ESXi and Proxmox: FileSystem Performance Comparison for Type-1 Hypervisors

Borislav Đorđević, *Member, IEEE*, Valentina Timčenko, *Member, IEEE,* Nenad Nedeljković, Nikola Davidović

***Abstract* — This paper presents the comparison of two representatives of type 1 hypervisors: Proxmox VE and VMware ESXi. Hypervisor acts like a lightweight operating system and runs directly on the host's hardware. The measurements are carried out on the same server and under the equivalent conditions, with the Linux Ubuntu 20.10 as the guest operating system using the Filebench 1.5-alpha1 software. The goal of this paper is to show an impact of different number of virtual machines on the performances of various file system and highlight the best combination. The results have been illustrated in graphical form.**

Keywords* — *Virtualization; Hypervisor; Proxmox; ESXi; Filebench; virtual machine

I. INTRODUCTION

The virtualization is considered as one of the most important topics in IT. It allows a single computer/server to use multiple operating systems simultaneously. It also helps in reducing the costs, because they can run multiple different services on a single server, leading to more efficient server utilization, easier system maintenance, and reduced hardware. As the power of a computer unit has significantly increased since 1960s when the IBM's presented its visionary idea of virtualization, this solution became popular in system implementation and maintenance [1].

There are several approaches for virtualization in IT environments: hardware, software, desktop, data, network, memory, storage, etc. The hardware virtualization implies the use of a hypervisor, which is an additional layer that lies between hardware and operating system (OS) and makes a slight delay for when accessing the resources for virtualized environment, providing lower performances when compared to bare metal or non-virtualized system [2], [3].

Actually, hypervisor is specialized firmware and/or software installed on single hardware that allows hosting of the VMs.

There are two types of hypervisors (Figure 1): type 1, that is executed directly on hardware and manages guest OSs (ESXi, Proxmox); and type 2 that is executed on the host OS (VirtualBox, VMware Workstation).

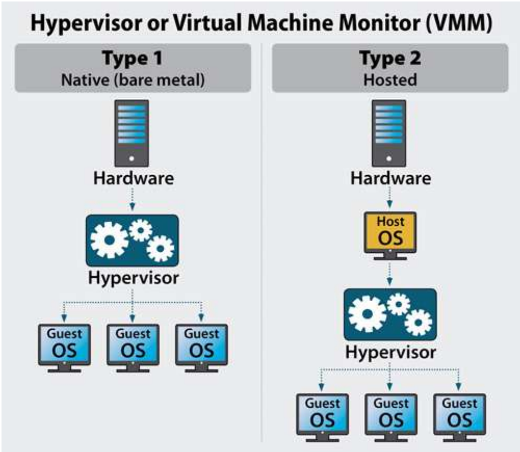

Figure 1. Hypervisor types and differences [4]

As the type 1 hypervisor has direct access to hardware, while type 2 hypervisor accesses hardware through host OS, we assume that type 1 hypervisor provides more scalability, reliability, and better performance [5].

II. RELATED WORK, OBJECTIVE AND MOTIVATION

This research is focused on the performance comparison of two type-1 hypervisors and results analysis. Since virtualization is the primary solution for systems ranging from small firms to large corporations, the arising question is: what is the best solution on the market? Some recent research addresses this issue from different perspectives, mostly considering VMware, KVM and Hyper-V hypervisors, and basing the results on Filebench or Bonnie++ [6]. This paper can provide a new picture of the situation since almost no research has focused on the Proxmox solution versus a commercial solution such as ESXi.

The primary goal of this paper is to compare performance using ESXi and Proxmox hypervisors on identical hardware, same VM parameters and the same guest OS – Linux Ubuntu 20.10 with ext4 as main file system (FS). Also, the disk we are testing has contained one of the three FSs: ext4, xfs or btrfs. Since we have used a Filebench workloads for testing, our idea was to find the best FS for each test. Selected workloads are: varmail, webserver and fileserver.

We have defined the mathematical model, measured the performances and interpreted the obtained results based on the mathematical model and hypotheses.

Borislav Đorđević - Institute Mihajlo Pupin, Volgina 15, 11000 Belgrade, Serbia, (borislav.djordjevic@pupin.rs)
Valentina Timčenko - Institute Mihajlo Pupin, Volgina 15, 11000 Belgrade, Serbia (valentina.timcenko@pupin.rs)
Nenad Nedeljković - VISER, School of Electrical and Computer Enginering of Applied Studies, Belgrade, Serbia (nedeljkovic1nenad@gmail.com)
Nikola Davidović – University of East Sarajevo, Faculty of Electrical Engineering, Vuka Karadzica 30, 71123 East Sarajevo, RS, BiH, (nikola.davidovic@etf.unssa.rs.ba)

III. Mathematical model

Variable T_W is calculated in accordance with the equation (1), and shows the total processing time for each workload.

$$T_W = T_{RR} + T_{SR} + T_{RW} + T_{SW} \quad (1)$$

Variables T_{RR} and T_{SR} represent random and sequential read time, and T_{RW} and T_{SW} random and sequential write time. There is an expected access time for every specific workload for the FS, which include following components:

$$T_{WORKLOAD} = T_{DIR} + T_{META} + T_{FL} + T_J + T_{HK} \quad (2)$$

$T_{WORKLOAD}$ represents the overall time for finishing all operations on the current workload, T_{DIR} the time needed to run all directory-related operations, T_{META} the time needed to complete all metadata operations, T_{FL} the time needed to go through all free lists operations, T_{FB} the time needed to carry out direct file blocks operations, T_J the time needed to complete journaling operations and T_{HK} the time needed to run housekeeping operation within the FS [7].

We have two candidates whose performances we compare:

1. Proxmox + VMs (guest OS).
2. ESXi + VMs (guest OS).

1. Proxmox + VMs (guest OS): The time to process the generated workload ($T_{W(Proxmox)}$) in this case depends on the benchmark interaction with guestOS FS, the characteristic of the FS, Virtual Hardware processing and the virtualization processing component of the Proxmox hypervisor (PVE-proc) is calculated in accordance to the following formula:

$$T_{W(Proxmox)} = f(BENCH, guestOS\text{-}FS, VH\text{-}proc, PVE\text{-}proc, hostOS\text{-}FS) \quad (3)$$

2. ESXi + VMs (guest OS): The time to process the generated workload ($T_{W(ESXi)}$) in this case depends on the benchmark interaction with guestOS FS, the characteristic of the FS, Virtual Hardware processing and the virtualization processing component of the ESXi hypervisor (ESXi-proc) is calculated in accordance with the following formula:

$$T_{W(ESXi)} = f(BENCH, guestOS\text{-}FS, VH\text{-}proc, ESXi\text{-}proc, hostOS\text{-}FS) \quad (4)$$

Since we are using the same settings for VMs on both hypervisors, the virtualization processing component will depend on the virtualization type and hypervisor processing as provided in the following formula:

$$PVE\text{-}proc = f(virt\_type, hyp\_proc) \quad (5)$$
$$ESXi\text{-}proc = f(virt\_type, hyp\_proc) \quad (6)$$

We are predicting the following:

- Based on the practical experience, it is expected that ESXi will produce better performance.
- Multiple VMs to have a significant performance drop compared to just one VM.

IV. File Systems

Linux FS is generally a built-in layer of a Linux OS used to handle the data management of the storage.

A. EXT4

The ext4 (fourth **ext**ended filesystem) is a journaling FS for Linux, and is developed as the extension of the ext3 [8]. It has the following characteristics [9]:

- Maximum FS size of up to 1 EB and maximum file size of nearly 16 TB.
- Hashed B-tree organizes and finds directory entries.
- Online defragmentation tool (e4defrag), which performs defragmentation of individual files or the whole FS.
- Easily detectable corruptions of files by metadata checksumming.

B. XFS

XFS is a high-performance journaling FS created by Silicon Graphics, Inc (SGI) in the last decade of 20th century [10]. It has the following characteristics [9]:

- Maximum FS size and maximum file size of nearly 8 EB.
- B+ tree organizes and finds directory entries.
- Delayed allocation for minimizing fragmentation and increasing performance.
- Implemented direct I/O for high throughput and non-cached I/O for DMA devices.

C. BTRFS

Btrfs ("better FS", "b-tree F S") is a copy-on-write (COW) FS based on B-trees. It was initially designed at Oracle Corporation in 2007 for the use in Linux [11]. It has the following characteristics [9]:

- Maximum FS size and maximum file size of nearly 16 EB.
- B-tree organizes and finds directory entries.
- Online defragmentation, offline FS check.
- Background based fixing errors on redundant files.

V. VMware ESXi and Proxmox

ESXi is an enterprise-class, type-1 hypervisor developed by VMware for deploying and serving VMs (Figure 2). It runs directly on hardware and significantly improves system performance [12]. The major part of architecture is VMkernel and processes that run on top of it. VMkernel has control of all hardware devices on server, manages resources and handles system processes. It receives requests from VMs for resources and presents the requests to the physical hardware [13].

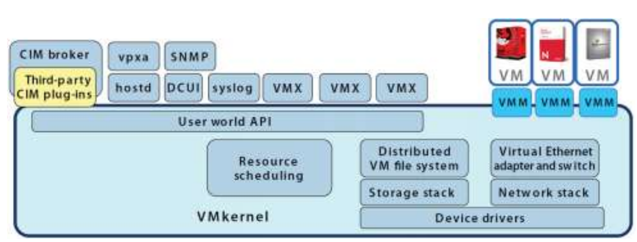

Figure 2. ESXi architecture [14]

The main processes that run on top of VMkernel are: [13]

• Direct Console User Interface (DCUI) — the low-level configuration and management interface, accessible through the console of the server, used primarily for initial basic configuration.

• The VM monitor, which is the process that provides the execution environment for a VM, as well as a helper process known as VMX. Each running VM has its own VMM and VMX process.

• Various agents used to enable high-level VMware Infrastructure management from remote applications.

• The Common Information Model (CIM) system: CIM is the interface that enables hardware-level management from remote applications via a set of standard APIs.

VMware uses VMFS. It is a special high-performance clustered FS. The main feature of this segment is ability to be shared by being simultaneously mounted on multiple servers. The VMFS datastore can be extended to span over several physical storage devices that include SAN LUNs and local storage. This feature allows you to pool storage and gives you flexibility in creating the datastore necessary for your virtual machines. [12]

Proxmox **V**irtual **E**nvironment – PVE (Figure 3) is a bare-metal hypervisor (runs directly on the hardware), to run VMs and containers. It is an open-source project, developed and maintained by Proxmox Server Solutions GmbH. For maximum flexibility, they implemented two virtualization technologies: full virtualization with KVM (Kernel-based Virtual Machine) and container-based virtualization (LXC) [15].

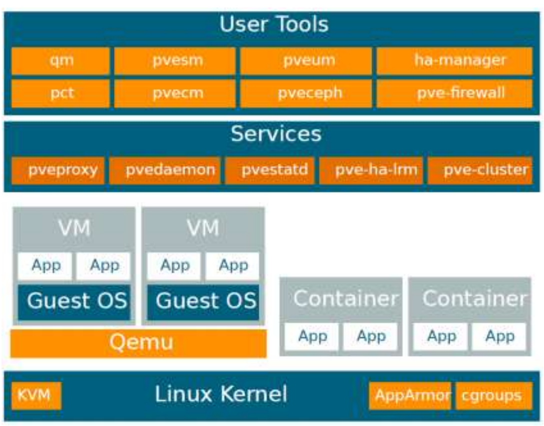

Figure 3. Proxmox architecture [15]

Proxmox uses a Linux kernel and is based on the Debian GNU/Linux Distribution. The source code is released under the GNU Affero General Public License, version 3. KVM was the first hypervisor to become part of the native Linux kernel (2.6.20). It is implemented as a kernel module, allowing Linux to become a hypervisor simply by loading a module. Benefits from the changes to the mainline version of Linux is optimization of hypervisor and the Linux guest Oss [16].

Proxmox natively supports running LXC (LinuX Containers) containers from the UI. These are similar to docker containers but behave more like a traditional VM.

Performance of KVM virtualization was the focus of this paper.

The main features for Proxmox VE [17]:

- Live migration;
- High availability;
- Scheduled backup;
- Command-line (CLI) tool;
- Flexible storage;
- OS template.

VI. Testing

The assumption of adequate testing is the application of a single hardware configuration, the same OS, and measurement methodology for all tests. The used server configuration has respectable hardware components although it is does not represent the latest technology.

The OS used is Ubuntu version 20.10, the latest instalment of Linux distribution (Table 1). During the installation process, we opted for minimal installation option which installs only essential packages and programs. The system disk uses EXT4 while the test disk is EXT4, XFS, or BTRFS.

All tests were performed using Filebench tool. Latest release of Filebench software was installed following instructions provided on the official GitHub repository of this project. Filebench is a program designed to measure the performance of FS and storage, and it can generate multiple workload types that simulate environments when using certain servers/services such as mail, web, file, database, etc. [18]. Before starting any tests, we made sure that all available updates were installed. Each VM was given 4 GB of RAM and 4 CPU cores.

TABLE I
Server Test Environment

HP ProLiant DL380 G7	
Component	***Characteristic***
CPU	2 x Intel Xeon E5540 QuadCore 2.53GHz
RAM	32GB DDR3
Storage Controllers	HP Smart Array P410i
Hard Drive 1	HP 10K SAS 146GB(DG0146)
Hard Drive 2	HP 7.2K SAS 500GB(MM0500)
PVE hostOS-FS	ext4
ESXi hostOS-FS	VMFS

The VM parameters are shown in Table 2. All used VMs have identical characteristics.

TABLE II
Virtual Machine Parameters

Component	***Characteristic***

vCPU	4
RAM	4GB
Disk	12GB + 32GB
OS	Linux Ubuntu 20.10
FS	ext4
Tested FS	ext4/xfs/btrfs

The focus of this paper is on measuring disk performance by comparing two hypervisors combined with three different FSs using 1, 2, or 3 VMs at the same time. It is expected that, as the number of VMs increases, performance will decline significantly in any combination.

Filebench is a very powerful and very flexible tool able to generate a variety of FS - and storage-based workloads. It implements a set of basic primitives like *create file*, *read file*, *mkdir*, *fsync* and uses WLM (the Workload Model Language - WML) to combine these primitives in complex workloads [18].

The files used for our benchmark were *varmail.f, webserver.f,* and *fileserver.f.* Those files are included in the Filebench software installation package, and were minimally edited to suit our needs.

The duration of each the tests was set to 120 seconds, which is the only change we made in *.f files with the goal of making the most realistic results. During the test execution, it was ensured that the impact of any external subject on system components was reduced to the minimum. The benchmark is run 3 times and the average value of the test is taken as final.

First, Proxmox VE was installed on server and nine VMs were generated, 3 for every FS. Tests were conducted in a way that one VM was first started and measured, then 2 and 3 VMs simultaneously. After that, disk is formatted and ESXi was installed. By the same principle, everything is applied to ESXi. From the generated data, the final conclusions were made by calculating the average values of the results.

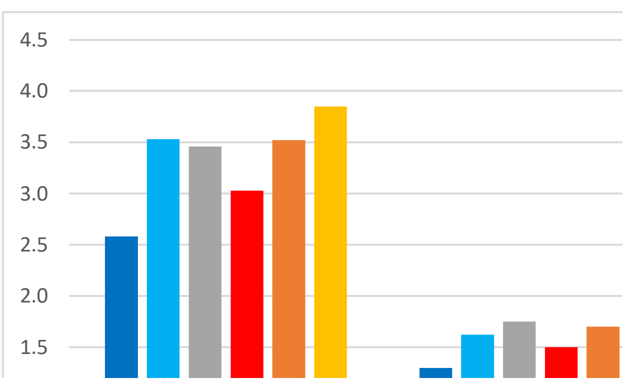

Figure 5. Varmail workload test results

TABLE III
BENCHMARK VARMAIL RESULTS

Varmail	1VM - (MB/s)	2VM - (MB/s)	3VM - (MB/s)
esxi - ext4	2.6	1.3	0.9
esxi - xfs	3.5	1.6	1.2
esxi - btrfs	3.5	1.8	1.0
pve - ext4	3.0	1.5	1.1
pve - xfs	3.5	1.7	1.2
pve - btrfs	3.9	1.9	1.5

Figure 5 and Table 3 show Varmail test results. Varmail emulates I/O activity of a simple mail server that stores each e-mail in a separate file (/var/mail/ server). The workload consists of a multi-threaded set of create-append-sync, read-append-sync, read and delete operations in a single directory. 16 threads are used by default [19].

For the Varmail workload, which is characterized by the dominant random reads and random writes, where random writes are represented by the synchronous transfers covered by equations (3) and (4), the main differences are components 3 (VH-proc), 4 (hypervisor-proc) and 5 (hostOS-FS).

When looking at the number of VMs, the combination of pve-btrfs was the best in each category, while esxi-btrfs and esxi-xfs had the same overall results with the ESXi hypervisor. We can conclude that the BTRFS FS is the best choice for a mail server.

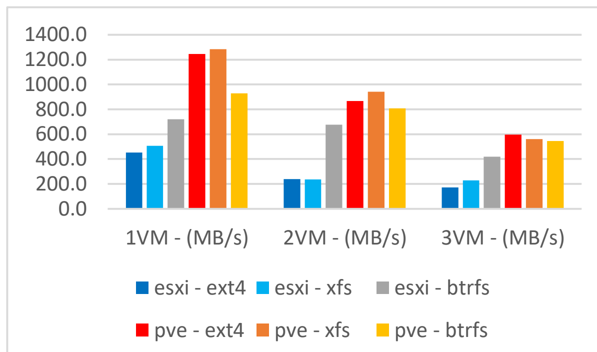

Figure 6. Webserver workload test results

TABLE IV
BENCHMARK WEBSERVER RESULTS

Webserver	1VM - (MB/s)	2VM - (MB/s)	3VM - (MB/s)
esxi - ext4	453.5	238.4	171.2
esxi - xfs	507.2	235.1	228.3
esxi - btrfs	720.0	677.5	419.1
pve - ext4	1243.9	864.9	595.0
pve - xfs	1284.1	940.5	561.8
pve - btrfs	928.8	808.0	544.7

Figure 6 and Table 4 show Webserver test results. Webserver emulates simple web-server I/O activity and produces a sequence of open-read-close on multiple files in a directory tree plus a log file append. 100 threads are used by default [19]. The Webserver workload is characterized by a dominant random read component as covered in equations (3) and (4), while the main differences are components 3 (VH-proc) and 5 (hostOS-FS). In both cases, VH-proc is Full-Hardware virtualization, but in Proxmox it is realized through QEMU. A large difference in performance in favor of Proxmox was observed in this test. The overall results of pve-xfs is 2.87 times better than esxi-xfs.

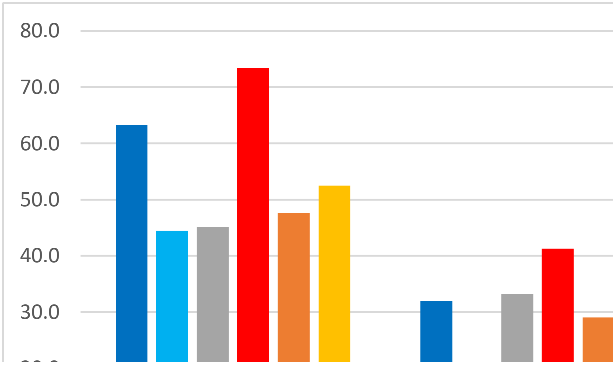

Figure 7. Fileserver workload test results

TABLE V
BENCHMARK FILESERVER RESULTS

Fileserver	1VM - (MB/s)	2VM - (MB/s)	3VM - (MB/s)
esxi - ext4	63.3	32.0	17.4
esxi - xfs	44.4	16.7	13.8
esxi - btrfs	45.1	33.2	10.7
pve - ext4	73.4	41.3	21.4
pve - xfs	47.6	29.0	18.3
pve - btrfs	52.5	26.4	20.4

Figure 7 and Table 5 show Fileserver test results. Fileserver - Emulates simple file-server I/O activity. This workload performs a sequence of creates, deletes, appends, reads, writes and attribute operations on a directory tree. 50 threads are used by default [19].

For the Fileserver workload, which is characterized by all kinds of data transfers, when considering equations (3) and (4), the main difference is component 5 (hostOS-FS). As ESXi uses VMFS, which is a clustered FS and represents a higher level of abstraction, while Proxmox uses EXT4, and the best FS in this test was EXT4, we conclude that this ruled in favor of Proxmox.

As in the previous two tests, this time too Proxmox came out as the winner but with a slightly smaller difference. We also have a match in the choice of FS: EXT4 gave the best overall results in both hypervisors.

VII. CONCLUSION

In this paper, we tested two respectable type 1 hypervisors: the commercial VMware ESXi solution and the open-source solution - Proxmox. Although it was expected that, due to its importance and big impact in the IT world, ESXi would provide better results, this did not happen. Proxmox won each comparator hypervisor + file system test. This was best seen during the webserver test where they were better almost 3 times and the third (VH-proc) and fifth (hostOS-FS) components of formulas (3) and (4) came to the fore.

If we only look at the performance of the FS, we get an interesting distribution. EXT4 performed best on fileserver, XFS on webserver, and BTRFS on varmail test.

For all 3 workloads we noticed that Proxmox is significantly better than ESXi. In the context of formulas (3), (4), (5), (6), we consider that the first two components, BENCH and guestOS-FS, in equations (3) and (4) have the same effect on for both hypervisors. The 3rd and 4th components, VH-proc, PVE-proc and ESXi-proc, differ significantly, where we notice that Proxmox is better. However, the main reason for Proxmox's victory is the 5th component (hostOS-FS). ESXi used a higher level of abstraction such as VMFS which slowed it down in this case, while Proxmox used a basic level of FS such as EXT4.

When we summarize all the test results, the used virtual machine operating system and hypervisors hostOS-FS, we can say that Proxmox is more optimized for Linux distribution.

The Proxmox virtualization system can be particulary useful for people starting their own business in small steps, without requiring additional costs. This does not mean that large companies do not use it. As already mentioned, this is an Open-Source solution and help for some of the possible problems can be found in a community where the number is unknown. If you still want to be insured, you can subscribe to the team of people behind this solution - Proxmox Server Solutions GmbH on more than favorable terms.

Interesting ideas for future work and research is to add fast Solid State Disks, comparative analysis of hypervisors using container virtualization or testing a different hypervisor such as Xen and Microsoft Hyper-V to determine which one achieves the best results.

ACKNOWLEDGMENT

The work presented in this paper has partially been funded by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

REFERENCES

[1] S. Meier, *IBM Systems Virtualization: Servers, Storage, and Software*, First edition, Redpaper, 2008.

[2] F. Bazargan, C. Yeun, J. Zemerly, “State-of-the-Art of Virtualization, its Security Threats and Deployment Models”, International Journal for Information Security Research. 3. 10.20533/ijisr.2042.4639.2013.0039, 2013.

[3] N. Yaqub, “Comparison of Virtualization Performance: VMware and KVM“, Master Thesis, Department of Informatics, Uviversity of Oslo, Norway, 2012.

[4] C. Taylor, 2020, What is a Hypervisor Server?, accessed 14 May 2021, https://www.serverwatch.com/virtualization/hypervisor-server/

[5] P. Vasconcelos, F. Araújo, G. Freitas, T. Marques, “KVM, OpenVZ and Linux Containers: Performance Comparison of Virtualization for Web Conferencing Systems”, International Journal of Multimedia and Image Processing. 6. 10.20533/ijmip.2042.4647.2016.0039, 2016.

[6] B. Đorđević, N. Maček, V. Timčenko, “Performance Issues in Cloud Computing: KVM Hypervisor’s Cache Modes Evaluation”, Vol. 12, No. 4, pp 147-165, 2015. http://uni-obuda.hu/journal/Dordevic_Macek_Timcenko_60.pdf,

[7] D. Vojnak, B. Đorđević, V. Timčenko, S. Štrbac, “Performance Comparison of the type-2 hypervisor VirtualBox and VMWare

Workstation”, 27th Telecommunications Forum (TELFOR), Belgrade, Serbia, pp. 1-4, doi: 10.1109/TELFOR48224.2019.8971213, 2019.

[8] THE EXT4 FILE SYSTEM, accessed 16 May 2021, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/ch-ext4

[9] D. Vujičić, D. Marković, B. Đorđević, S. Ranđić, “Benchmarking Performance of EXT4, XFS and BTRFS as Guest File Systems Under Linux Environment”, 2016.

[10] THE XFS FILE SYSTEM, accessed 16 May 2021, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/ch-xfsBtrfs

[11] O. Rodeh, J. Bacik, C. Mason, (2013). “BTRFS: The linux B-tree filesystem”, ACM Transactions on Storage (TOS). 9.10.1145/2501620.2501623, 2013.

[12] E. Sosa, *Mastering VMware NSX® for vSphere®*, First edition, Indianapolis, Indiana, John Wiley & Sons, 2020.

[13] VMware, “The Architecture of VMware ESXi”, Technical White Paper, 2007.

[14] B. Laurent, 2020, VMs Everywhere, accessed 24 May 2021, https://littlecorner.info/post/virtualization/

[15] Proxmox Server Solutions Gmbh, “Proxmox VE Administration Guide”, 2021.

[16] S. Aiiy, “Comparative analysis of proxmox VE and xenserver as type 1 open source based hypervisors”, International Journal of Scientific and Technology Research. 7. 72-77, 2018.

[17] S. Cheng, *Proxmox High Availability*, First edition, Birmingham, England, Packt Publishing, 2014.

[18] V. Tarasov, E. Zadok, S. Shepler, “Filebench: A Flexibile Framework for File System Benchmarking”, Vol. 41, No. 1, 2016.

[19] Filebench, 2017, accessed 23 April 2021, https://github.com/filebench/filebench/wiki