A Technology Case Study on Integrating Open Stack with SDN for Internet Connectivity using BGP

A Master's Project

Presented to

The Department of Telecommunications

In Partial Fulfillment

of the Requirements for the

Master of Science Degree

State University of New York

Polytechnic Institute

By

Raja Bhushan Rao Gonuguntla

Dec 2016

SUNY POLY

DEPARTMENT OF
COMPUTER AND INFORMATION SCIENCES

Approved and recommended for acceptance as a project in partial fulfillment of the requirements for the degree of Master of Science in Telecommunications

Jan. 19, 2017

DATE

Project Advisor Dr. Larry J. Hash

A Technology Case Study on Integrating Open Stack with SDN For Internet Connectivity using BGP

Declaration

I declare that this project is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

____

Raja Bhushan Rao Gonuguntla

01-19-2017

Executive Summary

There were many developments in Internet usage, which resulted in significant increase in Internet routing. With existing networking infrastructure, it is difficult to meet these requirements and causing more cost and less performance. Since network devices are hardware modules, processing them requires more power and more memory. However, if network protocols are developed using software modules, flexibility can be achieved in various programming applications and reduces dependency on hardware. The concept of using networking protocols as a software module can be explained using "***Software Defined Networking (SDN)***." With SDN, existing infrastructure can be integrated with various applications and centralized control protocols can be developed. One of the key components of SDN is integrating with Cloud Computing, where many applications can be built, which can be used for on-demand services. Integrating cloud computing with SDN will create dynamic networks and reduces infrastructure costs.

In this paper, a case was considered for providing better internet connectivity by building public & private networks using Open source cloud technology (OpenStack) and existing distribution environments. For connectivity, BGP was used as routing protocol as it is known to be well- suited for such environments. Both public and private networks were integrated with SDN for centralized control. OpenStack was used to build various network topologies using different plugins through SDN controller. This method allowed to develop SDN controller with global view of OpenStack networks. The same controller was connected to distributed layers using Open Flow protocol. Since, both OpenStack and distributed networks were attached to SDN controller, centralized control of network protocols could be achieved. This model of centralized networks could be very useful in reducing costs and improving network efficiency, especially in large scale deployments.

Project is dedicated to

Family &

Friends

Acknowledgement

Firstly, I would like to thank Dr. Larry Hash for all his help and advice in my studies and research. His patience and approachability has proved immensely helpful in every aspect of my graduate work.

I would also like to extend my thanks to the present and past members of the professors in Telecommunications especially Dr. John Marsh, Dr. Joshua S White and Prof Ali Ibrahim for all the help and advice in designing the course work.

Finally, I would like to thank my family; my parents and my brother, uncle and friends for all their support during my education here at Utica. I am especially grateful for their understanding and love.

Table of Contents

Table of Figures

1. Introduction

The Internet became necessity in these days, and there is an exponential increase in usage of the Internet applications like VOIP phones, Home automation, Web services and so on for day-to-day activities. Considering the usage of the Internet, current networking infrastructure (which is the backbone) is not sufficient to meet future needs. Moreover, with the innovation of the cloud computing where services can be provided based on demand, using the public or private clouds. Through cloud technologies, Network as a Service (NaaS) can be offered on demand through public or private cloud, which reduces infrastructure costs and hardware dependency.

In addition to the cloud technology, current network infrastructure needs more innovation for flexible control over routing, switching, security and load balancing. Centralized control of networking devices allows better functioning and reduces cost to provide services like internet. Without altering any functions of networking, centralized control can be achieved by using Software Defined Networking (SDN). With SDN, same networking topologies can be utilized for better performance and create a flexible environment.

By integrating the cloud technology with SDN, more robust networks can be built. With NaaS as a service; topologies like physical infrastructure can be built. Integration of NaaS with SDN provides more flexible network that can make use of existing infrastructure for more networks without any physical devices.

With BGP routing protocol that is being widely used by service providers for internet connectivity. There are two types of circuits for internet connectivity, which are Provider Edge (PE), and Customer Edge (CE). Using NaaS; one can create private networks that are connected to public networks using internet. However, NaaS is supported by OpenStack, Amazon Web Services (AWS) or Microsoft Azure. In this paper, use case of BGP routing protocol with Open Stack (Open Source) as interface and integrated to SDN, to check internet connectivity between customer and provider edge networks.

2. Thesis Statement

"Technology case study on Integrating Open Stack with SDN and existing networking infrastructure to provide internet connectivity to customers using BGP routing protocol."

Understandings from Thesis:

1. Understand current challenges in Networking.
2. How does Software Defined Networking (SDN) address those challenges.
3. The importance of cloud computing.
4. Widely used routing protocol BGP for providing the internet connectivity.
5. Integrating Open Source cloud computing Open Stack with SDN using BGP.

3. Audience View

Internet is one of the most important innovations in last few decades, and it has significantly impacted the lifestyle of people. Internet is a backbone network and is supported by various service providers. Protocols involved in providing backbone network services are largely dependent on the nature of network, i.e., private or public. While cost and usage efficiency are mainly determined by the features of these network protocols, and they are also influenced by challenges in the current distribution environment.

To understand this paper, the reader is expected to have a basic knowledge of Border Gateway Protocol (BGP) and Software Defined Networking (SDN). Familiarity with BGP routing protocols and why service providers prefer it to provide Internet connectivity can further assist in understanding this material. Additionally, readers should have some understanding about SDN and how it is used for centralized control of network protocols. By centralizing protocols, one can provide customization by integrating them with various applications. One such application that can be integrated with SDN is cloud computing, through which on demand services can be provided. Upon reading this paper, readers should be able to understand how SDN can be integrated with cloud computing technology using BGP routing protocol. This integration is current trend in networking research and is going to make a significant impact in the world of network engineering.

4. Background

With drastic increase in Internet usage, it has become difficult for service providers and end users to maintain hardware and software resources in the Network. To establish communication over Internet, most widely used routing protocol is Border Gateway Protocol (BGP). As this protocol consumes huge processing power and memory space, it is difficult to maintain the Networks' efficiency with current hardware capabilities. With existing network infrastructure, protocols work either in control or data plane where data plane forwards the packets and control plane decides the path followed by the packets. As this type of design requires more amount of resources, there is a need to separate data and control planes which can be done with the help of "*Software Defined Networking (SDN)*" [2] [11] [16].

The SDN architecture has three layers namely Data Layer, Control Layer, and Infrastructure Layer. Data Layer is a first layer where all the hardware or virtual devices reside. Second layer has controllers that help in managing network devices and decide the flow of packets between switches and routers. Last layer is an Infrastructure layer, where web services can be integrated with HTTP through "Application Program Interface (API)." Cloud technologies that are used in this architecture enable the separation of data and control planes, and helps in achieving centralized control of the network.

In general, cloud technology is considered as "Infrastructure as a Service (IaaS)" [15], which can provide various services like networking, computing, and storage on demand. Small and medium scale companies can make use of cloud technology as "Network as a Service (NaaS)" to reduce infrastructure cost. Some of the cloud computing technologies that support NaaS are Amazon Web Services (AWS), Microsoft Azure (MSAzure) and Open Stack. Amongst these technologies, OpenStack is open source and does not have licensing issues. There are various services that OpenStack supports like Storage, Network, and Platform etc. For this project, OpenStack has been used to provide Network as a Service (NaaS) and it has separate application within OpenStack for networking which is Neutron [8]. Neutron supports various routing and switching technologies and has plugins for each of them. It supports both distributed and centralized environments where virtual devices can be used as well. Within centralized environment, network protocols in open stack can be customized further. For this customization, open stack needs to be integrated with Software Defined Networking (SDN). With centralized control of Network, all the devices can be managed in more robust and efficient way.

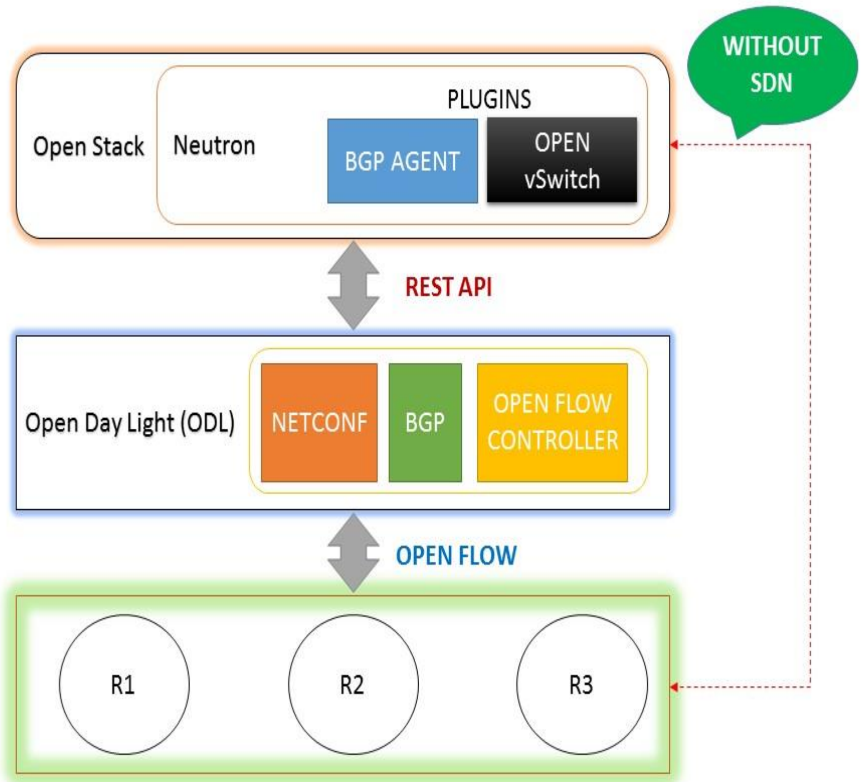

Figure 1: SDN Architecture with OpenStack Neutron API using BGP

As per Figure 1, Layer 1 has all the physical devices which will replace the current distributed network. OpenFlow is a protocol that is used to integrate all these devices in Layer 1 along with SDN controller in Layer 2 [1]. This protocol has flow tables that are forwarded to controllers and those tables help in differentiating control and data planes. Usually, OpenFlow is preferred to other protocols such as OpenFlex [17] as it is open source.

Layer 2 of SDN architecture acts as a brain for all the devices in Network. Controllers in this layer maintain the routing and switching tables of all the hardware in Layer 1. These controllers are Software modules, which are available from various vendors in the market. Currently, most widely used SDN controllers are Open Day Light (ODL) [8] [10] and Cisco Application Centric

Infrastructure (ACI) [9]. ODL is an open source controller which supports various technologies related to routing, Quality of Service (QoS), switching etc. Cisco ACI was not considered because it was Cisco proprietary and requires Nexus 9000 series switches [9]. On the other hand, ODL can be used in small and medium scale networks because it doesn’t require costly hardware such as Nexus.

With networking protocols customized at Layer 2 using SDN architecture, there is a possibility to manage all network devices from web. To achieve this, the application layer should be used as Layer 3 with OpenStack as NaaS protocol. Integration of application layer is achieved by using “Representational State Transfer (REST) [6] [18]” API, which will be in XML format through HTTP.

By integrating OpenStack Neutron [14] with SDN architecture discussed above, there is a possibility to maintain network in an efficient way. In Figure 1, a use case has been considered where BGP routing protocol operates along with OpenStack Neutron when integrated with SDN. With this integration, more networks can be created for Internet connectivity over BGP routing protocol. In addition to this, routing updates can be customized with the help of OpenStack.

OpenStack is a cloud computing technology, which is integrated to SDN controllers through REST API. When it is integrated with SDN, there are a few plugins that are used. Neutron on the other hand supports various routing or switching technologies such as creating networking topologies. There are various plugins in OpenStack which are used to maintain network protocols that are used for communicating with other layers. Open vSwitch and BGP Agent are the plugins, which were used in this use case. Here, Open vSwitch was used to create virtual networks that are integrated to SDN controller or physical devices. Whereas, BGP agent was used in sending BGP updates to the controller such that SDN controller has full routing updates from the OpenStack.

Centralization of control for OpenStack environment can be achieved by using SDN controller through REST API [18]. Where REST API is a protocol that is used by SDN to integrate allocation layer with the control layer. In the control layer, there are various useful modules. With ODL, there can be multiple projects that can be deployed and each project can evaluate the efficiency of network protocols. In this paper, some of the utilized projects of ODL are "Network Configuration Protocol (NETCONF) [12]", BGP and OpenFlow Controller. With NETCONF, logical connection of SDN controller with both the application layer and infrastructure layer of SDN architecture. In addition, NETCONF also converts the messages into XML format and this information is useful in transferring the data to application layer through REST API. Besides NETCONF, the BGP project module is used to get routing updates from both the application and the infrastructure layer of SDN architecture to maintain a global routing table. When it comes to centralized view for current existing distributed environment, OpenFlow Controller is useful.

The main purpose of OpenFlow controller is to maintain centralized database for all devices that are connected in the infrastructure layer of SDN architecture. OpenFlow controllers maintains the database of MAC address and IP address of all devices and update them in OpenFlow table. OpenFlow uses a technique called as ARP proxy [7] and through which MAC addresses of the devices are added into OpenFlow table. When it comes to BGP routing updates, OpenFlow table sends information to BGP routing table about the processing of routes using “destination MAC address [1] [7]”. Mapping of the destination IP address with the existing OpenFlow table calculates the route for the next-hop. This technique helps in offloading the processing on BGP route table and achieving centralized control.

5. Literature Reviews

5.1 BTSDN: BGP-Based Transition for the Existing Networks to SDN

About the paper: The authors of the paper "**BTSDN: BGP- Based Transition for the Existing Networks to SDN**" are "***Lin, P., Bi, J., & Hu, H***". This paper was published in the Year 2015, Pages, 1829-1843 of the journal *Wireless Personal Communications.*

Abstract: With current network architecture, which is a very close model and there is very less scope for the innovation. However, there is a technology called Software Defined Network (SDN) which decouples the data plane and the control plane allowing centralized control of the networks. With the wide acceptance of the academic and the researchers where Open Flow as the protocol used for SDN. In this paper, Open Flow as SDN protocol applied over the existing Border Gateway Protocol (BGP) called as a "*BGP transition over SDN (BTSDN)* [7]." BTSDN utilizes all the characteristics of Open Flow with BGP connectivity to internet same like legacy BGP router with some experimental results.

Introduction: With closed network architecture, only limited scope is available for innovations and full utilizition is not possible with existing protocols. SDN separates the data and the control plane allowing network protocols to expand their features. Currently, SDN protocol is used by the academic and the researchers with Open Flow as the main protocol. Moreover, SDN uses centralized approach through which all the networking devices are controlled which is control plane and Open Flow from the data plane. The controller which is used for the control plane can be "Flow Visor which achieves the network virtualization [7]." Since the SDN protocol is not fully implemented and here the researchers proposed SDN implementation over BGP routing protocol using the infrastructure. With BGP used as the internet routing protocol, one can achieve the same features using BTSDN.

Methodology: In this paper, the researchers outlined the importance of innovations in the network architecture with SDN and implementation over BGP routing protocol which is the backbone of the internet. With SDN technology, Open Flow which is the protocol of SDN and considered a use case for experimental analysis over BTSDN.

Background: With the exponential usage of the internet and with the advent requirement there should be a robust network architecture which is provided by SDN. Currently, SDN technology is widely used by the researchers and academic usage with Open Flow protocol integration between control and data layer. Open Flow SDN protocol is used over the data plane for sending information to centralized controller. Once data plane information is forwarded to controller which decides on the request done by the FloodLight controller and here in this paper, the researchers used software Quagga as a controller. Later, researchers considered use case on BGP which is widely utilized for the internet connectivity and explained SDN transition over BGP (BTSDN).

The main functions of the BGP routing protocol which is used for inter-domain (EBGP) and intra-domain (IBGP) are same to BTSDN while providing the internet. BTSDN packet delivery to other routers when it comes to intra-domain or inter-domain can be done by Open Flow by updating in separate tables which are routing (for next-hop) and ARP (Address Resolution Protocol) table.

With Quagga as a network controller which will send routing updates to other border routers based on the information from tables. When it is inter or intra-domain routing, BTSDN will update its routing tables based on "*ARP proxy module and destination MAC addresses rewriting* [7]." Proxy module is mainly used by the controller for frequent updates in the ARP table and then Open Flow switch inserts the destination MAC address in flow entry table to identify the BGP route. Used the same model for BTSDN in evaluating for routing changes dynamically which are used widely for the internet without altering the IP header.

Relevance: Researchers focused on SDN over BGP and evaluated the use of BTSDN in providing the internet. However, introduction of Cloud Technology created more options extending the project which is integrated with Open Stack (Open Source Cloud Technology) using application layer of SDN. One can integrate SDN with "*Application Program Interface (API)*" which are deployed over the web and there comes Open Stack which can be used as API. With Open Stack creating networking topologies with BGP as a dynamic routing protocol in evaluating for the internet connectivity.

Critique: The researchers in this paper, highlighted the importance of new technologies for creating networking topologies as there is less scope for current infrastructure to expand. SDN is one solution that can be used by decoupling data and control plane. The use case that the researchers have taken is quite necessary as BGP is mostly used for the internet routing protocol which they have evaluated by taking small topologies. The topology that is created was clearly explained and will act as a base for extending for other routing protocols or even cloud technologies.

5.2 Relationship-Oriented Software Defined AS-Level Fast Rerouting for Multiple Link Failures

About the paper: The authors of the paper "**Relationship-Oriented Software Defined AS-Level Fast Rerouting for Multiple Link Failures**" are "***Li, C., Li, X., Li, K., Huang, J., Feng, Z., Chen, S., Shi, Y***". This paper was published in the Year 2015, Pages 1-15 of the journal *Mathematical Problems in Engineering.*

Abstract: The Internet routing demand is increasing with large scale deployments and in parallel addressing the importance of link failures for those deployments. Border Gateway Protocol (BGP) is one of the critical routing protocols used by many different service providers and customers as the convergence time for link failures is less. With BGP, working principle is on Autonomous System (AS) relationships and there should be a way for fast rerouting of multiple link failures on AS path and fast rerouting. Moreover, with the emergence of SDN, fast rerouting of routing tables becomes more important as the controllers will automate and dynamically update the control plane. In this paper, the researchers have proposed fast rerouting for multiple link failures over AS relationships and implemented BGP routing protocol using Open Flow southbound API of SDN. The results show that the proposed model can be utilized for better network performance.

Introduction: With an increase in the number of applications using the Internet, demand for routing on the Internet has also increased. Due to this, number of link failures are also increasing. The current distributed environment doesn't have enough scope for customizing the usage of routing protocols. This can be done by using SDN by differentiating control and data plane. The other advantage of using SDN is centralized control over different protocols. BGP is the most widely deployed protocol for provisioning the Internet services. In this paper, researchers have proposed a new model for handling multiple link failures based on the AS relationships of BGP. The proposed model "Fast rerouting for multiple link failures by considering AS relationships (SDFRR-ASR) [6]" is evaluated based on the SDN architecture using OpenFlow protocol that differentiates both the control and the data layer. Because of the ability to optimize the control of protocols which are centralized, one can identify the routing paths and path failure links. The failure links that are identified based on the BGP relationships with neighboring routers have less convergence time compared to that of the existing distributed environment.

Methodology: In this paper, Researchers have evaluated the importance of fast rerouting of routing algorithms for better convergence as this is one of the critical parameters for network performance. The authors of this paper have proposed a unique approach to address multiple link failures with the help of SDN. They have used the BGP routing protocol to evaluate the multiple link failures as this paper mainly focuses on this aspect.

Background: Because of the rapid development of the network infrastructure the world over, many networking devices are being routed over the Internet. BGP is the routing protocol that is used by different service providers to provide network services to people all over the world, and even this protocol faces the technical challenges like convergence and link failures. Moreover, recent developments in the field of Networking such as the introduction of SDN created more importance for fast rerouting of routing algorithms. As SDN separates the control plane and the data plane, some applications like routing have centralized controller. Under this scenario, there can be single or multiple link failures which might create radical changes in the network performance. In this paper, Researchers have introduced a new concept called SDFRR-ASR to address multiple link failures over AS-relationships [6].

SDN separates the data plane and the control plane to provide centralized control to the networking protocols. By using this model, both centralized and distributed control of protocols can be achieved. Distributed control is achieved over BGP; this is where information about the routing table exists. On the other hand, centralized control collects all the required information from the data layer through OpenFlow protocol [6]. OpenFlow is an SDN protocol that integrates both the control and data layer.

The entire process of fast rerouting can be made possible with the SDN application layer where it is possible to customize updates from the control layer in calculating link-failures. The process of link failure happens in three steps, which are "Failure scenario recognition (FSR), protection path computation (PPC) and protection tunnel deactivation (PTD)" [6]. In the first step, one can identify failure scenarios based on BGP relationship with nearby routes. In the second step, PPC calculates alternative paths; this is done in the control layer of SDN based on the updates from OpenFlow, Routing tables, and AS-Relationships. Most of the routing updates need to be considered from the security perspective, where there would be need to consider the route policy filtering when calculating link failures. After this, optimal paths are defined for failed links. From these two steps, required information is gathered and this gathered information needs to be shared with devices in data layer. All these failed links are updated in the OpenFlow table. In the final step, OpenFlow table will be updated again by removing the entries of all the failed links, if they are no longer considered as failed links.

Relevance: In this research, primary focus is on "Integrating Open Stack with SDN for the Internet Connectivity using BGP". In this paper, Researchers have discussed fast rerouting of algorithms when SDN is used, and since this project is also dealing with BGP over SDN, this paper would be relevant to this research. Since the BGP routing protocol works on AS relationships, the fast rerouting algorithm proposed by the authors SDFRR-ASR was found to be useful in understanding the working of BGP over SDN in an in-depth manner.

Critique: In this paper, the authors' primary focus is on the mathematical models of AS relationship protocols used in SDN, in this case the protocol used is BGP. However, there could have been a bit more detailed explanation of SDN in general apart from explaining its mathematical models, as this could give the readers a better idea about the functioning of SDN.

5.3 Design Patterns and Extensibility of REST API for Networking Applications

About the paper: The authors of the paper "**Design Patterns and Extensibility of REST API for Networking Applications**" are "***Li, L., Chou, W., Zhou, W., & Luo, M***.". This paper was published in the Year 2016, Pages, 154-167 of the journal *IEEE Transactions on Network and Service Management.*

Abstract: By using Software Defined Networking (SDN), customizing network protocols to create applications and giving better performance for distributed networks can be done. Currently, most of the applications are deployed in the form of web services over the Internet. The integration of these web services with SDN can be done by using "Representation State Transfer (REST) [19]" API through HTTP requests. However, biggest challenge for this model is frequent changes and updates happening to applications. To accommodate these changes, REST API needs to understand these structural changes without disturbing applications. In this paper, Authors have introduced differential cache mechanism for HTTP navigation over REST API. The proposed model is implemented on the SDN controllers, thereby reducing 66% overhead on the REST API.

Introduction: With logical separation of control and data plane, centralized control can be achieved over the networks using SDN. Moreover, using SDN it is possible to customize or build various applications which can be integrated using REST API. By using REST API, one can integrate the Cloud Computing services with SDN. With frequent changes happening to services, it becomes challenging for the REST API to communicate with application layer of SDN through HTTP navigation.

In this paper, Researchers have proposed a new model for extension of REST API. This model is useful for large cloud deployments without disturbing services. The proposed REST API has a REST Chart, which defines HTTP navigation through XML language. To reduce the overhead of HTTP navigation, Researchers have used a differential cache mechanism and performed practical implementation of SOX SDN controller.

Methodology: In this paper, Researchers have discussed the framework of REST API protocol. This protocol is used for integrating the application and control layer of the SDN architecture. While explaining the framework, they have proposed a novel idea of a differential cache mechanism and demonstrated it by creating a small test bench. The result of their unique approach showed that their proposed model was 66% more efficient than the standard REST API.

Background: REST API is a framework between control and application layer of SDN architecture. Web access for SDN controllers can be provided by integrating REST API, which is driven using HTTP request. Moreover, multiple applications can be created with this integration, like using the cloud technology services such as OpenStack Neutron [5]. In this paper, Researchers have used REST API for integrating OpenStack Neutron with the SDN controller. The REST API uses HTTP navigation to perform this integration. While addressing the REST API architecture, Researchers have highlighted that it is capable of handling multiple requests from the SDN controller simultaneously.

The key challenges for REST API through HTTP driven navigation are, applications will change their functionality in real-time and frequent updates to these applications. Despite these difficulties, REST API needs to provide continued connection to its clients. HTTP driven navigation will identify connections for all devices and then convert them into HTTP links. The description of these links is derived from the REST chart. The REST Chart will have information regarding applications such as their hyperlinks and user credentials.

Moreover, the REST API has various design patterns which are used based on the application. Some of the design patterns which are most widely used are "Tree, Backtracking and Search patterns [5]". In the array, if there are multiple connections to clients, Tree pattern is used to connect to them. Backtracking is used to maintain the information regarding each hyperlink. This is used in case of broken links to the SDN controller. Search pattern is used when there are multiple links from the REST API to the client. This helps in finding the desired patterns in the database with the help of a search criterion.

In this paper, Researchers have used Backtracking pattern, because it creates a cache mechanism. It has two advantages; the links can be backtracked when they are broken and optimizing the traffic. When a connection for the same link exists, the REST API pulls information from cache which reduces overhead on the SDN controllers.

Relevance: In this paper, Researchers have explained about OpenStack Neutron and its interaction with the SDN controller. The researchers have used Flood Light as an SDN controller whereas Open Day Light (ODL) could also have been used because of its current popularity.

Critique: This article is very useful in understanding the basics of the REST API architecture. The researchers have performed detailed analysis in defining the framework and highlighting the importance of automation of applications which are hypertext driven navigation. In the end, Researchers have evaluated performance of REST API using OpenStack Neutron service with Flood Light SDN controller. The performance of REST API could also be evaluated using other SDN controllers such as ODL, Quagga etc.

6. Control Plane vs. Data Plane

Internet Protocol (IP) is backbone for network applications where there is flow of IP datagrams with either connection or connectionless oriented communication. The network traffic is classified into control and data plane. Most of the existing infrastructure is in distributed environment where both control and data plane exist on the same device. Due to these distributed environments, the cost of infrastructure increases and the network protocols are not efficiently used as well [3]. Whereas, by separating the data and control planes, more applications can be built with the help of Software Defined Networking (SDN).

Control Plane will define the route paths for IP packets based on source and destination addresses. However, with distributed environment, these packets are forwarded based on routing and MAC address tables in routers and switches. With this kind of model, there is a problem of consuming more processing power and more memory, which leads to degrading the performance of network applications [3]. To overcome this, there is a need to look for an alternative way, where all the data packets in a network are managed centrally. For example, if there are ten routers in a network then it is easy to have a global routing table for routers at one place, instead of having routing table on each router. With global routing table, requirements for memory and processing power are reduced. One more advantage of having this type of model is, it enables the users to have centralized control of all the data packets. With centralized control of routes, there is scope to support more network applications. On the other side, Data or Forward plane is responsible for forwarding the data packets [1] [3]. All the data packets are originated either from a physical or virtual network device.

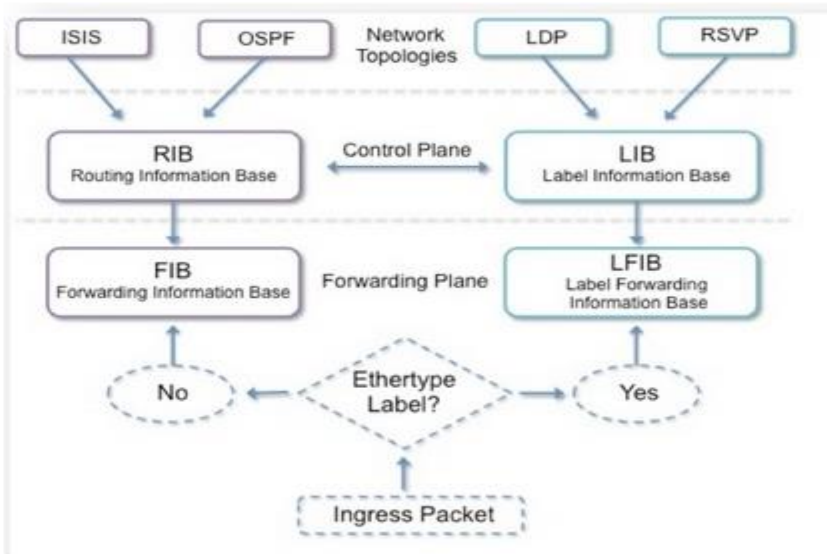

Figure 2: Control Plane Vs Data/Forwarding Plane

Source: http://networkstatic.net/the-control-plane-data-plane-and-forwarding-plane-in-networks

From the Figure 2, data and control planes are differentiated based on Layer 2 and Layer 3, i.e., switching and routing. There are two tables that are “Routing Information Base (RIB) and Label Information Base (LIB) [3]” which are related to routers and switches respectively. From the example, BGP routing table consists of the AS-path attributes of BGP. These updates along with the source and destination MAC address are present. By differentiating both the planes, a global routing table is present which reduces the processing power for each router.

7. Open Flow

OpenFlow is the standard protocol that is defined by "*Open Networking Foundation (ONF) [2]*" group, to communicate interfaces between the control and the infrastructure layer of SDN architecture. The logical connection between the two interfaces is through "*Transport Layer Security (TLS) [1]*" over port 6633. Where TLS is a protocol that provides security to data when communicating between two or more applications. As discussed earlier in the background section, OpenFlow protocol works on the devices with OpenFlow compatibility. These devices with OpenFlow compatibility can update or modify the Flow table with IP Address and MAC address through OpenFlow messages.

7.1 Open Flow messages in Switches

Open Flow messages are sent from either controller or flow table. The messages are classified as three subtypes viz., Controller-to-Switch, Symmetric and Asynchronous. As the name indicates, the controller-to-switch message is where controller sends the packet requesting for features supported on the switch. The second type, Symmetric message is where three-way handshake between the switch and controllers will happen. In the final type, a message will be initiated by the switch and sent to the controller asynchronously. With this message, switch will inform the controller about arrival of packet, port-status, and error, if any. The open flow table is updated to the switch.

7.2 BGP Route updates through Open Flow messages

With the separation of the data and the control plane, SDN controller manages network protocols. In this paper, SDN controller used is ODL which is an OpenFlow type controller project. Controller receives updates from Flow table, which operates over messages as discussed before. If there are multiple devices in the infrastructure layer, there can be sequence of messages which will update the flow table.

Once ODL receives messages from Flow table, BGP module in the controller pulls "Routing Information Base (RIB) [3]" from OpenFlow table leaving behind the MAC and IP Address. Through this all BGP updates of distributed environment achieves centralized control through Flow Table. Under this case, controller will act as boarder router. With new routing update, Flow table will use the concept of destination MAC re-writing. This can be explained when a router receives a packet, it will update header with destination MAC address. If destination MAC address matches with next-hop MAC address, it will accept packet else rejects it. This process will be repeated for all BGP routing updates. This solution reduces threshold on BGP controller as most of the route updates are filtered in flow table.

8. SDN Controller – Open Day Light

In the SDN architecture, controller is the main component that manages network protocols within the control layer. Any one between open source or proprietary controllers could be used in the architecture of SDN. Open source controllers can be integrated to any device without any dependence on vendors. Open Day Light (ODL) [10] controller is suitable for small or medium environments. ODL has several versions that are being maintained by Linux Foundation and current version is Boron.

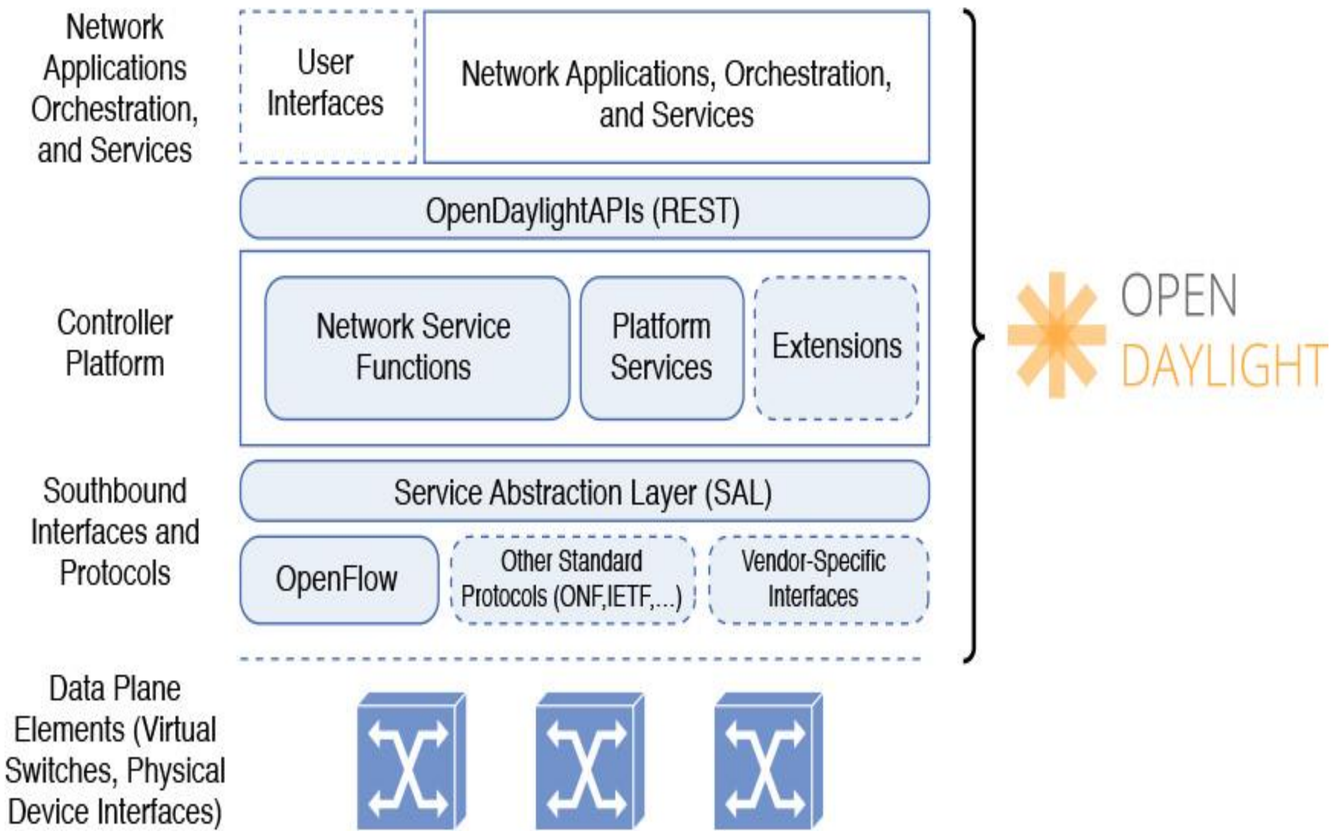

Figure 3: ODL common framework for SDN

Source: https://www.safaribooksonline.com/library/view/sdn-and-nfv/9780134307398/ch24.html

From the Figure 3, one can see that controller platform lies between infrastructure layer and application layer. ODL supports multiple "Network Service Functions", among which most of the network protocols that are used can be configured. Thus, it provides centralized control for the network. Besides network functions, ODL also supports the integration of applications within cloud computing or any other application in a distributed environment, using API calls. As discussed in the Background section, ODL can be integrated with cloud computing technology using OpenStack through REST API. Here, REST API which is driven through HTTP navigation serves as a logical connection between ODL and OpenStack.

Besides integrating with cloud computing technologies, ODL also supports various other network functions. After this, it will be determined, if BGP routing protocol is necessary for integrating application layer and infrastructure layer. In this paper, two plug-ins have been considered for performing the integration between application layer and infrastructure layer, they are OpenFlow [1] and NETCONF [12] respectively.

8.1 ODL with BGP

With ODL as a framework, BGP routing protocol can be configured to get the routing updates from both the infrastructure layer and the application layer. As discussed in the background section, BGP routing protocol will behave as expected after separation from the control and the data plane. BGP module maintains separate RIB for all the BGP routes. When offloading some of the functionalities of BGP, OpenFlow table sends the routing updates to the border router without intervening the BGP software module. With this flexibility, the applications can be integrated to customize the BGP protocol through NETCONF. NETCONF can also be used to send the routing updates from the application layer to the BGP RIB table.

9. Open Stack

Cloud computing is a technology which allows applications to be used on demand. With cloud computing, the customers could access various services such as "Computation as a Service (CaaS), Network as a Service (NaaS), Storage as a Service (STaaS) [11]" etc. on a pay-as-you-go basis. The Figure 4 shows the different service types offered by cloud computing. With NaaS, network services could be provided to the people that do not have access to these services, such as the Internet. NaaS could be provided using several cloud computing services which are open source or licensed. When providing service to small or medium business networks it is good to choose open source which is license free. OpenStack Neutron is an open source cloud computing software which provides service NaaS. It is programmed and maintained by Rackspace.

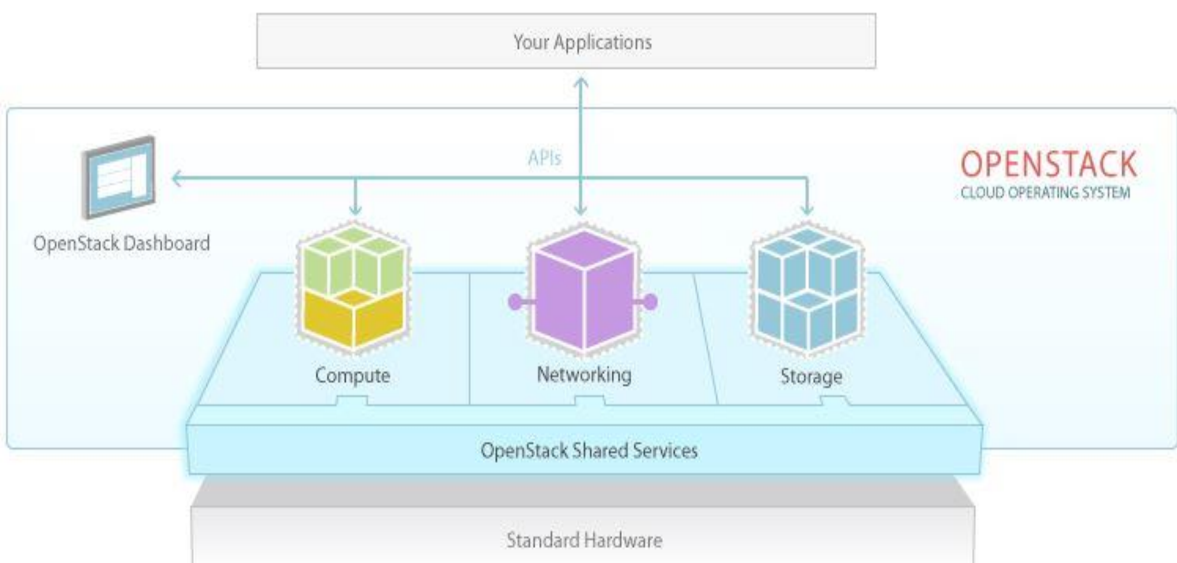

Figure 4: OpenStack Cloud Operating System
Source: https://www.openstack.org/software/

As discussed in the background section, with OpenStack Neutron, multiple network topologies can be created on the virtual machines. These networks are provided based on demand. This means, the customers only pay for the services they use and for the period they need them. Network topologies created in the OpenStack can be directly integrated with the existing distributed networks. Moreover, for better efficiency, these networks are integrated with the SDN controller to have centralized control.

9.1 OpenStack Integration with SDN

As discussed earlier, by integrating OpenStack with the SDN controller, more efficient networks can be created. The Figure 5 explains the high-level integration of OpenStack [16] with the ODL controller.

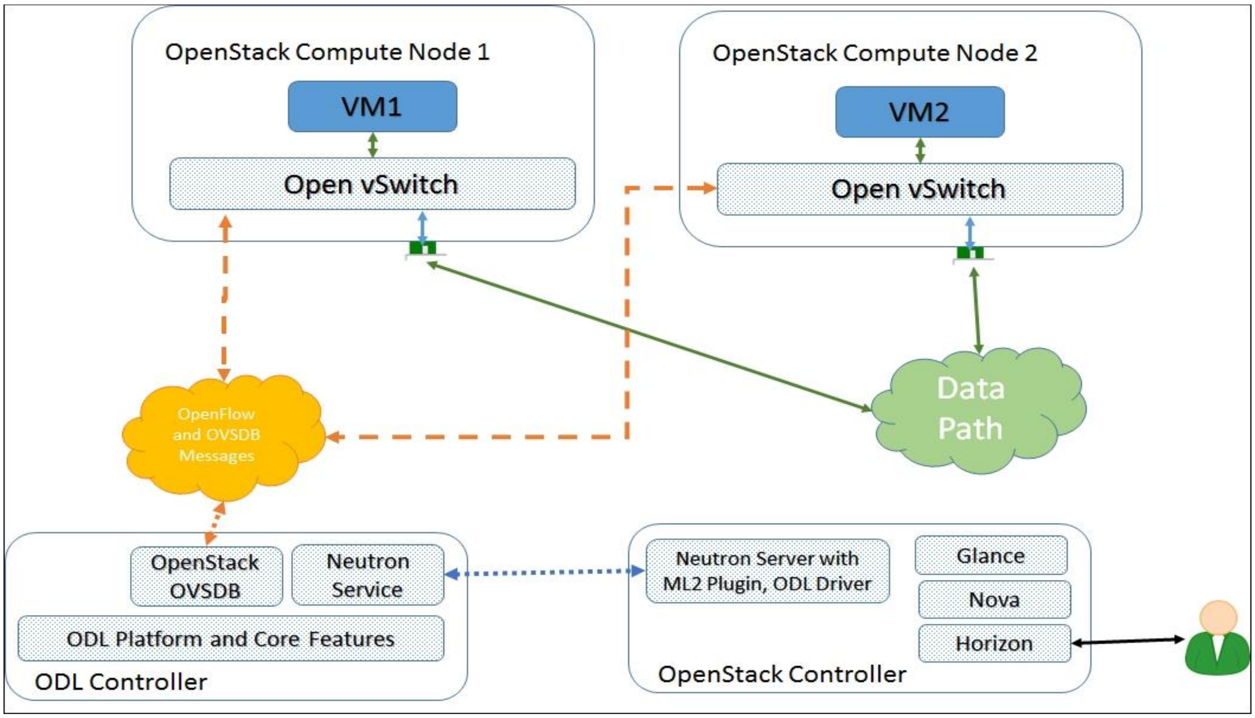

Figure 5: OpenStack Networking with Open Day Light (ODL)
Source: https://www.safaribooksonline.com/library/view/software-defined-networking-sdn/9781786465993/ch06.html

The detailed analysis of the Figure 5 shows the networks that are deployed over OpenStack on a Virtual Machine (VM). It also shows that the communication between the data packets and the control packets flows through Open vSwitch (OVS). All the control packets are received by the Open vSwitch Data Base (OVSDB) [16] plugin in the ODL controller. For Layer 2 operations like VLAN encapsulation, ML2 plugin inside the OpenStack controller helps the Neutron module to send the data packets internally. An L3 plugin is used for layer 3 communication using routing protocols like BGP, EIGRP, OSPF etc. From the background section, BGP service is used for L3 agent for all the BGP route updates from the Neutron to ODL controller. When deploying the BGP agents on the OpenStack Neutron, “Source, destination and Floating IP” are mandatory. The purpose of floating IP is for redundancy which is used when logical connection is failed for one connection.

10. Conclusion

The evolution of technology over the last few decades has resulted in the invention of important things, especially the Internet. To provide Internet services, routing updates should be distributed over various networks, hence maintaining them becomes a difficult task. The major drawbacks for this distributed environment are the cost of infrastructure and maintenance, as it consists of several randomly connected networks. To overcome these drawbacks, using Software Defined Networking (SDN) was considered to differentiate between control and data planes. This is because it can provide centralized control over existing distributed environments. With SDN, advantage of integrating with various cloud computing services can be achieved. Cloud computing provides customers with an option of renting additional services, if their demand increases. This could be attractive, since it follows a pay-as-you-go concept.

In this paper, a scenario of integrating SDN with OpenStack cloud computing using BGP routing protocol was explored. By considering an open source SDN controller like Open Day Light and an open source cloud technology like OpenStack, the cost of infrastructure could be substantially reduced in addition to obtaining centralized control. Multiple network topologies can be created on the fly using OpenStack based on customer requirements. This technology could be used more effectively while creating medium and small size networks at a relatively low cost.

11. Recommendations for further Study

SDN and OpenStack are both considerably new technologies, so more research may be conducted to improve security aspects of these technologies. Reducing the convergence time of BGP is another possible area of future research to consider, as this increases the reliability of the network. The researchers can also look for cross-vendor compatibility when cloud computing is integrated with SDN. It can be explained as, integration of OpenStack Neutron and AWS Virtual Tape library (VTL) for the same controller, where VTL is used for storage.

References

1. Azodolmolky, S. Introducing Open Flow. In *Software Defined Networking with OpenFlow*. Packt Publishing. Retrieved from https://www.safaribooksonline.com/library/view/software-defined-networking/9781849698726/ch04s02.html.
2. BGP dynamic routing Retrieved January 04, 2017, from http://docs.openstack.org/mitaka/networking-guide/adv-config-bgp-dynamic-routing.html
3. Cisco Application Centric Infrastructure. Retrieved January 04, 2017, from http://www.cisco.com/c/en/us/solutions/data-center-virtualization/application-centric-infrastructure/index.html
4. Doherty, J. (2016). The Open Day Light. In *SDN and NFV Simplified: A Visual Guide to Understanding Software Defined Networks and Network Function Virtualization*. Addison-Wesley Professional. Retrieved from https://www.safaribooksonline.com/library/view/sdn-and-nfv/9780134307398/ch24.html.
5. Li, C., Li, X., Li, K., Huang, J., Feng, Z., Chen, S., Shi, Y. (2015). Relationship-Oriented Software Defined AS-Level Fast Rerouting for Multiple Link Failures. *Mathematical Problems in Engineering*, *2015*, 1-15. doi:10.1155/2015/838340
6. Li, L., Chou, W., Zhou, W., & Luo, M. (2016). Design Patterns and Extensibility of REST API for Networking Applications. *IEEE Transactions on Network and Service Management, 13*(1), 154-167. doi:10.1109/tnsm.2016.2516946
7. Lin, P., Bi, J., & Hu, H. (2015). BTSDN: BGP-Based Transition for the Existing Networks to SDN. *Wireless Personal Communications*, *86*(4), 1829–1843. doi:10.1007/s11277-015-3145-0
8. OpenStack BGPVPN. Retrieved January 04, 2017, from http://docs.openstack.org/developer/networking-bgpvpn/
9. OpenStack Networking. Retrieved January 04, 2017, from http://docs.openstack.org/newton/networking-guide/intro-os-networking.html
10. RFC 6241 - Network Configuration Protocol (NETCONF). (2011, June). Retrieved from https://tools.ietf.org/html/rfc6241
11. SDN | Flowgrammable. Retrieved January 04, 2017, from http://flowgrammable.org/sdn/
12. Software -Defined Networking (SDN) Definition - Open Networking Foundation. Retrieved from https://www.opennetworking.org/sdn-resources/sdn-definition
13. Subramanian, S.,& Voruganti, S. (2016). Https://www.safaribooksonline.com/library/view/software-defined-networking-

sdn/9781786465993/ch06.html. In *Software-Defined Networking (SDN) with OpenStack*. Packt Publishing.

14. The Control Plane, Data Plane and Forwarding Plane in Networks. (2013, August 31). Retrieved January 04, 2017, from http://networkstatic.net/the-control-plane-data-plane-and-forwarding-plane-in-networks/
15. The OpenDaylight Platform | OpenDaylight. Retrieved January 04, 2017, from https://www.opendaylight.org/
16. Tidwell, R., Devesa, J., & Choudhary, V. OpenStack Open Source Cloud Computing Software BGP Routing. Retrieved January 04, 2017, from https://www.openstack.org/summit/tokyo-2015/videos/presentation/neutron-and-bgp-dynamic-routing
17. What is Cisco OpenFlow & OpFlex? Definition. Retrieved January 04, 2017, from https://www.sdxcentral.com/cisco/datacenter/definitions/cisco-openflow/
18. What's Software-Defined Networking (SDN)? Retrieved January 04, 2017, from https://www.sdxcentral.com/sdn/definitions/what-the-definition-of-software-defined-networking-sdn/
19. WP REST API v2 Documentation. Retrieved January 04, 2017, from http://v2.wp-api.org/