INFOTEH-JAHORINA Vol. 16, March 2017.

A performance comparison of Linux filesystems in hypervisor type 2 virtualized environment

Borislav Đorđević
Institute Mihailo Pupin, University of Belgrade
Belgrade, Serbia
bora@impcomputers.com

Valentina Timčenko
School of Electrical Engineering, Institute Mihailo Pupin, University of Belgrade
Belgrade, Serbia
valentina.timcenko@pupin.rs

Abstract— **This paper examines the basic concepts of virtualization taking into consideration five widely used Linux filesystems: Ext3, Ext4, ReiserFS, XFS, JFS. The performances of filesystems are compared on two different hypervisors, type 2: Oracle's VirtualBox and VMware's Workstation. The goal of this study is to provide a comprehensive comparison of the filesystem performances within each hypervisor, which will guide us to the proper hypervisors comparison.**

Keywords-component; Virtualization, Ext3, Ext4, Reiserfs, XFS, JFS, VirtualBox, Vmware Workstation, Postmark

I. INTRODUCTION

In its widest meaning, the virtualization is assumed to be a concept that defines techniques and methods for the computer resources abstraction. This methodology separates, isolates and organizes computer and network resources into specific work environments, allowing for hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, etc. Although defined with these attributes, virtualization is not limited only to partitioning and separating resources into smaller units, but involves a process of merging and connecting several physically separated units into the one which can be used as a whole [1].

Virtualization is based on the use of the virtual machines (VM), which stand for the software implementations of a computer that executes programs in the same way as when running an operation system on the real physical machine. There is a possibility to run a large number of VMs on a single physical machine, sharing the available resources for different applications, users and needs. Practically, VMs can simultaneously run different operating systems (OS) and applications on the same physical unit. The core of every virtualized environment is a hypervisor that manages the operation of VMs. It runs VMs and allows their communication with available hardware. The control of VM is much simple and has more flexible configuration than with the physical computer.

Hypervisors allow for multiple OSs to share hardware resources of a (physical) computer that they are installed on. Hypervisor controls the resources of the host processor while providing to the guest OS an adequate part/partition of available physical resources so that they do not affect the functioning of the host OS. Each partition is assigned a set of virtual resources, while hypervisor allocates and manages access to each VM to the available computing resources. Given that partitions are mutually isolated, it is easier to adjust functionalities of the installed OSs.

There are two basic hypervisor types. Type 1 hypervisor is executed directly on the hardware, while type 2 hypervisor executes in host OS.

Type 1 hypervisor by using VMs provides efficient access to resources as it directly communicates with the computer resources, thus it provides better protection (VMs mutual isolation) and more reliable operation (the loss of the functionality of one VM instance does not affect the stability of the host OS). Typical representatives are Xen, Citrix XenServer, VMware ESX Server. Type 2 hypervisor shows slightly worse performances in terms of utilization of computing resources, protection, and stability, but in contrast to type 1 hypervisors, the virtualization is provided in an easier way while the implementation is much faster. Typical representatives are VMware Workstation/Player/Fusion, Virtual Box, and MS Virtual PC.

II. LINUX FILE SYSTEMS UNDER THE ANALYSIS

A. Ext3

Ext3 is an upgraded ext2, with journaling functionality. It is fully compatible with ext2 [2]. Switching from ext2 to ext3 is simply achieved by creating a diary, which maintains records on transactions and reduces the risk of data loss. The logging of the transactions decreases system performances, mostly from the aspect of monitoring the overall activity of the filesystem (FS), whereas the performances are less decreased if tracking only the metadata activities. Ext3 has three modes for keeping the transaction log: journal, ordered, and writeback.

Journal mode keeps track of all changes in the FS, both in the field of meta-data and in directories. It provides higher reliability and lower performances. This mode inserts redundancy, which impacts and decreases the performances.

Ordered mode tracks the changes in the field of meta-data, whereby the changes in the FS objects are logged before updating the table nodes. This is the default log mode and guarantees the full synchronization of FS objects, and meta-data. This mode has less redundancy and faster operation.

Writeback mode tracks the changes in the field of metadata, whereby the table nodes can be updated prior to the entry of changes in the structures of the FS. This is the fastest mode but does not guarantee the consistency of the meta-data field and its synchronization which can lead to disagreeable situations, such as the appearance of new and old files versions.

B. Ext4

Ext4 FS can support volume sizes up to 1 Exabyte (EB) and files with size up to 16 terabytes (TB) [3, 4, 5]. However, Red Hat recommends using XFS instead of ext4 for disks larger than 100 TB. Ext4 has a feature called extents, which replaces traditional block mapping scheme used by ext2 and ext3. Extent is a group of contiguous physical blocks that is put for the use in such a structure and this way it reduces fragmentation and improves the performance of FS while working with large files. The size of an ext4 extent can be up to 128MB sequential space with 4 KB block size. When there are more than four extents per file, the rest of the extents are indexed in a tree.

Ext4 is compatible with ext2/ext3, making it possible to mount them as ext4. Ext4 uses "allocate-on-flush" technique to improve the performance and it is known as delayed allocation. It can delay the block allocation until the moment that the data are written to the disk. The allocation delay improves performance and reduces fragmentation for efficient simultaneous allocation of large amounts of data. Ext4 allows for the unlimited number of subdirectories. In order to enable the generation of larger directories and to allow uninterrupted service, ext4 by default uses indexed H-Tree. Ext4 uses checksums for enhancing journaling, reliability and eliminates the waiting for I/O during the log update, thus scarcely improving the performance.

C. ReiserFS

ReiserFS is one of the pioneering FSs that has introduced the "journaling" option [2]. It significantly increases performances when working with small files while other journaling FSs have weak performances with small files. E.g. when comparing to ex2, ReiserFS is 8 to 15 times faster when handling files smaller than 1KB.

ReiserFS solves the issue of internal fragmentation and increases the efficiency of disks utilization. Such high performances ReiserFS achieves by relying on the use of the optimized B + tree (one per filesystem) and dynamic allocation of index nodes. ReiserFS uses a variable size of the system block, thus a small file is written into the directory along with its "file info" structure. The diary is updated only with changes in meta-data. ReiserFS bad qualities are reflected in performances with sparse files, where ext2 performs far better. Also, when comparing to ext2, ReiserFS is slower when working with large files.

D. XFS

XFS is an original product of Silicon Graphics Inc. (SGI). It is a robust, full 64-bit FS, with a number of high-quality properties [3, 4, 6]. The first new functionality included to XFS are allocation groups, or linear memory regions of equal size, defined for each disk. Each allocation group has its own nodes table and a list of free space. Allocation groups are independent and can participate in the parallel I/O operations, thus XFS allows simultaneous and parallel I/O operations on the same FS. Allocation groups use efficient B + trees that keep information on the areas of free space and nodes. XFS optimizes the allocation of free space with delayed allocation. XFS keeps a diary for metadata. The literature provides information on XFS, such as that in the case of the large files it performs better that many other FSs.

E. JFS

JFS is a full 64-bit logging FS. It was primarily intended for IBM servers but has also been ported to the Linux OS. It provides high reliability, fast recovery, and high performance [7]. JFS is only logging the metadata. It is an extent-based FS, which allows flexible file manipulation on the disk. Extents are continuous sequences of blocks for file allocation. It specifies three parameters <logical offset, length, physical>. The JFS structure tree is based on the B + tree form. JFS uses a variety of system blocks sizes (512, 1024, 2048 and 4096 bytes), thus it is easy to customize the system files.

The big advantage is also that the nodes indexes and lists of free blocks are maintained dynamically. As for the directory, there are two possible organizations. The first scheme is used for small directories; the content of the directory is entered to its node. This feature dramatically improves the performance of working with small directories. Another scheme is used for large directories, structured in the form of an optimized B + tree, which accelerates the search, writing of new objects and their deletion. JFS is performing well with sparse and dense files. As a full 64-bit FS, JFS performs well for huge FSs.

F. General assumptions

For the specified workload, total workload processing time $T_{workload}$ for the file system includes five components given by the following equation:

$$T_{workload} = T_{Dir} + T_{Meta} + T_{FL} + T_{FB} + T_J + T_{HK} \quad (1)$$

where $T_{workload}$ is the total time needed to complete all operations on the workload, T_{DIR} the time needed to complete all directory-related operations, T_{META} the time needed to complete all metadata operations, T_{FL} the time needed to complete all free lists operations, T_{FB} the time needed to complete direct file blocks operations, T_J the time needed to complete journaling operations and T_{HK} the time needed to complete housekeeping operation within the FS.

ReiserFS and ext3 are 32-bit block-based FS, while ext4, XFS, JFS are extent-based 64 bit FS. Ext3 is simplest, while ext4 applies extent-trees feature, H-trees for directories, and delayed allocations. ReiserFS, xfs and JFS are based on the sophisticated B-trees. ReiserFS is a very suitable for working in small files workloads, as it applies the different technique for more efficient operations in that type of environments.

In the case of the virtualized environment, we define the characteristics of the requests for reading and writing files from/to the block devices as:

$$T_{disk\_request} = T_{block\_driver\_overhead} + T_{disk\_\cos t} \quad (2)$$

When calculating the generated overheads for block driver T_{BDO}, there are two important time components for drivers that are handling the request:

$$T_{BDO} = T_{BDF}(GOS) + T_{BBD}(HOS) \quad (3)$$

where T_{BDF}(GOS) denotes time consumed by the front end block driver for guest OS, and T_{BBD}(HOS) denotes time consumed by the back end block driver for host OS. The hypervisor provides the front driver for VMs while providing the back-end driver for Host OS.

III. Hypervisors under the examination

As it is explained in the introduction part of this paper, hypervisors are essential software for proper virtualization of the system [8]. This study considers two very popular representatives of hypervisor milieu: VirtualBox and VMware Workstation.

A. VirtualBox

VirtualBox is free virtualization software for installation on an existing OS. This application allows installation and simultaneous use of several guest OSs, each in a separate virtual environment [8].

The virtual environment includes network emulation, graphic and sound cards. After selecting the OS, it gets assigned a specific part of the RAM memory, following with the creation/allocation of the hard disk. The hard disks of VMs are stored in the form of image files, while in the same location the system stores the images of optical and floppy drives.

VirtualBox is virtualization software for 32-bit and 64-bit OS that are installed on Intel or AMD processor computers. The environment in which the OS is run is a virtual machine. The host OS is the OS on which VirtualBox is installed and within which a VM is run, while virtualized OS (in VM) is called the guest OS.

B. VMware Workstation

VMware is a type 2 hypervisor that allows users to set the VMs and use them simultaneously along with a physical machine. VMware supports the bridging of the existing network adapters and sharing of the physical disks and USB devices with VMs. It can simulate drives, where ISO image files can be mounted as the virtual optical drive, and virtual hard disks are implemented in a form of the .vmdk files [9].

VMware can make a copy of the VM state, the "snapshot", at any time. These images can later be used to efficiently restore VMs to a saved state at the moment of taking the snapshots of VMs. VMware allows grouping of multiple VMs in a single folder. Grouped VMs can be further turned on, off, suspended or perform as a single object, which is useful for testing client-server environments.

IV. FileSystem performance measurement

For the needs of testing FS performances under different hypervisors, VMware and Virtual box are installed and put into the operation on 64-bit Windows 7 Pro, while hypervisors are running Linux Ubuntu 14.04 64-bit, with kernel version 4.2.0-27 for testing the FSs. To each VM is assigned one processor, 1024 MB RAM, and100 GB hard disk partition. A separate 50 GB partition is assigned for the testing purposes. FS performance testing is carried out by Postmark benchmark [11].

We have proceeded with three test procedures. First, we have tested the workload of 50,000 transactions over 100,000 extra small files that have size in the range of 1B to 1000B. Then we have tested the workload of small files with size in the range of 1KB to 90 KB. We have created over 10,000 files and run 5,000 transactions. The third test was related to large files, having a size of 1MB to 30 MB. We have generated more than 100 files and performed 500 transactions. The average test duration was from 3 to 5 minutes. Due to the needs of obtaining statistical stability of the collected test results, as well as to encompass the possible impact of house-keeping operations that are used in Linux OS FSs, each test was repeated 3 times and average (mean) values were used for the needs of performance analysis. Table 1 provides information on test environment:

TABLE I. Test environment

Processor	Intel i5 4200U, 2 cores, @ 1.6 GHz.
Memory	4 GB DDR3, 1600 MHz
Hard Disk	Seagate ST1000LM024 HN-M101MBB
Disk char.	1 TB, SATA 3 (6 GB/s), buffer size 16 MB, rotat.time 5400 RPM, delay: 12 ms
OS	OS Windows 7 Ultimate 64-bita

A. VMware vs. Virtual Box: Extra small files test results

Results obtained with extra small files test for VMware and Virtual Box environments are provided in Table 2 and fig. 1; for small files, the results are given in Table 3 and fig. 2; and large files are examined based on the results presented in Table 4 and fig. 3.

TABLE II. Extra small files

kB/s	Ext3	Ext4	XFS	ReiserFS	JFS
VMware r.	15,8	26,19	18,96	81,89	7,07
VirtualB r.	19,08	44,8	39,81	89,91	11,58
VMware w.	82,04	136,01	98,49	425,35	36,73
VirtualB w.	99,08	232,68	206,75	466,99	60,16

For both VirtualBox and VMware, ReiserFS has far the best performances, while JFS is the weakest for analyzed workload of extra small files JFS.

The main reason for this results is that the workload forces a large number of metadata operations and additionally working with the ultra-small files, with B-trees structure and feature of merging objects makes ReiserFS very In the case of the VirtualBox, the significant advantage over the VMware is provided mostly because of the lower overhead for block driver that characterizes Virtual Box when there is a need for a high number of operations over the extra small objects (2) (3). efficient (formula (1)).

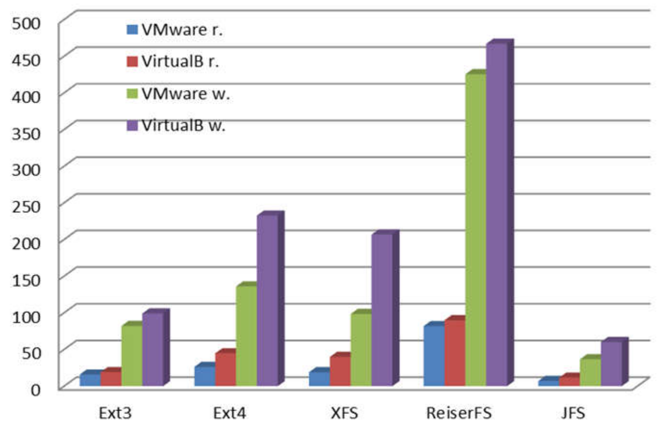

Figure 1. Extra small test results

B. *VMware vs. Virtual Box: Small files test results*

TABLE III. SMALL FILES

MB/s	Ext3	Ext4	XFS	ReiserFS	JFS
VMware r.	0,7	2,13	2,45	1,04	1
VirtualB r.	0,6	3,05	3,22	1,68	1,2
VMware w.	3,57	11,06	12,75	5,43	4,73
VirtualB w.	3,22	15,85	16,75	8,75	6,24

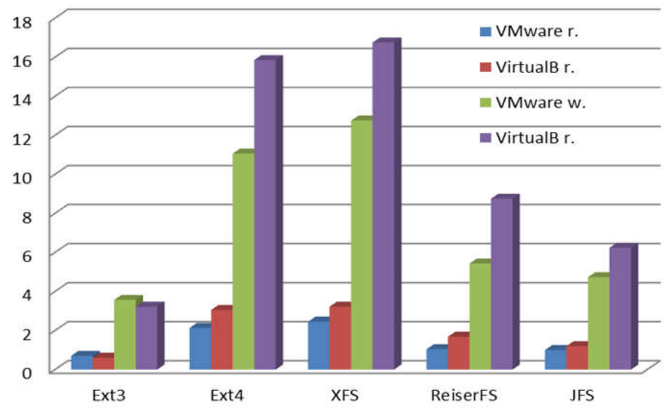

Figure 2. Small test results

XFS and ext4 are very similar in performances; both are operating excellent in comparison to other examined FSs. Ext3 has the weakest performances for the small files workload, for both examined hypervisors.

The objects are now bigger in size, thus ReiserFS is losing its primacy, and B + tree has aided XFS to dominate in this test. In the case of the ext4, its efficient features have helped to provide almost equally good performances as XFS. ext4 has solid performances, formula (1). According to formula (3), Virtual Box is significantly better than VMware.

C. *VMware vs. Virtual Box: Large files test results*

TABLE IV. LARGE FILES

MB/s	Ext3	Ext4	XFS	ReiserFS	JFS
VMware r.	5,2	14,23	14,19	12,43	15,03
VirtualB r.	8,2	21,35	23,19	18,51	10,78
VMware w.	8,75	20,28	20,22	17,71	21,41
VirtualB w.	11,36	30,42	33,04	26,38	15,36

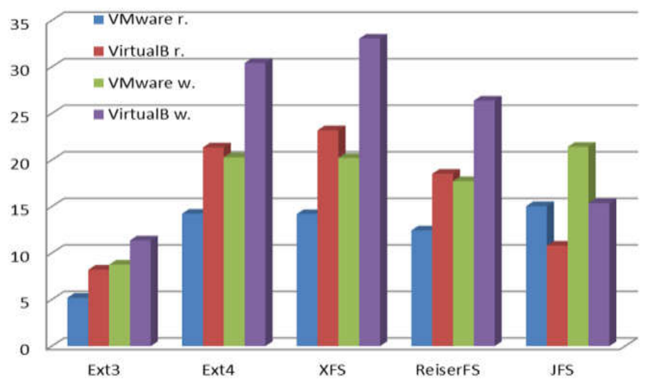

Figure 3. Large test results

The obtained results for the test with large files have indicated an interesting behavior. For some FS such as JFS, VMware is better, while in the case of other FSs, such as XFS, EXT4 and ReiserFS Virtual Box is a prefeable environment. Definitely, results have proved that JFS is optimized for large files, while ReiserFS is a good choice when working with ultra small and large files (formula (1)). In the case of workloads with dominantly large files, there are situations where, according to formulas 2 and 3, the hypervisor VMware has lower block driver overhead than at Vbox, mostly when working with JFS and in large files environments.

D. *VMware and Virtual Box comparison analysis*

Taking into consideration the three test sets and obtained results, it is clear that analyzed FSs are expressing extremely different behavior when used on native (bare-metal) hardware and differently when creating virtualized environment with type-2 hypervisors. In two of three tests, Virtual Box has shown significantly better behavior for all 5 examined FSs.

The exception is the case of large files and JFS mostly because of the lower disk block overhead. When comparing FSs, it is obvious that for the set environments ReiserFS was the best for extra small files and also very good in the case of the very large files. XFS and ext4 are performing constantly well for most of the tests. The worst characteristics go to JFS, as it shows strength only in environments based on larger files. Ext3 is much weaker than its 64-bit successor, ext4.

V. CONCLUSION

The obtained set of the test results clearly indicates that for both examined hypervisors the ReiserFS is the most efficient filesystem in the case of the extra small files workload. XFS is distinguished as the best FS when working with small files, while ext4 is just a step behind with solid performances. The most intrigant case is the workload based on the large files, where for the case of the VMware the most efficient filesystem is JFS. In the case of the VirtualBox the XFS and EXT4 have very similar performances, JFS is weak and performs just slightly better than ext3. In the case of the extra small files, the hypervisor VirtualBox has shown 20-50% faster operation than VMware. The results from the small files test have provided 17-35% better performances for Virtual Box, and 30-40% in the case of the large files workload.

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: V. Timcenko by grants TR-32037, TR -32025, and B. Djordjevic by grant III-43002.

REFERENCES

[1] Y. Dong, J. Dai, Z. Huang, H. Guan, K. Tian, Y. Jiang, “Towards High-quality I/O Virtualization,” 2009 ACM Int. Conf. Proc. Series, pp. 12-22, 2009.

[2] V. Prabhakaran, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. 2005. “Analysis and evolution of journaling file systems,” In Proc. of the annual conf. on USENIX Annual Technical Conference (ATEC '05). USENIX Association, Berkeley, CA, USA, 8-8.

[3] M. Larabel, “Linux 3.19 File-System Tests Of EXT4, Btrfs, XFS & F2FS,” in Software on 10 February 2015. [Online] http://www.phoronix.com/scan.php?page=article&item=linux-3.19-ssd-fs&num=1

[4] M. Larabel, “Linux 4.0 Hard Drive Comparison With EXT4 / Btrfs / XFS / NTFS / NILFS2 / ReiserFS,” in Software, on 8 April 2015. [Online] http://www.phoronix.com/scan.php?page=article&item=linux-40-hdd&num=1

[5] M. Avantika et al., “The new ext4 filesystem: current status and future plans,” Proc. of the Linux Symposium, Ottawa, Ontario Canada, 2007.

[6] “XFS Filesystem Structure, Documentation of the XFS filesystem on-disk structures,” Edition 0, Copyright © 2006 Silicon Graphics Inc., [Online] http://xfs.org/docs/xfsdocs-xml-dev/XFS_Filesystem_Structure/tmp/en-US/html/index.html

[7] S. Best, “JFS Layout,” IBM.
[Online] http://jfs.sourceforge.net/project/pub/jfslayout.pdf

[8] Dj. Pesic, B. Djordjevic, V. Timcenko, “Competition of virtualized ext4, xfs and btrfs filesystems under type-2 hypervisor, in the proceedings of TELFOR 2016, Belgrade.

[9] “VirtualBox”. [Online] https://www.virtualbox.org/

[10] “VMware,”. [Online] www.vmware.com/go/tryworkstation

[11] J. Katcher, “PostMark: A New File System Benchmark,” Technical Report TR3022. Network Appliance Inc, 1997.