University of Colorado  
Interdisciplinary Telecommunications Program  
TLEN 5700

Creation of a VM Agent to Perform First Hop Forwarding  
Using Layer2/Layer3/Layer4 and VM Intelligence

Concept Definition Document (CDD)  
11/01/2017

1.0 PROJECT INFORMATION

Project Customers

Name: Prof. Jose Santos  
Address: University of Colorado Boulder  
Phone: +1 (303) 735-0102  
Email: Jose.Santos@colorado.edu

Team Members

Name: Abhishek Gandhi  
Phone: +1 (720) 400-2826  
Email: Abhishek.Gandhi@colorado.edu  
Name: Shiva Chetan Goudar Naganagoud  
Phone: +1 (720) 400-9246  
Email: Shgo2887@colorado.edu

Name: Madhavi Rao  
Phone: +1 (720) 401-1774  
Email: Madhavi.Rao@colorado.edu  
Name: Harshal Shah  
Phone: +1 (720) 934-2948  
Email: Harshal.Shah@colorado.edu

Name: Soham Shah  
Phone: +1 (720) 935-8602  
Email: Soham.d.shah@colorado.edu
2.0 Project Description (10 pts)

As the Internet progresses, the demand for deploying voice, video and data services continue to rise at an exponential rate. According to the statistics presented in Statista, United States is one of the largest online markets worldwide, with over 290 million Internet users at the end of 2016 [1]. To support video streaming, file sharing, peer-to-peer application traffic, the design need high computing capacity servers and more redundancy is required in the network. Providing multiple links or paths to a destination is one of the best ways to increase capacity, provide failovers, and hence give companies the ability to run critical network services with low latency and zero to minimum downtime. Equal cost multipath mechanism (ECMP) is a routing technique used in the networking domain to forward packets along multiple paths of equal cost. The solutions used today attempt to balance the load in the network by hashing the flows to the available shortest paths. By deploying ECMP in data centers, the design can utilize multiple links to provide high redundancy and optimal bandwidth.

Currently, there are two major challenges faced by networking companies. First, ECMP solutions are either proprietary in nature or are customarily implemented and homegrown as per the company’s needs. Thus, they are not open sourced and available for others to use. Second, ECMP is ineffective for traffic distribution since factors such as network congestion and link performance are not taken into account [2]. ECMP also performs poorly when there are either changes in the network topology or the flow size. Additionally, the host is still unaware and incapable of load balancing and forwarding traffic to a different default gateway router in case of an upstream failure or congestion in the link to its current gateway router. If intelligence is available at the end host, it can strategically decide and change its own first hop in an event of a failure or degradation of performance in its link to the current default gateway router. This shall provide higher availability, minimal dropping of packets and decreased latency.

The proposed algorithm shall reduce end-end latency and enable better utilization of links in comparison to the current ECMP model. It also tackles the problem of route computation in case of link failure or topology changes. A successful project would mean that end hosts are intelligent enough to have complete visibility of the network links present in a data center via the controller, before making a decision to initiate traffic. Additional benefits include an integration with COTS (Commercially available off-the-shelf) devices. All the research, design, implementations and the algorithm itself, shall be open sourced for other individuals, companies and research groups, for reviewing, implementing and further augmenting our work.

The specific objectives of this project has been summarized into three success levels, described below. The project shall fulfil all of these levels for the highest success.

**Level 1:**

The project shall accomplish load balancing between the end-host and its respective first hop while combating the detrimental effects of link congestion. The project shall implement Quagga BGP to provide extensive routing capability and make decisions on whether the link to the access switch has crossed the predefined threshold for congestion. If the link is indeed congested, the VM shall predict the next best link available to reach the first hop and direct the traffic to a lesser congested link on the access layer.
**Level 2:**

The proposed project shall accomplish Layer 2 and Layer 3 load balancing using ECMP beyond the first hop, which is at the distribution and core levels. Load balancing at layer 2 shall be provided by transparently interconnecting many links. It shall have multiple layer 3 devices and when data or traffic is sent from the host, ECMP shall equally divide this data to the layer 2 and layer 3 devices. This shall ensure all the links and devices are used to forward traffic and congestion is reduced over a single link or device. A controller shall be used, which shall have an overall view of the architecture, bandwidth capacities and threshold values of the links and shall continuously monitor all the devices and links. The controller, which has a broader view of the network, shall assist the VM agent in making decisions for choosing alternate links in an event of network congestion or link failure. The end-host shall forward traffic on this next best alternate path as suggested by the controller and prevent the network from facing a downtime.

**Level 3:**

The project shall focus on implementing efficient VM migration in case of extreme congestion. Several low bandwidth networks are impacted negatively due to slow and inefficient VM migration as discussed in previous work (section 2.0). Our design shall address this issue by developing an application profile, a generic congestion-aware algorithm that shall be integrated on a controller. The application profile shall define the policies and thresholds for monitoring congestion and the statuses of the links. This shall serve as an input to the algorithm that provides better decision-making capability to the agent. In an event of performance degradation or link failure, VM monitoring and migrating tools such as PowerShell shall be directed to efficiently and dynamically migrate the VM’s to the least congested link with minimum delay. If time permits, the implementation shall be extended to load balance traffic at the transport layer (Layer 4).

A series of thorough tests shall be conducted to ensure data center requirements of high efficiency, redundancy and scalability are met and the migration of the VM’s to the best available links is achieved with minimum or no downtime.

The functional block diagram of the project is as shown in Fig. 1. The agent within each VM obtains information from VM Monitoring tools, app profiler and controller to optimize the ECMP routes. The agent within the host is expected to have Layer 2/Layer 3/Layer 4 VM intelligence. The host shall intelligently load balance traffic and migrate the VM accordingly.
Fig. 1: Functional Block Diagram

Fig 2: Concept of Operation
Fig. 2 explains the interaction between the agent installed on the VM and centralized controller. The diagram shows that a VM located in server S1 does load balancing for two possible paths to the distribution router. For the paths beyond the default gateway, the controller suggests the decision making to the agent, which then chooses a complete path to traverse to Server 7 via the core.

**Technical**

**CPE 1.1: Enable the controller to have visibility of the network, to predict and report link utilization**

The controller shall have network-wide visibility and compute link congestion. This is critical for the intelligent VM to decide the first hop and possible destinations for VM migration. The controller is enabled to anticipate the ramifications of the VM migration to the minimally congested server.

**CPE 1.2: Enable the Virtual Machine to predict the utilization of upstream paths**

The Virtual machine shall calculate the resource utilization of the links to the upstream routers with which it has formed BGP neighborship. Also, it shall foresee the utilization of the links beyond next hop, by communicating with the controller. This shall avoid sending traffic to the router whose upstream links are congested.

**CPE 1.3: Significant East-West traffic intelligence and control**

The proposed design shall have a centralized controller with the ability to monitor and predict link utilization for reducing failovers and congestion. This shall provide an efficient approach to handle increased machine-to-machine traffic.

**CPE 1.4: Develop algorithm to monitor the link congestion in the controller and enable the VM to make decisions based on that**

Our research shall develop an algorithm to determine the usage of the links on the controller and VMs, take decisions on the next hop based on those predictions, and choose the destination for VM migration.

**CPE 1.5: Platform Independent solution with widespread implementation**

The solution to the problem shall be platform independent. It shall be able to work perfectly across any device platform.
Logistical

CPE 2.1: Acquiring Licenses

Our research shall need licensing permissions to the tools that shall be used for execution of the project like VMware Vsphere, VMware PowerShell, and VM Manager Plus.

CPE 2.2: Acquiring network devices

The team shall need access to the equipment, which shall facilitate a large number of VMs providing abilities to run Quagga BGP, Routers, and Switches across different vendors.

Financial

To test the solution extensively, our project might need financial support to acquire licenses/devices, which are not available in the telecommunications lab.

3.0 Design Requirements (20 pts)

Functional requirements (FNC.X), as well as design requirements (DES.X.X) for the project are specified below to improve the project clarity.

FNC.1 The VM shall be able to load balance the packet to the first hop by using links with minimal congestion.

DES.1.1: The VM shall implement Quagga BGP to provide extensive routing capability at the host.

DES.1.1.1 The VM shall decide whether the link to the next hop has crossed the predefined threshold for congestion.

FNC.2 The project shall accomplish Layer 2 and Layer 3 load balancing beyond the first hop. This would ensure uniform distribution of traffic and efficient bandwidth utilization for all the links in the network.

DES.2.1 A centralized controller with overall network visibility shall continuously monitor devices and links and provide threshold values to detect congestion.

DES2.2 In case of congestion, the controller shall notify the agent in the VM to choose the least congested path.

FNC.3 In an event of congestion or extreme failure, migration of the VM to another server hardware shall be accomplished to keep the VM running without a failure of transmission of traffic.

DES.3.1. This shall be achieved by developing a generic congestion aware algorithm which shall be running on the end host.
DES.1.3.1 An App profile communicating with this algorithm and agent shall be running on the VM.

DES.3.2 With the predefined threshold and profiles from the App profile, a migration decision shall be taken whether to carry out the migration.

DES.3.4 A thorough testing mechanism shall be undertaken to ensure data center requirements of minimum or no downtime are met during the migration of the VM towards a better healthier or less congested link.

4.0 Key Design Options Considered (20 pts)

For the VM to successfully load balance packets at the first hop, a routing agent must be implemented at the host. The agent shall enable the host to form multiple neighborships with first hop routers. The VM would then load balance data traffic between multiple first hops. An option of First Hop Redundancy Protocols can be considered, but they are not immune to failures and do not provide additional intelligence to end hosts. There are several options available in the industry for implementing routing agents, which include Quagga BGP, ExaBGP, and Bird. Quagga is one of the most well known routing agents and works as a collection of several small daemons running individual routing protocols like OSPF, BGP, IS-IS and more. It has widespread acceptance, and several troubleshooting guides are available across different support forums. The downside of using Quagga is that the developmental work is slow, and there are several cases of instability seen [1]. ExaBGP offers a highly programmable BGP solution through Python. The downside of using ExaBGP is the fact that it does not employ FIB manipulation. Bird is compatible on several operating systems and offers a good route filtering mechanism. A detailed overview of these agents have been indicated below:

**Table 1: Comparison of different routing agents**

<table>
<thead>
<tr>
<th>Routing Agent</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quagga</td>
<td>- Compatible with Linux OS, and OpenFlow. Similar to iOS</td>
<td>- Performance issues in production networks [1]</td>
</tr>
<tr>
<td></td>
<td>- Widespread implementation in data centers. Backers include Cumulus Networks and Google</td>
<td>- BGP daemon is inefficient for Route Server capabilities</td>
</tr>
<tr>
<td>ExaBGP</td>
<td>- Dedicated for BGP implementation</td>
<td>- Lack of support in comparison to widely used Quagga agents</td>
</tr>
<tr>
<td></td>
<td>- Extensions available in terms of Flask and using HTTP APIs</td>
<td>- Requires high level of expertise and development experience</td>
</tr>
<tr>
<td>Bird</td>
<td>- Effective for BGP Route Server/BGP Route Reflector</td>
<td>- Operating model is different than iOS and JunOS</td>
</tr>
<tr>
<td></td>
<td>- Full-featured protocol suite, with additional features like IPv6 RA and multiple routing tables</td>
<td>- Runs a separate daemon for IPv4 and IPv6. No support for MP-BGP</td>
</tr>
</tbody>
</table>
Hypervisor or Virtual Machine Monitor (VMM) is a program that allows multiple Operating Systems to share a single hardware and manages the execution of different operating systems. Virtual Machines are created and run on the hypervisor. Several hypervisors such as VMware, OpenStack, KVM and Hyper-V are used in the market. A brief analysis of the hypervisors is shown in the following table:

Table 2: Comparison of different types of hypervisors

<table>
<thead>
<tr>
<th>Hypervisor</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMware</td>
<td>- Broad hardware compatibility options. &lt;br&gt; - Zero downtime between different physical hosts. &lt;br&gt; - Allows shutting of physical hosts not in use by VMs, thereby saving memory and CPU. &lt;br&gt; - Live migration out of physical hosts in order to provide maintenance.</td>
<td>- Free subscription for the first 60 days. &lt;br&gt; - Reduced application performance as it has too many applications inbuilt like VMotion, VMware High Availability (HA), Fault Tolerance(FT) and Site Recovery Manager (SRM) [7].</td>
</tr>
<tr>
<td>OpenStack</td>
<td>- Support for Linux and Windows based Operating systems. &lt;br&gt; - Open cloud platform and supports integration of various hypervisors [8].</td>
<td>- Does not support automatic shutdown of physical hosts not in use and no data recovery capability. &lt;br&gt; - No inbuilt VM migration tool available making it difficult to migrate VM’s.</td>
</tr>
<tr>
<td>KVM</td>
<td>- Free open source virtualization architecture. &lt;br&gt; - KVM offers a lightweight, high</td>
<td>- Live migration may cause service suspension for up to 2 seconds.</td>
</tr>
</tbody>
</table>
Layer 2 load balancing is typically done between the links of an EtherChannel. As Layer 2 load balancing is done by the hardware, it is fast and less CPU-intensive. EtherChannel Load-Balancing can be done through MAC addresses at Layer 2, IP addresses at Layer 3, or port number (source, destination or both) at Layer 4. The load balancing method used should ensure an even distribution of traffic across all the links. If the destination MAC or IP address is the same, then load balancing shall not be efficient as they use the same link. Moreover, if one pair of hosts have a lot of traffic between them using different services, the load balancing method based on port numbers is more efficient.

Layer 3 load balancing techniques include First Hop Redundancy Protocols (FHRP), Hash-based, Modulo-n, Highest Random Weight and Equal Cost MultiPath (ECMP) load balancing.

In FHRP, the different approaches are Hot Standby, Virtual Router, and Gateway Load Balancing protocols.

- HSRP and GLBP are Cisco Proprietary and VRRP is IEEE standard.
- HSRP and VRRP do not load balance traffic by default. However, load balancing can be achieved on a per-host basis. GLBP load balances based on round robin or weighted round robin methods.
- HSRP and GLBP support IPv6 whereas VRRP does not support IPv6 [2].

In the Hash-based algorithm, the header information such as source and destination address is used by the algorithm to compute a hash using which, it selects the path to send the traffic[3]. However, when same pair of source and destination addresses are used for hashing, the flow shall use the same link making the link utilization high for that path. Modulo N uses a simple algorithm which computes a modulus and applies to a hash function [3] to select the next hop among N possible hops. In Highest Random Weight algorithm, the load balancing is based on weights assigned to a link [3]. These methods do not consider the costs and delays involved in different paths and are susceptible to hashing traffic to an already congested link.

In the case of ECMP, the agent should be capable of routing with multiple routes to the first hop and should have equal cost to the destination along all the paths. ECMP can be done on the premise of per destination and per packet [4]. In the per-destination approach, the traffic to one destination is sent over one path and different destinations shall use different paths [4]. Hence, in this mode, all packets are subjected to similar delays across the path and shall reach the
destination in a sequence. Although the packets arrive in a sequential order and reduce the overhead in the destination buffers, this method shall fail to utilize the capacity of all the links in a scenario where all the traffic is sent to a particular destination. On the other hand, the per-packet approach shall load balance on the packet basis and shall ensure that all the links are used in a round robin or weighted round robin manner\[^4\]. However, this technique shall compel the packets to follow different paths and hence these packets shall be subjected to different amounts of delay at each hop. This shall have a consequence that the packets might arrive at different times at the destination and might trigger retransmissions of these delayed packets.

There are numerous operating systems that have been deployed in the industry. The most commonly used operating systems are:

- Microsoft Windows
- Mac OS X
- Linux/Unix

In addition, Solaris, HP-UX and DG-UX are other variants of Unix that are commonly used for Server computers. The project is penchant towards a Linux operating system. The key factors driving this preference are as follows:

- **Stability**: Downtime can have disastrous consequences in networking applications. Stability is a critical component to be considered for selecting an Operating System (OS). Linux distributions are more stable and have the ability to handle a large number of processes effectively as compared to its OS counterparts. Also, different routing agents such as quagga BGP and ExaBGP work great with a Linux operating system.

- **Total Cost of Ownership**: Since Linux is an open source software and does not have any licensing costs unlike its counterparts such as Windows and Mac OS. This not only reduces the overall cost of the project but also extends a scalable solution.

- **Flexibility**: The project requires VM’s to have Layer 2/Layer 3/Layer 4 intelligence. This would demand appropriate modifications in the operating system. Various parameters can be obtained and features incorporated by modifying the Kernel. Also, there is no commercial vendor that would enforce adoption of certain products and protocols.

- **Security**: Linux is considered more secure as compared to Windows and Mac OS. This is primarily due to various features of Linux such as Account Privileges, inbuilt iptables and efficient logging.

While the above features favor the adoption of a Linux operating system, if time permits, support for Windows shall be attempted to ensure that the project is OS independent and generic.
Table 3: Comparison of different Linux distributions

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Pros</th>
<th>Cons</th>
<th>Package Management</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CentOS</td>
<td>This distribution is extremely stable and reliable and extends long-term support and maintenance cycles. It is one of the most commonly deployed distribution for commercial use.</td>
<td>Associated with very long release cycles and thus packages are generally not up to date.</td>
<td>Yum and RPM</td>
<td>CentOS does not enforce mandatory package migration and software upgrades making it an ideal choice for commercial use</td>
</tr>
<tr>
<td>Debian</td>
<td>&quot;The distribution is very stable and reliable and provides support for a wide variety of packages. It extends support for a variety of architectures.&quot;</td>
<td>Very slow release cycles.</td>
<td>Apt and Deb packages</td>
<td>Provides long-term support for most of the available software packages which include bug fixes and security support. This distribution has over 50,000 software packages making it one of the largest software compilations available. This serves as a great solution for commercial use.</td>
</tr>
<tr>
<td>Ubuntu</td>
<td>This distribution is associated with solid release cycles based on long-term support and intermediate</td>
<td>Packages are less stable compared to its counterparts mainly due to frequent package</td>
<td>Apt and Deb packages</td>
<td>This distribution is strongly influenced by Debian. Ease of use is one of the main reason why this distribution emerges as the most</td>
</tr>
<tr>
<td>Distribution</td>
<td>Features</td>
<td>Unique Tools</td>
<td>Remarks</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>OpenSUSE</td>
<td>This is a stable distribution with numerous support packages. It also provides system administration tools such as &quot;YaST&quot; used for installation and configuration purposes. Includes unique tool such as YaST which is not interoperable. Affected by Novell's acquisition and Microsoft dealings.</td>
<td>YaST and RPM</td>
<td>This distribution is known for its user-friendly desktop environment. It uses a specialized installation and configuration tool called &quot;YaST&quot;.</td>
<td></td>
</tr>
<tr>
<td>Fedora</td>
<td>Numerous up to date software with recent releases available.</td>
<td>Yum and RPM</td>
<td>While Fedora is commonly deployed, it is not known for long-term support. Also, it has frequent software releases that make it unsuitable for long-term deployments. However, if the application demands a distribution that is up to date, Fedora serves as a great option.</td>
<td></td>
</tr>
</tbody>
</table>

The table above shows the comparison of different Linux distributions. For the reasons of stability, familiarity, and simplicity, CentOS shall be used as the Linux distribution [11].

The design provides various hardware options that can be used. Devices from vendors like Cisco, Juniper, Arista provide a wide variety of choice for choosing routers and switches for the network. Cisco provides options in both, Routers and Switches and the advantage of using Cisco
gear is that its CLI is user-friendly and widely accepted. Due to extensive documentation and support available for various features, Cisco is an option that is considered. On the other hand, Juniper devices also provide high-end devices for data center, however, the CLI is different from most vendor, which shall require added efforts to adapt to the new commands and CLI environment. Majority companies in the industry have some or the other flavor of a Cisco device in their networks.

However, compared to Cisco, Juniper is based on FreeBSD Unix shell, and even though a lot of Kernel functionalities are not available to the customer, it still does provide a filesystem hierarchy which would enable the data collection from files more conveniently, scripts can now be integrated and run on the device itself. The processing of operations and functions is handled separately in Juniper devices. This offers better internal modularity and scalability, providing equipment that is comparatively more robust to internal failures. Juniper also provides better options at the core layer of the network topology and this is evident from that fact that major service providers like Verizon, Charter Communications and Comcast use Juniper Routers for handling core traffic [5].

For Switching options, there is an incentive to use Arista gear as it provides UNIX like environment and capabilities integrated with Cisco CLI commands. Like Juniper, Arista also offers a modular OS where working and failure of one process doesn’t affect any other process in the OS. Another advantage of using Arista gear is its support for features like MLAG and chassis aggregation to enable redundancy and high availability in the network at multiple tiers of the topology.

Finally, that being said, the design goal is to avoid vendor lock-in and build a solution that is not bound to a particular vendor or platform.

The Controller is a critical component of the project. The controller shall collect the parameters of the link such as link bandwidth, link status, latency, congestion among other parameters. The VM shall make decisions of load balancing and migration based on these parameters at a particular instant of time. There are many implementations of the controllers already in the industry.

Some of the important parameters that will be collected by the controller are as follows:

- **Bit rate/Network pipe capacity** - The number of bits per second that the links can transmit. This is needed to know how much traffic the access, distribution and core switches and routers can handle during migration.

- **Migration time** - This is the time needed for migrating the VM from one link to another at times of link failure or congestion.

- **Total Downtime** - The time in seconds during which the network is inaccessible while migrating a VM from one link to another.

- **RAM requirement** - The RAM identifies the the speed at which any process can be completed. Sufficient amount of RAM optimizes the performance and provides the capability to support various types of software.
- **CPU available** - CPU defines the capacity of the hypervisor to perform functions or processes. CPU fetches instructions from RAM, processes it, computes results and instructs other components to take specific action based on the computed results. The higher the CPU the better and faster is the hypervisor.

- **Consistency after migration** - This defines how consistently the Virtual Machine is performing after it has been migrated to another server.

- **Size limitations of a VM** - This defines the maximum size of the Virtual Machine.

- **Disk Usage** - Disk usage defines the amount of memory the hypervisor is capable of storing. This parameter is checked to see if the server has sufficient memory capacity for a successful migration of Virtual Machine to occur.

- **Link status** - The link status is a key parameter which checks the physical status of the link, as well as the link layer (L2) status. Based on the status, the alternative paths could be chosen via ECMP.

- **Total number of VM’s supported** - The total number of Virtual Machines that a hypervisor can support.

Another critical point that needs to be considered for the success of the project is to identify the traffic paths, which shall be used by the agent to communicate with the controller. The flow diagram for the communication can be represented as follows:

![Figure 4: Communication between VM agent and centralised controller](image)

There are several options to enable this communication, and they have been listed as follows:

- **Existing links**: The existing links could be used to send the information from the controller to the agent. Here, it may happen that the links get congested, and the agent does not get the required information on time. This may affect the traffic sent currently from end hosts. Also the existing traffic may drain the actual data traffic flowing between the hosts which may result in severe performance degradation. Hence, this method is not recommended and shall not be used.
- **Out of band network**: Enabling an out of band network helps achieve the purpose of safely sending requisite information to the agent in a time-bound manner. In addition, the data collected from intermediate devices needs to be secured, which this method helps the team achieve. A separate network shall be assigned for this purpose, and this method shall be used for communication between the controller and the agent.

**Sampling or Frequency**: The proposed design will collect the data of the parameters mentioned below every second. However, the agent on the VM based on the data sample for 30 seconds will take the decision about the migration. This is done in order to ensure that the migration decisions are apt and the migrated VMs do not vacillate.

There are many implementations of the controllers already in the industry. Some of the major controllers are discussed below:

The Cisco APIC controller is widely used in data centers to monitor, automate and configure a set of parameters as required. It is capable of operating with six vCPUs and hence it offers robust performance. However, the Cisco APIC supports neither link con gestion monitoring, nor server monitoring. Another downside is that it cannot be used for VM migration in a datacenter.

The OpenStack controller is the best open source controller available. Additionally, it is the most lightweight of the controllers listed. The controller can support most of the hypervisors available in the market.

The VMware controller offers zero downtime, but it is difficult to monitor link congestion. However, the controller does offer server-monitoring capabilities. The controller is supported by most of the vendors, but works best when used with a VMware hypervisor. As far as the interoperability is concerned, the controller works well with almost all the heavily used OS platforms in the industry [15].

<table>
<thead>
<tr>
<th>Controller</th>
<th>Compatibility with Other devices</th>
<th>Licensing</th>
<th>Downtime during Migration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco APIC</td>
<td>Supported only in a network with Cisco devices only. Interoperability is a concern and not completely SDN based controller. More automation oriented and thus does not provide all the features of an SDN based controller in a data center.</td>
<td>Cisco-specific licensing and product purchase required to download the image for the controller. Single image price is approximately $20,000</td>
<td>Not specifically used for enabling migration in data center.</td>
</tr>
</tbody>
</table>
**5.0 Trade Study Process and Results (30 pts)**

The trade study provides a reasonable assessment of the pros and cons of the design options provided and aids in suggesting the best solution while eliminating the others. The trade study conducted brings forth the trade-offs that the design might have to undergo in order to make the design feasible.

<table>
<thead>
<tr>
<th>Table 5: Trade off parameters considered</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hypervisor</strong></td>
<td>Considering implementing of a very large data center with capability of supporting hosts using wide range of operating systems, VMware hypervisor seems to be the most feasible hypervisor. VMware supports both Windows and Linux based operating systems. VMware 6.0 can accommodate 8000 virtual machines, the host can support 480 physical CPU's and 12 TB of RAM [7]. Though the experiments will be using around 20 VM’s for testing purpose, higher capacity will be an important consideration for deploying in production network. VMware has an inbuilt VM migration tool vMotion which can provide live migration with zero downtime. More scalable than any other hypervisor, supports serial ports, provides data recovery and has enhanced security. VMware has trial period of 60 days, hence the project may use OpenStack as an alternate option or may require funding to buy VMware license.</td>
</tr>
</tbody>
</table>
Routing and Switching Hardware

Considering cost of buying new equipment, availability of devices and our design requirements for this project in mind, using Cisco and Arista gear in the network is a more viable options. Both Cisco and Arista provide an easy-to-use CLI environment. This eliminates the need to learn different CLI, which would be the case if Juniper devices were used in the network. Cisco and Arista devices provide enough link capacity and compute power to process and forward maximum traffic flowing through the network.

Routing Agent

The routing agent is a significant part of the design process. The ability of the agent to pair with next-hop routers, and handle routes effectively shall dictate the overall success eventually. While several open source routing agents have been discussed, ExaBGP offers several custom extensions, which shall be useful in the project. The agent works well even in high-speed production environments, as per the team’s interaction with Facebook Network Engineering Team. The ExaBGP agent should be capable of handling more than 300k+ routes and be resilient to failures by utilizing an underneath protocol like BFD. For testing failures, there shall be route flap tests for at least 500 routes, and intermediate packet loss and latency tests shall be the metrics to look for optimal performance here. However, Quagga could also be used since it may work equally well with VMWare hypervisors.

Controller

VMware ESX has support for Openstack controller. There is enough documentation online to enable this integration. This has an added advantage in terms of licensing as Openstack is Open source and VMware ESX has free trial version [6].

Load Balancing

Load balancing is an integral part to distribute the congestion across the links. While several load balancing techniques have been discussed they fail to consider the cost across paths while load balancing. However, ECMP load balances based on cost and delay. Moreover, by load balancing ECMP on per packet basis, the design shall ensure that the traffic is distributed across all the links. Though this might have consequences in packet reception order at the receiver, this method is the best to split the traffic across multiple links. However, if the TCP retransmissions are high due to packet loss/delay, ECMP load balancing with per flow-based approach can be used.

The trade-off matrices for the below two components of the project are as follows:

1. Choice of hypervisors
2. Choice of controllers
The following table lists the important trade-off parameters for Hypervisor:

### Table 6: Comparison of important trade-off parameters for hypervisors

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Weight (0-1)</th>
<th>Vmware</th>
<th>KVM</th>
<th>OpenStack</th>
<th>Hyper-V</th>
<th>Scale (1-5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Licensing</td>
<td>0.20</td>
<td>1.00</td>
<td>3.00</td>
<td>4.00</td>
<td>2.00</td>
<td>1: Indicates large licensing cost per Unit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5: Indicates no licensing cost per unit (Open Sourced)</td>
</tr>
<tr>
<td>Support</td>
<td>0.10</td>
<td>4.00</td>
<td>2.00</td>
<td>2.00</td>
<td>2.00</td>
<td>1: Indicates poor support community.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2: Indicates great support community.</td>
</tr>
<tr>
<td>Migration time</td>
<td>0.15</td>
<td>5.00</td>
<td>3.00</td>
<td>3.00</td>
<td>1.00</td>
<td>1: indicates the time taken for migration is high in the order of 5-10 minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5: Indicates the time taken for migration is either zero or minimal.</td>
</tr>
<tr>
<td>Total Downtime</td>
<td>0.20</td>
<td>5.00</td>
<td>4.00</td>
<td>3.00</td>
<td>2.00</td>
<td>1: Indicates significant downtime over 10 seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5: Indicates a zero downtime</td>
</tr>
<tr>
<td>Total RAM supported</td>
<td>0.15</td>
<td>4.00</td>
<td>2.00</td>
<td>3.00</td>
<td>2.00</td>
<td>1: Indicates the RAM memory supported is less. (&lt; 4TB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5: Indicates it can support very large RAM (&gt; 12 TB)</td>
</tr>
</tbody>
</table>
The following table lists the important trade-off statistics and parameters for controller:

Table 7: Comparison of important trade-off parameters for the controller [12],[13], [14]

<table>
<thead>
<tr>
<th>Features</th>
<th>Cisco APIC</th>
<th>VMware</th>
<th>Openstack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migration/Downtown</td>
<td>Variable</td>
<td>Zero</td>
<td>2 seconds</td>
</tr>
<tr>
<td>Required RAM</td>
<td>32/ 64 GB</td>
<td>12 GB</td>
<td>2/ 4 GB</td>
</tr>
<tr>
<td>Required ROM</td>
<td>500 GB recommended</td>
<td>60 GB minimum</td>
<td>5/10 GB</td>
</tr>
<tr>
<td>vCPU</td>
<td>6</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Link congestion monitoring</td>
<td>No</td>
<td>No</td>
<td>Configurable</td>
</tr>
<tr>
<td>Server monitoring</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The trade matrices for the controller can be represented as follows:

**Table 8: Comparison of important trade-off matrices**

<table>
<thead>
<tr>
<th>Features</th>
<th>Weight</th>
<th>Cisco APIC</th>
<th>VMware</th>
<th>OpenStack</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migration/Downtown</td>
<td>0.1</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1: Downtime greater than 10 seconds 5: Zero Downtime</td>
</tr>
<tr>
<td>Required RAM and ROM</td>
<td>0.05</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1: RAM less than 4GB and ROM less than 10GB 5: RAM greater than 32GB and ROM greater than 500GB</td>
</tr>
<tr>
<td>vCPU</td>
<td>0.05</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>1: 1 or less vCPU supported 5: 5 or more vCPU supported</td>
</tr>
<tr>
<td>Link congestion monitoring</td>
<td>0.3</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>1: Not Supported 5: Supported and Configurable</td>
</tr>
<tr>
<td>Server monitoring</td>
<td>0.2</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>1: Not Supported 5: Supported and Configurable</td>
</tr>
<tr>
<td>Licensing</td>
<td>0.2</td>
<td>1</td>
<td>3.5</td>
<td>5</td>
<td>1: Cost more than 5,000 USD 5: Open Source</td>
</tr>
<tr>
<td>IPv6 Support</td>
<td>0.1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1: Not Supported 5: Supported</td>
</tr>
<tr>
<td>Grand Total</td>
<td>1.0</td>
<td>0.93</td>
<td>2.5325</td>
<td>3.925</td>
<td></td>
</tr>
</tbody>
</table>

Other controllers, including VMware controllers are capable of monitoring failures in the network and perform a migration when required. However, a separate algorithm is required to inform the controller to monitor these above-mentioned parameters and perform a migration if certain predetermined thresholds are reached. This shall be implemented in the algorithm that shall be developed as a part of this project.
6.0 Selection of Baseline Design (20 pts)

In order to run multiple hosts in a single hardware, the project necessitates the use of a hypervisor. First hop load balancing can be achieved by using intelligent hosts. While all the hypervisors analysed do support a wide range of operating systems including Linux and Windows operating systems, VMware offers additional features which are significant to achieve our long-term objectives. The VM monitoring and migrating tool - vMotion is inbuilt into VMware, provides live migration with zero downtime and is vendor neutral. Additionally, it saves the CPU and memory utilization by shutting off the hosts which are not used by the VM. The project may use OpenStack hypervisor as the second best option for hosting virtual machines. This may be required in case the VMware license expires. OpenStack supports variants of hypervisors such as KVM, VMware and Hyper-V, hence it is possible to customize or integrate VMware in OpenStack.

The team believes that OpenStack controller shall be the best choice to monitor congestion and link health in the network. The biggest advantage of using OpenStack as the controller here is that, first; VMware ESX hypervisor (the project’s choice of hypervisor) is compatible with it. Second, Openstack is open source and thus available openly to everyone. Third, it avoids vendor lock-in and finally, it is flexible and can be customized as per the organization’s requirements. Openstack also allows configuring parameters and shall be compatible with our congestion aware algorithm.

As per the functional design pointer (FNC 1.1), the end host must be able to load balance the packet from itself to the first hop by using links with minimal congestion. Hence, there is a need to consider a viable routing agent to pair with our first hop BGP speaking routers. For choosing agents, the team considered several factors before reaching a definite conclusion. These factors include proprietary or nonproprietary solution, cost, performance, and flexibility in implementation. The team has chosen to use ExaBGP over the other options because it offers robust performance in large-scale datacenters, and is open sourced as well. Even though there is a definite drawback in terms of support available to more popular platforms such as Quagga and BIRD, the team believes that in terms of performance, ExaBGP would suit the project requirements better.

For implementing a network to build, implement and test our solution, the topology shall make use of Cisco and Arista hardware to carry out the traditional routing and switching functions in the datacenter. The agenda of FNC 1 and FNC 2 are reliant on deciding the optimum way to load balance across links at the first hop and beyond the first hop. The hashing based load balancing techniques like modulo N, Highest random weight do not consider parameters of the path like cost. The load balancing done by GLBP is Cisco proprietary and hence has limits to implementation on other platforms. While the other First hop redundancy protocols have load balancing methods restricted to per host basis. Hence, ECMP load balancing algorithm, which considers cost, and load balances based on MAC, IP addresses, and port numbers shall fit the needs of the design appropriately.
References:


2017].

