to each of k pods. The i-th port of any core switch is connected to pod i so that consecutive ports in the aggregation layer of each pod switch are connected to core switches on $k/2$ strides. Each edge switch is directly connected to $k/2$ end-hosts; each of

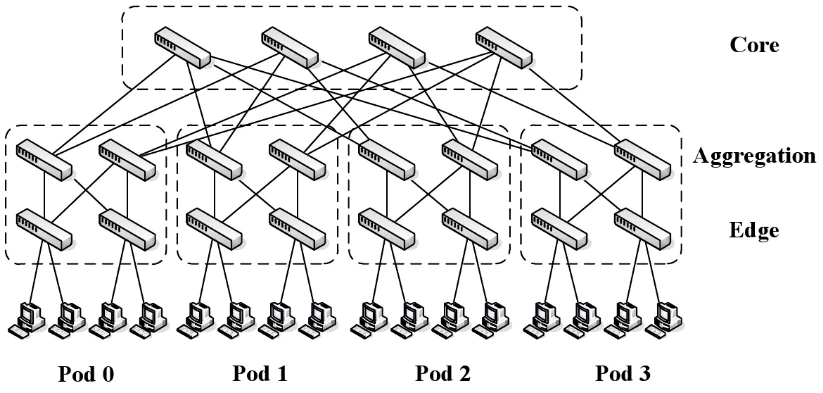


Fig. 3. Fat-tree topology (k = 4)

the remaining $k/2$ ports of an edge switch is connected to $k/2$ ports of an aggregation switch [12].

The above topologies have the properties that make them suitable for data center networks. However, data center topologies are not limited to the topologies presented in this Section. For example, BCube [13] is a data center network architecture based on hyper-cube topology. The interested reader can find a comparison of recent data center network topologies in [14].

B. Data Center Virtualization

A *Virtualized Data Center* is a data center where some or all of the hardware (*e.g.*, servers, routers, switches, and links) are virtualized. Typically, a physical hardware is virtualized using software or firmware called *hypervisor* that divides the equipment into multiple isolated and independent virtual instances. For example, a physical machine (server) is virtualized via a hypervisor that creates virtual machines (VMs) having different capacities (CPU, memory, disk space) and running different operating systems and applications.

A *Virtual Data Center* (VDC) is a collection of virtual resources (VMs, virtual switches, and virtual routers) connected via *virtual links*. While a Virtualized Data Center is a physical data center with deployed resource virtualization techniques, a Virtual Data Center is a logical instance of a Virtualized Data Center consisting of a subset of the physical data center resources. A *Virtual Network* (VN) is a set of virtual networking resources: virtual nodes (end-hosts, switches, routers) and virtual links; thus, a VN is a part of a VDC. *A network virtualization level* is one of the layers of the network stack (application to physical) in which the virtualization is introduced. In Figure 4, we show how several VDCs can be deployed over a virtualized data center.

Both network virtualization and data center virtualization rely on virtualization techniques to partition available resources and share them among different users, however they differ in various aspects. While virtualized ISP (VNs) networks mostly consist of packet forwarding elements (*e.g.*, routers), virtualized data center networks involve different types of nodes including servers, routers, switches, and storage nodes. Hence, unlike a VN, a VDC is composed of different types of virtual nodes (e.g., VMs, virtual switches and virtual routers) with diverse resources (e.g., CPU, memory and disk). In addition, in the context of network virtualization,

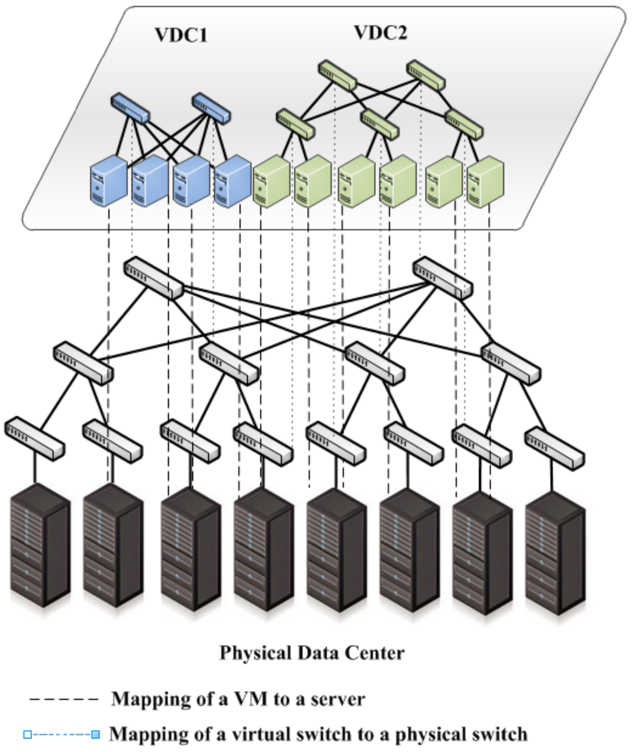

VDC1
VDC2
Physical Data Center
Mapping of a VM to a server
Mapping of a virtual switch to a physical switch

Fig. 4. Virtualization of a data center.

TABLE I
USED ABBREVIATION

Acronym	Description
DC	Data Center
VM	Virtual Machine
VN	Virtual Network
VDC	Virtual Data Center
$VLAN$	Virtual Local Area Network
ToR	Top-of-Rack
AS	Aggregation Switch
CR	Core Router
InP	Infrastructure Provider
SP	Service Provider
$IaaS$	Infrastructure-as-a-Service

virtual links are characterized by their bandwidth. Propagation delay is an important metric when nodes are geographically distributed. However, since a data center network covers a small geographic area, the propagation delay between nodes is negligible; hence, it is always ignored when defining VDC virtual links [15], [16].

Another key difference between data center networks and ISP networks is the number of nodes. While the number of nodes in ISP backbones is in order of hundreds (*e.g.*, 471, 487, and 862 nodes in Sprintlink, AT&T, and Verio ISPs, respectively [17]), it can go up to thousands in today's data centers (*e.g.*, around 12000 servers in one Google Compute cluster [18]). This can potentially raise scalability issues, and increase management complexity.

Furthermore, different from ISP networks, data center networks are built using topologies like the conventional tree, fat-tree, or Clos topologies with well defined properties, allowing to develop embedding algorithms optimized for such particular topologies (*e.g.*, Ballani *et al.* [16] proposed an embedding scheme applicable only to tree topology).

In summary, data center network virtualization is different from ISP network virtualization, because one has to consider different constraints and resources, specific topologies, and degrees of scalability. For a survey of ISP network virtualization the interested reader is referred to [8].

C. Business Model

In this section, we define main stakeholders in DC virtualization environment.

Specifically, one of the differences between the traditional networking model and network virtualization model is participating players. In particular, whereas the former assumes that there are two players: ISPs and end-users, the latter proposes to separate the role of the traditional ISP into two: an *Infrastructure Provider* (InP) and a *Service Provider* (SP). Decoupling SPs from InPs adds opportunities for network innovation since it separates the role of deploying networking mechanisms, *i.e.*, protocols, services (*i.e.*, SP) from the role of owning and maintaining the physical infrastructure (*i.e.*, InP).

In the context of data center virtualization, an InP is a company that owns and manages the physical infrastructure of a data center. An InP leases virtualized resources to multiple service providers/*tenants*. Each tenant creates a VDC over the physical infrastructure owned by the InP for further deployment of services and applications offered to end-users. Thus, several SPs can deploy their coexisting heterogeneous network architectures required for delivering services and applications over the same physical data center infrastructure.

III. Literature Survey

The virtualization of data center networks is still in its infancy, and recent research has mainly focused on how to provide basic functionalities and features including the partitioning of data center network resources, packet forwarding schemes and network performance isolation. Accordingly, we focus our attention in this survey on:

- Packet forwarding schemes which specify the rules used to forward packets between virtual nodes.
- Bandwidth guarantees and relative bandwidth sharing mechanisms that provide network performance isolation and more efficient network resource sharing, respectively.
- Multipathing techniques used to spread the traffic among different paths in order to improve load-balancing and fault-tolerance.

Nevertheless, there are other features worth considering when virtualizing data centers such as security, programmability, manageability, energy conservation, and fault-tolerance. So far, however, little attention has been paid to these features in the context of data center virtualization. We provide more details about the challenges related to these features in the future research directions section (Section V). In the following, we briefly describe the features we focus on in the paper, and then survey the proposals.

A forwarding scheme specifies rules for sending packets by switching elements from an incoming port to an outgoing port. A FIB allows to map MAC address to a switch port when making a decision about packet forwarding.

To support relative bandwidth sharing, congestion-controlled tunnels [6] may be used, typically implemented within a shim layer that intercepts all packets entering and leaving the server. Each tunnel is associated with an allowed sending rate on that tunnel implemented as a rate-limiter. The other alternatives [27] are group allocation for handling TCP traffic, rate throttling for controlling UDP traffic, and centralized allocation for supporting more general policies, *e.g.*, handling specific flows. The first alternative uses fair queueing; the second one relies on a shim layer below UDP.

One of the techniques to achieve bandwidth guarantee is the use of rate-limiters [7], [15], [16], [26]. In particular, a rate-limiter module is incorporated into a hypervisor of each physical machine; its role is to ensure that every VM does not exceed the allocated bandwidth. GateKeeper runs as a user level process in the Linux hypervisor (dom 0). It relies on the Open vSwitch (also running in the hypervisor) to track rates of each flow. Bandwidth guarantee in GateKeeper is achieved through rate limiter implemented using Linux hierarchical token bucket (HTB) scheduler running in the Xen hypervisor (dom 0) in the end hosts. CloudNaaS relies on Open vSwitch, which, although not explicitly stated, can be used for rate limiting. The deployment of rate limiters located at end-hosts makes it possible to avoid explicit bandwidth reservation at switches as long as the VDC management framework ensures that traffic crossing each switch does not exceed corresponding link capacity.

The main multipathing mechanisms used in data center networks are ECMP (Equal Cost Multipathing) [29] and VLB (Valiant Load Balancing) [30], [31]. To achieve load balancing, ECMP spreads traffic among multiple paths that have the same cost calculated by the routing protocol. VLB selects a random intermediate switch that will be responsible for forwarding an incoming flow to its corresponding destination. ECMP and VLB are implemented in L3 switches. On the other hand, path diversity available in data center networks can be exploited not only to provide fault tolerance but also to improve load balancing. In particular one effective technique to achieve load balancing is to create multiple VLANs that are mapped to different paths between each source and destination pair. This allows to distribute the traffic across different paths [20], [22].

TABLE II
CLASSIFICATION OF THE PROPOSALS IN LITERATURE

Proposal	References	Feature			
		Forwarding scheme	Bandwidth guarantee	Multipathing	Relative bandwidth sharing
Traditional DC	[4], [5], [19]	√		√	
SPAIN	[20]			√	
Diverter	[21]	√			
NetLord	[22]	√		√	
VICTOR	[23]	√			
VL2	[24]	√		√	
PortLand	[25]	√		√	
Oktopus	[16]		√		
SecondNet	[15]	√	√		
Seawall	[6]				√
Gatekeeper	[26]		√		
NetShare	[27]			√	√
SEC2	[28]	√			
CloudNaaS	[7]	√	√		

In Table II, we provide the classification of the surveyed projects according to the features they cover, and emphasize that a project may address more than one feature. A checkmark shows the features that are inherent to the surveyed proposals.

A. Traditional data center (DC)

Virtualization in current data center architectures is commonly achieved by server virtualization. Each tenant owns a group of virtual servers (machines), and isolation among tenants is achieved through VLANs. Data centers relying on this simple design can be implemented using commodity switches and popular hypervisor technologies (*e.g.*, VMware [4], Xen [5]). Besides, tenants can define their own layer 2 and layer 3 address spaces.

The main limitation of current data center architectures is scalability since commodity switches were not designed to handle a large number of VMs and the resulting amount of traffic. In particular, switches have to maintain an entry in their FIBs (Forwarding Information Base) for every VM, which can dramatically increase the size of forwarding tables. In addition, since VLANs are used to achieve isolation among tenants, the number of tenants is limited to 4096 (the number of VLANs allowed by the 802.1q standard [19]).

B. Diverter

Supporting logical partitioning of IP networks is essential for better accommodation of applications and services needs in large-scale multi-tenant environments like data centers. Diverter [21] is a software-only approach to network virtualization for a data center network that assumes no need for configuring switches or routers.

Diverter is implemented in a software module (called VNET) installed on every physical machine. When a VM sends an Ethernet frame, VNET replaces the source and the destination MAC addresses by the ones of the physical machines that host the source and the destination VMs, respectively. Then switches perform packet forwarding using the MAC addresses of the physical machines. VNET uses a modified version of the ARP protocol to discover any physical machine hosting a particular VM. Diverter requires that every VM have an IP address format encoding the tenant identity, the subnet, and the virtual machine address (currently use $10.tenant.subnet.vm$); thus, no address clashes occur between tenants. VNET performs routing between subnets using MAC address rewriting, which gives the illusion of traversing a gateway. Summarizing, Diverter provides layer 3 network virtualization that allows every tenant to control his own IP subnets and VMs addresses.

The main limitation of the proposal is that it does not provide any QoS guarantee, the support of which the authors consider as future work.

C. NetLord

To maximize revenue, providers of Infrastructure-as-a-Service (IaaS) [32] are interested in a full utilization of their resources. One of the most effective ways to achieve that is by maximizing the number of tenants using the shared infrastructure. NetLord [22] is a network architecture that strives at scalability of tenant population in data centers. The architecture virtualizes L2 and L3 tenant address spaces, which allows tenants to design and deploy their own address spaces according to their needs and deployed applications.

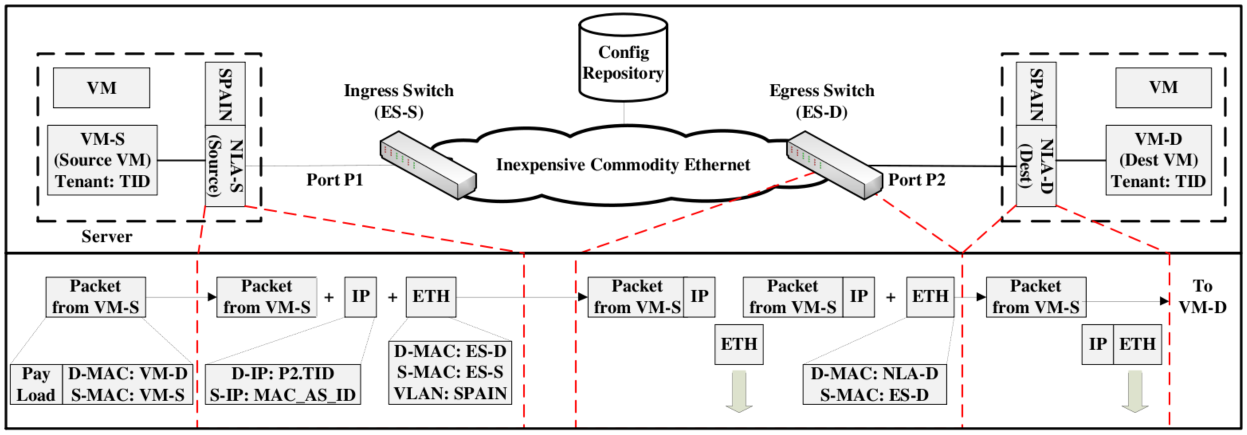


Fig. 5. NetLord architecture

The key idea of NetLord presented in Figure 5 is to encapsulate tenant L2 packets and transmit them over L2 fabric employing L2+L3 encapsulation. A L2 packet header specifies MAC addresses of a source VM and a destination VM. A source NetLord Agent (NLA) deployed on each physical server controls all VMs on the server. Specifically, it encapsulates an L2 packet by adding an extra L2 header and L3 header as follows. The extra source and destination L2 addresses determine the MAC addresses of the ingress and egress switches of a server hosting a source VM respectively. The extra source IP address reveals the ID of a tenant MAC address space, which enables the tenant to use multiple L2 spaces. The extra destination IP address specifies the port of an egress switch for forwarding packets to a destination server, and the ID of a tenant hosting a source and a destination VMs.

A packet is transferred over a data center network to an egress switch through the underlying L2 fabric over the path chosen by VLAN selection algorithm of SPAIN [20] (scheme relying on the VLAN support in existing commodity Ethernet switches to provide multipathing)[1]. Packet forwarding from an egress switch to a destination server is based on an L3 lookup of an egress port. An NLA forwards packets on a destination server to a destination VM using the IDs of a tenant and its MAC address space, and the destination L2 address of a VM in the encapsulated tenant packet. To support virtual routing, NetLord uses the same routing mechanism as Diverter [21]. To support SPAIN multipathing and keep per-tenant configuration information, NetLord uses several databases (referred to as Configuration Repository in Figure 5).

NetLord assumes that the edge switches support basic IP forwarding, however, not every Commercial Off-the-Shelf (COTS) switch [33] does that. The proposed encapsulation implies a higher packet size, which increases drops and fragmentation. Besides, NetLord uses SPAIN for multipath forwarding operating on a per-flow basis, which is not scalable. Finally, although the architecture provides isolation among tenants, it does not support any bandwidth guarantee.

[1]The description of SPAIN is provided later in the paper

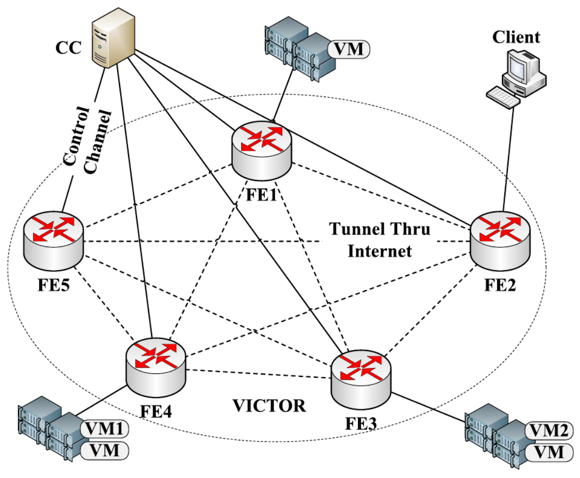


Fig. 6. VICTOR architecture.

D. VICTOR

Cloud tenants have a need to migrate services across data centers, to balance load within and across data centers, or to optimize performance of their services. On the other hand, cloud users want to have fast and efficient delivery of services and data. One approach that allows to achieve the above objectives of tenants and users is migration of VMs. To avoid service interruption, a VM should keep the same IP address during migration. Although that is not a challenge for migration within the same IP network, providing migration over different networks is not straightforward. VICTOR (Virtually Clustered Open Router) [23] is a network architecture for supporting migration of VMs across multiple networks that enables migrating VMs to keep their original IP addresses.

The main idea of VICTOR shown in Figure 6 is to create a cluster of Forwarding Elements (FE) (L3 devices) that serve as virtual line cards with multiple virtual ports of a single virtualized router. Thus, the aggregation of FEs performs data forwarding for traffic in a network. FEs are distributed over several networks, which helps to support migration of VMs across multiple networks. The control plane is supported by one or several Centralized Controllers (CC). A VM is deployed on a server connected to only one edge FE. A CC maintains a topology table that specifies the connectivity between FEs, and an address table that determines the connectivity between each VM and a FE, to which a server hosting the VM is connected. CC calculates the routing path from each FE to VMs and spreads that information among FEs, which rely on these routing tables for forwarding packets.

The main limitation of VICTOR is that it requires supporting FIBs of large sizes leading to scalability issues concerning FEs.

E. VL2

VL2 [24] is a data center network architecture that aims at achieving flexibility in resource allocation. In VL2, all servers belonging to a tenant (termed "service" in the paper) share a single addressing space regardless of their physical location meaning that any server can be assigned to any tenant.

VL2 is based on a non-oversubscribed Clos topology (see Figure 2) that provides easiness of routing and resilience. Packets are forwarded using two types of IP addresses: location-specific addresses (LAs) and application-specific addresses (AAs) used by switches and servers, respectively. VL2 relies on a directory system for AA-to-LA mappings. Before sending a packet, a VL2 server encapsulates the packet with the LA address of the destination ToR switch. Switches are not aware of AA addressing since they forward packets using LAs only. At the destination ToR switch, the packet is decapsulated and delivered to the destination AA server. To exploit path diversity, VL2 design relies on VLB and ECMP to spread traffic among multiple paths.

The separation between the addressing spaces of switches and servers improves the scalability of VL2 since ToR switches do not have to store forwarding information for a large number of servers. Furthermore, the VL2 directory system eliminates the need for ARP and DHCP requests, which are common sources of broadcast traffic in data centers. In addition, VLB and ECMP allow for a graceful degradation of the network after failures.

One limitation of VL2 is the lack of absolute bandwidth guarantees between servers, which is required by many applications (*e.g.*, multimedia services). The proposal is also highly coupled to the underlying (Clos) topology, and requires that switches implement OSPF, ECMP, and IP-in-IP encapsulation, which can limit its deployment.

F. PortLand

VM population scalability, efficient VM migration, and easy management are important characteristics of current and next-generation data centers. PortLand [25] addresses all these issues for a multi-rooted fat-tree topology (see Figure 3). In particular, the architecture proposes an L2 routing mechanism employing the properties of that topology. It supports plug-and-pay functionality for L2, which significantly simplifies administration of a data center network.

The main idea of PortLand is to use a hierarchical Pseudo MAC (PMAC) addressing of VMs for L2 routing. In particular, a PMAC has a format of *pod.position.port.vmid*, where *pod* is the pod number of an edge switch, *position* is its position in the pod, *port* is the port number of the switch the end-host is connected to, and *vmid* is the ID of a VM

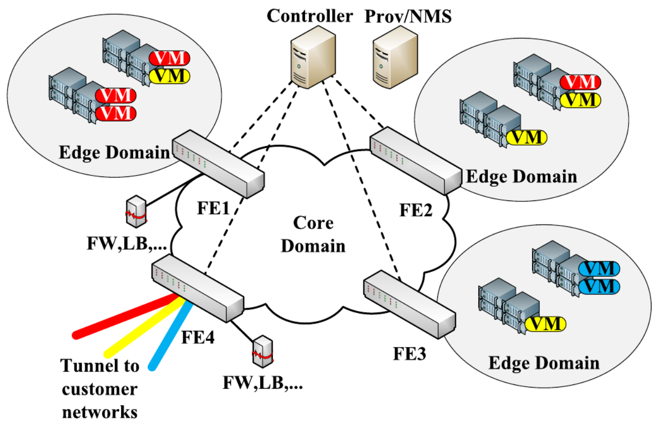


Fig. 7. SEC2 architecture.

deployed on the end host. The fabric manager (a process running on a dedicated machine) is responsible for helping with ARP resolution, multicast, and fault tolerance. Using k-port switches, forwarding table at each switch is limited to $O(k)$ records due to the properties of a multi-rooted fat-tree topology. An edge switch, to which a server hosting a VM is connected, maps an actual MAC (AMAC) of the VM to PMAC. The position of a switch in the topology may be set manually by an administrator, or, automatically through Location Discovery Protocol (LDP) proposed by the authors that relies on the properties of the underlying topology.

Despite PortLand benefits, there are several issues limiting the architecture. First, it requires multi-rooted fat-tree topology making PortLand inapplicable to other used data center network topologies. Second, resolving ARP requests by a single server (*i.e.*, the fabric manager) makes the architecture vulnerable to malicious attacks on the fabric manager, which lead to service unavailability if the fabric manager fails to perform address resolution. Third, each edge switch should have at least half of its ports connected to servers.

G. SEC2

To ensure wide adoption of cloud computing over data centers, it is important to provide all tenants with security guarantees. In particular, one of the important security issues is isolation of virtual networks dedicated to different tenants. Although using VLANs may be a potential solution for supporting isolated networks in a data center, there are several limitations of VLANs. First, the maximum number of VLANs is 4K because of the size of the VLAN ID space. Second, per-user control of security policy is a challenge. Third, having a large number of VLANs in the same data center network may induce complexity in network management and increased control overhead. Secure Elastic Cloud Computing (SEC2) [28] aims to resolve these drawbacks by separating packet forwarding and access control.

SEC2 is a data center network architecture that uses network virtualization techniques to provide secured elastic cloud computing service as shown in Figure 7. Network virtualization is supported through Forwarding Elements (FEs) and a Central Controller (CC). FEs are essentially Ethernet switches with the ability to be controlled from a remote CC that stores address mapping and policy databases. FEs perform address mapping, policy checking and enforcement, and packet forwarding. The network architecture has two levels: one core domain and multiple edge domains incorporating physical hosts. An edge domain is assigned a unique eid, and is connected to the core domain by one or more FEs. Each customer subnet has a unique $cnet\ id$ so that a VM can be identified by *(cnet id, IP)*. To isolate different customers within each edge domain, SEC2 uses VLAN with the scope limited within the same edge domain, thus, eliminating the limitation of the number of customers that can be supported due to VLAN ID size. If a customer offers public access to a VM, an FE forces all external packets to traverse through firewall and NAT middleboxes before reaching the private network. The advantage of SEC2 is that it does not require specialized routers or switches across the entire data center network. In addition, SEC2 supports VM migration [23] and VPC (Virtual Private Cloud) service, in which each user private network in a cloud is connected to its on-site network via Virtual Private Network (VPN) [34].

One of the limitations of SEC2 is that one edge domain cannot support VLANs of more than 4K different tenants. In addition, since FEs add outer MAC header when destination VM is not within the edge domain, SEC2 requires switches that support jumbo frames.

H. SPAIN

The current spanning tree protocol (STP) used for large Ethernet LANs is inefficient for supporting modern data center networks, since it does not exploit path diversity offered by data center networks, resulting in limited bi-section bandwidth and poor reliability [35]. Smart Path Assignment In Networks (SPAIN) [20] uses the VLAN support in existing commodity Ethernet switches to provide multipathing over arbitrary topologies.

SPAIN computes disjoint paths between pairs of edge switches, and pre-configures VLANs to identify these paths. An end-host agent installed on every host spreads flows across different paths/VLANs. To improve load balancing and avoid failures, the agent can change paths for some flows. For instance, if the traffic is not evenly spread across the paths, the agent can change the VLANs used by some flows. The agent also detects failed paths and re-routes packets around the failures by using a different path.

Whereas SPAIN provides multipathing, and improves load balancing and fault-tolerance, the proposal has some scalability issues. In particular, although path computation algorithm proposed by SPAIN is executed only when the network topology is designed or significantly changed, the scheme is computationally expensive for complicated topologies. In addition, SPAIN requires that switches store multiple entries for every destination and VLAN; it creates more pressure on switch forwarding tables than the standard Ethernet does. Furthermore, the number of paths is limited to the number of VLANs allowed by the 802.1q standard (4096) [19]. Finally, maintaining a mapping table between flows and VLANs leads to an additional overhead in each end-host.

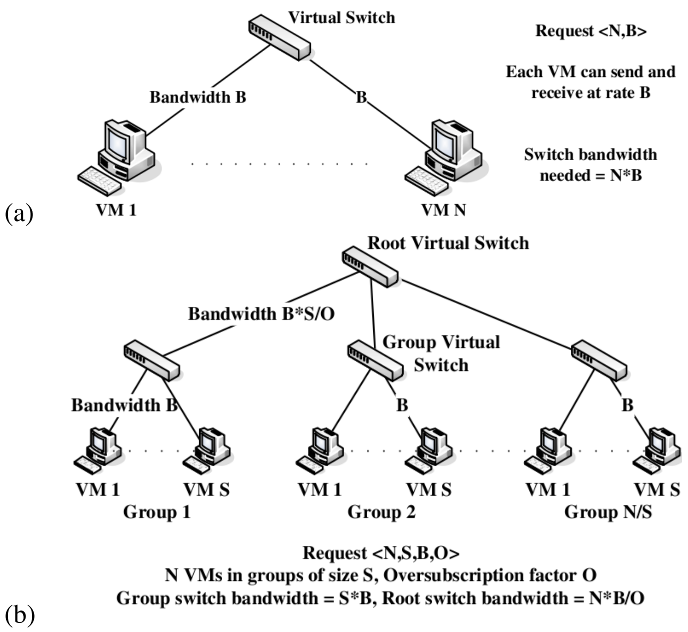


Fig. 8. Abstractions in Oktopus: (a) virtual cluster; (b) virtual oversubscribed cluster.

I. Oktopus

Although infrastructure providers offer to tenants on-demand computing resources through allocating VMs in data centers, they do not support performance guarantees on network resources to tenants. The mismatch between the desired and achieved performance by tenants leads to the following problems. First, variability of network performance induces unpredictable application performance in data centers making application performance management a challenge. Second, unpredictable network performance can decrease application productivity and customer satisfaction, leading to revenue losses. Oktopus [16] is the implementation of two virtual network abstractions ($virtual\ cluster$ and $virtual\ oversubscribed\ cluster$) for controlling the trade-off between the performance guarantees offered to tenants, their costs, and the provider revenue. Oktopus not only increases application performance, but offers better flexibility to infrastructure providers, and allows tenants to find a balance between higher application performance and lower cost.

A virtual cluster shown in Figure 8a provides the illusion of having all VMs connected to a single non-oversubscribed virtual switch. This is geared towards data-intensive applications like MapReduce that are characterized by all-to-all traffic patterns. A virtual oversubscribed cluster illustrated in Figure 8b emulates an oversubscribed two-tier cluster that is a set of virtual clusters interconnected via a virtual root switch-that suits applications featuring local communication patterns. A tenant can choose the abstraction and the degree of the oversubscription of the virtual network based on the communication pattern of the application the tenant plans to deploy in the VDC (*e.g.*, user-facing web-applications, data intensive applications). Oktopus uses a greedy algorithm for the resource allocation to the VDC.

The main limitation of Oktopus is that it works only for tree-like physical network topologies. Thus, an open question

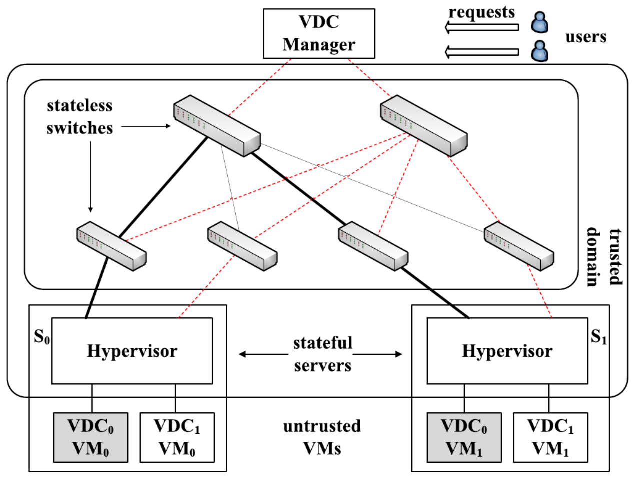


Fig. 9. SecondNet architecture.

is how to implement the abstractions of Oktopus for other topologies.

J. SecondNet

SecondNet [15] focuses on providing bandwidth guarantees among multiple VMs in a multi-tenant virtualized data center. In addition to computation and storage, the architecture also accounts for bandwidth requirements when deploying a VDC.

The main component of the SecondNet architecture shown in Figure 9 is the VDC manager that creates VDCs based on a requirement matrix that defines the requested bandwidth between VM pairs. SecondNet defines three basic service types: a high priority end-to-end guaranteed service (type 0), a better than best-effort service (type 1) that offers bandwidth guarantees for the first/last hops of a path, and a best-effort service (type 2). SecondNet uses a modified forwarding scheme called port-switching source routing (PSSR) that forwards packets using predefined port numbers instead of MAC addresses. PSSR improves the scalability of the data plane as paths are calculated at the source node. In this way, intermediate switches do not have to make any forwarding decision.

SecondNet achieves high scalability by moving information about bandwidth reservations from switches to server hypervisors. Besides, SecondNet allows resources (VM and bandwidth) to be dynamically added to or removed from VDCs (*i.e.,* elasticity). Using migration, SecondNet is also able to handle failures and reduce resource fragmentation. In addition, PSSR can be implemented with Multiprotocol Label Switching (MPLS) [36], which makes it easily deployable.

The main limitation of SecondNet is that its performance may depend on the physical topology of a network. For example, while the BCube [13] network achieves high network utilization, VL2 and fat-tree networks cannot. Further, SecondNet does not consider other performance characteristics that can be important to tenants such as latency.

K. Gatekeeper

Rodrigues *et al.* [26] look at the problem of network performance isolation formulating the associated requirements and devising a new scheme meeting those requirements named Gatekeeper. In particular, the authors argue that a solution for network performance isolation should be scalable in terms of the number of VMs, predictable in terms of the network performance, robust against malicious behaviour of tenants, and flexible concerning the minimum and maximum performance guarantees.

Gatekeeper focuses on providing guaranteed bandwidth among VMs in a multiple-tenant data center, and achieving a high bandwidth utilization. In general, achieving a strict bandwidth guarantee often implies non effective utilization of a link bandwidth when free capacity becomes available. Gatekeeper addresses this issue by defining both the minimum guaranteed rate and maximum allowed rate for each VM pair. These parameters can be configured to achieve minimum bandwidth guarantee, while ensuring that link capacities are effectively utilized by tenants. Gatekeeper creates one or more logical switches that interconnect VMs belonging to the same tenant. The virtual NIC (vNIC) of each receiving VM monitors the incoming traffic rate using a set of counters, and reports congestion to the vNIC of the sender that is exceeding its minimum guarantee by the largest amount. The rate limiter at the sender uses this information to control its traffic rate to reduce the level of congestion. Although fault-tolerance is not discussed in the paper, we believe that fault-tolerance is easily implementable by Gatekeeper since each vNIC can simply recalculate the fair share of each flow upon detecting a failure.

Like many existing schemes, Gatekeeper does not consider other performance metrics such as latency. Besides, Gatekeeper is still under development: the key features like dynamic creation and deletion of rate limiters are yet to be implemented. Furthermore, the scale of the experimental evaluation is small (with only two tenants and six physical machines). We believe that a complete implementation and more realistic experimental evaluation are necessary to truly evaluate the effectiveness of Gatekeeper in real cloud environments.

L. CloudNaaS

CloudNaaS [7] is a virtual network architecture that offers efficient support for deploying and managing enterprise applications in clouds. In particular, the architecture provides a set of primitives that suit the requirements of typical enterprise applications including application-specific address spaces, middlebox traversal, network broadcasting, VM grouping, and bandwidth reservation. Although many other datacenter network virtualization proposals have already addressed some of these issues (application specific address spaces and bandwidth reservation), they do not fully address all of the above issues. Motivated by this observation, CloudNaaS aims at providing a unified, comprehensive framework for running enterprise applications in clouds.

CloudNaaS relies on OpenFlow forwarding to achieve the objectives (*e.g.*, middlebox traversal) mentioned above. An application deployment in CloudNaaS includes several steps. First, an end-user specifies the network requirements to the cloud controller using the primitives defined by a network policy language. After the network requirements are translated into a communication matrix, the cloud controller determines the placement of VMs and generates network-level rules that can be installed on switches. Currently, CloudNaaS use a modified greedy bin-packing heuristic for placing VMs that takes into consideration communication locality. In addition, CloudNaaS provides several techniques to reduce the number of entries required in each switch, which include: (1) using a single path for best-effort traffic; (2) using limited paths for QoS traffic based on the number of traffic classes specified by type-of-service (ToS) bits; and, (3) assigning contiguous addresses to VMs placed behind the same edge switch, and use wildcard bits for aggregating IP forwarding entries. Besides, CloudNaaS also supports online mechanisms for handling failures and changes in the network policy specification by re-provisioning the VDCs. Currently, CloudNaaS is implemented using OpenFlow-enabled switch for forwarding; the end hosts use Open vSwitch based network stack for forwarding and compliance with OpenFlow.

One limitation of CloudNaaS is that limiting the traffic to a few paths may lead to congestion and/or poor network utilization. Finding a better trade-off between scalability and network utilization is still a challenging problem for CloudNaaS.

M. Seawall

Seawall [6] is a bandwidth allocation scheme that allows infrastructure providers to define how the bandwidth will be shared in a data center network with multiple tenants. The idea of Seawall is to assign weights to network entities generating traffic (*e.g.*, VMs, process), and to allocate bandwidth according to these weights in a proportional way. Seawall uses congestion-controlled tunnels between pairs of network entities to enforce bandwidth sharing policies. A shim layer implemented as a NDIS (Network Driver Interface Specification) packet filter is responsible for intercepting packets and adapting the rate the sender transmits packets at.

Seawall enforces bandwidth isolation among different tenants, and prevents malicious tenants from consuming all network resources. Besides, Seawall requires that a physical machine maintains state information only for its own entities, which improves scalability. Further, Seawall is agnostic to the transport protocol used by tenants, the number of flows used by an entity, and the number of destinations an entity sends traffic to. In all cases, Seawall shares the bandwidth proportionally and enforces isolation. Moreover, Seawall allows weights to be dynamically modified to accommodate changes in tenants requirements. Although Seawall does not address failures explicitly, it is adaptive to dynamic network conditions, making it fault-tolerant.

The first Seawall prototype was implemented only on Windows 7 and Hyper-V. Moreover, without admission control, it is unlikely that Seawall will be able to achieve absolute bandwidth guarantees for an increasing number of entities.

N. NetShare

NetShare [27] tackles the problem of bandwidth allocation in virtualized data center networks proposing a statistical multiplexing mechanism that does not require any changes in switches or routers. NetShare allocates bandwidth for tenants in a proportional way and achieves high link utilization for infrastructure providers. In NetShare, data center network links are shared among services, applications, or corporate groups, rather than among individual connections. In this way, one service/application/group cannot hog the available bandwidth by opening more connections.

NetShare can be implemented in three possible ways: group allocation, rate throttling, and centralized allocation. NetShare uses group allocation to handle TCP flows. Group allocation uses fair queueing to provide fair bandwidth allocation among different services and is implemented through Deficit Round Robin (DDR) [37]. Rate throttling is used to control the traffic generated by UDP sources and avoid excessive bandwidth consumption, and is implemented through a shim layer placed below UDP at each host. The shim layer controls the sending rate by analyzing the traffic measured at the receiver side and adjusting the sending rate accordingly. To implement more general policies, *e.g.*, to allocate unused bandwidth to specific flows, the scheme uses the centralized bandwidth allocation. NetShare relies on the routing protocol to handle failures, and multipath is possible with the use of ECMP.

Scalability of NetShare can be an issue because queues have to be configured at each switch port for each service/application. Moreover, NetShare relies on the specific features of Fulcrum switches to implement its mechanisms, which reduces its deployability. In addition, NetShare aims to achieve fairness in bandwidth allocation, and, thus, does not provide any absolute bandwidth guarantees to services.

IV. Comparison

Whereas the previous section surveys prominent research proposals and their salient features, this section compares these proposals using a set of qualitative metrics. In particular, we evaluate each proposal using the following five criteria: scalability, fault-tolerance, deployability, QoS support, and load-balancing. Scalability and fault-tolerance are important design concerns for data centers comprising large numbers of servers and network resources, and expected to support a large number of tenant applications. As data centers typically use commodity servers and network hardware today, deployability is a key issue that concerns how much change to the infrastructure is necessary for implementing a particular architecture. QoS is an increasing concern of tenants and is important to the success of virtualized data center architectures. Finally, load-balancing is an important objective of network operators for traffic engineering and minimizing congestion in data center networks. We summarize the results of our comparison in Tables III-VI. Each table compares the proposals using specific criteria of a particular feature. In the following subsections, we will provide detailed discussion of our evaluation of each performance metric.

A. Scalability

Achieving high scalability in virtualized data centers requires address spaces that support large number of tenants and their VMs. Furthermore, since todays commodity switches often have limited memory size, it is necessary to keep the

TABLE III
QUALITATIVE COMPARISON OF THE FORWARDING SCHEMES

Proposal	Scalability	Fault-tolerance	Deployability	QoS	Load balancing
$Traditional\ DC$	Low	No	High	No	No
$Diverter$	High	Yes	High	No	No
$NetLord$	High	No	Low	No	Yes
$VICTOR$	Low	Yes	Low	No	No
$VL2$	High	Yes	Low	No	Yes
$PortLand$	High	Yes	Low	No	Yes
$SecondNet$	High	Yes	High	Yes	No
$SEC2$	Low	No	Low	No	No
$CloudNaaS$	Low	Yes	Low	Yes	No

TABLE IV
QUALITATIVE COMPARISON OF THE PROPOSALS REGARDING MULTIPATHING

Proposal	Scalability	Fault-tolerance	Deployability	QoS	Load balancing
$Traditional\ DC$	Low	No	High	No	No
$SPAIN$	Low	Yes	High	No	Yes
$NetLord$	High	No	Low	No	Yes
$VL2$	High	Yes	Low	No	Yes
$PortLand$	High	Yes	Low	No	Yes

TABLE V
QUALITATIVE COMPARISON OF THE PROPOSALS REGARDING BANDWIDTH GUARANTEE

Proposal	Scalability	Fault-tolerance	Deployability	QoS	Load balancing
$Oktopus$	High	Yes	High	Yes	No
$SecondNet$	High	Yes	High	Yes	No
$Gatekeeper$	High	Yes	High	Yes	No
$CloudNaaS$	Low	Yes	Low	Yes	No

TABLE VI
QUALITATIVE COMPARISON OF THE PROPOSALS REGARDING RELATIVE BANDWIDTH SHARING

Proposal	Scalability	Fault-tolerance	Deployability	QoS	Load balancing
$Seawall$	High	Yes	High	No	No
$NetShare$	Low	Yes	Low	No	Yes

number of forwarding states in each switch at minimum for achieving high scalability.

Table VII shows the maximum number of tenants, VMs per tenant, and the size of the forwarding table. The maximum numbers of tenants and VMs depend mainly on the number of bits used to identify tenants and VMs. The number of VMs per tenant depends on the address space supported by IPv4, which can be extended when using IPv6. Depending on the forwarding scheme, the size of the forwarding table depends on the number of VMs, physical machines, switches or pods. In practise, the number of VMs is higher than the number of physical machines, which is in turn higher than the number of switches. We also notice that VICTOR and Portland do not support multi-tenancy.

Among the architectures surveyed in the paper, SecondNet, Seawall, and Gatekeeper achieve high scalability by keeping states at end-hosts (*e.g.*, hypervisors) rather than in switches. NetLord and VL2 achieve high scalability through packet encapsulation maintaining the forwarding state only for switches in the network. Diverter is also scalable, because its switch forwarding table contains only MAC addresses of the physical nodes (not those of VMs). On the other hand,

TABLE VII
SCALABILITY OF THE FORWARDING SCHEMES

Proposal	**Criteria**		
	Number of tenants	**Number of VMs per tenant**	**Size of the forwarding table per path**
Traditional DC	2^{12} (the number of VLANs allowed by 802.1q)	2^{32}	Number of VMs
Diverter	2^t (t is the number of bits used to identify the tenant, $t < 32$)	2^{32-t}	Number of physical machines
NetLord	2^{24} (24 bits used for the tenant id)	2^{32}	Number of edge switches
VICTOR	multi-tenancy is not supported	2^{32}	Number of VMs
VL2	2^t (t is the number of bits used to identify the tenant, $t < 32$)	2^{32-t}	Number of edge switches
PortLand	multi-tenancy is not supported	2^{32}	Number of pods
SecondNet	Unlimited (the tenant id is handled by the management server)	2^{32}	Number of neighbours (neighbours could be servers or switches)
SEC2	2^{12} per edge domain (the number of VLANs allowed by 802.1q)	2^{32}	Number of VMs
CloudNaaS	Unlimited (the tenant id is handled by the management server)	2^{32}	Number of edge switches

SPAIN, VICTOR, and CloudNaaS are less scalable because they require maintaining a per-VM state in each switching/forwarding element. Although CloudNaaS provides some optimization for improving scalability, such an optimization limits path-diversity provided in the network and deteriorates overall effectiveness of the approach. Further, CloudNaaS is currently implemented using OpenFlow, and OpenFlow is known to have scalability issues in large data center due to use of centralized controllers. SEC2 is not scalable because the addressing scheme limits the numbers of tenants and subnets supported in the network. NetShare relies on a centralized bandwidth allocator, which makes it difficult to scale to large data centers.

B. Fault-tolerance

In the context of virtualized data centers, fault-tolerance covers failure handling of components in the data plane (*e.g.*, switches and links) and control plane (*e.g.*, lookup systems). We found that most of the architectures were robust against failures in data plane components. For instance, SecondNet uses a spanning tree signalling channel to detect failures, and its allocation algorithm to handle them. A SPAIN agent can switch between VLANs in the occurrence of failures, NetLord relies on SPAIN for fault-tolerance, and VL2 and NetShare rely on the routing protocols (OSPF). Diverter, VICTOR, and SEC2 employ the underlying forwarding infrastructure for failure recovery. Schemes such as Oktopus and CloudNaaS handle failure by re-computing the bandwidth allocation for the affected network. Schemes including Seawall and Gatekeeper can adapt to failures by re-computing the allocated rates for each flow.

Control plane components in data center network architectures include centralized lookup systems for resolving address queries (NetLord, VICTOR, VL2, Portland, SEC2), centralized flow management technologies (CloudNaaS uses OpenFlow), spanning tree signalling (SecondNet), and routing protocols (NetShare and VL2). Failures of these control plane components can lead to malfunctioning of part or the whole data center and result in inability to detect failures in the data plane.

The impact of failures in architectures with control plane based on spanning tree protocols depends on the time that the protocol takes to converge after topology changes. Adaptations in the basic spanning tree protocol such as *Rapid Spanning Tree Protocol (RSTP)* [38] can reduce the convergence time. Similar to STP, failures in instances of routing protocols such as OSPF require routes recalculation, which may take a variable time depending on the size of the network and on the current protocol configuration. However, as shown in [24], the convergence time of OSPF (less than one second) is not a prohibitive factor in real data center networks.

OpenFlow, used by CloudNaas, is based on a centralized controller that defines the behaviour of OpenFlow-based switches through a set of rules and associated actions. The centralized design of the OpenFlow controller makes it prone to failures and performance bottlenecks. HyperFlow [39] is a proposal aiming at providing logically centralized but physically distributed OpenFlow controllers. In HyperFlow, when a failure occurs in one controller, the switches associated with the failed controller are reconfigured to communicate with another available controller.

Distributed lookup systems can be used to minimize the negative impact of failures in address lookup systems. For example, VL2 architecture proposes the use of replicated state machine (RSM) servers to implement a replicated *directory system*, which enables reliability without affecting performance.

TABLE VIII
DEPLOYABILITY OF THE FORWARDING SCHEMES

Proposal	Used features			
	Physical machines	**Edge switches**	**Core switches**	**Centralized management Server**
Traditional DC	Commodity	Commodity	Commodity	No
Diverter	MAC address rewriting	Commodity	Commodity	No
NetLord	MACin-IP encapsulation; SPAIN	IP forwarding	Commodity	Yes
VICTOR	Commodity	IP routing	IP routing	Yes
VL2	Commodity	IP-in-IP encapsulation; IP routing; VLB; ECMP	IP routing; VLB; ECMP	Yes
PortLand	Commodity	MAC address rewriting; forwarding based on MAC address prefixes; ECMP ARP management; Location Discovery Protocol	Location Discovery Protocol; ECMP	Yes
SecondNet	Commodity	MPLS	MPLS	Yes
SEC2	Commodity	MAC-In-MAC encapsulation	Commodity	Yes
CloudNaaS	Commodity	IP routing; QoS; forwarding based on IP address prefixes	IP routing; QoS; forwarding based on IP address prefixes	Yes

C. Deployability

As mentioned previously, deployability is a key aspect of any data center network virtualization architecture. In our comparison summarized in Tables III-VI, we evaluate the deployability of an architecture as *high* if the architecture can be deployed over commodity switches with software modifications. On the other hand, *low* deployability refers to architectures requiring devices with the features that are not available in every switch (*e.g.*, support L3 forwarding, specific protocols).

We summarize our detailed comparison with respect to deployability in Table VIII, which describes the required features to be implemented in hypervisors (on physical machines), edge switches, and core switches. Commodity switches support mainly L2 forwarding and VLAN technology whereas Commodity hypervisors create only isolated VA forwarding scheme specifies rules for sending packets by switching elements from an incoming port to an outgoing port. A FIB allows to map MAC address to a switch port when making a decision about packet forwarding.Ms. The table also shows, which scheme requires a centralized management server. Depending on the scheme, this server can have different functionalities such as address management (Portland, VL2), tenants management (NetLord and VL2), routing computation (VICTOR and SEC2), and resource allocation (SecondNet).

We can observe that while some surveyed architectures (SPAIN, Diverter, and Seawall) require change only in the hypervisor, most of the surveyed architectures require extra hardware features. In particular, these features include MAC-in-MAC (SEC2) encapsulation, L3 forwarding (VL2, NetLord), DRR (NetShare), network directory service (NetLord, VL2, Portland, VICTOR, SEC2), and programmable hardware (CloudNaaS) that may not be easily supported by commodity switches. Thus, implementing those architectures can increase the overall cost of the network. Nevertheless, with hardware evolution and wide adoption of programmable hardware, it is not excluded that these technologies become common place in the near future.

Finally, we would like to mention that data centers managers tend to deploy commodity equipment, which are cheap and easily replaceable. Using this equipment is not always a synonym of lack of scalability. For instance, in the case of traditional data centers, commodity switches have to store MAC addresses of all hosted VMs. It induces a scalability issue because commodity switches often have a limited amount of resources (*i.e.*, size of the FIB table) [22]. However, the forwarding scheme proposed in NetLord requires commodity switches to store only MAC addresses of edge switches. The number of switches being much smaller than the number of VMs in a data center drastically improves scalability. In both conventional and NetLord architectures, commodity switches are used, however, the forwarding scheme makes the difference, hence there is no scalability problem in NetLord.

D. QoS Support

QoS in virtual networks is achieved by allocating guaranteed bandwidth for each virtual link. Oktopus, SecondNet, Gatekeeper, and CloudNaaS provide guaranteed bandwidth allocation for each virtual network. On the other hand, Seawall and NetShare provide weighted fair-sharing of bandwidth among tenants; however, they do not provide guaranteed bandwidth allocation meaning that there is no predictable performance. Whereas the remaining architectures do not discuss QoS issues, we believe that it is possible to support QoS in these architectures by properly combining them with the ones that support bandwidth guarantee (*e.g.*, incorporating Oktopus into NetLord).

E. Load-balancing

Load-balancing is a desirable feature for reducing network congestion while improving network resource availability and application performance. Among the architectures surveyed in the paper, SPAIN and NetLord (which relies on SPAIN) achieve load-balancing by distributing traffic among multiple spanning trees. To achieve load balancing and realize multipathing, Portland and VL2 rely on ECMP and VLB. Lastly, Diverter, VICTOR, and SEC2 are essentially addressing schemes that do not explicitly address load-balancing.

F. Summary

Our comparison of different proposed architectures reveal several observations. First, there is no ideal solution for all the issues that should be addressed in the context of data center network virtualization. This is mainly because each architecture tries to focus on a particular aspect of data center virtualization. On the other hand, we believe that it is possible to combine the key features of some of the architectures to take advantage of their respective benefits. For example, it is possible to combine VICTOR and Oktopus to deploy virtualized data center with bandwidth guarantees while providing efficient support for VM migration. Second, finding the best architecture (or combination) requires a careful understanding of the performance requirements of the applications residing in the data centers. Thus, the issues discussed in this section require further research efforts in the context of different cloud environments.

V. FUTURE RESEARCH DIRECTIONS

In this section, we discuss some of the key directions for future explorations regarding data center network virtualization.

A. Virtualized Edge Data Centers

Most of the existing studies so far on data center network virtualization have been focusing on large data centers containing several thousands of machines. Although large data centers enjoy economy-of-scale and high manageability due to their centralized nature, they have their inherent limitations when it comes to service hosting. In particular, economics factors dictate that there will be only a handful of large data centers built in locations where construction and operational (e.g. energy) costs are low [40]. As a result, these data centers may be located far away from end users, resulting in higher communication cost and potentially sub-optimal service quality in terms of delay, jitter and throughput.

Motivated by this observation, recent proposals such as mist [41], EdgeCloud [42], micro-data centers [43], nano-data centers [44] have been put forward to advocate building small-scale data centers for service hosting at the network edge (e.g. access networks), where services can be hosted close to the end users. In this paper we adopt the terminology of *edge data centers* to refer to small data centers located at network edge. While not as cost-efficient as large data centers, edge data centers offer several key benefits compared to large remote data centers [43]: (1) They can offer better QoS for delay-sensitive applications such as video streaming, online gaming, web telephony and conferencing. (2) They can reduce network communication cost by reducing the traffic routed across network providers. (3) The construction cost of edge data centers is lower compared to large remote data centers. In fact, many existing telecommunication and Internet Service Providers (ISP) are willing to leverage their existing infrastructure to provide value-added services using edge data centers [45]. Therefore, it is envisioned that future cloud infrastructures will be multi-tiered, where edge data centers will complement remote data centers in providing high quality online services at low cost.

Similar to large data centers, virtualization is required in edge data centers for supporting VDCs from multiple tenants with diverse performance objectives and management goals. However, virtualizing edge data centers also imposes several new research challenges:

- For a service provider, one fundamental problem is how to best divide the service infrastructure between remote and edge centers to achieve the optimal tradeoff between performance and operational cost? This problem is commonly known as the service placement problem [46]. Finding a solution to this problem is essential for service providers to use edge data center-based service architectures. This problem shares many similarities with the traditional replica placement problem [47]. However, existing solutions have not studied the dynamic case, where demand and system conditions (e.g. resource price and network conditions) can change over time. In this case, if the placement configuration needs to be changed, it is also necessary to consider the cost of reconfiguration (such as VM migration) in the optimization model.
- How to efficiently manage services hosted in multiple data centers? As there can be a large number of edge data centers, monitoring and controlling resources in such a large infrastructure have inherent challenges and can potentially incur a significant overhead. Minimizing this management overhead is a major issue worth investigation.

We believe addressing the above research challenges will be crucial to the success of multi-tiered cloud infrastructures.

B. Virtual data center embedding

Accommodating a high number of VDCs depends on a efficient mapping of virtual resources to physical ones. This problem is commonly referred to as embedding and has been the subject of extensive research in the context of network virtualization [48]–[51]. Data center architectures like SecondNet [15] and Oktopus [16] have proposed heuristics to cope with the NP-hardness of the embedding problem. However, there are several other issues concerning the design of virtual data center embedding algorithms:

- In a virtualized data center, there are other resources besides physical servers that can be virtualized. They include routers, switches, storage devices, and security systems. Existing embedding solutions for data centers have only focused on achieving the optimal VM embedding to satisfy bandwidth and processing requirements. We believe the embedding algorithms for VDCs should consider requirements for other resources as well.

- Resource demand for data center applications can change over time, which means that embedding of VDCs is also subject to change. Designing a VDC re-embedding algorithm that considers reconfiguration cost of VMs (e.g. migration cost) and virtual topologies is still an open research problem.
- Energy consumption is a big concern in data centers since it accounts for a significant amount of the data center operational costs. Carey reports [52] that according to the Environmental Protection Agency (EPA) data centers consumed about 3% of the US total electricity use in 2001. Moreover, estimated energy consumption of data center servers was about 2% of the world's electricity. According to [53], computing equipment of a typical 5000-square-foot data center, which includes processors, server power supplies, other server components, storage and communication equipment, consumes 52% of the total DC energy usage; supply systems consisting of the UPS (uninterruptible power supply), power distribution, cooling, lighting, and building switchgear consume 48%. Greenberg et al. [54] report that a network of a data center consumes 10-20% of its total power. Designing "green" virtual data center embedding algorithms that take into account energy consumption will help administrators to reduce costs and comply with the new environmental concerns. In particular, network virtualization helps in reducing energy consumption through decreasing the number of physical routers/switches that need to be active by consolidating a large number of virtual resources on a smaller number of physical ones. However, despite recent effort on designing energy-aware data center networks [55], none of the existing embedding algorithms has considered energy cost. The main challenge in reducing energy consumption is how to jointly optimize the placement of both VMs and VNs for saving energy.
- Fault-tolerance is another major concern in virtualized data centers. The failure of a physical link can cause disruption to multiple VDCs that share the link. In order to minimize the application performance penalty due to failures, it is necessary for the tenants to find embeddings that are fault tolerant. Existing work on survivable virtual network embedding [56] represents an initial step towards this direction.
- Finally, some tenants may wish to deploy VDCs across data centers from multiple geographical regions. This raises the problem of embedding VDCs across multiple administrative domains. In this context, devising a framework that finds an efficient embedding without sacrificing the autonomy of individual infrastructure providers becomes a challenging problem. Recent work such as Polyvine [57] represents an initial effort for tackling this problem.

Finally, the problem of VDC embedding also raises the question of finding ideal data center physical topologies for VDC embedding. Even though the proposed data center architectures have relied on different network topologies such as Fat-Tree and Clos, it is unclear which topology is best suited for VDC embedding. For example, it has been reported that the SecondNet embedding scheme achieves high server and network utilization for a BCube topology [15] than for a fat-tree topology. Thus, we believe it is important to analyze the effect of VDC embedding on the design of physical data center network topologies.

C. Programmability

Network programming is motivated by the desire to increase flexibility and innovation by decomposing network functions to facilitate the introduction of new protocols, services, and architectures. Simply stated, network programmability can be defined as the ability to run third party code on a network device (e.g., a router) in both the control plane and the data plane. Network programmability has recently received renewed attention in the research community. In particular, the concept of software defined networking (SDN) aims at providing a simple API for programming network control plane. Network programmability benefits from virtualization techniques regardless of the context considered (*i.e.*, ISP network or DC). For example, running a customized code on a virtual node not only has no effect on other virtual nodes of the network (thanks to isolation), but also does not cause disruption in the physical substrate, which was a major concern for the adoption of network programmability. In the context of virtualized data centers, network programmability provides a modular interface for separating physical topologies from virtual topologies, allowing each of them to be managed and evolved independently. As we have already seen, many architectural proposals surveyed in the paper are relying on network programming technologies such as Openflow. However, in the context of virtualized multi-tenant data centers, network programmability needs to address a number of research challenges:

- Current data center network architecture proposals only allow for controlling layer 2 and layer 3 protocols. This design mandates the use of the same layer 2 and layer 3 protocols (e.g. IPv4 and Ethernet) by all tenants. Providing programming APIs for virtualization at different layers of the network stack will add significant flexibility to data center networks.
- While programmable networking technologies offer management flexibilities to tenants and infrastructure providers, they also open up opportunities for malicious tenants to misuse the infrastructure. Infrastructure providers need to determine how to provide access and how much control to delegate to tenants so that, the tenants get a satisfactory level of flexibility in-terms of programming the network devices while ensuring safe and secured co-existence of multiple tenants.
- Network vendors may offer non-standard, proprietary programming APIs. An interesting research challenge is to understand the impact of building a data center network infrastructure from heterogeneous equipments with

different hardware level APIs. Introducing heterogeneity has both advantages and disadvantages. The major disadvantage is the administrative overhead introduced by the divergent interfaces, while the advantage is that some vendor-specific features may be desirable in certain situations.

Currently, OpenFlow [58] and its virtualization layer FlowVisor [59] are the most prominently proposed technologies for achieving programmability in data center networks. OpenFlow is an abstraction layer that allows users to define the behaviour of networking switches by using special components called controllers. FlowVisor [59] is a network virtualization layer that allows multiple controllers (one controller per tenant) to share a single OpenFlow switch. Recent research efforts have been carried to deploy OpenFlow in data center networks [39], [60], [61]. One of the limitations of OpenFlow is scalability. Currently, OpenFlow adopts a centralized architecture where a single controller is responsible for managing all OpenFlow-enabled switches in the network. Since a large data center network typically serves million of flows simultaneously, an OpenFlow switch may become a performance bottleneck. There are some proposals that aim to overcome this issue. In particular, DevoFlow [61] controls only over a subset of the flows (*i.e.*, long lived "elephant" flows), and HyperFlow [39] uses distributed controllers with a unified logical view. Similarly, the scalability of FlowVisor is also a subject needing further investigation, given the large number of tenants involved in a virtualized data center. A possible avenue for improving FlowVisor scalability is to determine the optimal number and placement of FlowVisor instances in a programmable data center network. Finally, other programmable platforms (*e.g.,* active networks, mobile agents, and Script MIB) could also be evaluated in the context of virtualized data center networks.

D. Network performance guarantees

Commercial data centers today are home to a vast number of applications with diverse performance requirements. For example, user-facing applications, such as web servers and real-time (e.g. gaming) applications, often require low communication latency, whereas data-intensive applications, such as MapReduce jobs, typically desire high network throughput. In this context, it is a challenging problem to design scalable yet flexible data center networks for supporting diverse network performance objectives. Data center network virtualization is capable of overcoming these challenges by dividing a data center network into multiple logical networks that can be provisioned independently to achieve desired performance objectives. For instance, many proposed architectures, such as SecondNet and Oktopus, proposed mechanisms to allocate guaranteed bandwidth to each virtual data center. However, providing strict bandwidth guarantee can lead to low utilization if tenants do not fully utilize the allocated bandwidth. On the other hand, weighted fair-sharing based solutions, such as Seawall and Gatekeeper, are capable of achieving high resource utilization. However, they do not provide hard resource guarantees to each virtual data center. There is an inherent conflict between maximizing network utilization and providing guaranteed network performance. Designing a bandwidth allocation scheme that finds a good trade-off between these two objectives is a key research problem in virtualized data center environments.

On the other hand, existing work on data center network virtualization has been primarily focusing on bandwidth allocation for achieving predictable throughputs. The issue of providing guaranteed delay is still an open problem, as it not only requires isolated bandwidth allocation, but also effective rate control mechanisms. For example, S^3 [62] is a flow control mechanism that aims to meet flow deadlines. One particular challenge in data center environment is the TCP incast collapse problem [63], where simultaneous arrival of packets from a large number of short flows can overflow the buffer in network switches, resulting in significant increase in network delay. We believe any solution that provides delay guarantees in data center networks must also have the capability of handling TCP incast collapse. We believe the problem of providing delay guarantee in multi-tenant environments still needs further investigation.

E. Data center management

In a virtualized data center environment, infrastructure providers are responsible for managing the physical resources of the data center while service providers manage the virtual resources (e.g. computing, storage, network, I/O) allocated to their virtual data centers. An important advantage of virtualized data centers is that the physical resources are managed by a single infrastructure provider. This allows the infrastructure provider to have a full view of the system thus facilitating efficient resource allocation and handling of failures. However, there are still several challenges that need to be addressed in virtualized data centers including:

- Monitoring is a challenging task due to the large number of resources in production data centers. Centralized monitoring approaches suffer from low scalability and resilience. Cooperative monitoring [64] and gossipping [65] aim to overcome these limitations by enabling distributed and robust monitoring solutions for large scale environments. A key concern is to minimize the negative impact of management traffic on the performance of the network. At the same time, finding a scalable solution for aggregating relevant monitoring information without hurting accuracy is a challenge that needs to be tackled by monitoring tools designed for data centers. Lastly, providing customized and isolated views for individual service providers and defining the interplay between the monitoring systems of the Infrastructure providers and service providers also require further exploration.
- Efficient energy management is crucial for reducing the operational cost of a data center. One of the main challenges towards optimal energy consumption is to design energy-proportional data center architectures, where energy consumption is determined by server and network utilization [55], [66]. ElasticTree [55], for example, attempts to achieve energy proportionality by dynamically powering off switches and links. In this respect, data center network virtualization can further contribute to

reducing power consumption though network consolidation (e.g. through virtual network migration [67]). However, minimizing energy consumption can come at the price of VDC performance degradation. Thus, designing energy-proportional data center architectures factoring in network virtualization, and finding good tradeoffs between energy consumption and VDC performance are interesting research questions.

- Detection and handling of failures are fundamental requirements of any data center architecture because failures of a physical resource can potentially affect multiple service providers. Most existing architectures rely on reactive failure handing approaches. Their main drawback is the potentially long response time, which can negatively impact application performance. Ideally, fault management should be implemented in a proactive manner, where the system predicts the occurrence of failures and acts before they occur. In practise, proactive fault management is often ensured by means of redundancy, e.g., provisioning backup paths. Offering high reliability without incurring excessive costs is an interesting problem for future exploration.

F. Security

Security has always been an important issue of any network architecture. The issue is exacerbated in the context of virtualized data centers due to complex interactions between tenants and infrastructure providers, and among tenants themselves. Although the virtualization of both servers and data center networks can address some of the security challenges such as limiting information leakage, the existence of side channels and performance interference attacks, today's virtualization technologies are still far from being mature. In particular, various vulnerabilities in server virtualization technologies such as VMWare [68], Xen [69], and Microsoft Virtual PC and Virtual Server [70] have been revealed in the literature. Similar vulnerabilities are likely to occur in programmable network components as well. Thus, not only do network virtualization techniques give no guaranteed protection from existing attacks and threats to physical and virtual networks, but also lead to new security vulnerabilities. For example, an attack against a VM may lead to an attack against a hypervisor of a physical server hosting the VM, subsequent attacks against other VMs hosted on that server, and eventually, all virtual networks sharing that server [71]. This raises the issue of designing secure virtualization architectures immune to these security vulnerabilities.

In addition to mitigating security vulnerabilities related to virtualization technologies, there is a need to provide monitoring and auditing infrastructures, in order to detect malicious activities from both tenants and infrastructure providers. It is known that data center network traffic exhibits different characteristics than the traffic in traditional data networks [72]. Thus, appropriate mechanisms may be required to detect network anomalies in virtualized data center networks. On the other hand, auditability in virtualized data centers should be mutual between tenants and infrastructure providers to prevent malicious behaviors from either party. However, there is often an overhead associated with such infrastructures especially in large-scale data centers. In [73], the authors showed that it is a challenge to audit web services in cloud computing environments without deteriorating application performance. We expect the problem to be further exacerbated when extending to network activities in a VDC. Much work remains to be done on designing scalable and efficient mechanisms for monitoring and auditing virtualized data centers.

Finally, in a multi-tenant data center environment, different tenants may desire different levels of security. This introduces the additional complexity of managing heterogeneous security mechanisms and policies. Furthermore, the co-existence and interaction of multiple security systems expected in a multi-tenant data center is an issue that has not been addressed before. For example, potential conflicts between firewalls and intrusion detection systems policies of infrastructure providers and service providers, need to be detected and solved [74].

G. Pricing

Pricing is an important issue in multi-tenant data center environments, not only because it directly affects the income of the infrastructure provider, but also because it provides incentives for tenants to behave in ways that lead to desired outcomes, such as maximum resource utilization and application performance [1]. Generally speaking, a well-designed pricing scheme should be both fair and efficient. Fairness means that identical good should be sold at identical price. Efficiency means the price of the good should lead to efficient outcomes (e.g. matching supply and demand). Today, infrastructure providers promise to provide resources to tenants in an on-demand manner, and charge tenants flat-rates for both VM and network usage. Despite being fair, this simple pricing scheme still suffers from several drawbacks. First, the cost of VMs and network for a single tenant can be dependent on each other. For example, poor network performance can prolong the running time of tenant jobs, resulting in increased VM usage cost [75]. Data center network virtualization can address this problem by allocating guaranteed bandwidth for each VM [16]. For virtual data centers with best-effort connectivity, the recent proposal of dominant resource pricing (DRP) [75] seems to be a promising solution to eliminate the dependency between VM and network usage.

The second drawback of current data center pricing schemes is that they do not provide incentives for tenants to achieve desired outcomes. In particular, they do not (1) encourage purchase of resources when demand is low, and (2) suppress excessive demand (while giving priorities to important applications) when demand is high. A promising solution to this problem is to use market-driven pricing schemes, where resource price fluctuates according to supply and demand. In this perspective, Amazon EC2 spot instance service represents the first commercial endeavour towards fully market-driven pricing schemes. Similar techniques can also be used for virtualized data center networks, where resources prices for different service classes (e.g. guaranteed bandwidth, best-effort) vary according to resource demand. However, designing a market-driven resource allocation scheme that allocates multiple resource types (e.g. VMs and bandwidth) with different service quality guarantees is still a challenging problem.

While the discussion so far has been focusing on pricing resources inside data centers, the case for pricing cloud resources outside data centers is also a challenging one. In particular, when a tenant wishes to deploy a virtual data center across multiple data centers, it is necessary to develop mechanisms not only to help tenants decide appropriate embedding of VDCs across multiple networks, but also to allow both the tenant and infrastructure providers to negotiate for service quality and pricing schemes. Existing work such as V-mart [76] represents initial efforts in this direction.

VI. Conclusion

Data centers have become a cost-effective infrastructure for data storage and hosting large-scale network applications. However, traditional data center network architectures are ill-suited for future multi-tenant data center environments. Virtualization is a promising technology for designing scalable and easily deployable data centers that flexibly meet the needs of tenant applications while reducing infrastructure cost, improving management flexibility, and decreasing energy consumption.

In this paper, we surveyed the state of the art in data center network virtualization research. We discussed the proposed schemes from different perspectives, highlighting the trends researchers have been following when designing these architectures. We also identified some of the key research directions in data center network virtualization and discussed potential approaches for pursuing them.

Although current proposals improve scalability, provide mechanisms for load balancing, ensure bandwidth guarantees, there are challenging and important issues that are yet to be explored. Designing smart-edge networks, providing strict performance guarantees, devising effective business and pricing models, ensuring security and programmability, supporting multi-tiered and multi-sited data center infrastructures, implementing flexible provisioning and management interfaces between tenants and providers, and developing efficient tools for managing virtualized data centers are important directions for future research.

References

[1] Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/.

[2] D. Carr, “How Google Works,” July 2006.

[3] J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” in *Proc. USENIX OSDI*, December 2004.

[4] WMware. http://www.vmware.com.

[5] Xen. http://xen.org.

[6] A. Shieh, S. Kandulaz, A. Greenberg, C. Kim, and B. Saha, “Sharing the Data Center Network,” in *Proc. USENIX NSDI*, March 2011.

[7] T. Benson, A. Akella, A. Shaikh, and S. Sahu, “CloudNaaS: A Cloud Networking Platform for Enterprise Applications,” in *Proc. ACM SOCC*, June 2011.

[8] M. Chowdhury and R. Boutaba, “A Survey of Network Virtualization,” *Computer Networks*, vol. 54, no. 5, pp. 862–876, 2010.

[9] “Data Center: Load Balancing Data Center Services SRND,” 2004.

[10] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., 2004.

[11] C. Leiserson, “Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing,” *IEEE Trans. Comput.*, vol. 34, no. 10, pp. 892–901, 1985.

[12] M. Al-Fares, A. Loukissas, and A. Vahdat, “A Scalable, Commodity Data Center Network Architecture,” in *Proc. ACM SIGCOMM*, August 2008.

[13] 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,” in *Proc. ACM SIGCOMM*, August 2009.

[14] L. Popa, S. Ratnasamy, G. Iannaccone, A. Krishnamurthy, and I. Stoica, “A Cost Comparison of Datacenter Network Architectures,” in *Proc. ACM CoNext*, November 2010.

[15] C. Guo, G. Lu, H. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, “SecondNet: A Data Center Network Virtualization Architecture with Bandwidth Guarantees,” in *Proc. ACM CoNEXT*, December 2010.

[16] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, “Towards Predictable Datacenter Networks,” in *Proc. ACM SIGCOMM*, August 2011.

[17] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, “Measuring ISP Topologies with Rocketfuel,” *IEEE/ACM Trans. Netw.*, vol. 12, no. 1, pp. 2–16, 2004.

[18] Q. Zhang, M. F. Zhani, Q. Zhu, S. Zhang, R. Boutaba, and J. Hellerstein, “Dynamic Energy-Aware Capacity Provisioning for Cloud Computing Environments,” in *Proc. IEEE/ACM International Conference on Autonomic Computing (ICAC)*, September 2012.

[19] “IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks,” IEEE Std 802.1Q-2005, May 2006.

[20] J. Mudigonda, P. Yalagandula, M. Al-Fares, and J. Mogul, “SPAIN:COTS Data-Center Ethernet for Multipathing over Arbitrary Topologies,” in *Proc. ACM USENIX NSDI*, April 2010.

[21] A. Edwards, F. A, and A. Lain, “Diverter: A New Approach to Networking Within Virtualized Infrastructures,” in *Proc. ACM WREN*, August 2009.

[22] J. Mudigonda, P. Yalagandula, B. Stiekes, and Y. Pouffary, “NetLord: A Scalable Multi-Tenant Network Architecture for Virtualized Datacenters,” in *Proc. ACM SIGCOMM*, August 2011.

[23] F. Hao, T. Lakshman, S. Mukherjee, and H. Song, “Enhancing Dynamic Cloud-based Services using Network Virtualization,” in *Proc. ACM VISA*, August 2009.

[24] A. Greenberg, J. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. Maltz, P. Patel, and S. Sengupta, “VL2: A Scalable and Flexible Data Center Network,” in *Proc. ACM SIGCOMM*, August 2009.

[25] R. 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,” in *Proc. ACM SIGCOMM*, August 2009.

[26] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. Guedes, “Gatekeeper: Supporting Bandwidth Guarantees for Multi-tenant Datacenter Networks,” in *Proc. WIOV*, June 2011.

[27] T. Lam, S. Radhakrishnan, A. Vahdat, and G. Varghese, “NetShare: Virtualizing Data Center Networks across Services,” Technical Report CS2010-0957, May 2010.

[28] F. Hao, T. Lakshman, S. Mukherjee, and H. Song, “Secure Cloud Computing with a Virtualized Network Infrastructure,” in *Proc. USENIX HotCloud*, June 2010.

[29] C. Hopps, “Analysis of an Equal-Cost Multi-Path Algorithm,” IETF RFC 2992, November 2000.

[30] R. Zhang-Shen and N. McKeown, “Designing a Predictable Internet Backbone Network,” in *Proc. ACM HotNets*, November 2004.

[31] ——, “Designing a Predictable Internet Backbone with Valiant Load-Balancing,” in *Proc. IWQoS*, June 2005.

[32] N. Leavitt, “Is Cloud Computing Really Ready for Prime Time?” *Computer*, vol. 42, no. 1, pp. 15–20, January 2009.

[33] http://www.cisco.com/en/US/prod/collateral/switches/ps5718/ps6545/product_data_sheet0900aecd80322aeb.html.

[34] E. Rosen and Y. Rekhter, “BGP/MPLS IP Virtual Private Networks (VPNs),” IETF RFC 4364, February 2006.

[35] R. Perlman, “An Algorithm for Distributed Computation of a Spanning Tree in an Extended LAN,” *ACM Computer Communication Review*, vol. 15, no. 4, pp. 44–53, September 1985.

[36] E. Rosen, A. Viswanathan, and R. Callon, “Multiprotocol Label Switching Architecture,” IETF RFC 3031, January 2001.

[37] M. Shreedhar and G. Varghese, “Efficient Fair Queuing Using Deficit Round-Robin,” *IEEE/ACM Trans. Netw.*, vol. 4, no. 3, pp. 375–385, 1996.

[38] “IEEE Std 802.1D-2004, IEEE Standard for Local and Metropolitan Area Networks, Media Access Control (MAC) Bridges.” 2004.

[39] A. Tootoonchian and Y. Ganjalir, “HyperFlow: a Distributed Control Plane for OpenFlow,” in *Proc. NSDI INM/WREN*, April 2010.

[40] Í. Goiri, K. Le, J. Guitart, J. Torres, and R. Bianchini, “Intelligent Placement of Datacenters for Internet Services,” in *Proc. IEEE ICDCS*, June 2011.

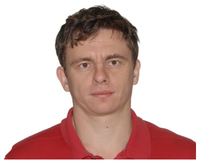
[41] B. Ahlgren, P. Aranda, P. Chemouil, S. Oueslati, L. Correia, H. Karl, M. Söllner, and A. Welin, "Content, Connectivity, and Cloud: Ingredients for the Network of the Future," *IEEE Commun. Mag.*, vol. 49, no. 7, pp. 62–70, July 2011.
[42] S. Islam and J.-C. Gregoire, "Network Edge Intelligence for the Emerging Next-Generation Internet," *Future Internet*, vol. 2, no. 4, pp. 603–623, December 2010.
[43] k. Church, A. Greenberg, and J. Hamilton, "On Delivering Embarrassingly Distributed Cloud Services," in *Proc. ACM HotNets*, October 2008.
[44] V. Valancius, N. Laoutaris, C. Diot, P. Rodriguez, and L. Massoulié, "Greening the Internet with Nano Data Centers," in *Proc. ACM CoNEXT*, December 2009.
[45] M. B. Mobley, T. Stuart, and Y. Andrew, "Next-Generation Managed Services: A Window of Opportunity for Service Providers," CISCO Technical Report, 2009.
[46] D. Oppenheimer, B. Chun, D. Patterson, A. Snoeren, and A. Vahdat, "Service Placement in a Shared Wide-Area Platform," in *Proc. USENIX ATEC*, June 2006.
[47] L. Qiu, V. Padmanabhan, and G. Voelker, "On the Placement of Web Server Replicas," in *Proc. IEEE INFOCOM*, April 2001.
[48] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration," *ACM Computer Communication Review*, vol. 38, no. 2, pp. 17–29, April 2008.
[49] M. Chowdhury, M. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," in *Proc. INFOCOM*, April 2009.
[50] M. R. Rahman, I. Aib, and R. Boutaba, "Survivable Virtual Network Embedding," in *Proc. IFIP Networking*, May 2010.
[51] N. F. Butt, N. M. M. K. Chowdhury, and R. Boutaba, "Topology-Awareness and Reoptimization Mechanism for Virtual Network Embedding," in *Proc. IFIP Networking*, May 2010.
[52] Energy Efficiency and Sustainability of Data Centers. http://www.sigmetrics.org/sigmetrics2011/greenmetrics/Carey_GreenMetricsKeynote060711.pdf.
[53] Energy Logic: Reducing Data Center Energy Consumption by Creating Savings that Cascade Across Systems. http://www.cisco.com/web/partners/downloads/765/other/Energy_Logic_Reducing_Data_Center_Energy_Consumption.pdf.
[54] A. Greenberg, J. Hamilton, D. Maltz, and P. Patel, "The Cost of a Cloud: Research Problems in Data Center Networks," *ACM Computer Communication Review*, vol. 39, no. 1, pp. 68–73, 2009.
[55] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "ElasticTree: Saving Energy in Data Center Networks," in *Proc. USENIX NSDI*, April 2010.
[56] M. Rahman, I. Aib, and R. Boutaba, "Survivable Virtual Network Embedding," *NETWORKING 2010*, pp. 40–52, 2010.
[57] M. Chowdhury and R. Boutaba, "PolyViNE," in *Proc. ACM VISA*, August 2010.
[58] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *ACM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, April 2008.
[59] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Can the Production Network Be the Test-bed?" in *Proc. USENIX OSDI*, October 2010.
[60] A. Tavakoli, M. Casado, T. Koponen, and S. Shenker, "Applying NOX to the Datacenter," in *Proc. ACM HotNets*, August 2009.
[61] A. Curtis, J. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjeer, "DevoFlow: Scaling Flow Management for High-Performance Network," in *Proc. ACM SIGCOMM*, August 2011.
[62] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowstron, "Better never than Late: Meeting Deadlines in Datacenter Networks," in *Proc. ACM SIGCOMM*, August 2011.
[63] H. Wu, Z. Feng, C. Guo, and Y. Zhang, "ICTCP: Incast Congestion Control for TCP," in *Proc. ACM CoNEXT*, November 2010.
[64] K. Xu and F. Wang, "Cooperative Monitoring for Internet Data Centers," in *Proc. IEEE IPCCC*, December 2008.
[65] F. Wuhib, M. Dam, R. Stadler, and A. Clemm, "Robust Monitoring of Network-wide Aggregates through Gossiping," *IEEE Trans. Network Service Management*, vol. 6, no. 2, pp. 95–109, 2009.
[66] H. Yuan, C. C. J. Kuo, and I. Ahmad, "Energy Efficiency in Data Centers and Cloud-based Multimedia Services: An Overview and Future Directions," in *Proc. IGCC*, August 2010.
[67] Y. Wang, E. Keller, B. Biskeborn, J. van der Merwe, and J. Rexford, "Virtual Routers on the Move: Live Router Migration as a Network-Management Primitive," *ACM Computer Communication Review*, vol. 38, pp. 231–242, August 2008.
[68] VMWare vulnerability. http://securitytracker.com/alerts/2008/Feb/1019493.html.
[69] Xen vulnerability. http://secunia.com/advisories/26986.
[70] Virtual PC vulnerability. http://technet.microsoft.com/en-us/security/bulletin/MS07-049.
[71] J. Szefer, E. Keller, R. Lee, and J. Rexford, "Eliminating the Hypervisor Attack Surface for a More Secure Cloud," in *Proc. ACM CSS*, October 2011.
[72] T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding Data Center Traffic Characteristics," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 92–99, 2010.
[73] A. Chukavkin and G. Peterson, "Logging in the Age of Web Services," *IEEE Security and Privacy*, vol. 7, no. 3, pp. 82–85, June 2009.
[74] E. Al-Shaer, H. Hamed, R. Boutaba, and M. Hasan, "Conflict Classification and Analysis of Distributed Firewall Policies," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 10, pp. 2069–2084, 2005.
[75] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "The Price Is Right: Towards Location-Independent Costs in Datacenters," 2011.
[76] F.-E. Zaheer, J. Xiao, and R. Boutaba, "Multi-Provider Service Negotiation and Contracting in Network Virtualization," in *Proc. IEEE/IFIP NOMS*, April 2010.

Md. Faizul Bari is a Ph.D. student at the School of Computer Science at the University of Waterloo. He received M.Sc. and B.Sc. degrees in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET) in 2009 and 2007, respectively. He has served as a reviewer for many international conferences and journals. His research interests include future Internet architecture, network virtualization, and cloud computing. He is the recipient of the Ontario Graduate Scholarship, Presidents Graduate Scholarship, and David R. Cheriton Graduate Scholarship at University of Waterloo. He also received Merit Scholarship and Dean's award at BUET.

Raouf Boutaba received the M.Sc. and Ph.D. degrees in computer science from the University Pierre & Marie Curie, Paris, in 1990 and 1994, respectively. He is currently a professor of computer science at the University of Waterloo and a distinguished visiting professor at the division of IT convergence engineering at POSTECH. His research interests include network, resource and service management in wired and wireless networks. He is the founding editor in chief of the IEEE Trans. Network and Service Management (2007-2010) and on the editorial boards of other journals. He has received several best paper awards and other recognitions such as the Premiers Research Excellence Award, the IEEE Hal Sobol Award in 2007, the Fred W. Ellersick Prize in 2008, and the Joe LociCero and the Dan Stokesbury awards in 2009. He is a fellow of the IEEE.

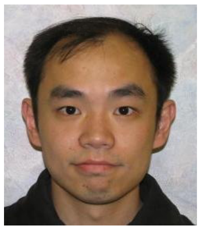
Rafael Pereira Esteves is a Ph.D. student in computer science at the Institute of Informatics of the Federal University of Rio Grande do Sul (UFRGS), Brazil. He received his M.Sc. (2009) and B.Sc. (2007) degrees from the Department of Informatics of the Federal University of Pará, Brazil. From May 2011 to April 2012 he was a visiting student at the David R. Cheriton School of Computer Science of the University of Waterloo, Canada, with the Network and Distributed Systems Group. His research interests include network management, network virtualization, cloud computing, and Future Internet.

Lisandro Zambenedetti Granville received the MSc and PhD degrees in computer science from the Institute of Informatics (INF) of the Federal University of Rio Grande do Sul (UFRGS), Brazil, in 1998 and 2001, respectively. Currently, he is a professor at INF-UFRGS. Lisandro is co-chair of the Network Management Research Group (NMRG) of the Internet Research Task Force (IRTF) and vice-chair of the Committee on Network Operations and Management (CNOM) of the IEEE Communications Society (COMSOC). He was also technical program committee co-chair of the 12th IEEE/IFIP Network Operations and Management Symposium (NOMS 2010) and 18th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2007). His research interests include network and services management, software defined network, network virtualization, and information visualization.



Maxim Podlesny received degrees of Bachelor of Science and Master of Science in Applied Physics and Mathematics from Moscow Institute of Physics and Technology, Russia in 2000 and 2002 respectively and the degree of Ph.D. in Computer Science from Washington University in St. Louis, USA in 2009. Dr. Podlesny currently works as a Postdoctoral Fellow at the University of Waterloo, Canada. His work has appeared in top conferences and journals such as ACM SIGCOMM, IEEE INFOCOM, IEEE Journal on Selected Areas in Communications, and ACM Computer Communication Review. His research interests are network congestion control, service differentiation, Quality-of-Service, transport protocols, data centers, and network virtualization.

Md Golam Rabbani received the B.Sc. degree in Computer Science and Engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2007, and the Master of Information Technology degree from Monash University, Australia, in 2011. He worked as a system engineer in a telecommunication company from 2007 to 2009. He is currently pursuing his M.Math. degree in Computer Science at University of Waterloo, under the supervision of Prof. Raouf Boutaba. His research interests include data center, cloud computing, future Internet architecture, and wireless communication.



Qi Zhang received his B. A. Sc. and M. Sc. from University of Ottawa (Canada) and Queen's University (Canada), respectively. He is currently pursuing a Ph. D. degree in Computer Science from University of Waterloo. His current research focuses on resource management for cloud computing systems. He is also interested in related areas including network virtualization and management.

Mohamed Faten Zhani received Engineering and M.S. degrees from the National school of computer science, Tunisia in 2003 and 2005, respectively. Then he received his PhD from the University of Quebec In Montreal, Canada in 2011. Since then, he has been a postdoctoral research fellow at the University of Waterloo (Ontario, Canada). His research interests are in the areas of network performance evaluation, resource management in virtualization-based environments and Cloud Computing.