

# A Comparative Analysis of Native OS And Guest OS Through VirtualBox for UBUNTU 18

S H I V A M   S A H   ( 1 7 B C E 2 3 8 6 )

Department of Computer Science  
SCHOOL OF SCOPE  
Vellore institute of technology  
Vellore, Tamilnadu ,India

*vit.ac.in*

## Abstract

*In todays world The winning development and readying of large-scale net services rely critically on performance. Even little regressions in interval will translate directly into important energy and user expertise prices. Despite the widespread use of distributed server infrastructure (e.g., in cloud computing and net services), there's very little analysis on the way to benchmark such systems to get valid and precise inferences with stripped knowledge assortment prices. properly A/B testing distributed net services will be amazingly tough as a result of interdependencies between user requests (e.g., fsearch results, social media streams, photos) and host servers violate assumptions needed by customary applied mathematics tests.*

*In this project we'll be examination benchmarking analysis of native and guest os through facilitate of virtual box which can facilitate peolpe to check and analyze capabilities of that system is best that purpose (for e.g. graphics,memory,file transfer speed, central processor abilities).*

**Keywords** *Virtualization, Virtual Machine, System Performance, Network, software, Overhead*

## 1. Introduction

Traditional personal computer systems (i.e., workstations) were designed for the use of a single OS. If this trend had continued until today, most of the components of such machines would have been left underutilized. Since the

evolution of cloud computing, systems virtualization techniques [1] have been established as one of the core substructures of cloud computing. Virtualization technology presents an abstraction between the bare computer (physical resource set) and the application running on top of it. The abstraction is called Hypervisor. The concept is built on what is called a Virtual Machine Monitor (VMM) or Virtual Machine (VM), which is a software representation of a physical machine (i.e., computer) that executes programs like a real machine. Durairaj and Kannan define Hypervisor as a piece of software that manages the sharing of hardware platform among multiple systems [2]. VMMs are categorized based on the virtualization process (i.e., Compute, Network, and Storage) (see Fig.1). Each of these virtualization sets presents some unique characteristics.

Virtualization techniques involve two independent nodes; these are the Host (i.e., primary system environment), which acts as the target of the virtualization, and the Guest (i.e., secondary environment), which acts as the source of virtualization. Ghosh et al. argue that virtualization involves the construction of an isomorphism from the guest state to the Host state [3]. Thus, the virtualization application requires dividing the computer hardware and software resources by emulating the Host hardware resources.

In being it computing, storage, or network, the principal function of virtualization is to decouple the Guest applications from the underlying Host system. Thus, virtualization aims at enhancing resource management through efficient resource allocation in a multi-tenanted environment. Some of the popular virtual machine monitors (hypervisors) are Xen, KVM, VMWare, and HyperV. Most VMs run modified Guest kernels that are virtualization-aware and use special hypercall APIs to interact with the Hypervisor directly. Para-virtualization of I/O operations decreases the number of transitions between the Guest VM and the hypervisor, resulting in performance gains. Apart from the gains, the paper

assesses other performance issues, which virtualization technique presents (i.e., either to the Host, the Guest, or both).

To explore this, we apply an experimental research design to examine the different methods, through which virtualization is supported in computer systems. The effect of various OS and VM configurations on system performance is also explored.

The rest of the paper is structured as follows: Section 2 provides an overall picture of the literature review and related work, while Section 3 briefly explains the applied methodology. In we present and discuss our results while Section 6 draws the conclusions, giving some pointers for future work.

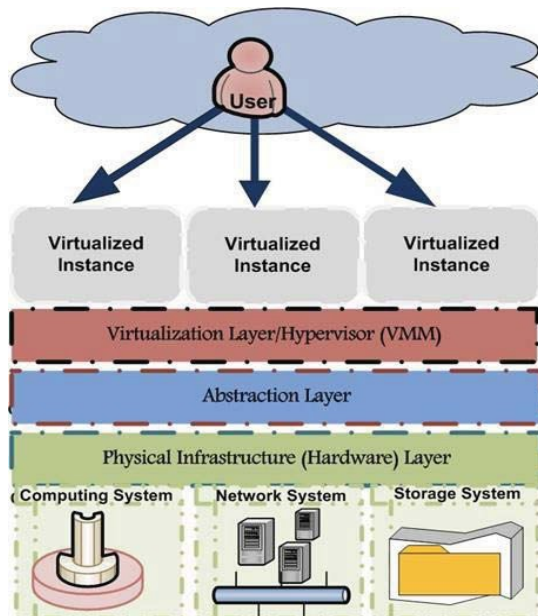


Figure1. 1: Hypervisor Architecture

## 2.1 Softwares used

**VIRTUALBOX** : Oracle VM VirtualBox (formerly Sun VirtualBox, Sun xVM VirtualBox and Innotek VirtualBox) could be a free and ASCII text file hosted hypervisor for x86 virtualization, developed by Oracle Corporation. Created by Innotek GmbH, it had been nonheritable by Sun Microsystems in 2008, which was, in turn, nonheritable by Oracle in 2010.

VirtualBox is also put in on Windows, macOS, Linux, Solaris and There are ports to FreeBSD and Genode. It supports the creation and management of guest virtual machines running Windows, Linux, BSD, OS/2, Solaris, Haiku, and OSx86, as well as restricted virtualization of macOS guests on Apple hardware. For some guest operative systems, a "Guest Additions" package of device drivers and system applications is on the market, which generally improves performance, particularly that of graphics.

### Emulated setting

#### Running Ubuntu Live CD under VirtualBox on Ubuntu

Users of VirtualBox will load multiple guest OSes beneath one host operating-system (host OS). every guest are often started, paused and stopped severally at intervals its own virtual machine (VM). The user will severally set up every VM and run it beneath a selection of software-based virtualization or hardware aided virtualization if the underlying host hardware supports this. The host OS and guest OSs and applications will communicate with one another through variety of mechanisms together with a typical writing board and a virtualized network facility. Guest VMs also can directly communicate with one

## 2.1 Software-based virtualization

In the absence of hardware-assisted virtualization, VirtualBox adopts a regular software-based virtualization approach. This mode supports 32-bit guest OSs that run in rings zero and three of the Intel ring design.

2.3 • The system reconfigures the guest OS code, which might usually run in ring zero, to execute in ring one on the host hardware. as a result of this code contains several privileged directions that cannot run natively in ring one, VirtualBox employs a Code Scanning and Analysis Manager (CSAM) to scan the ring zero code recursively before its 1st execution to spot problematic directions so calls the Patch Manager (PATM) to perform unchanged fix. This replaces the instruction with a jump to a VM-safe equivalent compiled code fragment in hypervisor memory.

2.4 • The guest user-mode code, running in ring three, usually runs directly on the host hardware in ring three.

2.5 In each cases, VirtualBox uses CSAM and PATM to examine and patch the offending directions whenever a fault happens. VirtualBox conjointly contains a dynamic recompiler, supported QEMU to recompile any real mode or protected mode code entirely (e.g. BIOS code, a DOS guest, or any software package startup).

2.6 Using these techniques, VirtualBox can do a performance love that of VMware.

## 2.7 Hardware-assisted virtualization

2.8 VirtualBox supports each Intel's VT-x and AMD's AMD-V hardware-assisted virtualization. creating use of those facilities, VirtualBox will run every guest VM in its own separate address-space; the guest OS ring zero code runs on the host at ring zero in VMX non-root mode instead of in ring one.

2.9 VirtualBox supports some guests (including 64-bit guests, SMP guests and bound proprietary OSs) solely on hosts with hardware-assisted virtualization.

## 2.10 Device virtualization

2.11 The system emulates exhausting disks in one amongst 3 disk image formats:

2.12.1 VDI: This format is that the VirtualBox-specific VirtualBox Disk Image and stores information in files bearing a ".vdi" extension.

2.13.2 VMDK: This open format is employed by VMware merchandise like VMware digital computer and VMware Player. It stores information in one or a lot of files bearing ".vmdk" name extensions. one virtual fixed disk could span many files.

2.14.3 VHD: This format is employed by Windows Virtual laptop and Hyper-V, and is that the native virtual disk format of the Microsoft Windows software package, beginning with Windows seven and Windows Server 2008 R2. information during this format area unit hold on in an exceedingly Indian file bearing the ".vhd" extension.

2.15 VirtualBox virtual machine will, therefore, use disks antecedently created in VMware or Microsoft Virtual laptop, also as its own native format. VirtualBox also can connect with iSCSI targets and to raw partitions on the host, victimization either as virtual exhausting disks. VirtualBox emulates IDE (PIIX4 and ICH6 controllers), SCSI, SATA (ICH8M controller) and SAS controllers to that exhausting drives are often hooked up.

2.16 VirtualBox has supported Open Virtualization Format (OVF) since version two.2.0 (April 2009)

2.17 Both ISO pictures and host-connected physical devices are often mounted as CD/DVD drives. for instance, the optical disc image of a Linux distribution are often downloaded and used directly by VirtualBox.

By default, VirtualBox provides graphics support through a custom virtual graphics-card that's VESA compatible. The Guest Additions for Windows, Linux, Solaris, OpenSolaris, or OS/2 guests embody a special video-driver that will increase video performance and includes extra options, like mechanically adjusting the guest resolution once resizing the VM window[34] or desktop composition via virtualized WDDM drivers .

For associate local area network network adapter, VirtualBox virtualizes these Network Interface Cards

- AMD PCnet PCI II (Am79C970A)
- MD PCnet-Fast III (Am79C973)
- Intel Pro/1000 MT Desktop (82540EM)
- Intel Pro/1000 MT Server (82545EM)
- Intel Pro/1000 T Server (82543GC)
- Paravirtualized network adapter (virtio-net)

The emulated network cards permit most guest OSs to run while not the requirement to seek out and install drivers for networking hardware as they're shipped as a part of the guest OS. A special paravirtualized network adapter is additionally obtainable, that improves network performance by eliminating the requirement to match a selected hardware interface, however needs special driver support within the guest. (Many distributions of Linux hip with this driver enclosed.) By default, VirtualBox uses NAT through that web code for end-users like Firefox or ssh will operate. Bridged networking via a number network adapter or virtual networks between guests also can be organized. Up to thirty six network adapters are often hooked up at the same time, however solely four area unit configurable through the graphical interface.

**2.18**For a sound card, VirtualBox virtualizes Intel HD Audio, Intel ICH AC'97 and SoundBlaster sixteen devices.

**2.19**A USB 1.1 controller is emulated so any USB devices hooked up to the host is seen within the guest. The proprietary extension pack adds a USB two.0 orG

USB 3.0 controllers and, if VirtualBox acts as an RDP server, it can also use USB devices on the remote RDP client as if they were connected to the host, although only if the client supports this VirtualBox-specific extension (Oracle provides clients for Solaris, Linux and [Sun Ray](#) thin clients that can do this, and have promised support for other platforms in future versions).

Feature set

- 64-bit guests (hardware virtualization support is required)
- Snapshots
- Seamless mode — the flexibility to run virtualized applications aspect by aspect with traditional desktop applications
- Shared writing board
- Shared folders
- Special drivers and utilities to facilitate switch between systems
- instruction interaction (in addition to the GUI)
- Public API (Java, Python, SOAP, XPCOM) to manage VM configuration and execution
- Nested paging for AMD-V and Intel American state (only for processors supporting SLAT and with SLAT enabled)
- restricted support for threeD graphics acceleration (including OpenGL up to (but not including) 3.0 and Direct3D nine.0c via Wine's Direct3D to OpenGL translation)
- SMP support (up to thirty two virtual CPUs per virtual machine), since version three.0
- conveyance (aka Live Migration)
- 2nd video output acceleration (not to be mistaken with video cryptography acceleration), since version three.1
- EFI has been supported since version three.1 (Windows seven guests aren't supported)

• Storage emulation options

- NCQ support for SATA, interface and SAS raw disks and partitions
- SATA disk hotplugging
- Pass-through mode for solid-state drives
- Pass-through mode for CD/DVD/BD drives — permits users to play audio CDs, burn optical disks, and play encrypted videodisk discs

- will disable host OS I/O cache
- permits limitation of IO information measure
- PATA, SATA, SCSI, SAS, iSCSI, disc controllers
- VM disk image encoding victimization AES128/AES256
- Storage support
- Raw disk access — permits physical disk partitions on the host system to seem within the guest system
- Vmware Virtual Machine Disk (VMDK) format support — permits exchange of disk pictures with VMware
- Microsoft VHD support
- QEMU quantum field theory and qcow disks
- HDD format disks (only version 2; versions three and four aren't supported) utilized by Parallels virtualization merchandise

## 2.2BENCHMARKS USED:

A.PHORONIX check SUITE: Phoronix check Suite (PTS) could be a free and ASCII text file benchmark software system for in operation system} and different operating systems that is developed by Michael Larabel and Matthew Tippet. The Phoronix check Suite has been supported by sites like UNIX system.com, LinuxPlanet and has been known as "the best benchmarking platform" by Softpedia. The Phoronix check Suite is additionally utilized by Tom's Hardware,

### •Features

- Supports over 220 check profiles and over sixty check suites;
- Uses associate degree XML-based testing design. Tests embrace Mencoder, Ffmpeg and lumen sensors along with OpenGL games like Doom three, Nexuiz, and Enemy Territory: Quake Wars, and plenty of others;
- Contains a feature known as PTS international wherever users ar ready to transfer their check results and system data for sharing. Then through death penalty one command, different users will compare their check results to a particular system in associate degree easy-comparison mode;
- enable report benchmark results to the Phoronix international on-line database;
- enable compare results side-by-side;
- Is protractible and new tests may be else easily;
- will do anonymous usage reporting;
- will do automated Git bisecting on a performance basis to search out performance regressions. It features statistical significance verification.

### •Components

- Phoromatic
- Phoromatic is associate degree web-based remote check management system for the Phoronix check Suite. It will automatic programming of tests. it's aimed toward the enterprise. It will manage multiple check nodes at the same time at intervals a check farm or distributed setting.
- Phoromatic hunter
- Phoromatic hunter is associate degree extension of Phoromatic that gives a public interface into check farms. presently their reference implementations autonomously monitor the performance of the Linux kernel on a everyday, Fedora Rawhide and Ubuntu.
- PTS Desktop Live[
- PTS Desktop Live was a stripped down x86-64 Linux distribution, that included Phoronix check Suite a pair of.4. it



absolutely was designed for testing/benchmarking computers from a LiveDVD / LiveUSB setting.

- Phodevi

- Phodevi (Phoronix Device Interface) is a library that provides a clean, stable, platform-independent API for accessing software system and hardware data.

- PCQS

- Phoronix Certification & Qualification Suite (PCQS) could be a reference specification for the Phoronix check Suite.

## B. SYSBENCH

- Sysbench is one in every of a few other Benchmarking utilities that may be accustomed check out VPS and Cloud Server performance. Sysbench is one in every of the foremost common benchmarking utilities, thus it is easy to search out and compare performance results since tons of individuals use Sysbench and similar tests.

- Sysbench permits you to check out performance within the following areas:

- • C.P.U. -- this is often a CPU performance check that uses a most prime quantity worth that is outlined once the check is run to see however long it takes the system to calculate the numbers. you'll specify what number threads to use for this check, creating it a fast and straightforward thanks to check C.P.U. performance for numerous amounts of threads.

- • OLTP -- this is often a info / MySQL performance check that helps to estimate best case scenario MySQL performance. you'll simulate scan solely workloads, by solely victimization choose queries, otherwise you will check mixed scan and write environments. virtually anyone WHO writes regarding MySQL performance uses Sysbench OLTP for performance tests and results.

- • fileio -- this is often your disk IO check. the same as FIO, however abundant less advanced in terms of output and choices. you'll check sequent reads and writes, yet as random reads and writes. Sysbench fileio permits you to change block size for IO, and therefore the ability to toggle direct io, sync, async, and numerous different IO connected functions.

- This check mode may be accustomed manufacture numerous styles of file I/O workloads. At the prepare stage SysBench creates a such that range of files with a such that total size, then at the run stage, every thread performs such that I/O operations on this set of files. once the worldwide --validate possibility is employed with the fileio check mode, SysBench performs checksums validation on all information scan from the disk. On every write operation the block is full of random values, then the substantiation is calculated and hold on within the block together with the offset of this block at intervals a file. On every scan operation the block is valid by comparison the hold on offset with the important offset, and checksum with the real calculated checksum.

## 2.2 Performance (Virtualization) Evaluation Approach

Performance (Virtualization) analysis Approach Virtualization technology has been wide employed in server consolidation, superior computing, flexibility, dynamic, and ascendable computing environments. for example, Yiching Liou and babyminder subgenus Chen [8] [6] argue that virtualization techniques lead to improved IT infrastructure and services. In their study, they known and mentioned numerous impacts of virtualization [8]. Among them ar effective management of virtualized cloud environments, introduces new and distinctive challenges, like economical CPUs programing for virtual machines, effective allocation to handle each hardware intensive

## C.System Profile and Benchmarking

Is a system profiler and benchmark for UNIX operating system systems, that gathers info concerning the is given on AN easy and intuitive interface.

info is classified in four main groups: laptop, Devices, Network, and Benchmarks.

The Computer cluster contains principally info concerning basic code. This includes however isn't restricted to info concerning the operative system (kernel version, C library, distribution, etc.), kernel modules, venue info, filesystem usage, users/groups,

and development tools. The Devices cluster contains info concerning the hardware, as detected by the operative system.

Hardinfo is incapable of detective work hardware by itself, it simply lists what the

operating system was ready to realize. This includes, however will not limit to, info

concerning the processor (including cache layout, known bugs, and have flags with their explanation), devices (such as USB, PCI, and Input devices), among alternative things.

The Network cluster lists info concerning network interfaces, routing table, DNS servers, and alternative networking-related topics. The Benchmarks cluster permit activity easy benchmarks to check CPU and FPU capabilities, as well as a number of the graphical user interface capabilities as well. In previous versions, it was potential to send the benchmark results to a central server, so it was

potential to match the benchmarks with alternative Hardinfo users; this service has been deactivated within the unit of time. Reports may be generated either by invoking Hardinfo with special command-line parameters (see below), or by clicking the "Generate Report" button in the interface. Reports will be customized in the GUI and saved in either HTML or plain text formats.

HardInfo comes with the subsequent benchmark tests:

1. electronic equipment Blowfish
2. electronic equipment CryptoHash
3. electronic equipment Fibonacci
4. electronic equipment N-Queens
5. FPU FFT
6. FPU Raytracing
7. CPU ZLIB
8. GPU RAYTRACING

Highlighting any take a look at so choosing 'refresh' from the highest toolbar can rerun any antecedently completed benchmarks. Generate Reports Simply choose 'Generate Report' from the highest toolbar, a dialogue box can seem permitting you to export all desired laptop, device, network or benchmark results to a hypertext mark-up language file. Upon completion associate degree choice to open the file directly for viewing is provided.

and I/O intensive workloads. during a connected study, Christophe Pelletineas [9] claimed, virtualization on UNIX operating system was much better than a similar technology on Windows. Full virtualization may be a sort that possesses the foremost overhead as a result of it needs the employment of a full trap-and-emulate theme to be able to virtualize a machine. Jianhua et al. [10] compared the performance of VMs with the hardware, RAM, and Disk I/O performance of 2 totally different VMs. LINPACK benchmark was accustomed judge hardware performance, LMBench for Ram and IOZone for disk I/O. it absolutely was additionally ascertained in their study that hardware performance of the Guest OS running VM was nearly a similar as that of the Host OS. Their study was restricted to

observe and compare Another study by Eric Van Hensbergen [11] mentioned the impact of numerous virtualization technologies and techniques. Similarly, Mino Rf and Dugki Min [12] have additionally looked at the impact of virtual machine overhead on a virtualized computing atmosphere. From another purpose of read, Damola and Johnsson [13], studied Network Performance Monitor for Virtual Machines and ended that Network communication between a virtual machine on prime of a hypervisor and an out of doors consumer do expertise congestion It is additionally argued that, within the virtualized atmosphere, congestion and network performance degradation, might originate from either real congestion on the physical network outside the physical pc or from the poor resource allocation or sharing of the common resources inside the physical pc. mistreatment experimental style study, [14] and [15] investigated the performance of multiple VMs (both on Host and Guest machines). The result was inconclusive relating to that VM had higher hardware performance compared to its Host. different connected works embrace Kejiang Ye et al. [10] UN agency applied a combinative analysis methodology to research performance from one VM to multiple VMs. during a similar manner of thinking, Praveen and Vijayrajan [16] propose a theme that manages the performance of VMs Hosting computing machines. what is more, Jensen D. E [17] examined the interference between numerous Guest machines once a lot of VMs planning a framework to work out if degradation is

going on from Associate in Nursing external virtualization layer or not. in addition, a study conducted by [18] and [19] centered on determinant the variations in performance overhead and examination the performance delivered to the VM of 4 virtualization solutions for the x86 design. Having a completely totally different approach, Pitropakis et al. [20][21] studied the role of malicious insiders in virtualized environments through the mirrored system calls. in line with Yeten et al. [22], the experimental style methodology is another to ancient sensitivity analysis. the essential plan behind this system is to vary multiple parameters at a similar time so most abstract thought is earned. during a connected study, Yeten et al. argue that when the suitable style is established and also the corresponding experiments (simulations) ar performed, the results is investigated by fitting them to a response surface. Experimental style methodology has been applied in numerous disciplines [23], [24], [25], [26], [27], [28], [29] and [30], however, its application in systems virtualization is comparatively novel, a gap, this study tries to fill. Throughout the literature, a good range of studies have examined the performance of virtual machines on each the Host and Guest machines. However, the dearth of in-depth understanding of the performance factors that impact the potency and effectiveness of the virtual machines necessitates more analysis on the topic. Taking into thought the last reality, our work Associate in Nursing experimental study on the performance of Guest OS and Host OS during a virtualized atmosphere consisting of various OS setups.

fig 3.2

### 3.1 Sysbench comparision between Guest OS and Native OS

**1.CPU** – This is a hardware performance take a look at that uses a most prime quantity price that is outlined once the take a look at is run to work out however long it takes the system to calculate the numbers. you'll be able to specify what percentage threads to use for this take a look at, creating it a fast and simple thanks to take a look at hardware performance for varied amounts of threads.command is below:

**sysbench --test=cpu --cpu-max-prime=20000 run**

**for native os:**

**Cpu speed: events per second= 362.46**

**total time:18.0021s**

**total no. of events=3626**

**Latency:**

**min=2.58**

**avg=2.76**

**max=3.62**

**95<sup>th</sup> percentile=3.62**

**sum=9997.37**

**Threads fairness:**

**events(avg/stddev)=3626**

**execution time=9.9974/8**

## 3. Experiments and Results

### General Information about Native OS

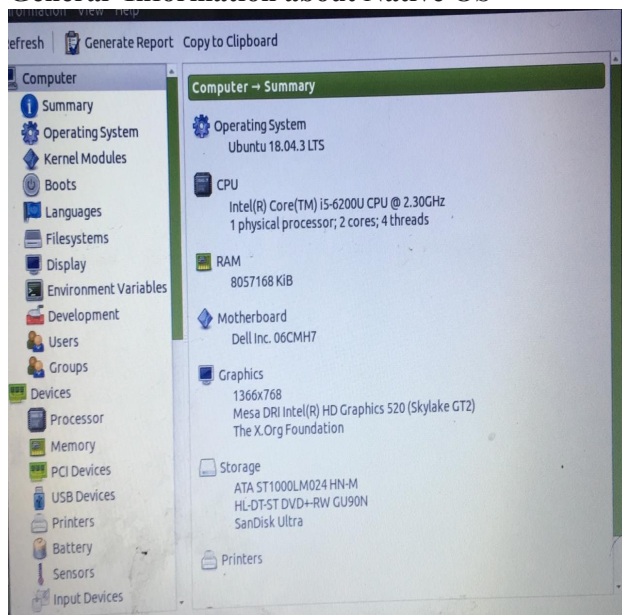
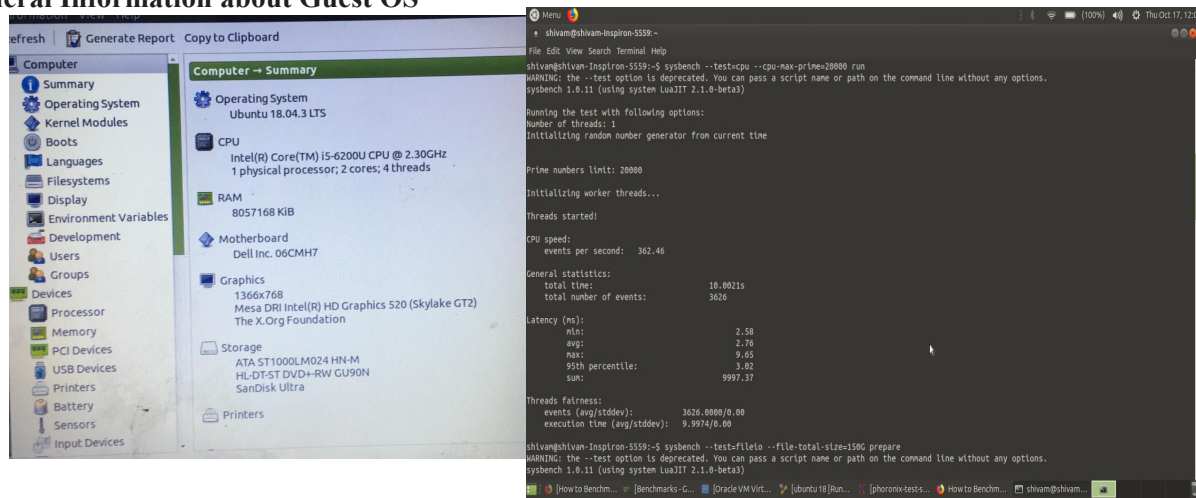


fig 3.1

### General Information about Guest OS



### for guest os:

cpu speed: events per second= 371.72

total time:10s

total no. of events=3718

### Latency:

min=2.58

avg=2.69

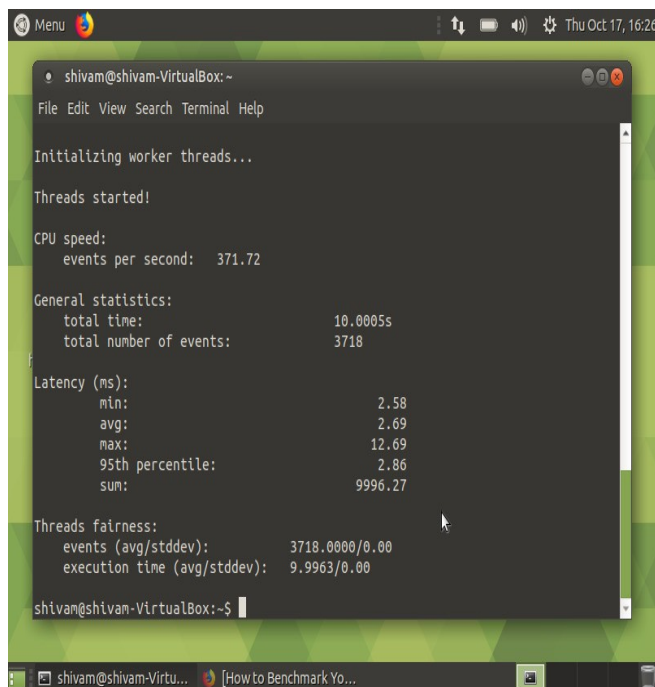
max=12.69

95<sup>th</sup> percentile=2.86

sum=9996.27

### Threads fairness:

events(avg/stddev)=3718



```
shivam@shivam-VirtualBox:~  
File Edit View Search Terminal Help  
  
Initializing worker threads...  
Threads started!  
  
CPU speed:  
  events per second:   371.72  
  
General statistics:  
  total time:          10.0005s  
  total number of events: 3718  
  
Latency (ms):  
  min:                 2.58  
  avg:                 2.69  
  max:                 12.69  
  95th percentile:    2.86  
  sum:                 9996.27  
  
Threads fairness:  
  events (avg/stddev): 3718.0000/0.00  
  execution time (avg/stddev): 9.9963/0.00  
  
shivam@shivam-VirtualBox:~$
```

execution time=9.9963/0.00

## b. Fileio

This check mode are going to be accustomed manufacture various types of file I/O workloads. At the prepare stage SysBench creates a like form of files with a like total size, then at the run stage, each thread performs like I/O operations on this set of files. cleanup inside the on high of example the first command creates 128 files with the complete size of 3 GB inside the present directory, the second command runs the actual benchmark and displays the results upon completion, and additionally the third one removes the files used for the take a glance at

```
$ sysbench --num-threads=16 --test=fileio --file-total-size=3G --file-test-mode=rndrw prepare
```

```
$ sysbench --num-threads=16 --test=fileio --file-total-size=3G --file-test-mode=rndrw run  
$ sysbench --num-threads=16 --test=fileio --file-total-size=3G --file-test-mode=rndrw
```

### for nativeos:

#### File operations:

read/s=66.50

write/s=43.33

fsyncs/s=139.68

#### Throughput:

read,Mib/s=1.04

wriiten,Mib/s=0.68

#### General statics:

total time=10.0068s

total no. of events=2499

#### Latency(ms)

min:0.01

avg: 64.05

max:471.33

95<sup>th</sup> percentile:211.60

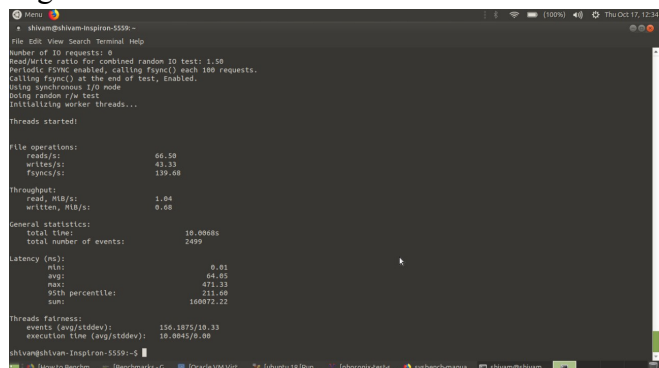
sum:160072.22

#### Threads fairness:

events(avg/stddev)=156.1875/10.33

execution time(avg/stddev)=10.0045/0.00

fig 3.5



```
shivam@shivam-Inspiron-5590:~  
File Edit View Search Terminal Help  
  
Initializing worker threads...  
Threads started!  
  
File operations:  
  reads/s:             66.50  
  writes/s:            43.33  
  fsyncs/s:            139.68  
  
Throughput:  
  read, Mib/s:          1.04  
  written, Mib/s:        0.68  
  
General statistics:  
  total time:           10.0068s  
  total number of events: 2499  
  
Latency (ms):  
  min:                  0.01  
  avg:                  64.05  
  max:                  471.33  
  95th percentile:    211.60  
  sum:                 160072.22  
  
Threads fairness:  
  events (avg/stddev): 156.1875/10.33  
  execution time (avg/stddev): 10.0045/0.00  
  
shivam@shivam-Inspiron-5590:~$
```

#### File operations:

read/s=17.49

write/s=11.51

fsyncs/s=35.35

#### Throughput:

read,Mib/s=0.27

wriiten,Mib/s=0.18

#### General statics:

total time=10.8259s

total no. of events=699

#### Latency(ms)

min:0.00

avg: 234.84

max:1702.19

95<sup>th</sup> percentile:773.68

sum:164150.94



Threads fairness:  
events(avg/stddev)=43.6875/6.56  
execution time(avg/stddev)=10.2594/0.23

```
Menu
cvsbench-manual.pdf - Mozilla Firefox
shivam@shivam-VirtualBox:~
File Edit View Search Terminal Help
shivam@shivam-VirtualBox:~$ sysbench --num-threads=16 --test=fileio --file-total-size=3G --file-test-mode=rndrw run
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
WARNING: --num-threads is deprecated, use --threads instead
sysbench 1.0.11 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 16
Initializing random number generator from current time

Extra file open flags: 0
128 files, 24MiB each
3GiB total file size
Block size 16KiB
Number of IO requests: 0
Read/Write ratio for combined random IO test: 1.50
Periodic fsync enabled, calling fsync() each 100 requests.
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing random r/w test
Initializing worker threads...

the prepare stage the following table is created in the specified database
(please find details)
```

guest os: fig 3.6

### 3.2. Phoronix Test Suite Test Comparison for Game Graphic

oa1920by1080	
PHORONIX-TEST-SUITE.COM	Phoronix Test Suite 9.0.1
Intel Core i5-6200U @ 2.80GHz (2 Cores / 4 Threads)	Processor
Dell 06CMH7 (1.0.1 BIOS)	Motherboard
Intel Xeon E3-1200 v5/E3-1500	Chipset
8192MB	Memory
1000GB Seagate ST1000LM024 HN-M	Disk
Intel HD 520 3GB (1000MHz)	Graphics
Realtek ALC3234	Audio
Realtek RTL810xE PCI + Intel 3160	Network
Ubuntu 18.04	OS
5.0.0-31-generic (x86_64)	Kernel
MATE 1.20.1	Desktop
X Server 1.20.4	Display Server
modesetting 1.20.4	Display Driver
4.5 Mesa 19.0.2	OpenGL
GCC 7.4.0	Compiler
ext4	File System
1366x768	Screen Resolution
- Scaling Governor: intel_pstate powersave - l1tf: Mitigation of PTE Inversion; VMX: conditional cache flushes SMT vulnerable + mds: Mitigation of Clear buffers; SMT vulnerable + meltdown: Mitigation of PTI + spec_store_bypass: Mitigation of SSB disabled via prctl and seccomp + spectre_v1: Mitigation of usercopy/swapgs barriers and __user pointer sanitization + spectre_v2: Mitigation of Full generic retpoline IBPB: conditional IBRS_FW STIBP: conditional RSIB filling	

In this take a look at we have a tendency to square measure aiming to benchmark game known as openarena and comapre its graphics through phoronix take a look at suite as game is put in and also the command as: phoronix-test-suite benchmark openarena and press enter. .

For native os:fig 3.7

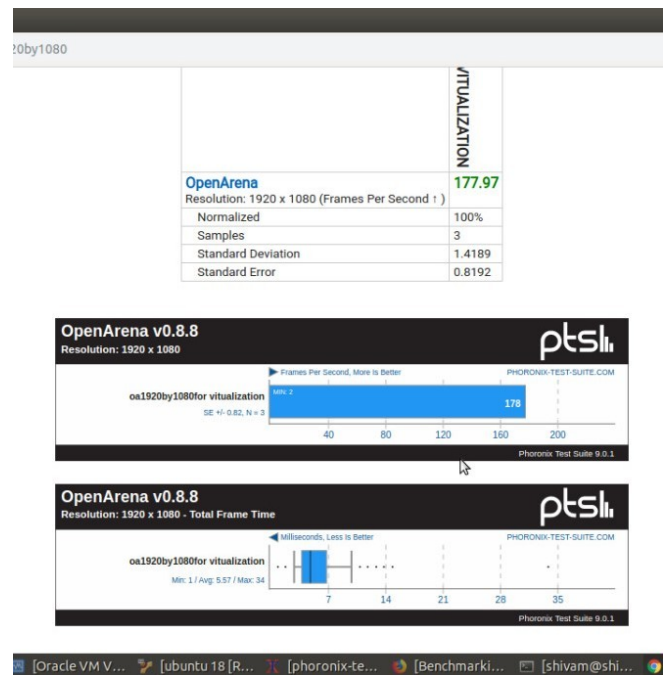


fig 3.8 result is 178 fps in 1920\*1080 resolution

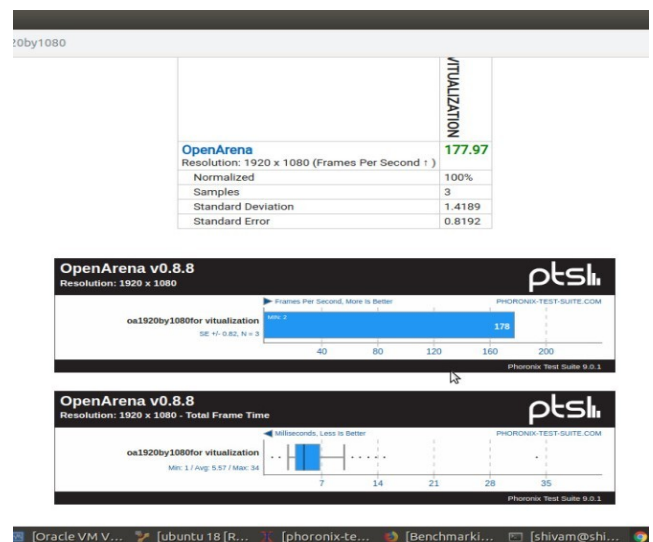


fig 3.9 for guest os:

```
phoronix-test-suite
Run A Test
Run A Suite [A Collection Of Tests]
Run Complex System Test
Show System Hardware / Software Information
Show Available System Sensors
List Available Tests
List Recommended Tests
Set Test Run Repetition
Search Tests / Suites / Results
Exit
ect Task:
Run A Test
Run A Suite [A Collection Of Tests]
Run Complex System Test
Show System Hardware / Software Information
Show Available System Sensors
List Available Tests
List Recommended Tests
Set Test Run Repetition
Search Tests / Suites / Results
Exit
ect Task: █
```

fig 3.10

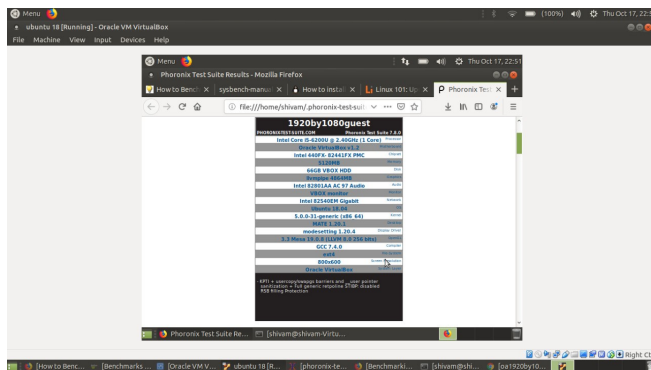
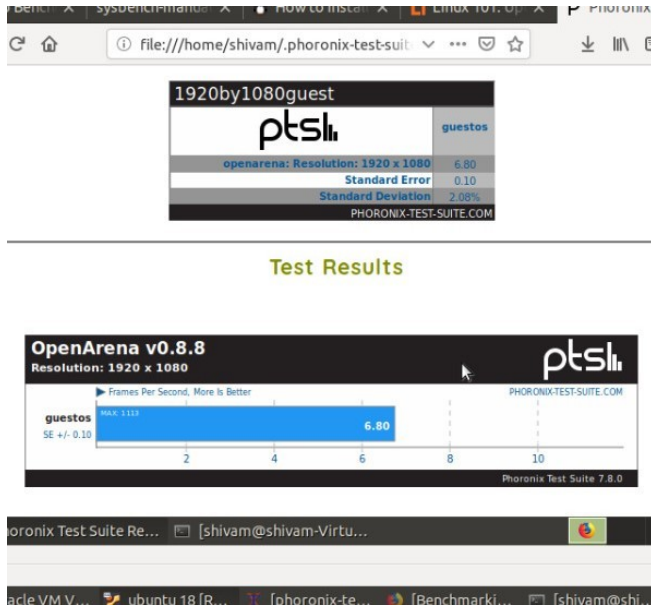


fig 3.11 FOR GUEST OS: result is 6.80fps for 1920\*1080 resolution

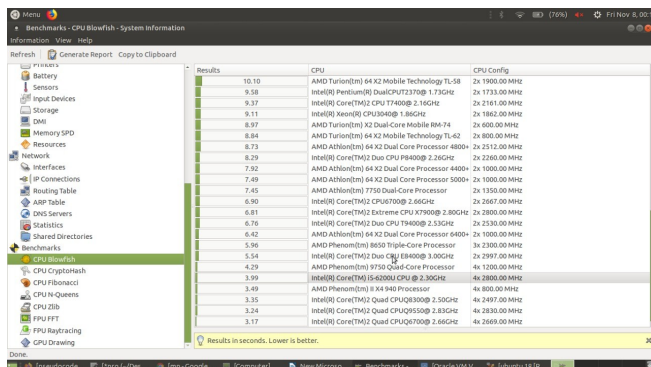


### 3.3 System Profile and Benchmarking Native and Guest OS FOR HARDWARE COMPARISONS

#### FOR NATIVE OS:

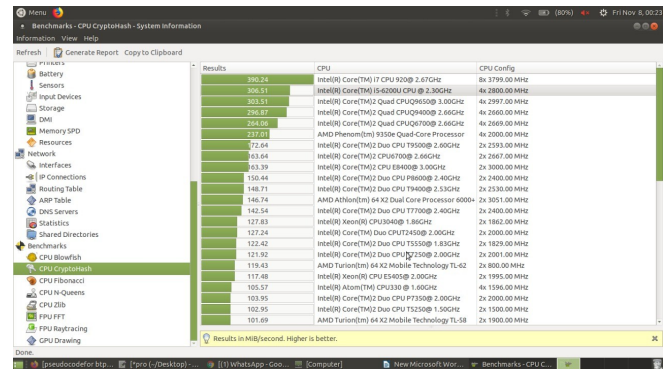
#### 1.CPU Blowfish (lower is better)

3.99 value at native os



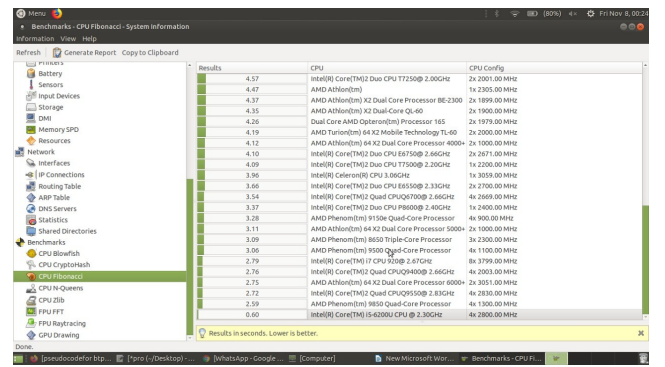
#### 2.CPU CryptoHash ( higher is better)

306.51 value at native os



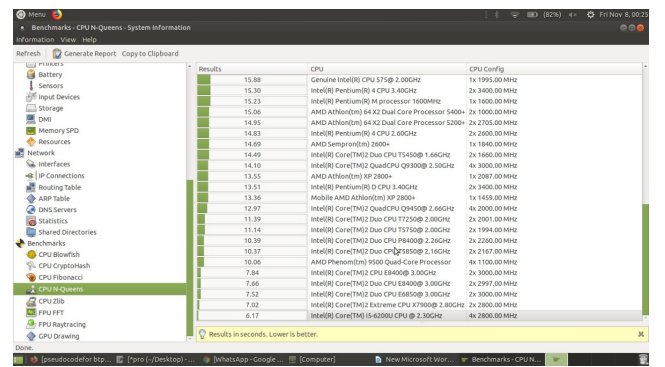
#### 3.CPU Fibonacci(lower is better)

0.60



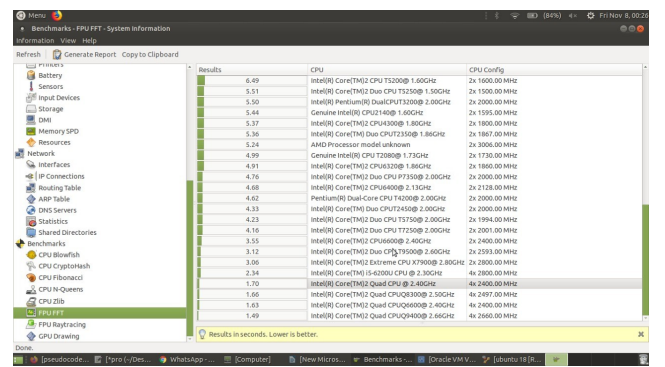
#### 4.CPU N-Queens (lower is better)

6.17



#### 5.FPU FFT(lower is better)

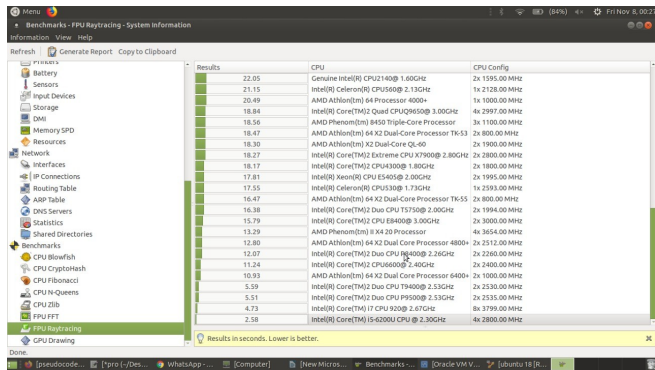
1.70





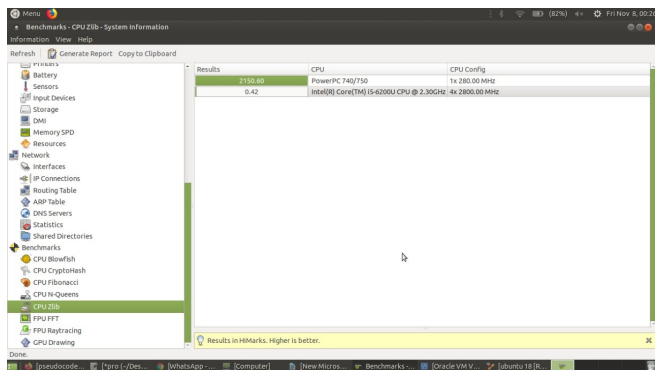
### 6.FPU Raytracing(lower is better)

2.58



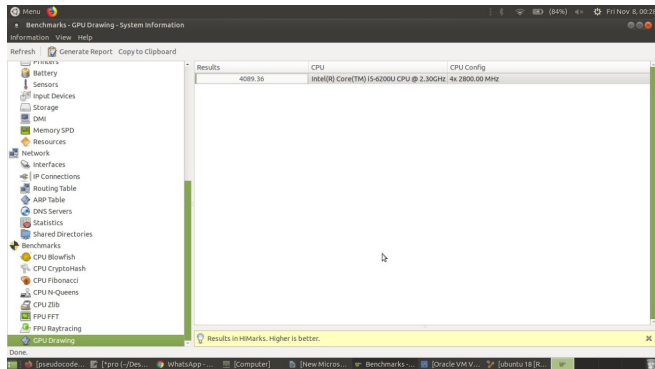
## 7.CPU ZLIB (HIGHER is better)

0.42



### 8.GPU DRAWING(HIGHER is better))

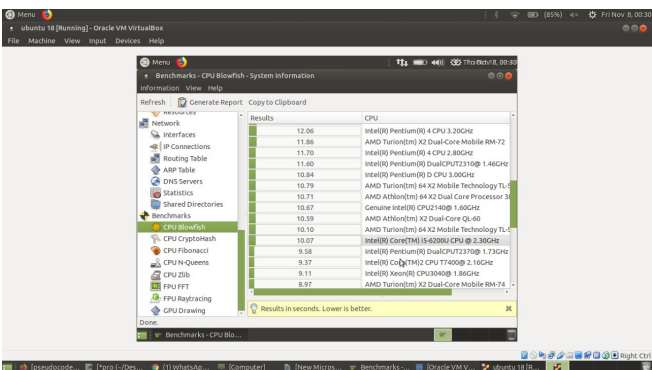
4089.6



**FOR GUEST OS:**

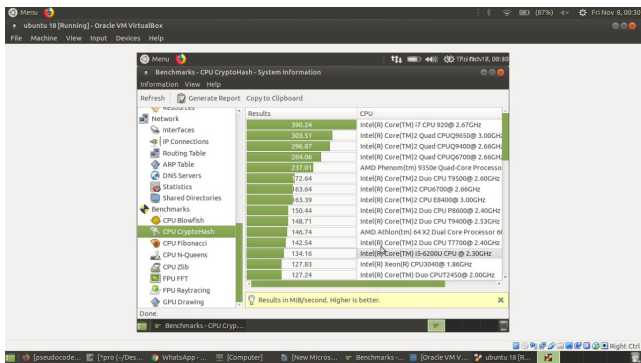
### 1.CPU BLOWFISH:( lower is bettr)

**10.07** value at guest os



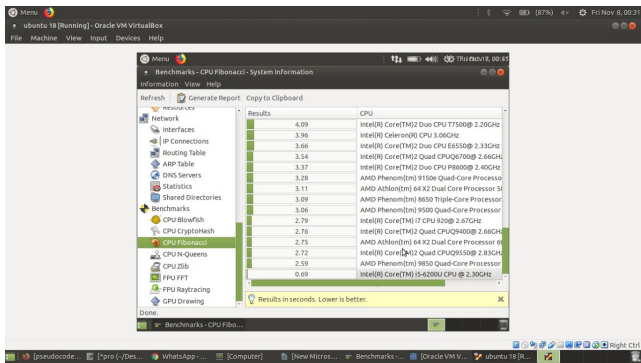
## 2.CPU CRYPTOHASH( higher is better)

134.16



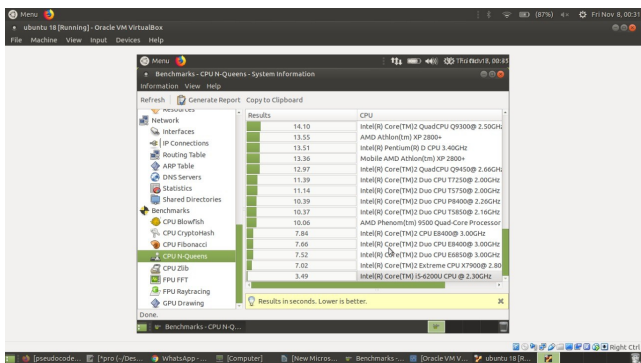
### 3.CPU FIBONACCI(lower is better)

0.69



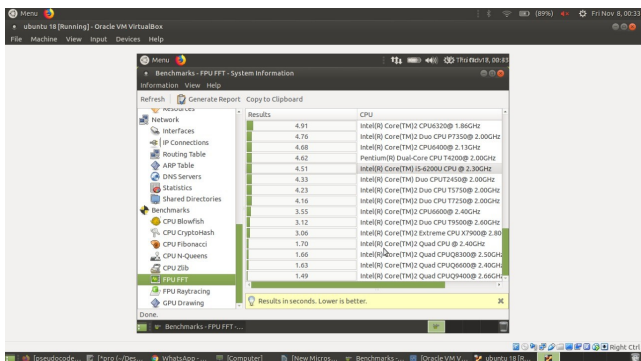
#### 4.CPU N-QUEENS(lower is better)

3.49



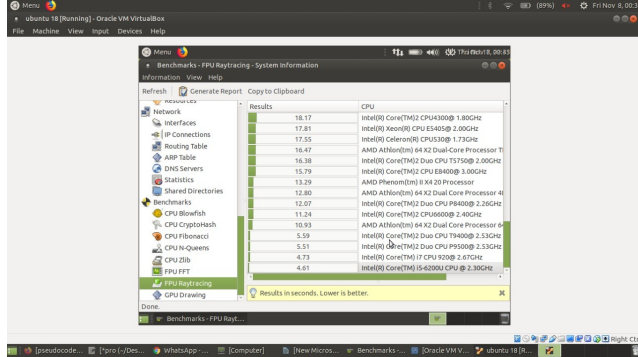
### 5.FPU FFT(lower is better)

4.51



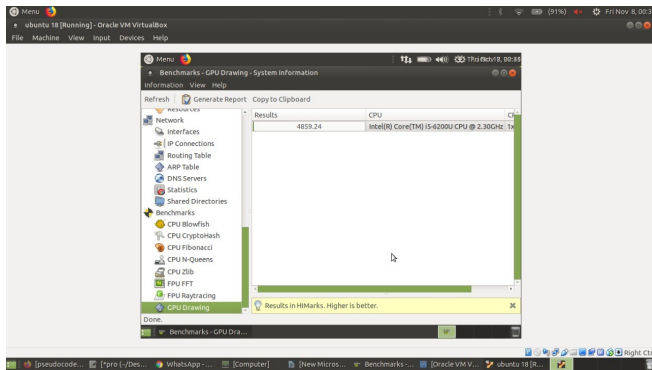
## 6.FPU RAYTRACING(lower is better)

4.61

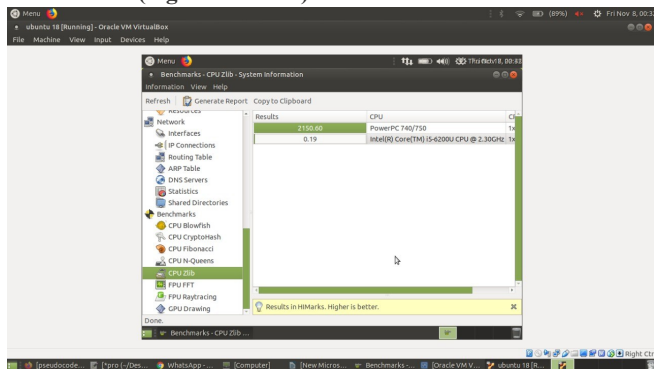


## 7.GPU DRAWING(higher is better)

4859.24



## 8.CPU ZLIB(higher is better) is 0.19



Sn.	BENCHMARKS	GUEST OS (G)	NATIVE OS (N)	COMPARISON acc. To their properties
1.	CPU blowfish	10.07	3.99	N is better
2..	CPU Cryptohash	134.16	306.51	N is better
3.	CPU Fibonacci	0.69	0.60	G is better
4.	Cpu Zlib	0.19	0.42	N is better
5.	CPU N-Queens	3.49	6.17	G is better
6.	FPU FFT	4.51	1.70	N is better
7.	FPU Raytracing	4.61	2.58	N is better
8.	GPU Drawing	4859.24	4089.6	G is better

## 4.DISCUSSION:

As you can see through the different comparison s between guest os and native os,firstly, considering

## REFERENCES

the general system information of ubuntu 18 on virtual box and native os , we would know why the virtual os has less capability in cpu speed, file speed, frame per second that is to calculate graphics, like in while we are using sysbench

we get to know cpu sped events per second is 362.46

in native os when no. of event is 3626 and cpu speed event per second is 371.2 in virtual box and vent no. is 3718 while it shows virtual os has more cpu speed,we see difference in file operations

which is better in native os read\s=66.50,

write\s =43.33, fsync\s =139.68 while read\s=17,

write\s=,11fsync\s=35 that means file operations are better in

native os than guest os . As we go further we see in when

phoronix test suite is used for calculating graphics for game called openarena

we see graphics is much better in native os having almost 300 fps

for 1920 \*1080 resolution and but in guest os it is only

6.8 fps in 1920\*1080. That is real low graphics for any game.

Now while we use system profile and benchmarking we see

various comparisons of hardware capability in guest os capability to other systems and in native os also.

## 5.Conclusion

As we learned through various comparisons between guest os and native os we come to understand that both are necessary in industry as we see one 's advantage overcome other's disadvantage like cpu speed is faster in guest os so we will use as it is purposed efficiently and als we saw through system profile and benchmarking ,there are many result which says how in guest os the hardware is better and as for other's disadvantages we see other hardwares a

how they are compared in native os which in some part better and some in not

to conclude ,there will be vry much importance in future for vms and to know where to work most efficiently through comparison like we did here will be the most factor to be efficient and succesful.

[1] <https://imysql.com/wp->

[content/uploads/2014/10/sysbench-manual.pdf](#)

[3] [https://en.wikipedia.org/wiki/Phoronix\\_Test\\_Suite](https://en.wikipedia.org/wiki/Phoronix_Test_Suite)

[4] <https://www.guru99.com/cloud-computing-for-beginners.html>

[5] <https://en.wikipedia.org/wiki/VirtualBox>

[2] <https://wiki.mikejung.biz/Sysbench>