Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

HIGHER PERFORMANCE DOCKER BASED TELEMEDICINE INTEGRATION USING VIRTUALIZED ENVIRONMENT

Jamal Kh-Madhloom[1], MohdKhanapiAbd Ghani[1]

[1]Biomedical Computing and Engineering Technologies (BIOCORE) Applied Research Group,
Faculty of Information and Communication Technology,
UniversitiTeknikal Malaysia Melaka, Malaysia

engineerjamal112@yahoo.com, khanapi@utem.edu.my

Abstract

Telemedicine that is delivery via virtualization is the use of virtual health machine that can virtualize the greater part of the equipment assets, including processors, memory, and network availability including the health equipments. With the virtualization, physical equipment assets can be shared by at least one virtual machines. According to the necessities from Popek and Goldberg, there are three perspectives to fulfill the virtualization. To start with, the virtualization ought to give an identical domain to run a program contrasted with a local framework. To comprehend virtualization, hypervisor ought to be addressed first. Hypervisor empowers correspondence amongst equipment and a virtual machine so that the virtualization fulfills with this deliberation layer (hypervisor). Hypervisor is originally called virtual machine screen (VMM) from these two terms (Hypervisor and VMM) are regularly regarded as equivalent words, yet according to the distinction from Agesen et al, a virtual machine screen (VMM) is a product that manages CPU, memory, I/O data exchange, interrupt, and the instruction set on a given virtualized environment. This manuscript underlines the integration of telemedicine for the delivery of dynamic medical services on assorted segments in optimized aspects.

Keywords: Cloud Delivery of Health Services, Docker, Software Defined Networking, Telemedicine

A hypervisor may allude to an operating framework (OS) with the VMM. Commonly, a hypervisor can be partitioned into Type 1 and Type 2 hypervisor in view of the diverse level of execution. Sort 1 is sitting on equipment and the correspondence amongst equipment and virtual machine is immediate. The host operating framework is not required in Type 1 hypervisor since it runs straightforwardly on a physical machine. Because of this reason, it is now and again called an 'exposed metal hypervisor'.

VMware vSphere/ESXi, Microsoft Windows Server 2012 Hyper-V, Citrix XenServer, Red Hat Enterprise Virtualization (RHEV) and open-source Kernel-based Virtual Machine (KVM) are distinguished in this class. Sort 2 hypervisor is on the operating framework to manage virtual machine effectively with the support of equipment arrangement from operating framework. The additional layer amongst equipment and virtual machine in the sort 2 hypervisors causes inefficiency contrasted with the sort 1 hypervisor. Virtual Box and VMware Workstation are in this class.

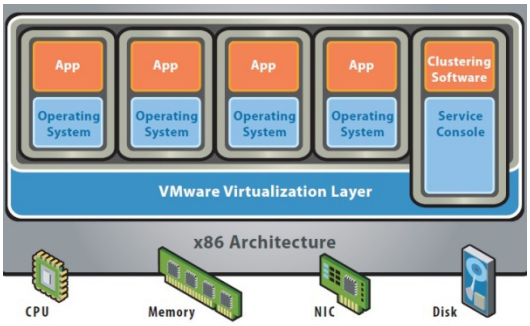
Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02



Bare-Metal (Hypervisor) Architecture

Fig.1. Hypervisor Architecture

The terms of Host or Guest machine (or domain) are utilized as a part of the hypervisor to depict diverse parts. Host machine (domain) contains a hypervisor to manage virtual machines, and Guest machine (domain) implies each virtual machine sitting on a hosted machine in a protected and confined environment with its own consistent domain.

With these separated parts, the hypervisor can offer asset limits to different virtual machines on the same physical machine. In other words, the hypervisor is a product layer that makes a virtual domain with virtualized CPU, memory and I/O (stockpiling and network) gadgets by abstracting without end the underlying physical equipment. Virtual machine (VM) commonly alludes to an epitomized substance including the operating framework and the applications running in it also.

TYPES OF VIRTUALIZATION

- **Full virtualization -** Full virtualization gives virtualization without modifying guest operating framework. In x86 engineering, dealing with benefits instructions is a key component for virtualizing equipment. VMware offers binary interpretation of operating framework asks for so that virtualizing favored instructions can be finished without backings from either equipment or operating framework. There are Microsoft Virtual Server and VMware ESXi using this system.

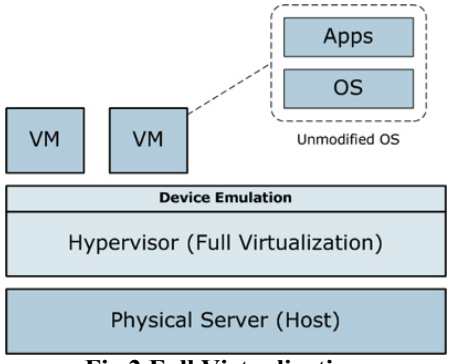


Fig.2 Full Virtualization

- **Para-virtualization-** Xen gather initially created paravirt-operations (later called by paravirtualization) to bolster elite and effective asset segregation with slight adjustments to the guest operating framework. Xen saw that full virtualization upheld guest domains without an adjustment in the operating framework, yet there were negative influences on performance because of the utilization of shadow page tables. Paravirtualization (PV) requires the adjusted OS kernel with framework calls to manage advantaged instructions. Xen registers guest OS page tables specifically with the MMU with a read-just access to keep away from the overhead and intricacy regarding the updating shadow page tables in full virtualization. With the interface between a virtual machine and a hypervisor, paravirtualization accomplishes superior without the help from equipment expansions on x86. Paravirtualization underpins unmodified application binary interface (ABI) so that client applications don't require any progressions. Paravirtualization is additionally called operating framework assisted virtualization in light of the familiarity with a hypervisor on guest OS. Xen, UML and VMware bolster paravirtualization.

- **Hardware Assisted virtualization -** To enhance performance of virtualization, Intel

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

and AMD gives virtualization expansions to x86 processors. Intel Virtualization Technology (VT) and AMD Virtualization (AMD-v) are increasing velocities for favored instructions including memory management unit (mmu), coordinated I/O gadgets (iommu). With this equipment assisted virtualization innovation, altered guest OS is pointless to empower virtualization in light of the fact that VMM manages benefit instruction at a root mode which is a ring - 1 without affecting the guest OS.
Using Second Level Address Translation (SLAT), settled paging in Intel EPT (Extended Page Table) or AMD RVI (Rapid Virtualization Indexing), memory management has been improved and the overhead of translating guest physical addresses to genuine physical addresses has been decreased. Early CPUs for x86 don't have virtualization expansions which are not included in equipment assisted virtualization.

- **Operating System-level virtualization (Shared Kernel Virtualization) -** Operating framework gives separated allotments to run virtual machines in a similar kernel. With a chroot operation, which is a move of a root catalog for a certain procedure with a detachment to outside registries, OS-level virtualization empowers segregation between numerous virtual machines on a mutual OS. Overhead is exceptionally restricted in this model because of the advantages of running under operating frameworks with a common kernel. Emulating gadgets or communicating with VMM is a bit much. The guest os and the host os ought to have a similar OS or kernel. Running Windows on Linux host is incompatible. There are Solaris Containers, BSD Jails, LXC, Linux vServer, and Docker.

SOFTWARE IMPLEMENTATIONS FOR VIRTUALIZATION

Bare-metal virtualization hypervisors

- Microsoft Hyper-V
- VMware ESX and ESXi
- Citrix XenServer
- Oracle VM

Hosted virtualization hypervisors

- VMware Workstation/Fusion/Player
- VMware Server
- Microsoft Virtual PC
- Oracle VM VirtualBox
- Red Hat Enterprise Virtualization
- Parallels Desktop

CONTAINERS

The approach of online programming, administrations and stages alongside commoditization of computing administrations implies that scaling has happened to paramount significance. Considerably more, new paradigms in programming advancement – lithe systems, which abbreviate the way between the engineer and the generation situations, additionally added to an increase in appropriation among ventures and new businesses alike.

These new advancement hones and another arrangement of necessities, as far as continuous integration and continuous sending meant that there was an appropriate seeding ground for thoughts like containerization. Containerization tools empower "unchanging nature" in the infrastructure: container applications that are worked 'on the spot' and 'set up' with an indistinguishable setup and same conditions from the original creators intended.
This is one of the main reasons (or maybe even the main driving power) of their fast selection and increased utilization. These tools permit application engineers to concentrate on the application instead of focusing on the infrastructure while bringing versioning to application picture dispersion. It even goes similarly as bringing ideas that "customarily" had a place with an alternate region: you can do pulls, pushes and confers on Docker pictures, ideas

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

obtained from Source Code Management Software (like Git and Mercurial).

To get any disarray off the beaten path, Docker alludes both to an open-source extend (https://github.com/docker/docker) and additionally the organization behind it – Docker Inc. (once in the past dotCloud). dotCloud was a PaaS player who manufactured Docker for internal utilize. When they understood its potential, they rotated and concentrated only on developing Docker.

The significance of the venture in the more noteworthy biological system was likewise perceived by the vast majority of the industry, along these lines allowing Docker to bring 40M$ up in the latest funding round. Following the turn dotCloud (the PaaS offering of the organization) was "migrated" to Berlin-based cloudControl. Docker is only one of the accessible containerization offerings and, the same number of comparative ventures, depends on kernel highlights that (in Linux) have been accessible for over 6 years (since around 2007).

Notwithstanding, as with other innovations (see e.g. blast in cell phone deals with the introduction of iPhone) Docker has just put a client (and designer) neighborly interface around prior parts. Similar ideas showed up 2-3 years prior in Solaris OS (assemble 51 in Solaris 10), known as Solaris Containers w/Solaris Zones. While initially Docker was a wrapper for LXC (Linux Containers), these days it manages libcontainer – a brought together interface for cgroups and kernel namespaces.

DOCKER

While there is a huge amount of likenesses there are likewise no less than two tons of contrasts between the two. The most important distinction is that, in Docker, there's no overhead introduced by the need to copy a whole OS for the virtual machine. Thusly, Docker gives better performance (as far as speed and limit) when for custom applications in a container, contrasted with virtual machines, gave that the VM host and LXC host have a similar equipment qualities.

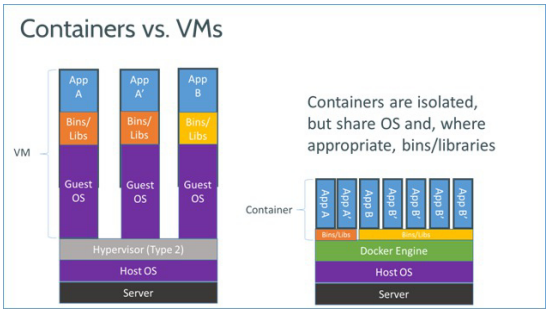


Fig.3 Container Vs. Virtual Machine

Both containers and Virtual Machines address a similar issue – disconnection and control of parts of an application – however this is accomplished in various path as containers surrender a portion of the separation for a more effective use of the (host) framework assets.

DOCKER: AN INTRODUCTION

URL - https://www.docker.com

Docker depends for its execution surroundings, on elements in the host's kernel – LXC. Be that as it may, it additionally needs filesystem bolster in the purported UFS (Union File System). Docker Filesystems Multilayer Early forms of Docker depended on AuFS, yet later discharges have received a "nonpartisan" approach, with an inclination for device mapper.

This utilization of a duplicate on-compose FS is really what makes Docker look like Git (and Docker Hub like GitHub). Docker's two main ideas are Containers and Images, where Containers are, well, containers running the application and Images are the Containers-at-Rest (i.e. spared container state). To make a relationship with class ideas in OOP, Images are class definitions, while Containers are class instances at run-time.

With everything taken into account there are only 28 commands that the tool called Docker (self-entitled "independent runtime" for Linux containers) sees, yet they wrap all the specified capacities – including the control, (for example, begin/stop a container) and the meta-management, (for example, push/draw to/from the store).

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

For instance, starting a container running Redis is as basic as*:
$ docker pull mydockerfile/redis
$ docker run -d --name redis -p 6379:6379 mydockerfile/redis

This will allow one to subsequently use to connect to the Redis server.
<container_host_ip>:6379

The latest version of Ubuntu Linux (14.04 LTS) comes with activated Docker support. It still needs to be installed, though that's just an 'apt-get install docker.io'.

KEY FEATURES OF DOCKER

Application-driven: the incremental change that Docker brings on LXC is core interest on organization of use versus sending of machines
Compactness: once distributed, any picture will yield precisely the same wherever it runs • versioning: much like Git, permits submit/push and pulls on existing pictures, verifying contrasts from one focus on the other
Computerization: there are tools that permit a machine, once running, to achieve a particular state
Sharing: using Docker Hub anybody can expand on existing machines or make accessible their pictures for others

Reusability: a picture can be 'fork'- ed for two distinct purposes.

INSTALLING DOCKER ON WINDOWS

The initial step is to download and install the most recent Docker for Windows work from its GitHub storehouse. As should be obvious in the following figure, Docker for Windows utilizes a conventional Windows installer.
In the event that you acknowledge the installation defaults (which I exceptionally recommend you do), the product gives you the following three parts:

Boot2Docker management tool and ISO: This is the Docker administration command-line interface (CLI).

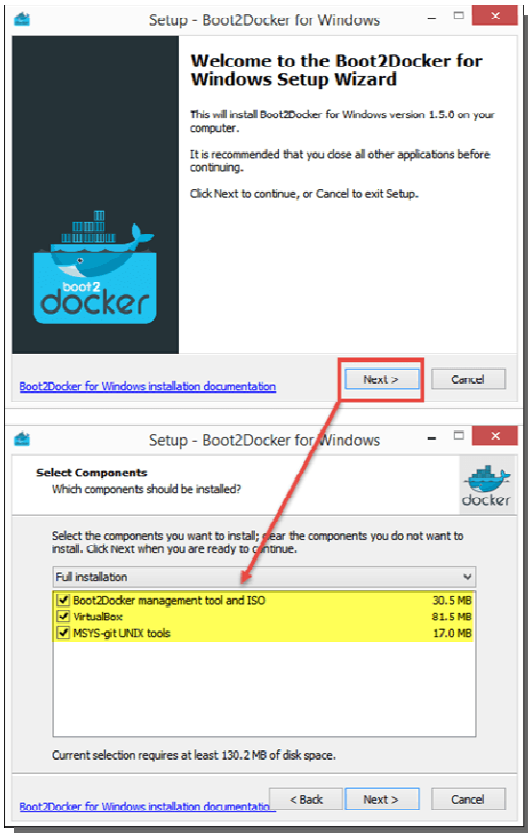


Fig. 4 Boot2Docker Installation

Virtual Box: We require the center of the Oracle Virtual Box VM desktop virtualization item to permit us to run the center Linux code from inside Windows.

MSYS-git UNIX tools: The Docker Hub will pull containers from open Git storehouses, so we need customer tools accessible on our framework.
Next, double tap the Boot2Docker Start Desktop symbol to begin the Boot2Docker management tool. You'll be provoked to permit Virtual Box to bind two virtual network interfaces, and after a short time you'll see an unequivocally Linux-y command quick, as appeared in the following screenshot.

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

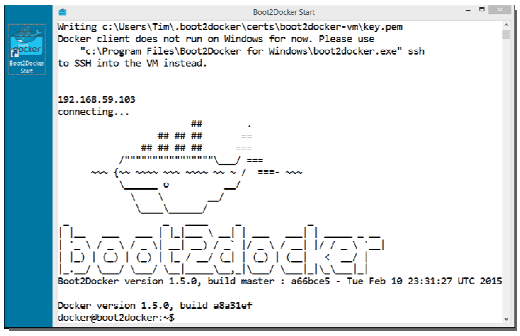


Fig. 5 Boot2Docker Start Screen

LAUNCHING TENSORFLOW

Open up "Docker Quickstart Terminal". It ought to bring up a MinGW-sort shell. You could likewise utilize Windows cmd.exe or Powershell.exe however they have additional arrangements that you have to do before you can run docker.exe

Dispatch the TensorFlow container. Set up port forwarding, mount substance of registry into/home.

$ docker run - it - p 8888:8888 - p 6006:6006 - v/c/Users/name/Dropbox/mypath:/home b.gcr.io/tensorflow/tensorflow

The first run through the command is run, it will Download and install TensorFlow. A short time later, this ought to bring you into a Linux VM.

The command above mounts an organizer in your Windows host machine into the container. Its desirable over do things thusly, in light of the fact that the container does not hold on your documents as a matter of course.

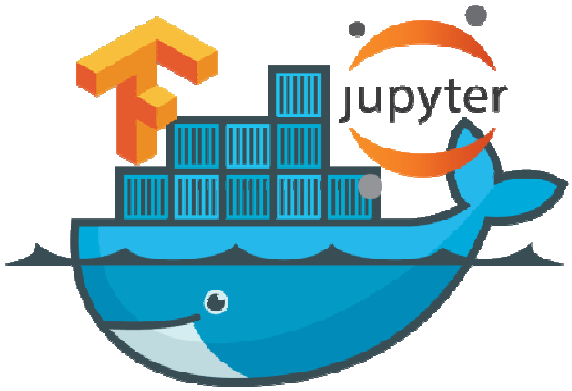


Fig. 6 TensorFlow and Jupyter in Docker

It as of now accompanies Jupyter installed, yet you won't have the capacity to get to the note pads by navigating to localhost:8888 (or whatever port Jupyter begins on). You will likewise want to open port 6006 to have the capacity to utilize TensorBoard (right now, exposing a port on a live container is unrealistic).

Instead, since it is running in a VM, not just do you have to forward the port (thus the - p 8888:8888), yet the IP address you get to should be the IP address of the VM, not the Windows Machine. Thus, you have to find the address of the docker machine running the container.

Listing the docker containers.

$ docker ps
CONTAINER ID IMAGE
COMMAND CREATED STATUS
PORTS NAMES
444413d7235233 b.gcr.io/tensorflow/tensorflow
"/bin/bash" 2 minutes ago Up 2 minutes
6006/tcp, 0.0.0.0:8888->8888/tcp MyPath
$ docker psdocker-machine ls
$ docker-machine ls
NAME ACTIVE DRIVER STATE URL
SWARM ERRORS
*default * virtualbox Running*
tcp://192.168.99.100:2376
$ docker-machine ip default
<My IP Address>

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

Navigate web browser to 192.168.99.100:8888 (or whatever webserver port your web app is running on) and you should be able to see your web apps.

TENSORFLOW AS A RESEARCH ENVIRONMENT

If you want to use this container for your research environment all you need to do is to clone the docker container, so that you can make changes to it.
In the Docker Quickstart terminal,
docker pull b.gcr.io/tensorflow/tensorflow-full
$ docker images

For example if you installed a python package like ipdb and want to make changes, type:
$ docker commit MyPath mypath/tensorflow
$ docker run mypath/tensorflow

Now, the container is ready for research purposes and implementation of algorithms.

TELEMEDICINE AND ITS INTEGRATION ON DOCKER

Telemedicine is the use of telecommunication and information technology to provide clinical health care from a distance. It has been used to overcome distance barriers and to improve access to medical services that would often not be consistently available in distant rural communities. It is also used to save lives in critical care and emergency situations. Although there were distant precursors to telemedicine, it is essentially a product of 20th century telecommunication and information technologies. These technologies permit communications between patient and medical staff with both convenience and fidelity, as well as the transmission of medical, imaging and health informatics data from one site to another. Early forms of telemedicine achieved with telephone and radio have been supplemented with videotelephony, advanced diagnostic methods supported by distributed client/server applications, and additionally with telemedical devices to support in-home care Telemedicine can be beneficial to patients in isolated communities and remote regions, who can receive care from doctors or specialists far away without the patient having to travel to visit them. Recent developments in mobile collaboration technology can allow healthcare professionals in multiple locations to share information and discuss patient issues as if they were in the same place. Remote patient monitoring through mobile technology can reduce the need for outpatient visits and enable remote prescription verification and drug administration oversight, potentially significantly reducing the overall cost of medical care. Telemedicine can also facilitate medical education by allowing workers to observe experts in their fields and share best practices more easily. Telemedicine also can eliminate the possible transmission of infectious diseases or parasites between patients and medical staff. This is particularly an issue where MRSA is a concern. Additionally, some patients who feel uncomfortable in a doctors office may do better remotely. For example, white coat syndrome may be avoided. Patients who are home-bound and would otherwise require an ambulance to move them to a clinic are also a consideration. The downsides of telemedicine include the cost of telecommunication and data management equipment and of technical training for medical personnel who will employ it. Virtual medical treatment also entails potentially decreased human interaction between medical professionals and patients, an increased risk of error when medical services are delivered in the absence of a registered professional, and an increased risk that protected health information may be compromised through electronic storage and transmission. There is also a concern that telemedicine may actually decrease time efficiency due to the difficulties of assessing and treating patients through virtual interactions; for example, it has been estimated that a teledermatology consultation can take up to thirty minutes, whereas fifteen minutes is typical for a traditional consultation. Additionally, potentially poor quality of transmitted records, such as images or patient progress reports, and decreased access to relevant clinical information are quality assurance risks that can compromise the quality and continuity of patient care for the reporting doctor. Other obstacles to the implementation of telemedicine include unclear legal regulation for some telemedical practices and difficulty claiming reimbursement from insurers or government programs in some fields. Another

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

disadvantage of telemedicine is the inability to start treatment immediately. For example, a patient suffering from a bacterial infection might be given an antibiotic hypodermic injection in the clinic, and observed for any reaction, before that antibiotic is prescribed in pill form.

The integration of Telemedicine Objects on Docker and Virtualization Environment can be done for the cloud and hypervisor based delivery of the medical services for higher degree of optimization with minimum delay and higher efficiency.

CONCLUSION

This article focus on the assorted types of virtualization approaches, tools and technologies with the specific case of Docker Implementation for different applications. The concept of virtualization and its applications are quite immense and the more we dig into the matter, the more astonishing results we can fetch. Now days, the virtualization based technologies are widely and prominently used in the health services and Internet of Things (IoT) based integrations and these are quite effectual.

ACKNOWLEDGEMENTS

This research has been supported by Biomedical Computing and Engineering Technologies (BIOCORE) Applied Research Group, Faculty of Information and Communication Technology, UniversitiTeknikal Malaysia Melaka, Malaysia.

REFERENCES

[1] Tanenbaum, A.S., and Wetherall.D.J. *Computer Networks: Pearson New International Edition: University of Hertfordshire*. Pearson Higher Ed, 2013.

[2]. Forouzan, A. Behrouz. *Data Communications & Networking (sie)*.Tata McGraw-Hill Education, 2006.

[3] WAN-Wide Area Network: http://compnetworking.about.com/cs/lanwan/g/bldef_wan.htm.

[4] http://www.buzzle.com/articles/advantages-and-disadvantages-of-computer-networks.html.

[5] http://en.wikipedia.org/wiki/Software-defined_networking#Limitations_of_other_networking_technologies.

[6] Diego Kreutz, Fernando M. V. Ramos, Paulo EstevesVerı´ssimo, Christian Esteve Rothenberg, SiamakAzodolmolky and Steve Uhlig, "Software-defined networking: A comprehensive survey."*proceedings of the IEEE* 103.1 (2015): 14-76.

[7] Monsanto C., Reich J., Foster N., Rexford, J., & Walker, D. (2013, April). Composing Software Defined Networks. In *NSDI* (pp. 1-13).

[8] Voellmy, Andreas, Hyojoon Kim, and Nick Feamster. "Procera: a language for high-level reactive network control." *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012.

[9] Features of SDN Architecture available at: http://www.opennetworking.org/sdn-resources/sdn-definition.

[10] Braun, Wolfgang, and Michael Menth. "Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices." *Future Internet* 6.2 (2014): 302-336.

[11] OpenFlow Switch: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf

[12] OpenFlow ports: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf

[13] Advantages of Software Defined Networking available at: http://www.ingrammicroadvisor.com/big-data/7-advantages-of-software-defined-networking.

[14] SafaeZerrik, Amina El ouadghiri, Driss EL ouadghiri, RachidAtay, Mohamed Bakhouya, Jaafar Gaber. "Towards a decentralized and adaptive software-defined networking architecture." *Next Generation*

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

Networks and Services (NGNS), 2014 Fifth International Conference on. IEEE, 2014.

[15] Naudts, B., Kind, M., Westphal, F., Verbrugge, S., Colle, D., &Pickavet, M. (2012, October). Techno-economic analysis of software defined networking as architecture for the virtualization of a mobile network. In *Software Defined Networking (EWSDN), 2012 European Workshop on* (pp. 67-72). IEEE.

[16] HP-networking-SDN available at: http://webobjects.cdw.com/webobjects/media/pdf/HP/HP-networking-SDN.pdf.

[17] Ng, Eugene. *Maestro: A System for Scalable OpenFlow Control*. TSEN Maestro-Technical Report TR10-08, Rice University, 2010.

[18] Diego Kreutz, Fernando M. V. Ramos, Paulo EstevesVerı´ssimo, Christian Esteve Rothenberg, SiamakAzodolmolky and Steve Uhlig, "Software-defined networking: A comprehensive survey."*proceedings of the IEEE* 103.1 (2015): 14-76.

[19] Bar-Geva, Yitzhak, Jon Crowcroft, and Helen Oliver. "Tearing down the Protocol Wall with Software Defined Networking." *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. IEEE, 2013.

[20] Venkatraman, K., Parthasarathy, V., Kumaar, S. S., &Jayalakshmi, M. (2014, February). Supervision of network through software defined networking. In*Information Communication and Embedded Systems (ICICES), 2014 International Conference on* (pp. 1-7). IEEE.

[21] Naudts, B., Kind, M., Westphal, F., Verbrugge, S., Colle, D., &Pickavet, M. (2012, October). Techno-economic analysis of software defined networking as architecture for the virtualization of a mobile network. In *Software Defined Networking (EWSDN), 2012 European Workshop on* (pp. 67-72). IEEE.

[22] Kobayashi, M., Seetharaman, S., Parulkar, G., Appenzeller, G., Little, J., Van Reijendam, J.& McKeown, N. (2014). Maturing of OpenFlow and Software-defined Networking through deployments. *Computer Networks*, *61*, 151-175.

[23] Schmid, Stefan, and JukkaSuomela. "Exploiting locality in distributed sdn control." *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013.

[24] Li, Li Erran, Z. Morley Mao, and Jennifer Rexford. "CellSDN: Software-defined cellular networks." *Techinical Report, Princeton University* (2012).

[25] Voellmy, A., Wang, J., Yang, Y. R., Ford, B., &Hudak, P. (2013, August). Maple: Simplifying SDN programming using algorithmic policies. In *ACM SIGCOMM Computer Communication Review* (Vol. 43, No. 4, pp. 87-98).ACM.

[26] Sarla Sharma and Manu Sood, "An Architectural Proposal for an SDN based Data Centre: A Case Study." *International Journal* 2.10 (2013).

[27] Dhamecha, Kapil, and Bhushan Trivedi. "SDN Issues–A Survey." *International Journal of Computer Applications* 73.18 (2013): 30-35.

[28] Gupta, Mukta, Joel Sommers, and Paul Barford. "Fast, accurate simulation for SDN prototyping." *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013.

[29] Jammal, M., Singh, T., Shami, A., Asal, R., & Li, Y. (2014). Software defined networking: State of the art and research challenges. *Computer Networks*, *72*, 74-98.

[30] de Oliveira, R. L. S., Schweitzer, C. M., Shinoda, A. A., &Prete, L. R. (2014, June). Using Mininet for Emulation and prototyping software-defined networks. In *Communications and Computing (COLCOM), 2014 IEEE Colombian Conference on* (pp. 1-6). IEEE.

[31] McKeown, Nick, et al. "OpenFlow: enabling innovation in campus networks."*ACM SIGCOMM Computer Communication Review* 38.2 (2008): 69-74.

Publication Impact Factor : 2.348

International Journal of Enterprise Computing and Business Systems

ISSN (Online) : 2230-8849

Volume 7 Issue 2 July - December 2017

International Manuscript ID : 22308849072017-02

[32] Lantz, Bob, Brandon Heller, and Nick McKeown. "A network in a laptop: rapid prototyping for software-defined networks." *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*.ACM, 2010.

[33] Network Emulation available at: http://en.wikipedia.org/wiki/Network_emulation

[34] Roy, Arjun, Kenneth Yocum, and Alex C. Snoeren. "Challenges in the emulation of large scale software defined networks." *Proceedings of the 4th Asia-Pacific Workshop on Systems*.ACM, 2013.

[35] Heller, Brandon. *Reproducible Network Research with High-fidelity Emulation*. Diss. Stanford University, 2013.

[36] Mininet as a Container Based Emulator for Software Defined Networks available at: http://www.ijarcsse.com/docs/papers/Volume_4/12_December2014/V4I12-0364.pdf