Received September 14, 2015, accepted October 2, 2015, date of publication December 9, 2015,
date of current version December 16, 2015.

Digital Object Identifier 10.1109/ACCESS.2015.2499271

Software-Defined Network Function Virtualization: A Survey

YONG LI[1], (Member, IEEE), AND MIN CHEN[2], (Senior Member, IEEE)

[1]State Key Laboratory on Microwave and Digital Communications, Tsinghua National Laboratory for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

[2]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

Corresponding author: M. Chen (minchen@ieee.org)

This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2013CB329105, in part by the National Natural Science Foundation of China under Grant 61300224, Grant 61301080, Grant 61171065, Grant 61273214, Grant 91338203, and Grant 91338102, and in part by the International Science and Technology Collaboration Program under Grant 2014DFT10070 through the Ministry of Science and Technology, China, and National Natural Science Foundation of China under Grant 61572220.

ABSTRACT Diverse proprietary network appliances increase both the capital and operational expense of service providers, meanwhile causing problems of network ossification. Network function virtualization (NFV) is proposed to address these issues by implementing network functions as pure software on commodity and general hardware. NFV allows flexible provisioning, deployment, and centralized management of virtual network functions. Integrated with SDN, the software-defined NFV architecture further offers agile traffic steering and joint optimization of network functions and resources. This architecture benefits a wide range of applications (e.g., service chaining) and is becoming the dominant form of NFV. In this survey, we present a thorough investigation of the development of NFV under the software-defined NFV architecture, with an emphasis on service chaining as its application. We first introduce the software-defined NFV architecture as the state of the art of NFV and present relationships between NFV and SDN. Then, we provide a historic view of the involvement from middlebox to NFV. Finally, we introduce significant challenges and relevant solutions of NFV, and discuss its future research directions by different application domains.

INDEX TERMS Software-defined networks, network function virtualization, middlebox, service chain, network virtualization.

I. INTRODUCTION

Current network services rely on proprietary appliances and different network devices that are diverse and purpose-built [1]–[3]. This situation induces the so-called network ossification problem, which prevents the operation of service additions and network upgrades. To address this issue and reduce capital expenditures (CapEx) and operating expenditures (OpEx), virtualization has emerged as an approach to decouple the software networking processing and applications from their supported hardware and allow network services to be implemented as software [4]–[6]. Leveraging virtualization technologies, ETSI Industry Specification Group proposed Network Functions Virtualization (NFV) to virtualize the network functions that are previously carried out by some proprietary dedicated hardware [7], [8]. By decoupling the network functions from the underlaying hardware appliances, NFV provides flexible provisioning of software-based network functionalities on top of an optimally shared physical infrastructure. It addresses the problems of operational costs of managing and controlling these closed and proprietary appliances by leveraging low cost commodity servers.

On the other hand, with the development of Software-Defined Networking (SDN) and as more abstractions are introduced into network architectures [9]–[11], the trend of integrating SDN with NFV (the software-defined NFV architecture) to achieve various network control and management goals has seen an noticeable growth. SDN when applied to NFV can help in addressing the challenges of dynamic resource management and intelligent service orchestration. Through NFV, SDN is able to create a virtual service environment dynamically for a specific type of service chain,

consequently the dedicated hardware and complex labor work to provide a new coming service request is avoid. In conjunction with the use of SDN, NFV further enables real-time and dynamic function provisioning along with flexible traffic forwarding.

Software-defined NFV leverages network virtualization and logically centralized intelligence to minimize the service providing cost and maximize the utilization of network resource. In this case, the obtained higher resource utilization will introduce less investigation on the hardware equipments, which on the other hand simplifies networking operations. Moreover, by automating current manually intensive network configuration, provisioning, and management, the time and operation complexity are significantly reduced and manual errors are dramatically decreased, which offers better scalability. On the other hand, especially in large-scale networks, deploying and providing a new kinds of service usually results in a long and repeated process that requires long cycles of validation, verifying, and testing. By automating the control, managing and orchestration of the NFV related infrastructure, the deploying time and operation cost for network configuration and operation changes for these new services will be significantly shortened.

Service chaining is the main area that software-defined NFV can play an important role [12], [13]. In the current networks, a service chain include a set of hardware dedicated network appliances offering services such as load balancers, firewall, Deep Packet Inspection (DPI), Intrusion Detection System (IDS), and etc., to support a dedicate networking processing and applications [14]–[16]. When it comes a new service requirement, new hardware devices must be deployed, installed and connected by some order, which is extremely time-consuming, complexity, high-cost and error-prone. This kind of networking service providing requires dedicate plan of networking changes and outages, which on the other hand incurs high OPEX. This situation is exacerbated when a lot of different kinds of service sequences are dedicated to different traffic flows by an operator. On the other hand, the architecture of software-defined NFV is able to simplify the service chain deployment and provisioning. It enables easier and cheaper service providing in the local area network, enterprise networks, data center, and Interent service provider networks.

This survey introduces the state-of-the-art of NFV and its main challenges within the software-defined NFV architecture. Service chaining is highlighted and discussed as a core application of NFV in different contexts. We further provide guidelines for future developments of NFV in various application scenarios. In Section II, we introduce the software-defined NFV architecture as the state-of-the-art of NFV and present relationships between NFV and SDN. Then, we provide a historic view of the involvement from middlebox to NFV in Section III. After survey the current technology of service chain in Section IV, we introduce significant challenges and relevant solutions of NFV in Section V, and discuss its future research directions by different application domains in Section VI. Finally, we conclude the paper in Section VII.

II. SOFTWARE-DEFINED NETWORK FUNCTION VIRTUALIZATION

To reduce CapEx and OpEx introduced by diverse proprietary appliances, NFV was proposed to exploit and take advantage of the virtualization technology. NFV allows network operators and service providers to implement network functions in software, leveraging standard servers and virtualization technologies, instead of run on purpose-built hardware. Recent trends of increased user information demands, explosion of traffic and diverse service requirements further drive NFV to be integrated with SDN, forming the software-defined NFV architecture. This architecture offers great flexibility, programmability and automation to the operators in service provisioning and service model.

A. NETWORK FUNCTION VIRTUALIZATION

Diverse and fixed proprietary appliances make the service deployment and testing increasingly difficult. NFV was proposed as a key technology to benefit IT virtualization evolution [4]–[6] by separating the hardware network functions from the underlying hardware appliances by transferring network functions from dedicated hardware to general software running on commercial off-the-shelf (COTS) equipments, i.e., virtual machines [17]–[20]. These software applications are running on standard IT platforms like high-performance switches, service, and storage. By NFV, the different network functions can be deployed in different locations of the networks such as data-centers, network nodes, and end-node of network edge as required. Currently, the market of NFV includes switching elements, network appliances, network services and applications. Here we summary the commonly used network functions considered for NFV [7], [21].

- Network switching elements [22], i.e., Broadband Network Gateway (BNG), carrier grade NAT, Broadband remote access server (BRAS), and routers.
- Mobile network devices, i.e., Home Location Register/ Home Subscriber Server (HLR/HSS), Serving GPRS Support NodeMobility Management Entity (SGSNMME), Gateway support node/Packet Data Network Gateway (GGSN/PDN-GW), RNC, NodeB and Evolved Node B (eNodeB) [23].
- Virtualized home environments [24], [25].
- Tunneling gateway devices, i.e., IPSec/SSL virtual private network gateways.
- Traffic analysis elements, i.e., Deep Packet Inspection (DPI), Quality of Experience (QoE) measurement.
- Service Assurance, Service Level Agreement (SLA) [26] monitoring, Test and Diagnostics.
- Next-Generation Networks (NGN) signaling such as Session Border Controller (SBCs), IP Multimedia Sub-system (IMS).

- Application-level optimization devices, i.e., Content Delivery Network (CDNs) [27], load balancers, cache nodes, and application accelerators.
- Network security devices, i.e., Firewalls [28], intrusion detection systems, DOS attack detector, virus scanners, spam protection, etc.

The major advantage of using NFV is to reduce middleboxes deployed in the traditional networks to take the advantages of cost savings and bring flexibility. On the other side, NFV technology also supports the co-exists of multi-tenancy of network and service functions, through allowing the usage of one physical platform for different services, applications, and tenants.

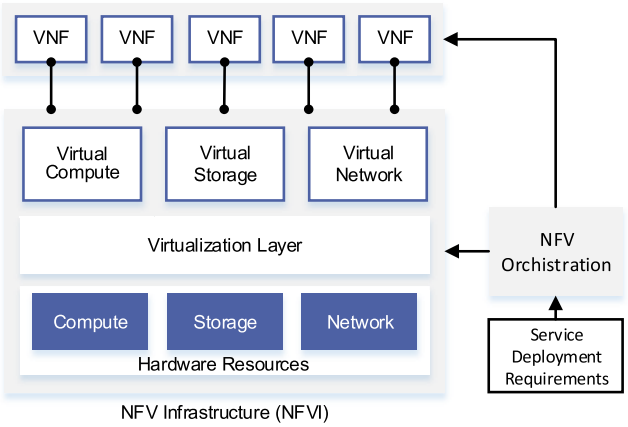

FIGURE 1. Illustration of the NFV framework.

B. NFV FRAMEWORK

ETSI defines the NFV architectural framework (showing in Fig. 1) enabling virtualized network functions (VNF) to be deployed and executed on a Network Functions Virtualisation Infrastructure (NFVI), which consists of commodity servers wrapped with a software layer that abstracts and logically partitions them [7], [29]. Above the hypervisor layer, a VNF is typically mapped to one VM in the NFVI. The deployment, execution and operation of VNFs on the NFVI are steered by a Management and Orchestration (M&O) system [30], whose behaviour is driven by a set of metadata describing the characteristics of the network services and their constituent VNFs. The M&O system includes an NFV Orchestrator in charge of the lifecycle of network services, a set of VNF managers in charge of the lifecycle of the VNFs and a virtualized infrastructure manager, which can be viewed as an extended cloud management system responsible for controlling and managing NFVI resources [31], [32].

C. SOFTWARE-DEFINED NETWORKS

Software-Defined Network (SDN) is an important and recently emerging network architecture to decouple the network control from the data forwarding by directly programming [33]–[36]. With its inherent decoupling of control plane from data plane, SDN offers a greater control of a network through programming [37], [38]. This combined

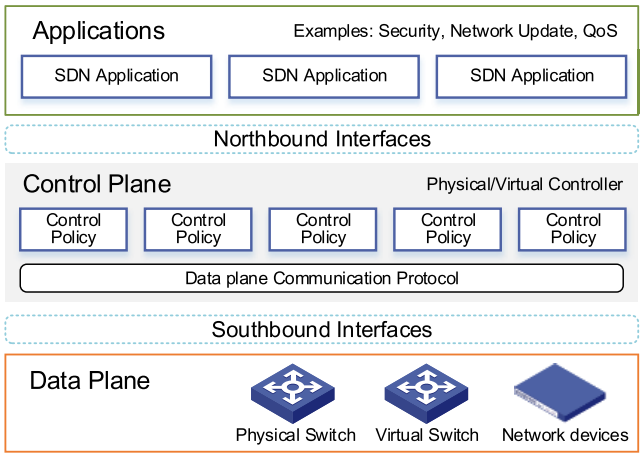

FIGURE 2. SDN architecture.

feature would bring potential benefits of enhanced configuration, improved performance, and encouraged innovation in network architecture and operations. Especially, SDN offers a promising alternative for traffic steering by programmatically configuring forwarding rules [39]. Fig. 2 depicts the SDN architecture [21], [40], [41]. There are three different layers:

- Application Layer: This layer covers an array of applications focusing on network services, and they are mainly software applications communicating with the control layer.
- Control Layer [42]–[44]: As the core of SDN, the control layer consists of a centralized controller, which logically maintains a global and dynamic network view, takes requests from the application layer, and manages the network devices via standard protocols.
- Data-plane Layer: Infrastructure including switches, routers and network appliances. In SDN context, these devices are programmable and support standard interfaces [45].

The application layer utilizes the northbound APIs to communicate with the SDN controller, which enable different control mechanisms for the networks. The southbound APIs define the communication interface between the controller layer and data plane devices, which on the other hand enable the application to control the forwarding device via this flexible and programmable way.

D. NFV V.S. SDN

NFV and SDN are closely related and highly complementary to each other. NFV can serve SDN by virtualizing the SDN controller (which can be regarded as a network function) to run on cloud, thus allows dynamic migration of the controllers to the optimal locations. In turn, SDN serves NFV by providing programmable network connectivity between VNFs to achieve optimized traffic engineering and steering [29], [46]. However, NFV and SDN are completely different from the concepts to the system architecture and functions, which are summarized by the following aspects:

- NFV is a concept of implementing network functions in software manner, while SDN is concept of achieving centrally controlled and programmable network architecture to provide better connectivity.
- NFV aims at reducing CapEx, OpEx, and space and power consumption, while SDN aims at providing network abstractions to enable flexible network control, configuration and fast innovation.
- NFV decouples the network functions from the proprietary hardware to achieve agile provisioning and deployment, while SDN decouples the network control plane from the data plane forwarding to provide a centralized controller via enabling programmability.

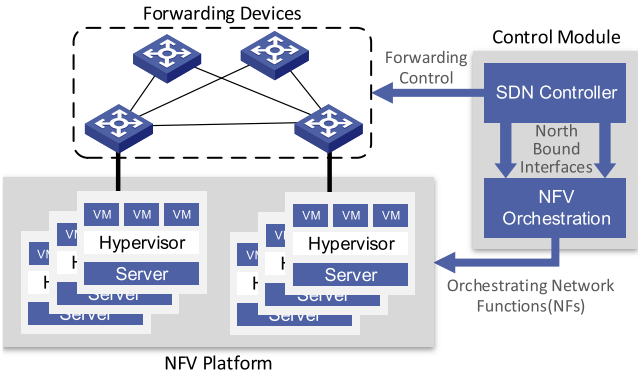

FIGURE 3. Software-defined NFV system.

E. SOFTWARE-DEFINED NFV ARCHITECTURE

The software-defined NFV system is illustrated in Fig. 3. It consists a control module, forwarding devices and NFV platform at the edge of the network. The logic of packet forwarding is determined by the SDN controller and is implemented in the forwarding devices through forwarding tables. Efficient protocols, e.g., Openflow [47]–[51]), can be utilized as standardized interfaces in communicating between the centralized controller and distributed forwarding devices. The NFV platform leverages commodity servers to implement high bandwidth NFs at low cost. Hypervisors run on the servers to support the VMs that implement the NFs. This platform allows customizable and programmable data plane processing functions such as middlebox of firewalls, IDSes, proxies, which are running as software within virtual machines, where NFs are delivered to the network operator as pieces of pure software.

The SDN controller [43], [44], [52], [53] and the NFV orchestration system compose the logical control module. The NFV orchestration system is in the charge of provisioning for virtualized network functions, and is controlled by the SDN controller through standard interfaces. After obtain the network topology and policy requirements, the control module computes the optimal function assignments (assigning network functions to certain VMs) and translates the the logic policy specifications into optimized routing paths. The function assignments are enforced by the NFV orchestration system and the controller steer the traffic traveling through the required and appropriate sequence of VMs and forwarding devices by installing forwarding rules into them.

III. FROM MIDDLEBOX TO NFV

While NFV receives a large amount of attentions from both the industry and academic world, the idea of decoupling the software layer and the underlying hardware has been around for many years. Though NFV does not limit in virtualizing middleboxes, the concept of NFV was initiated in the context of middlebox. In this section, we introduce the evolution from traditional purpose-built middlebox to NFV, during which consolidated middlebox and software-defined middlebox acted as transitional paradigms.

A. MIDDLEBOX OVERVIEW

A middlebox, also named network appliance, is a networking forwarding or processing device that transmits, transforms, filters, inspects, or control network traffic for purposes of network control and management [2], [54]–[56]. A middlebox service or function is a method or operation performed by a network device that needs specific intelligence about the applications. Typical examples of middleboxes, i.e., network appliance, include network address translators that modify packets' destination and source addresses, and firewalls that filter unwanted or malicious traffic, and The following are commonly deployed middleboxes [57]:

1) Network Address Translator (NAT) [58]: NAT is utilized to replace the source and/or destination IP addresses of certain packets that traverse over it. Typically, NAT is deployed to share a single IP address by multiple end hosts, i.e., computers: hosts ''behind'' the NAT are assigned a private IP address, and their packets destined to Internet will traverse the NAT middlebox that replaces their private address with the public address to communicate with the public Internet.
2) Firewall (FW) [28]: Firewall is utilized to filter traffic according to a set of pre-defined security policies by rejecting packets with specific fields headers of the IP and transport, or using more complex policies of inspecting packets at the application and session layer.
3) Intrusion Detection System (IDS) [59]: IDS is utilized to monitoring the network to detect security anomalies. Since it does not filter data in real-time, they usually are capable of more complex packet processing than hte middlebox of firewalls that need to made the accept/reject decision when the packet arrives.
4) Load Balancer (LB) [60]: The middlebox of network load balancer is to split network traffic across multiple different servers, with the aims of optimizing resource use, minimizing network response time, maximizing system throughput, and avoiding overload of other resource.

5) WAN Optimizer: WAN Optimizer improves bandwidth consumption and shorten network transmission latency between different endpoint in the WAN. Typically, they are deployed near the sending or receiving communication host, and then cache and compress traffic passing by.
6) Flow Monitor (FM): The middlebox of flow monitor is utilized to collect information of the flows in the network for the utilization of traffic analysis or trouble shooting. It is widely utilized in the data center or service providers' networks.

B. CONSOLIDATED MIDDLEBOX

Traditionally, a new type of middlebox was usually emerging as a solution for some specific need, then integrated into the network of infrastructure by the widely deployment. This deployment approach leads to significant inefficiency in the use and management of infrastructure hardware resources. Prior to NFV, researchers turned to the age-old idea of consolidation to address the above challenges by systematically re-architecting middlebox infrastructure to exploit opportunities for consolidation [1], [61]–[63]. Now, we provide an overview for the efforts on consolidating middleboxes, which are precursors to the current NFV paradigm.

1) CoMb [61]

To address the important resource management and controlling problems that arise in exploiting the benefits of middlebox deployment, CoMb is proposed by consolidating individual middleboxes through decoupling the software and hardware, which enables software-based implementations of middlebox to deploy and run on a the general and consolidated hardware platform. On the other hand, CoMb consolidates the management of different middlebox into a single centralized controller, which takes a unified and network-wide configurations and controlling for policy requirements across the overall traffic and applications. This is in contrast to today's approach where the middleboxs is controlled and managed separately. CoMb addressed these important resource control and management challenges, which results in reducing network provisioning cost and overhead in the deployment and operation of middlebox devices.

2) APLOMB [1], [62]

APLOMB is proposed to enable the traffic processing in the third-party middlebox device and service providers running in the data centers and cloud. APLOMB allows enterprise networks, as well as individual end hosts, to tunnel their traffic to and from a cloud service, which applies middlebox processing to their traffic. In this way, it avoids the costly and management cost of administering middleboxes in a local-region network.

3) INTEGRATE MIDDLEBOXES INTO NETWORK [63]

There has been a trend to reduce the middleboxes by deploying the network services and related processing into the network forwarding devices like switch/router's computing modules or separate server and machines. Following such idea, [63] is proposed to remove the dedicated hardware middleboxes and move the related network processing services on network platform and standard servers. In order to provide efficient in-network services on top of various processing modules in the network devices, they proposed a flexible control system that integrate the network processing modules and forwarding devices in an automated way.

C. SOFTWARE-DEFINED MIDDLEBOX

As SDN evolves, the principles of abstracting the architecture of network from the control and data plane have been investigated in various contexts. This idea introduces some unique opportunities for the development of middleboxes [64]. Inspired by the idea of SDN, some researchers proposed a software-defined middlebox and corresponding networking architecture, with the aim of providing fine-grained and programmable control over the Middlebox state and network forwarding. Now, we summary an overview of the software-defined middleboxes.

1) ENABLING MIDDLEBOX INNOVATION [56]

Ref. [56] is an early effort on designing software-centric middlebox, which runs on general-purpose hardware platforms controlled and managed through open APIs. A research agenda is proposed with the target of manage a single or an ensemble of middleboxes. To enable fast middlebox innovation, this work explore an approach through three different strategies: software-centric implementations of middlebox that decouple hardware from the software; multiple software-based middlebox are implemented on a shared general hardware platform; and, finally centralized controlling and management with open APIs to provide, control and manage the deployment of the middlebox.

2) OpenMB [65]

OpenMB consists of somehow modified middleboxes by exposing a southbound API for importing/exporting the complicate states of middlebox, where the centralized controller implements the open API to define how state can be set and accessed. OpenMB-enabled middleboxes allow a variety of dynamic scenarios to be realized without influence on the correctness or performance of middleboxes, which is crucial to continued innovation in software-defined middlebox.

3) xOMB [66]

xOMB (Extensible Open MiddleBox) provides programmable, flexible and scalable middleboxes on the platform of general hardware like servers and operating systems to achieve high efficiency flow controlling. It utilize general programmable processing approaches with user-defined modules for network packet parsing, data transforming, and flow forwarding. By these design, xOMB shows how middleboxes can be utilized to support different services.

IEEE Access
The journal for rapid open access publishing

IV. SERVICE CHAINING

Service chaining is an important model for network service providers, in which NFV plays an important role. It is utilized to organize the service function deployment, where the ability of specifying an ordered list of service processing for the service's traffic flows [67] is provided. A chain defines the required processing or functions and the corresponding order that should be applied to the data flow. These chains require integration of service policy and the above applications to achieve optimal resource utilization.

Traditional service chaining mainly rely on manual configuration which is tedious, error-prone and clumsy. SDN provides new capability steer traffic dynamically based on user requirements. However, hardware-based middleboxes limit the benefit of SDN due to their fixed functionalities and deployment. NFV is a good enabler for SDN. With the ability of dynamic function provisioning offered by NFV and the centralized control of SDN, new opportunities emerge in service chaining. Better performance and resource utilization can be achieved with the software-defined NFV architecture.

A. SDN&MIDDLEBOX BASED SERVICE CHAINING

SDN offers the flexible control approach and enables dynamic traffic forwarding, and these style of traffic control for middlebox-specific flow can realize flexible and efficient service chaining with no need to generate any placement or introduce some constraints on middleboxes, which are on the other hand easily supported by current SDN standards [73]. Three are some important works in this topic, which are introduced below.

1) SYMPLE [74]

SYMPLE (Software-defIned Middlebox PoLicy Enforcement) is a software-defined policy enforcement layer for traffic steering. It enables the network managers and operators to specify a high-level abstractions of logical middlebox routing policy, and it then further automatically translates the policy into control rules with the knowledge of the physical network topology, forwarding device capacities, and resource constraints of the whole networks. Without modifying any middleboxes and network devices, SYMPLE offers efficient data plane for packet processing, and automatically dealing with specifiable packet modifications, which is more modest compared to ongoing and parallel work developing new visions for SDN or middleboxes.

2) StEERING [75]

StEERING, short for SDN inlinE sERvices and forwardiNG, is a scalable framework for dynamically routing traffic through any sequence of middleboxes. With simple centralized configuration, StEERING can explicitly steer different types of flows through the desired set of middleboxes, scaling at the level of per-subscriber and per-application policies. Built on top of SDN, StEERING can support efficient forwarding for a large number of applications and subscribers.

3) FLOWTAG [76]

The dynamic, traffic-dependent, and hidden actions of middleboxes make it hard to systematically enforce and verify network-wide policies, and to do network diagnosis. Flowtag is a complement for SDN based service chaning approaches, dealing with the dynamic changes imposed by middleboxes. FlowTags-enhanced middleboxes export tags to provide the required network context. On the other hand, the SDN controllers is able to configure the operations of tag generation and consumption by the FlowTags APIs. These operations benefit restore bindings between packets and their origins, and guarantee that packets of flow follow policy-required paths. This approach requires minimal changes in middleboxes and the overhead of FlowTags is comparable to traditional SDN mechanisms.

B. SERVICE CHAINING IN THE SOFTWARE-DEFINED NFV ARCHITECTURE

SDN and NFV together have the potential to benefit service operators satisfy user service level agreements, accurately monitor and control network traffic, which further reduces the minimize operating cost [77]. On one hand, NFV moves network functions out of dedicated hardware boxes to the software based on general hardware platform. On the other hand, SDN moves control functions out of the hardware and places it in the software controller. Therefore, the service deployment and service chains can be provided and reconfigured in the controller. In this scenario, not only flexible and dynamic operations are allowed, the chance for operation error and events will be much smaller because the network controller has an overall view, which benefits reducing the probability of inconsistent configurations.

Moving the required network functions into software means that deploying the service chain no longer requires acquiring dedicated middlebox. In this case, the network functions execute as the software running on virtual machines with the control of a hypervisor, which enable flexibility computational and networking resource provision. Thus, since the computational capacity can be increased when it is required, there's no need to over-provision. On the other hand, software-defined NFV service chaining also benefits the network upgrade process. For geographically distributed networks, upgrading network devices requires a large amount of cost. Moreover, the error happening in the network updates and re-configuration can bring down the entire network may outage on interconnecting providers' networks. However, with the software-defined NFV, service providers is able to create new chains without radically changing hardware. Finally, service operator can utilize these service chaining techniques to placing themselves, instead of the third party provider. With intelligent service chaining, complexity of resource provisioning is significantly reduced. Thus service

providers can deliver services on demand without the help of third parties.

The software-defined NFV architecture is still in research phase. A unified control and orchestration framework is required to integrate the SDN controller, forwarding elements and virtual network functions. Moreover, due to the existence of dynamic function and resource provisioning, this framework should also provide coordinated control of both network forwarding state and network functions' states [78]. Fig. 4 illustrates an example of the service chaining process. Taking user policies as inputs, the control module assigns the NFs fulfilling these services in an optimal way and meanwhile the optimal routing paths of all policies are selected taking account of the resource constraints. Then the service functions are chained by the centralized controller and the traffic flows are steered according to the service chains.

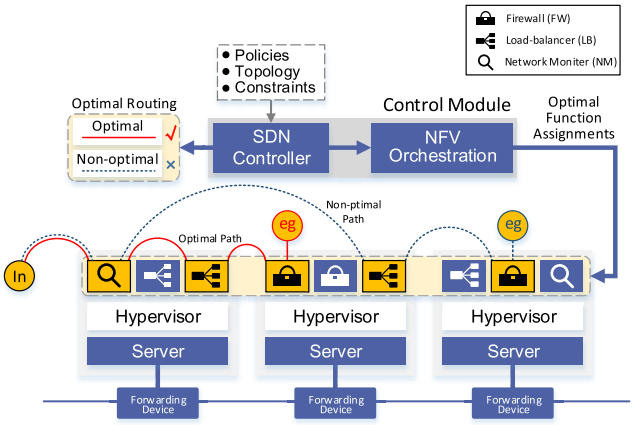

FIGURE 4. Service chaining in the software-defined NFV architecture.

V. CHALLENGES AND PROBLEMS OF NETWORK FUNCTION VIRTUALIZATION

NFV is an important innovation and a promising approach for the service operators and providers. However, it also faces several challenges. In this section, the corresponding challenges, open problems, and related solutions are summarized with the classification organized in Table 1.

A. FUNCTION VIRTUALIZATION

The virtualized functions should meet performance requirements to support packet processing at line-rate for multiple tenants [29], [79]. First, since neither the hypervisors nor the virtual machines have been optimized for the processing of middlebox, obtaining high performance, i.e., high I/O speed, fast packet processing, short transmission delays, etc, from standard servers is the main challenge for function virtualization. Further, as a server may implements a large amount of functionality, their platforms should host a wide range of virtual machine and software packages. Finally, NFV hardware and software platforms should support multi-tenancy, because they are concurrently run by software belonging to the different operators. These co-located VNFs should be isolated not only from a security but also a performance point of view. Below we discuss some important related works on function virtualization.

1) DPDK [80]

DPDK is a set of libraries and drivers for fast packet processing for the network functions. It could be run on a wide range of processors. However, the DPDK system has some limitation to support virtualization, and it along cannot support flexible, high performance functionality in the environment of NFV.

2) NetVM [81]

NetVM is a software platform for running diversity network functionality at line-speed based on the general commodity hardware. It takes advantage of DPDK's high throughput packet processing capabilities, and further enables flexible traffic steering and overcomes the performance limitations of hardware switching. Thus, It provides the capability to support network functions chains by flexible, high-performance network elements.

3) ClickOS [57], [82]

ClickOS is a high-performance, virtualized software network function platform. It provides small, booting quickly, and little delay virtual machines, and over one hundred of them can be concurrently run while guaranteeing ine-rate pipe on the general commodity server. To achieve high performance, ClickOS relies an extensive overhaul of Xen's I/O subsystem to speed up the networking process in middleboxes. ClickOS is proof that software solutions alone are enough to significantly speed up virtual machine processing, to the point where the remaining overheads are dwarfed by the ability to safely consolidate heterogeneous middlebox processing onto the same hardware.

B. PORTABILITY

The NFV framework is expected to support the loading, executing and moving VNFs across different but standard servers in multi-vendor environments. This capability is known as portability [83]. These virtualized network functions defeats the portability goal and key benefits of NFV, namely the capability of multi-tenancy and resource isolation. Furthermore, once instantiated, a NF leveraging SR-IOV cannot be migrated to another server. The portability challenge is how to achieve high performance leveraging hardware accelerators and at the same time have hardware independent NFs. This approach ensures that the VNFs are OS-independent and resource isolation is also guaranteed since the VNFs are executed on independent VMs and are decoupled from the underlying OS by the hypervisor layer.

C. STANDARD INTERFACES

NFV rely on existing infrastructure to touch the customer. In this case, it is also highly unlikely that a upgrade of the physical network or entire operational support systems will be feasible. This is a management software integration challenge with the interfaces between NFV and underlying

TABLE 1. NFV challenges.

Challenges	Description	Solution
Function Virtualization	Virtualized functions should meet certain requirements to support packet processing at line-rate: (1) High performance (high I/O speed, fast packet processing, short transmission delays, etc.) (2) Support multi-tenancy (3) OS-independent	Important related works: (1) DPDK, a set of libraries and drivers for fast packet processing. (2) NetVM, a system for running network functionality and middlebox at line-speed in general commodity hardware. (3) ClickOS, a small, quick-boot, low-delay, virtualized software middlebox platform.
Portability	The NFV framework is expected to load, execute and move VNFs across different but standard servers in multi-vendor environments. This capability is known as portability.	Deploying network functions via a virtual software environment enhances the portability. This approach ensures that the VNFs are OS-independent and resource isolation is also guaranteed.
Standard Interfaces	Standardized API should be developed to enable NFV to reach the costumers via underlying infrastructure and to be centrally controlled and managed.	Both VNFs and computing resources are described via standard templates. Normalized north- and south-bound should be developed between these layers.
Function Deployment	Fine-grained deployment, control and management of network functions, are needed in the context of NFV-enabled network nodes, for various optimization purposes.	A monitoring system collecting and reporting on the behaviour of the resources, and a unified control and optimization system with various optimization engines should be developed.
Traffic Steering	In the software-defined NFV architecture, traffic steering should be jointly optimized with function deployment, making the optimization problem difficult to solve.	To achieve online computing of traffic steering, heuristic algorithms should be designed to reduce the computing complexity.

infrastructure [84], [85]. On the other hand, the interfaces between the centralized controller and VNFs should also be standardized. To smoothly bridge NFV with upper and lower layers, the VNFs and the underlaying computing platform should be described by standard templates that enable flexible controlling and management. Thus, north- and south-bound need to be developed. North-bound interactions are used to control and manage functions to different types of instances, e.g., physical servers, VM and VNFs. Since network functions need service-oriented APIs to be controlled directly or indirectly, each network service has a specific operation policy and SLA. Moreover, VNFs could use the north-bound API for the requests. On the other hand, the south-bound API are utilized to communicate with the NFVI and request information from other framework entities. Thus, how to design a flexible and efficiency API for both the north-bound and south-bound communications are important problems in the research and development of NFV technologies.

D. FUNCTION DEPLOYMENT

Fine-grained deployment, control and management of network functions are needed in the context of NFV-enabled network nodes, for various optimization purposes [86]–[88]. Thus, many challenges are related to algorithm and system design regarding of function deployment.

One of these challenges is to automatically provide network and function process resources according to the usage of the resources involved [89]. A similar and probably even more important challenge is to achieve automatic placement and allocation of the VNFs, since the placement and assignment of the VNFs significantly impact the performance of service chaining [87], [88]. Both automated provisioning and placement require a global view of the resources and a unified control and optimization system with various optimization engines running in it. Another issues is to translate higher-level policies, which is generated from the resource allocation and optimization mechanisms, into lower level configurations [90], [91]. Templates and standards should be developed to guarantee automated and consistent translation. For example, when there is a need to achieve high-level goal of reducing the networking transmission delay, the optimization engine may require an algorithm to provision and place virtual functions ensuring that the least overall transmission delay is achieved. Conversely, when we require to achieve the minimum maximum link utilization, it would need a different optimization engine with a different algorithm. For more effective operation and control, the optimization approach should support real-time swap to make provisioning and placements that dynamically match the high-level policies from the operator and application.

E. TRAFFIC STEERING

SDN offers new agility of traffic steering by allowing the network operators and service providers to specify a logical control policy, and then it automatically translates this into data plane forwarding rules. Prior to this, the routing paths are carefully selected by the optimization framework taking into account the physical topology, link capacities, and network resource constraints. Solid work has been done on traffic steering in hardware based middlebox systems. However, in the software-defined NFV architecture, traffic steering is

jointly optimized with NF deployment that can achieve better composition. However, the unified optimization paradigm also makes the optimization problem difficult to solve since more variables are introduced and twisted. To achieve online computing of traffic steering, heuristic algorithms should be designed to reduce the computing complexity.

VI. APPLICATIONS AND FUTURE WORK

Software-defined NFV technology is in the usage of delivery significant benefits in niche applications today, while its full scale use and benefits have yet to be achieved. In this section, we look at what should happen in the next phase of software-defined NFV development following the journey and success the concept has enjoyed so far. We describe the major domains that are expected to dominate the software-defined NFV scenario over next few years.

A. CLOUD COMPUTING

Cloud computing [92]–[94] enable globally distributed services and enterprises to quickly deploy, manage and optimize their computing infrastructure dynamically. Partitioning or replicating a service across multiple globally distributed instances allow these services to move closer to the users thus providing richer user experiences, avoid infrastructure bottlenecks, and implement fault tolerance [35], [95]–[98].

NFV is an enabler of such dynamic service provisioning. By replacing service elements with virtual network functions, New functions can be added or improved by updating a software image, rather than waiting for a vendor to develop and manufacture a dedicated appliance. Furthermore, while integrated with SDN, service providers can express and enforce application traffic management policies and application delivery constraints at the required level of granularity [89], [99], [100].

NFV allows service providers to provide better services to the users by dynamically changing their deployment topologies or traffic allocations based on user access patterns, user mobility, infrastructure load characteristics, infrastructure failures and many such situations that may cause service degradation, disruption or churn [85]. Similarly, replicated service instances might need to be moved/instantiated/released to mask infrastructure failures, load conditions, or optimize the deployment based on access patterns and social interaction graphs. NFV, as well, can provide intelligent infrastructure support for such dynamic service deployment scenarios. Moreover, since NFV offers good support for multi-tenant usage, it is available for wide area dynamic multi-cloud environments that can be shared by multiple providers to implement their specific distributed service delivery contexts.

Below we enlist some important pioneering works trying to implement NFV in clouds.

1) CloudNFV [101]

CloudNFV is a multi-vendor consortium, which mainly aims to build an unified data model that incorporates data and policies of services and resources, in addition to orchestration process. It aims to decouple the creation of management information from the way it is presented. CloudNFV concerns two challenges of NFV orchestration in cloud, namely to embed services and network functions into physical/virtual infrastructures and the trade-off between automating SLA and price negotiation.

2) THE REALTIME CLOUD [102]

The realtime cloud relies on combing cloud, NFV and service provider SDN. Together, these technologies together enables more fluid, more dynamic and more responsive to new service needs. By enabling efficient control and management, and orchestration across network resources and applications, network-enabled Cloud, NFV and SDN together are able to help operators ensure they provide efficient and scale services.

3) CLOUDBAND [103]

CloudBand is Alcatel's end-to-end NFV solution and platform. Being open and multi-vendor, it supports the stringent needs of carriers and speedup the evolution to NFV. By introducing the CloudBand ecosystem, Alcatel is making it available to the NFV community for free with the goal of fostering collaboration and experimentation, which enable to accelerate NFV adoption and create new business and application opportunities.

B. MOBILE NETWORK

On the other hand, NFV considers all network functions for virtualization through well-defined standards, i.e., in mobile network, NFV targets at virtualizing mobile core network and the mobile-network base station [104]–[111]. NFV also benefits data centers owned by mobile service providers [112], including mobile core network, access networking and mobile cloud networks.

For the core networks, which is the most important part of mobile networks [113], [114], NFV allows the cellular providers to adopt a network more akin to the data centers, which consist of a fabric simple forwarding devices, with most functionality executed in commodity servers that are close to the base stations. Some network functions can even be fulfilled by packet-processing rules installed directly in the switches [105], [108], [110]. In the system, a logically-centralized controller is able to steer the network traffic through the required network functions to realize service chaining.

For the access networks, the base station is also considering to utilize the virtuliazation technology [118], [119]. Thus, SDN and NFV are applied to the wireless access networks [115]–[117], [120] to sharing their remote base-station infrastructure to achieve better coverage and services with the minimum investment of CAPEX and OPEX.

C. ENTERPRISE NETWORK

NFV will no doubt to be widely utilized in the enterprise [122]–[125]. Network managers would like to consume as

much or as little of the network as they need, but there is a gap between what enterprise customers want and what service providers can offer today, which can be address by NFV. It enables the dynamic provisioning of virtual network services on commodity servers within minutes instead of months.

NFV for the enterprise will require their platform to become more comfortable embracing software L4-7 services, as well as changes in their operation models. An understanding of how to optimize performance with DPDKs, and potentially even looking at programmable hardware, will be needed as well. Another challenge is the time and process it takes to re-architect monolithic services appliances that were predominantly deployed for north-south traffic. This can be achieved by the way that there may be many more appliances, but each supporting smaller workloads and be optimized for east/west traffic.

VII. CONCLUSION

In this work, we investigate a comprehensive overview of NFV within the software-defined NFV architecture. We introduce NFV its relationship with SDN. We also look at the history of NFV, presenting how middleboxes evolve to virtual network functions. In particular, we choose service chaining as a typical application of NFV. Furthermore, software-defined NFV challenges and possible solutions are presented. Finally, promising research areas are illustrated and future directions are presented.

REFERENCES

[1] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, "Making middleboxes someone else's problem: Network processing as a cloud service," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 13–24, 2012.

[2] Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang, "An untold story of middleboxes in cellular networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 374–385, 2011.

[3] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker, "Middleboxes no longer considered harmful," in *Proc. 6th Symp. OSDI*, vol. 4. 2004, p. 15.

[4] G. Schaffrath *et al.*, "Network virtualization architecture: Proposal and initial prototype," in *Proc. 1st ACM Workshop Virtualized Infrastruct. Syst. Archit.*, 2009, pp. 63–72.

[5] N. M. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.*, vol. 54, no. 5, pp. 862–876, 2010.

[6] N. M. M. K. Chowdhury and R. Boutaba, "Network virtualization: State of the art and research challenges," *IEEE Commun. Mag.*, vol. 47, no. 7, pp. 20–26, Jul. 2009.

[7] R. Guerzoni *et al.*, "Network functions virtualisation: An introduction, benefits, enablers, challenges & call for action," in *Proc. SDN OpenFlow World Congr.*, 2012, pp. 1–16.

[8] F. Yue. (2013). "Network functions virtualization—Everything old is new again," Tech. Rep. [Online]. Available: http://www.f5.com/pdf/white-papers/service-provider-nfv-white-paper.pdf

[9] A. Manzalini *et al.*, "Software-defined networks for future networks and services: Main technical challenges and business implications," Tech. Rep., 2014.

[10] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On scalability of software-defined networking," *IEEE Commun. Mag.*, vol. 51, no. 2, pp. 136–141, Feb. 2013.

[11] X. Ge, H. Cheng, M. Guizani, T. Han, "5G wireless backhaul networks: Challenges and research advances," *IEEE Netw.*, vol. 28, no. 6, pp. 6–11, Nov. 2014.

[12] A. Friis-Christensen, R. Lucchi, M. Lutz, and N. Ostländer, "Service chaining architectures for applications implementing distributed geographic information processing," *Int. J. Geographical Inf. Sci.*, vol. 23, no. 5, pp. 561–580, 2009.

[13] R. Lemmens, R. De By, M. Gould, A. Wytzisk, C. Granell, and P. Van Oosterom, "Enhancing geo-service chaining through deep service descriptions," *Trans. GIS*, vol. 11, no. 6, pp. 849–871, 2007.

[14] A. Greenberg *et al.*, "A clean slate 4D approach to network control and management," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 41–54, 2005.

[15] C. Tschudin and R. Gold, "Selnet: A translating underlay network," Uppsala Univ., Uppsala, Sweden, Tech. Rep. 2003-020, 2001.

[16] D. A. Joseph, A. Tavakoli, and I. Stoica, "A policy-aware switching layer for data centers," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 51–62, 2008.

[17] J. R. Santos, Y. Turner, G. Janakiraman, and I. Pratt, "Bridging the gap between software and hardware techniques for I/O virtualization," in *Proc. USENIX Annu. Tech. Conf.*, 2008, pp. 29–42.

[18] N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, F. Huici, and L. Mathy, "Towards high performance virtual routers on commodity hardware," in *Proc. ACM CoNEXT Conf.*, 2008, Art. ID 20.

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

[20] A. Greenhalgh, F. Huici, M. Hoerdt, P. Papadimitriou, M. Handley, and L. Mathy, "Flow processing and the rise of commodity network hardware," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 2, pp. 20–26, 2009.

[21] S. K. N. Rao, "SDN and its use-cases-NV and NFV," *Network*, vol. 2, p. H6, 2014

[22] A. Doria *et al.* "General switch management protocol (GSMP) V3," Tech. Rep., 2002, doi: http://dx.doi.org/10.17487/RFC3292.

[23] T. Wu, L. Rui, A. Xiong, and S. Guo, "An automation PCI allocation method for eNodeB and home eNodeB cell," in *Proc. IEEE 6th Int. Conf. Wireless Commun. Netw. Mobile Comput. (WiCOM)*, Sep. 2010, pp. 1–4.

[24] A. Berl, H. de Meer, H. Hlavacs, and T. Treutner, "Virtualization in energy-efficient future home environments," *IEEE Commun. Mag.*, vol. 47, no. 12, pp. 62–67, Dec. 2009.

[25] R. Mortier *et al.*, "Control and understanding: Owning your home network," in *Proc. IEEE 4th Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2012, pp. 1–10.

[26] H. Ludwig *et al.*, "Web service level agreement (WSLA) language specification," IBM Corp., New York, NY, USA, Tech. Rep., 2003, pp. 815–824.

[27] F. T. Leighton and D. M. Lewin, "Content delivery network using edge-of-network servers for providing content delivery to a set of participating content providers," U.S. Patent 6 553 413, Apr. 22, 2003.

[28] E. D. Zwicky, S. Cooper, and D. B. Chapman, *Building Internet Firewalls*. Sebastopol, CA, USA: O'Reilly Media, 2000.

[29] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, "NFV: State of the art, challenges, and implementation in next generation mobile networks (vEPC)," *IEEE Netw.*, vol. 28, no. 6, pp. 18–26, Nov./Dec. 2014.

[30] A. Gember *et al.* (2013). "Stratos: A network-aware orchestration layer for virtual middleboxes in clouds." [Online]. Available: http://arxiv.org/abs/1305.0209

[31] J. Case, M. Fedor, M. Schoffstall, and J. Davin, *A Simple Network Management Protocol (SNMP)*, document 1157, 1989.

[32] A. Leinwand and K. F. Conroy, *Network Management: A Practical Perspective* (Unix and Open Systems Series), vol. 1, 2nd ed. Reading, MA, USA: Addison-Wesley, 1996.

[33] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," HAL, Bengaluru, India, Tech. Rep. hal-00825087, 2013.

[34] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 27–51, Jun. 2014.

[35] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: A survey," *IEEE Commun. Mag.*, vol. 51, no. 11, pp. 24–31, Nov. 2013.

[36] Open Networking Fundation, "Software-defined networking: The new norm for networks," Open Netw. Found., ONF White Paper, 2012.

[37] M. Casado, T. Koponen, S. Shenker, and A. Tootoonchian, "Fabric: A retrospective on evolving SDN," in *Proc. 1st Workshop Hot Topics Softw. Defined Netw.*, 2012, pp. 85–90.

[38] N. Foster *et al.*, "Frenetic: A network programming language," *ACM SIGPLAN Notices*, vol. 46, no. 9, pp. 279–291, 2011.
[39] N. Handigol, S. Seetharaman, M. Flajslik, N. McKeown, and R. Johari, "Plug-n-serve: Load-balancing Web traffic using OpenFlow," in *Proc. ACM SIGCOMM Demo*, 2009, pp. 1–2.
[40] A. Doria *et al.*, *Forwarding and Control Element Separation (ForCES) Protocol Specification*, document 5810, 2010.
[41] B. Raghavan, M. Casado, T. Koponen, S. Ratnasamy, A. Ghodsi, and S. Shenker, "Software-defined Internet architecture: Decoupling architecture from infrastructure," in *Proc. 11th ACM Workshop Hot Topics Netw.*, 2012, pp. 43–48.
[42] R. Bifulco, R. Canonico, M. Brunner, P. Hasselmeyer, and F. Mir, "A practical experience in designing an OpenFlow controller," in *Proc. IEEE Eur. Workshop Softw. Defined Netw. (EWSDN)*, Oct. 2012, pp. 61–66.
[43] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "On controller performance in software-defined networks," in *Proc. USENIX Workshop Hot Topics Manage. Internet, Cloud, Enterprise Netw. Services (Hot-ICE)*, vol. 54. 2012, pp. 1–6.
[44] M. Monaco, O. Michel, and E. Keller, "Applying operating system principles to SDN controller design," in *Proc. 12th ACM Workshop Hot Topics Netw.*, 2013, Art. ID 2.
[45] G. Lu *et al.*, "Serverswitch: A programmable and high performance platform for data center networks," in *Proc. NSDI*, vol. 11. 2011, pp. 1–14.
[46] D. Drutskoy, E. Keller, and J. Rexford, "Scalable network virtualization in software-defined networks," *IEEE Internet Comput.*, vol. 17, no. 2, pp. 20–27, Mar./Apr. 2013.
[47] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.
[48] ONF Solution Brief, "OpenFlow-enabled SDN and network functions virtualization," Open Netw. Found., 2014.
[49] E. Ng, "Maestro: A system for scalable OpenFlow control," TSEN, Rice Univ., Houston, TX, USA, Maestro-Tech. Rep. TR10-08, 2010.
[50] C. Rotsos, N. Sarrar, S. Uhlig, R. Sherwood, and A. W. Moore, "OFLOPS: An open framework for OpenFlow switch evaluation," in *Passive and Active Measurement*. Berlin, Germany: Springer, 2012, pp. 85–95.
[51] R. Wang, D. Butnariu, and J. Rexford, "OpenFlow-based server load balancing gone wild," in *Proc. 11th USENIX Conf. Hot Topics Manage. Internet, Cloud, Enterprise Netw. Services*, 2011, p. 12.
[52] A. Dixit, F. Hao, S. Mukherjee, T. V. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," in *Proc. 2nd ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw.*, 2013, pp. 7–12.
[53] N. Gude *et al.*, "NOX: Towards an operating system for networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 3, pp. 105–110, 2008.
[54] D. Joseph and I. Stoica, "Modeling middleboxes," *IEEE Netw.*, vol. 22, no. 5, pp. 20–25, Sep./Oct. 2008.
[55] M. Allman, "On the performance of middleboxes," in *Proc. 3rd ACM SIGCOMM Conf. Internet Meas.*, 2003, pp. 307–312.
[56] V. Sekar, S. Ratnasamy, M. K. Reiter, N. Egi, and G. Shi, "The middlebox manifesto: Enabling innovation in middlebox deployment," in *Proc. 10th ACM Workshop Hot Topics Netw.*, 2011, Art. ID 21.
[57] J. Martins *et al.*, "ClickOS and the art of network function virtualization," in *Proc. 11th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Seattle, WA, USA, 2014, pp. 459–473.
[58] Network Working Group Published in Computer Communication Review, K. Egevang, and P. Francis, "The IP network address translator (NAT)," Tech. Rep. RFC 1631, May 1994.
[59] C. H. Rowland, "Intrusion detection system," U.S. Patent 6 405 318, Jun. 11, 2002.
[60] M. K. Bowman-Amuah, "Load balancer in environment services patterns," U.S. Patent 6 578 068, Jun. 10, 2003.
[61] V. Sekar, N. Egi, S. Ratnasamy, M. K. Reiter, and G. Shi, "Design and implementation of a consolidated middlebox architecture," in *Proc. 9th USENIX Conf. NSDI*, 2012, pp. 323–336.
[62] J. Sherry, "Future architectures for middlebox processing services on the Internet and in the cloud," M.S. thesis, Dept. EECS, Univ. California, Berkeley, Berkeley, CA, USA, 2012.
[63] J. Lee, J. Tourrilhes, P. Sharma, and S. Banerjee, "No more middlebox: Integrate processing into network," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 459–460, 2010.
[64] A. Gember, P. Prabhu, Z. Ghadiyali, and A. Akella, "Toward software-defined middlebox networking," in *Proc. 11th ACM Workshop Hot Topics Netw.*, 2012, pp. 7–12.
[65] A. Gember, R. Grandl, J. Khalid, and A. Akella, "Design and implementation of a framework for software-defined middlebox networking," in *Proc. ACM SIGCOMM Conf. (SIGCOMM)*, 2013, pp. 467–468.
[66] J. W. Anderson, R. Braud, R. Kapoor, G. Porter, and A. Vahdat, "xOMB: Extensible open middleboxes with commodity servers," in *Proc. 8th ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, 2012, pp. 49–60.
[67] P. Quinn and T. Nadeau, *Service Function Chaining Problem Statement*, document draft-quinn-sfc-problem-statement-02, 2013.
[68] *Cisco Catalyst 6500 Series Switches Solution*, Cisco Syst., Inc., San Jose, CA, USA, 2015.
[69] B. Davie and Y. Rekhter, *MPLS: Technology and Applications*. San Mateo, CA, USA: Morgan Kaufmann, 2000.
[70] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet indirection infrastructure," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 4, pp. 73–86, 2002.
[71] R. Rao, "Multi-service network switch with policy based routing," U.S. Patent 6 789 118 B1, Sep. 7, 2004.
[72] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 1–12, 2007.
[73] J. Blendin, J. Rückert, N. Leymann, G. Schyguda, and D. Hausheer, "Position paper: Software-defined network service chaining," in *Proc. 3rd EWSDN Workshop*, Sep. 2014, pp. 109–114.
[74] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu, "SIMPLE-fying middlebox policy enforcement using SDN," in *Proc. ACM SIGCOMM Conf. (SIGCOMM)*, 2013, pp. 27–38.
[75] Y. Zhang *et al.*, "StEERING: A software-defined networking for inline service chaining," in *Proc. 21st IEEE ICNP*, Oct. 2013, pp. 1–10.
[76] S. K. Fayazbakhsh, L. Chiang, V. Sekar, M. Yu, and J. C. Mogul, "Enforcing network-wide policies in the presence of dynamic middlebox actions using FlowTags," in *Proc. 11th USENIX Symp. NSDI*, 2014, pp. 533–544.
[77] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "SOLuTIoN: SDN-based optical traffic steering for NFV," in *Proc. 3rd Workshop Hot Topics Softw. Defined Netw.*, 2014, pp. 227–228.
[78] A. Gember-Jacobson *et al.*, "OpenNF: Enabling innovation in network function control," in *Proc. ACM Conf. SIGCOMM*, 2014, pp. 163–174.
[79] H. Masutani *et al.*, "Requirements and design of flexible NFV network infrastructure node leveraging SDN/OpenFlow," in *Proc. Int. Conf. Opt. Netw. Design Model.*, May 2014, pp. 258–263.
[80] *Intel Data Plane Development Kit*, Intel, Santa Clara, CA, USA, 2015.
[81] J. Hwang, K. K. Ramakrishnan, and T. Wood, "NetVM: High performance and flexible networking using virtualization on commodity platforms," in *Proc. 11th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2014, pp. 1–12.
[82] J. Martins, M. Ahmed, C. Raiciu, and F. Huici, "Enabling fast, dynamic network processing with ClickOS," in *Proc. 2nd ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw.*, 2013, pp. 67–72.
[83] D. King and C. Ford, "A critical survey of network functions virtualization (NFV)," in *Proc. iPOP*, 2013, pp. 1–21.
[84] W. Shen, M. Yoshida, T. Kawabata, K. Minato, and W. Imajuku, "vConductor: An NFV management solution for realizing end-to-end virtual network services," in *Proc. 16th Asia-Pacific Netw. Oper. Manage. Symp. (APNOMS)*, Sep. 2014, pp. 1–6.
[85] M. Yoshida, W. Shen, T. Kawabata, K. Minato, and W. Imajuku, "MORSA: A multi-objective resource scheduling algorithm for NFV infrastructure," in *Proc. 16th APNOMS*, Sep. 2014, pp. 1–6.
[86] M. Bouet, J. Leguay, and V. Conan, "Cost-based placement of virtualized deep packet inspection functions in SDN," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Nov. 2013, pp. 992–997.
[87] M. Scholler, M. Stiemerling, A. Ripke, and R. Bless, "Resilient deployment of virtual network functions," in *Proc. 5th Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops (ICUMT)*, Sep. 2013, pp. 208–214.
[88] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, "The dynamic placement of virtual network functions," in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, May 2014, pp. 1–9.
[89] R. Cannistra *et al.*, "Enabling autonomic provisioning in SDN cloud networks with NFV service chaining," in *Proc. Opt. Fiber Commun. Conf. Exhibit.*, Mar. 2014, pp. 1–3, paper Tu2I-4.

[90] T. Benson, A. Akella, and D. A. Maltz, "Mining policies from enterprise network configuration," in *Proc. 9th ACM SIGCOMM Conf. Internet Meas. Conf.*, 2009, pp. 136–142.
[91] N. Kang, J. Reich, J. Rexford, and D. Walker, "Policy transformation in software defined networks," in *Proc. ACM SIGCOMM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2012, pp. 309–310.
[92] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.
[93] Z. Liu, Y. Li, D. Jin, L. Su, and L. Zeng, "M2cloud: Software defined multi-site data center network control framework for multi-tenant," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 517–518, 2013.
[94] L. Qian, Z. Luo, Y. Du, and L. Guo, "Cloud computing: An overview," in *Cloud Computing*. Berlin, Germany: Springer, 2009, pp. 626–631.
[95] H. Wang, Y. Li, Y. Zhang, and D. Jin, "Virtual machine migration planning in software-defined networks," in *Proc. IEEE Conf. INFOCOM*, Apr./May 2015, pp. 487–495.
[96] J. Liu, Y. Li, D. Jin, L. Su, and L. Zeng, "Traffic aware cross-site virtual machine migration in future mobile cloud computing," *Mobile Netw. Appl.*, vol. 20, no. 1, pp. 62–71, Feb. 2015.
[97] M. Chen, Y. Hao, Y. Li, C. Lai, and D. Wu, "On the computation offloading at ad hoc cloudlet: Architecture and service models," *IEEE Commun.*, vol. 53, no. 6, pp. 18–24, Jun. 2015.
[98] J. Liu, Y. Li, and D. Jin, "SDN-based live VM migration across datacenters," in *Proc. ACM Conf. SIGCOMM*, 2014, vol. 44. no. 4, pp. 583–584.
[99] F. Derakhshan, H. Grob-Lipski, H. Roessler, P. Schefczik, and M. Soellner, "Enabling cloud connectivity using SDN and NFV technologies," in *Mobile Networks and Management*. Springer International Publishing, 2013, pp. 245–258.
[100] M. Chen, S. Gonzalez, Q. Zhang, and V. Leung, "Code-centric RFID system based on software agent intelligence," *IEEE Intell. Syst.*, vol. 25, no. 2, pp. 12–19, Mar./Apr. 2010.
[101] *CloudNFV*. [Online]. Available: http://www.cloudnfv.com/, accessed 2015.
[102] Combining Cloud, NFV, and Service Provider SDN, "The real-time cloud," Ericsson, Stockholm, Sweden, White Paper Uen 284 23-3219, Feb. 2014.
[103] *Alcatel-CloudBand*. [Online]. Available: http://www.alcatel-lucent.com/solutions/cloudband, accessed 2015.
[104] M. Yang, Y. Li, D. Jin, L. Zeng, X. Wu, and A. V. Vasilakos, "Software-defined and virtualized future mobile and wireless networks: A survey," *Mobile Netw. Appl.*, vol. 20, no. 1, pp. 4–18, Feb. 2015.
[105] S. Paul and R. Jain, "OpenADN: Mobile apps on global clouds using OpenFlow and software defined networking," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2012, pp. 719–723.
[106] J. Liu, Y. Li, M. Chen, W. Dong, and D. Jin, "Software-defined Internet of Things for smart urban sensing," *IEEE Commun. Mag.*, vol. 53, no. 8, pp. 55–63, Sep. 2015.
[107] M. Yang *et al.*, "Cross-layer software-defined 5G network," *Mobile Netw. Appl.*, vol. 20, no. 3, pp. 400–409, 2015.
[108] X. Jin, L. E. Li, L. Vanbever, and J. Rexford, "SoftCell: Scalable and flexible cellular core network architecture," in *Proc. 9th ACM Conf. Emerg. Netw. Experim. Technol.*, 2013, pp. 163–174.
[109] M. Moradi, L. E. Li, and Z. M. Mao, "SoftMoW: A dynamic and scalable software defined architecture for cellular WANs," in *Proc. 3rd Workshop Hot Topics Softw. Defined Netw.*, 2014, pp. 201–202.
[110] L. E. Li, Z. M. Mao, and J. Rexford, "CellSDN: Software-defined cellular networks," Dept. Comput. Sci., Princeton Univ., Princeton, NJ, USA, Tech. Rep., 2012.
[111] A. Basta, A. Blenk, M. Hoffmann, H. J. Morper, K. Hoffmann, and W. Kellerer, "SDN and NFV dynamic operation of LTE EPC gateways for time-varying traffic patterns," in *Proc. 6th Int. Conf. Mobile Netw. Manage.*, Sep. 2014, pp. 63–76.
[112] A. Basta, W. Kellerer, M. Hoffmann, H. J. Morper, and K. Hoffmann, "Applying NFV and SDN to LTE mobile core gateways. The functions placement problem," in *Proc. 4th Workshop All Things Cellular, Oper., Appl., Challenges*, 2014, pp. 33–38.
[113] J. De Vriendt, P. Laine, C. Lerouge, and X. Xu, "Mobile network evolution: A revolution on the move," *IEEE Commun. Mag.*, vol. 40, no. 4, pp. 104–111, Apr. 2002.
[114] T. C. Y. Wang and S. H. Moritz, "Mobile unit tracking system," U.S. Patent 5 365 451, Nov. 15, 1994.
[115] M. Yang, Y. Li, D. Jin, L. Su, S. Ma, and L. Zeng, "OpenRAN: A software-defined ran architecture via virtualization," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 549–550, 2013.
[116] Y. Niu, Y. Li, M. Chen, D. Jin, and S. Chen, "A cross-layer design for software defined millimeter-wave mobile broadband system," *IEEE Commun. Mag.*, pp. 1–18, Sep. 2015.
[117] Y. Li, P. Hui, D. Jin, and S. Chen, "Delay-tolerant network protocol testing and evaluation," *IEEE Commun. Mag.*, vol. 53, no. 1, pp. 258–266, Jan. 2015.
[118] Z. Zhu *et al.*, "Virtual base station pool: Towards a wireless network cloud for radio access networks," in *Proc. 8th ACM Int. Conf. Comput. Frontiers*, 2011, Art. ID 34.
[119] G. Bhanage, I. Seskar, R. Mahindra, and D. Raychaudhuri, "Virtual basestation: Architecture for an open shared WiMAX framework," in *Proc. 2nd ACM SIGCOMM Workshop Virtualized Infrastruct. Syst. Archit.*, 2010, pp. 1–8.
[120] B. Pfaff, J. Pettit, T. Koponen, K. Amidon, M. Casado, and S. Shenker, "Extending networking into the virtualization layer," in *Proc. HotNets*, 2009, pp. 1–6.
[121] B. Rimler and N. Rasmussen, "Mobile data center," U.S. Patent 2006 0 082 263 A1, Apr. 20, 2006.
[122] Y. Zhou, X. Yang, Y. Li, D. Jin, L. Su, and L. Zeng, "Incremental re-embedding scheme for evolving virtual network requests," *IEEE Commun. Lett.*, vol. 17, no. 5, pp. 1016–1019, May 2013.
[123] A. K. Nayak, A. Reimers, N. Feamster, and R. Clark, "Resonance: Dynamic access control for enterprise networks," in *Proc. 1st ACM Workshop Res. Enterprise Netw.*, 2009, pp. 11–18.
[124] X. Ge, B. Yang, J. Ye, G. Mao, C.-X. Wang, and T. Han, "Spatial spectrum and energy efficiency of random cellular networks," *IEEE Trans. Commun.*, vol. 63, no. 3, pp. 1019–1030, Mar. 2015.
[125] Z. Liu, Y. Li, B. Cui, L. Su, D. Jin, and L. Zeng, "GrainFlow: Enable testing for future Internet architectures by per-bit customization," *Comput. Netw.*, vol. 69, pp. 121–132, Aug. 2014.

YONG LI (M'09) received the B.S. degree in electronics and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2007, and the Ph.D. degree in electronics engineering from Tsinghua University, Beijing, China, in 2012. In 2012 and 2013, he was a Visiting Research Associate with Telekom Innovation Laboratories and The Hong Kong University of Science and Technology, respectively. From 2013 to 2014, he was a Visiting Scientist with the University of Miami. He is currently a Faculty Member with the Department of Electronic Engineering, Tsinghua University. His research interests are in the areas of networking and communications.

MIN CHEN (M'08–SM'09) was an Assistant Professor with the School of Computer Science and Engineering, Seoul National University (SNU), from 2009 to 2012. He was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of British Columbia (UBC), for three years. Before joining UBC, he was a Post-Doctoral Fellow with SNU for one and a half years. He is currently a Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology (HUST), where he is also the Director of the Embedded and Pervasive Computing Laboratory. He has authored over 260 paper publications, including over 100 SCI papers, over 50 IEEE TRANSACTIONS/journal papers, six ISI highly cited papers, and one hot paper. He has also authored the books entitled *Internet of Things (IoT): OPNET IoT Simulation* (HUST Press, 2015) and *Big Data Related Technologies* (Springer Series in Computer Science, 2014) in big data. His Google Scholars Citations reached over 5300 with an h-index of 34. His top paper was cited 648 times, while his top book was cited 420 times in 2015. His research focuses on IoT, mobile cloud, body area networks, emotion-aware computing, healthcare big data, cyber physical systems, and robotics. He received the best paper award from the IEEE ICC 2012, and the Best Paper Runner-Up Award from QShine 2008.