Nevila Resulaj Xoxa *et al*, International Journal of Computer Science and Mobile Applications,
Vol.2 Issue. 8, August- 2014, pg. 26-32 **ISSN: 2321-8363**

Improved Performance of Ext4 File System by Using Extent-

Nevila Resulaj Xoxa[1], Kastriot Tema[2], Igli Tafa[3]
[1]Tirana University
Natural Science Faculty
Informatics Department
Albania

[2,3]Polytechnic University of Tirana
Information Technology Faculty
Computer Engineering Department
Albania

Abstract: This paper will discuss the performance improvements in the Ext4 file system in Linux 64-bit, coming by use the extent-s. The traditionally Unix-derived file-systems like Ext3 use an indirect block mapping scheme. This in inefficient and not aligned state of the art status where we aim for file systems. To solve this problem are proposed maps extent. Our aim in this paper is to analyze and go conclusions that is a good choice to extend functionality setting it in file system and if so what are its problems and how we can further improve this technique. For performance analysis and for extraction conclusions will use references in which measurements have been made using different benchmarks under different size and load files.

1. Introduction

The file systems Ext2 and ext3 are used by a large number of users. This is the merit of their reputation for high durability between different versions and not being state of the art in the file system which means lack of an efficient scheme for organizing and data blocks. Today the idea of using the extent-tion has become a challenge to traditional schema file to the systems of default unix using indirect block mapping [3] [10].To start analyzing the use of advancements that provides extent-s in contrast with the old scheme. Then seek to prove our expectations using results obtained from tests in references [2] [3] [4] [5] [6]. We finally will handle problems that this scheme has and solutions that are proposed for them.

2. Objective Principal

The main objective is to show that the use of techniques that extend offers better performance than indirect mapping blocks. Today ext4 aims provide expansibility, to maintain consistency of ext3 and increase performance. He is an advancement ext3 that has established new advancements well it meets these requirements.

2.1 Expected conclusions

Since most of the file system blocks tend to allocate continuously one after another, maps s-extent are an efficient way to file mapping between logical and physical blocks, while for small files of old scheme with the efficient use of memory. Expected that ext4 that use extent to have higher performance than ext3. An extent is a descriptors of a large number blocks of continual, in contrast that the use of hundreds of entry for description of each block [4].

3. Performance Analysis

In reference [3] for testing purposes are selected 3 benchmark, Flexible File-system Benchmark FFSB, Postmark and IO zone. FFSB is configured load large file and used to test the functionality of the extent in ext4. An extent is a descriptors of a large number of blocks of continual we contrast this with the use hundreds entry for description of each block [4]. Postmark is used to test the performance in low load for smaller file. Postmark creates a large number of small file and stores them in all file-system. Postmark creation operations performed reading, deleting the following file and measures the time their performance. The operations performed in order random.to ensure fairness in the simulation.. We finally use IO zone to determine the overall performance of ext4.In reference [3] The tests were conducted on a 2.6 kernel.

3.1Comparisons of Performances [4]

In this section we will see some performance comparisons ext3 implementation in the kernel 2.4 and kernel 2.6 (here we implemented extent-s), the initiator of the creation of extent, Alex Thomas where patch of the extent of the initial version was introduced in August 2003. Since then results received work inspired community of Linux developers for this technical advancement. We update the subsequent were added late allocations and multi-block allocations. The combination of these three techniques improved the increased throughput in writing sequentially. The combination of these three techniques improved the increased throughput in writing sequentially. In this test

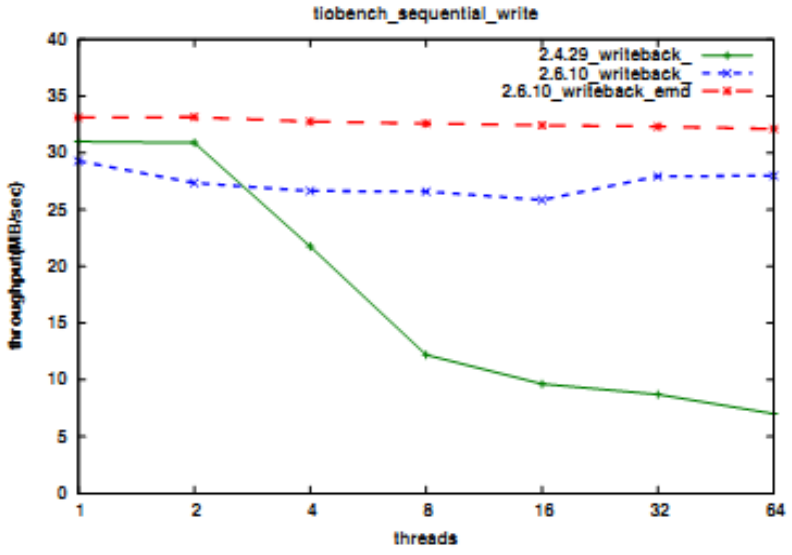

Fig 3.1 Comparison of throughput with Tiobench [6]

Three added functionality we offer improvements to kernel 2.6 throughput compared by their absence. We expect that these improvements have come from techniques extent, delayed allocation and multi-block allocation.

3.2Testing with Postmark (ext4 vs. ext3 vs. ext2)

3.2.1 Test for small file [5]

Files used for this procedure are relatively small, they vary from 1KB up to 100 KB. Postmark was configured to create 4000 file and the number of transactions performed 5000.Performance for each file operation (creation, only creation, creating transactions, wiping, wiping only) are given in the figure below.

Fig 3.2 Operations the files with transactions [3]

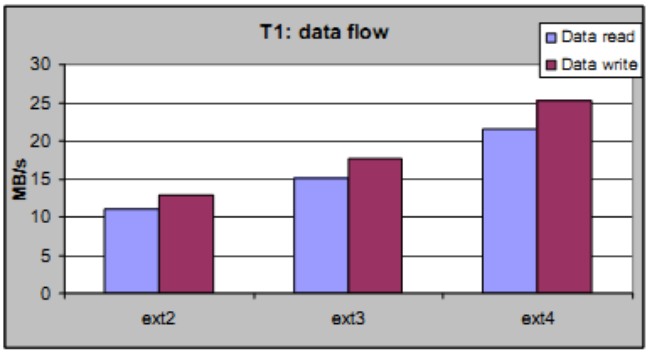

Fig 3.3The test result 1 [3]

This configuration creates about 1.6 GB of data read / write. The test for these data is presented in Fig. 3.3.In this test ext4 shows superiority in performance compared to ext3, ext2.Ext4 is more 40% faster than ext3 and almost twice faster than EX2. On the other hand ext3 is about 35% faster than ext2.The explanation for this behavior explained that in ext4 been integrated some advanced techniques that are extent allocation sites, promotion of journaling techniques and improvement of the buffer cache [3]. Loads for this test characterized by a small number of file-s (4000). By take consider the formula:

$$E[FS\_access] = E[directory\_op] + E[metadata\_op]$$
$$+ E[free\_list\_mngm] +$$
$$E[direct\_file\_accesses] + E[journaling\_time]$$

it is expected that the component E [direct_file_accesses] to be dominant. The results obtained show a better performance in writing to ext4, this is taken from delayed allocation and many block allocation. Under the presence of pre-allocation and offered extents less file fragmentation. Performances are also grown in writing by the use of techniques to an improved journaling as checksum etc. This characteristic increases in reading performance. Influence in read / write and especially in the creation and deletion files also has the index with H-tree of directories. The difference in performance becomes visible only when the deletion test, where Postmark creates and then deletes a large number of file-s. One of the advancements in the use of ext4 is Htree, while ext3 does not use.

3.2.2 Test for very small file

This is a very intensive testing procedure after processes a very large number of small files 1B ranging from 1KB up in tests performed in this reference file was generated 30000 and 50000 and performs actions. Performance results for each the Operation are given in the figure below.

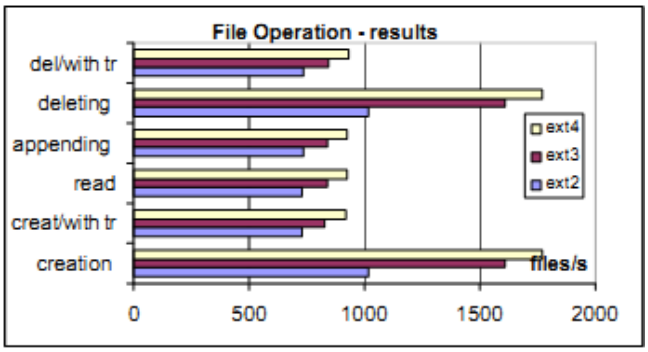

Fig.3.4Actions with file in the presence of other transactions [5]

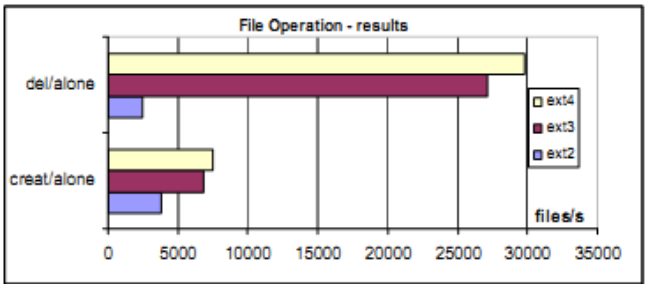

Fig3.5**Actions without transaction other** [5]

In this test file number increased to 30,000 which brings reading 14MB writing and 32MB of data. This testing procedure generates a large number of requests for metadata and I / O. So the results obtained of data flow shown below.

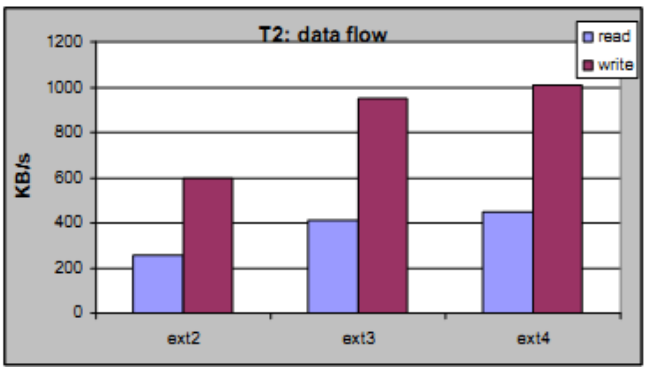

Fig 3.6Test results the data flow.[5]

In this test with very small ext4 file has shown good performance with the ext3, and ext2, however, the differences are smaller than in the previous test. Ext4 is more than 10% faster than ext3 and about 70% faster than ext2.At the same time ext3 is 60% faster than ext2. Loads for this test is characterized by a high number files, high number of operations create / delete, file size on very small (1byte-1K). In this case most of the time they spend access to directories and carried on with metadata. Performance is only visible when Postmark performs wiping action or reading. Indexing of directories with ext4 H-tree in many performance increases compared it with that of maintaining ext3 file entry uses linked lists, it does not efficiently use the directories with more entry. Also new technique used is based on ext4 in placing all i-node in a directory, different from ext3 that uses a repository entry for an i-node reference. This technique avoids the need for the requested i-node during a readdir because of breast i-node is read from memory during readdir. From the above we come to conclusions that writing and reading in ext4 has higher performance than the Ext2/Ext3 techniques are applied because of delayed allocation and allocation muti-block, techniques as use of pre-allocation extent and do not affect the performance during this test This test shows that journaling FS, ext3 and ext4 are better than Ext2. This means that journaling techniques combined the cache mechanism boost performance.

3.2.3 Test for large file

This is a more intensive test. The purpose of this test is performance testing under conditions of large Files range from 1K to 300K.

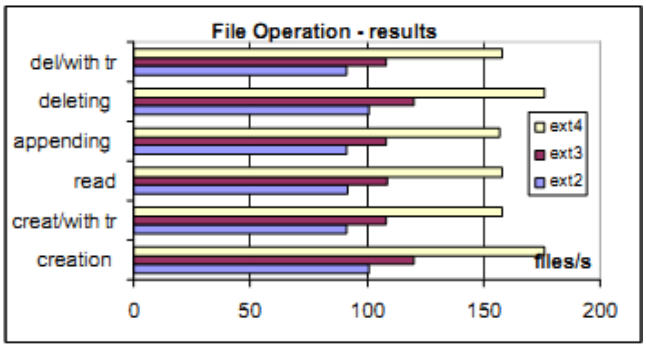

Fig.3.7Actions with files in the presence of other transactions [5]

Postmark configuration for this test is: Files generated number is 4000, the number of operations performed is 50000 results for performance of any act with each file are as follows.

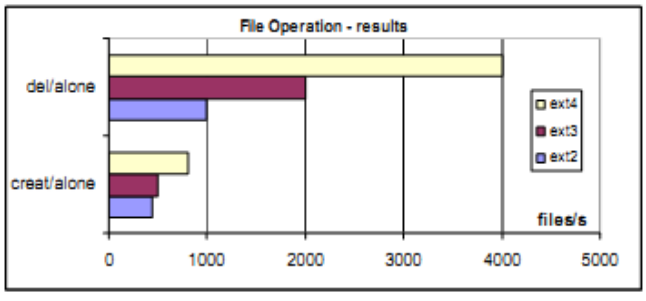

Fig 3.8Actions without transaction other [5]

The total number of sheep that were read from disk was 4.7GB of data and the number of writing was 5.4 GB. The test results for the flow of data are.

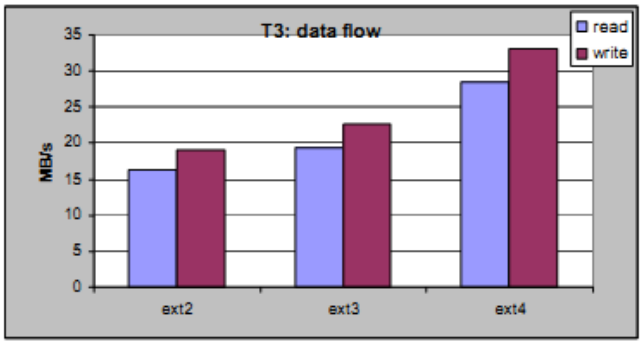

Fig 3.9The test results for the flow of data [5]

In this test we used large file and the results obtained confirmed yet that ext4 has better performance than its predecessor. In large ext4 file is 46% faster than the 73% that ext3 and ext42, while ext3 is 18% faster than ext2.Reasons for this behavior is the use and the extent to ext4 while NGOs journal's use of meta data and during major operations transform no serious effect on performance. So, sites extent is the cause of this performance, especially when comparing with ext3/ext2. Loads for this test is characterized by a small number file (4000) and average action creation / deletion. During this procedure the head most of the time accessing dominates File-level access directly. Same as in the first test ext4 has better performance in writing and this is because of the use of delayed allocation, allocation and by the presence of blocks extent. Performance in reading and writing increased by the use of journaling, also indexing the directory with H tree increases performance in writing during the creation and deletion of files, have to also look at the results in that ext4 has better performance in writing that ext3 that does not use H tree.

3.3 Test with FFSB (ex4 vs. xfs) [3]

The purpose of testing with FFSB is to test the performance that offers us the extent use-s, for this benchmark configure big load file. As a reference for comparison of systems performance since we XFS is known for high performance with large file systems, ext3 while that use reference point to show how increased performance the introduction of the extent and other characteristics of the test which is use in reference [2] example is shown below:

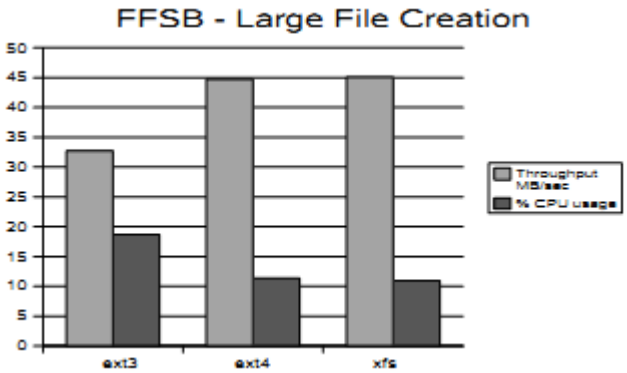

Fig.3.10 FFSB Comparison in writing sequential.[3]

From the results obtained we come to the conclusion that new techniques set in the ext4 align performance ext4 with that big xfs writings during sequential. As we expected that the extent verify-mails and delayed allocation Boost performance in the creation of large Files, which makes it more convenient in daily use normal pc.

4. Problems in the work and in the future [3]

Extent sites are not very efficient for representation Files very fragmented. For this file we can do a new type extends, one extend block-mapped. In order to increase the stability of the data in the disk, is proposed creation of a tail extent whatever extent we block's header that extent. This tail will keep the number of i-node that accesses the block and a block checksum extent of it (not the data).

5. Conclusions

The test conclusions have confirmed most of our assumptions. The use of the technique of the resulting extent more efficient and has great contribution to increasing performance, so the old techniques of mapping indirect the blocks should not be used with because we did not benefit from it any, except cost performance. Come to conclusions that the use of journaling techniques together with the cache not only has not slowed down the system but the increased performance tending to the sustainability of the file system. The results analyzed in reference should support us to use ext4. So the extent we use file systems can be considered a good choice.

6. References

[1]A .Mathur, M.Cao,A .Dilinger:"*Ext4:The new generation of ext3 file system"* Login_press Vol.32, No.3, pp.25-30, June 2006

[2] Tweedie S, "*EXT3, Journaling File system"*, Ottawa Linux Symposium, Ottawa Congress Centre, Ottawa, Ontario, Canada, July 2000

[3] M. Avantika, *" The new ext4 filesystem: current status and future plans*", Proceeding of Linux Symposium, Ottawa, Ontario, Canada, June 2007

[4]M Cao, Thodore Y. Ts'o, Badari, and A Tomas*." State of të art: Where we are with the ext3 filesystem".* In Ottawa Linux Symposium, 2005.

[5] B.Djordjrvic, V.Timcenko, *"Ext4 file system Performance Analysis in Linux Environment"*,Technical Report TR 32025 , 2008

[6] R. Bryant, R.Forester *"Filesystem Performance and Scalability in Linux 2.4.17*",USENIX Annual Technical Conference,pp. 259–274,2002

[7]S Tweedie and T. Y Ts'o. "*Planned extensions to the linux ext2/3 filesystem."*USENIX Annual Technical Conference, pp 235–244, 2002

[8]Margo I. Seltzer, Gregory R. Ganger "*Journaling versus Soft Updates: Asynchronous Meta-data Protection in File Systems"* USENIX Annual Technical Conference, pp 235–244, 2000

[9]Jonathan Corbet. "*Which filesystem for samba4?",* Technical report, 2004 http://lwn.net/Articles/112566/

[10] D. Phillips,"A directory index for ext2", 5th Annual Linux Showcase and Conference, pp 173–182, 2001

