GALGOTIAS UNIVERSITY
(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Testing various virtualization solution for different cloud workloads

A Project Report of Capstone Project – 2

Submitted by

GHANSHYAM JI JHA

(1513107016 / 15SCSE107014)

in partial fulfillment for the award of the degree

of

Bachelor of Technology

IN

Computer Science and Engineering With Specialization of business analytics and optimization

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Mr. Prem Prakash Agrawal, M.Tech

Professor

APRIL / MAY- 2020

GALGOTIAS UNIVERSITY

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report “Testing various virtualization solution for different cloud workloads” is the bonafide work of “GHANSHYAM JI JHA (1513107016)” who carried out the project work under my supervision.

SIGNATURE OF HEAD	**SIGNATURE OF SUPERVISOR**
Dr. MUNISH SHABARWAL,	Mr. Prem Prakash Agrawal,
PhD (Management), PhD (CS)	MTECH
Professor & Dean,	Professor
School of Computing Science & Engineering	School of Computing Science & Engineering

ACKNOWLEDGMENT

This research was partially supported by Mr. Amit Chaudhary ’Amit2cha@gmail.com’. We thank our colleague from “TheRoasters” who provided insight and licensing of all products. This paper and the research behind it would not have been possible without the exceptional support of my supervisor, Mr. Prem Prakash Agrawal. His enthusiasm, curiosity and willingness to focus on everything motivated me and continued my work. I want to thank my parents, without their blessings I cannot fulfill my purpose.

Abstract

In early days when real dedicated servers were used people didn’t really cared for top performance out of the machine as physical machines in most cases there were always some head room left for any burst use. Since technologies like virtualization came, our focus changed to getting most performance out for machines for its use cases. This project is designed to testing various virtualization solutions for different cloud workloads and technology such as qemu, lxc, hyper v for various cloud consumption like hosting, high performance computing, database hosting and various other use, so we can give a guideline regarding the use of hypervisor software to get most performance out of a machine.

TABLE OF CONTENTS

TABLE OF TABLE

TABLE OF FIGURES

Introduction

1.1 Overall Description

Virtualization is the process or method to create an abstract layer of physical resources such as CPU, Network as well as Storage. This abstraction allows for better utilisation of the hardware. Virtualization came into existence in 1960s and was brought into practice by IBM first. They wanted to utilize their existing mainframe resources by making them virtually available as they were being wasted.

Many years ago, we started using virtualization technology to test servers and use resources more efficiently. Virtualization was not very popular when VMware came on the market as a hypervisor. During the 1980's and 1990's, Desktop Computing came into existence along with x86 servers. This is why Virtualization was eventually dropped. desktop computing and x86 servers. This is why Virtualization was eventually forgotten. The introduction of client-server applications and also wide popularity of Windows along with Linux OS made server computing expensive. Many challenges lie ahead, and not enough disaster prevention that leads to rise of Virtualization of x86 again. Virtualization improves scalability and resource utilization by a high degree. Using virtualization, several operating systems can run simultaneously on a single CPU. This reduces overall costs and differs from multitasking, which means running several programs on the same OS. Virtualization is a major enabler for Cloud Computing.

However, even Virtualization isn’t perfect. There are many emerging technologies which have proved to improve virtualization. Will these technologies replace virtualization in the future? This is one of the answers we are looking for. One of the technologies that is making its way into the virtualization world is Cloud Containerization. Containerization is an easy way to complete machine virtualization, which involves access application in its equipment.. Your operating environment. It offers many benefits of loading applications on a virtual machine, as the application can be run on any physical machine without having to worry about dependencies.

Another major breakthrough technology is Server less Computing. When using an unstructured computer, developers write the code, and the cloud server monitors the rest. The developer should not think of servers, operating systems, software, or management. Of course, there is a physical server to runs the code, but that’s the cloud service provider’s job. As with the micro service/container scenario, server less computing bypasses the virtual machine layer and functions run on bare metal. At this point, server less computing is relatively naive and use cases are in short supply.

1.2 Purpose

We plan to publish a hands on bible on virtualization and all the major breakthroughs in virtualization. This research is more like a journey of Virtualization from before it started to what its future is. We plan to not just study but to perform a thorough empirical evaluation on these technologies and present comprehensive results. This will help in getting a deeper insight and understanding of these technologies to know which hypervisor is good for which workload.

1.3 Motivation and Scope

The data center industry has moved way beyond simple virtualization, and is exploring new avenues to make virtualization an even more powerful platform. Virtualization has become the trusted workhorse in IT environments of all sizes. It allows organizations to maximize the utilization of their hardware assets, reduce power consumption and migrate one system to another with little to no service disruptions.Major organizations are adopting virtualization these days. Not many are aware of this, but it also plays a critical role in the cloud computing trend. Because of virtualization, the cloud can efficiently deliver the flexible, scalable, cost effective services it is so well known for. Virtualization is now being replaced by several new technologies which will be much more popular in the future. This thorough study on Virtualization along with the experimental research is going to give a deeper understanding of how these technologies work as well as a comparison among them

Literature Review

The data center industry has gone far beyond simple virtualization, exploring new ways to make virtualization an even stronger platform. Virtualization has become a trusted workhorse in IT environments of all sizes. This allows organizations to maximize the utilization of their hardware assets, reduce power consumption and move one system to another without disruption to a few services. Large organizations are adopting virtualization nowadays. Not many are aware of this, but it also plays a critical role in the cloud-computing trend. Because of virtualization, the cloud can effectively provide the scalable, scalable, affordable services they are so well known for.

Virtualization is now being replaced by several new technologies, which will be much more popular in the future. This thorough study on Virtualization along with the experimental research is going to give a deeper understanding of how these technologies work as well as a comparison among them.

2.1 Memory Management in VMWare ESXi- Understanding ESXi- Host Memory States

Suppose your ESXi host has a total memory of 40 GB, but you have 10 virtual machines and each is configured with 4 GB with a total memory of 40 GB. However, in fact the total memory available from your ESXi host is only 30 GB. This over commitment can be achieved with memory management techniques, and not all VMs use 100% of allocated memory at all times. If this happens, the ESXi host will actively use its memory restoration techniques to deal with this condition effectively.

Some techniques, which are used to manage memory, are as follows:

2.1.1. Transparent Page Sharing

Transparent page sharing (TPS) is a memory management technology in virtualization that condenses redundant memory pages on a host into one page

If the hypervisor identifies identical memory pages on multiple virtual machines (VMs) on a host, it shares them among virtual machines (VMs) with pointers. This frees up memory for new pages. If a VM's information on that shared page changes, the hypervisor writes the memory to a new page and readdresses a pointer.

Transparent page sharing is not available for large memory pages. Typically, only small memory pages are identical.

2.1.2 Memory ballooning

Memory ballooning is a memory management feature used in most virtualization platforms which allows a host system to artificially enlarge its pool of memory by taking advantage or reclaiming unused memory previously allocated to various virtual machines.

Ballooning is a process where the ESXi host reclaims memory back from the virtual machine. Ballooning is an activity that happens when the ESXi host is running out of physical memory. The demand of the virtual machine is too high for the host to handle. Each VM has a driver installed via VMware tool

2.1.3. Memory Compression

Memory Compression. Memory Compression. ESXi provides a memory compression cache to improve virtual machine performance when you use memory overcommitment. Memory compression is enabled by default. When a host's memory becomes overcommitted, ESXi compresses virtual pages and stores them in memory.

2.1.4. Hypervisor-level memory swapping

ESXi host never tries to restore memory through memory management Techniques until it is contested in memory. Techniques for restoring such memory Since (Memory ballooning, Compression or swapping) will take action based on Free ESXi host memory.

There are 4 different host countries of ESXi:

1. High

2. Soft

3. Hard

4. Low

High: By default, transparent page sharing will always work.

Soft: Memory ballooning will launch when ESXi enters the soft state and stays active until the ESXi returns to a high state.

Hard & Low: Hard and low: ESXi uses memory compression and Hypervisor-level memory swapping when ESX is in hard or low mode.

Low: Low: If the host's memory usage is above low, the ESXi host will stop creating the new pages for virtual machines and will continue to compress and replace until it clears more memory.

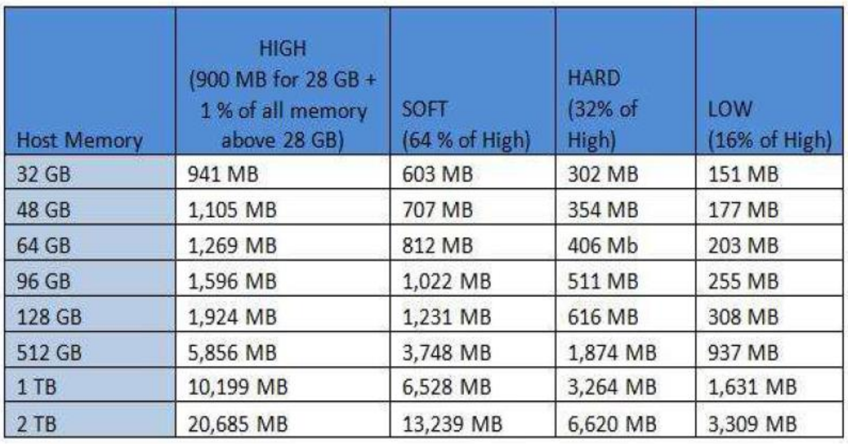
Before vSphere 5, High was set by default at 6%, Soft at 4%, Hard at 2%, and Low at 1%. If the free memory of the ESXi host is less than the specified percentage, ESXi uses the appropriate memory restoration techniques to restore the memory, but think of the host with more memory. It is not necessary to protect that much free. Let's take an example, the ESXi 5.0 host can run with 2 TB of memory. Using these defined values like vSphere 5.0, a host will begin to require memory even if it has 100 GB of free memory. This is not a great option. So with vSphere 5.x, these predefined values have been changed to effectively handle the host restoration techniques for the ESXi host with more memory.

With vSphere 5, the high state level is adjusted according to the amount of memory in the host. Here's the high calculation -> 900MB for 28GB + 1% of all memory over 28GB (if the host holds 28GB of memory)

Soft -> (64% of High)

Hard -> (32 % of High)

Low -> (16 % of High)



Host Memory	HIGH (900 MB for 28 GB + 1 % of all memory above 28 GB)	SOFT (64 % of High)	HARD (32% of High)	LOW (16% of High)
32 GB	941 MB	603 MB	302 MB	151 MB
48 GB	1,105 MB	707 MB	354 MB	177 MB
64 GB	1,269 MB	812 MB	406 Mb	203 MB
96 GB	1,596 MB	1,022 MB	511 MB	255 MB
128 GB	1,924 MB	1,231 MB	616 MB	308 MB
512 GB	5,856 MB	3,748 MB	1,874 MB	937 MB
1 TB	10,199 MB	6,528 MB	3,264 MB	1,631 MB
2 TB	20,685 MB	13,239 MB	6,620 MB	3,309 MB

Table 1: Above are few of the example of Memory Reclamation Levels of ESXi host

Proposed Methodology

The methodology for our performance comparison of hypervisors is to testing various virtualization solutions for different cloud workloads and technology such as qemu, lxc, hyper v for various cloud consumption like hosting, high performance computing, database hosting and various other use.

Experimental Setup: The purpose of our system setup is to provide complete fairness in all aspects that can affect system performance.

Hardware Setup: For fair comparison, hardware settings are completely identical for all supervisors using a single server machine (Dell R620 server), which has two processor 2* intel xeon2670(20 core) with RAM 128GB DDR3 and HDD 3*900GB SAS 10k.

Tools:

Datadog: is a cloud-based service that provides monitoring of servers, digital, tools and services through a SaaS-based data analytics platform. we are using Datadog to The analysis includes CPU utilization, Disk Usage, Disk Latency, Memory Breakdown and MySQL and Apache statistics. The load is at its peak from 4:00pm to 7:00pm.

Datadog uses a Go based agent, rewritten from scratch since its major version 6.0.0 released on February 28, 2018 It was formerly Python based, forked from the original created in 2009 by David Mytton for Server Density (previously called Boxed Ice). Its backend is built using a number of open and closed source technologies including D3, Apache Cassandra, Kafka, PostgreSQL, etc.

In 2014, Datadog support was broadened to multiple cloud service providers including Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform and Red Hat OpenShift. Today, the company supports over 350 integrations out-of-the-box.

Sysbench is an open source bench marking tool. It examines CPU usage related to the operating system, memory usage, Disk IO and MySQL performance. It is particularly important to use these computer parameters when starting a critical database.

Sysbench is one of a few **other Benchmarking utilities** that can be used to test out VPS and Cloud Server performance. Sysbench is one of the most common benchmarking utilities, so it's easy to find and compare performance results since a lot of people use Sysbench and similar tests.

Sysbench allows you to test out performance in the following areas:

- **CPU** -- This is a CPU performance test which uses a maximum prime number value that's defined when the test is run to determine how long it takes the system to calculate the numbers. You can specify how many threads to use for this test, making it a quick and easy way to test CPU performance for various amounts of threads.

- **OLTP** -- This is a Database / MySQL performance test which helps to estimate best case scenario MySQL performance. You can simulate read only workloads, by only using SELECT queries, or you can test mixed read and write environments. Almost anyone who writes about MySQL performance uses Sysbench OLTP for performance tests and results.

- **fileio** -- This is your disk IO test. Similar to FIO, but much less complex in terms of output and options. You can test sequential reads and writes, as well as random reads and writes. Sysbench fileio allows you to modify block size for IO, and the ability to toggle direct io, sync, async, and various other IO related functions.

Hypervisors we are using:

I. **KVM:** Kvm represents the latest generation of open source virtualization. The aim of the project was to create a modern hypervisor who builds on the experience of the previous generation. It simply makes the kernel into hypervisor when you install the kvm kernel module. However, because the standard Linux kernel is the supervisor, it enjoys the changes to the standard kernel (memory support, scheduler, and so on.

II. **HYPER-V** : Microsoft Hyper-V Microsoft Hyper-V is a hypervisor-based virtualization technology for x64 versions of Windows Server It exists in two versions: as a standalone product called Hyper-V Server and as a installable role / component in Windows Server [18]. There is no difference between MS Hyper-V in either of these two versions. The hypervisor is the same regardless of the installed version.

III. **HYPER CORE**: The uniqueness of the scale computing solution is the patented Hyper Core software. Hyper Core preloaded on all HC3 nodes, ready to deploy straight out of the box, well tuned for any type of HC3 node with nothing extra to license or install. Hyper Core continuously monitors all virtual machines, software, and hardware to automatically detect and respond to common infrastructure events, maintain application availability, and simplify data center management. Scale Computing HC3 and Hyper Core architecture have been designed to provide highly accessible, scalable computing and storage services while maintaining operational simplicity through smart software automation and architecture simplicity.

IV. **LXC**: Linux Containers (LXC) is a project aimed at building a toolkit for isolating processes from other processes. As mentioned, the cone is not developed as a security feature and can be escaped from the system - LXC tries to create environments (containers) that should be immune to its process and child processes and protect the system even if an attacker manages to escape the container. Besides, it provides an interface for reducing the resource consumption of delicate containers.

V. **QEMU**: QEMU is an open source machine virtualization emulator. Run programs for a different Linux / BSD purpose, in every supported architecture Run operating systems for every computer, in every supported architecture: It mimics the machine processor through dynamic binary translation and provides a set of different hardware models and devices for the machine, enabling it to run a variety of guest operating systems. It can also be used with KVM to run virtual machines at near-natural speed (by utilizing hardware extensions such as Intel VT-x). QEMU can also emulate user-level processes, enabling complex applications for a single architecture.

VI. **ESXI**: - VMware ESXi (formerly ESX) VMware vSphere is a software package that includes many software components such as vCenter, ESXi and the vSphere client [13]. VSphere is not a specific software that you can install and use, "it's just a package name that has other subcomponents." The hypervisor comes in the form of VMware ESXi, which is a type 1 hypervisor (bare metal). All virtual machines or guest operating systems are installed on the ESXi server. To install, manage, and access the virtual servers located above the ESXi server, a different part of a vSphere suit called a vSphere or vCenter client is needed.

Testing For More Efficient Containerization

Actual dataset for creating a new instance (Total size in GB vs. Number of Instances):

CentOS installation size:

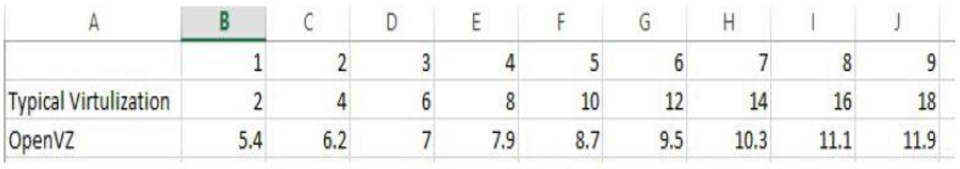
Initial Installation size: 1.3 GB

After critical update: 2.0 GB

OpenVZ sizing:

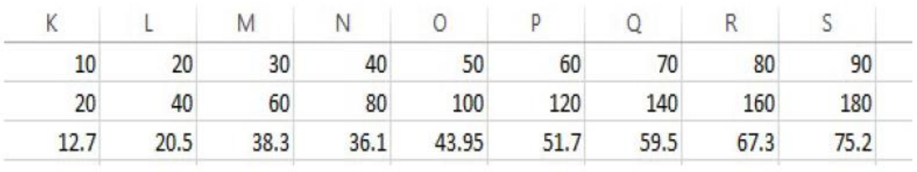
Initial installation size: 3.0 GB

After CentOS template: 3.2 GB



A	B	C	D	E	F	G	H	I	J
	1	2	3	4	5	6	7	8	9
Typical Virtulization	2	4	6	8	10	12	14	16	18
OpenVZ	5.4	6.2	7	7.9	8.7	9.5	10.3	11.1	11.9

Table 2: Dataset for creating a new instance.



K	L	M	N	O	P	Q	R	S
10	20	30	40	50	60	70	80	90
20	40	60	80	100	120	140	160	180
12.7	20.5	38.3	36.1	43.95	51.7	59.5	67.3	75.2

Table 3: Dataset for creating a new instance (contd.)

For High Performance Computing testing:

we are computing to "Initializing random number generator, limit to Prime numbers 200000" in 6 different hypervisors on same hardware one at a time to figure out which hypervisor is working more efficiently using sysbench to Benchmark Cpu performance.

Output / Result / Screenshot

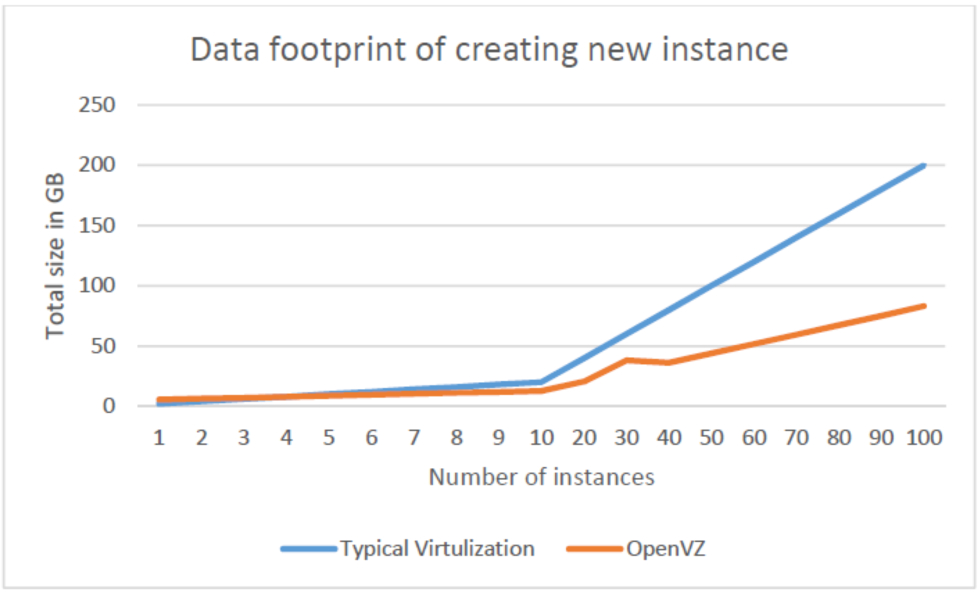


Figure 1: Data footprint of creating a new instance.

DATADOG RESULT

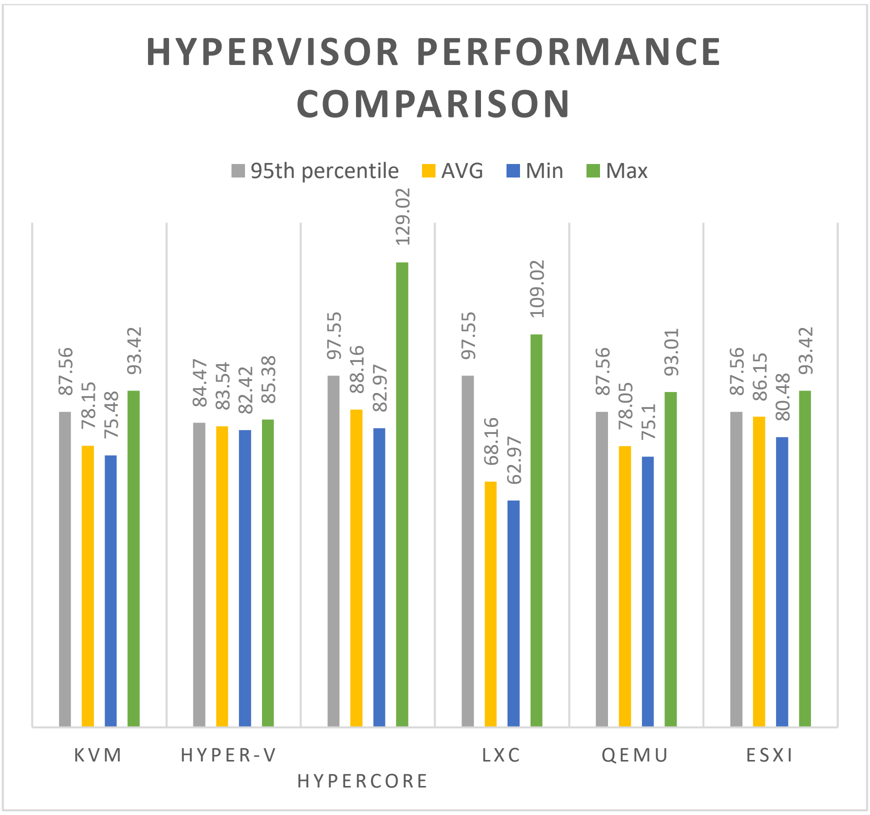


Figure 5: performance comparison chart.

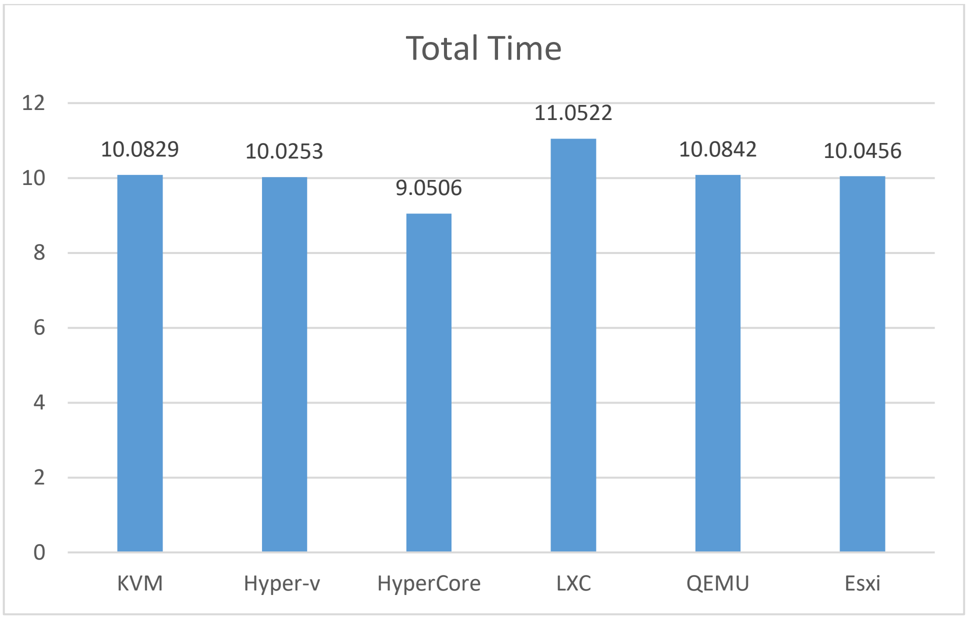


Figure 6: maximum time taken to complete the computing.

Conclusion:

For More Efficient Containerization

As we can see from the above graph in figure 1, the size for creation of a new instance is almost the same for traditional VM as well as OpenVZ at first. But after a while, when the number of instances increase significantly (in our case after 10 instances), the size of virtual instance increases too heavily compared to OpenVZ. This proves that for large number of instances and better utilization, Virtualization may not be the best choice. It could incur significant overhead when it comes to size. Containerization is a lightweight alternative to Virtualization.

Specific containers, code, applications or processes isolated. This gives everything in the container a neat envelope to manage, including moving it across different hosts. While you might think of a virtual machine deploying a server for multiple operating systems, containers run above the operating system, so unlike a VM, they do not require an operating system to boot when created. In essence, they can virtualize an operating system to provide an easier package of an app than a VM. We have so far dived a little into the advantages of using containers.

For High Performance Computing

After a thorough analysis of the six hypervisors, we came across some interesting results as shown in the graph above. HC3 Hypercore took the least amount of time (09.0506s) to generate random numbers up to the provided limit which was 2,00,000. We can conclude that it is the fastest. Second is Hyper-V with 10.0253s, followed by ESXi, KVM, Qemu and finally the slowest LXC.

In terms of performance latency, the following are the best and worst respectively:

LXC has the least minimum latency among all, while HC3 Hypercore has the greatest minimum latency among all. In terms of average, LXC has the least average while HC3 Hypercore has the most. Hyper-V beat LXC in terms of least maximum latency, while HC3 Hypercore has the maximum. At last, Hyper-V has the minimum 95th percentile while HC3 Hypercore and LXC have the maximum.

We can conclude that LXC followed by have the best performance in terms of latency. HC3 Hypercore while being the fastest of them all, has the maximum latency.

Hyper-V and ESXi are easy to use and install but it comes under Premium pricing, where as KVM and Qemu is open source but it is quite difficult to install. For web/cloud Hosting we don’t need computational power, what we need a system like Lxc which doesn’t need a VM dividing servers into more than one operating system. Also, an LXC provides Virtualization of an operating system which is so much lighter as compared to a Virtual Machine. HC3 Hypercore is a Premium pricing but it is quite optimized to do high power computing. In the end, it's a matter of preference. All major cloud providers use different kinds of hypervisors based on their needs.

REFERENCES

1. “History of Virtualization” by harrisdy (October 19, 2009)

   http://www.infobarrel.com/History_of_Virtualization

2. Containerization by Kiante Evans

   https://quizlet.com/322447201/containerization-flash-cards/

3. “What's the future of server virtualization” by Neal Weinberg (July 11, 2018)

   https://www.networkworld.com/article/3285906/whats-the-future-of-servervirtualization.html

4. “Virtualization: A look at the past, present and future of IT’s trusty workhorse” by Contel Bradford (May 29, 2019) https://blog.storagecraft.com/virtualizationa-look-at-the-past-present-and-future-of-its-trusty-workhorse/

5. “VMWare memory management part 1- Understanding ESXi host memory states” by Mohammed Raffic (December 22, 2014)

   http://www.vmwarearena.com/esxi-memory-management-part-1-understanding-esxi-host-memory-states/

6. https://wiki.openvz.org/Features

7. https://en.wikipedia.org/wiki/LXC

8. “Difference between OpenVZ and LXC” by Sentris Network LLC

   https://www.sentris.net/billing/knowledgebase/126/Difference-between-OpenVZ-and-LXC.html

9. “Introduction to containers: Concept, Pros and Cons, Orchestration, Docker and other Alternatives” by flow.ci (September 30, 2016)

   https://medium.com/flow-ci/introduction-to-containers-concept-pros-andcons-orchestration-docker-and-other-alternatives-9a2f1b61132c

10. “How Serverless is changing the technology stack” by Dan Njoku (February22, 2019)

    https://dannjoku.com/blogs/posts/HowServerlessIsChangingTheTechnologyStack.html

11. “Will containers kill the virtual machine?” by Brandon Butler

    https://www.networkworld.com/article/2910559/will-containers-kill-thevirtual-machine.html