CHAPTER

NETWORK FUNCTIONS VIRTUALIZATION

10

Many casual followers of SDN sometimes confound *Network Functions Virtualization* (NFV) with SDN. Indeed these two technologies are closely related though they are not the same. While this work is about SDN, the symbiosis between it and NFV drive the need for a more detailed look at this sister technology. NFV was introduced in a presentation titled "Network Functions Virtualisation; an introduction, benefits, enablers, challenges and call for action" in 2012 at the SDN and OpenFlow World Congress [1]. Shortly after it was introduced, the *European Telecommunications Standards Institute* (ETSI) took the lead on NFV. In this chapter, we will define NFV and compare and contrast it with SDN so that our reader will understand its relationship to the focus of our work.

10.1 DEFINITION OF NFV

The following definition is provided in [1]:

> Network Functions Virtualisation aims to transform the way that network operators architect networks by evolving standard IT virtualisation technology to consolidate many network equipment types onto industry standard high volume servers, switches and storage, which could be located in Data Centers, Network Nodes and in the end user premises... It involves the implementation of network functions in software that can run on a range of industry standard server hardware, and that can be moved to, or instantiated in, various locations in the network as required, without the need for installation of new equipment.

This definition made reference to a figure which we reproduce in Fig. 10.1. This vision and definition can be further decomposed. ETSI has created an approach [1] to migrate a *Physical Network Function* (PNF) to a *Virtual Network Function* (VNF). The core concept is that those physical functions run on commodity hardware with virtualized resources driven by software. As we described in Section 3.7, the IT world has been virtualizing compute, storage, and to a lesser extent networks for many years.

Many vendors offer network virtualization (NV) products, and their technologies vary. ETSI has created an NFV framework that will lead to standard solutions from a plethora of network vendors. This framework defines an *NFV Infrastructure* (NFVI), where VNFs are created and managed by an *NFV Orchestrator* (NVFO) and VNF Manager. The concept of the orchestrator is similar to a hypervisor in cloud computing. The orchestrator will be responsible for instantiating resources such as compute, memory, and storage to support the virtual machine hosting the VNF. It is also responsible for grouping NFVs into *service chains*, a suite of connected network services over a single network connection.

Software Defined Networks. http://dx.doi.org/10.1016/B978-0-12-804555-8.00010-7

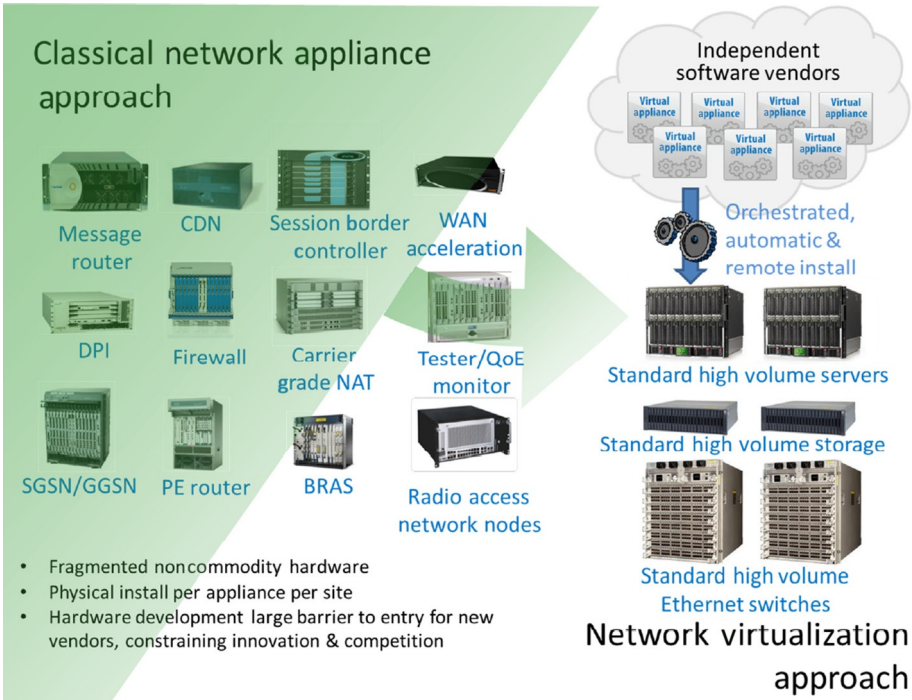

FIG. 10.1

Vision for NFV.

(Courtesy of Network Functions Virtualisation; an introduction, benefits, enablers, challenges & call for action. In: SDN and OpenFlow World Congress. Germany: Darmstadt; 2012. Retrieved from: http://portal.etsi.org/NFV/NFV_White_Paper.pdf.)

The concept of the VNF Manager is to monitor and manage the health of the VNF running on the virtual machine. The ETSI NFV group expects that the VNF will implement a network entity with well-defined, standards-based behavior and interfaces. In brief, NFV formalizes the concept of taking a network function that runs on a physical device and placing the function on a virtual machine running on a virtual server infrastructure [2].

The ETSI model for NFV has defined a management and orchestration architecture and nicknamed it MANO. In reality, MANO is the most challenging part of virtualizing PNFs. Considering the size of networks in carriers like BT, AT&T, Verizon, and others, management and orchestration is a daunting task. Bringing up thousands of virtual machines with network functions running on them could easily result in chaos. As such, the need for management in the NFV solution was a focus of the early efforts. The MANO [3] framework has the following functional blocks:

- *NFVO*: This is used for on-boarding of new *Network Service* (NS), *VNF Forwarding Graph* (VNF-FG), and VNF Packages, NS lifecycle management (including instantiation, scale-out/in, performance measurements, event correlation, termination) global resource management, validation and authorization of NFVI resource requests, and policy management for NS instances.

- *VNF Manager*: This provides lifecycle management of VNF instances, and overall coordination and adaptation for configuration and event reporting between NFVI, the *Element Management System* (EMS), and the *Network Management System* (NMS).
- *Virtualized Infrastructure Manager*: This controls and manages the NFVI compute, storage, and network resources within one operator's infrastructure subdomain. It is responsible for the collection and forwarding of performance measurements and events.

Network virtualization (NV) has existed for some time. It is not unusual to have trouble distinguishing NV from NFV, so we will attempt to clarify this here. NV creates an overlay of the physical network. Instead of connecting two different domains with physical wiring in a network, NV creates tunnels through the existing network. This saves time and effort for network administrators and technicians. NV is well-suited to providing connectivity between virtual machines. On the other hand, NFV virtualizes layer four through seven functions. Examples include firewalls, load balancers, *Intrusion Detection Systems* (IDS), *Intrusion Protection Systems* (IPS), and others. IT cloud administrators spin up VMs by pointing and clicking. NFV and NV allow network administrators to have the same capabilities for the network. For example, Embrane, recently bought by Cisco, is an example of a company that has an IDS/IPS service that can be instantiated as an NFV.

10.2 WHAT CAN WE VIRTUALIZE?

Newcomers to NFV may reasonably ask "what can we virtualize?" While an answer of "anything" might appear glib, it is not far from the truth. Almost any PNF can be virtualized. As an example, the list [4] of ETSI NFV use cases includes a wide range of functions:

- AR: Enterprise Access Router/Enterprise CPE
- PE: Provider Edge Router
- FW: Enterprise Firewall
- NG-FW: Enterprise NG-FW
- WOC: Enterprise WAN Optimization Controller
- DPI: Deep Packet Inspection (Appliance or a function)
- IPS: Intrusion Prevention System and other Security appliances
- Network Performance Monitoring

Another example in [4] is the virtualization of the *IP multimedia subsystem* (IMS). The IMS is a session control architecture to support provisioning of multimedia services over the *Evolved Packet Core* (EPC) and other IP-based networks. Some of the network functions being virtualized include the *Proxy Call Session Control Function* (P-CSCF), *Serving Call Session Control Function* (S-CSCF), *Home Subscriber Server* (HSS), and *Policy and Charging Rules Function* (PCRF). These prototype efforts quickly accelerated the delivery of real products to the market. As early as 2014, Alcatel-Lucent [5] announced a portfolio of VNFs including the EPC, IMS, and the *Radio Access Network* (RAN). A list of some possible NFV *elements* [6] is shown in Fig. 10.2.[1] NFV elements are the

[1]For the curious reader, there is no actual association between the physical elements shown in Fig. 10.2 and the elements of NFV. The physical elements are for visual effect only.

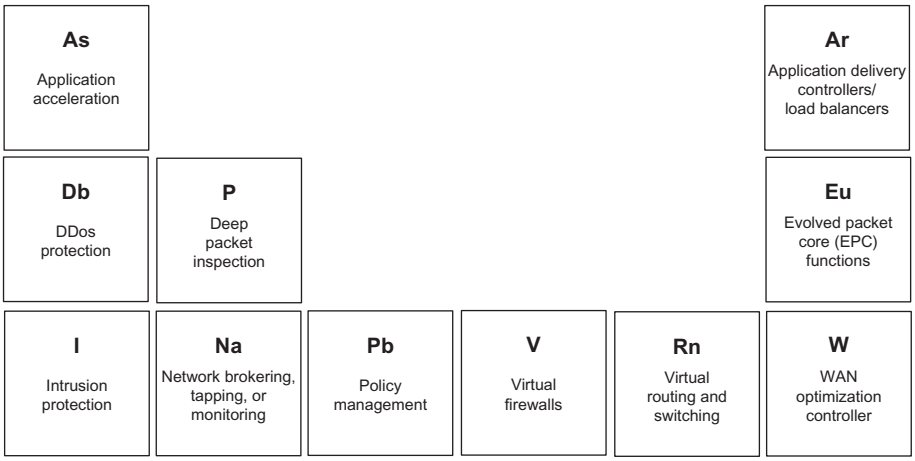

FIG. 10.2

Elements of NFV.

discrete hardware and software requirements that are managed in an NFV installation to provide new communication and application services that have traditionally been performed by application-specific appliances. Table 10.1 lists companies actively selling NFV products.

10.3 STANDARDS

ETSI is the *de facto* standards body for NFV. After being chartered as the standard bearer for NFV in 2012, ETSI published the first five specifications for NFV in 2013 [7]. As of this writing, ETSI has 20 specifications [8] freely available for download.

DISCUSSION QUESTION

When you visit the ETSI website, you will see that they are involved in many different standards and activities outside of NFV. Do you think it would be beneficial to have NFV separated into its own organization singularly focused on NFV?

10.4 OPNFV

Open platform for NFV (OPNFV) is an open source project focused on accelerating NFV's evolution through an integrated, open platform. Using the ETSI architectural framework as a launch pad, OPNFV was created to promote an open source platform to "accelerate the introduction of NFV products and services" [9]. The ETSI NFV efforts sought to standardize solutions from NFV vendors who previously were creating proprietary implementations. The next evolutionary step is to bring open source products

Table 10.1 Leading NFV Vendors

Company	Carrier NFV	Wireless Carriers	Enterprise and Data Centers	Products
Arista	Yes	Yes	Yes	Distribution, core, and access switches
Blue Coat			Yes	Forward and reverse proxy—ProxySG
Brocade	Yes			Distribution, core, access switches, firewall, load balancer, WAN edge, application accelerator
Checkpoint			Yes	Edge/perimeter firewalls
Cisco	Yes	Yes	Yes	UCS fabric interconnect, edge/perimeter firewalls, routers, load balancer, application acceleration, wireless LAN controller, WAN optimization (vWAAS)
Citrix			Yes	Load balancer
Ericsson	Yes	Yes		Carrier aggregation
F5	Yes			Application accelerators
Fortinet			Yes	Edge/perimeter firewall
HP	Yes		Yes	
Hitachi	Yes		Yes	Access switches
Huawei	Yes	Yes		Universal service routers
Lenovo/IBM			Yes	
Juniper	Yes		Yes	Edge/perimeter firewall, compute firewall VGW
Palo Alto Networks	Yes			Device role, edge/perimeter firewall
Radware			Yes	Load balancer/ADC
Riverbed			Yes	Load balancer/ADC, WAN optimization
VMware			Yes	Edge gateway, virtual access switches, vCenter
Nokia[a]	Yes	Yes		
Dell				Access switches
Force10				Access switches

[a] *Alcatel-Lucent is now part of Nokia.*

to the table. Thus OPNFV is very complementary to the ETSI NFV efforts. OPNFV includes an initial build of the NFVI and *Virtual Infrastructure Manager* (VIM) components. The diagram of their second release Brahmaputra is shown in Fig. 10.3.

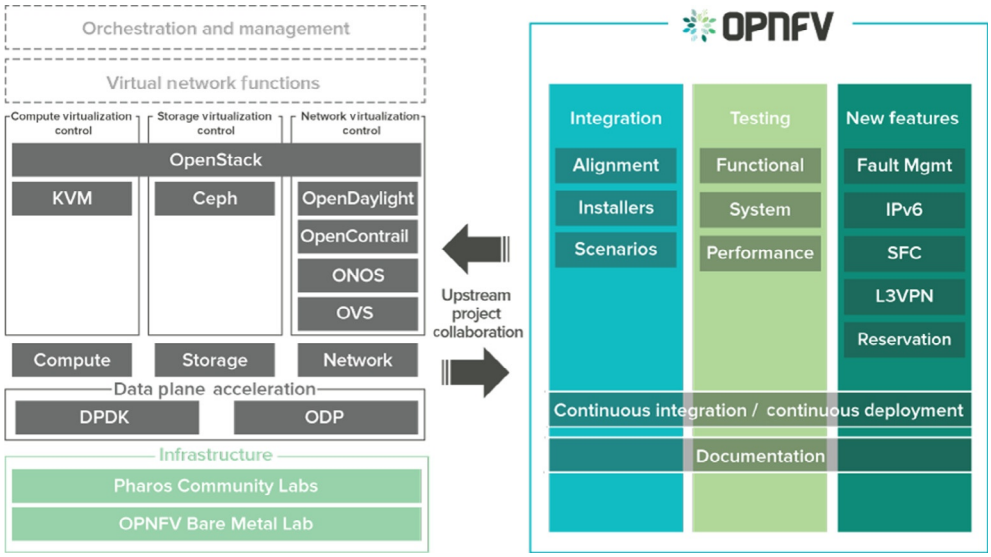

FIG. 10.3

OPNFV second release Brahmaputra.

(Courtesy of OPNFV. Linux foundation: collaborative projects. Retrieved from: https://www.opnfv.org.)

10.5 LEADING NFV VENDORS

Each of the companies we list in Table 10.1 have multiple offerings in the NFV arena. As this is a rapidly evolving field, the data contained in this table really only reflects the marketplace at the time of writing of this work. Looking across these vendors, they each bring some expertise to bear. The table indicates whether or not the vendor is a major participant in the NFV markets of *Carrier NFV*, *Wireless Carriers*, and *Enterprise/Data Centers*. If the company is not listed under one of these categories, this does not imply that they have no relevant offerings, but merely that they are not a major player. A real-time list of various vendors involved in SDN, NFV, and NV is available in [10].

10.6 SDN VS NFV

Open SDN makes the network programmable by separating the control plane (i.e., telling the network what goes where) from the data plane (i.e., the sending of packets to specific destinations). It relies on switches that can be programmed through an SDN controller using an industry standard control protocol, such as OpenFlow. As described previously, NV and NFV add virtual tunnels and functions, respectively, to the physical network. SDN, on the other hand, changes the forwarding behavior of the physical network. The business need for SDN arises from the need to provide faster and more flexible service fulfillment [11]. SDN's scope is to be able to control and manage anything that could contribute to a service. This includes bringing up VNFs under the management of the VNFI. NFV, conversely, is about reducing the cost and time to provide network functions that contribute to the delivery of

a service. As previously described, NFV deploys network functions as software on general purpose machines.

DISCUSSION QUESTION

We have discussed NFV and SDN. Since they are complementary, should the organizations be combined? What are the pros and cons of a combined organization?

The SDN controller views a VNF as just another resource. ETSI's NFV architecture framework includes components that *turn up*[2] VNFs and manage their lifecycle. Since SDN and NFV are complementary, network engineers need not choose between them, but rather understand how they might work together to address their needs. For this synergy to be effective, SDN and NFV need to know what the other is doing. The ETSI framework stipulates that there is a VNF manager that can turn up or turn down a VNF. If that VNF is part of an active service being provided to a customer, then the framework should not turn down the VNF. This mandates a common repository that SDN and NFV may access to understand the state of a service. Ideally, some type of repository holding state for an individual or set of VNFs should exist. SDN should be able to communicate to the NFV framework to bring up VNFs to support a provisioned service. The state of this service should be maintained so that the NFV framework and its MANO capabilities understand what is in use and when it can turn down a VNF. Fig. 10.4 depicts a scenario where SDN and NFV would collaborate. The controller would contact the NFV MANO to spin up or tear down VNFs and remain aware of the state of the VNF through a common data store.

10.6.1 WHEN SHOULD NFV BE USED WITH SDN?

NFV should be used with SDN when there are many physical network elements that you wish to virtualize. If one only had a few VNF's, it would not make sense to implement the overhead of the ETSI VNF reference architecture. However, the complexity of management, monitoring, and orchestrating a large number of VNFs and their service chains creates a need for the NFV framework. The ETSI VNF reference architecture is well suited to managing the complexity. As previously mentioned, SDN would view the VNFs as just another resource and readily manage the complex environment supporting vast numbers of VNFs. SDN is not constrained to managing just VNFs, as the scope of SDN is much broader than NFV. The promise of SDN applications relying on external information to help manage the network will enhance new service offerings.

10.7 IN-LINE NETWORK FUNCTIONS

In-Line Network Functions typically include functionality such as load balancers, firewalls, IDS, and IPS. These services must be able to inspect the packets passing through them. These functions are

[2]We will use the terms *turn up* and *turn down* interchangeably with *spin up* and *spin down*, respectively, in our discussion on NFV.

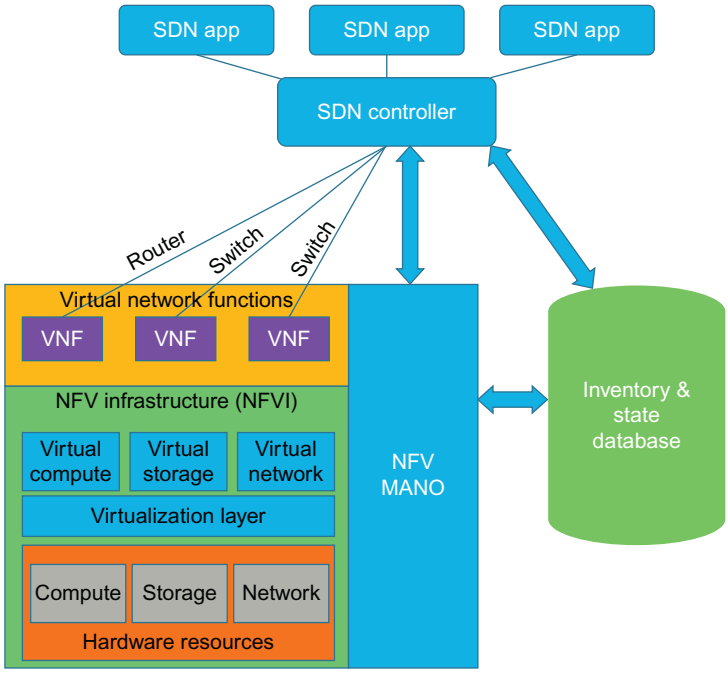

FIG. 10.4

SDN, NFV collaborating with VNFs.

often delivered in specialized appliances which can be very expensive. These appliances are *bumps-in-the-wire*, meaning that they are inserted into the data plane. In some cases, the *bump* shunts traffic to an out-of-band processor, which can result in cost savings. In other cases, the value-added service performed by the appliance is performed at line rate and the increased cost of such devices is necessary if performance is of paramount concern. Data centers and SPs which must employ these types of services welcome any novel means to drive down the costs associated with these devices.

There are many schemes for load balancing traffic, but all involve some level of packet inspection, choosing the destination in order to evenly distribute the load across the entire set of servers.

Firewalls serve the role of admitting or denying incoming packets. The decision criteria involve packet inspection and can be based on factors such as destination or source IP address, destination or source TCP or UDP port, or something deeper into the payload, such as the destination URL.

IDS and IPS solutions analyze packets for malicious content. This entails performing process-intensive analysis of the packet contents to determine if there is a threat present. While an IPS is a bump-in-the-wire, an IDS can run in parallel with the data plane by looking at mirrored traffic. Such systems require some traffic mirroring mechanism. A hybrid system can be implemented in the SDN paradigm by shunting only those packets requiring deeper inspection to an appliance that is off of the data path. This approach has the advantage of removing the bump-in-the-wire, as packets that are not

subject to deeper inspection suffer no additional latency at all. Also, since not all packets need to be processed by the appliance, fewer or less powerful appliances are needed in the network. This can be realized in SDN by simply programming flow rules, which is much simpler than the special *Switch Port ANalyzer* (SPAN) ports that are in use on contemporary traditional switches.

In the following three sections, we study examples of SDN being used to virtualize three in-line network functions: (1) Server Load Balancing, (2) Firewalls, and (3) IDS. These simple examples illustrate cases where there is strong overlap between NFV and SDN.

10.7.1 SDN APPLIED TO SERVER LOAD-BALANCING

Load balancers must take incoming packets and forward them to the appropriate servers. Using SDN and OpenFlow technology, it is straightforward to imagine a switch or other networking device having rules in place which would cause it to act as a load balancer appliance at a fraction of the cost of purchasing additional, specialized hardware.

OpenFlow 1.0 supports matching against the basic 12-tuple of input fields as described in Section 5.3.3. A load balancer has a rich set of input options for determining how to forward traffic. For example, as shown in Fig. 10.5, the forwarding decision could be based on the source IP address. This would afford server continuity for the duration of time that that particular endpoint was attempting to communicate with the service behind the load balancer. This would be necessary if the user's transaction with the service required multiple packet exchanges and if the server needed to maintain state across those transactions.

The fact that the controller will have the benefit of understanding the load on each of the links to the servers, as well as information about the load on the various servers behind the firewall, opens up other possibilities. The controller could make load balancing decisions based on a broader set of criteria than would be possible for a single network appliance.

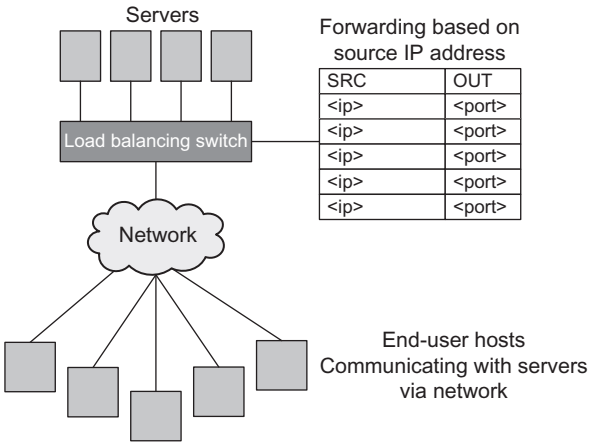

FIG. 10.5

Load balancers using OpenFlow.

Many people confuse network load balancing with server load balancing. Network load balancing is simple and quite fast because it relies on simple parameters like IP address and the TCP port. Application server load balancing is different and more complicated. This is where SDN's centralized knowledge along with SDN applications can play a major role. True application server load balancing must take into consideration many more variables outside of the network. These include the load on the server and processing time, among others. SDN's centralized, global network view can be augmented by an SDN application that can gather many more variables outside of the IP address and TCP port. This includes the server load, its processing time for transactions, etc. The SDN application can then assist the SDN controller by helping it take into consideration these other factors when routing transactions to be processed to particular servers.

10.7.2 SDN APPLIED TO FIREWALLS

Firewalls take incoming packets and forward them, drop them or take some other action such as forwarding the packet to another destination or to an IDS for further analysis. Like load balancers, firewalls using SDN and OpenFlow have the ability to use any of the standard 12 match fields for making these forward or drop decisions. Packets matching certain protocol types (e.g., HTTPS, HTTP, Microsoft Exchange) are forwarded to the destination and those that do not match are dropped. Simple firewalls that are ideally suited to an SDN-based solution include firewalls that are based on blocking or allowing specific IP addresses or specific TCP/UDP ports. As with load balancers, SDN-based firewalls may be limited by a lack of statefulness and inability for deeper packet inspection.

One SDN use case applied to firewalls is to insert an OpenFlow-enabled switch in front of a battery of firewalls. One benefit of this is that the SDN switch can load balance between the firewall devices, permitting greater scale of firewall processing. More importantly, the SDN switch can shunt *known-safe* traffic around the firewalls. This is possible since OpenFlow rules can be programmed such that we offload traffic from the firewalls that is known to be problem-free. The benefit of this is that less firewall equipment and/or power is needed and less traffic is subjected to the latency imposed by firewall processing.

Another example of an SDN firewall-type implementation that slices the network into different departments is FlowVisor [12]. It slices the network into different parts to be managed by different controllers for each network slice. This allows the FlowVisor OpenFlow controller to enforce isolation between different networks on the same campus.

10.7.3 SDN APPLIED TO INTRUSION DETECTION

IDS systems have typically been passive security devices that would listen to all traffic (egress and ingress) to identify suspicious traffic and potential attacks. An IDS looks for application protocol violations and uses rules that analyze statistics and traffic patterns that may indicate an attack. Just as in the case of a firewall, an SDN application can be created to support a VNF that would act as a traditional *network tap* and passively route packets to the application for analysis. There is some concern whether the performance of VMs in this role can compete with customized hardware optimized for this task. It is noteworthy that Cisco has commercialized virtual IPSs and IDSs [13].

An IDS is intended to observe network traffic in order to detect network intrusions. An IPS additionally attempts to prevent such intrusions. The range of intrusions that these systems are intended

to recognize is very broad and we do not attempt to enumerate them here. We will, however, consider two particular IDS functions that lend themselves well to SDN-based solutions.

The first type of intrusion detection we consider requires deep packet inspection of payloads, searching for signs of viruses, trojans, worms, and other malware. Since this inspection is processing-intensive, it must take place somewhere outside the data plane of the network. Typical solutions utilize a network tap, which is configured to capture traffic from a particular source, such as a VLAN or physical port, and copy that traffic and forward it to an IDS for analysis. Physical taps such as this are mostly port-based.

One pre-SDN solution is based on a network patch-panel device which can be configured to take traffic from an arbitrary device port and forward it to the analysis system. This requires patch panel technology capable of copying traffic from any relevant network port. This technology is available from companies such as *Gigamon* and *cPacket Networks*. In large networks the cost of such approaches may be prohibitive. Forwarding traffic on uplinks is also sensitive to the volume of traffic, which may exceed the capacity of the uplink data path.

Another pre-SDN solution for IDS is to use configuration tools on the network devices to configure SPAN ports. This allows the switch to operate as a virtual tap, where it copies and forwards traffic to the IDS for analysis. Taps such as this can be based on port or VLAN, or even some other packet-based filter such as device address.

OpenFlow is well-suited to this application since it is founded on the idea of creating flow entries that match certain criteria (e.g., device address, VLAN, ingress port) and performing some action. A tap system using OpenFlow can be programmed to set up actions on the device to forward packets of interest either out through a monitor port or to the IP address of an IDS.

At the end of Section 10.7.2 we described an SDN switch being used to front-end a battery of firewalls. This strategy may also be applied to a battery of IPS appliances. The same rationale of shunting safe traffic around the IPS appliances while allowing the system to scale by using multiple IPS units in parallel applies here as well.

The FlowScale open source project done at InCNTRE is an example of this kind of front-end load balancing with the SPAN concept. FlowScale sends *mirrored* traffic to a battery of IDS hosts, load-balancing the traffic across them.

10.8 CONCLUSION

For the reader interested in additional use cases of NFV, a concrete example of an NFV implementation of DPI is provided in [14]. A deeper technical treatment of an NFV solution called *ClickOS* can be found in [15]. ClickOS is a high-performance, virtualized network appliance platform that can be used to implement a number of the NFV use cases [4] we cited earlier.

NFV complemented by SDN will have a large impact on data centers, networks, and carriers. As the network is virtualized, cost savings will be dramatic due to reduced labor costs and policy-enforced configuration through automated solutions. Under ETSI's leadership, more and more solutions from vendors will be interoperable. Further, the industry solutions will be less costly as open source is increasingly embraced by customers and NFV vendors. The promise of OPNFV is to promote the use of open source products within the ETSI NFV architectural framework. Given the huge cost savings for carriers and other customers, we expect the adoption of NFV to accelerate over the next few years.

Addressing the different use cases which we have described in the last three chapters requires a wide variety of controller applications. In Chapter 12 we will examine some sample SDN applications in detail and see how domain-specific problems are addressed within the general network programming paradigm of SDN. First, though, we should acknowledge that this important movement could only take place because of key players that have pushed it forward. For that reason, in Chapter 11 we will take a step back from technology and review the individuals, institutions and enterprises that have had the greatest influence on SDN since its inception.

REFERENCES

[1] Network Functions Virtualisation; an introduction, benefits, enablers, challenges & call for action. In: SDN and OpenFlow World Congress. Germany: Darmstadt; 2012. Retrieved from: http://portal.etsi.org/NFV/NFV_White_Paper.pdf.

[2] McCouch B. SDN, network virtualization, and NFV in a nutshell. Network Computer; 2014. Retrieved from: http://www.networkcomputing.com/networking/sdn-network-virtualization-and-nfv-in-a-nutshell/a/d-id/1315755.

[3] Network Functions Virtualization: Mano—management and orchestration. Retrieved from: http://www.network-functions-virtualization.com/mano.html.

[4] Network Functions Virtualisation (NFV); use cases. Retrieved from: http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf.

[5] Alcatel-Lucent delivers suite of virtualized network functions, ushering in the next phase of mobile ultra-broadband for service providers. Paris: Alcatel-Lucent Press Release; 2014. Retrieved from: https://www.alcatel-lucent.com/press/2014/alcatel-lucent-delivers-suite-virtualized-network-functions-ushering-next-phase-mobile.

[6] SDX Central. An overview of NFV elements. Retrieved from: https://www.sdxcentral.com/resources/nfv/nfv-elements-overview/.

[7] ETSI. ETSI publishes first specifications for network functions virtualisation. France: ETSI; 2013. Retrieved from: http://www.etsi.org/news-events/news/700-2013-10-etsi-publishes-first-nfv-specifications?highlight=YToxOntpOjA7czozOiJuZnYiO30=.

[8] ETSI. Our standards. Retrieved from: http://www.etsi.org/standards.

[9] OPNFV. Linux foundation: collaborative projects. Retrieved from: https://www.opnfv.org.

[10] SDX Central. SDN & NFV Services Directory. Retrieved from: https://www.sdxcentral.com/nfv-sdn-services-directory/.

[11] Open Networking Foundation. TR-518 relationship of SDN and NFV, issue 1; 2015. Retrieved from: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/onf2015.310_Architectural_comparison.08-2.pdf.

[12] Al-Shabibi A. Flowvisor. Open Networking Lab; 2013. Retrieved from: https://github.com/OPENNETWORKINGLAB/flowvisor/wiki.

[13] Cisco. Cisco NGIPSv for VMware data sheet. Cisco FirePOWER 7000 Series Appliances. Retrieved from: http://www.cisco.com/c/en/us/products/collateral/security/firepower-7000-series-appliances/datasheet-c78-733165.html.

[14] Bremler-Barr A, Harchol Y, Hay D, Koral Y. Deep packet inspection as a service. In: Proceedings of the 10th ACM international conference on emerging networking experiments and technologies (CoNEXT '14). New York, NY: ACM; p. 271–82.

[15] Martins J, Ahmed M, Raiciu C, Olteanu V, Honda M, Bilfulco R, et al. ClickOS and the art of network function virtualization. In: Proceedings of the 11th USENIX conference on networked systems design and implementation (NSDI'14). Berkeley, CA: USENIX Association; p. 459–73. Retrieved from: https://www.usenix.org/system/files/conference/nsdi14/nsdi14-paper-martins.pdf.