JOURNAL OF APPLIED
COMPUTER SCIENCE
Vol. 22 No. 2 (2014), pp. 65-80

Filesystems Performance in GNU/Linux Multi-Disk Data Storage

Mateusz Smoliński[1]

[1]*Lodz University of Technology*
Faculty of Technical Physics, Information Technology and Applied Mathematics/Institute of Information Technology
ul. Wólczańska 215, 90-924 Lodz, Poland
mateusz.smolinski@p.lodz.pl

Abstract. *This paper introduces research results of I/O performance for modern filesystems configured on multi-disk storage configuration in GNU/Linux operating system. Filesystem performance tests include basic file operations statistics depending on local disks management software RAID or logical volume manager LVM. The results allow to choose the appropriate filesystem to data storage space configuration in GNU/Linux, that provides bests performance.*
Keywords: *storage space, filesystem performance, multi-disk configuration.*

1. Introduction

Operating systems offer efficient data storage and processing. For security and performance reasons data storage space is often divided to zones, each one includes block device configuration and filesystem. Many storage zones offers also flexibility, because each zone can be configured according to stored data characteristic. I/O performance for single zone grow, when in its configuration many disks are utilized with striping mechanism. It offers storage space of many disks as one logical block device (volume). Single I/O operation on striped logical volume can

be performed in parallel by many disks. Stripping has two main parameters: number of disks and disk allocation unit (chunk).

GNU/Linux kernel in version 2.6 and newer offers various striping implementation like Device Mapper or software software RAID (Redundant Array of Independent Disk) [1,2,3]. Device Mapper implementation is used by LVM (Logical Volume Manager), which offers flexible and persistent logical volume configuration. Configuration of logical volume base on earlier volume group configuration, which uses physical volumes. Stripping allocation policy for logical volume requires many physical volumes assigned to volume group and each physical volume should be localized on separate disk. Except striping LVM software offers other allocations policy for each managed logical volume. In any LVM configuration logical volume size is logical extent multiplicity. In executed storage performance tests was used default logical extend size 4MB. Software RAID implementation in Linux kernel offers many disk array levels, each has specific characteristic to data distribution between managed disks. Stripping is used in software RAID level 0 (RAID0).

Data storage zone has only one filesystem of fixed type. Filesystems type differs in logical and physical layer. Logical layer is responsible for file organization like namespace, files attributes. Filesystem physical layer corresponds to data and metadata distribution on block devices. Metadata are additional data in filesystem, that provide organization of data saved in stored files.

Kernel of GNU/Linux supports many filesystem types. For research purposes three modern and popular filesystems types were chosen: XFS, EXT4 and BTRFS [1, 2, 3]. Configuration of each filesystem include default settings except size of allocation unit, which was fixed to 4KB. For each storage performance testing scenario its filesystem is mounted in read write asynchronous mode in GNU/Linux operating system.

2. Storage test environment

All storage space configuration scenarios were configured in the same environment using identical program tool bonnie++. Storage performance test was obtained in GNU/Linux operating system from Fedora 18 distribution for x86_64 computer architecture (kernel version 3.6.10). Computer hardware used for storage testing include: i7-2600 CPU, 16GB memory, six identical SATA-2 Western

Table 1. The parameters of tested multi-disk storage configuration scenario

Filesystem type	Disk manager	Disks No.	Chunk size
BTRFS,EXT4,XFS	LVM,RAID0	2,3,4,5	16KB,64KB,256KB,1MB

Digital disks model WD500AAKX with 465GB capacity and 16MB cache. Operating system files were installed on the first of the six disks, other were used in storage configuration scenarios.

Software used for storage configuration: btrfs-progs in version 0.20, mdadm in version 3.26, xfs-progs in version 3.1.8, e2fsprogs in version 1.42.5, bonnie++ in version 1.96. Single program run includes one bonnie++ process creation for sequential data read and write in block to 32GB regular file and performing operation on 102400 thousands empty regular files distributed in 1000 directories. File operations statistic for single scenario include: number of sequentially and randomly created files, number of files which attributes are read (stat) and number of deleted files. In any storage configuration scenario executed operations utilize both regular files data and filesystem metadata.

Storage performance tests takes into account basic file operations for zone's storage configuration scenario, which includes multi-disk striping configuration and filesystem type that manage storage space of logical volume. Any storage configuration scenario has also fixed number of disks and chunk size.

3. Filesystems charasteristic

Performance analysis of results obtained in tested scenarios provides characteristic of each filesystem in multi-disk storage configurations.

3.1. BTRFS

Design of BTRFS bases on b-tree structure with write on copy (COW) rule update method, which assures its consistency without journaling. All BTRFS metadata are organized as b-tree structures and storage space allocation is dynamic using chunks during filesystem operation. Additionally in B-tree structures can also store some saved in file data fragments. BTRFS uses 64bit addressing, so it can store files up to 16EB [2].

B-tree filesystem has internal block device manager, that can manage storage space of many disks (even various HDD and SSD). BTRFS offers also logical subvolumes, that can store files separately or using clones and snapshoting. It also offers on-line resizing and defragmentation, which are performed in background.

In every subvolume BTRFS differentiates chunks with data and metadata and uses for them predefined allocation policy using striping or mirroring. Checksumming is very important reliability feature in BTRFS, which stores checksums for any data and metadata chunks. Any I/O operation in filesystem need calculate the checksum and compare its value with stored checksum pattern. Checksum mismatch causes log entry and BTRFS automatically tries to find valid data block stored in other mirrored chunk. If no mirrored chunk is present, BTRFS return a error that could be served by task that orders I/O operation.

For storage performance testing purposes many scenarios with BTRFS were prepared, in which BTRFS has not directly managed disks. Its internal block device manager has only one logical volume, in each tested BTRFS only one subvolume is created. As all tested filesystem BTRFS has identical 4KB block size and also it uses the same size for b-tree node and leaf.

3.2. EXT4

Fourth version of popular journaled EXT filesystem introduces many improvements with backward compatibility. As earlier versions EXT4 use resource groups that limit file fragmentation. Used in EXT4 48bit addressing supports filesystem storage up to 1EB with file sizes up to 16TB. It doubles the limit in EXT3 for maximum files in single directory. Implemented in EXT4 extent feature provides continuous block allocation, additionally it supports also persistent pre-allocation block device space for file. New extended filesystem versions has improvement in reliability, because EXT4 implements journal checksumming [4].

For storage performance testing EXT4 was created on single logical volume and its default configuration was used with specified 4KB block size. In every storage configuration scenario with EXT4, its journal is stored on the same logical volume. In filesystem creation process no explicit optimization for striping were used. All metadata structures are prepared statically when filesystem is created. EXT4 has limit of stored files, that is predefined in creation time.

3.3. XFS

XFS is popular journaled filesystem ported from IRIX to GNU/Linux. It was developed as filesystem with 64bit addressing. It can support logical volumes up to 16EB and stored file size limited by 8EB. XFS is transactional and uses logical journal. Other XFS features include delayed and sequential block allocation with various size extent. It limits the file fragmentation using separate allocation resource groups. Part of its internal metadata base also on b-tree structures. XFS can be online resized and defragmented [5].

Storage performance testing scenarios has XFS configured with 4KB block on single logical volume. As in any other storage configuration scenario no explicit optimization was used for striping between disks.

4. Filesystem operations performance

Filesystem structures type have impact on performance of basic file operation. I/O performance for single disk managed by BTRFS, EXT4, XFS has various data read and write speed. Figure 1 presents data read speed form file localised in each tested filesystem when it directly manages single disk, I/O performance differences between XFS and other tested filesystems are up to 14 percent. The performance results measured for each tested filesystem localized on single disk are reference point for performance comparision with any other multi-disk filesystem storage configuration.

4.1. Filesystem storage with striped logical volume

All tested filesystems localized in logical volume, which data are striped on many disks have better I/O performance like in single disk storage configuration scenario (fig. 1). However increasing number of disks that belong to striped logical volume managed by LVM with chunk 64KB does not always increase results in file operation statistics. For example XFS characteristic unlike other tested filesystems shows best performance of files creation in 2 disks scenario (fig. 2.)

In storage scenario with LVM striping on two disks XFS has lowest sequentiol files attributes reading statistics (fig. 3). In any other tested storage configuration base on LVM striping it has the best performance results in reading files attributes. BTRFS and EXT4 statistics in sequential or random files attribute reading are independent from striped disk number.

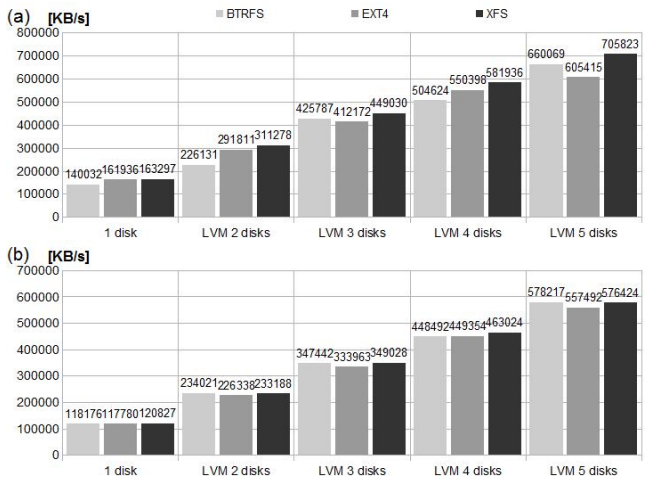

Figure 1. I/O performance for tested filesystems, striping on disks managed by LVM with chunk 64KB: (a) data read speed, (b) data write speed.

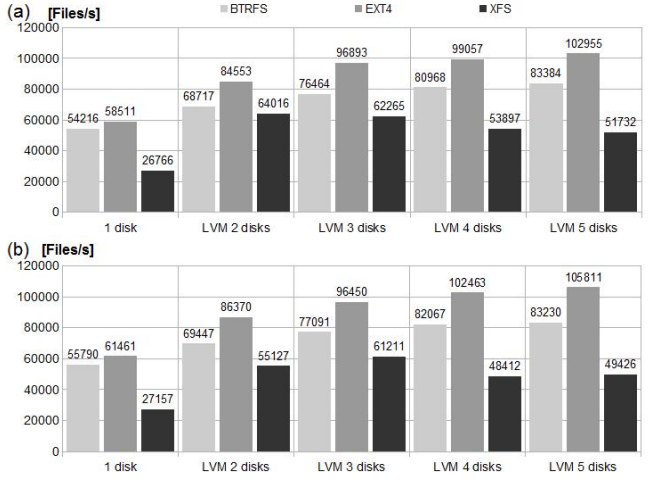

Figure 2. Files creation statistics for tested filesystems, striping on disks managed by LVM with chunk 64KB: (a) sequential files operations, (b) random files operations.

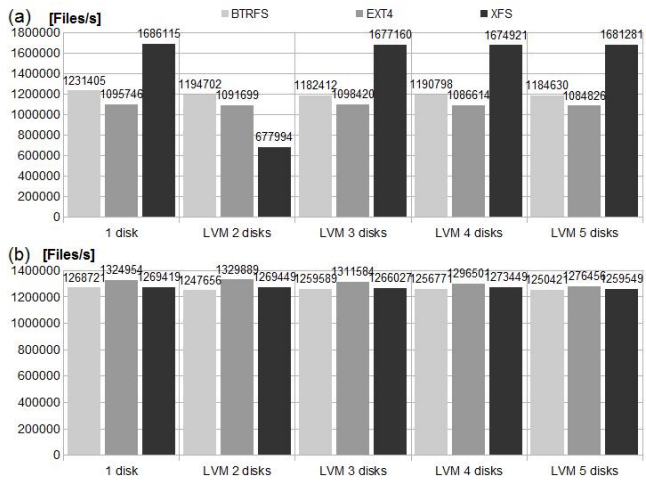

Figure 3. Files attributes reading statistics for tested filesystems, striping on disks managed by LVM with chunk 64KB: (a) sequential files operations, (b) random files operations.

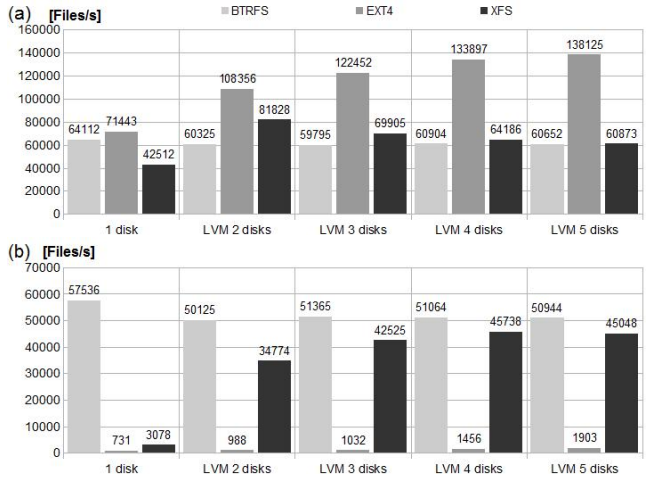

Figure 4. Files deletion statistics for tested filesystems, striping on disks managed by LVM with chunk 64KB: (a) sequential files operations, (b) random files operations.

Performance of file deletion also significantly varied between tested filesystems, especially for random file deletion between BTRFS and EXT4. This is result of using b-tree metadata structures [5]. Additionally XFS statistics show that it has eight times higher performance in random file deletion in storage scenario with striping on two or many disks managed by LVM than in scenario with single disk (fig. 4).

Also chunk size used in managed by LVM disks striping has impact on files operations performance in tested filesystems placed in logical volume, especially for data read from file localized in EXT4 or XFS filesystem. File operation statistics of each tested filesystem in dependence on chunk size in five disks striping managed by LVM were presented on figures 5 - 8.

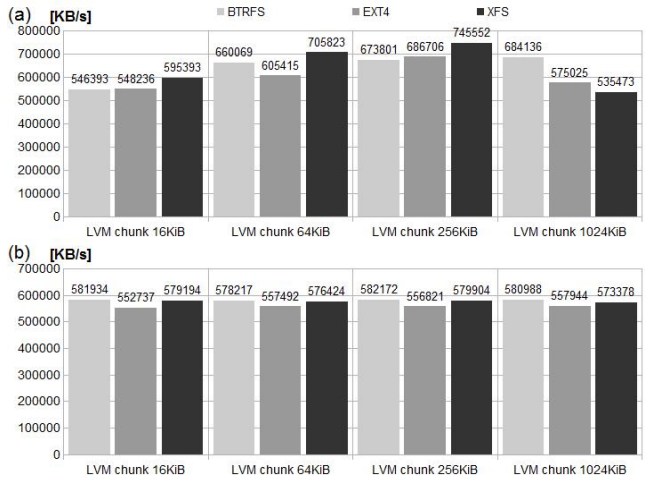

Figure 5. I/O performance for tested filesystems according to chunk size in LVM striping on 5 disks: (a) data read speed, (b) data write speed.

Research results show that BTRFS has the smallest differences relative to the chunk size in basic file operation performance, even thought it did not always have the best efficiency as XFS or EXT4 (fig. 6, 8). The XFS has the biggest differences relative to the chunk size in file creation and deletion. In tested storage scenario the best sequential and random file operation performance in XFS for multi-disk configuration was achieved in LVM striping on 5 disks with 256KB chunk size.

File operation statistics for EXT4 like BTRFS is not high dependable on chunk

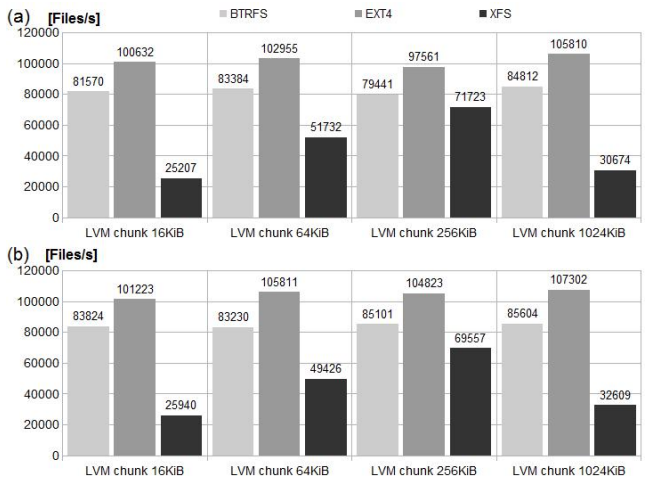

Figure 6. Files creation statistics for tested filesystems according to chunk size in LVM striping on 5 disks: (a) sequential files operations, (b) random files operations.

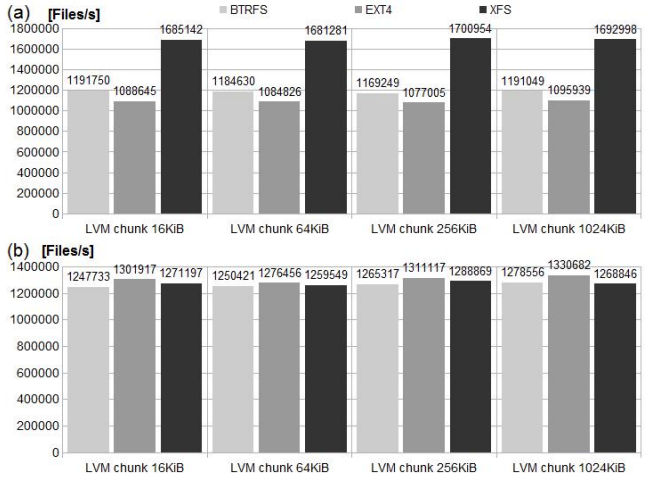

Figure 7. Files attributes reading statistics for tested filesystems according to chunk size in LVM striping on 5 disks: (a) sequential files operations, (b) random files operations.

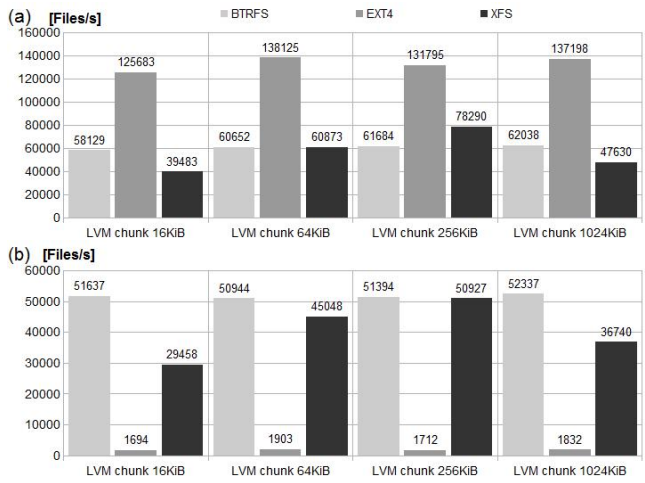

Figure 8. Files deletion statistics for tested filesystems according to chunk size in LVM striping on 5 disks: (a) sequential files operations, (b) random files operations.

size in LVM disk striping. The EXT4 filesystem has the best performance in sequential and random file creation and sequential file deletion in multi-disk striping managed by LVM (fig. 6, 8).

4.2. Filesystem storage with software disk array striping

Filesystem performance testing tool shows that I/O performance in multi-disk storage configuration managed by software disk array with level 0 has similar as LVM I/O performance if used chunk size is identical (fig. 1, 9).

If striping on disks is managed by software RAID then basic file operation performance characteristic for BTRFS and EXT4 is similar to adequate storage configuration scenario when striping on disks is managed by LVM. However XFS localised on software RAID level 0 shows less performance in sequential file creation and sequential file deletion operations. In RAID level 0 storage space configuration with two disks it is up to 169 percent less files created (fig. 2, 10) and 99 percent less deleted files then in two disk striping managed by logical volume manager. (fig. 4, 10).

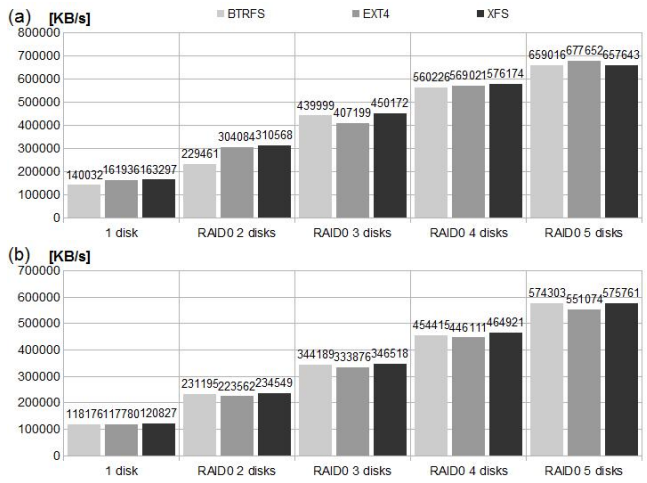

Figure 9. I/O performance for tested filesystems, striping on disks managed by software RAID with level 0 and chunk 64KB: (a) data read speed, (b) data write speed.

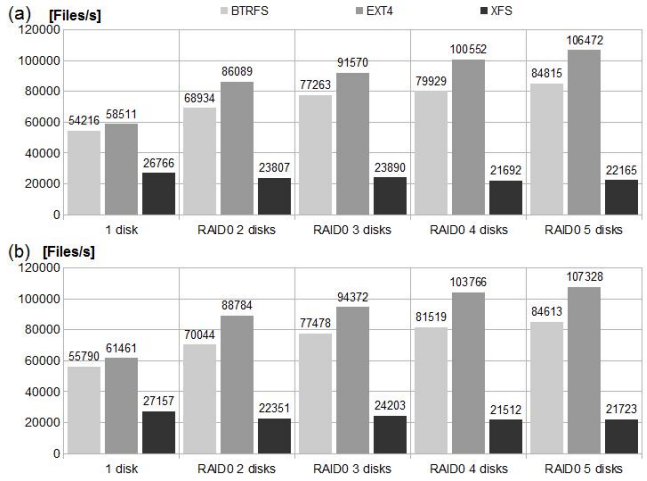

Figure 10. Files creation statistics for tested filesystems, striping on disks managed by software RAID with level 0 and chunk 64KB:(a) sequential files operations, (b) random files operations.

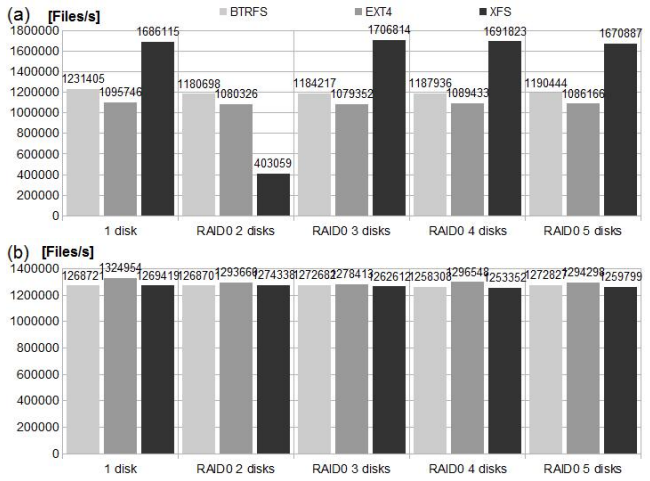

Figure 11. Files attributes reading statistics for tested filesystems, striping on disks managed by software RAID with level 0 and chunk 64KB: (a) sequential files operations, (b) random files operations.

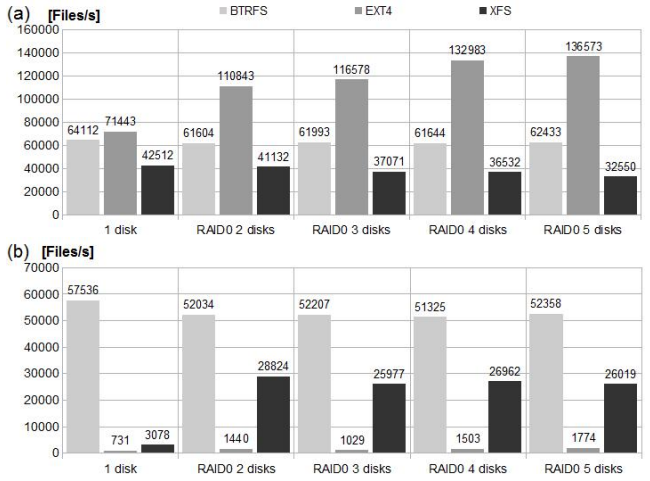

Figure 12. Files deletion statistics for tested filesystems, striping on disks managed by software RAID with level 0 and chunk 64KB: (a) sequential files operations, (b) random files operations.

Software disk array striping configuration also has chunk size parameter. Analysis of measured performance results for each tested filesystem localized in striped storage space managed by software array show that files operations statistic for BTRFS and EXT4 according to chunk size is similar to striping managed by LVM (fig. 13 - 16). However XFS has lowest basic file operation statistics when striping is managed by software disk array. Differences in sequential or random file creation and deletion were up to 75 percent worse than in identical multi-disk storage configuration managed by LVM (fig. 6, 8, 14, 16).

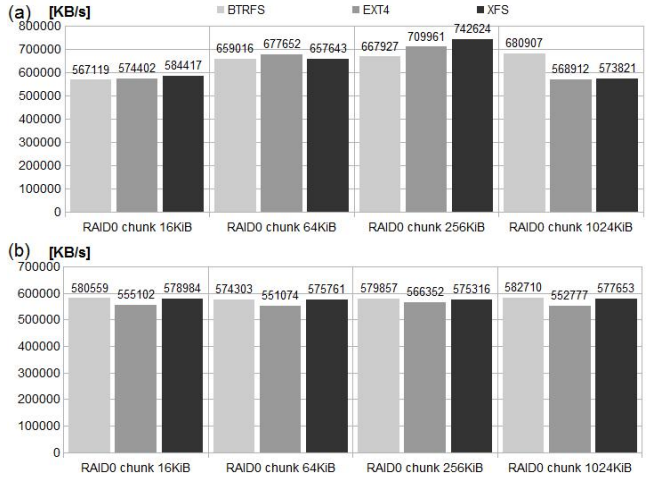

Figure 13. I/O performance for tested filesystems according to chunk size in software RAID level 0 with 5 disks: (a) data read speed, (b) data write speed.

Taking into account research results statement only BTRFS from tested filesystems provides high efficiency in all base files operations performed either in sequential and random manner. File operations performance in BTRFS are also least dependent on chunk size in filesystem multi-disk storage speace configuration whether striping is managed by software disk array or logical volume manager implemented in GNU/Linux operating system.

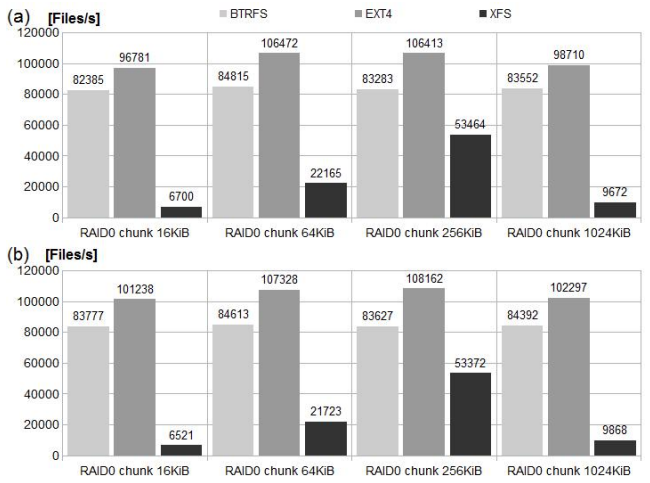

Figure 14. Files creation statistics for tested filesystems according to chunk size in software RAID level 0 with 5 disks: (a) sequential files operations, (b) random files operations.

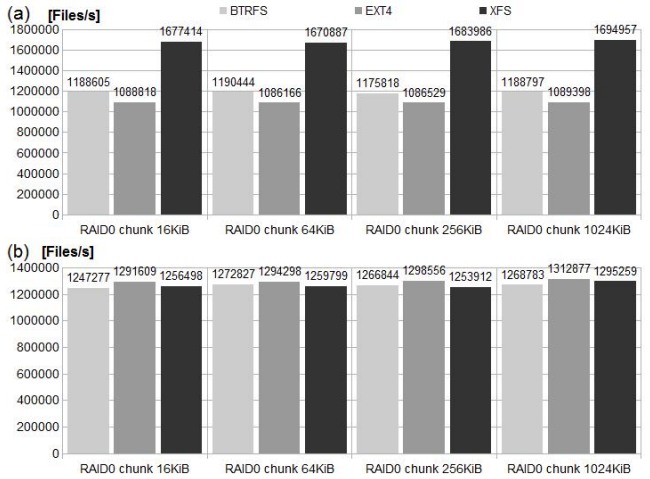

Figure 15. Files attributes reading statistics for tested filesystems according to chunk size in software RAID level 0 with 5 disks: (a) sequential files operations, (b) random files operations.

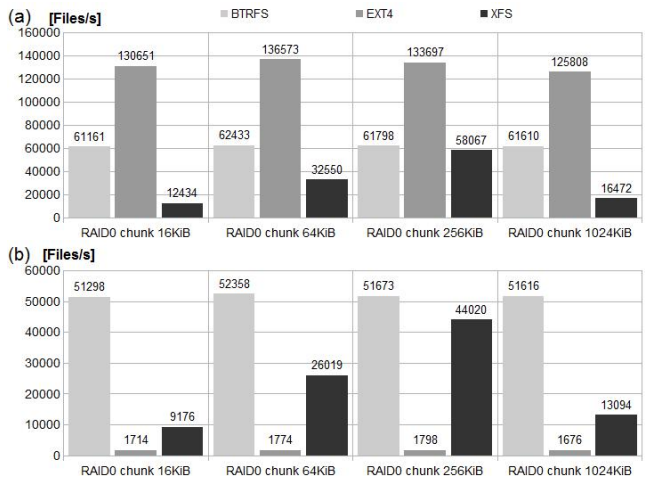

Figure 16. Files deletion statistics for tested filesystems according to chunk size in software RAID level 0 with 5 disks: (a) sequential files operations, (b) random files operations.

5. Conclusions

Configuration of storage zones in modern GNU/Linux operating system is crucial for I/O performance also in multi-disk configurations. Present versions of Linux kernel include various implementation of filesystem type and striping mechanism. In storage zone configuration fitting logical volume with striping on multiple disks and filesystem type has impact on basic file operation performance.

Analysis of research results shows that increasing number of disk in storage space configuration provides better I/O performance but not always guarantees improvement of file operation performance (i.e. increasing disk number managed by LVM or software disk array where XFS is placed).

Performance of file operations in XFS unlike other tested filesystems strong depends on chunk size in striping managed by LVM or software disk array. Additionally performance of files creation and deletion in XFS significantly decreases when striping is managed by software disk array.

The EXT4 filesystem unlike XFS and BTRFS does not include b-tree structures, therefore EXT4 in each tested storage configuration scenario random file deletion has the worst performance results. In other storage configurations EXT4

filesystem is efficient, its performance characteristic (includes both file data read and write and other basic file operations) increases when more disks are used and is not depend to chunk size in striping managed by LVM or software disk array. Modern BTRFS is usually not efficient as EXT4 in basic file operations but it has similar file operation characteristic. In multi-disk storage BTRFS provides least diverse of performance results in all statements for tested storage configuration scenarios.

References

[1] Sobell, M. G., *Fedora and RedHat Enterprise Linux*, Prentice Hall, 6th ed., 2012, (in English).

[2] Bacik, J., *Btrfs, The Swiss Army Knife of Storage*, USENIX, Vol. 37, 2012.

[3] Rodeh, O. and Bacik, J., *The Linux B-tree filesystem*, Tech. rep., IBM, 2012.

[4] Avantika, M., MingMing, C., and Suparna, B., *The new ext4 filesystem: current status and future plans*, In: Linux Symposium, 2007, (in English).

[5] C., H., *XFS the big storage file system for Linux*, USENIX, Vol. 34, No. 2, 2009, (in English).