1. Network Programmability: A Historical Perspective

Walter Cerroni[1], Stefano Giordano[2]

[1]University of Bologna - Italy

[2]University of Pisa - Italy

Abstract: *The concept of network programmability is completely reshaping the way communication infrastructures and services are designed, developed, deployed and maintained. New principles, architectures, technologies and solutions are fostering a paradigm shift that enables a unprecedented level of innovation in the telecommunication field, which is more and more becoming intertwined with the computing field. However, any (r)evolution in any field has its roots in a series of capstone events and premises that contributed to that (r)evolutionary vision. In this chapter we present a historical perspective of network programmability and the closely related concept of network virtualization, with an account of the most important evolutionary steps and disruptive innovations that brought us to the current idea of a programmable and "softwarized" network.*

1.1 Introduction

It is often stated that, after a long period of immobility, the acceptance of disruptive changes in new directions comes from the consciousness of new critical problems, which can be solved only with a profound knowledge of the processes characterizing the core activity of an institution, a company, or even a business. Experts in economics often claim that "innovation comes from desperation." And this was indeed the case for the Internet at the beginning of the 21st century. After thirty years of research, experiments and successful deployment of what we can consider as the winner of a war between a plethora of competing technologies and architectural alternatives, the third millennium planetary – or even inter-planetary – Internet has struggled through many critical issues, including dynamic access control, load balancing, energy efficiency, user mobility, address space depletion, just to mention a few. If it is true that who wins the war sets the rules, then in this case the winner, i.e., the TCP/IP network architecture, forced its peripheral devices (hosts, end systems) to adapt to its "dogmatic" design approach, made of end-to-end congestion control, destination-based routing, single path hierarchical routing, a core-edge structure, addresses that represent the identifier of an interface (often of an entire system) and, at the same time, provide topological information, etc. Nevertheless, that very same network architecture experienced its most revolutionary jumps ahead always outside of the network itself, i.e., on an abstraction of the abstraction, on overlays such as the web or the peer-to-peer network. More and more at that stage the solutions to the main problems called for new functional components to be added as "patch enablers,"

2. Software Defined Networking (SDN)

Sergio Palazzo[1], Giuseppe M. Milotta[2], Giacomo Morabito[1]

[1]University of Catania - Italy

[2]Mediterranean University of Reggio Calabria - Italy, and CNIT, Catania - Italy

Abstract: *The Internet architecture has remained almost unchanged at its core since its birth to the late 2000s. In the last decade, at long last, the rise of the Software Defined Networking (SDN) paradigm is radically changing not only the architecture of the networks and the way they are designed and managed, but, also, the way in which they are thought and taught. In this chapter we will review the main motivations and concepts of SDN. Although we will often give references to OpenFlow, which the most widely known implementations are based to, our discussion will be general as we will cover other relevant SDN approaches introduced for specific scenarios.*

2.1 Introduction

In the previous chapter, Software Defined Networking (SDN) has been put in the context of the communication and networking technology evolution. In this chapter we will focus on the conceptual and technical characteristics of SDN.

More specifically, in Section 2.2 we will start by defining what is SDN, and then, we will describe its architecture. In Section 2.2 we will also provide an overview of OpenFlow, which is the most widely deployed SDN solution. In Section 2.3, we will focus on the main abstractions introduced by SDN. In Section 2.4 we will discuss the major security challenges involved by the SDN approach, along with the most common approaches proposed to address them. Then, in Section 2.5, we will describe the *network slicing* concept, which plays a key role for the support of *multitenancy* in the networking domain. Finally, in Section 2.6 we will provide the reader with some tools for experiencing the key SDN concepts in the practice.

2.2 SDN: definitions and architecture

In this section we will first define *Software Defined Networking* along with its fundamental concepts; then, we will describe the architecture of software defined networks.

We will conclude this section with a high level description of OpenFlow, which is the most widely deployed SDN solution: such a description can be viewed as a framework where the concepts described in the following sections can be put into a practical context.

3. Network Function Virtualization (NFV)

Sergio Palazzo, Christian Grasso, Giovanni Schembra

University of Catania - Italy

Abstract: *This chapter describes Network Function Virtualization (NFV), a network paradigm emerged in the last years with the aim of sustaining the evolution process of telecommunications networks, commonly known as network softwarization. After a description of the motivations underlying the definition of NFV and the tangible benefits that are obtained thanks to this technology, the chapter will describe the softwarization approach introduced with NFV. Then, the chapter will provide a description of the standardized NFV architecture, considering the NFVI, the service plane and the NFV MANO, and an overview of enabling technologies of virtualization used to host virtual network functions (VNF). Then, an overview of the main challenges and open issues related to the application of the NFV networking paradigm will be introduced. Finally, an example of installation and deployment of a service chain made up of five NFVs to realize a security network service will conclude the chapter.*

3.1 Introduction

Traditionally, network operators have deployed a huge number of physical proprietary devices for every part of the network. However, with the increase of user demands on network functions, they are required to acquire many specialized network hardware (known as middleboxes or hardware appliances) to satisfy the plethora of heterogeneous user requirements. Middleboxes embody a large variety of specialized functions to forward, classify, or transform traffic based on the packet contents. Examples of middlebox functions include (but are not limited to) L2 Switch, Router, Network Address Translation (NAT), Firewall (FW), Deep Packet Inspection (DPI), Intrusion Detection System (IDS), Load Balancer (LB), Wide Area Network (WAN) optimizer, Multimedia Caches, Video Transcoders and Quality of Service (QoS) Monitors.

Recent studies [1] have shown that, in an enterprise network, middleboxes are ubiquitous and their number is comparable with the number of routers and switches needed to maintain the operation of the network [2]. Unfortunately, their usage suffers of several shortcomings. In fact, they are expensive, require specialized managing personnel, and present high delay in introducing new networking features. Moreover, they have high energy costs and short lifecycles. In addition, middleboxes have vendor-specific interfaces and slow protocol standardization [3, 4]. Deploying new network services (NSs) with middleboxes is also a tedious process, as technicians are required to visit specific sites and place the middleboxes in a pre-defined order to form the correct service function chains (SFCs). This results in high Capital Expenditure (CAPEX) and Operational Expenditure (OPEX) [5, 6].

4. The Programmable Data Plane

Giuseppe Bianchi[1], Roberto Bifulco[2]

[1]University of Roma "Tor Vergata" - Italy

[2]NEC Laboratories Europe - Germany

Abstract: *This chapter provides an overview of technologies and solutions for programmable data planes, a recent trend in network softwarization emerged around 2014. The programmable data plane demonstrates that the dichotomy among flexibility of software and high performance of hardware is a false one, and can be overcome by carefully designing* domain-specific architectures *capable to very efficiently perform a set of well defined operations on received packets, but still providing the control plane with the programmatic capability to systematically, rapidly, and comprehensively reconfigure the data plane.*

4.1 Introduction

Future networked services will challenge network infrastructures in many ways. Not only bandwidth consumption will continue to increase, but this will come along with new computation/processing demands, as well as tight latency requirements when real-time processing is mandated. Examples of such demanding services include, but obviously do not nearly limit to, virtual/augmented reality, holography, unmanned control, network support for artificial intelligence and machine learning, multi-user gaming, large scale IoT scenarios, traffic analytic for security, and so on - all requiring massive communication, computation and storage capabilities [1].

Such an heterogeneity of scenarios and requirements complicate the design of network systems. Appliances such as switches, network interface cards, and the huge variety of diverse *middleboxes* appeared throughout the last 30 years (for address translation, traffic management, session control, security, performance enhancement, transcoding, monitoring, etc), had to necessarily evolve and become way more flexible and general so as to reduce design costs and simplify management.

The quest for programmability

Such need has favored the emergence of network functions programmable via software. Programmability introduces a significant change in the relationship between device suppliers and network operators. A programmable device frees the operator from waiting for traditional network equipment release cycles, which last for years, when new features are launched. In fact, a new functionality can be quickly implemented and deployed directly by the operator using the device programming interface. On the other hand, programmability frees device suppliers from designing network equipment for a wide range

5. Multi-level Orchestration of Micro-Services and Network Functions

Raffaele Bolla[1,2], Roberto Bruschi[1,2], Franco Davoli[1,2], Chiara Lombardo[1] and Jane Frances Pajo[1,2]

[1]CNIT S2N National Laboratory, Genova - Italy

[2]University of Genova - Italy

Abstract: *What we consider in this chapter regards essentially the separation of concerns across vertical application domains and network service providers' domains, as well as through multiple network segments (core/cloud and edge) regarding the orchestration of multiple virtual entities. The latter cover, on one side, the micro-services that compose an application graph and, on another, the Virtual Network Functions (VNFs) involved in providing a network service chain. The lifecycles of both can be realized in a physical space spanning the network edge (including in a broad vision also User Equipment – UE), the backhaul and core networks and parts of cloud datacenters. We recall the concept of cloud-native applications, and the need to make such applications "5G-ready", by including network awareness in their component micro-services. This entails the presence of multiple orchestration levels, whose mutual interaction is mediated by the Operational Support System (OSS) of a Network Service Provider (NSP). We also highlight the need of mechanisms and abstractions that allow end-to-end dynamic lifecycle monitoring and reconfiguration support across all application and network service components.*

5.1 Introduction

The Fifth Generation (5G) of mobile radio networks and edge computing technologies are widely expected to constitute the second wave of the Data Revolution [1], and to play a key role in the digital transformation enabled by Big Data and Cloud Computing technologies towards a new hyper-connected society. In this profound transformation, modern cloud technologies and architectures are recognized as the foundations of the upcoming 5G ecosystem, along with the "softwarization" revolution in telecommunication infrastructures, and have become key enablers for new (more pervasive and more network-integrated) computing paradigms like, for instance, Fog and Mobile Edge Computing (MEC) [2, 3, 4, 5]. In its turn, the unstoppable softwarization trend in networking has been enabled by the flexibility allowed by the Network Functions Virtualization (NFV) [6] and Software Defined Networking [7] paradigms.

5G has been designed to provide flexible support for radically new and extremely heterogeneous vertical applications, requiring any custom mix of network and radio services. Customized instances of the 5G network (i.e., 5G network slices [8]), providing the

6. Network Slicing

Carla Raffaelli[1], Paolo Monti[2], Federico Tonini[2]

[1]University of Bologna - Italy

[2]Chalmers University of Technology, Gothenburg - Sweden

Abstract: *Network slicing is emerging as a key enabling technology to support new service needs, business cases, and the evolution of programmable networking. As an end-to-end concept involving network functions in different domains and administrations, network slicing calls for new standardization efforts, design methodologies, and deployment strategies. This chapter aims at addressing the main aspects of network slicing with relevant challenges and practical solutions.*

6.1 Introduction

6.1.1 Background, concepts and motivations

Network slicing is emerging as a key technology for programmable networks thanks to the maturity reached by network virtualization techniques and emerging business opportunities, especially, but not exclusively, in relation to 5G and further generations of cellular networks. Network slicing is targeted to accommodate end-to-end services while maintaining the quality of service requirements even in changing network conditions. As a consequence, it has triggered the activity of the main standardization bodies, including 3GPP, IETF, ITU-T, supported by many alliances and groups.

A network slice is a set of network functions and resources, either virtual or physical, which are logically correlated to provide programmable end-to-end connectivity and services on-demand, according to performance requirements, as similarly defined by main consortia [1], [2].

This concept can be seen as an evolution of the Infrastructure as a Service (IaaS) cloud computing model, that has been pushed forward to a much more flexible and dynamic paradigm as network slicing, where a thorough adoption of virtualization, potentially for any network function, is applied. As a consequence, network slicing offers extremely high potential in shaping network platforms with high flexibility involving different networks, cloud resources, operators, and business players, which also translates into a challenging increased complexity in network control, management, and orchestration [3].

Some aspects can be identified to characterize network slicing:

- end-to-end: an intrinsic property of network slicing to facilitate service delivery to end users, customers and applications;
- resource sharing: to allow better network resource utilization levels in the presence of many different and differently evolving service needs;

7. VNF Placement and Artificial Intelligence (AI)-based Resource Optimization

Vincenzo Eramo[1], Antonio Cianfrani[1], Francesco Giacinto Lavacca[2], Marco Polverini[1], Tiziana Catena[1]

[1]University of Rome "La Sapienza" - Italy

[2]Fondazione Ugo Bordoni, Rome - Italy

Abstract: *Network Function Virtualization (NFV) utilizes the virtualization technologies to consolidate various types of network appliances onto standard high-volume off-the-shelf servers. These technologies bring significant advantages to service providers including the the capital and operation reduction and also improving the flexibility and innovation in the offered services. The chapter is focused on the discussion of the resource allocation problem in static and dynamic traffic scenario. Particular emphasis is placed on Artificial Intelligence (AI)-based traffic and resources prediction techniques.*

7.1 Introduction

The NFV paradigm allows network operators to deploy new services faster and in a flexible way, exploiting the possibility of executing Network Functions (NFs) as virtualized software running on commercial-off-the-shelf servers, i.e. Virtual Network Functions (VNF) [1]-[2]. Anyway, the deployment of NFVs requires novel strategies for the allocation of computing and network resources. More in detail, algorithms for the joint allocation of cloud and bandwidth resources to network services composed by the ordered interconnection of VNFs leads, are needed. In this chapter, we describe VNF allocation algorithms in the Multi-Cloud NFV network scenario [3]-[4], i.e. remote data-centers are available for computing/storage resources. Considering the network services requests, we differentiate two scenarios: i) the static case and ii) the dynamic one. In the former case, the set of VNF request is known in advance and considered not variable over time. In the latter case, network services requests change over time, thus requiring solutions such as VNF reconfigurations and migrations; for the dynamic traffic case, we describe VNF placement algorithms exploiting Artificial Intelligence (AI) solutions.
The chapter is organized as follows. We describe the reference scenario in Section 7.2. The resource allocation problem in static traffic scenario is illustrated in Section 7.3. AI-based resource allocation methodologies in dynamic traffic scenario are illustrated in Section 7.4. The main conclusions are reported in Section 7.5.

8. Service Function Chaining

Davide Borsatti, Franco Callegati, Gianluca Davoli

University of Bologna - Italy

Abstract: *Service Function Chaining (SFC) is one of the key innovative technologies in networking with NFV and SDN. Very briefly we can say that SFC allows to program dynamically the path of specific traffic flows through a given set of VNFs. At a first glance SFC may look like a subset of conventional routing problems, but it is not and thus requires specific technologies to be fully functional in a network. In this Chapter we will review the definitions of SFC, we will explain what is peculiar of this technology and review the relevant standardization activities. Then we will show some examples of implementation strategies of SFC in real life scenarios.*

8.1 Introduction

8.1.1 Network Services and the Service Function Chain

The traditional service deployment model is strictly related to the underlying resources and network topology. Today, network value-added services are typically built as a combination of appliances accomplishing specific tasks and connected according to well defined architectures that influence the overall effectiveness of the solution. This approach is rather static and complex, often vendor dependent and leaves very limited room to flexibility. Last but not least, it does not provide any overall management and/or orchestration capability.

The coupling of NFV and SDN opens up the opportunity to re-think from scratch the way network services are designed. Hardware components are replaced with Virtual Network Functions (VNFs) talking over Virtual Network Infrastructures (VNIs). In both cases either deployment and connectivity acquire a degree of programmability never seen before. The VNFs may be activated and de-activated almost on the fly and may also be *moved* from a cloud infrastructure to another at will, as well as the connectivity among them may be re-designed on the fly by means of SDN.

In this context the Service Function Chain (SFC) may be defined as the the sequence of VNFs that a packet (or flow of packets) must traverse. A simple example of what this means is shown in Fig. 8.1, for the two different cases of many VNFs on the same server and of many VNFs on different servers on the same cloud infrastructure.

SFC may also be extended to scenarios where the VNFs may be located in different data centers and/or in different network domains. This is shown with a rather general example in Fig 8.2. The Figure shows that SFCs may be accomplished with VNFs distributed along the network path and that a given SFC may be spatially diverse from another with similar objectives, while sharing some of the VNFs (for instance flows $f_1(t_0)$ and $f_2(t_0)$ share the same VNFs in the edge data center at the source and then follow different paths in the core).

9. Intent-based Networking

Molka Gharbaoui[1], Chiara Contoli[2], Luis M. Contreras[3], Barbara Martini[1], Walter Cerroni[2]

[1]CNIT, Pisa - Italy

[2]University of Bologna - Italy

[3]Telefónica I+D - Spain

Abstract: *Automated network resource management represents a new research frontier that still has to be fully explored to replace manual and error-prone tasks and make network configuration more secure and dynamic. In this context, Intent-based Networking is considered a promising solution that allows the adoption of high level requirement specifications while automating the network configuration. This chapter introduces the concept of Intent-based Networking and outlines its main features. It also presents the current standardization efforts to formalize it and gives an overview of the different applicative areas that have implemented the intent concept so far.*

9.1 Introduction to Intent-based Networking

The so-called *network softwarization* process represents the shift that has been taking place in the last decade towards an unprecedentedly dominant role of software in communication networks. Among the many advantages that such an approach can bring to network and communication service management, one of the most relevant features is network *programmability*, i.e. the possibility to conceive the network infrastructure, as well as the computing resources involved in service delivery, as a general purpose entity that can receive instructions by means of relevant Application Programming Interfaces (APIs). Such APIs are typically offered by northbound interfaces of existing Software Defined Networking (SDN) controllers and Network Functions Virtualisation (NFV) orchestrators, and as such the level of abstraction they provide depends on the specific solution adopted by the underlying platforms.

In order to understand the role of abstractions, let us consider the example of a NFV service chain composed by a number of Virtualized Network Functions (VNFs) running on a few compute nodes in a cloud environment, as illustrated in Figure. 9.1. The service chain is deployed by programming both cloud and network controllers so that the traffic exchanged by the user traverses an ordered series of network functions specifically required by the service itself, e.g. VNF 1, VNF 5 and VNF 6 in this example. The cloud controller is in charge of managing the computing resources and the VNF instance life-cycle, corresponding to the tasks performed by NFV orchestration components such as Virtualized Infrastructure Manager (VIM) and VNF Manager (VNFM). The network controller, typically an SDN controller, is needed to program the network switches (both

10. Security Implications, Issues and Approaches in Programmable Networks

Raffaele Bolla[1,2], Alessandro Carrega[1], Franco Davoli[1,2], Matteo Repetto[1,3] and Odnan Ref Sanchez[1]

[1]CNIT S2N National Laboratory, Genova - Italy

[2]University of Genova - Italy

[3]IMATI-CNR, Genova - Italy

Abstract: *The emerging trend in network softwarization has led to a programmable networking architecture that improves the traditional control of network systems from hardware-based configurations to a fully-softwarized approach. The benefits from this transition are numerous; however, the impact on network security aspects may turn out to be negative, unless proper changes are introduced in the way security applications are designed and deployed. In particular, the multi-tenant environment, the dynamic nature of current Network Services, and the ongoing integration of software-intensive embedded systems and global communication networks into Cyber-Physical Systems have introduced further security considerations that need to be addressed. We examine the main motivations to go beyond the traditional "security perimeter" vision and the current trends in cybersecurity leveraging network programmability. Then, we examine a service-centric architectural framework that adopts centralized management to ensure end-to-end security, by gathering security context information from "light" local agents deployed on the service functions and by utilizing state-of-the-art technologies for external threat protection. We conclude the chapter with the discussion of a cyber-range approach to test the security of virtualized networking environments.*

10.1 Introduction

Software Defined Networking (SDN) and Network Functions Virtualization (NFV) are the key technologies that enable programmable networks [1, 2, 3]. The evolution towards the programmable network paradigm has led to improved efficiency and flexibility of traditional network services. Together with cloud environments, these technologies are seen as the innovation wave leading the softwarization that will revolutionize the Information and Communications Technology (ICT) and telecommunications industry [3]. Upcoming technologies such as 5G and beyond are expected to further accelerate this transition, by effectively integrating computing, storage, and communication resources in large pervasive environments.

On the other hand, evolving business models and the large potential behind Cyber-Physical Systems (CPSs) are fostering the transition from monolithic to modular architectures, spanning multiple administrative and business domains, as shown in Figure 10.1.

11. Software Acceleration Techniques for High-speed Programmable Networks

Leonardo Linguaglossa

Télécom Paris - France

Abstract: *Network programmability has provided an effective approach to enable innovation in network systems. By replacing static, expensive middleboxes with equivalent pieces of software implementing the same functionality, operators can significantly reduce their CAPEX/OPEX expenditures, and engineers can rapidly design, test and deploy novel architectures and services, thus reducing the time-to-market for network applications. However, the flexibility provided by software solutions comes at a cost: purpose-specific hardware has the clear advantage of optimized performance (i.e., throughput, latency) with respect to pure software-based solutions. The introduction of software acceleration techniques represented an essential step towards the increasing popularity of the SDN/NFV paradigm shift, by reducing the performance gap between hardware-based and software-based systems. Thanks to such techniques, modern software-networking solutions can operate at multi-10Gbps rate (up to hundreds of Gbps) on commodity servers equipped with regular COTS components. In this chapter, we cover the aspects related to software acceleration techniques in a bottom-up fashion: we first provide an overview of high-speed software networking on COTS architectures, and we then explore the evolution of softwarized networking by focusing on performance acceleration and the design space for high-speed programmable networks.*

11.1 Introduction

In the last decades, the networking industry had experienced a major architectural shift towards the softwarization of network devices [1]. Besides the growing popularity of programmable hardware such as OpenFlow switches [2, 3] or P4 switches [4, 5], commonly adopted Network Interface Cards (NICs) are being replaced with more programmable counterparts known as SmartNICs [6]. In line with this trend, network functions are being executed in pure software on top of commodity servers, rather than using static expensive middleboxes [7].

Together with the growing expansion of cloud appliances [8], current trends show a tremendous amount of network functionalities being executed via software instead of hardware. This context has provided the strong foundation for the evolution of novel network paradigms such as Software Defined Networking (SDN), or Network Function Virtualization (NFV). Some efforts to bring software-programmability in the network environment date back to the end of the twentieth century, as for the Click modular router [9], one of the first frameworks that allowed users to map network functions into pieces of software, thus originating the shift from hardware-specific components to more

12. 5G Radio Access Network Virtualization

Justine Cris Borromeo, Koteswararao Kondepu, Luca Valcarenghi

Scuola Superiore Sant'Anna - Italy

Abstract: *5G and beyond networks are expected to be based on a cloud/virtualized Radio Access Network (vRAN). In vRAN the next generation eNodeB (gNB) is decoupled in Radio Unit (RU), Distributed Unit (DU), and Central Unit (CU). RU and DU are connected through the so called fronthaul interface while DU and CU are connected through the so called midhaul. DU and CU are virtualised (i.e., running in containers or virtual machines). Usually, the fronthaul interface requires a high-bandwidth and low latency connection while midhaul requirements are less stringent. However such requirements depend on the functions that are implemented in the DU and in the CU (i.e., their functional split). Additional requirements might also come from the the vertical applications exploiting the 5G network that might be less stringent than the one imposed, for example, by the midhaul. The virtualization of the RAN functions might heavily impact vRAN latency and jitter performance. This chapter provides first an overview of possible implementation of vRAN functions. Then, it experimentally evaluates whether latency specification presented by 3GPP standards are met in the fronthaul/midhaul link of a vRAN implementation utilizing the 5G-Advanced Research on NetwOrking (ARNO-5G) testbed. Furthermore, it estimates if the virtualized RAN components (e.g. CU, DU) can afflict the fronthaul/midhaul performance when the Option 2, Option 7-1, and Option 8 splits are applied and different virtualization technologies are considered. Finally it provides possible solutions to decrease the end-to-end vRAN latency by exploiting 5G function programmable hardware acceleration.*

12.1 Introduction

The Next Generation Radio Access Network (NG-RAN) is planned to be densely deployed to address the stringent requirements of the current and future 5G applications on throughput, latency, and scalability [1]. Moreover, Network Function Virtualization (NFV) enables an easy introduction of new network services by adding dynamic programmability to network devices (e.g., as routers, switches, and applications servers) that, in turn, empowers fast, and flexible deployment of new network and management services. The exploitation of Virtual Network Functions (VNFs) is foreseen also in 5G NG-RAN [2] and Next Generation Core (i.e., NG Core) [3]. The VNF also has an important role in the easy and fast deployment of the 5G network services and better service agility [4].

The NG-RAN exploits the NFV and VNF concepts to transform the traditional network into a virtualized Radio Access Network (vRAN). The vRAN provides different possibilities to decompose the traditional RAN stack into three different components,

13. Optical Networks Evolution: From Programmability to Machine Learning and Spatial Division Multiplexing

Filippo Cugini[1], Massimo Tornatore[2], Nicola Sambo[3], Cristina Rottondi[4], Andrea Marotta[5]

[1]CNIT, Pisa - Italy

[2]Politecnico di Milano - Italy

[3]Scuola Superiore Sant'Anna, Pisa - Italy

[4]Politecnico di Torino - Italy

[5]University of L'Aquila - Italy

Abstract: *This chapter presents the latest evolution of network programmability in the context of optical transport infrastructures. In particular, it provides an overview of the most recent data and control plane optical technologies enabling dynamic and vendor-neutral configuration. The chapter then introduces the applicability of machine learning in programmable optical networking and it provides future directions in the evolution towards spatial division multiplexing (SDM).*

13.1 Introduction

This chapter focuses on the programmability of optical networks. It first provides an overview of the most recent optical data plane devices and technologies that constitute the underlying infrastructure for optical connectivity. In particular, Sect. 13.2 presents the most advanced devices for programmable transmission and switching in the optical layer. In Sect. 13.3, these devices are considered in the framework of network disaggregation, decoupling hardware and software control so to enable vendor-neutral configuration and management. Sect. 13.4 presents selected examples of programmable disaggregated optical network operations, while Sect. 13.5 introduces new trends in programmable optical networking based on machine learning. Finally, Sect. 13.6 provides insights on the evolution of optical networks towards spatial division multiplexing (SDM). This section also provides a brief overview of the worldwide unique SDM infrastructure of the CNIT Lab in the city of L'Aquila.

14. Software Defined Wide Area Networking (SD-WAN)

Sebastian Troia, Guido Maier

Politecnico di Milano - Italy

Abstract: *A reliable Wide Area Network (WAN) has become a necessity for businesses to transmit critical data across multiple branches and to increase their revenues. Many solutions and different network structures have been proposed over years such as leased lines, Frame Relay or Multi-Protocol Label Switching Virtual Private Networks (MPLS VPN). Each solution is intended to be better than its predecessors in terms of reliability and Quality of Service (QoS). Software Defined Wide Area Networking (SD-WAN) is an emerging paradigm that introduces the advantages of software defined networking (SDN) into enterprise networking. SD-WAN can support differentiated services over public WAN by dynamically changing the flow forwarding rules over an overlay network based on monitoring data and service requirements. Thanks to these capabilities, it is possible to overcome high cost of guaranteed QoS services such as MPLS. Most of the SD-WAN solutions commercially available today are based on proprietary controllers and proprietary Customer Provider Edge (CPE) devises. In this chapter, we present an implementation of SD-WAN based on open source components such as OpenDaylight SDN controller and Open Virtual Switch (OvS). This work targets the monitoring features of an SD-WAN solution and explores active and passive monitoring approaches to understand their advantages and limitations. Our implementation provides an overlay WAN with controlled performance in terms of delay and losses over low-cost Internet connectivity.*

14.1 Introduction

The rapid evolution of Information Technologies (IT) increases consistently the demand of higher capacity and higher quality networks. Enterprise users can gain significant advantages from network control and configuration operations. This is becoming more and more important as organizations adopt new technologies that continue to push networks. Enterprises Networking (EN) has become more crucial with advancing technologies and their contribution to the enterprise's revenues. Many different service models have been proposed to enhance Wide Area Networks (WAN) in terms of cost and quality. Traditional network services such as leased lines, Asynchronous Transfer Mode (ATM), Frame Relay (FR), Internet or Multi-Protocol Label Switching (MPLS) have different advantages and disadvantages. We can refer to MPLS as the milestone of EN. Despite MPLS provides guaranteed Quality of Service (QoS), there is a trade-off between cost and quality. High cost of MPLS pushes enterprises to use Internet/broadband services. Software Defined Wide Area Networking (SD-WAN) can achieve QoS without using MPLS and with low cost by taking advantage of broadband Internet [1]. SD-WAN is a revolutionary way to

15. Segment Routing over IPv6 (SRv6)

Francesco Lombardo[1], Carmine Scarpitta[1,2], Andrea Mayer[1], Paolo Lungaroni[3], Pier Luigi Ventre[4], Stefano Salsano[1] Ahmed Abdelsalam[5]

[1]University of Roma “Tor Vergata” - Italy

[2]GARR Consortium - Italy

[3]CNIT - Italy

[4]Open Networking Foundation - USA

[5]Cisco Systems - USA

Abstract: *The Segment Routing (SR) architecture is based on the loose source routing concept: forwarding and service instructions can be added to packet headers by a source node. SRv6 is the IPv6 based instantiation of the SR architecture. SRv6 benefits of the great scalability properties of SR coming from the source routing approach, and adds unprecedented versatility thanks to the use of IPv6. SRv6 can be used as transport solution (“underlay”) and/or as service layer (“overlay”), in IP backbones, access networks and data center networks. The first part of this chapter provides a short introduction to SRv6. The second part of the chapter discusses an open source ecosystem for SRv6, developed by the ROSE project. The ecosystem includes the SRv6 data plane based on Linux networking, an SDN based control plane, an SRv6 performance monitoring solution (SRv6-PM) and a system for collecting monitoring data based on big data tools.*

15.1 Introduction on SRv6

The evolution of IP networks is driven by very challenging requirements, such as: i) the integration of tens of billions devices and of tens of millions of services in the cloud; ii) the support of very high capacity and/or very low latency interconnections, which may need to be accompanied with high availability and adequate security/privacy. Nonwithstanding these challenging requirements, network operators needs also to reduce the operational and capital expenditures associated with the provisioning of their services. The softwarization of the network is obviously one of the answers to these trends, helping to cut costs and to provision a more scalable network. As a fundamental complement to the network softwarization, we present the SRv6 network architecture. SRv6 offers great benefits to network operators, service providers and users for the support of advanced services with a simplification of the network architecture and operations compared to the existing solutions for IP transport backbones (e.g. MPLS, SR-MPLS).

16. The P4 Ecosystem for Network Programmability

Daniele Moro[1], Davide Sanvito[2], Carmelo Cascone[3], Giacomo Verticale[1], Antonio Capone[1]

[1]Politecnico di Milano - Italy

[2]NEC Laboratories Europe - Germany

[3]Open Networking Foundation - USA

Abstract: *The new wave of SDN products acknowledges that preventing ossification of the network requires data plane programmability, in addition to the control plane programmability that determined the success of Network Operating Systems such as ONOS or OpenDaylight. The P4 language is emerging as the de-facto standard for data plane programmability. In this chapter, we provide an introduction to the P4 language and tool-chain and review some of the most interesting use cases of the programmable data plane. As an example, we discuss more in detail the Software Defined Broadband Network Gateway and the In-band Network Telemetry framework.*

16.1 Abstracting a switch through a language

In the early SDN era, programmability was an hallmark of the control plane, while the data plane was considered as a fixed function object controlled via an open protocol (e.g., OpenFlow). Hence, development focused on network controllers (e.g., ONOS [1], OpenDaylight [2], or Ryu [3]) and on the applications running on top of them. With disaggregation, network devices such as router and switches, are broken up into a data plane component and a control plane component, and each component can be purchased separately [4]. The new generation of SDN acknowledges that preventing ossification of the network also requires data plane programmability [5]. In this context, P4 aims at being the programming language for the data plane.

High-speed switches and routers process multiple packets at the same time, using a multi-stage pipeline approach. This architecture enables to scale-up the amount of traffic that a single switch can treat. Current state-of-the-art data-center switches can process more than 10 Tbps of traffic. Each stage of the pipeline can be configured to operate on different combinations of header fields, such that the forwarding decision is built incrementally while packets traverse the pipeline [6, 7].

A fundamental characteristics of the switch pipeline implementation is whether the stages are fixed-function or programmable. Fixed-function stages can only operate on protocols and headers known a priori and defined by the vendor, while programmable stages can be dynamically programmed to process user-defined header fields.

17. Extended Berkeley Packet Filter

Sebastiano Miano, Fulvio Risso

Politecnico di Torino - Italy

Abstract: *The extended Berkeley Packet Filter (eBPF) is an in-kernel virtual CPU for packet filtering that has been introduced in Linux in 2013. While originally made to capture and process network traffic, eBPF has introduced also the capability to trace and inspect any kernel function, which rapidly became one of the most successful features nowadays, curiously used even more used than traditional network processing capabilities. This Chapter will provide an architectural view of eBPF, it will give an insight on its tracing capabilities, then it will explore in more depth the case for eBPF technology applied to packet processing.*

17.1 Introduction

The Berkeley Packet Filter (BPF) was originally proposed by Steven McCanne and Van Jacobson in 1992 [1] as an efficient mechanism for **packet capture** in Berkeley Unix, where user-level programs need access to raw network traffic to implement complex network diagnostic and analysis tools. Their idea was to implement a user-defined *packet filtering* mechanism inside the Unix kernel that was able to discard unwanted packets as early as possible, avoiding unnecessary copies across kernel and user space that may result in a huge performance overhead. In particular, they introduced a new kernel architecture for packet capture that was able to send to user space only the relevant packets thanks to *(i)* a set of instructions and an in-kernel virtual CPU (vCPU) for executing programs written in *BPF assembly language*, which selects only packets relevant for the specific requesting application and *(ii)* a simple per-application buffer used to copy only the desired bytes of the packet to the application itself.

Given its success and unprecedented performance advantages, this architecture has been ported over the years to many operating systems such as Solaris, Windows and Linux. It was part of the Linux kernel since version 2.5, with several popular applications and tools (e.g., Wireshark, tcpdump, ntop, etc.) using it.

In 2014, Alexei Staravoitov proposed an extended version of BPF [2], called **extended BPF (eBPF)**, which includes improvements to both the BPF vCPU and its overall architecture, making it more optimized for modern hardware, and increasing the possible usages of BPF in software products. The original number of registers was increased from 2 to 11, and their width was changed from 32 to 64 bits, opening the possibility to create more complex applications. In eBPF, packets are no longer a copy of the original data, hence eBPF applications can directly operate and modify the packet content. Moreover, an enhanced version of the JIT compiler that was first introduced in Windows in 2001 [3] made eBPF 5x times faster than, what is now called, “classic” BPF (cBPF). As a consequence, today the cBPF is not used anymore, and legacy applications that still rely on the cBPF syntax are automatically converted from the cBPF bytecode to the eBPF.