Programmable, Expressive, Scalable, and Agile Service Function Chaining for Edge Data Centers

Cristina Klippel Dominicini[1], **Magnos Martinello**[2], **Moisés Renato Nunes Ribeiro**[2]

[1]Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo (IFES) – Serra/ES

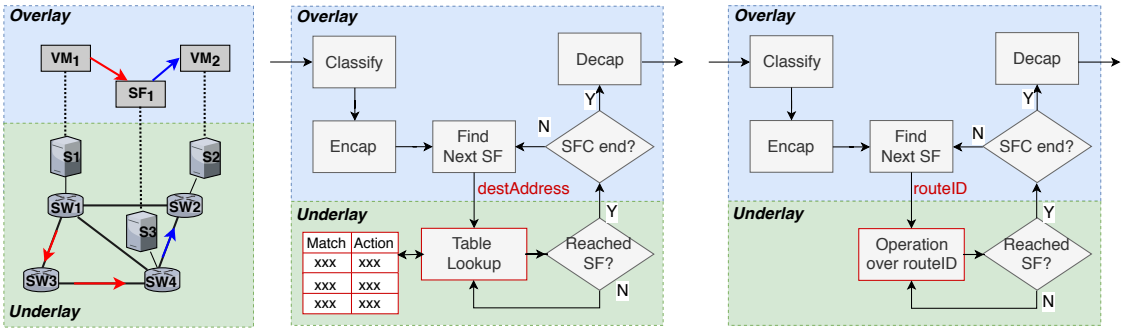
[2]Universidade Federal do Espírito Santo (UFES) – Vitória/ES

cristina.dominicini@ifes.edu.br, magnos@inf.ufes.br, moises@ele.ufes.br

Abstract. *Edge computing transfers processing power from large remote data centers (DCs) to distributed DCs at the edge of the network. This shift requires the ability to provide network functions virtualization (NFV) solutions that can efficiently manage and combine a large number of dynamic services in a resource-constrained DC. However, the routing mechanisms of traditional data center networks are not adequate for the dynamic composition of these services, because they are complex, rigid, subject to large delays in the propagation of control information, and limited by the size of switches' routing tables. In addition, traditional service function chaining (SFC) solutions in the service overlay are often decoupled from routing decisions in the network underlay, and restrict path selection options by traffic engineering. In this way, the NFV orchestrator cannot explore the full capacity of the network to provide composite services. To tackle these issues, this thesis investigated a programmable, expressive, scalable, and agile SFC proposal that allows dynamic and efficient orchestration of the network infrastructure of edge DCs with commodity network equipment. The proposal exploits virtualization and programmability technologies of DC networks, server-centric DCs, fabric networks, and a source routing mechanism based on the residue number system (RNS). As proof-of-concept, we developed prototypes with production DC technologies, such as OpenFlow, OpenStack, Open vSwitch and P4. The results of functional and performance tests showed that the proposed SFC scheme provides mechanisms to the NFV orchestrator that allow traffic engineering to make optimized decisions in the selection of network paths. This thesis also paves the way for exploring RNS-based source routing properties in SFC schemes, which can provide features such as fast failure reaction and forwarding without packet rewrite. In a broader analysis, the student published 22 papers in journals and conferences, contributed to funding initiatives, worked on international and national research projects, supervised undergraduate students, and leaded initiatives with innovation impacts.*

1. Problem and motivation

Network functions virtualization (NFV) has attracted a lot of attention as a solution for reducing costs and providing scalable network services [Mijumbi et al. 2016]. In parallel, edge computing portends new types of services and transfer processing power from large remote data centers (DCs) to distributed DCs at the edge of the network [Mao et al. 2017]. In this scenario, the adoption of NFV brings new challenges, because it requires the efficient orchestration of a large number of dynamic services in a resource-constrained DC.



(a) SFC layers. (b) Table-based SFC. (c) SFC with source routing.

Figure 1. SFC problem. Adapted from [Dominicini et al. 2020].

One of the main challenges is how to enable the composition of customized services by steering a large number of flows across a set of service functions (SFs) to provide service function chaining (SFC) [Medhat et al. 2017]. To provide such composite services, a service overlay topology is built "on top" of the existing network underlay topology [Quinn et al. 2018], as shown in Fig. 1(a). This is a far more complex challenge than routing between two end nodes, since it involves capturing, classifying, and steering the traffic for each virtualized SF. To ensure maximum performance, the network controller or distributed agents in charge of traffic engineering should be able to select among the paths in the underlay according to dynamic demands [Jyothi et al. 2015].

Fig. 1(b) shows how most of the current SFC solutions perform traffic steering based on tables. After classification, the SFC encapsulation carries information used to determine the network address of the next SF to be executed. After determining the address of the current SF, the packet is delivered to the underlying routing mechanism, where the forwarding is executed using per-hop table lookups based on the destination address until it reaches the SF. At this moment the encapsulation information is checked again to find the address of the next SF and this loop proceeds until the chain terminates, when the packet is decapsulated and delivered to destination.

However, there are fundamental problems in this approach that compromise efficient network orchestration: (i) the commonly adopted underlying routing mechanisms cannot represent all the possible paths between two endpoints due to limited capacity of switch forwarding tables [Jyothi et al. 2015]; (ii) as they are based on per-hop table lookup, the overhead to dynamically change a path is high, because it may involve modifications in all the nodes in the path [Dominicini et al. 2020]; and (iii) traffic engineering decisions are made in a decoupled way considering placement, chaining, and routing [Quinn et al. 2018]. On the other hand, emerging networking architectures and equipment are allowing for softwarization of DCNs and flexible dataplane programmability at line rate, both at the switch and the network interfaces, opening up unprecedented opportunities for the development of innovative networking solutions [McCauley et al. 2019].

2. Objectives

To tackle the described problem, this thesis investigated a SFC solution that can provide greater synergy between the NFV orchestration and the underlying DC infrastructure. More specifically, this work aims at designing a **programmable, expressive, scalable, and agile SFC solution** that enables dynamic and efficient orchestration of the network underlay of **edge data centers with commercial off-the-shelf (COTS) equipment**. As

a specific objective, this work aimed to implement prototypes of all the proposals for functional and performance tests in a close-to-production environment.

3. Hypotheses, Proposal and Contributions

This thesis is based on the following hypotheses:

- **Algorithmic forwarding can replace forwarding tables in the design of scalable SFC solutions.** Differently from table-based methods, it defines the output port using a fixed logic based on information about the current and destination nodes [Dominicini et al. 2020], and can reduce the number of forwarding states.
- **Strict source routing (SSR) allows to specify any underlay path between SFs and design agile, and expressive SFC solutions.** In SSR, the source (or edge) can specify all the path to destination in the packet header [Jin et al. 2016]. Thus, it reduces the control plane overhead to manage paths when compared with per-hop methods, and can achieve optimal throughput performance [Jyothi et al. 2015].
- **The residue number system (RNS) can provide SSR with algorithmic forwarding for SFC.** In RNS-based SSR, the route identifier is calculated according to the Chinese remainder theorem (CRT), and the nodes receive identifiers as pairwise co-prime numbers. Then, the output port in each node can be directly calculated by the *modulo* (remainder of division) of the route identifier of the packet by the node identifier, without using tables [Martinello et al. 2014].
- **SDN and fabric paradigms offer a programmable solution for SFC.** Using SDN, a logically centralized controller can dynamically program network elements [Martinello et al. 2014]. Besides, fabric networks separate the network in edge elements that provides SFC classification, and core elements that are responsible for efficient packet transport [Casado et al. 2012].

Fig. 1(c) illustrates how our proposal uses SSR for determining the specific path of each SFC segment. The path information is inserted in the encapsulation stage in the form of a route label. This identifier is passed to the underlying routing layer, where each hop perform a simple operation over the route label. If the path needs to change to fit any dynamic demand, the source only needs to encapsulate a new route label, and intermediary nodes will continue to operate over the received identifier without any modification.

In summary, this thesis proposed, implemented, and evaluated three interrelated solutions (see Fig. 2): (i) **VirtPhy**, a NFV orchestration architecture for server-centric topologies; (ii) **KeySFC**, a topology-independent SFC scheme that uses RNS-based SSR and network fabric; and (iii) **PolKA**, an algorithmic SSR mechanism based on RNS and Galois Field polynomials that can be implemented over commodity network hardware. The combination of these three solutions builds a powerful framework to enable SFC.

The first observation of this research was that current NFV solutions are tailored to network-centric DCs, neglecting the potential contribution of hybrid and server-centric DCs. The second observation was that SFC approaches usually base their routing mechanisms in tables, which restricts the traffic engineering mechanisms for path selection. Based on this, we proposed the **VirtPhy** architecture, whose main contributions were:

- We proposed how server-centric DCNs can fit in the NFV ETSI reference architecture and provide efficient SFC with algorithmic routing, while eliminating the need for tables and hardware switches.

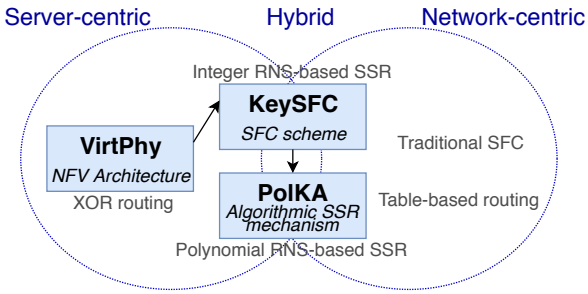


Figure 2. Overview of solutions proposed in this Thesis.

- We applied the concept of server-based network fabric to SFC: core software switches execute algorithmic forwarding, and SDN-enabled edge software switches use flow tables only for classifying SFC flows.
- We explored routing mechanisms for SFC that consider characteristics of the underlay network topology (e.g., XOR routing in the hypercube topology).
- We implemented and evaluated a hypercube prototype using OpenStack and OpenFlow technologies, proving it is viable to deploy these ideas in production DCs. The software switches were implemented using Open vSwitch (OvS), and the datapath in core nodes was modified to perform algorithmic XOR routing.

A limitation of VirtPhy is the dependence on the routing algorithms provided by the underlay topology. Furthermore, the choice of using per-hop algorithmic forwarding does not allow the full path specification by the traffic engineering. To tackle these issues, we proposed an SFC scheme, called **KeySFC**, that explores SSR and uses a topology-independent algorithmic forwarding mechanism based on RNS [Martinello et al. 2014]. For each SFC segment, SDN-enabled edge software switches classify flows for SFC and embed a RNS identifier in the packet header, which encodes the information about the path. At core switches, the forwarding engine executes a *modulo* operation over the path identifiers and the node identifiers to discover the output ports. In this way, it eliminates table lookups in the core, reducing the number of forwarding states and control plane overhead. The main contributions of KeySFC were:

- We proposed a SFC scheme that uses RNS-based SSR, and supports different DC networks (server-centric, network centric, or hybrid).
- We proposed an SFC scheme that offers the following features to traffic engineering for each chain segment: (i) selection of any available path, (ii) specification of all the forwarding elements in the path, and (iii) agile path migration.
- We implemented KeySFC in a DC testbed orchestrated by OpenStack, demonstrating its feasibility in production. Results for latency, jitter, and throughput indicate that our RNS-based SSR solution can provide efficient packet transport.

The data plane implementation of KeySFC modifies the OvS datapath to perform the *modulo* operation, but the use of software switches has performance limitations for low latency and jitter scenarios. Also, the *modulo* operation does not map to the instruction set of commodity network hardware. The work in [Liberato et al. 2018] demonstrated it is possible to achieve high performance with NetFPGAs, but it requires specialized hardware. To meet both cost and performance requirements, we proposed **PolKA**, a SSR approach that can be implemented in programmable network switches. To this end, we reinterpreted the integer RNS-based SSR mechanism to a binary polynomial arithmetic using Galois field (GF) of order 2 [Shoup 2009]. The main contributions of PolKA were:

- We brought RNS-based SSR expressiveness closer to elementary binary operations. The immediate benefit of this approach is to enable the reuse in RNS-based SSR of commodity embedded network functions that are based on polynomial arithmetic. Also, to the best of our knowledge, this is the first time a work applies the CRT theorem with Galois field to solve routing problems.
- We developed a technique that allows the execution of the polynomial *modulo* operation by reusing common CRC (cyclic redundancy check) operations, which are also based on GF(2) polynomials and supported by P4 standard architectures.
- We implemented emulated (Mininet) and hardware (SmartNICs) prototypes using P4. The tests showed that it is possible to take advantage of RNS properties using PolKA without loosing performance when compared to traditional SSR methods.
- We integrated PolKA as the SSR mechanism of KeySFC, demonstrating our SFC solution can be implemented in commercial P4-enabled network equipment.

4. Related works

4.1. SSR

Although many works have investigated the benefits of SSR over table-based approaches, most of them use Port Switching (PS) [Jin et al. 2016, Jyothi et al. 2015]. In PS, the route identifier is represented as a stack of ports or addresses and the forwarding operation is a stack pop. This thesis argues that RNS-based SSR presents interesting properties that do not exist in PS, such as forwarding without header modifications and resilience. Moreover, it offers fully stateless routing, because it does not update any state in the route label or in core nodes. Related works on RNS-based SSR developed fast failure reaction mechanisms [Gomes et al. 2016], integrated this scheme with SDN [Martinello et al. 2014], investigated techniques to improve the scalability of the route identifier [Ren et al. 2018], and evaluated latency constraints for multicast in DCs [Liberato et al. 2018]. However, these works rely on integer RNS arithmetic, and the integer *modulo* operation cannot be implemented in current commodity network hardware. Therefore, they either use software implementations [Martinello et al. 2014], or depend on synthesizing integer division to ASICs or NetFPGAs [Liberato et al. 2018]. Other works explored polynomial RNS [Shoup 2009], but haven't applied these concepts to routing. To the best of our knowledge, PolKA is the first work to combine RNS with GF(2) polynomials to solve routing problems, and to allow the reuse of CRC hardware for routing.

4.2. SFC

The SFC problem can be divided in three phases: service modeling, resource allocation, and traffic steering [Hantouti et al. 2018]. In the service modeling phase, a service request generates a graph that is given as input to the resource allocation phase. Then, the decisions from this phase are deployed in the network infrastructure using traffic steering mechanisms, which are the focus of this thesis. Many related works employ NFV and SDN to enable SFC, as surveyed in [Medhat et al. 2017, Hantouti et al. 2018]. Most commercial SFC solutions, such as OpenDayLight, ONOS, Openstack, and OPNFV, are based on the architecture proposed by RFC 7665 and the network service headers (NSH) protocol [Quinn et al. 2018]. The NSH protocol uses per-hop tables both for discovering the next SF and for routing packets, which may lead to scalability and agility issues. In addition, attachment of headers expands packet size, causing an increase of traffic, and potential fragmentation problems. Also, routing decisions are decoupled from chaining decisions and executed by different mechanisms [Quinn et al. 2018].

Table 1. Publications- Journals

#	Class	Qualis/IF*	Publication
01	A	A1	DOMINICINI, C. K., et al. *KeySFC: Agile Traffic Steering using Strict Source Routing for Enabling Efficient Traffic Engineering*. In Computer Networks, vol. 167, p. 106975, 2020.
02	C	A1	BOTH, C., et al. *FUTEBOL Control Framework: Enabling Experimentation in Convergent Optical, Wireless, and Cloud Infrastructures*. In *IEEE Communications Magazine*, vol. 57, no. 10, pp. 56-62, 2019.
03	C	B1	DO CARMO, A. P., et al. *Programmable intelligent spaces for Industry 4.0: Indoor visual localization driving attocell networks.* Transactions on Emerging Telecommunications Technologies, v. 3, p. 20-40, 2019.
04	A	2.217	FRASCOLLA, V., et al. *Optimizing C-RAN Backhaul Topologies: A Resilience-Oriented Approach Using Graph Invariants.* Applied Sciences Basel, v. 9, p. 136, 2019.
05	A, B	B1	DOMINICINI, C. K., et al. *VirtPhy: Fully Programmable NFV Orchestration Architecture for Edge Data Centers.* IEEE Transactions on Network and Service Management, v. 14, p. 1-1, 2017.
06	C	B1	MAFIOLETTI, D. R., et al. *Latency Measurement as a Virtualized Network Function using Metherxis.* ACM SIGCOMM Computer Communication Review, 46(4), 2016.

** Qualis Triennium 2013-2016 or Impact Factor (IF).*

Other popular SFC method is Segment Routing [Abdullah et al. 2019], in which an ordered list of segments can be encoded as a stack of MPLS labels. The next segment is popped from the stack, and a lookup operation in the forwarding table is performed in each hop. However, although the source may need to represent long explicit paths, most MPLS equipment only support a limited stack depth (about 3 to 5 labels). Therefore, Segment Routing commonly specifies a subset of network elements in the path, and delegates the routing between elements to the underlying network that employs ECMP shortest paths, which may lead to inefficient traffic distribution [Abdullah et al. 2019].

Another alternative is to include a chain identifier in existing packet fields, like MAC addresses, and use some type of algorithmic forwarding to operate over this identifier. VirtPhy [Dominicini et al. 2017] is an example of this approach that uses a high performance forwarding mechanism based on XOR operations, but it does not allow the full specification of underlay paths. SFC approaches that use RNS-based SSR are also examples of this approach, as proposed by CRT-Chain [Ren et al. 2018] and KeySFC [Dominicini et al. 2020]. In CRT-Chain, the authors proposed a scheme for SFC using one label for routing in the physical layer, and one for routing between the SFs in an overlay layer. However, their simulated results focused only on reducing the labels length, their proposal has performance limitations, and they did not specify how to implement data and control planes [Dominicini et al. 2020].

All related works are tailored to network-centric topologies. On the other hand, VirtPhy supports server-centric topologies, and KeySFC supports any DC topology. In contrast to Segment Routing and NSH, KeySFC does not restrict the path selection by traffic engineering, eliminates forwarding tables, and performs simple operations over route identifiers for forwarding decisions. Furthermore, to the best of our knowledge, KeySFC is the first work to propose a SFC scheme using RNS-based SSR that is implementable and efficient for DC networks. Finally, we integrated PolKA with KeySFC to enable the exploitation of polynomial RNS-based SSR for SFC in commercial equipment.

5. Publications and other results

Tables 1 and 2 list the 22 publications produced during this thesis for journals and conferences, respectively, in reverse chronological order according to the following classification: (A) papers authored as first or second author, (B) papers co-authored with supervised students, (C) papers co-authored as collaborator. In addition, the PhD student supervised 3 undergraduate students, produced 5 papers jointly with them, and two of these students are now co-supervised by her in a Master's program. Moreover, she was awarded with travel grants for the conferences: ACM SIGCOMM 2016, IEEE NFV-SDN 2016, ACM

Table 2. Publications - Conferences

#	Class	Qualis**	Publication
01	A	-	DOMINICINI, C. K., et al. *PolKA: Polynomial Keybased Architecture for Source Routing in Network Fabrics.* **Accepted at IEEE NetSoft 2020 (to appear).**
02	A	A1	MAFIOLETTI, D. R., et al. *PIaFFE: A Place-as-you-go In-network Framework for Flexible Embedding of VNFs.* **Accepted at IEEE ICC 2020 (to appear).**
03	A, B	-	DOMINICINI, C. K., et al. *KeySFC: Agile Traffic Steering using Strict Source Routing.* In ACM SOSR 2019, Poster session, San Diego, 2019.
04	A, B	B2	VALENTIM, R. , et al. RDNA Balance: Balanceamento de Carga por Isolamento de Fluxos Elefante em Data Centers com Roteamento na Origem. In: SBRC 2019, Gramado, 2019.
05	C, B	B2	CASTANHO, M. S., et al. *Cadeia Aberta: Arquitetura para SFC em Kernel Usando eBPF.* In: SBRC 2019, Gramado, 2019.
06	A, B	A2	CASTANHO, M., et al. *PhantomSFC: A Fully Virtualized and Agnostic Service Function Chaining Architecture*. In IEEE ISCC 2018, Natal, 2018.
07	C	-	MARQUES, P. et al. *Optical and wireless network convergence in 5G systems - an experimental approach.* In IEEE CAMAD 2018, Barcelona, pp. 1-5, 2018.
08	A	-	DOMINICINI, C. K., et al. *Enabling Experimental Research Through Converged Orchestration of Optical Wireless and Cloud Domains*. In EUCNC 5G and Beyond 2018, Ljubljana, Slovenia, 2018.
09	C	-	CERAVOLO, I., et al. O2CMF: *Experiment-as-a-Service for Agile Fed4Fire Deployment of Programmable NFV.* In OFC 2018, San Diego. SDN/NFV Demonstration Zone, 2018.
10	C	-	GOMES, R. L., et al. *How can emerging applications benefit from EaaS in open programmable infrastructures?* In IEEE S3C, Natal-RN, 2017.
11	C	B5	CERAVOLO, I., et al. *Proposta e Implementação de um Framework de Controle para Testbeds Federados que Integram Nuvem e SDN*. In SBRC 2017 - WPEIF, Belem, 2017.
12	A	B2	DOMINICINI, C. K., et al. *VirtPhy: A Fully Programmable Infrastructure for Efficient NFV in Small Data Centers.* IEEE NFV-SDN 2016, Palo Alto, CA, USA, November 2016.
13	A	B3	GOMES, R. R., et al. *Analytical Modeling Approach of Routing Deflection for Intra-domain Networks.* In: CSBC 2016 - WPerformance, Porto Alegre-RS, 2016.
14	C	-	MAFIOLETTI, D. R., et al. *Metherxis: Virtualized Network Functions for Micro-second Grade Latency Measurements.* In: ACM SIGCOMM Workshop LANCOMM 2016, Florianopolis-SC, 2016.
15	C	B4	GOMES, R. R., et al. *KAR: Key-for-Any-Route Resilient Routing System.* In IEEE/IFIP DSN-W, Toulouse, 2016.
16	C	B2	GUIMARAES, R. S., et al. *ROUTEOPS: Orquestração de Rotas Centrada na Operação de Sistemas Autônomos.* In SBRC 2016, Salvador-BA, 2016.

*** Qualis 2016 for Computer Science Conferences.*

SIGCOMM 2017, and ACM SOSR 2019.

Besides, as a direct result of this thesis three prototypes were built with production DC technologies, such as OpenFlow, OpenStack, OvS and P4. Furthermore, during her PhD, she helped to write research projects to get funding, and actively worked in several national and international projects, such as FUTEBOL from European Union's H2020[1], GT-NosFVeraTO from RNP, Universal projects from CNPq (432787/2016-0, 428311/2018-0), and several projects at FAPES (94/2017, 560/2018, 269/2019). In special, an undergraduate project supervised by her motivated the GT-NosFVeraTO project, which, in turn, originated a startup company called Vixphy[2]. Finally, she was also invited to join the Intel Software Innovator Program as result of her efforts to use cutting-edge networking technologies[3].

References

Abdullah, Z. N., Ahmad, I., and Hussain, I. (2019). Segment routing in software defined networks: A survey. *IEEE Communications Surveys Tutorials*, 21(1):464–486.

Casado, M. et al. (2012). Fabric: a retrospective on evolving sdn. In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 85–90. ACM.

Dominicini, C. K. et al. (2017). Virtphy: Fully programmable nfv orchestration architecture for edge data centers. *IEEE TNSM*, 14(4):817–830.

[1]http://www.ict-futebol.org.br/

[2]http://www.vixphy.com.br/

[3]https://software.intel.com/en-us/intel-software-innovators/meet-innovators

Dominicini, C. K. et al. (2020). KeySFC: Traffic steering using strict source routing for dynamic and efficient network orchestration. *Computer Networks*, 167:106975.

Gomes, R. R. et al. (2016). Kar: Key-for-any-route, a resilient routing system. In *2016 IEEE/IFIP International Conference on DSN-W*, pages 120–127.

Hantouti, H., Benamar, N., Taleb, T., and Laghrissi, A. (2018). Traffic steering for service function chaining. *IEEE Communications Surveys Tutorials*, pages 1–1.

Jin, X. et al. (2016). Your data center switch is trying too hard. In *Proceedings of the ACM SOSR*, pages 12:1–12:6, New York, NY, USA. ACM.

Jyothi, S. A. et al. (2015). Towards a flexible data center fabric with source routing. In *Proceedings of the ACM SOSR*, pages 10:1–10:8, New York, NY, USA. ACM.

Liberato, A. et al. (2018). RDNA: Residue-Defined Networking Architecture Enabling Ultra-Reliable Low-Latency Datacenters. *IEEE TNSM*.

Mao, Y. et al. (2017). A Survey on Mobile Edge Computing: The Communication Perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358.

Martinello, M. et al. (2014). Keyflow: a prototype for evolving sdn toward core network fabrics. *IEEE Network*, 28(2):12–19.

McCauley, J. et al. (2019). Thoughts on load distribution and the role of programmable switches. *SIGCOMM Comput. Commun. Rev.*, 49(1):18–23.

Medhat, A. M. et al. (2017). Service Function Chaining in Next Generation Networks: State of the Art and Research Challenges. *IEEE Communications Magazine*, 55(2):216–223.

Mijumbi, R., , et al. (2016). Management and orchestration challenges in network functions virtualization. *IEEE Communications Magazine*, 54:98–105.

Quinn, P., Elzur, U., and Pignataro, C. (2018). Network service header (nsh). RFC 8300, RFC Editor.

Ren, Y., Huang, T., Lin, K. C., and Tseng, Y. (2018). On scalable service function chaining with $\mathcal{O}(1)$ flowtable entries. In *2018 IEEE INFOCOM*, pages 702–710.

Shoup, V. (2009). *A computational introduction to number theory and algebra*. Cambridge university press.