Virtualized Cyber Security: Elastically Scalable DDoS Scrubbing and Threat Deception

A Sandvine Technology Showcase

Executive Summary

In early 2017, a tier-1 converged access communications service provider (CSP) in Asia-Pacific issued Sandvine a challenge: demonstrate a distributed denial-of-service (DDoS) attack solution that achieves elastic scale by means of a Virtual Network Function Manager (VNFM).

Such a solution overcomes the enormous inefficiencies of legacy DDoS solutions, which typically consist of proprietary hardware that is dimensioned for rare peaks, and otherwise sits idle, consuming operational resources for no gains.

This paper explains how Sandvine met the challenge, by working with partners Dell EMC, Intel, and Rift.io.

- The solution leveraged the compute capabilities of two Sandvine VNFs: the Traffic Steering Engine is dynamically provisioned to meet the bandwidth and packet forwarding rate demands of a sudden DDoS attack; the Policy Traffic Switch performs the DDoS scrubbing.
- Dell EMC provided the underlying infrastructure via the DSS 9000 rack scale infrastructure. Based on hyperscale principles, the DSS 9000 provides compute, storage, networking, power and cooling, and open management in a pre-integrated rack. Management is provided at the rack level and is based on the Distributed Management Task Force (DMTF) Redfish specification.
- The Intel RSD Pod Manager and the Pod Management Foundation API provided the tools and APIs to integrate platform discovery, lifecycle, boot, configuration, and telemetry capability required to accomplish the orchestration capability of the NFV infrastructure.
- Rift.io provided the autoscaling framework that allows the Sandvine virtual instances to scale as required by the attack load.

In addition to the DDoS component of this proof-of-concept, Sandvine took the opportunity to demonstrate carrier-scale threat deception.
Introduction

In January 2017, the European Advanced Networking Test Center (EANTC) validated several important functions of both the Sandvine Traffic Steering Engine (TSE) and Sandvine’s virtualized Policy Traffic Switch (PTS).1,2

Upon seeing these results, a tier-1 converged access communications service provider (CSP) in Asia-Pacific approached Sandvine and issued a challenge: this CSP experienced the ‘pain’ and very real costs of legacy approaches to defending against distributed denial of service (DDoS) attacks, and challenged Sandvine to demonstrate a solution that achieved elastic scale by means of a Virtual Network Function Manager (VNFM).3

The Evolution of Network Functions Virtualization

Network Functions Virtualization (NFV) is likely the highest-impact technology shift in telecommunications this decade, for both CSPs and traditional vendors; certainly, NFV—and the closely related adoption of Software-Defined Networking (SDN)—represents the largest strategy shift in telecom since the migration from analog services to digital services.

From the early ‘stewardship’ by the European Telecommunications Standards Institute (ETSI), commitment and buy-in from large CSPs—including AT&T4—that signaled the future will be software, to the large-scale availability and general proliferation of software-based Virtual Network Functions (VNFs), network operators have transformed the landscape from the central office to the datacenter.

“We’ve been on this software-defined journey for quite some time now, and there’s really three core components that go with this program: virtualization, which is basically virtualizing these network functions; control, which is what level of SDN control do we have for that function; and automation, which is how much of that particular function have we automated in terms of the care, feeding, support, and monitoring.”

— Andre Fuetsch, CTO, AT&T

This enormous shift has driven manufacturers/vendors to begin to move away from proprietary, sometimes cumbersome architectures, towards embracing large-scale community-driven open source initiatives such as OpenStack.

But this shift brings with it uncertainty for all parties: monolithic, proprietary architecture-based products tend to have much tighter vertical quality assurance processes around components, design, and certifications (e.g., Apple’s successful consumer products).

By opening up the network function to commercial off-the-shelf (COTS) components, the performance, service level agreements (SLAs), and reliability guarantees of closed systems, as well as the related support and service contracts, are now distributed across many third-party components. The result is that hardware infrastructure, host environments, management and orchestration (MANO), element management systems (EMS), and hardware dependencies now must co-exist across a complex and sometimes dynamic environment provided by a long list of vendors and organizations.

To secure their roles in these new architectures and to seize the opportunities that such democratization presents, vendors like Sandvine have embraced this new paradigm of running network functions on decoupled third-party infrastructure.

However, these new architectures are not without their challenges: the multitude of combinations of hardware

---

2. This test was only the latest in a long line of network functions virtualization (NFV) milestones achieved by Sandvine—for a more complete timeline of milestones, please see the Appendix.
3. While the EANTC validation demonstrated elastic scale, the scaling did not make use of a true VNF Manager (VNFM) to do so.
4. The quote from Andre Fuetsch is from this interview with SDX Central: https://www.sdxcentral.com/articles/news/qa-att-cto-andre-fuetsch-links-software-migration-to-5g/2017/05/
components, supplied by a long list of vendors, deployed on varied software stacks (both open and closed) or on proprietary forks of these software stacks mean that balance has to be struck—outright packet processing performance and maximum compatibility are two goals that are often at odds. Performance can be maximized on a targeted subset of hardware components, which limits interoperability and openness; compatibility can be maximized at the expense of optimized performance.

In practice, most implementations will fall somewhere in-between these two extremes, but the technology and industry media devotes all its attention to either end of the spectrum.

The Challenge: Demonstrate an Elastically Scalable DDoS Solution

The tier-1 converged access CSP wished to deploy a cost-effective DDoS solution across its many datacenter locations; these datacenters are near the transit and exchange points, and serve millions of broadband fixed and mobile customers.

The general parameters of the challenge are:

• To concentrate the security functions into the existing/established datacenters and to dynamically re-route/forward any inbound and outbound traffic identified for DDoS mitigation through either the closest or most available datacenter using routing-based multi-homed services that leverage the routing infrastructure for service availability

• To be built upon a fully standard NFV architecture, to run on COTS hardware, to be available on-demand, to be repurposed on-demand for multiple simultaneous functions that may co-exist (beyond security), and to be capable of self-provisioning from the infrastructure up to service through a MANO stack; these requirements go well beyond simply having compute nodes available

Like many CSPs, they understood first-hand the enormous inefficiencies of legacy DDoS solutions, which typically take the form of proprietary hardware that is dimensioned for rare peaks—hardware that sits idle for the vast majority of time, consuming operational resources for no gains. The general costs associated with such large-scale DDoS mitigation (or “scrubbing”) platforms, from a perspective of hardware, network resources, IT personnel, and routing infrastructure, are burdensome for anyone requiring a DDoS solution—and for CSPs perhaps most of all, due to the scale and complexity of their networks, and due to their exposure to attacks.

Compared to these legacy solutions, an NFV-based DDoS platform that fully leverages NFV layers and provides on-demand scale-out to meet any sudden need is a game-changer.

Partner Ecosystem

This paper explains how Sandvine met the challenge, by working with partners Dell EMC, Intel, and RIFT.io.

<table>
<thead>
<tr>
<th>Sandvine Partner</th>
<th>Partner Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dell EMC</td>
<td>Provided the underlying infrastructure via the DSS 9000 rack scale infrastructure. Based on hyperscale principles, the DSS 9000 provides compute, storage, networking, power and cooling, and open management in a pre-integrated rack. Management is provided at the rack level and is based on the Distributed Management Task Force (DMFT) Redfish specification—industry standard open management APIs that ensure interoperability with heterogeneous systems.</td>
</tr>
<tr>
<td>Intel</td>
<td>Intel ® RSD enables CSPs to avoid vendor lock-in and to realize total cost of ownership benefits around shared power and cooling. While not utilized in this test, in a real-world deployment Intel ® RSD also enables native (i.e., non-VM) workloads to run side-by-side virtual machines in a common hardware infrastructure.</td>
</tr>
<tr>
<td>RIFT.io</td>
<td>Provided the autoscaling framework that allows the Sandvine virtual instances to scale as required by the attack load.</td>
</tr>
</tbody>
</table>
In combination, these technologies form a virtualized solution capable of automatic self-provisioning to meet the scaling requirements imposed by the largest DDoS attacks that can be unleashed on the Internet.

In addition to the DDoS component of this proof-of-concept, Sandvine took the opportunity to demonstrate a new and unique feature: a carrier-scale threat deception feature called QuickSand.

For both demonstration use cases, we leveraged the compute capabilities of the Sandvine VNFs: the Traffic Steering Engine is dynamically provisioned to meet the bandwidth and packet forwarding rate requirements imposed by a sudden DDoS attack on the network; the Policy Traffic Switch performs the dedicated security functions (e.g., the DDoS mitigation and scrubbing, and the threat deception).

Figure 1 shows the network topology used in this demonstration, overlayed on a simplified version of the CSP’s network:

- In the top-right of the diagram, we show a Security Pod, which is where the attack mitigation takes place; this same structure is repeated throughout the network, with each of the CSP’s datacenters housing a Security Pod.
- This implementation relies on Anycast to route traffic to the Security Pods; with this approach, once traffic is identified as needing to go to a Security Pod, the network’s routers send the traffic to whichever pod represents the least-cost destination.
- Anycast also automatically accounts for datacenter availability.
- Because we rely on Anycast, each Security Pod has the same network address (i.e., *lo:0).
- The different network Areas (e.g., Area0, Area1, etc.) represent OSPF (Open Shortest Path First) logical roles.
- Net0, Net1, etc. are routable networks.
- The large/thick, curved arrows represent Anycast tunnels that represent the book-ended flow of malicious traffic; with this book-ending, malicious and non-malicious traffic is routed into a Security Pod, and the non-malicious traffic is allowed to continue to the ultimate destination, in accordance with the challenge parameters.
Technical Details

This section explains in detail the components used in this demonstration.

The emergence of hyperconverged infrastructure has brought forward the ability for an entire compute/storage/network NFV infrastructure (NFV-I) to fit under a single MANO/VIM (Virtual Infrastructure Manager) stack. This fully integrated approach blurs the line between MANO and VIM by enabling orchestration and lifecycle management of the NFV-I under a single management stack.

Dell EMC DSS 9000

The Dell EMC DSS 9000 is a pre-integrated rack-level solution that incorporates insights from proven hyperscale design with the unique needs of large-scale organizations, including CSPs. It is software-defined infrastructure that's rooted in open standards, readily optimized for multiple workloads and specifically designed for easy management. The DSS 9000 is available in a variety of rack heights and is built to workload requirements and power usage, while scaling up to 96 nodes per rack.

This particular demonstration leveraged the Dell EMC DSS 9000 built with the specifications in Table 2.

<table>
<thead>
<tr>
<th>Components</th>
<th>Description</th>
</tr>
</thead>
</table>
| Hardware   | • Dell EMC DSS 9000  
             • 8x Dell EMC compute sleds  
             • 4x Dell EMC storage sleds  
             • 2x Dell EMC S4048-ON networking switches |
| Software   | • Canonical Ubuntu MaaS (Metal as a Service)  
             • Intel ® RSD Pod Manager  
             • Canonical OpenStack Ocata with libvirt 1.3.1  
             • RIFT.ware  
             • Ubuntu 16.04.2 LTS Host OS  
             • Ansible  
             • Sandvine Policy Traffic Switch (PTS) Virtual Series v7.30  
             • Sandvine Traffic Steering Engine v1.00 |

Management of the DSS 9000 is provided at the rack level and is based on the DMTF Redfish specification. Redfish is a standard, open API intended for the management of all hardware (i.e., compute, storage, and networking). It replaces IPMI-over-LAN and is a modern, scalable, secure management API.
Virtualized Cyber Security: Elastically Scalable DDoS Scrubbing and Threat Deception
A Sandvine Technology Showcase

Intel ® RSD

Combined with a Redfish-compliant hardware resource manager like Intel ® RSD, this becomes a key piece of the puzzle for transforming scale-out environments. Intel ® RSD software allows for the disaggregation of compute, storage, and network hardware resources, and introduces the ability to more efficiently pool and utilize these resources. It is built to address the needs of CSPs who want the ability to treat hardware resources as a service.

Intel ® RSD is a set of APIs and software, with Intel ® Pod Manager (PODM) at its heart. Intel PODM acts as a resource manager for disaggregated pools of hardware resources by communicating using standard Redfish APIs to hardware-aware software within the DSS 9000; this hardware-aware software is called the Pooled System Management Engine (PSME).

With Pod Manager and PSME, administrators don't need to know the physical location of infrastructure components; instead, when an administrator requests resources for a workload, the resources are located from anywhere in the rack through an automated discovery process, and they are allocated to the workload dynamically.

From a management and orchestration perspective, Dell EMC and Intel have collaborated using Intel ® RSD and OpenStack to allow for more simplified and automated management of scale-out resources on the DSS 9000. Dell EMC has integrated the software components to provide an easier and seamless “kickstart” for workload deployment. ¹

Table 3 provides a more detailed explanation of how the components work together.

<table>
<thead>
<tr>
<th>Technology Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Xeon processor E5-2650v4</td>
<td>The Intel Broadwell Xeon microarchitecture provided dense compute cores with low power footprint. The core density provided the ability to deploy all of the Sandvine VNF instances across multiple cores within each Intel Xeon processor.</td>
</tr>
<tr>
<td>Intel Xeon processor E5-2680v4</td>
<td></td>
</tr>
<tr>
<td>Intel Xeon processor E5-2630v4</td>
<td></td>
</tr>
<tr>
<td>Intel DPDK 16.07 LTS</td>
<td>Intel DPDK poll mode driver powers the data plane performance of the Sandvine PTS and TSE VNFs. With DPDK, the VNF packet forwarding performance is on par with network appliances.</td>
</tr>
<tr>
<td>Intel ® RSD</td>
<td>Intel ® RSD in the form of the Dell EMC DSS 9000 provided a platform where compute, network, and storage are available through a single orchestrated hardware environment with power and density efficiencies.</td>
</tr>
<tr>
<td>Intel ® RSD Pod Manager</td>
<td>The Intel Pod Manager and the Pod Management Foundation API provided the tools and APIs to integrate platform discovery, lifecycle, boot, configuration, and telemetry capability required to accomplish the orchestration capability of the NFV infrastructure.</td>
</tr>
<tr>
<td>Intel 82599 10 Gb Ethernet Adapter</td>
<td>The Intel 10 Gb network adapter combined with DPDK to provide high-throughput network connectivity and accessibility to the Sandvine VNFs.</td>
</tr>
<tr>
<td>Canonical OpenStack</td>
<td>Canonical OpenStack was used as the Virtual Infrastructure Management and Orchestration.</td>
</tr>
<tr>
<td>Canonical MaaS</td>
<td>Canonical MaaS, combined with Intel PODM and Redfish API provided the discovery, provisioning, and instantiation of compute host. This machine provisioning layer was required to prepare compute nodes for OpenStack.</td>
</tr>
<tr>
<td>Ansible Playbook</td>
<td>Playbook was used to deploy and execute the VNF packages and to perform telemetry for scale out.</td>
</tr>
<tr>
<td>RIFT.io</td>
<td>RIFT.io provided the service orchestration, deployment, and scale-out functionality as part of the MANO role.</td>
</tr>
</tbody>
</table>

¹ More information on the various software components involved, how they interact and the new orchestration capabilities they bring to rack scale data centers, is available in Transforming Management for Modern Scale-Out Infrastructure [http://en.community.dell.com/techcenter/extras/m/white_papers/20443552/download] and The OpenStack Kickstart Guide for the DSS 9000 (MAAS Edition) [http://en.community.dell.com/techcenter/extras/m/white_papers/20443553].
**RIFT.ware**

RIFT.io’s RIFT.ware software provides an autoscaling framework for VNFs that do not readily support scale-in/scale-out, to enable CSPs to deploy services that dynamically create new capacity to meet demand. RIFT.io’s Autoscaling Framework brackets any third-party, unmodified VNF with autoscaler VMs to provide a compound, dynamically scalable multi-VM VNF that responds to changing network demands.

RIFT.ware’s Autoscaler Framework works with MANO’s lifecycle management functions to support the scaling use cases described in the ETSI NFV specifications. The combination ensures new capacity (e.g., VMs or VNFs) is added and removed seamlessly from service, with minimal impact to upstream and downstream systems, and ensures even load distribution across all available capacity.

Additionally, the Autoscaler Framework also uses lifecycle management to automate all aspects of scaling a multi-VNF network service, including:

- Configuration of any new VNFs
- Rebalancing load across all VNFs
- Busy-out of user sessions, in the case of scale-in
- Reprogramming any load balancers in the network service

RIFT.ware also provides a high-speed, fully redundant, elastic autoscaler that can be used for stateful load distribution based on simple session identification (e.g., source IP address, IP 5-tuple)—the RIFT.ware Autoscaler is fully integrated with the RIFT.ware Autoscaler Framework and can be programmed via user-defined plugin scripts to update packet matching patterns.

The Autoscaler Framework supports both elastic input/output and elastic pools of service VMs and has been used in conjunction with many types of applications (e.g., firewalls, load balancers, routers, session border controllers, and evolved packet core gateways).

The Autoscaler Framework also supports flexible triggering mechanisms that allow automated actions based on simple threshold crossings or more sophisticated integration with third-party service assurance platforms over the MANO Os-Ma-nfvo reference point.

Using RIFT.ware, service providers can create complex use cases such as:

- Automated instantiation of new VNFs in response to degradation of user experience
- Automated recovery during loss of VNF or scaling group
- Proactive scaling based on capacity planning trends
- Reactive scaling due to outages based on pre-determined disaster recovery plans
Elastically Scalable DDoS Scrubbing

In this demonstration, Policy Traffic Switch virtual network functions perform as scrubbers to defend against a distributed denial of service attack. Based upon a DDoS attack that increases in magnitude, the network’s scrubbing capacity is automatically scaled with the addition of PTS VNFs.

Configuration

Table 4 shows the configuration for the major network elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Traffic Switch</td>
<td>Configured to scrub network traffic by detecting and blocking flow flood attacks:</td>
</tr>
<tr>
<td></td>
<td>package network_protection \</td>
</tr>
<tr>
<td></td>
<td>address_scan “false” \</td>
</tr>
<tr>
<td></td>
<td>flow_flood “true” \</td>
</tr>
<tr>
<td></td>
<td>mitigation_action “block” \</td>
</tr>
<tr>
<td></td>
<td>syn_flood “false”</td>
</tr>
<tr>
<td>Traffic Steering Engine</td>
<td>Configured to steer all traffic to the available Policy Traffic Switch instances.</td>
</tr>
<tr>
<td>RIFT</td>
<td>Configured to scale up a new scrubber (i.e., a Policy Traffic Switch VNF) for every 10,000 new flows per second of network traffic.</td>
</tr>
</tbody>
</table>

Steady State

Prior to the demonstration procedure, the network is in a steady state with a mix of traffic spread across a small number of flows (Figure 3 and Figure 4).

In the steady state, the number of ‘background’ flows fluctuates between about 20 and 30; the active flows represent a mix of traffic types and are of no specific importance to the DDoS scrubbing demonstration itself.
Part 1: Small Attack, No Scaling

In the first part of the demonstration, a flow flood attack is triggered with a rate of 4,000 flows per second, with a flow timeout of one second; Figure 5 shows the 4,000 attack flows added to the network's background traffic.

Sandvine Control Center summarizes and displays the traffic observed during five-second intervals; consequently, Figure 6 shows 20,000 attack flows observed and mitigated (i.e., scrubbed).
Since this attack volume is capably handled by a single PTS Virtual Series instance, RIFT has not yet triggered a second instance.

Figure 6: Sandvine Control Center showing the small attack during the first part of the demonstration; Control Center summarizes a five-second window, so the 4,000 flows per second is represented as 20,000 observed attack flows.
Part 2: Larger Attack and Dynamic Scaling

For the next part of the demonstration, the attack scale is increased to 12,000 flows per second (Figure 7).

Figure 7: Launchpad: Dashboard showing the 12,000 flows per second used in the second stage of the demonstration

For this demonstration, RIFT is configured to scale up a new PTS Virtual Series instance for every 10,000 flows; Figure 8 shows that the 12,000 flows per second attack has caused another PTS instance to begin scaling. In the meantime, the single initial PTS instance is able to handle the larger attack.

Figure 8: Ubuntu administration panel showing another PTS instance being scaled up
The Sandvine Control Center (which shows the 60,000 flows observed in each five-second interval) detects the new PTS Virtual Series instance (Figure 9), as does the Traffic Steering Engine. Once the new PTS instance is online, the TSE balances the attack load between the two available PTS Virtual Series instances.

Figure 9: Sandvine Control Center showing the 12,000 flows per second-attack used in the second part of the demonstration; notice that Control Center automatically detected the new PTS instance

This dynamic-scaling approach is easily extensible to DDoS attacks of any size.

**Threat Deception**

In this demonstration, a Policy Traffic Switch virtual network function deceives malware reconnaissance scanning, with the dual aims of:

1. **(Tactically) Protecting** network nodes by extending the time and resources it takes for an automated attacker to perform reconnaissance scanning
2. **(Strategically) Disrupting** the economics of the attack itself, thereby making the network an undesirable target

Specifically, the Policy Traffic Switch's QuickSand feature deceives a scan performed by the Nmap security scanner.6

---

6. Learn more at [https://nmap.org/](https://nmap.org/)
Configuration

Table 5 shows the configuration for the major network elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Traffic Switch</td>
<td>Configured to deceive reconnaissance scanning by blackholing TCP traffic and echoing UDP traffic:</td>
</tr>
<tr>
<td></td>
<td>package network_protection \</td>
</tr>
<tr>
<td></td>
<td>address_scan &quot;true&quot; \</td>
</tr>
<tr>
<td></td>
<td>port_scan &quot;psd&quot; \</td>
</tr>
<tr>
<td></td>
<td>function &quot;tarpitScanners&quot;() \</td>
</tr>
<tr>
<td></td>
<td>\</td>
</tr>
<tr>
<td></td>
<td>if expr(Flow.Layer4Protocol = Protocol.TCP) then \</td>
</tr>
<tr>
<td></td>
<td>tarpit(Type: Tarpit.Blackhole) \</td>
</tr>
<tr>
<td></td>
<td>if expr(Flow.Layer4Protocol = Protocol.UDP) then \</td>
</tr>
<tr>
<td></td>
<td>tarpit(Type: Tarpit.Echo) \</td>
</tr>
<tr>
<td></td>
<td>block \</td>
</tr>
<tr>
<td></td>
<td>) \</td>
</tr>
<tr>
<td></td>
<td>if expr(Flow.Detected.AddressScan or Flow.Detected.PortScan) then \</td>
</tr>
<tr>
<td></td>
<td>tarpitScanners() \</td>
</tr>
<tr>
<td>Traffic Steering Engine</td>
<td>Configured to steer all traffic to the available Policy Traffic Switch instances</td>
</tr>
</tbody>
</table>

Part 1: Scan without Deception

Figure 10 shows how the test network behaves (i.e., as a sum of nodes) when scanned by malware; Table 6 summarizes the results.

![Figure 10: Nmap results without threat deception](image)

<table>
<thead>
<tr>
<th>Number of IP Addresses Scanned</th>
<th>Ports Scanned per IP Address</th>
<th>Available Hosts Detected</th>
<th>Total Open Ports Detected</th>
<th>Total Scan Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>1,000</td>
<td>3</td>
<td>4</td>
<td>2.11 seconds</td>
</tr>
</tbody>
</table>
Part 2: Scan with QuickSand Threat Deception Enabled

Figure 11 shows how the test network behaves when the network is scanned while the QuickSand deception is active; Table 7 summarizes the results.

Figure 11: Nmap results when the QuickSand deception feature is active
A few important points are worth highlighting:

- The cost to the hypothetical attacker (e.g., in compute resources, in botnet rental time, etc.) has increased more than seven (7) times, from 2.11 seconds to 15.02 seconds.

- The attacker has learned very little about the network, so the next phase of the hypothetical attack will be untargeted; rather than learning that three hosts are active on a total of four ports, the attacker has been deceived into believing (or at least detecting) that 256 hosts are available on a total of 256,000 ports—the potential attack space has grown by a factor of 64,000!

Such deception tactics have been proven in enterprise networks, but have never been adapted for the high-scale world of communications service provider networks. This demonstration shows that these tactics are valid in the CSP environment; furthermore, by utilizing the scaling techniques of the previous demonstration, the approach can be extended to the scale of CSP networks in a straightforward manner.
Appendix: A Timeline of Sandvine’s Virtualization Milestones

Achieving Functionality

With any new technology wave, there is a gradual progression from a theoretical foundation, to proving fundamental concepts (e.g., functional interaction, performance explorations, etc.), and finally to demonstrations of commercial viability. From there, chasms are crossed and the technology is adopted until it eventually becomes ubiquitous wherever practical.7

Virtualization for network policy control functions is no exception; initially, Sandvine’s NFV activities focused on exploring and proving functionality:

• In May 2013, Sandvine teamed up with Heavy Reading to explore some of the theoretical issues around software-defined networking and NFV in a joint whitepaper Policy Control & SDN: A Perfect Match? In the press release announcing the paper, Sandvine stated that, “Some technical details will need to be worked out. Traffic needs to be steered in and out of the network functions cloud with full subscriber, state, and resource capacity awareness. To provide a consistent level of service to subscribers on a per-session basis requires session-aware load balancing, or partitioning, across resources.”

• In October 2013, Sandvine became the first network policy control vendor to demonstrate a completely virtualized solution, at the SDN & OpenFlow World Congress9

• In February 2014, Sandvine announced a business services offering based on a virtualized policy control platform10

• By July 2014, the Sandvine Virtual Series had been deployed by a number of CSPs11

Achieving Performance

Up to this point, the commercial deployments were relatively smaller scale, with customers deploying the Sandvine Virtual Series in locations with relatively low data plane throughput requirements and for which high-scale proprietary equipment was not economical.

With functionality proven, the next wave of benchmarks and milestones dealt with performance. In March 2015, a Sandvine blog post, The PTS Virtual Series: Virtualized Performance in the Real World, detailed how the PTS Virtual Series had achieved 155 Gbps in a single system Dell EMC PowerEdge R730x with Intel Xeon® E5-2699 v3 processors.12

This achievement was an important milestone, as it showed that virtualization could deliver the same performance density, at scale, as purpose-built equipment.

However, real-world carrier deployments often demand many hundreds of gigabits per second of performance. To achieve this scale, many hardware elements must be clustered and load-balanced together, cooperating to achieve the singular objective of implementing network policy control.

In October 2015, in a blog post called Breaking the Virtual Terabit Barrier and detailed in an accompanying Technology Showcase document, Sandvine demonstrated that virtualized solutions can exceed the performance and efficiency of proprietary solutions.13

This demonstration proved that NFV is a cost-effective alternative to proprietary network appliances, and that

virtualized policy control deployments can meet the scale, efficiency, and business objectives of communications service providers.

These performance milestones proved that capable COTS infrastructure, data plane acceleration technologies, and Sandvine's own virtualized solutions were sufficiently developed to meet the performance (e.g., throughput, density, power efficiency) needs of CSPs.

**Extending Functionality**

From this point, while the development and outright performance optimization of the Sandvine Virtual Series continues to be an engineering goal, the next major objective was to demonstrate the ability to truly exploit NFV and its promises of 'infinite scale' within a MANO framework to take virtualized solutions past their hardware appliance counterparts.

With these concepts in mind, Sandvine's attention turned to MANO and towards applying the unique advantages of NFV (e.g., elastic, flexible resource allocation in response to dynamic, evolving network needs, etc.) to the world's most pressing network challenges.

To address one significant challenge with implementing NFV and SDN at high scale, in October 2016, Sandvine released the Traffic Steering Engine, which enables CSPs to use intelligently steer traffic through Service Function Chains.\(^{14}\)

Sandvine engineered the Traffic Steering Engine to fulfil the proposed Service Function Forwarder (SFF) role of the IETF standard, as well as the proposed Traffic Steering Support Function (TSSF) from the ETSI/3GPP standards, making its function essential for operators interested in deploying NFV and Service Function Chaining using industry standards.

In January 2017, the European Advanced Networking Test Center validated several important functions of both the Traffic Steering Engine and Sandvine's virtualized Policy Traffic Switch:

- The Traffic Steering Engine can elastically scale-out new service chains as demand dictates, while ensuring the traffic distribution mechanism between multiple service chains remained intact
- The Traffic Steering Engine and PTS Virtual Series have failover and hitless recovery when in a high-availability scenario, ensuring CSPs can deploy Sandvine's virtualized products in situations requiring redundancy and network continuity

This validation further proved that virtualized network functions are viable substitutes for hardware solutions.

---
