1

Overview of Network and Service Management

Marco Mellia[1], Nur Zincir-Heywood[2], and Yixin Diao[3]

[1] Department of Electronics and Telecommunications, Politecnico di Torino, Torino, Italy
[2] Faculty of Computer Science, Dalhousie University, Halifax, Nova Scotia, Canada
[3] PebblePost, New York, NY, USA

1.1 Network and Service Management at Large

Nowadays the network, i.e. the Internet, has become a fundamental instrument to effectively support high value solutions that involve our daily life. Born to carry mainly data, today we use the Internet to watch high-definition videos, conduct video conferences, stay informed, participate in social networks, play games, buy goods, and do business. All these value-added services call for maintaining superior network service levels – where service disruption is not tolerated, and the quality of the service must be guaranteed. As a result of the many impacts of digitalization, the Internet has become increasingly complex and difficult to manage, with mobile broadband access networks able to connect billions of users at hundreds of megabit per seconds, backbone networks extending for thousands of kilometers with multi terabit per second channels, and huge datacenters hosting hundreds of thousands of servers, virtual machines, and applications.

The need for network and service management raised together with the first network concepts, with fundamentals that were defined within the International Organization for Standardization's Open Systems Interconnection (ISO/OSI) reference model [1, 2]. Telephone networks started moving to digital services in the 1970s, which created the need to manage these services automatically [3]. Computer communication technology radically changed the networking paradigm, with Transmission Control Protocol/Internet Protocol (TCP/IP) leading to the birth of the Internet as we know it today. Originally, computer network management was mostly a manual activity, in which the network administrator knew the configuration by hearth of each device and was able to quickly intervene

Communication Networks and Service Management in the Era of Artificial Intelligence and Machine Learning, First Edition. Edited by Nur Zincir-Heywood, Marco Mellia, and Yixin Diao.

in case of problems. Nowadays, with networks of billions of devices, millions of nodes, thousands of applications, network and service management has evolved to be as much as possibly automated. The advances with centralized and distributed approaches have enabled the Network Operation Center (NOC) to visualize and control the network in an as much as possible automatic fashion. Today, with the abilities to collect and process large amount of data, network and service management is facing a new stimulus toward the complete automation, with machine learning and artificial intelligence approaches that start being deployed in operations.

Network and service management fundamentally implements a control loop in which data about the status of the network is collected to be then processed in a centralized or distributed fashion to detect changes, with the goal to define which actions to implement, react, and control the changes. Figure 1.1 presents a high level overview of the overall process. From the left, data about network status is collected to continuously monitor its health. Big data technologies coupled with machine learning and artificial intelligence solutions allow to collect, analyze, and derive plans to resolve issues, which are then distributed to the network devices to implement the desired changes. In the following, we present an overview of technologies to face the monitoring and execute steps. We explicitly focus on the protocols to collect and monitor the status of the network and to distribute the management decisions. We instead leave for specific chapters the description of the algorithms and approaches which are – by definition – very dependent on the use case and on the specific technologies. Our goal in this chapter is to provide a quick overview of the latest trends in the technologies for network and service management, and to give a high-level overview of solutions in dominant scenarios

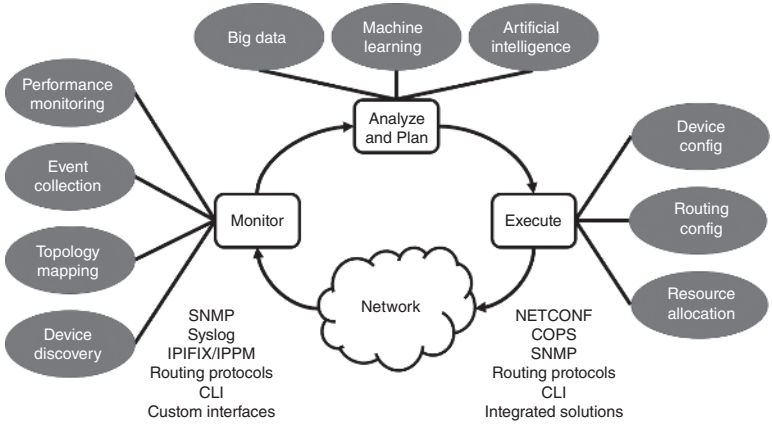

Figure 1.1 Network and service management at large.

so that the reader gets a view of the bigger picture of the problems. We leave specific solutions to the single chapters along with examples and more in-depth discussions. We focus on the Internet mainly, being it the nowadays dominant network.

1.2 Data Collection and Monitoring Protocols

Any decision process must be guided by the ability to obtain data about the status of the system. In a typical network, devices from different vendors, with different functionalities, different capabilities, different administrative domains create heterogeneous scenarios where collecting data calls for standardized instruments and tools. Often this heterogeneity produces custom solutions provided by each vendor, offering advanced and proprietary solutions to interact with the different and custom devices. Here we present an overview of the major standard protocols that allow one to collect data from network devices, leaving custom solutions out of this description.

1.2.1 SNMP Protocol Family

Original TCP/IP network management is based on the Simple Network Management Protocol (SNMP) family. SNMP standardizes the collection and organization of information about devices on an IP network. It is based on the manager/agent model with a simple request/response format. Here, the network manager issues a request and the managed agents will send responses in return. SNMP exposes management data in the form of variables organized in a Management Information Base (MIB) which describes the system status and configuration. These variables can then be remotely queried and manipulated, allowing both the collection of information and the changes in configuration – provided the manager has controlling authorization on such variables. SNMPv1 is the original version of the protocol [4]. More recent versions, SNMPv2c and SNMPv3, feature improvements in performance, flexibility, and especially security [5, 6].

Via this simple approach, an authorized agent can remotely check and change the configuration of devices under its administrative domain, propagating changes, while obtaining an updated picture of the network status. SNMP offers a means thus both to collect information from and to control the network devices, but does not provide any means to define which is the best configuration to deploy.

1.2.2 Syslog Protocol

Similarly to SNMP, the Syslog protocol family [7] offers mechanisms for collection of logging information. Initially used on Unix systems and developed since

1980, the protocol introduces a layered architecture allowing the use of any transport protocols. The Syslog protocol enables a machine to send system log messages across networks to event message collectors. It implements a push approach, where the devices send information to the collectors. The protocol is simply designed to transport and distribute these event messages, enabling the centralized collection of logs from servers, routers, and devices in general. Differently from SNMP – Syslog does not allow to distribute any configuration, which shall be achieved using other communication channels.

Messages include a facility code and a severity level. The former identifies the type of program that is logging the message (e.g. kernel, user, mail, daemon, etc.). The latter defines the urgency of the message (e.g. emergency, alert, critical, error, warning, debug, etc.). This allows for simple filtering and easy reading of the messages. When operating in a network, syslog uses a client-server paradigm, where the collector server listens for messages from clients. Born to leverage User Datagram Protocol (UDP), recent versions support TCP and Transmission Level Security (TLS) protocol for reliable and secure communications.

Syslog suffers from the lack of standard message format, so that each application supports a custom set of messages. It is common that even different software releases of the same application use different formats, thus making the parsing of the messages complicated by automatic solutions.

1.2.3 IP Flow Information eXport (IPFIX)

Both syslog and SNMP allow to collect information about the status of devices. Internet Protocol Flow Information Export (IPFIX) Protocol defines instead a means to collect in a standard way information about the traffic flowing in the network. The granularity at which it works is the flow, i.e. a group of packets having the same source and destination [8]. It defines the components involved in the measurement and reporting of information on IP flows. A Metering Process generates Flow Records; an Exporting Process transmits the information using the IPFIX protocol; and a Collecting Process receives it as IPFIX Data Records. The IPFIX protocol is a push mechanism only, and IPFIX cannot distribute configurations to the Exporters. As Syslog, it offers the means to collect information about the traffic flowing in a network, but does not provide any means to process it. Being based on traffic meters, it opens the possibility of implementing traffic profiling, traffic engineering, QoS monitoring, and intrusion detection solutions that analyze the flow-based traffic measurements and generate valuable feedback to the network managers. IPFIX is an evolution of NetFlow, a custom predecessor introduced by Cisco in 1996 to collect and monitor IP network flow information. IPFIX not only supports the Stream Control Transmission Protocol (SCTP) at the

transport layer but also allows the use of the TCP or UDP to offload the meter application.

NetFlow and IPFIX protocols are examples of "metadata-based" techniques which can provide valuable operational insight for network performance, security, and other applications. For instance, in IP networks, metadata records document the flows. In each flow record, the "who" and "whom" are IP addresses and port numbers, and the "how long" is byte and packet counts. Direct data capture and analysis of the underlying data packets themselves can also be used for network performance and security troubleshooting, e.g. exporting the raw packets. This typically involves a level of technical complexity and expense that in most situations does not produce more actionable understanding vs. an effective system for the collection and analysis of metadata comprising network flow records.

The main critical point of IPFIX is its lack of scalability, for the data collection at the exporter, and the excessive the network load at the collector. This forces often to activate packet sampling options which limits visibility.

1.2.4 IP Performance Metrics (IPPM)

Internet Protocol Performance Metrics (IPPM) is an example of a successful standardization effort [9]. It defines metrics for accurately measuring and reporting the quality, performance, and reliability of the network. These include connectivity, one-way delay and loss, round-trip delay and loss, delay variation, loss patterns, packet reordering, bulk transport capacity, and link bandwidth capacity measurements. It offers a standard and common ground to define and measure performance so that even measurements performed by different vendors and implementations shall refer to the same monitored metric. In a nutshell, it opens the ability for common performance monitoring.

Among the standard protocols, the One-Way Active Measurement Protocol and Two-Way Active Measurement Protocol (OWAMP [10] and TWAMP [11], respectively) metrics specification allows delay, loss, and reordering measurements. OWAMP can be used bi-directionally to measure one-way metrics in both directions between two network elements. However, it does not natively support round-trip or two-way measurements. The TWAMP extends the OWAMP capabilities to add two-way or round-trip measurement. Two hosts are involved in the measurement. In the case of OWAMP, the sender and the receiver collaborate actively to measure the desired performance index. For instance, to compute the one-way-delay, both take a proper timestamp of the measurement packet, at the sending and receiving time, respectively. In the TWAMP, the receiver can act as a simple reflector that just sends back (or to a third party) the probe packet sent by the sender, with no additional computation effort.

Open source and proprietary implementations are readily available for both IPv4 and IPv6 protocol stacks. These are commonly integrated in monitoring platforms [12] as well, namely Perfsonar [13] or RIPE Atlas [14].

1.2.5 Routing Protocols and Monitoring Platforms

Routing protocols are among the most successful deployed solutions to manage a network. A routing protocol specifies how routers communicate each other to exchange information that allows them to get the current network topology and compute the paths to reach possible destinations. Routing protocols give the Internet the ability to dynamically adjust to changing conditions such as topology changes, links and node failures, and congestion situations. There are two main classes of routing protocols in use on IP networks. Interior gateway protocols based distance-vector routing protocols, such as Routing Information Protocol (RIP) [15], Enhanced Interior Gateway Routing Protocol (EIGRP) [16], or based on link-state routing protocols, such as Open Shortest Path First (OSPF) [17], Intermediate System to Intermediate System IS-IS [18], are used in networks that belong to the same administrator domain, i.e. within the same Autonomous System (AS). Interior gateways protocols base their decision on the minimization of the path costs, defined as the sum of link costs. As such, they aim at minimizing the cost of routing the traffic, i.e. maximizing the performance. Exterior gateway protocols aim instead at exchanging routing information between Autonomous Systems and finding the most convenient path – in terms of Autonomous Systems – to reach the destination. Here, Border Gateway Protocol (BGP) [19] is the de facto only choice. It is a path-vector routing protocol and it makes routing decisions based on network policies and rules and not based on cost functions. BGP allows network operators to define routing policies that reflects administrative costs and political decisions in terms of agreements between Autonomous Systems.

Given the importance of optimizing exterior routing policies and the partial view that each network operator can get of the global Autonomous System (AS) level topology, several mechanisms are in place to gain visibility on the current Internet routing. Among those, the University of Oregon Route Views Project [20] leverages information provided by *collectors*, vantage points that expose their partial view of the BGP data, to create interactive maps, which are historized and made browsable via an ecosystem of tools and software that simplify the management and query of the information [21]. Thanks to Routeviews and the information exposed by BGP, it is possible to observe Internet-wide outages [22, 23], routing hijacking [24], routing anomalies [25], or check the IPv4 address space utilization [26].

All the above-mentioned routing protocols implement closed loop mechanisms – from monitoring to actions. Another category of routing protocols enable traffic engineering and network management opportunities. Among those,

Multiprotocol Label Switching (MPLS) [27] is a routing technique based on the label swapping principle. Each node along the path reads the incoming packets' label and uses it to quickly route the packets to the next hop. Before the forwarding operation, the packet label is replaced with a new label that indicates the next forwarding operation to be done at the next node. Via a concatenation of labels, packets follow a pre-computed path (a so called MPLS tunnel), which is distributed to all the nodes along the path prior the actual transmission. This on the one hand avoids complex look-ups in the routing table, and on the other hand it enables the definition of explicit and well-controlled paths that traffic flows will follow. By computing explicit tunnels is then possible to implement complex traffic engineering policies [28], setup end-to-end virtual private networks (VPNs) [29], and design specific protection mechanisms that quickly recover connectivity in case of failures [30].

1.3 Network Configuration Protocol

As said, while there has been a standardized means to collect information about the status of devices and of traffic, each vendor typically offers its own mechanisms to distribute configurations. The heterogeneity of devices, vendors, and versions makes indeed it difficult to define a common and flexible structure able to support and fit different requirements. This hampered the adoption of standard protocols, which are confined to a mostly academic design, with little deployment.

1.3.1 Standard Configuration Protocols and Approaches

The NETCONF protocol is an example of a standard mechanisms that allow to install, manipulate, and delete the configuration of network devices [31]. It uses an XML-based data encoding for the configuration data as well as the protocol messages. A key aspect of NETCONF is that it allows the functionality to closely mirror the native command-line interface of the device. It provides a standard way for authentication, data integrity, and confidentiality. For this, it depends on the underlying transport protocol for this capability. For example, connections can be encrypted in TLS or SSH, depending on the device support. Along with NETCONF, a data modeling language defining the semantics of operational and configuration data, notifications, and operations has been defined via the introduction of the YANG modeling language [32]. Neither NETCONF nor YANG ever succeed in becoming an actual standard, given the difficulty to find a common and flexible ground that fits all requirements.

The Internet Engineering Task Force (IETF) defined a general policy framework for managing, sharing, and reusing policies in a vendor-independent, interoperable, and scalable manner [33]. The Policy Core Information Model (PCIM) is an

object-oriented information model for representing policy information. It specifies two main architectural elements: the Policy Enforcement Point (PEP) and the Policy Decision Point (PDP). Policies allow an operator to specify how the network is to be configured and monitored by using a descriptive language. It allows the automation of management tasks, according to the requirements set out in the policy module. The IETF Policy Framework has been accepted by the industry as a standard-based policy management approach and has been adopted by the third Generation Partnership Project (3GPP) standardization as well.

The Common Open Policy Service (COPS) is a protocol that provides a client/server model to support policy control. The COPS specification is independent of the type of policy being provisioned (QoS, security, etc.) but focuses on the mechanisms and conventions used to distribute information between PDPs and PEPs. COPS has never been widely deployed because operators found its use of binary messages complicates the development of automated scripts for simple configuration management tasks.

1.3.2 Proprietary Configuration Protocols

As previously said, each vendor has implemented its own solution to collect, change, distribute configurations and system updates. Big vendors such as Cisco Systems, Juniper Networks, Huawei, etc. provide different suites that range from solutions for simple local area networks (LANs), to internet provider scale solutions. The so called Network Management Systems [34] simplify the management of the administered network offering centralized solutions that allow one to perform device discovery, monitoring and management, network performance analysis, intelligent notifications, and customizable alerts. To interact with devices, they build on standard protocols such as SNMP or syslog, but often use also custom solutions based on Command Line Interfaces (CLI) that can be reached via SSH or telnet (deprecated for security reasons). For instance, the Cisco Configuration Professional is a Graphical User Interface (GUI)-based device management tool for Cisco access routers. This tool simplifies routing, firewall, Intrusion Prevention System (IPS), VPN, unified communications, wide area network (WAN) and LAN configurations through GUI-based easy-to-use wizards.

1.3.3 Integrated Platforms for Network Monitoring

As previously said, vendors and third party companies offer a portfolio of management solutions, which range to simple network management for small deployments, to Internet Service Provider scale solutions, from LAN to Data Center Networks.

The main goal of these platforms is to offer a unified view of the network and service status. These platforms are able to collect data from devices belonging to an administration domain via SNMP, Syslog, IPFIX, and proprietary solutions. Often they implement an automatic discovery mechanism to find and add devices to their collection base so to minimize administrator intervention. Via a GUI, they present views of the status of the network, showing time series of link and CPU load, divided by applications or origin-destination of the traffic. The administrator is thus offered a unified view of the network status, with the ability to drill down into more details directly interacting with the GUI. They can also detect network node and connection health problems by using simple threshold-based algorithms. In such cases, alerts can be issued to warn the administrators. Figure 1.2 reports the Zabbix architecture as an example.

From an architecture point of view, all these platforms are similar. They have *proxy modules*, also called *agent modules*, to interact with different protocols and devices to collect data, which is then stored in a *database module*, based on open source solutions like MySQL, Postgre SQL, or commercial solutions like Oracle SQL. A typically *web-based GUI* or dashboard allows the administrator to interact and navigate through the data. The dashboard can offer also configuration

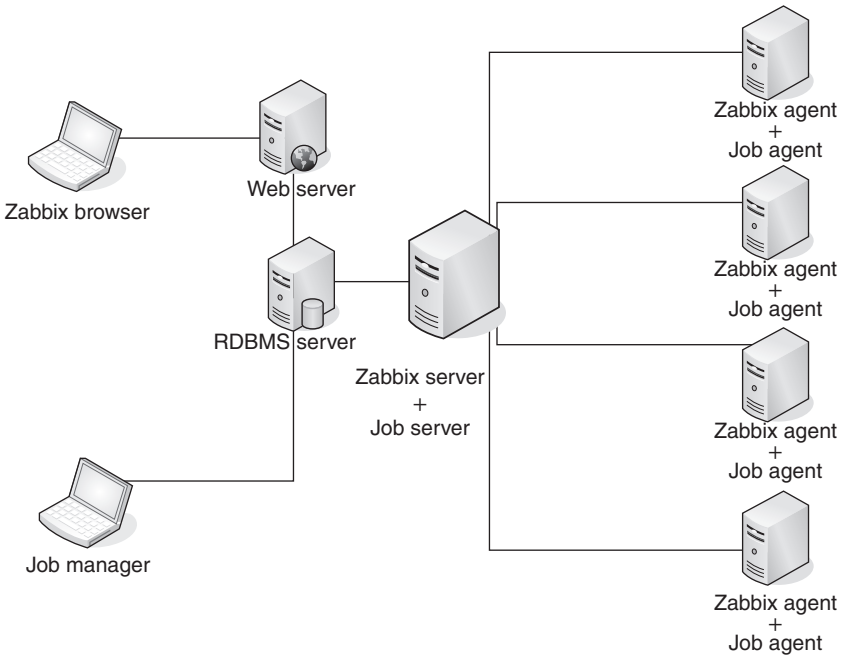

Figure 1.2 Example of monitoring architecture. Source: Courteously from Zabbix.

abilities, typically opening management connection with the devices. At last, a *media gateway* allows the system to raise and distribute alarms, via email, short message service, chat systems, ticketing systems, etc.

Some platforms are open source. They allow to integrate data collected from various deployment into a single centralized center, but rarely offer the ability to change the underlying configuration due to the difficulties in interfacing with different devices. Among those, Zabbix (https://www.zabbix.com), Nagios (https://www.nagios.org), or Cacti (https://www.cacti.net) are the oldest, with more modern solutions like LibreNMS (https://www.librenms.org) or Observium (https://www.observium.org) emerging as novel and more reactive solutions.

Proprietary solutions offer typically more options and flexibility, and include also the ability to change the network setup. Each vendor has a portfolio of solutions that fits different scenarios and deployment sizes, from small LANs to national-wide Internet Service Providers. Solutions are also available from independent vendors that have typically multi-platform support.

1.4 Novel Solutions and Scenarios

In the previous sections of this chapter, we have described the most standard approach to control and manage a network. Here we briefly present the most recent approaches which are still under investigations by the research and technical communities, with development quickly tacking grounds.

1.4.1 Software-Defined Networking – SDN

Software-defined networking (SDN) technology is an approach to network management that separate the control plane from the data plane. In the original internet design indeed, the control plane – where control protocols and management actions are performed – is tightly embedded in the data plane – where packets are routed and forwarded. SDN separates the two planes, so that switches become pure forwarding devices, while all the control and management operations are relegated to a centralized controller. The controller defines forwarding rules, which are then send to switches that use them to forward packets along the proper and desired path. This enables dynamic, programmatically efficient network configuration to improve network performance and monitoring. Martin Casado introduced the idea of relying to a centralized controller to improve network management in 2007 [35]. Since then, SDN technology has become mainstream [36], with support first for campus network, then extending its support for data center networks, and more recently in WANs via the SD-WAN [37], bringing in the WAN area the benefits of decoupling the networking hardware from its control mechanism.

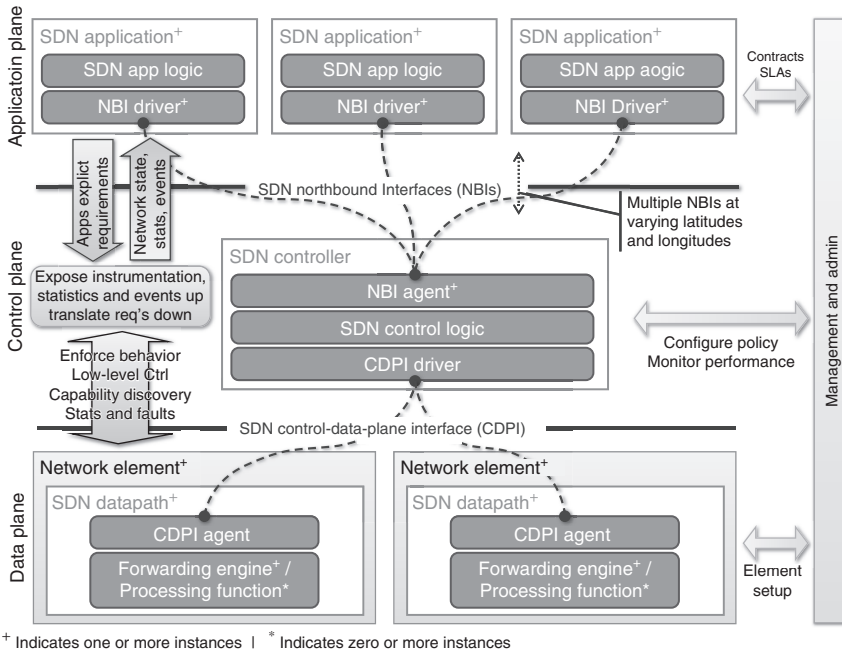

Figure 1.3 The SDN architecture. Source: Courteously from Open Networking Foundation.

The SDN architecture identifies three planes – adding an application plane on the top of the control plane. Figure 1.3 depicts the overall architecture. SDN applications are programs that directly and programmatically communicate their requirements and desired behavior via the northbound interface to the SDN network controller. Applications get an abstracted global view of the network, and suggest decisions and actions such as explicit routes, filtering rules, etc. The SDN controller sits in between. It is a logically centralized entity that translates the requirements from applications to actual action to be implemented by the control plane elements, and provides the applications an updated a common view of the network status. Logically centralized, it can be implemented in a distributed fashion to guarantee both scalability and reliability. It supports both the concept of federated controllers – each responsible of managing a portion of the network; and of hierarchical controllers – where higher hierarchy controllers summarize the information received by lower layers and make it available to applications. At the bottom, the data plane – or the Datapath – is the logical network of devices which offer forwarding and data processing capabilities. Data forwarding engines are in charge of quickly switching packets. They communicate with the SDN controller via the southbound interface, which defines standard Application Programming

Interfaces (API) to exchange information. Traffic processing functions implement decision based on packet payload. For instance, switching decision can be done considering both the sender and receiver addresses – enabling per-flow routing. Similarly, filtering decision can be based on TCP port numbers.

SDN is often associated with the OpenFlow protocol [38] that enables the remote communication with the network plane elements and the controller. However, for many companies, it is no longer an exclusive solution, and proprietary techniques are now available like the Open Network Environment and Nicira's network virtualization platform. They all offer the standard API to communicate via the southbound interface.

1.4.2 Network Functions Virtualization – NFV

Network Functions Virtualization (NFV) is a network architecture that strongly builds on the top of virtualization concepts [39]. It offers the ability to virtualize network nodes and functions into building blocks which can be connected and chained to create more complex communication services. A virtualized network function (VNF) consists of one or more virtual machines and containers that run specific software to implement networking operations in software. Firewalls, access list controllers, load balancers, intrusions detection systems, VPN terminators, etc. can thus be implemented in software – without buying and installing expensive hardware solutions.

NFV consists of three main components as sketched in Figure 1.4: On the top, the VNFs to be implemented, using a software solution; the network functions virtualization infrastructure (NFVI) sits in the middle and offers the hardware

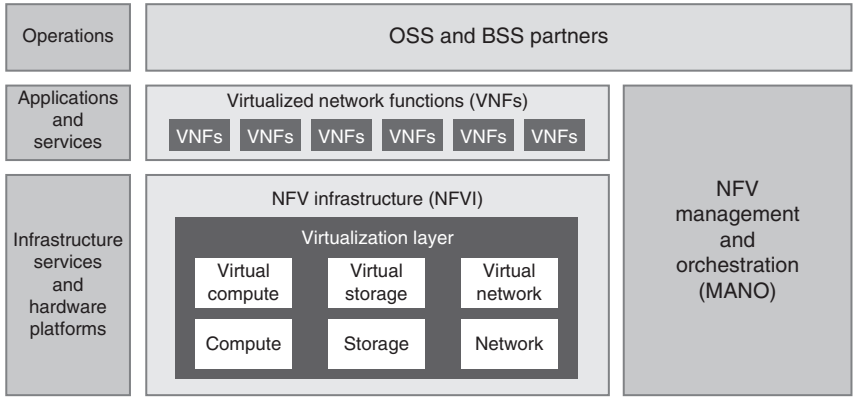

Figure 1.4 Network functions virtualization architecture. Source: Courteously from Juniper Networks.

components over which deploy the VNFs. It includes the physical servers and the network devices that build the NFV infrastructure; at last, the NFV MANagement and Orchestration (MANO) framework allows to manage the platform offering data repositories and standard interfaces to exchange information. To build a complex function, basic blocks can be chained so that a processing pipeline is built. This is called "service chaining" and allows the reuse of highly specialized and efficient blocks to build complex functionalities.

Considering the management operations, clearly NFV requires the network to instantiate, monitor, repair, and bill for the services it offers. NFV targets indeed the large carrier scenario, being it a data center manager, or an internet service providers. These functionalities are allocated to the orchestration layer, which must manages VNFs irrespective of the actual hardware and software technology sitting below.

NFV is a means to reduce cost and accelerate service development and deployment. Instead of requiring the installation of expensive hardware with dedicated functionalities, service providers rely on inexpensive network devices, storage systems, and servers to run virtual machines that implement the desired network function. When a customer asks for a net functionality, the service provider can simply spin up a new virtual machine to implement that function. This has also the benefit to reduce the dependency on dedicated hardware devices, and improve robustness via migration capabilities that move services in case of failures or maintenance operations.

Clearly, NFV calls for standard to allow interoperability of solutions. Since 2012, over 130 of the world's leading network operators have recently joined together to form a European Telecommunications Standards Institute (ETSI) Industry Specification Group (ISG) for NFV (https://www.etsi.org/technologies/nfv). NFV is also fundamental in the 5G arena, where all the advanced functionalities offered by the network like network slicing, edge computing, or decentralized radio management functions are implemented on the top of NFV.

Bibliography

1 Caruso, R.E. (1990). Network management: a tutorial overview. *IEEE Communications Magazine* 28 (3): 20–25.

2 Klerer, S.M. (1988). The OSI management architecture: an overview. *IEEE Network* 2 (2): 20–29.

3 (1984). Specification of Signalling System No. 7.

4 Case, J.D., Fedor, M., Schoffstall, M.L., and Davin, J. (1990). RFC 1157: simple network management protocol (SNMP). *Request for Comments, IETF*.

5 Case, J., McCloghrie, K., Rose, M., and Waldbusser, S. (1996). RFC 1901: introduction to community-based SNMPv2. *Request for Comments, IETF*.

6 Harrington, D., Presuhn, R., and Wijnen, B. (2002). RFC 3411: an architecture for describing simple network management protocol (SNMP) management frameworks. *Request for Comments, IETF*.

7 Gerhards, R. (2009). RFC 5424: the syslog protocol. *Request for Comments, IETF*.

8 Claise, B., Bryant, S., Sadasivan, G. et al. (2008). RFC 5101: specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information. *Request for Comments, IETF*.

9 Paxson, V., Almes, G., Mahdavi, J., and Mathis, M. (1998). RFC 2330: framework for IP performance metrics. *Request for Comments, IETF*.

10 Almes, G., Kalidindi, S., and Zekauskas, M. (1999). RFC 2679: a one-way delay metric for IPPM. *Request for Comments, IETF*.

11 Hedayat, K., Krzanowski, R., Morton, Al. et al. (2008). RFC 5357: a two-way active measurement protocol (TWAMP). *Request for Comments, IETF*.

12 Bajpai, V. and Schönwälder, J. (2015). A survey on internet performance measurement platforms and related standardization efforts. *IEEE Communication Surveys and Tutorials* 17 (3): 1313–1341.

13 Hanemann, A., Boote, J.W., Boyd, E.L. et al. (2005). PerfSONAR: a service oriented architecture for multi-domain network monitoring. In: *International Conference on Service-Oriented Computing* (A. Hanemann, J.W. Boote, E. L. Boyd et al.), 241–254. Springer.

14 RIPE NCC Staff (2015). Ripe atlas: a global internet measurement network. *Internet Protocol Journal* 18 (3). http://ipj.dreamhosters.com/wp-content/uploads/2015/10/ipj18.3.pdf

15 Malkin, G. (1998). RFC 2453: RIP version 2. *Request for Comments, IETF*.

16 Savage, D., Ng, J., Moore, S. et al. (2016). RFC 7868: Cisco's enhanced interior gateway routing protocol (EIGRP). *Request for Comments, IETF*.

17 Moy, J. (1998). RFC 2328: OSPF version 2. *Request for Comments, IETF*.

18 Vasseur, J.P., Shen, N., and Aggarwal, R. (2007). RFC 4971: intermediate system to intermediate system (IS-IS) extensions for advertising router information. *Request for Comments, IETF*.

19 Shalunov, S., Teitelbaum, B., Karp, A. et al. (2006). RFC 4656: a one-way active measurement protocol (OWAMP). *Request for Comments, IETF*.

20 Meyer, D. (1997). University of Oregon Route Views Project. http://www.routeviews.org/routeviews/.

21 Orsini, C., King, A., Giordano, D. et al. (2016). BGPStream: a software framework for live and historical BGP data analysis. *Proceedings of the 2016 Internet Measurement Conference*, pp. 429–444.

22 Giotsas, V., Dietzel, C., Smaragdakis, G. et al. (2017). Detecting peering infrastructure outages in the wild. *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pp. 446–459.

23 Luckie, M. and Beverly, R. (2017). The impact of router outages on the AS-level internet. *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pp. 488–501.

24 Sermpezis, P., Kotronis, V., Dainotti, A., and Dimitropoulos, X. (2018). A survey among network operators on BGP prefix hijacking. *ACM SIGCOMM Computer Communication Review* 48 (1): 64–69.

25 Padmanabhan, R., Dhamdhere, A., Aben, E. et al. (2016). Reasons dynamic addresses change. *Proceedings of the 2016 Internet Measurement Conference*, pp. 183–198.

26 Livadariu, I., Elmokashfi, A., and Dhamdhere, A. (2017). On IPv4 transfer markets: analyzing reported transfers and inferring transfers in the wild. *Computer Communications* 111: 105–119.

27 Rosen, E., Viswanathan, A., and Callon, R. (2001). RFC 3031: multiprotocol label switching architecture. *Request for Comments, IETF*.

28 Xiao, X., Hannan, A., Bailey, B., and Ni, L.M. (2000). Traffic engineering with MPLS in the internet. *IEEE Network* 14 (2): 28–33.

29 Pepelnjak, I. and Guichard, J. (2002). *MPLS and VPN Architectures*, vol. 1. Cisco Press.

30 Huang, C., Sharma, V., Owens, K., and Makam, S. (2002). Building reliable MPLS networks using a path protection mechanism. *IEEE Communications Magazine* 40 (3): 156–162.

31 Enns, R., Bjorklund, M., Schoenwaelder, J., and Bierman, A. (2011). RFC 6241: network configuration protocol (NETCONF). *Request for Comments, IETF*.

32 Bjorklund, M. (2010). RFC 6020: Yang - a data modeling language for the network configuration protocol (NETCONF). *Request for Comments, IETF*.

33 Yavatkar, R., Pendarakis, D., Guerin, R. et al. (2000). RFC 2753: a framework for policy-based admission control. *Request for Comments, IETF*.

34 Martin-Flatin, J.-P., Znaty, S., and Hubaux, J.-P. (1999). A survey of distributed enterprise network and systems management paradigms. *Journal of Network and Systems Management* 7 (1): 9–26.

35 Casado, M., Freedman, M.J., Pettit, J. et al. (2007). Ethane: taking control of the enterprise. *ACM SIGCOMM Computer Communication Review* 37 (4): 1–12.

36 Kirkpatrick, K. (2013). Software-defined networking. *Communications of the ACM* 56 (9): 16–19. https://doi.org/10.1145/2500468.2500473.

37 Yang, Z., Cui, Y., Li, B. et al. (2019). Software-defined wide area network (SD-WAN): architecture, advances and opportunities. *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, IEEE, pp. 1–9.

38 McKeown, N., Anderson, T., Balakrishnan, H. et al. (2008). OpenFlow: enabling innovation in campus networks. *SIGCOMM Computer Communication Review* 38 (2): 69–74. https://doi.org/10.1145/1355734.1355746.

39 Han, B., Gopalakrishnan, V., Ji, L., and Lee, S. (2015). Network function virtualization: challenges and opportunities for innovations. *IEEE Communications Magazine* 53 (2): 90–97.