The Edge Resource Center: Leveraging NFV and SDN for High Availability/High Performance Network Functions

Jeff DeMent, Erich Arnold, and Mircea Orban
ARRIS Group, Inc.

Abstract

Service providers around the world are embracing Network Function Virtualization (NFV) and Software Defined Networks (SDN) as enabling technologies to help increase service velocity and reduce costs. While some network functions may be suitable for deployment in a centralized data center, others (such as content caching and distributed denial of service attack (DDOS) detection and mitigation) are most effectively deployed at the network edge in order to meet the necessary latency targets and avoid degrading the user's quality of experience. This paper explores the concept of an Edge Resource Center, a small-scale virtual machine infrastructure (VMI) co-located with the CCAP devices in a cable head-end that supports high availability and high capacity SDN/NFV resources.

INTRODUCTION

Software Defined Networks (SDN) and Network Function Virtualization (NFV) are creating enormous interest as enabling technologies to help increase service velocity and reduce costs. SDN helps increase service velocity by adding a level of programmability to the network that's not typically found in purpose-built network elements. This programmability typically comes in the form of abstract RESTful web-service style application program interfaces (APIs) for configuring, monitoring, and managing network elements and services, allowing service provider and third party developers to quickly create new network applications. This promises the opportunity for increased automation of the service provisioning and service management functions, including the potential for end user self-provisioning, resulting in faster and more accurate deployment of new services.

Network Function Virtualization aims to reduce network operator CAPEX and OPEX by implementing network functions on commercial off-the-shelf (COTS) servers rather than on dedicated hardware appliances. NFV takes some of the technologies developed over the last decade to enable large scale multi-tenant data centers– namely virtualization of compute and storage resources, in many cases implemented with open source software – and applies them to network functions that have typically been implemented with ASICs, specialized packet processors, and embedded software. Using SDN or other techniques, multiple virtual network functions (VNFs) can be linked together into a *service chain*, applying the appropriate functions to each user's traffic in the proper order. Standard data center virtual infrastructure management software can be employed to create a dynamic elastic network service infrastructure where common compute, storage, and network resources are allocated to specific network functions based on demand.

One of the value propositions for SDN and NFV deployments proposed for cable MSOs is the opportunity to move network service functions onto standard virtualization infrastructure in large centralized data centers. While some network functions may be suitable for deployment in a centralized data center, others (such as content caching and

DDOS detection and mitigation) are most effectively deployed at the network edge in order to use network bandwidth more efficiently and to meet the desired latency targets to avoid degrading the user's quality of experience. This paper explores the concept of an Edge Resource Center (ERC), a small-scale virtual machine infrastructure (VMI) co-located with the CCAP devices in a cable head-end that supports high availability and high capacity SDN/NFV resources, along with the enhancements needed to allow the CCAP device to participate in the distributed NFV service chain.

PROBLEM STATEMENT

The motivation for MSOs to develop an architecture for network services based on SDN and NFV is well-documented: costs to satisfy exponentially increasing bandwidth demand are growing faster than service revenues, making rapid development and introduction of new revenue-generating services a necessity.

To meet the network operational goals needed to address this business challenge – increased service velocity, reduced equipment cost, and lower operational complexity – the industry is looking to technologies and solutions that have been successful for the large-scale cloud computing providers. Some of the desired characteristics are:

- A cloud-based architecture using standard IT virtualization technology
- Use of standard COTS servers to replace multiple network appliances, reducing costs
- Service scalability, including the ability to locate network functions at different locations
- Elastic compute and network capability for efficient resource utilization with changing service demands
- Network programmability, based on RESTful APIs, for rapid service creation and increased automation
- Open standards and interfaces, to avoid vendor lock-in
- Co-existence and compatibility with existing network equipment

This is an aggressive list and it remains to be seen whether many of these benefits can be realized. Some of the challenges to be addressed are listed below.

Packet latency: Diverting each user packet to a centralized data center and forwarding it through multiple service functions, each of which inspects the packet and makes forwarding decisions, can add significant latency and impact the user's perceived quality of experience for latency-sensitive applications such as gaming. When the service functions are running in virtualized environments, the packet latency can be further degraded by multiple hops through virtual switches and guest O/S stacks.

Software complexity/reliability: Mission-critical services need to be engineered to the same "5-9s" level of reliability as the existing access network equipment such as modern CCAP platforms. Development of fault-tolerant software has always been a challenging exercise, even in a single-vendor homogenous hardware environment. Integration of loosely-coupled software modules from multiple vendors into a highly-available service offering adds significant complexity to the modules themselves and to the SDN controller/orchestrator and associated applications responsible for managing the service.

Congestion avoidance and mitigation: Adding a centralized programmable SDN controller with a global view of network operating conditions would seem to simplify the job of steering user traffic around congested points to keep services operating efficiently. However, programmability can be a double-edged sword; SDN applications (potentially from different vendors) can choose conflicting recovery actions, resulting in congestion spread rather than abatement. Techniques to isolate congestion and apply consistent mitigation policies are needed.

The remainder of this paper provides an overview of some of the candidate SDN and NFV technologies under investigation and introduces the Edge Resource Center as an architectural model for addressing these challenges.

TECHNOLOGY BACKGROUND

SDN Overview

SDN traces its history back through at least three decades of research and experimentation in programmable networks in the telecom (e.g., Advanced Intelligent Network) and data networking fields [1], but has really gained traction with the development of the OpenFlow™ protocol in 2007-2008 and its subsequent standardization by the Open Network Foundation. This generation of SDN attempts to break the dependence on network equipment vendors (and the resulting long development times) to implement new network features by completely separating the control plane (e.g., policy and routing) from the forwarding plane. This allows the control plane to be moved to general purpose computing platforms, using OpenFlow technology to program the forwarding tables in the now simplified (and theoretically cheaper) data plane forwarding equipment.

In recent years the SDN focus, particularly in service provider networks, has shifted from complete decoupling of the control and forwarding planes, to augmenting existing networks with additional the programmability needed to rapidly design, prototype, test, and deploy new network services. This requires programmatic access to network configuration and monitoring functions as well as programmable forwarding. Accordingly, additional protocols beyond the OpenFlow protocol are now considered as part of the SDN toolset: NETCONF and SNMP for configuration and management, PacketCable™ Multimedia (PCMM) for policy and QoS in the cable access network, and others.

SDN Architecture

The main distinguishing feature of the SDN architecture is a software-based SDN controller, which discovers (or is configured with) the network topology, provides abstract "northbound" APIs to applications for network control and management, and communicates forwarding, configuration, and policy instructions to the network elements it controls. A number of both open source and proprietary SDN controllers have been developed over the past few years. Currently one of the more popular controllers is from the open source OpenDaylight project, referred to from here on as ODL (figure 1).

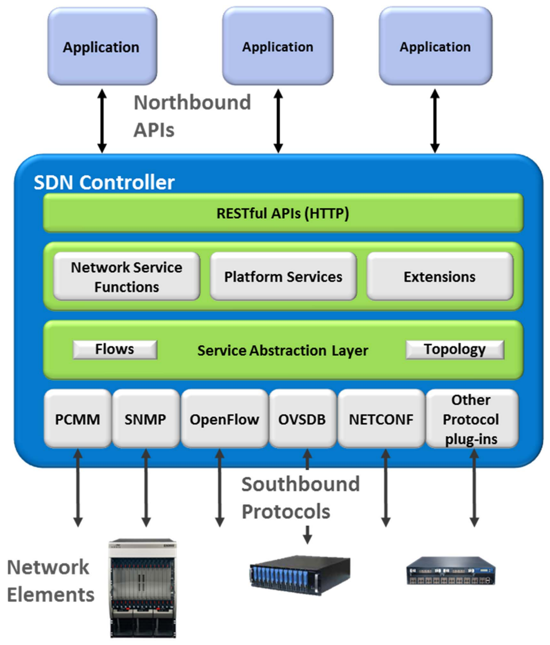

Figure 1: OpenDaylight Controller Architecture

The ODL controller consists of an extensible model-driven service abstraction layer that performs the mapping between API requests from applications (or internal higher-level services) and the southbound protocols that execute the requests via interactions with network elements such as switches (physical and virtual), routers, and CCAP devices. Southbound protocols are implemented as Java™ plug-ins, allowing new protocols to be added without requiring frequent new releases. ODL's capability to support legacy protocols such as SNMP and PCMM, as well as vendor-specific protocols, is an important feature for interoperating with existing cable network infrastructure.

Early SDN proponents favored a logically centralized SDN controller (although possibly distributed across multiple compute nodes for increased capacity and/or redundancy), on the theory that a centralized controller with global knowledge of the network topology and current conditions could help optimize packet forwarding decisions in the network elements. While this may be feasible for smaller networks, it becomes impractical for large geographically dispersed service provider networks with a variety of equipment types. SDN architectures for service provider networks today generally include multiple domain-specific controllers, for example, an access-network controller, data-center controller, and core network controller as illustrated in figure 2. These controllers may then be coordinated by an orchestration layer or "controller of controllers."

The prototype ERC includes a highly available OpenDaylight SDN controller cluster that controls the CCAP devices, local switches, and virtual switch infrastructure for the entire head-end. It's anticipated that at some point in the future the ERC controller may contain an interface to an orchestrator or higher-level controller that coordinates the activities of all head-ends and other network regions.

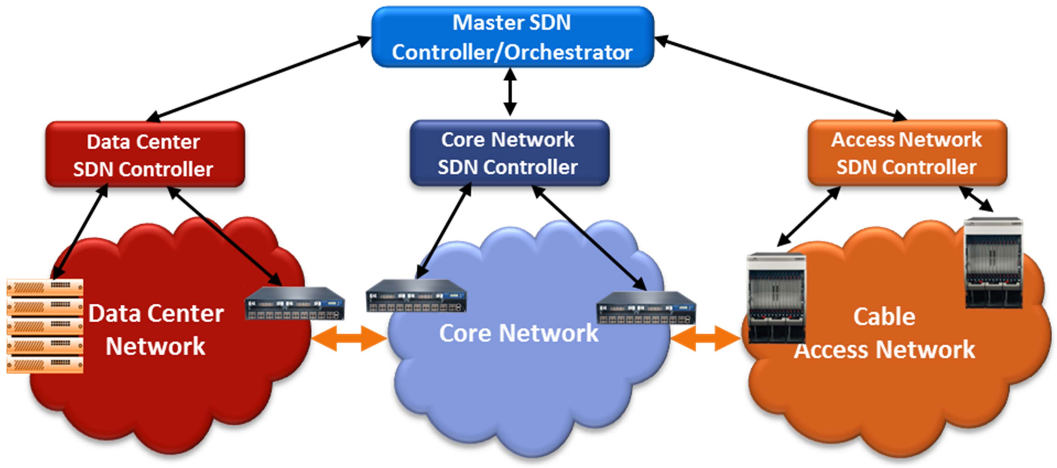
Master SDN Controller/Orchestrator
Data Center SDN Controller
Core Network SDN Controller
Access Network SDN Controller
Data Center Network
Core Network
Cable Access Network

Figure 2: Distributed Domain-specific SDN Controllers

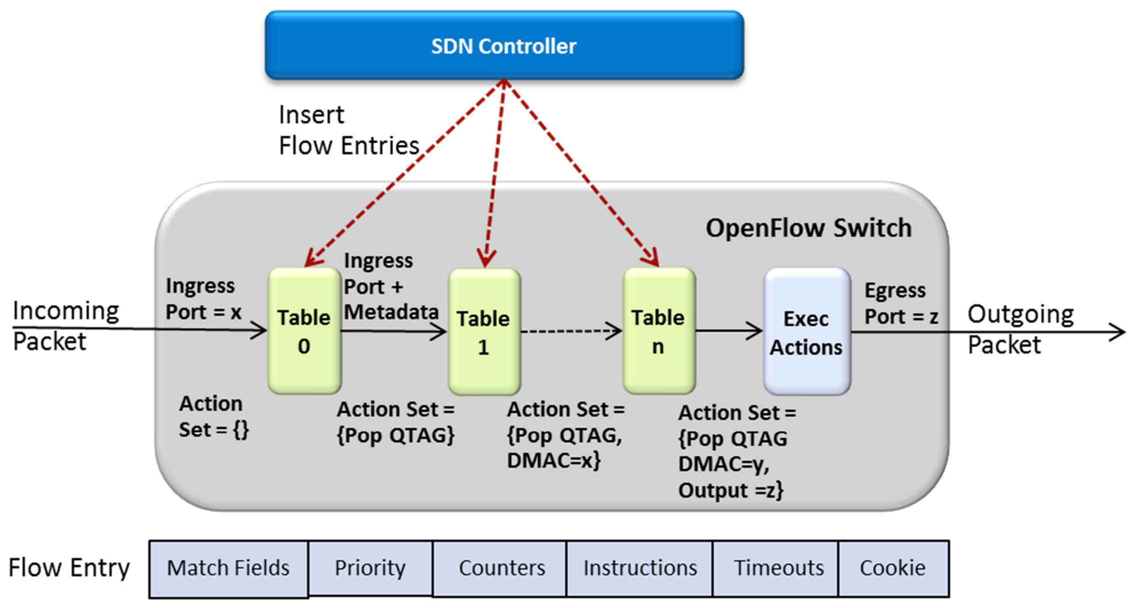

Figure 3: OpenFlow Switch Model

<u>OpenFlow Protocol</u>

The OpenFlow protocol provides a standardized interface for a control process (e.g., an SDN controller) to program flow tables in switches and routers [2]. The OpenFlow switch model is comprised of one or more flow tables, with each table entry containing a matching rule (set of header fields, including wildcards) used to match against incoming packets, along with a set of actions to perform on matching packets such as output port selection, header field overwrites, and counts to increment (figure 3). Later versions of OpenFlow technology allow multiple tables to be chained together, with packet metadata (intermediate results) passed from table to table to be used as additional matching criteria, and the resulting actions accumulated until the end of the chain. The OpenFlow switch initiates a secure connection to its designated controller to advertise capabilities and accept flow programming commands.

While OpenFlow technology has been proposed as a replacement for the embedded switch/router control plane, we propose here using it to augment the existing control plane in cable access network elements. Adding OpenFlow capability to a CCAP device allows for fine-grained dynamic traffic steering of packets at the network edge. For example, reacting to a detected DDOS attack by steering all packets destined for the attack target through a stateful firewall that scrubs out the attack packets while allowing the valid traffic to pass. OpenFlow protocols can also be used to program virtual switches to direct flows between network functions in the ERC.

<u>Other SDN Protocols</u>

Several existing protocols have also been repurposed for use in SDN, particularly for support of service provisioning via the SDN controller. We have already mentioned PCMM, a cable-specific protocol for configuring dynamic DOCSIS® service flows with appropriate QoS attributes, and SNMP for network management, particularly status and statistics collection. Both are implemented in the prototype ERC.

Two other SDN-related protocols that are being considered for future use in the ERC are

NETCONF and BGP-Link State (BGP-LS). NETCONF is an IETF®-originated protocol for modifying configuration data in network elements. In particular it differentiates the "running" (active) configuration data store from the "startup" configuration data store and "candidate" configuration data stores that may become active when committed. NETCONF data stores are modeled by the YANG data modeling language, which can be directly mapped to XML. The NETCONF protocol is based on a remote procedure call (RPC) model.

BGP-LS is a set of extensions to the well-established Border Gateway Protocol to allow BGP-speaking network elements to share link state and traffic engineering information with external agents such as an SDN controller. Its main purpose is to enable topology discovery in the SDN controller [3].

Network Function VirtualizationOverview

NFV aims to change the network operator cost structure by moving from the current practice of deploying a separate hardware-based network appliance per function to a model with all software-based functions on running COTS hardware. With NFV, each server is capable of performing multiple roles by leveraging standard IT industry compute, storage, and network virtualization techniques [4].

The ETSI NFV Industry Specification Group has taken the lead in publishing a number of documents, including general requirements, a reference architectural framework, use cases, and best practices for performance and portability. The ETSI NVF architecture includes a standard virtual machine infrastructure (VMI) capable of hosting one or more virtualized network functions (VNFs), each of which may also have an element manager to manage it. The architecture is agnostic to the actual virtualization technology (e.g., hypervisor) employed.

The reference framework also includes an NFV management and orchestration component (known as MANO) which is responsible for lifecycle management of the VNFs [5]. The MANO is responsible for normal FCAPS functions. It also has the responsibility for understanding the operational characteristics of the VNF (called key performance indicators, or KPI) and scaling the VNF up by adding resources (for example, CPU or memory) or creating new VM instances, or scaling down by reallocating resources or removing VM instances. The Orchestrator component of the MANO also integrates with the operator's OSS/BSS system.

The prototype ERC deploys a highly available OpenStack® cluster to implement the MANO virtual infrastructure management component. VNF management (e.g., dynamic scaling of VNF capacity) and orchestration are topics for future research.

VMs and Docker™ Containers

Virtual Machine technology has been widely deployed in data centers for Web-based applications and is the basis for NFV. VM technology is usually implemented with a hypervisor that logically partitions a server's physical resources – CPUs, memory, disk space, and network interfaces – and allocates the resource slices to individual VMs. Each VM contains its own operating system instance (called the guest O/S), allowing for different operating systems to be running in different VMs on the same physical server. These extra layers of software add overhead and can be the source of additional packet latency for data plane VNFs.

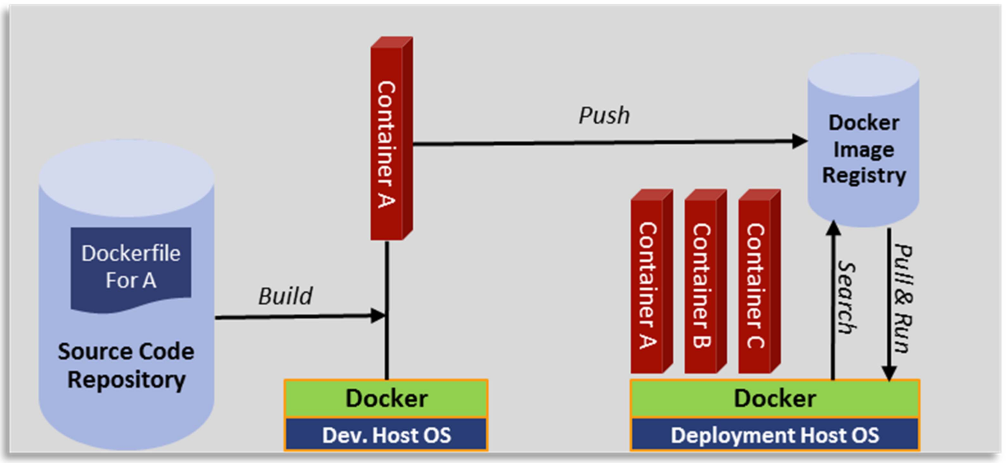

Figure 4: Example Docker Build and Deployment Model

Docker containers are independent but complimentary technology that has become popular recently. The Docker architecture is an open platform for distributing and deploying cloud applications based on the Linux container model [6]. Docker containers are used to provide an isolated runtime environment for the application that encapsulates all components, libraries, and configuration dependencies. Docker containers are hardware and platform-independent, allowing applications packaged as Docker containers to run on many different platforms (both physical and virtualized) and improves portability across different operating systems. Figure 4 illustrates the Docker build and deployment model for applications packaged in Docker containers.

A Docker container differs from a virtual machine in that it only encapsulates the application and its dependencies, but not the guest operating system. This makes it lighter weight than a VM – it runs as a process in user space on a host operating system – but still provides isolation from other applications running on the same platform. Note that the actual deployment platform for the application may be either a physical server or a VM itself.

EDGE RESOURCE CENTER OVERVIEW

An Edge Resource Center (ERC) is a small-footprint virtual machine infrastructure (VMI) co-located with a group of CCAP devices in a cable head-end that provides highly available/high performance SDN/NFV resources (figure 5). Locating these resources at the network edge helps to control packet latency for network functions requiring minimal and/or predictable latency to provide users with the desired quality of experience. It can also help reduce network bandwidth utilization by keeping traffic that can be processed locally in the head-end rather than back-hauling it to a centralized data center, only to be forwarded back to the originating head-end.

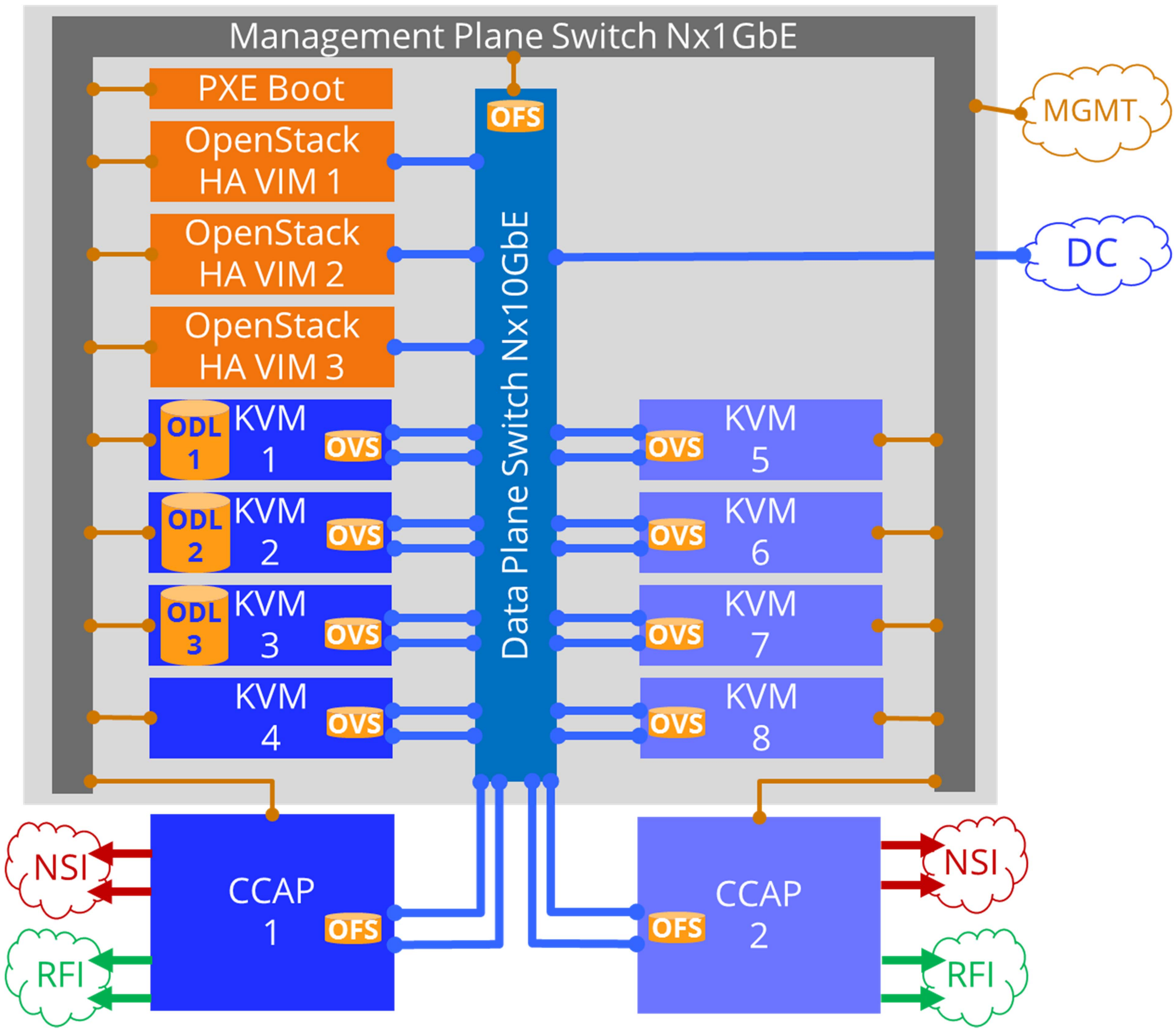

Figure 5: Edge Resource Center Model – Scaled for 2 CCAP devices

ERC Functional Components

The primary components of the ERC are:

COTS Server Array: The ERC hosts are low cost COTS servers running the KVM (Kernel-based Virtual Machine) hypervisor that provide high capacity elastic compute and network resources. Each server runs an Open vSwitch (OVS) instance that provides layer 2 switching between VMs in the same server as well as access to external networks, and can be controlled via OpenFlow technology. This configuration supports a large population of lightweight VMs that provide local subscriber services such as virtualized firewalls, deep packet inspection (DPI) filters, edge caches, and virtual CPE. The servers typically provide only minimal storage resources, but can be upgraded to support edge services such as content caching that have more significant storage requirements.

OpenStack VIM: The ERC VMI is managed by a local OpenStack Virtual Infrastructure Manager in a high-availability cluster controlling the KVM hypervisor hosts (compute/storage nodes) in a distributed redundant configuration. OpenStack software provides a web-based dashboard for creating and removing VM instances, configuring virtual networks, and managing virtual storage volumes. It also includes an image service for managing VM images with different guest operating systems and configurations.

OpenDaylight SDN Controller: An OpenDaylight SDN controller (ODL) cluster is also hosted in a high-availability VM configuration on the KVM hosts, to support dynamic network configuration and automation. The ERC ODL configuration includes a PCMM protocol plug-in to support dynamic service flows with specified QoS attributes in the cable access network, as well as OpenFlow protocol support for dynamic service chaining and SNMP for configuration and monitoring.

Switched Management Network: An Nx1 Gbps switched management network connecting all devices.

Switched Data Plane Network: An Nx10 Gbps switched data plane network connecting the servers and CCAP devices, with the potential to connect to remote data centers. The data plane switch also utilizes OpenFlow technology.

CCAP Devices: The CCAP devices provide the termination point for the cable access network. The ERC CCAPs include a prototype OpenFlow switch capability to augment their standard routing and switching capability.

Service Availability and Performance

One of the primary goals of the ERC is to support virtual network functions at the same "5-9s" level of reliability as the existing CCAP platforms. Achieving this level of reliability requires that both the network function itself and the supporting infrastructure services – i.e., the OpenStack VIM and the OpenDaylight SDN controller – operate at this level of reliability.

Since the underlying server hardware does not typically meet this target, software redundancy techniques are needed to meet this goal. Both OpenStack and ODL are deployed in the ERC as 3-instance clusters, using existing open source clustering and data replication technologies. Interfaces to applications employ virtual IP addresses that can float between cluster instances.

High availability for individual network functions running on the ERC is up to the VNF developer. Different redundancy models are used depending on whether the VNF is stateful or stateless. Stateless VNFs are commonly deployed in an active/active model using load-balancing. A failure of a VNF instance using this model can affect as little as a single transaction, with remaining instances taking over the load until a new VNF instance can be created. Availability for stateful VNFs is a little more complicated, typically requiring multiple instances of the VNF in active/backup arrangements, with state data replicated or check-pointed from the active instance to the backup instances. A failure of the active instance in this case requires interaction with the SDN controller to reconfigure the traffic flow towards the backup instance, with a resynchronization phase later when the failed VNF instance is restored. The same clustering and replication technologies employed by OpenStack software and ODL may be used by VNFs using either model.

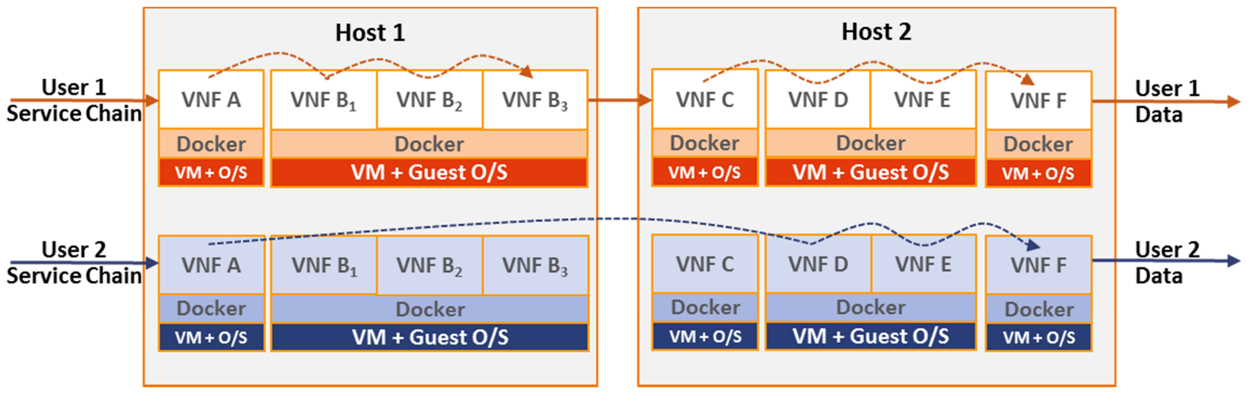

Figure 6: The VM Container Ship Model

Key performance characteristics for any deployed network function are bandwidth and packet latency. In virtualized environments, delivering predictable bandwidth and packet latency depends on many factors: the capability of the underlying hardware (e.g., number of CPU cores, threading capability, memory and I/O bandwidth), the performance of the virtualization software and O/S stacks (including the virtual switch implementation), and the loading factors of other functions sharing the platform to name just a few.

The ETSI NFV Industry Specification Group has published a set of best practices for performance and portability of NFV data plane workloads to help guide network operators in making appropriate choices for virtualization infrastructure [7]. The recommendations include suggested requirements for the architecture and capabilities of both the underlying hardware and hypervisor/host O/S. Also included are proposed templates for specifying the capabilities of a compute host and its virtualization infrastructure and for specifying the requirements of a VNF virtual machine image. The virtual infrastructure manager or orchestration system could then match VNF image requirements against host capabilities when deciding where to locate the VNF.

The VM Container Ship Model

One of the models under evaluation in the prototype ERC to help manage both performance and availability of VNFs is the use of a combination of VMs and Docker containers to group related VNFs or multiple components of a single logical VNF. In this model, the VM acts as a “container ship” running the related Docker containers as independent processes on the VM’s guest O/S (figure 6).

This model can have several advantages.

- The related VNF components share a single guest O/S image and can use more efficient inter-process communication techniques than switching between VM images through the virtual switch.
- The related VNF components can comprise a single “fault group” with a *shared fate*. This means that the failure of a single VNF component fails the group. This can simplify the design of the VNF components since each component does not have to be designed to deal with the failure and recovery of its dependent components; instead the entire VM instance can be destroyed and restarted.

- Multiple small VNFs in a per-subscriber service chain could be co-located in a single VM image. This "vertical slicing" of related VNFs at the per-subscriber level can limit the impact of a VM failure to a single subscriber and take advantage of the improved performance and simplified recovery strategy described above.

USE CASES

Two possible use cases for ERC-based services are discussed below: DDOS detection/mitigation and network DVR playback over IP. The DDOS protection use case illustrates the use of SDN with the OpenFlow protocol for dynamic service chain creation, while the nDVR playback case illustrates the use of SDN with PCMM for dynamic bandwidth allocation in the cable access network.

DDOS detection/mitigation

MSO-provided DDOS protection is a good candidate for deployment using an SDN and NFV-based architecture in the ERC. A typical implementation for DDOS protection today in an MSO network might be to connect a dedicated DDOS appliance supporting a small number of subscribers in the data path between the Internet and the CCAP device serving the subscriber. The DDOS appliance examines all traffic to and from the subscriber, mostly just passing it through except for the rare times when an attack is in progress.

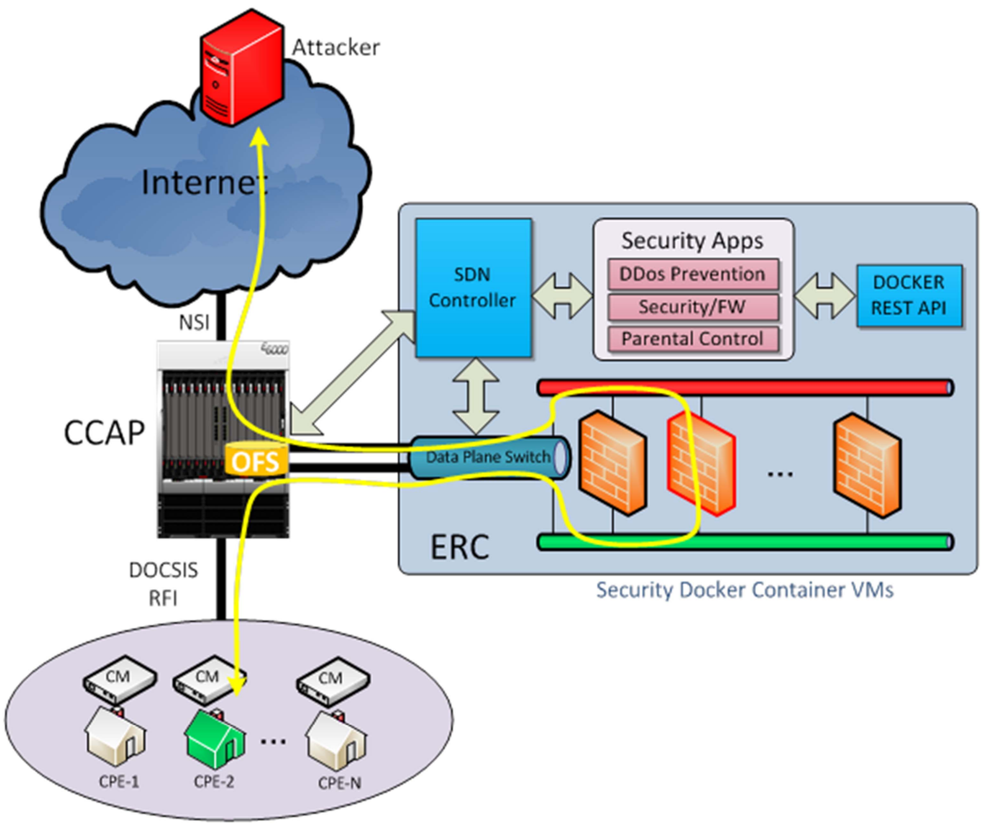

Figure 7: Example DDOS Detecton/Mitigation Service in ERC

An SDN/NFV approach using ERC resources (figure 7) instead would have 2 components:

- A DDOS detection application implemented as an SDN application, using OpenDaylight northbound APIs to monitor traffic statistics and derive usage patterns for DDOS protection subscribers.
- A virtualized stateful firewall network function deployed in the ERC as a Docker container or as a standalone VM. The firewall can be dynamically created when a DDOS attack is detected or, for faster response or simplified load balancing, a firewall instance can be statically created for each subscriber.

The dynamic DDOS protection service works as follows.

1) The user logs into a self-service application to provision the DDOS service. Depending on the details of the service, the IP address or addresses to be protected may be derived from this request, or might be manually entered by the subscriber (other provisioning scenarios are possible, not considered further here).
2) The self-service application notifies the DDOS detection application of the subscriber IPs to be protected. The DDOS detection application begins monitoring traffic statistics for the designated IP addresses via the ODL instance in the head-end serving the user. All data traffic continues to be passed straight through the network to/from the user device.
3) When the onset of a DDOS attack is detected, the DDOS application requests a stateful firewall instance to be allocated in the ERC. Using the OpenFlow protocol (via ODL), the application inserts flow entries in the serving CCAP device to forward all traffic to and from the attacked IP addresses through the stateful firewall VNF.
4) The firewall VNF monitors traffic in both directions to identify legitimate downstream traffic, passing it back through the CCAP device to the user. Attack traffic is discarded at the firewall.
5) Once the attack has ended, the DDOS application removes the CCAP flow entries for that subscriber, and traffic flow resumes directly from/to the user without going through the stateful firewall VNF. The firewall VNF may be destructed at this point or may just go idle, waiting for the next attack on that subscriber.

Network DVR Playback

The IP nDVR playback use case (figure 8) uses the PCMM capability of the ERC SDN controller to provision a new service flow across the DOCSIS network with desired QoS characteristics for the user's playback session. The new service flow serves 2 purposes: (1) allocating the required DOCSIS bandwidth for the video playback session depending on the video quality (SD, HD, or UHD) and (2) ensuring the video playback is not counted in the user's normal High Speed Data service counts.

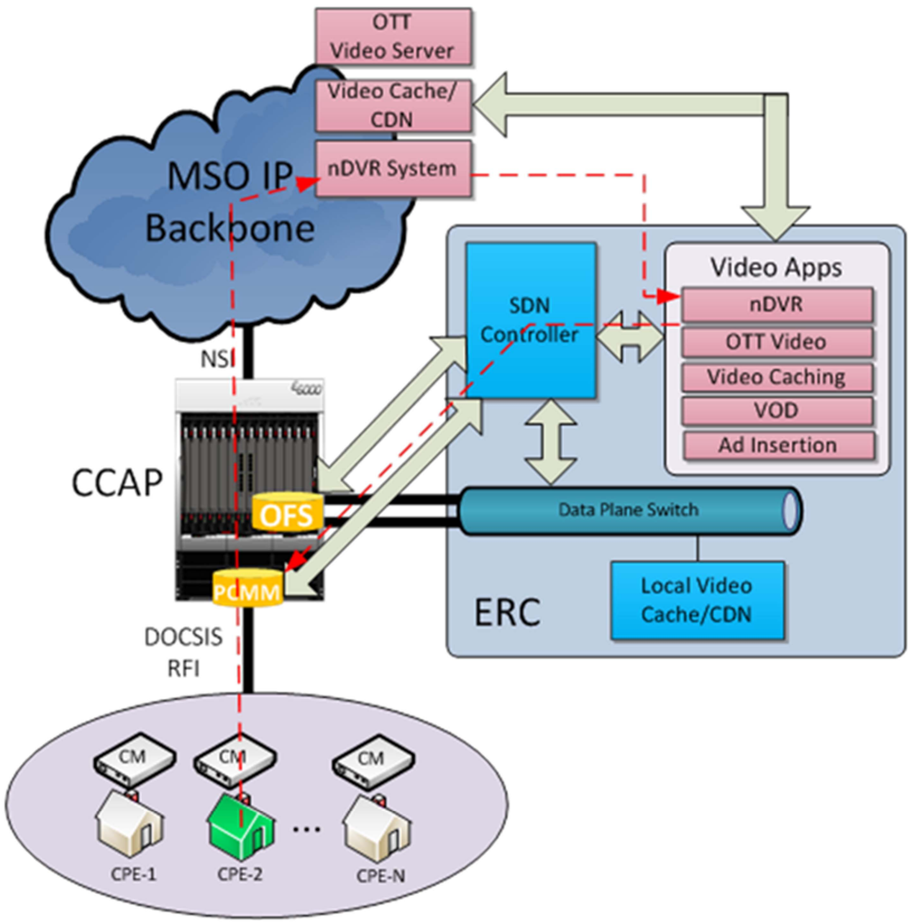

Figure 8: ERC nDVR Playback Example

The nDVR playback operation is fairly simple.

1) At the time of the playback request, the nDVR server notifies the nDVR application in the head-end ERC of the user's request, including the destination IP address and port number to be used for playback and the required bandwidth.
2) The local nDVR application uses the ODL PCMM API to provision a new service flow for the playback session, including the classifiers (IP address & port number) to identify the playback traffic and the required QoS settings.
3) The downstream playback session IP packets are classified to the newly created downstream service flow by the CCAP device serving the subscriber.
4) At the end of the playback session, the nDVR server notifies the local nDVR application to remove the associated service flow.

<u>CONCLUSIONS & FUTURE WORK</u>

Network operators have embraced SDN and NFV as critical technologies for improving service velocity and reducing costs, but some network functions have latency and availability requirements that may be difficult to meet using a standard centralized IT cloud infrastructure. Locating these functions at the network edge can be an effective strategy to ensure predictable performance that provides users with the desired quality of experience. It can also help reduce network bandwidth utilization by keeping traffic that can processed locally in the head-end rather than back-hauling it to a centralized data center, only to be forwarded back to the originating head-end.

We propose deploying these services in an Edge Resource Center (ERC), a small-scale virtual machine infrastructure co-located with the CCAP devices in a cable head-end/hub that supports high availability, high capacity SDN/NFV resources. The ERC includes high-availability OpenStack and OpenDaylight clusters and supports a large population of lightweight VMs that provide local subscriber services such as NFV firewalls, DPI filters, edge caches, and virtual CPE. Adding OpenFlow capability to the CCAP devices takes advantage of their native packet classification functions to add programmable dynamic service chaining capability to the ERC.

The ERC supports a deployment model that groups related VNFs or VNF components packaged in Docker containers onto a single VM that acts as a "container ship" to help offset the overhead introduced by the hypervisor and vSwitch for packets that traverse multiple VNFs. The VM container ship also acts as a fault group for the related VNF components to help simplify VNF failure and recovery strategies.

As SDN and NFV technologies are evolving rapidly there will surely be new capabilities needed in the ERC. The following are just a few of the capabilities being considered for future study:

- Adding additional service chaining capabilities such as the Network Service Header (NSH) being standardized in the IETF SFC work group for passing packet metadata between VNFs;
- Adding NETCONF or other protocols to add additional support for automating service provisioning;
- Integrating the ERC with an orchestration system, or "controller of controllers", to support distributed service chains with reach beyond the head-end; and
- Adding data analytics capabilities to facilitate new applications in the ERC.

REFERENCES

[1] J. R. a. E. Z. N. Feamster, "The road to sdn: an intellectual history of programmable networks," *ACM SIGCOMM Computer Communication Review,* vol. 44, no. 2, April 2014.

[2] N. McKeown and others, "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Computer Communication Review,* vol. 38, no. 2, April 2008.

[3] H. Gredler, J. Medved, S. Previdi, A. Farrel and S. Ray, *North-Bound Distribution of Link-State and TE Information using BGP,* IETF draft draft-ietf-idr-ls-distribution-10.

[4] ETSI Industry Specification Group for NFV, "Network Functions Virtualisation – Introductory White Paper," October 2012. [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper.pdf.

[5] ETSI GS NFV-MAN 001 V1.1.1, *Network Functions Virtualisation (NFV); Management and Orchestration,* 2014-12.

[6] "What is Docker?," [Online]. Available: https://www.docker.com/whatisdocker/. [Accessed 4 March 2015].

[7] ETSI GS NFV-PER 001 V1.1.2, *Network Functions Virtualisation (NFV); NFV Performance & Portability Best Practises,* 2014-12.