San Diego State University

Department of Electrical & Computer Engineering

SOFTWARE DEFINED NETWORKING

Project Report for EE665

Spring 2015

Prepared by

Arunima Koneripur Prasad (818459337)

Iswarya Shanmukhi Gabbita (818457998)

Table of Contents

Table of Figures

ABSTRACT

Our particular research involved the study of Software Defined Networking. Our main focus being the architecture, design, concept, applications and key challenges of software defined networking, we also analyzed the network modelling and applications of software defined networks.

SDN is advancing at a rapid rate, impacting various technologies and is predicted to bring about a revolution in the field of networks. Some of its important applications we would like to probe into are as follows

- Infrastructure as a service (IaaS) - SDN virtual networking combined with virtual machines and virtual storage can emulate elastic resource allocation.
- SDN especially OpenFlow, has been used in a carrier's production network to provide virtualized networking for cloud service. The virtualized network is also extended to inter datacenter connection services for data migration without IP renumbering, and for bandwidth provisioning/ de provisioning on demand.
- Another application is that the SDN architecture may enable, facilitate or enhance network-related security applications due to the controller's central view of the network, and its capacity to reprogram the data plane at any time.

1. Introduction

The networking industry is growing at a rapid pace and this has resulted in the inability of existing networking architectures to meet current market scenarios. One of the primary limitations of today's networks is complexity. In order to modify an existing network, various parameters need to be considered. This has made current architectures relatively static and the network is unable to dynamically adapt to the changing traffic patterns, varied applications, and ever increasing user demands. The complexity of today's networks makes it very difficult to apply a consistent set of policies to increasing mobile users, which increases the risk of security breaches, and other negative consequences. Another major issue faced is that of scalability. As demands rapidly increase, the network becomes more complex and there is a need to scale networks to provide high-performance. Such scaling cannot be done with manual configuration.

This mismatch between current market requirements and network capabilities has led the industry to look into new options. In response, the industry has created the software-defined networking (SDN) architecture and is developing associated standards.

2. Need for a new architecture

The growing mismatch between the network requirements and available resources demanded the need for a new approach to networking. Some of the factors that led to this are as follows.

- CHANGING TRAFFIC PATTERNS
  Within a service provider, traffic patterns have changed significantly. In contrast to client-server applications where the bulk of the communication occurs between one client and one server, today's applications access different databases and servers, before returning data to the end user device. At the same time, users are changing network traffic patterns as connect from anywhere, at any time.

- CONSUMERISATION OF IT
  Users are increasingly using mobile personal devices such as smartphones, tablets, and notebooks to access the network. There is therefore an increased pressure to accommodate these personal devices while protecting data and intellectual property.

- RISE IN CLOUD SERVICES
  Today’s service providers have accepted both public and private cloud services, resulting in unprecedented growth of these services. But they now want access to applications, infrastructure, and other resources. Providing such a service, whether in a private or public cloud, requires scaling of computing, storage, and network resources.

3. Concept

Software-defined networking (SDN) is an architecture that is dynamic, manageable, cost-effective, and adaptable, seeking to be suitable for the high-bandwidth and dynamic nature of today's applications. It is a new approach to computer networking that allows the abstraction of lower-level functionality. This is done by decoupling the control plane from the data plane. [5]

The control plane is the system that makes decisions about where traffic is sent. It is part of the router architecture that deals with drawing the network map, or the information contained in a routing table that decides what has to be done with incoming packets. The control plane logic can also define packet prioritization, or if a certain packet has to be discarded.

On the other hand, the data plane is concerned with actual traffic forwarding. It is part of the router architecture that deals with what has to be done to the incoming packets. It refers to a table in which the router looks up the destination address of the arriving packets and retrieves information necessary to determine the path that has to be taken for that packet.

In the SDN architecture, the control plane and data plane are decoupled, network intelligence and state are logically centralized and the underlying network infrastructure is abstracted from the applications. One of SDN's defining characteristics is that it centrally places the intelligence of a network system. While this increases the flexibility of network utilization, it also keeps its complexity hidden from operators—which is why SDN is easy to operate and maintain. [5]

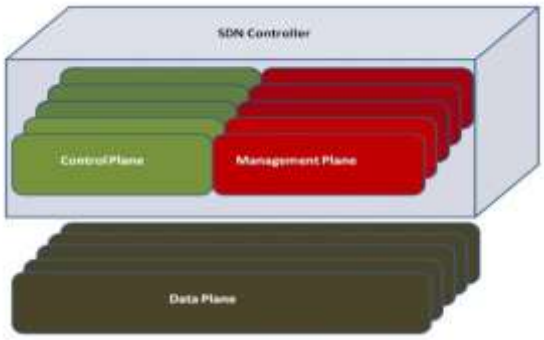

Figure 1: SDN Implementation

[Photograph] Retrieved from http://forums.juniper.net/t5/The-New-Network/Applications-of-Software-Defined-Networking/ba-p/170626

4. Architecture

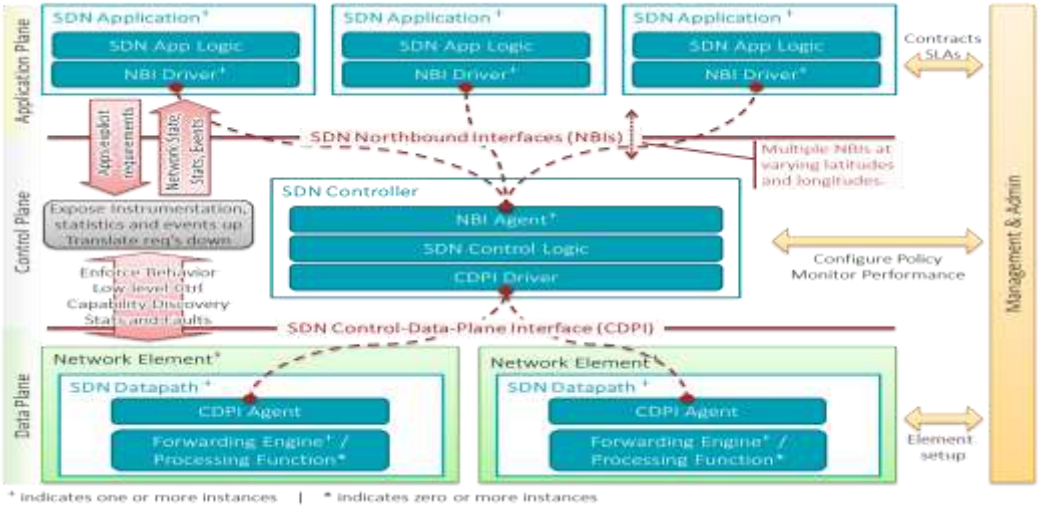

Figure 2: SDN Architecture

[Photograph] SDN Architecture Overview (PDF), Version 1.0, December 12, 2013.

SDN Application (SDN App):

SDN Applications are programs that explicitly, directly, and programmatically communicate their network requirements and desired network behavior to the SDN Controller via Northbound Interfaces (NBIs). In addition they may consume an abstracted view of the network for their internal decision making purposes. [5]

SDN Controller:

The SDN Controller is a logically centralized entity in charge of translating the requirements from the SDN Application layer down to the SDN Datapaths and providing the SDN Applications with an abstract view of the network. [5]

SDN Datapath:

The SDN Datapath is a logical network device, which exposes visibility and uncontended control over its advertised forwarding and data processing capabilities. It comprises a CDPI agent and a set of one or more traffic forwarding engines and zero or more traffic processing functions. These engines and functions may include simple forwarding between the datapath's external interfaces or internal traffic processing or termination functions. [5]

SDN Control to Data-Plane Interface (CDPI):

The SDN CDPI is the interface defined between an SDN Controller and an SDN Datapath, which provides at least programmatic control of all forwarding operations, capabilities advertisement, statistics reporting and event notification. [5]

SDN Northbound Interfaces (NBI):

SDN NBIs are interfaces between SDN Applications and SDN Controllers and typically provide abstract network views and enable direct expression of network behavior and requirements. This may occur at any level of abstraction (latitude) and across different sets of functionality (longitude). [5]

In SDN, the southbound interface is the OpenFlow (or alternative) protocol specification. Its main function is to enable communication between the SDN controller and the network nodes so that the router can discover network topology, define network flows and implement requests relayed to it via northbound APIs.

5. OpenFlow Protocol

OpenFlow is the first standard communications interface defined between the controls and forwarding layers of an SDN architecture. The protocol is mostly used for the southbound interface of SDN, which separates the control plane from the data plane. OpenFlow was initially proposed by researchers at Stanford University and is now standardized by the Open Networking Foundation (ONF). OpenFlow allows direct access to and manipulation of the forwarding plane of network devices such as switches and routers.

Many a times the OpenFlow concept is confused with that of SDN. It must be noted that, OpenFlow does not equal SDN and is merely a subset of it. SDN emphasizes applications that drive network usability and business requirements, while OpenFlow is a technology, to link an SDN controller and network devices. In other words, OpenFlow is the protocol that's implemented with SDN. These two concepts co-exist but are not interchangeable.

The OpenFlow architecture consists of three basic concepts. (1) The network is built up by OpenFlow-compliant switches that compose the data plane; (2) the control plane consists of one or more OpenFlow controllers; (3) a secure control channel connects the switches with the control plane. [7]

5.1 OPENFLOW SWITCH

An OpenFlow compliant switch is basically a forwarding device that forwards packets according to its flow table. This table has a set of entries, which consists of match fields, counters and instructions.

Header Fields	Counters	Actions

Figure 3: Flow table entry for OpenFlow 1.0

[Figure] W.Braun and M.Menth, "Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices,"

The "header fields" in a flow table entry describe to which packets this entry is applicable. They consist of a wildcard-capable match over specified header fields of packets. The "counters" are reserved for collecting statistics about flows. They store the number of received packets and bytes, as well as the duration of the flow. The "actions" specify how packets of that flow are handled. Common actions are "forward", "drop", "modify field", etc. [7]

5.2 BASIC PACKET FORWARDING

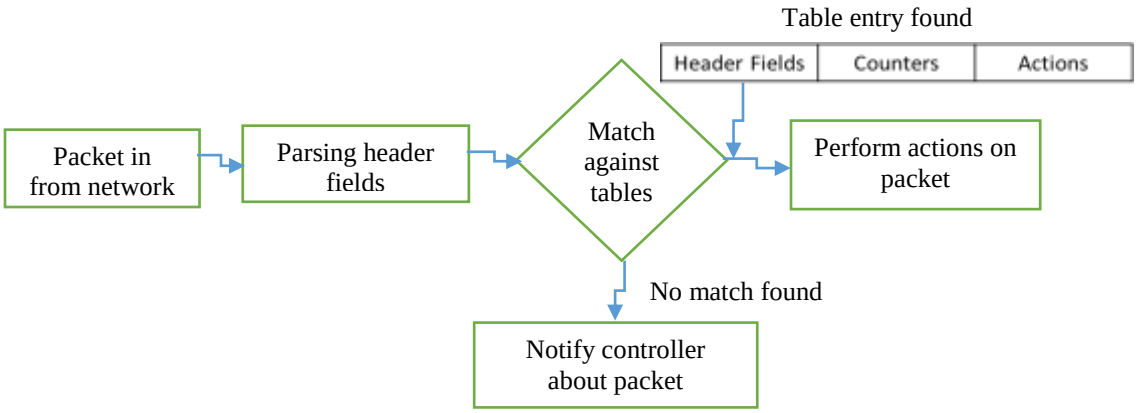

Figure 4: Basic packet forwarding with OpenFlow in a switch

[Figure] W.Braun and M.Menth, "Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices,"

The basic packet forwarding mechanism with OpenFlow is illustrated in Figure 4. When a switch receives a packet, it parses the packet header, which is matched against the flow table. If a flow table entry is found where the header field wildcard matches the header, the entry is considered. If several such entries are found, packets are matched based on prioritization, i.e., the most specific entry or the wildcard with the highest priority is selected. Then, the switch updates the counters of that flow table entry. Finally, the switch performs the actions specified by the flow table entry on the packet, e.g., the switch forwards the packet to a port. Otherwise, if no flow table entry matches the packet header, the switch generally notifies its controller about the packet, which is buffered when the switch is capable of buffering.

6. Network Modelling

6.1 JOINT ALLOCATION & SCHEDULING

In a SDN-enabled network, forwarding ability is not the only one but control ability can explicitly provide the other interesting one for network applications with a standard control protocol, such as OpenFlow. OpenFlow switches use flow table to cache network control rules. TCAM is currently a major implementation of flow table for hardware - based looking up and forwarding. However, this kind of special-purpose memory is costly and energy eager so that the capacity of memory cannot be very large. Current generation of OpenFlow switches can support almost 1K flow entries. Since the destination address based one-dimension looking up and forwarding in the traditional IP network faces the challenge of expandability, then a multi-dimension flow-level caching in SDN will surely cause "flow-explosion". Switches with limit forwarding capacity are far from enough to accommodate all of network control rules. Therefore, flow table, besides of bandwidth, will become the other key network resource to impact control abilities in consideration of the limit capacity of current OpenFlow switches [8].

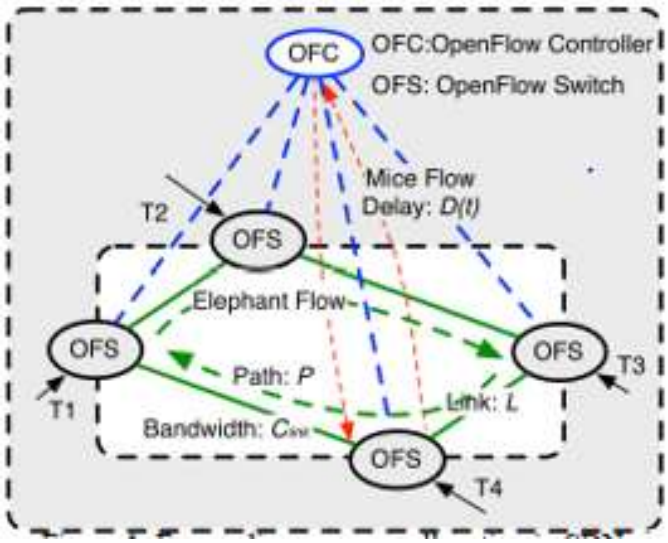

Figure 5: Forwarding resource allocation in SDN

[Figure] T.Feng, J.Bi, K.Wang, *"Joint Allocation and Scheduling of Network Resource for Multiple Control Applications in SDN,"*

6.2 ANALYZING AND MODELING RESOURCE ALLOCATION IN SDN

A. Problem Statement

To effectively allocate network resource, MPLS TE is a current popular operation using many tunnels between ingress-egress router pairs. The tunnels of MPLS TE are

marked with priorities and different types of services. Switches can map different DSCP bits in the IP header to different priorities queues. However, there are two problem of MPLS TE to allocate network capacity. One is poor efficiency caused by a local, greedy resource allocation model of MPLS TE. The other is poor sharing caused by the lack of network-wide fairness. Therefore, our goal of forwarding plane is to carry more traffic and support flexible network-wide sharing, as shown in Fig. 5. In Fig. 5, OpenFlow switches (OFS) are connected in the data plane of SDN while an OpenFlow controller (OFC) is running in the control plane of SDN. The centralized OFC estimates service demands and computes shortest forwarding paths for each service according to a global network view and services weight (e.g. the value or priorities of a service). When services weigh t are equal, the OFC will allocate bandwidth in the max-min manner.

On the other hand, because the f low table capacity of every OpenFlow switch is limited, flow table in OpenFlow switches cannot cache all of flows generated by control applications. Internet traffic feature with power law distribution means a few popular prefixes will contribute most of traffic load and the rest prefixes have only trivial or even no traffic for a certain period of time. Then, sacrificing some forwarding performance and loading some popular flows into switches is a trade-off in the condition of the limitation of flow table. Meanwhile, experimental research show the popularity of a prefix only has short term stability [8].

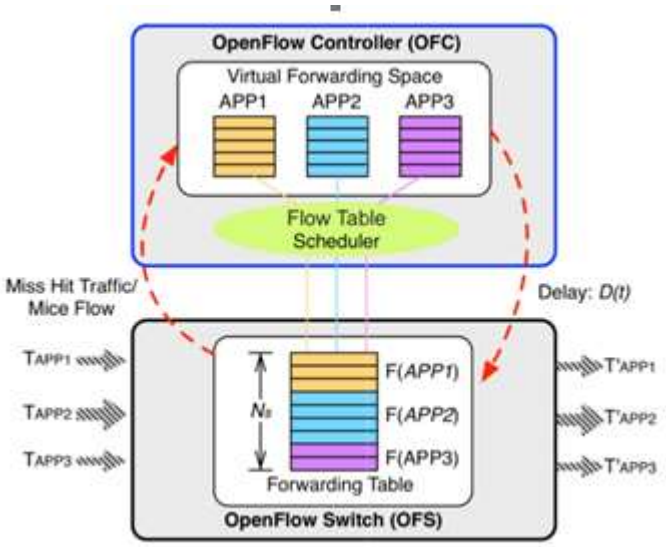

Figure 6: Flow table allocation for multiple applications in SDN

[Figure] T.Feng, J.Bi, K.Wang, *"Joint Allocation and Scheduling of Network Resource for Multiple Control Applications in SDN,"*

Because the traffic on a prefix can be bursty, and a popular prefix cannot have heavy traffic loads for all the time, a dynamic scheduling for flow table is required in an interval. During the interval, unmatched flows will be sent to the controller and redirected to output port through the control plane. The traffic volume of a flow across a SDN network is invariable as long as the path is fixed during a scheduling interval. This

process of redirection will only introduce the delay of transmission from the switch to the controller compared to forwarding directly in the data plane. More detailed, the delay of such process in SDN consists of the transmission time of unmatched packets to the controller, the computing time of control application for the packets and the transmission time back to switch. Keeping flow entries in the hardware flow table will therefore avoid such delay of packet forwarding, shown as in Fig. 6.

Generally speaking, not only link bandwidth but also flow table capacity should be taken into consideration in the network resource management of SDN. Both of these two resources should be allocated according to bandwidth cost and delay cost with the pay-for-use of different control applications. More bandwidth means higher transmission rate, while more flow table capacity means lower transmission delay [8].

B. Results

VFS will decide each flow whether to be forwarded by hardware flow table or not according to the hit rate of a flow. If a flow is popular, it will be forwarded by hardware flow table directly. If a flow is not popular, it will be redirected to the controller. Through the measurement of the packets, the mean delay of every flow and the global mean delay of SDN network is measured. To compare the global mean delay with deferent scheduling policy, random scheduling policy is also implemented, as shown in Fig. 7. The mean packet delay of the network with the current scheduling model is lower than random scheduling model. When the size of flow table is very limited, this delay is large, as we increase the flow table size, both algorithm can produce less delay time. As we reach another extreme, that is, the flow table size is unlimited, both algorithm generate the same result since now there is no flow selection involved [8].

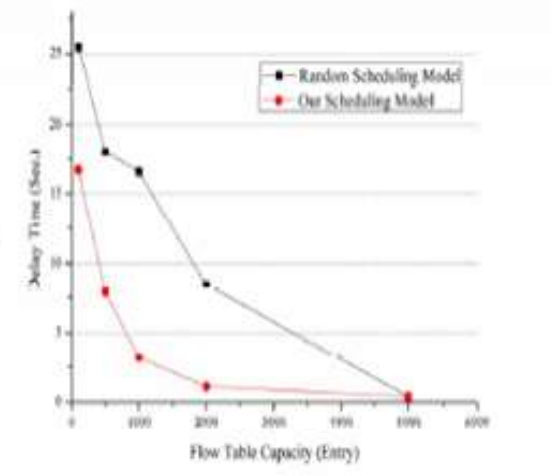

Figure 7: Delay Time Comparison for the two policies

[Figure] T.Feng, J.Bi, K.Wang, *"Joint Allocation and Scheduling of Network Resource for Multiple Control Applications in SDN,"*

6.3 **APPLICATION AWARE ROUTING**

Software-defined network (SDN) architecture allows service providers to build networks with increased application awareness, which can be built into the network by developing SDN controller applications that keep track of application-level characteristics and use that intelligence to provision flow into the network switches [9]. Some of the application-level characteristics and intelligence that can be leveraged for building content-routing applications are:

SERVICE AVAILABILITY

Traditional network monitoring services check only the L2 link or L3 network path availability. However, there may be instances when the content-delivering application could be down or unavailable. For example, an Apache server delivering Web content may not be running, or the content server may not be capable of delivering the content using the requested protocol; or a user may request a video using HTTP protocol despite the server being capable of delivering content only using RTSP protocol. Content-routing applications can be designed to perform service-availability checks before provisioning flows in the network switches [9].

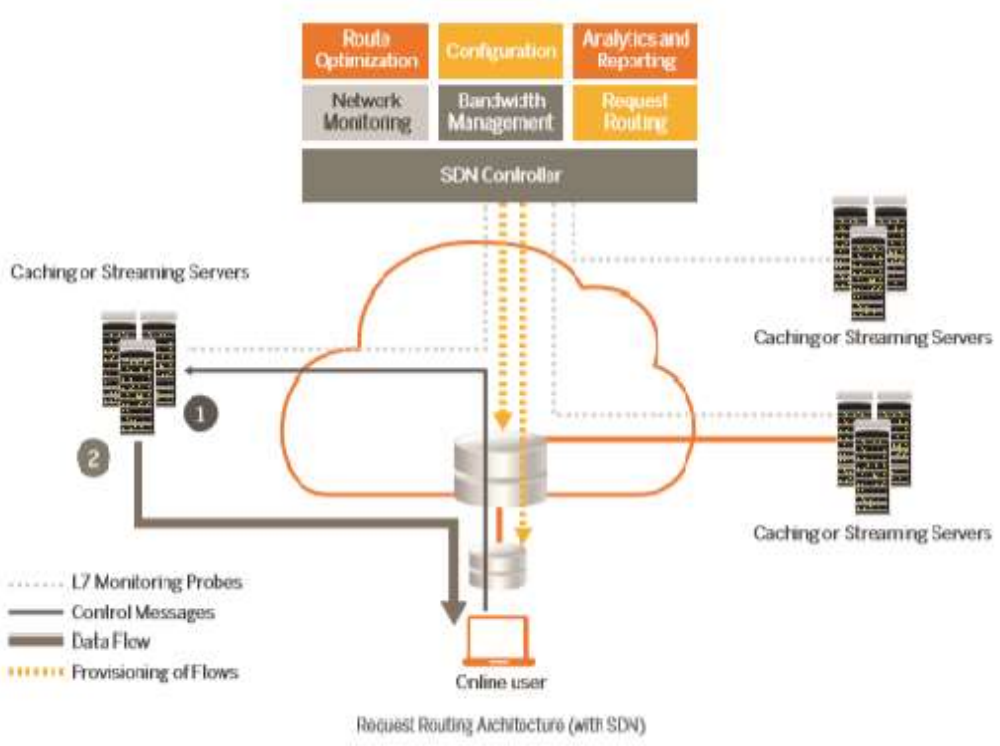

Figure 8: Request Routing Architecture

S. Velrajan, "Application Aware Routing in Software Defined Networks," Aricent Networks.

CONTENT AVAILABILITY

A content-routing application can check the availability of content in a content server before routing requests to the server. In a CDN, content is pre-populated or stored dynamically in caches/content servers based on the content type, cache ability, popularity, content size, etc. A content-routing application can communicate with caches or content servers in the network to gain intelligence about content availability. This intelligence can then be used to route requests to the appropriate server where the content resides [9].

BANDWIDTH MANAGEMENT

Content-routing applications can periodically monitor bandwidth requirements and re-provision flows in the network switches to match the bandwidth, latency, and QoS requirements of the service. For example, if a premium user is watching online video, the content-routing application can route such user requests through a network path that with a bandwidth capacity and lower latency. This ensures no buffering of video, and therefore a better user experience. When network conditions change (due to congestion, link failure, etc.), the flows can be re-provisioned automatically by the content-routing application so that user traffic is routed through a new optimal network path [9].

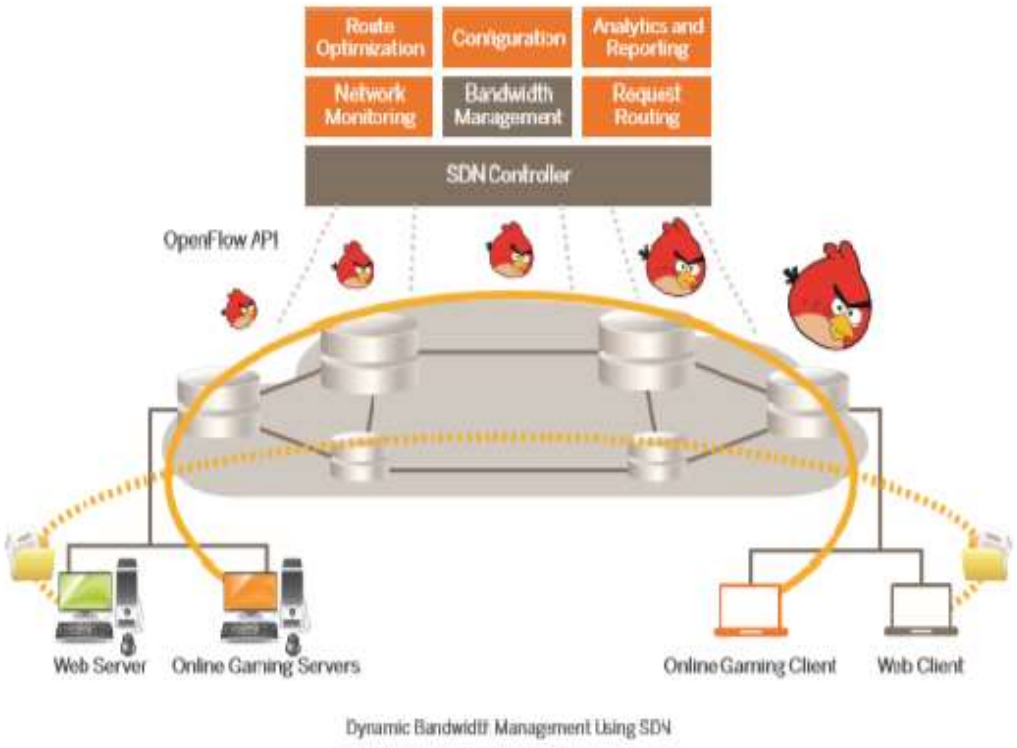

Figure 9: Dynamic bandwidth Management

S. Velrajan, "Application Aware Routing in Software Defined Networks," Aricent Networks

Content-routing applications running on an SDN controller can make routing decisions based on application-level insights and characteristics. This provides a scalable architecture for application-aware request routing. SDN-based architecture eliminates the need for a dedicated or special-purpose content router in the CDN network. It provides granular control for the CDN service providers to influence the quality of experience (QoE) for users and dynamically take corrective actions based on the network conditions. Application-aware routing using SDN architecture allows service providers to route requests based on content availability, service availability, and application-level bandwidth/latency requirements without losing sight of changing network conditions.

7. Applications

7.1 USE CASES FOR SDN

USE CASE	ROLE OF SDN
Bandwidth on demand	Enable programmatic controls on carrier links to request extra bandwidth when needed.
Performance on demand	API-driven service in which the network dynamically ensures not only that the appropriate capacity is available for a given application, but also that a guaranteed level of performance is delivered.
Dynamic WAN interconnects	Create dynamic interconnects at interchanges between enterprise links or between service providers using high-performance switches.
Dynamic WAN reroute – move large amounts of trusted data bypassing expensive inspection devices	Provide dynamic, yet authenticated, programmable access to flow-level bypass using APIs to network switches and routers.
Virtual edge / CPE	In combination with NFV, replace existing customer premises equipment (CPE) at residences and businesses with light-weight versions, moving common functions and complex traffic handling to the service provider edge or data center.
Virtual private cloud	Operator offers a virtual cloud service within its network that provides enterprise-grade security, performance and control attributes associated with a private cloud.
Service chaining / traffic steering with SDN	In SDN networks, the ability to associate the right traffic to the right appliance, thus making use of the appliance only when needed. (Can be used in conjunction with NFV.)
Network virtualization – multi-tenant networks	To dynamically create segregated topologically-equivalent networks across a data center, scaling beyond typical limits of VLANs today at 4K.
Network virtualization – stretched networks	To create location-agnostic networks, across racks or across data centers, with VM mobility and dynamic reallocation of resources.
Network slicing	Having several customers each running different types of protocols over different virtual topologies based on a single physical network.

Figure 10: SDN Role in Use Cases

7.2 **SDN APPLICATIONS OVERVIEW**

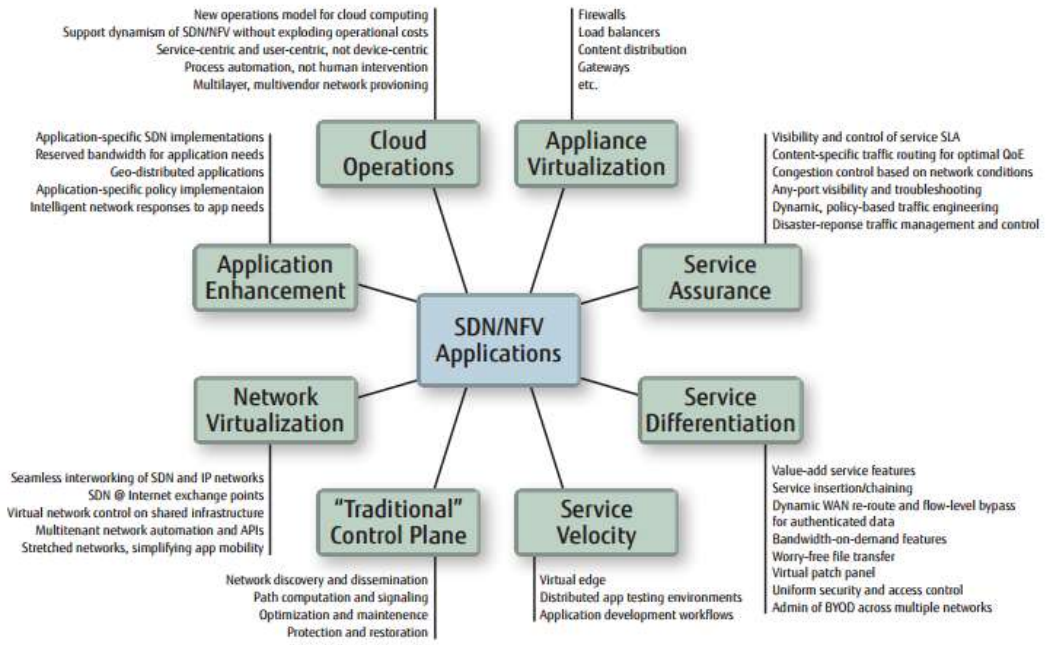

Figure 11: Application Overview of SDN

7.3 **INFRASTRUCTURE AS A SERVICE (IaaS):**

Infrastructure as a Service is a form of cloud computing that provides virtualized computing resources over the internet. In this model, a third-party provider hosts hardware, software, servers, storage and other infrastructure components on behalf of its users. The resources offered can be adjusted on-demand. This makes IaaS well-suited for workloads that are temporary. The IaaS customers pay on a per-use basis. This eliminates the expense of deploying in-house hardware and software.

Figure 12: IaaS Platform

[Photograph] Retrieved from http://quersystem.com/consolidacion-de-servidores/solucion-infraestructura-como-servicio-iaas/

7.4 INTEL & WINDRIVER ACCELERATED vSWITCH SOLUTION

Intel and Wind River designed a virtualization solution that entirely bypasses the vSwitch in the Linux kernel in order to boost packet switching throughput [10]. While the companies kept the control path the same, they took the vSwitch data path out of the Linux kernel and recreated the data plane in Linux user space by building switching logic on top of the Intel DPDK library. The diagram below illustrates how customers can take a vApp (i.e., load balancer, firewall, etc.) and serve all the VMs on the platform.

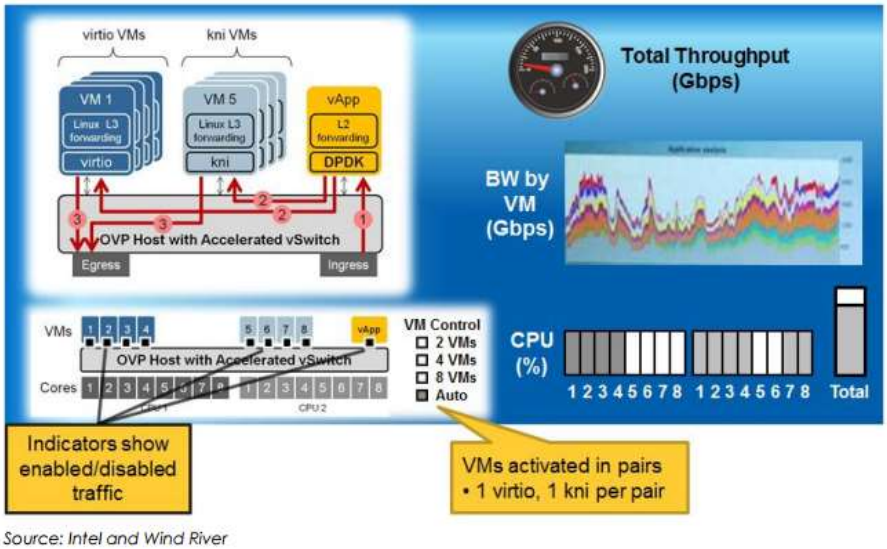

Figure 13: vSwitch Implementation

S. Perin and S.Hubbard, "Practical Implementation of SDN & NFV in the WAN," HP, Intel and WindRiver, October 2013.

Intel and Wind River use a shared memory model, so that when packets come in the vSwitch analyzes the header and does not have to copy the packet. The vSwitch passes a pointer to get better overall performance [10].

BENEFITS OF ACCELEREATED vSWITCH

The Intel/HP/Wind River partnership, as exemplified in the Accelerated vSwitch demonstration, is a real example of a supplier eco-system built specifically around SDN and NFV and based on open standards and low-cost hardware [10].

- The Accelerated Open vSwitch is a "horizontal" platform that can be used across multiple use cases emerging for both SDN and NFV.
- The accelerated vSwitch solution can remove the vApp as a bottleneck and thereby enable higher VM utilization and up to a tenfold improvement in throughput compared to the standard Open vSwitch.
- The solution also addresses latency in a virtual environment, which is a requirement in certain applications.

8. Key Challenges

SDN holds great promise in terms of simplifying network deployment and operation along with lowering the total cost of managing enterprise and carrier networks by providing programmable network services. However, a number of challenges remain to be addressed. This section focuses on four specific questions arising from the challenges of SDN [11].

- **Performance vs. Flexibility: How can the programmable switch be achieved?**

One fundamental challenge of SDN is how to handle high touch, high security, high performance packet processing flows in an efficient manner. There are two elements to consider; performance and programmability/flexibility [11].There are a number of initiatives underway to allow programmability of existing network technologies in a manner conformant with the goals of SDN. Beyond these, the SDN programmability and performance problem remains a challenge to achieve node bandwidth beyond 100Gbps.

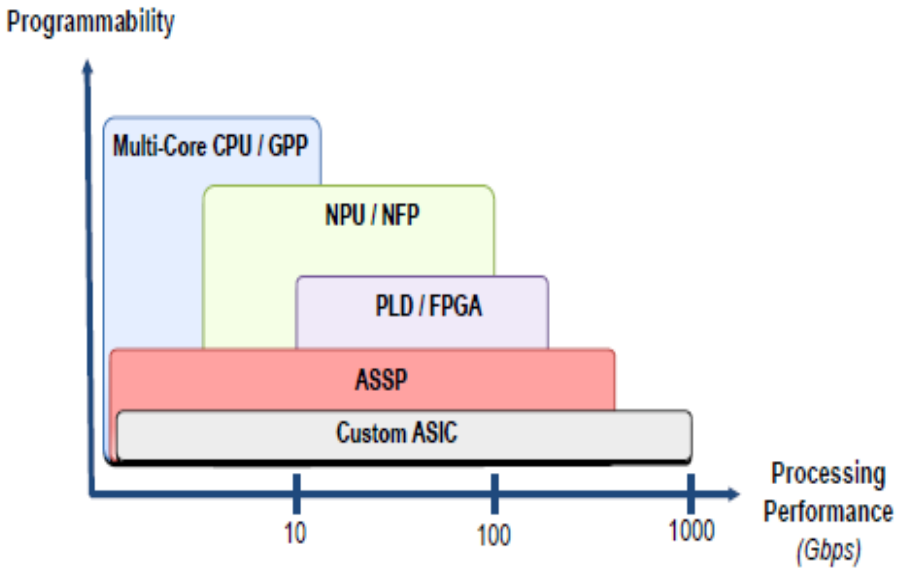

Figure 14: Network Processing - Performance vs. Programmability

S. Sezer, S. Scott-Hayward, P. K. Chouhan, "Are we ready for SDN? - Implementation Challenges for Software-Defined Networks,"

General Purpose Processors (CPU/GPP) provide the highest flexibility. High-level programming languages and design tools enable the highest design abstraction and the rapid development of complex packet processing functions. The limitation of CPU implementation, however, is its performance and power dissipation, constrained by the general purpose architecture.

Network Flow Processors (NPU/NFP) are optimized processor architectures for network processing. Dedicated hardware accelerators and various interface technologies are used for acceleration while reducing power dissipation. However, the flexibility of

implementation is reduced as more detailed knowledge of the device is required in order to define the packet/flow processing function and to take full advantage of the device's parallel processing capabilities.

Programmable Logic Devices (PLD) or Field Programmable Gate Arrays (FPGAs) have evolved into a technology for telecommunication and network processing. In comparison to microprocessors, PLDs are configured using hardware design tools. This technology is ideal for implementing highly parallel and pipelined data paths that are tailored for individual network processing functions.

Application Specific Standard Products (ASSP) are the cornerstone of high-performance networks. They are designed and optimized for widely used functions or products aiming for high-volume. The drawback of ASSPs is their limited flexibility. Core ASSP domains are physical and data-link layer products, switching and wireless products.

Application Specific Integrated Circuits (ASIC) are proprietary devices custom-built by system vendors (e.g. Cisco, Huawei, Juniper etc.) when standard products are unavailable and programmable solutions are unable to meet performance constraints. As an application-specific solution, ASICs offer the lowest flexibility while providing the highest performance, power and cost benefits.

Taking into account the programmability/performance trade-off of data processing technologies, it is evident that only a hybrid approach will provide an effective technology solution for SDN. Main SDN node functions can be decomposed into clusters of sub-functions such that feature-specific technologies are used to satisfy the best performance versus programmability trade-off in terms of power dissipation, cost and scalability.

- **Scalability: How to enable the Controller to provide a global network view?**

The issue can loosely be split into controller scalability and network node scalability. The focus here is on controller scalability in which three specific challenges are identified. The first is the latency introduced by exchanging network information between multiple nodes and a single controller. The second is how SDN controllers communicate with other controllers using the east and westbound APIs. The third challenge is the size and operation of the controller back-end database [11].

Within a pure SDN environment, a single controller or group of controllers would provide control plane services for a wider number of data-forwarding nodes, thus allowing a system-wide view of network resources. Other approaches that match the goals of SDN with existing routing protocols involve addition of an orchestration layer exposing an API that application elements may use to request desired performance from the transport layer.

- **Security: How can the Software-Defined Network be protected from malicious attack?**

Potential security vulnerabilities exist across the SDN platform. At the controller-application level, questions have been raised around authentication and authorization

mechanisms to enable multiple organizations to access network resources while providing the appropriate protection of these resources. Not all applications require the same network privileges and a security model must be put in place to isolate applications and support network protection [11].

The controllers are a particularly attractive target for attack in the SDN architecture open to unauthorized access and exploitation. Furthermore, in the absence of a robust, secure controller platform, it is possible for an attacker to masquerade as a controller and carry out malicious activities. Considerably greater damage could be done by such an attack on an SDN controller.

A security technology such as Transport Layer Security (TLS) with mutual authentication between the controllers and their switches can mitigate these threats. Even at a lower level, individual network nodes, hosts or users could be targeted undermining the desired network performance. The security of SDN will only be as good as the defined security policy. Implementation of existing authentication and authorization mechanisms can resolve some aspects of the security challenge. Meanwhile, threat detection and protection techniques will continue to evolve.

Interoperability: How can SDN solutions be integrated into existing networks?

It would be straightforward to deploy a completely new infrastructure based on SDN technology. For this, all elements and devices in the network would be SDN-enabled. However, there exists a vast, installed-base of networks supporting vital systems and businesses today. To simply "swap-out" these networks for new infrastructure is not going to be possible and is only well suited for closed environments such as data centers and campus networks [11].

The transition to SDN therefore requires simultaneous support of SDN and legacy equipment. The IETF Path Computation Element (PCE) could help in gradual or partial migration to SDN. With PCE, the path computation component of the network is moved from the networking node to a centralized role while traditional network nodes not using PCE continue to use their existing path computation function. A specific protocol (PCEP) enables communication between the network elements. However, PCE does not provide complete SDN. The centralized SDN controller supports complete path computation for the flow across multiple network nodes.

Further development is required to achieve a hybrid SDN infrastructure in which traditional, SDN-enabled and hybrid network nodes can operate in harmony. Such interoperability requires the support of an appropriate protocol which both introduces the requirements for SDN communication interfaces and provides backward compatibility with existing IP routing and MPLS control plane technologies. Such a solution would reduce the cost, risk and disruption for enterprise and carrier networks transitioning to SDN. Introducing a new protocol requires consideration of standardization and where this standardization will be of most benefit. SDN is a relatively new technology. So, as would be expected, not all customer requirements are yet in place. These will be resolved, but setting out the current barriers that exist to adoption is important.

9. Barriers to SDN Adoption

Networking analysts have identified several barriers that hamper adoption of SDN [11]:

• **Lack of standards for full-device control** - Trying to resolve new software to existing devices has meant a lack of clarity over how generalized network service creation should be. Should an SDN offering cover device setup or simply the networking operations? As it is early in SDN adoption, this question will be answered by the market in time. Open Flow was one initiative intended to answer this issue, but while it provides the southbound API to manage the hardware layer, no standard exists for the northbound API to the application layer.

• **A lack of service control software -** Service control software is the technology that handles the build of routes and traffic control with an SDN network. Every SDN implementation requires service-control software to create the virtual networks. SDN users rely on network equipment manufacturers to provide the required service control software.

• **Multivendor network control -** One of the obvious value propositions of hybrid SDN is its ability to overlay across different infrastructure. Having service control software that can exercise control over heterogeneous offerings is critical to the success of hybrid SDN.

• **Managing control traffic-** As more and more control traffic is created, the metadata around control becomes larger than the actual application traffic itself. A more hierarchical approach towards traffic control and higher levels of distributed process and control will help with this issue.

• **Boundary functions are needed-** A problem with discrete as opposed to hybrid SDN is the lack of clarity as to how the individual SDN deployments will interact on the boundary between the SDN and the existing architecture. As hybrid SDN gains traction, these perimeter issues will be address more rapidly [11].

REFERENCES

[1] "Software Defined Network SDN". Scribd.com.

[2] "SDN-architecture-overview-transparent" by Open Networking Foundation (ONF) - SDN Architecture Overview (PDF), Version 1.0, December 12, 2013.Licensed under CC BY-SA 3.0 via Wikimedia Commons <http://commons.wikimedia.org/wiki/File:SDN-architecture-overview-transparent.png#/media/File:SDN-architecture-overview-transparent.png>

[3] Software-Defined Networking: The New Norm for Networks". White paper. Open Networking Foundation. April 13, 2012.

[4] E.Al-Shaer and S. Al-Haj, "Flow Checker: Configuration analysis and verification of federated OpenFlow infrastructures," 2010.

[5] Wikipedia, Google

[6]Juniper Networks <http://forums.juniper.net/t5/The-New-Network/Applications-of-Software-Defined-Networking/ba-p/170626>

[7] W.Braun and M.Menth, "Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices,"

[8] T.Feng, J.Bi, K.Wang, "Joint Allocation and Scheduling of Network Resource for Multiple Control Applications in SDN,"

[9] S. Velrajan, "Application Aware Routing in Software Defined Networks," Aricent Networks

[10] S. Perin and S.Hubbard, "Practical Implementation of SDN & NFV in the WAN," HP, Intel and WindRiver, October 2013.

[11] S. Sezer, S. Scott-Hayward, P. K. Chouhan, "Are we ready for SDN? - Implementation Challenges for Software-Defined Networks,"

[12]C.Wilson, H.Ballani, T.Karagiannis, "Demand-Aware Flow Allocation in Data Center Networks," ACM SIGCOMM Computer Communication Review, 2011.

[13] http://www.cisco.com/web/europe/ciscoconnect2013/pdf/DC_3_SDN.pdf

[14] https: //www.sdxcentral.com/use%20cases/

[15] http://packetlife.net/blog/2013/may/2/what-hell-sdn/

[16]<GigaOmResearch>http://research.gigaom.com/report/sdn-meets-the-real-world-implementation-benefits-and-challenges