Research journal of Applied Sciences, Engineering and Technology 4(17): 2934-2947, 2012
ISSN: 2040-7467
© Maxwell Scientific Organization, 2012
Submitted: December 22, 2011 Accepted: February 06, 2012 Published: September 01, 2012

File Systems for Various Operating Systems: A Review

Isma Irum, Mudassar Raza, Muhammad Sharif
Comsats Institute of Information Technology Wah Cantt, Pakistan

Abstract: The hardware and software both technologies are getting advanced, there are changes being introduced everyday. Therefore the storage techniques are needed to be updated for these changes to be compatible and for efficient utilization. A plethora of storage techniques or file systems have been introduced. A survey of these file systems is presented here in this study, which covers the purpose, working, advantage and limitations of the included file systems

Keywords: Cluster FS, distributed FS, flash memory FS, journaling FS, log structured FS, storage systems

INTRODUCTION

File system is the way of storing the data on physical storage devices like disk, magnetic tapes, compact disk, flash drives etc or the hierarchical organization of data established by operating system. In computing environment operating system is responsible for data organization and file systems management. With the passage of time the storage needs are changed and the amount of data increased. A file system must be reliable, persistent, secure, efficient, fault tolerant and scalable. To achieve these properties and to keep pace with changing computing needs and storage requirements, different techniques and file systems are introduced by time. To make the efficient use of our data file system is very important. Therefore, a survey for the file systems is presented in this paper. The categories included are: cluster file system, distributed file system, parallel file system, flash file system, journaling file system, log file system, mobile file system, multimedia file system and network file system (Fig. 1).

Comparison table parameters: The different parameters are used to characterize and evaluate the file system's performance. Load balanced, scalable, cryptography, adaptable, anonymity, persistent, storage type, read/write, platform characterizing parameters used here (Ragib *et al*., 2005).

Cluster file systems: When multiple object based storage devices are attached to a network, client's data acquired in client's request is temporarily stored to the client's memory and then written to object based storage device but sometimes the memory of client is erased before writing to object based storage device because of smaller space available on object based storage device than the size of file to be written. OASIS-OSD is proposed algorithm to solve this problem. For writing operation, an object based storage device is selected. The contents of file are kept on written unless an error message is generated from object based storage device that the space is finished. Then remaining pages of file that are not written yet are stored and then written to another object based storage device with available space, the proposed method was implemented with OASIS (Myung-Hoon *et al*., 2007). The cluster technology achieves high performance by grouping cheap servers to cluster. Though these cluster file systems achieve better performance but a modification in the software is needed on client side, which limits their wide application. Therefore a network file access interface is combined with cluster file system to cope with this problem. CFS-SI consists of three components: file server node, metadata server node, I/O node. The standard network file system runs on the file server node and it accepts the requests of clients in standard network file access patterns. File server node saves all metadata on metadata node and does not save any data of CFS-SI. Then this saved metadata is used to access I/O node. Likewise the whole process of the network file system is completed (Jun *et al*., 2009). Most supercomputers are found in form of large clusters now days which need for compact, enhanced and dispersed metadata processing techniques. An ideal metadata processing policy requires the automatic balance of name space locality and even distribution without any manual processing. DDG partitions the name space into hierarchical units dynamically using triple defined distribution granularity. Another technique S2PC-MP is proposed for cross over operation's consistency. It decreases the overhead by getting commit operation with metadata operations during normal processing and can recover metadata consistency quickly after any crash occurring in servers (Xiong *et al*., 2011).

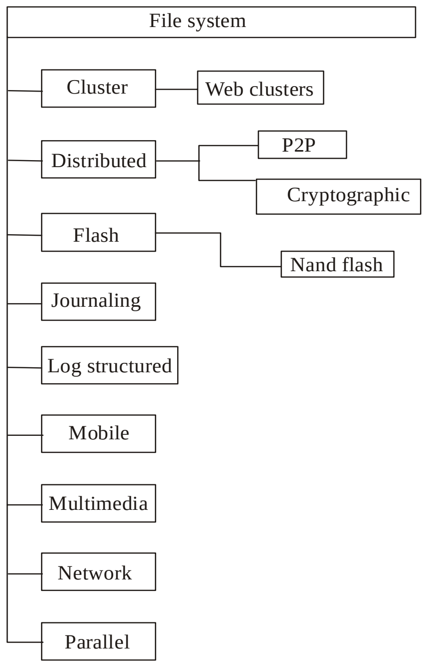

Fig. 1: DFD file systems

DISCUSSION

Web cluster file systems: The idea of TH-CluFS is basically originated from the working of high speed networks usually access the remote data as fast as they are accessing the data located locally. This emphasis the need of an I/O balancing technique for cluster files systems. Initially the files are transferred from busy nodes to free nodes to share the I/O load. Finally, disk cache and memory are combined to make file cache of memory disk. Unique cache performs the function of I/O balancing at file cache level. This certainly performs the performance of web server clusters as compared to other traditional methods (Wei *et al*., 1999) (Table 1). Web proxy servers have an important place in today's web architecture. Disk I/O is a problem among the major problems appeared against the performance of proxy servers as shown by previous studies. Traditional file systems do not have good performance for proxy servers and have large overheads. UCFS is a system that is developed for enlightening the I/O performance importance of proxy servers. UCFS maintains the tables for metadata available in memory and about the all load of metadata updates and searches are eliminated. It reflects a clustered file system that uses huge disk transfers for the improvement of performance of writing operation of disk. Clustered file system also enhances the read operation and does not create garbage. Consequently UCFS improves the performance of proxy servers as shown by result of experiments (Jun *et al*., 2002) (Table 2).

Distributed file systems: There are many cases in which application need the exclusive access to a certain file but privileges given to some file are not preempt able and it limits the access to that file. Local file system deals with this problem by just assigning the rights and checks the instances of an open file for more opens against some more requests. This required more overhead by having the need for one more server for registering the instances of opened files. To address with exclusive access problem semi preempt able locking mechanism is introduced; the server is unknown of any state of a file opened globally. This state is maintained at client side individually. When a request of opened file comes the server sends the messages to client to resolve the conflict when client releases the held lock then the requesting client uses it (Randal *et al*., 2000). The advancements in memory and processor technology consistently emerge the disks with stronger processing power and compact cache memory. By this increased processing power disk is allowed to perform more operation than only the common disk operations. The latency of data manipulation can be reduced by giving application processing offloading part to disk. Suck disks are known as active disks. ADFS is a file system in which data server is active disk based. Data files resided on these active disks have the capabilities of operations and making objects. Application like databases, operations related to application are run by disk processor means clients are returned with the results only instead of data files. In this way ADFS can reduce the system overhead (Hyeran *et al*., 2001). Usually large distributed file systems separate the metadata operation from the reading writing operations of a file. But existing systems follow restricted metadata management strategy because their goal is a managing the data in a distributed fashion and I/O performance is more preferred than metadata. The DP of metadata is developed and checked for metadata at large scale. DP sorts the whole metadata, calculates the division position and divides metadata among a number of metadata servers. The division positions are changed with the changing volume of metadata and workload. To avoid decreasing throughput, DP uses the replication policy (Jong-Hyeon *et al*., 2008). A tool developed for the purpose of load balancing over many servers in distributed file system management from computing environment of distributed base. The technique is based on the data mining procedures and graph analysis algorithm. The data mining techniques are used to identify distributed file system, file patterns and graph analysis relocates the file sets across the different file servers (Alexandra and Archana, 2002). Farsite is a distributed file system to provide availability by making copies of files on many desktop computers. These copies of files use a considerable storage space. If it is possible to get back the storage space then it is significant. Calculation of over 500 desktop file systems describes that about 50% of the used space was used by replicated files. A technique

Table 1: Comparison between different cluster file systems

Technique	Load balanced	Scalable	Encryption	Adaptable	Anonymity	Persistent	Storage type	Operations	Testing platform
TH-CLUFS (Wei *et al*., 1999)	-	Yes	-	Yes	-	Yes	File	Read	Linux
UCFS (Jun *et al*., 2002)	Yes	Yes	-	Yes	-	Yes	Blocks	Read/write	Unix
OASIS-OSD (Myung-Hoon *et al*., 2007)	No	-	-	Yes	Yes	Yes	File	Read/write	Linux
CFS-SI (Jun *et al*., 2009)	Yes	-	-	Yes	-	Yes	Blocks	Read/write	Unix
DDG (Xiong *et al*., 2011)	Yes	Yes	-	Yes	-	Yes	File	Read/write	Linux

Table 2: Advantage and limitation table for cluster file system

Technique	Advantage	Limitation
TH-CluFS(Wei *et al*., 1999)	Combines the advantage of parallel I/O systems and distributed file system and is suitable for cluster file system..	Cannot be implemented on distributed file system independently due to the volume dispersion conflict as in distributed file systems volumes stay on a single server but TH-CluFS spans multiple servers which are specified by system automatically.
UCFS (Jun *et al*., 2002)	User space is used to run the whole system. Therefore, it is not difficult and not expensive as well from the implementation point.	Tested and implemented as study projects, no real world implementation yet.
OASIS-OSD (Myung-Hoon *et al*., 2007)	Uses two techniques namely i-prevention & ii-detection and recovery to ensure sufficient space on OSD for a file and write of file to the OSD respectively.	A locking technique is used for imposing mutual exclusion on the process in writing operations which creates load to the system.
CFS-SI (Jun *et al*., 2009)	Load is balanced by providing network file system with many file server nodes, all metadata is saved in single metadata server nodes that why the data isconsistent and system design is simple, all files are divided among different I/O nodes, that is why the capacity of I/O node and network bandwidth is better utilized.	Besides the advantage of single metadata server node there is the main disadvantage; there can be performance limitation and single point failure problem.
DDG (Xiong *et al*., 2011)	Distributes the object in a namespace among the servers of metadata, this results the metadata throughput positively, in case of server failures keeps the cross results the metadata server operation which results a reliable and available file system.	To measure the performance, a relatively small platform for testing is used. One name space storage server, six metadata servers and eight client nodes used instead of a full system. The validity of results is assumed for large systems.

is presented to regain the used storage space occupied by folly duplicated files due to farsite file system mechanism incidentally. The technique consists of the following strategies, Convergent, Using encryption the duplicate files are converted to a single file even encrypting by two different keys, SALAD, A database that is used to aggregate the components and location information of file (Douceur *et al*., 2002). There is difference between the storage of continuous media data and traditional text based data and between the bandwidth needs. The file systems supporting continuous media data require large volumes and high bandwidth. AD-DFS is a distributed continuous media file system developed using autonomous disks. ADs can do little processing and directly attached to the network. Linux platform's implementation is also given Cuneyt and Sarit (2003).

Peer to peer file system: SDA-DFS is file allocation, the two techniques of replication and fragmentation are used at servers. The security of file including secrecy and integrity are secured even a subset of servers are victims of some security attack. The algorithm is of adaptive nature because of change in read write patterns of file allocation and client's location in network change (Alessandro *et al.*, 2003). A peer to peer algorithm designed for the purposes of transparent read and write from the storage devices using an interface named FUSE, Provides two features of high data availability by using the replication and high fault tolerance by using decentralization. Due to DHT calls it is scalable. DRFS is suitable for cooperative environments. For data storage it uses random, independent of contents identifier and maintains high performance and low overhead with the help of simultaneous many read and writes. Implementation is associated with an office where DRFS is installed on employee's machine, they request and store file (Dalibor *et al*., 2009).

Cryptographic file system: A disk system attached with the secure network system is presented which shows that cryptographic security can be implemented with the distributed file system. The type of these systems is affordable with the high speed processors of today. The most reliable and secure technique for the user is signing the checksum of each block for which they are using encryption based on public key and on disk side is to check every block for authentication before writing to it. Three schemes are used scheme1 is slower because of signature generation and checking sum. Scheme 2 decreases the load on disk of CPU by giving the responsibility of check to the server. Scheme 3 is faster because it does not use signature generation and checking techniques (Ethan *et al*., 2001). BRAVE is object based distributed secure file system having strong security characteristics which are similar in behavior and meaning to other distributed system. By using SCARED object storage devices file system and metadata can be stored without opening the information on devices about this data. The relation of file and data, all directory and file data can be encrypted. This provides with the encryption of file system metadata rather than encryption of file filters. Every device is treated as a separate unit therefore BRAVE file system can make different parts for administration. With the help of key servers separate security mechanisms can be used in similar file system (Benjamin *et al*., 2002). Providing security is very necessary for data storage systems. Typically, storage space comes with complex network system today. These networks are located at secure locations as data centers. But they are still vulnerable for attacks, cryptographic file system curtail the threat of attacks by getting encryption and integrity protection techniques together, provide end-end security to clients. SAN file system is used and key management technique is implemented on SAN file system. The hash trees are used to do file encryption and integrity protection as well. Both techniques are implemented at client file system driver (Roman and Christian, 2007) (Table 3, 4).

Flash file systems: It reflects the achievement of cost effectiveness as it is the most desirable and critical factor during development process of mobile consumer devices. Applying the compression mechanism is simple but an effective approach towards the achievement of cost effectiveness. An analysis of techniques for compression of mobile devices for consumers, at file system level is given in this study. Traditional file systems of compression are optimally used for disk oriented system and have rich resources of computing, they are not well suited for mobile devices, as with weak power of processing and little memory (Seunghwan *et al*., 2007). Due to the frequent writing of small data inserts a gap between span of life and persistency. In synchronous writes of small data one page is wasted at least as the NAND flash provides support for a page level I/O only. Wasting of page, results in the reduction of usage and life span of NAND memory. It uses the NOR flash as a log to store the data, whenever small data is entered at end of file. The maintained log in NOR are then transferred to the NAND flash in page alignment fashion (Chul, 2008). Increases the performance of flash memory by implementation of state transition and utilization of reallocation blocks it reduces the flash memory operations. Read operation is improved by limited quantity of log blocks that minimizes the table size for sector mapping. State transition is implemented using both techniques of in place and out place in a block. This ensures the utilization of all sectors of data block before allocation of a log block (Kwon and Chung, 2008). Flash memory is getting popularity due to its particular properties of huge capacity, non volatile, consuming low power and resisting the shock. Hard disk drive has been replaced by flash drive in many applications particularly in embedded systems. It is an important topic of research to implement the file systems on flash memory. Boot loader is used to identify the mapping table physical address. The entries of mapping table are tuple of file id, file index address. File index is helpful in loading the related log records oof file which builds the file’s metadata. In case the file metadata is not found in cache the log record helps file system construct the metadata (Shun-Fa and Chin-Hsien, 2009). The design of flash file system should take three factors into consideration for the sake of efficiency for non linear editing; these are frame header updates, system calls and data indexing. NLE-FFS is designed for non linear editing and its architecture is based on the phase change of NAND flash and RAM, for multimedia devices supporting NLE. Three new concepts are introduced: new scheme for data indexing, system calls and H-data block, which deals with large multimedia files and flexible data management, minimizes overhead of rewriting and to reduce frame header updates overhead respectively. Byte level updates are allowed instead of updates at page level, thus many bytes of frame header are updated (Man-Keun *et al*., 2009). Due to physical properties of NAND flash memory many flash file systems are proposed but these systems have performance overhead and scalability issues due to management of metadata in flash memory All metadata is stored in virtual storage of metadata, applies phase change RAM. PFFS2 manages metadata in a fixed location virtually and by in place updates at byte level (Youngwoo and Kyu Ho, 2011). Deduplication is necessary for NLE, as large data is duplicated because of it. Deduplication file system is introduced for NAND flash memory in embedded system. Duplication generated by NLE is predicted to reduce the overheads of computation. More, shared data management scheme, data indexing and garbage collection are proposed. There is a possibility for reducing the write

Table 3: Advantage and limitation table for distributed file system

Technique	Advantage	Limitation
SDA-DFS (Alessandro *et al*., 2003)	Provides secured allocation of files dynamically in such a distributed structure at large scale.	For optimized distribution a specific stable pattern of read and write operation is required by sub networks.
DRFS (Dalibor *et al*., 2009)	In case of peer failures, the root sub directory is saved and replication makes it possible to recover the data, this provides persistency.	Scalability depends on DHT.
SNAD (Ethan *et al*., 2001)	Authenticated encrypted storage is achieved.	Scheme 3 is faster but it is impossible to find who has written a file at last.
BRAVE (Benjamin *et al*., 2003)	The brave file system provides strong mutual authentication between the clients and storage devices by introducing a simple keying method.	The brave file system provides strong mutual authentication between the clients and storage devices by introducing a simple keying method.
CS-DFS (Roman and Christian, 2007)	Provides strong security by combining encryption and integrity protection.	The read operations are not fast because to verify a data page there is a need to go to all hash tree nodes.
SPL-DFS (Randal *et al*., 2000)	Provides high performance file access.	High performance strongly depends on the presence of temporary locality and little sharing of data between the clients.
ADFS (Hyeran *et al*., 2001)	A more scalable file system can be achieved by distributing the state information to active disks, with increasing availability and reliability.	When all the subdirectories of root exist in one complete disk, the optimal case occurs.
DP (Jong-Hyeon *et al*., 2008)	Maximum throughput and high distributi on is achieved.	Read operations are slower 10 to 40% than write operation.
MAP-DFS (Alexandra and Archana, 2002)	Provides intelligent decision on mapping read-write file set with feasible dependencies to a number of file servers.	Graph analysis introduces a problem of graph coloring which is NP-hard for general graphs. Due to this it is intractable in worst case.
Farsite (Douceur *et al*., 2002)	By reclaiming the used storage space the space resources can be reused.	Can not be implemented on a corporate network.
AD-DFS(Cuneyt and Sarit, 2003)	Equally distributes the functionality of file system and load among the entities participating.	A batch size of 32 can give the best result of performance if batch size is less than 32 then there is considerable drop in throughput.

Table 4: Comparison between different distributed file systems

Technique	Load balanced	Scalable	Encryption	Adaptable	Anonymity	Persistent	Storage type	Operations	Testing platform
SDA-DFS (Alessandro *et al.*, 2003)	-	Yes	Yes	Yes	Yes	Yes	File	Read write	
DRFS (Dalibor *et al*., 2009)	-	Yes	-	Yes	-	Yes	File	Read write	Unix
SNAD (Ethan *et al*., 2001)	Yes	Yes	Yes	Yes	Yes	Yes	Chunks	Read write	Unix
BRAVE (Benjamin *et al*., 2003)	-	Yes	Yes	Yes	Yes	Yes	File	Read	Linux
CS-DFS (Roman and Christia, 2007)	-	Yes	Yes	Yes	Yes	Yes	File	Read write	Unix, windows
SPL-DFS (Randal *et al*., 2000)	Yes	No	No	Yes	Yes	Yes	File	Read	
ADFS (Hyeran *et al*., 2001)	Yes	-	-	Yes	-	Yes	File	Read write	Unix
DP (Jong-Hyeon *et al*., 2008)	Yes	Yes	No	Yes	-	Yes	File	Read write	Linux
MAP-DFS (Alexandra and Archana, 2002)	yes	yes	-	yes	-	yes	file	Read write	
FARSITE (Douceur *et al*., 2002)	-	Yes	Yes	Yes	Yes	Yes	File	Read write	Window s 2000
AD-DFS (Cuneyt and Sarit, 2003)	Yes	Yes	-	Yes	-	Yes	File	Read write	Linux

operations for repeated data and using NAND flash memory in efficient way (Man-Keun and Seung-Ho, 2010).

NAND flash file systems: Flash memory in particular NAND flash memory has proven to be a major technique for data storage. An interface at block level of translation is needed in between the chips of flash memory and the present file system. Present file system are developed for disk like storage, therefore they are not efficient for flash memory storage. A faster flash file system for NAND flash memory is proposed; CFFS uses a technique named "pseudo-hot-cold" and separates by non similar flash blocks allocation of data and metadata. Separating the

Table 5: Advantage and limitation table for flash file system

Technique	Advantage	Limitation
CFFS (Seung-Ho and kyu-Ho, 2006)	Outperforms in system booting time and garbage collection overheads.	For reducing mount time the physical address of index block is stored in first block, therefore, the first block is erased again and again, this let the wear-leveling performance to get worse.(Sang *et al*., 2009)
.NAMU (Sang *et al.,* 2009)	Uses the new index structure that is well suited for the large files, like multimedia files, decreases accesses. the mount time as scans only blocks of index, reduces memory usage by storing the data in segments instead of pages.	Supports the sequentially accesses of files not random If number of files get larger than some defined threshold then the mount time gets longer.
MNFS (Hyojun *et al*., 2009)	Satisfies the real time needs of multimedia files in mobile devices as the bandwidth is increasing of multimedia data.	Non multimedia files also exist in mobile devices, for these files MNFS may be an inefficient way of storage management.
LECRAMFS (Seunghwan *et al*., 2007)	Provides arbitrary performance even if the memory of the system is not sufficient for holding a working set of files.	Compression ratio is not sufficient as comparing with other compressed file systems.
HFFS (Chul, 2008)	No garbage because of merging small data in the NOR flash Ensures persistency and life span of data written thanother flash file systems.	If log block size is big then the long delay is observed as write latency, to avoid this log block size must be 32 KB.
EAST (Kwon and Chung, 2008)	Innovative reallocation Blocks, limiting the number of log blocks and state Transition mechanisms-	-
LMM-FS (Shun-Fa and Chin-Hsien, 2009)	Low-memory management for log-based file systems on flash memory.	When the cache size was larger than 193,600 bytes, no Significant improvement of the hit rate was observed. As A result, the proposed method with small cache size can still Provide reasonable performance.
NLE-FFS (Man-Keun *et al*., 2009)	It allows Byte-level data updates instead of entire page updates. Thus, The overhead caused by frame header updates can be Effectively reduced.	However, it did Not consider large data duplication incurred by NLE operation, Which yields large capacity over head and consumes much time. (Man-Keun and Seung-Ho, 2010) .
PFFS2 (Youngwoo and Kyu Ho, 2011)	Solves the scalability problem of previous NAND flash file systems.	In case of limited pram the performance is subject To the higher locality access.
NLE-DFFS (Man-Keun and Seung-Ho, 2010)	With this file system, it is possible to reduce write operations for redundant data and thus use NAND flash memory space efficiently suitable for mobile embedded systems having low processing capability.	To save the swapping cost In case of metadata is small; the total number of page writes is similar to other file systems.

data and metadata plays a role in improvement of garbage collection performance comparatively than the other methods (Seung-Ho and kyu-Ho, 2006). As the old flash memory file system of NAND, stores the file data in pages, as the number of files and size of files get larger, the problem domain of scanning area get larger at time of mounting linearly. In this paper a new index structure for data is proposed, uses index pages of child and root, for mounting of file system uses index blocks and reduces the scanning area, for writing operation of files uses segments as unit of storage for flash memory accesses sequentially, for garbage collection uses the counting of erases in a file of single blocks for the improvement of file system performance (Sang *et al*., 2009). It is particularly designed for NAND flash memory, targets the needs of devices like MP3 players, digital camcorders and personal media players. This technique uses the hybrid mapping and block allocation of files, with the help of block allocation table and upward directory(Hyojun *et al*., 2009) (Table 5, 6)

Journaling file systems: To evaluate the performance in terms of robustness of journaling file system the method is proposed under failure of disk writes. Constructs models how journaling file system orders writes of disks under numerous modes and these models are used to repair write failures (Vijayan *et al*., 2005). Dual FS keeps data and metadata in two separate devices and manages them differently. Metadata is managed as log structured file system and data is managed in groups. It greatly decreases the I/O time in workloads taken by file system (Juan *et al*., 2007). Robustness of journaling file systems is evaluated through this method under disk write failures. Models are constructed for different journaling file system modes and are used to inject the faults into system (Vijayan *et al*., 2005). Dual journaling method is used in this technique, dual in a sense that two types of data is stored from beginning and ending to the centre portion of storage device (Jeong-Ki *et al*., 2006). Rapid recovery from crashes is possible by using journaling file systems.

Table 6: Properties of NAND flash memories from different flash file systems

Technique	Type of index page	Flash page size	Metadata size	No. of index entries	Files size min	File size max	Writing speed/time	Reading speed/time	OS
CFFS (Seung-Ho and kyu-Ho, 2006)	Class 1	512 B	256 B	64	96 KB	12 MB	200 µs	12 µs	Embedded Linux
	Class 2	2 KB	256 B	448	1916 KB	960 MB			
NAMU (Sang *et al*., 2009)	Root	512 B	320 B	482	4 KB	768 KB	371 KB/s	27099 KB/s	Linux
	Root	2 KB	320 B	432	864 KB	54 MB			
	Child	512B	0	128	64 KB	2 MB			
	Child	2 KB	0	512	1 MB	64 MB			
MNFS (Hyojun *et al*., (2009)	-	-	-	-	16 KB	128 KB	1500 KB/s	-	FTL
LECRAMFS (Seunghwan *et al*., 2007)	Set 1	2 KB	4.4 MB	272	1.7 KB	49.6 KB	Read only	9 s	Linux
	Set	22KB	7.8 MB	234	1.7 KB	99.1 KB			
HFFS (Chul, 2008)	NOR	-	4 MB	-	64 KB	-	2B/11.5 µs	2B/90 ns	Linux 2.6
	NAND	2 KB	128 MB	-	-	-	2 KB/400 µS	2 KB/125 µS	
EAST (Kwon and Chung, 2008)	-	-	-	-	-	-	200 µs	15 µs	-
LMM-FS (Shun-Fa and Chin-Hsien, 2009)	-	16896B	-	32	-	-	-	-	Linux
NLE-F FS (Man-Keun *et al*., 2009)	-	2 KB	-	-	128 KB	-	200 µs	15 µs	-
PFFS2 (Youngwoo and Kyu Ho, 2011)	Direct	512 B	-	128	4 KB	16 KB	1600/100000 (TRANSACTIONS)	900/1000 (TRANSACTIONS)	Linux
	1-indirect	2B	-	512	260 KB	4 MB	-	-	
	2-Indirect	-	-	-	32.26 MB	2 GB	-	-	
NLE-DFFS (Man-Keun and Seung -Ho, 2010)	-	2 KB	-	-	128 KB	-	-	-	-

Table 7: Comparison of log structured file systems

Technique	Compared to	Total operations	Performance increase	CPU utilization
WOLF (Jun *et al*., 2002)	LFS	-	26%	21%
BabuDB (Stender *et al*., 2010)	Ext4, Berkeley	9.9 million	94.4, 495%	-
ZBD (Thanos *et al*., 2010)	Postmark, SPEC SFS	50,000	80, 35%	311%
ZEST (Paul *et al*., 2008)	-	868 MB/s	96.4%	-
CMFS (Taizhong *et al*., 2010)	MINFS	-	14%	-
PROFS (Jun and Yimingl, 2007)	LFS	-	32%	10%

An analysis is presented for recovery of crashes in which different techniques and their strategies of crash recoveries are discussed (Sanders *et al*., 2002). Speed and reliability are two most important elements of performance matrix for file systems. However, there are still problems due to hardware, power and software failures. This study presents the implementation of journaling file systems in limited resources availability and derives a file system structure for embedded systems (Ge and Zhu, 2008). Currently, the data integrity and consistency are maintained through logging techniques. The new technique uses the atomic write block method to preserve the consistency of data (Michael and Amnon, 2002).

Log structured file systems: By dividing active and inactive data into segments buffers in memory and then writing to disk segments, disk segments are forced to create bimodal distribution. Active segments are invalidated quickly but inactive segments are remained untouched. This method decreases the garbage collection overhead (Jun *et al*., 2002). The performance of distributed file systems is limited by their metadata server. Log structured merge trees are better than traditional storage trees (Stender *et al*., 2010). Transparent compression at I/O path can increase the storage space efficiency in online storage (Thanos *et al*., 2010). Zest is based on two file systems log structured and parallel systems to achieve various performance efficiencies (Paul *et al*., 2008). The next generation of solid state and non volatile memory is Storage Class Memory (SCM). It combines the advantages of DRAM like the robustness and high performance with the low cost hard disk (Taizhong *et al*., 2010). To boost the I/O performance of log structured file systems, performance oriented data reorganizing scheme is introduced. This scheme rearranges the data on hard disk while the garbage collection and system idle time (Jun and Yimingl, 2008) (Table 7).

Mobile file systems: With the development of mobile devices into portable information devices it is becoming necessary for the designing of a distributed file system for such communication devices. In this task one must keep in view the available little storage, little ability of computation and unreliable cellular networks. Mobile code technology separates the common file systems into two parts one is client part that includes some files requiring little computing and some operation; the other part is server part containing a lots of operations and files with heavy computing abilities. The file system of portable device is the part of server's file system (Yurong *et al*., 2001). The transportation of large files from a client to server on weak connection is a critical problem of mobile file systems. Operation shipping technique suggests the updating of shipping operation on server instead of updating the large file itself over the network. User sends its operation to a strongly connected client to a server, client reperforms the operation, reproduce files and check the originality of files then sends the operation to the sever on client's behalf (Yui-Wah *et al*., 2002). For global roaming the mobile network must introduce personal mobility, network portability and terminal mobility, for this a unique personal number for each user of mobile is needed. Database architecture is introduced to contain the all location independent numbers consists of sub database systems; each subsystem comprises three levels tree architecture, each of which interconnects through root node (Zuji and Christos, 2004). MPEG-4 video streaming is declared as defacto standard for current multimedia mobile services like voip, video conferencing etc but illegal user still use without copyright and paying for the service. Therefore a protection scheme is introduced that implements the symmetric encryption like DES, on little segments of video plane so that the users who do not have the permission or pay for it can not use the video format (Kim *et al*., 2005). Mobile phone, MP3 players and digital cameras use the flash memory and the FAT file system is implemented on mobile devices with a little amendment, by considering the problems with FAT file system two techniques have been proposed namely sector reservation and ACPA, which reduces the internal overhead and removes the changes made frequently in file allocation table (Park and Ohm, 2006). Network partitions tend the file servers to update their data under file system mode server's reintegration. Its design must assure the safe file system operation in case of failure and concurrency. This can be achieved by implementing the server's integration in paradise file systems (Azzedine and Al-Shaikh, 2006). Compressed file systems are well suited for mobile devices because of storage shortage but side by side it requires more I/O and computational overheads. To cope with these problems the proposed technique introduces swapping and replacement strategy, suggests the keeping of most accessed data of compressed file system in main memory and uncompressed data in a swap space. This help to increase hit ratio and decrease many operations of imitation and decompressing (Kwon *et al*., 2008). Growth of mobile electronic devices market, the protection and security of data and design of the data management itself are getting a significant place. A solution is presented where a linux kernel provides hardware support for encryption tasks. The approach is verified on linux crypto API a system that stores the data in safe way Pedraza *et al*. (2007) (Table 8).

Multimedia file systems: There is a significant demand for video on demand application with the growing networking and computing. Minimum probability of blocking is invented in this paper as load sharing is required for these systems. For placement of multimedia file the genetic algorithm and bin packing algorithm are combined (Kit-Sang, 2001). Solution to let hard disk drives with a single main stream to play and record multiple video streams, handles different disk types and different streams with different bit rates (Li *et al*., 2003). The problem of Broadcast file system based on digital storage media-command and control protocol, is solved by the solution presented in this paper, the protocol is innovatively changed to form a hierarchical structure that can be transmitted. To differentiate broadcast file system from the traditional system optimized strategies are introduced for caching and receiving data from network (Zhang *et al.*, 2004). By nature real time multimedia files are accessed sequentially it favors the data placement in seek fashion optimally. Scalable encoding scheme gives the ability to player to change the playback rate of content of multimedia. But this sequential playback does not match with sequential scan of file. This problem adds the complexity to structure and design of file system. This method is useful to eliminate the retrieval of useless blocks and decreases the head movements too (Won *et al*., 2006). Tracking of a file assignment problem is solved via genetic algorithm for video on demand system at large scale. Problem of File assignment is the replacement and allocation of blocks of movie files to disk that the probability of blocking is optimally minimized (Jun, *et al*., 2008). First 128 bytes are left untouched in DICOM for the metadata directly accessed. This free space can be utilized to declare a file as TIFF. TIFF with ICC is better than any other technology of DICOM. DICOM-TIFF16-ICC files are good in all cases where a DICOM viewer reopens the big frames frequently (Jacques, 2008)

Network file systems: There is a lot of studies has been done on distributed system in the past but they are not widely accepted yet for large networks. Traditional network file systems are suitable for strongly interconnected networks and do not work effectively in wide area settings. Many techniques of optimization have been introduced for wide networks but those do not consider the file characteristics rather introduce too much computing overhead in presence of good networks. A

Table 8: Advantage and limitation table for mobile file system

Technique	Advantage	Limitation
(Yurong *et al*., 2001)	Provides a gateway for implementation of practical file system on mobile file in portable devices.	Number of requests and response time are directly proportional; more requests larger response times.
(Yui-Wah *et al*., 2002)	Provides mobile computing and ease the user to take their work everywhere they want independent of location and other to, environmental limitations imposed by their network.	N/A
(Zuji and Christos, 2004)	Achieves significant database throughput, to meet the delays of location tracking and call delivery, in mobile networks for global roaming.	Results the high costs of maintenance because to distribute loads of database it adds additional nodes than a common tree structure.
(Kim *et al*., 2005)	Provides a scheme for securing MPEG-4 video for protecting copyrights and ownership laws in mobile devices.	Uses DES as encryption scheme, though it is a light weight encryption technique but take only 8 bits as input from MBs of data and produces the same length of encrypted as original data.
(Park and Ohm, 2006)	Provides deterministic response time than the traditional FAT file system in mobile devices.	The best response time is changed by 0.1 m sec but the worst is unchanged.
(Azzedine and Al-Shaikh, 2006)	Provides a conflict free mobile file system and persistency of data.	If the entries in server transaction log exceeds the 10000 the number of conflicts increases, but more than one server transaction log causes more time due to server switches.
(Kwon *et al*., 2008)	Presents a different swapping and replacement technique that significantly increases hit ratio and decreases the I/O overhead.	Swap area can only contain the data like heap or stack data. The pages containing an image, program or data file can not be stored in swap area because there are original copies of those in secondary memory.
(Kwon *et al*., 2008)	Presents a different swapping and replacement technique that significantly increases hit ratio and decreases the I/O overhead.	Swap area can only contain the data like heap or stack data and can not contain images, data or program, because they are original copies of those in secondary memory.
(Pedraza *et al*., 2007)	Provides a programmer with the benefits of hardware acceleration that a programmer does not need to rewrite the code.	The hardware performance strongly depends upon pipelining level.

distributed file system has been proposed that targets the local area network of high bandwidth to dialup connections of low bandwidth (Weisong *et al*., 2004). Mostly P2P file systems are open system when a node connects to a system it can access the data, but it is not good for PVR based file sharing as PVR denotes private device. This technique invents a file sharing model in which two nodes are connected with each other and data is shared only between those two connected nodes. Shared files contain large data like multimedia data the model should make available the space for node by increasing the sharing degree and decreasing the data placement redundancy (Seungtaek *et al*., 2005). File systems are implemented in operating system's kernel for the sake of speed and high performance; this is also true for network file systems. CPU speed is increasing day by day as technology is advancing, but the network equipments is not able to support the such high speed thus the implementation at kernel's level is wasting, that is the reason the file systems are converted to be implemented on user level space to take full advantage of available bandwidth and other network resources. This technique favors the complete implementation of file system at user's space (Ivan and Mario, 2006). Web, file transfers and streaming are kept on top of the TCP, therefore TCP analyses of data delivery and identification of three mismatches are done existing in common file systems. As remedy of these mismatches the three techniques are designed this ends these three mismatches (Sang and Kyu, 2007). Frac controls the transmission of messages between client and server. It determines the roles, duties and rights of users, produces the virtual namespace that gives an interface to update the access control and query. No need to make any changes in the client or server file system model (Aniruddha *et al*., 2007). Bit torrent is a famous P2P file system; in this Study two different classes for peers are introduced to analyze the effect of free riding of bit torrent. Bit torrent is successful to maintain prevention of a system from without seeds free riding but may not be successful with a high number of seeds. Therefore a method of allocation of seed bandwidth is proposed (Minglu *et al*., 2008). Unstructured data has larger amount than the structured data, structured data exist in databases where it is managed automatically. Companies keep their data in arrays as unstructured data; the reason behind this is when they reequip their system they must have to upgrade or update the structured data. Companies are now turning to file area networks. File

Table 9: Comparison of network file systems

Technique	Load balanced	Scalable	Encryption	Adaptable	Anonimity	Persistent	Storage type	Operation
(Weisong *et al*., 2004)	Yes	Yes	No	Yes	-	Yes	File	Read write
(Seungtaek *et al*., 2005)	Yes	Yes	No	Yes	-	Yes	File	Read write
(Ivan and Mario, 2006)	-	Yes	No	Yes	-	Yes	Block	Read write
(Sang and Kyu, 2007)	Yes	Yes	No	Yes	-	Yes	File	Read write
(Aniruddha *et al*., 2007)	-	Yes	No	Yes	-	Yes	File	Read write
(Minglu *et al*., 2008)	Yes	Yes	No	Yes	-	Yes	Block	Read write
(David, 2007)	Yes	Yes	No	Yes	-	Yes	File	Read write
(Daniel, 2008)	-	No	No	Yes	-	Yes	File	Read write
Haiying (2010)	Yes	Yes	No	Yes	-	Yes	File	Read write
Haiying (2010)	Yes	Yes	No	Yes	-	Yes	File	Read write
(Song *et al*., 2010)	Yes	Yes	No	Yes	-	Yes	Block	Read write
(Ryosuke *et al*., 2010)	-	No	No	Yes	-	-	-	-

area network overcomes the problem of managing the unstructured data by key concepts of global namespaces and a virtual hierarchy of directories, files and folder to hide the complexity of unstructured data in network (David, 2007). Great detail of properties of peer to peer overlay topologies is presented including dynamics and focusing on modern networks. Graph related properties of individual snapshot is discussed and gotten into focus (Daniel, *et al*., 2008). File replication and consistency are other design parameters which are considered in peer to peer systems. File replication usually generates extra overhead by generating replica, extra nodes and low use of replica, the proposed method combines the both consistency and replication in a systematic manner that it achieves high efficiency in replication and maintains the consistency at low cost Haiying (2010). Another method for increase of replica utilization and query efficiency at low cost is presented. It enhances the replica utilization by using query traffic hub and frequently requesting nodes Haiying (2010). In distributed conditions the utilization of cache is may differ greatly for different clients. For great utilization of cache in a sharing mode the cooperative cache is introduced. It allows clients who have taken advantage from larger caches to forward the data to those clients who are under utilizing the cache and it requires the effective analysis of cache data (Song *et al*., 2010). For wireless transfer of file an antenna package is proposed. The antenna package produces the radiation from its open ended wave guide that is built in its side. The two antennas one for transfer and the other for receiving are fabricated (Ryosuke *et al*., 2010) (Table 9).

Parallel file systems: Parallelism came into being because of computational demands; parallel systems try their best to perform the large and heavy I/O requests in the presence of other potential workloads. This paper presents a technique to access small files in such large environments optimally (Philip *et al*., 2008). Common file systems are expected to face many failures; therefore it is a good remedy to take snapshot of the whole system for the enhancement of the reliability. PVFS provides the functionality of snapshot. It is deployed on a cluster of systems (Kwangho *et al*., 2008). In parallel I/O systems service time and arrival time are very critical. For the purpose of file assignment in such systems two algorithms are introduced named sort partitioning and hybrid partitioning aiming for decreasing the load balance at all disks (Lin-Wen *et al*., 2000).

CONCLUSION

It will not be wrong to say that computer system operations can not be performed without storage operations. Every operation needs the support of primary or secondary memory. A big part of efficiency and performance of operating system depends on the storage system as it controls the efficiency of I/O also. For an efficient Operating system it is very necessary for it to adopt a robust and efficient storage system to adopt. Therefore file systems are very important and critical. With the passage of time and changing computing needs and hardware file systems needs are also changed, therefore a number of techniques are proposed and file system has achieved a great attention of researchers. According the way of storage and implication systems the file systems fall into various categories; a review of some of these has been covered in this Study including their computing environments, performance characteristics and other parameters.

REFERENCES

Alessandro, M., L.V. Mancini and S. Jajodia, 2003. Secure dynamic fragment and replica allocation in large-scale distributed file systems. IEEE Transact. Parallel Distribut. Syst., 14(9): 195-200.

Alexandra, G. and S. Archana, 2002. A load balancing tool based on mining access patterns for distributed file system servers. Proceedings of the 35th Hawaii International Conference on System Sciences, (ICSS' 2002), 7-10 Jan. 2002, IBM Almaden Res. Center, San Jose, CA, USA, pp: 1248-1255.

Aniruddha, B., S. Smaldone and L. Iftode, 2007. FRAC: Implementing role-based access control for network file systems. Proceedings of the Sixth IEEE International Symposium on Network Computing and Applications, (SNCA' 2007), 12-14 July 2007, Rutgers Univ., Piscataway, pp: 95-104.

Azzedine, B. and R. Al-Shaikh, 2006. Servers reintegration in disconnection-resilient file systems for mobile clients. Proceedings of the 2006 International Conference on Parallel Processing Workshops, (PPW' 2006), SITE, Ottawa Univ., Ont., pp: 6-114.

Burns, R.C. R.M. Rees and D.D.E. Long, 2000. Semi-Preemptible Locks for a Distributed File System. Proceeding of the IEEE International Conference Performance, Computing and Communications Conference, 2000, (IPCCC '00), Dept. of Comput. Sci., IBM Almaden Res. Center, San Jose, CA, pp: 397-404.

Benjamin, C.R., M.A. Smith and D. Diklic, 2002. Security considerations when designing a distributed file system using object storage devices. Proceedings of the First International IEEE Security in Storage Workshop, (SSW' 2002), 11 Dec. 2002, IBM Almaden Res. Center, San Jose, CA, USA, pp: 24-34.

Chul, L., S.H. Baek, A.H. Park, 2008. A hybrid flash file system based on NOR and NAND flash memories for embedded devices. IEEE Transact. Comput., 57(7): 1002-1008.

Cuneyt, A. and M. Sarit, 2003. A scalable bandwidth guaranteed distributed continuous media file system using network attached autonomous disks. IEEE Transact. Multimedia, 5(1): 71-96.

Douceur, J.R., A. Adya, W.J. Bolosky, D.R. Simon and M. Theimer, 2002. Reclaiming space from duplicate files in a serverless distributed file system. Proceedings of the 22nd International Conference on Distributed Computing Systems, Microsoft Research, pp: 14.

Dalibor, P., B. Thomas, V.H. Fabio, H. David and S. Burkhard, 2009. The design and evaluation of a distributed reliable file system. Proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies, (PDCAT' 2009), 8-11 Dec. 2009, Dept. of Inf. IFI, Univ. of Zurich, Zurich, Switzerland, pp: 348-353.

Daniel, S., R. Reza and S. Subhabrata, 2008. Characterizing unstructured overlay topologies in modern P2P file-sharing systems. IEEE/ACM Transact. Network., 16(2): 267-280.

Ge, Q. and Y. Zhu, 2008. The structure and implementation of journaling file system on embedded system. Proceedings of the 2008 International Seminar on Future BioMedical Information Engineering, (FBIE '08), IEEE Computer Society Washington, DC, USA, pp: 140-143.

Geer, D., 2007. Improving Data Accessibility with File Area Networks. Published by the IEEE Computer Society, 40(11): 14-17.

Haiying, S.J., 2010. IRM: Integrated file replication and consistency maintenance in P2P systems. IEEE Transact. Parallel Distribut. Syst., 21(1): 100-113.

Haiying, S.J., 2010. An efficient and adaptive decentralized file replication algorithm in P2P file sharing systems. IEEE Transact. Parallel Distribut. Syst., 21(6): 827-840.

Hyeran, L., K. Vikram, W. Chirag and H.C.D. David, 2001. Active disk file system : A distributed, scalable file system. Proceedings of the First Eighteenth IEEE Symposium on Mass Storage Systems and Technologies, (SMSST' 2001), IEEE Computer Society Washington, DC, USA, pp: 101.

Hyojun, K., W. Youjip and K. Sooyong, 2009. Embedded NAND Flash File System for Mobile Multimedia Devices. IEEE Trans. Consumer Electr., 55(2): 545-552.

Ivan, V. and Z. Mario, 2006. Network distributed file system in user space. Proceedings of the 28th International Conference Information Technology Interfaces (ITI'2006), Fac. of Electr. Eng. Comput., Zagreb Univ., pp: 669-674.

Jacques, F., 2008. Color management for DICOM images considered as TIFF 16. J. Display Technol., 4(4): 410-414.

Jeong-Ki, K., L. Hyung-Seok and K. Heung-Nam, 2006. Dual Journaling Store Method for Embedded Systems. The 8th International Conference Advanced Communication Technology, ICACT 2006, 20-22 Feb. 2006, Embedded Software Res. Lab., Electron. Telecommun. Res. Inst., Daejeon, 2: 1241-1244.

Jong-Hyeon, Y., P. Yong-Hun, L. Seok-Jae, J. Su-Min and Y. Jae-Soo, 2008. Design and implementation of a non-shared metadata server cluster for large distributed file systems. Proceedings of the International Symposium on Computer Science and its Applications, (CSA '08), pp: 343-346.

Juan, P., C. Toni and M.G. Jose, 2007. The design of new journaling file systems: The dualfs Case. IEEE Transact. Comput., 56(2): 267-281.

Jun, G., W. Yi, T. Kit-Sang, C. Sammy, W.M.W. Eric, T. Peter and Z. Moshe, 2008. Evolutionary optimization of file assignment for a large-scale video-on-demand system. IEEE Transact. Knowledge Data Eng., 20(6): 836-850.

Jun, L., D. Bin, Z. Yi, R. Ly and L. Daiw, 2009. A high performance cluster file system with standard network file system interface. Int. Forum Informat. Technol Applicat., 1: 397-400.

Jun, W. and H. Yiming, 2008. PROFS-Performance-Oriented Data Reorganization for Log-structured File System on Multi-Zone Disks. Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems, (MASCOTS '01), IEEE Computer Society Washington, DC, USA, pp: 285.

Jun, W. and H. Yiming, 2007. A novel reordering write buffer to improve write performance of log-structured file systems. IEEE Transact. Comput., 52(12): 1559-1572.

Jun, W., M. Rui, Z. Yingwu and H. Yiming, 2002. UCFS-a novel user-space, high performance, customized file system for web proxy servers. IEEE Transact. Comput., 51(9): 1056-1073.

Kim, G., D. Shin and D. Shin, 2005. Intellectual property management on MPEG-4 video for hand-held device and mobile video streaming service. IEEE Trans. Consumer Electr., 51(1): 139-143.

Li, H., S.R. Cumpson, R. Jochemsen, J. Korst and N. Lambert, 2003. A scalable HDD video recording solution using a real-time file system. IEEE Trans. Consumer Electr., 49(3): 663-669.

Kwon, O., Y. Yunjung and K. Kern, 2008. Swapping Strategy to Improve I/O performance of mobile embedded systems using compressed file systems. Proceedings of the 14th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, (RTCSA '08), IEEE Computer Society Washington, DC, USA, pp: 169-176.

Kwon, S. J. and T.S. Chung, 2008. An efficient and advanced space-management technique for flash memory using reallocation blocks. IEEE Trans. Consumer Electr., 54(2): 631-638.

Kit-Sang, T., K. King-Tim, C. Sammy and W.M.W. Eric, 2001. Optimal File Placement in VOD System Using Genetic Algorithm. IEEE Transact. Indust. Elect., 48(5): 891-897.

Kwangho, C., Jin-sookim and M. Seungryoul, 2008. Snappvfs: Snapshot-able parallel virtual file system. Proceedings opf the 14th IEEE International Conference on Parallel and Distributed Systems, (PDS' 2008), Comput. Sci. Dept., Korea Adv. Inst. of Sci. Technol., Daejeon, South Korea, pp: 221-228.

Lin-Wen, L., S. Peter and V. Radek, 2000. Computer society, file assignment in parallel I/O systems with minimal variance of service time. IEEE Transact. Comput., 49(2): 127-140.

Miller., E., D. Long, W. Freeman and B.. Reed, 2001. Strong Security for Distributed File Systems. IEEE International Conference on Performance, Computing and Communications, 2001, California Univ., Santa Cruz, CA, pp: 34-40.

Man-Keun, S. and L. Seung-Ho, 2010. Deduplication flash file system with PRAM for non-linear editing. IEEE Transact. Consumer Elect., 56(3): 1502-1510.

Man-Keun, S., K. Sungahn, P. Youngwoo and P. Kyu Ho, 2009. NLE-FFS: A flash file system with PRAM for non-linear editing. IEEE Transact. Consumer Elect., 55(4): 2016-2024.

Michael, O. and B. Amnon, 2002. Atomic writes for data integrity and consistency in shared storage devices for clusters. Proceedings of the Fifth International Conference on Algorithms and Architectures for Parallel Processing, (AAPP' 2002), Elsevier Science Publishers B. V. Amsterdam, The Netherlands, 20(4): 539-547.

Minglu, L., Y. Jiadi and W. Jie, 2008. Free-riding on bittorrent-like peer-to-peer file sharing systems: Modeling analysis and improvement. IEEE Transact. Parallel Distribut. Syst., 19(7): 954-966.

Myung-Hoon, C., K. Hong-Yeon, K. June and K. Myung-Joon, 2007. Design and implementation of a file spanning among multiple OSDS in OASIS file system. Int. Conf. Adv. Commu., 1:782-785.

Nowoczynski, P., Stone, Yanovich and Sommerfied, 2008. Zest Checkpoint Storage System for Large Supercomputers. Petascale Data Storage Workshop, (PDSW '08), Pittsburgh Supercomput. Center, Pittsburgh, PA, pp: 1-5.

Pedraza, C., J. Castillo, J.I. Mart nez, P. Huerta and C.S. de La Lama, 2007. Self-Reconfigurable Secure File System for Embedded Linux. Published in IET Computers and Digital Techniques, November 2008, URJC, Mostoles, pp: 461-470.

Philip, C., L. Sam, R. Robert, V. Murali, K. Julian and L. Thomas, 2008. Small-file access in parallel file systems. Proceedings of the 2009 IEEE International Symposium on Parallel&Distributed Processing, (IPDPS '09), IEEE Computer Society Washington, DC, USA, pp: 1-11.

Park, S. and S.Y. Ohm, 2006. New techniques for real-Time FAT file system in mobile multimedia devices. IEEE Trans. Consumer Electr., 52(1): 1-9.

Ragib, H., A. Zahid, Y. William, B. Larry and C. Roy, 2005. Survey of peer-to-peer storage techniques for distributed file systems. Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC' 2005), IEEE Computer Society Washington, DC, USA, pp: 205-213.

Roman, P. and C. Christian, 2007. Cryptographic security for a high-performance distributed file system. Proceedings of the 24th IEEE Conference on Mass Storage Systems and Technologies (MSST), San Diego, CA, pp: 227-232.

Ryosuke, S., N. Hiroshi, H. Yasutake, H. Jiro and A. Makoto, 2010. Cost-Effective 60-ghz antenna package with end-fire radiation for wireless file-transfer system. IEEE Transact. Microwave Theor. Tech., 58(12): 3989-3995.

Stender, J., K. Bj¨orn, H. Mikael and H. Felix, 2010. BabuDB: Fast and efficient file system metadata storage. 2010 International Workshop on Storage Network Architecture and Parallel I/Os (SNAPI), 3-3 May 2010, Zuse Inst. Berlin, Berlin, Germany, pp: 51-58.

Sanders, D.A., L.M. Cremaldi, V. Eschenburg, C.N. Lawrence, C. Riley, D.J. Summers, D.L. Petravick, 2002. Redundant Arrays of IDE Drives. IEEE Trans. Nucl. Sci., 49(4): 1834-1840.

Sang, O., P. Graduate and J.K. Sung, 2009. An efficient multimedia file system for NAND flash memory storage. IEEE Trans. Consumer Electr., 55(1): 139-145.

Sang, S.L. and H.P. Kyu, 2007. TPF: TCP Plugged File System for Efficient Data Delivery over TCP. IEEE Transact. Comput., 56(4): 459-459.

Seung-Ho, L. and P. kyu-Ho, 2006. An efficient NAND flash file system forflash memory storage. IEEE Transact. Comput., 55(7): 906-912.

Seunghwan, H., B. Hyokyung and K. Kern, 2007. lecramfs: An efficient compressed file system for flash-based portable consumer devices. IEEE Trans. Consumer Electr., 53(2): 481-488.

Seungtaek, O., K. Jin-Soo, K. Ki-Sok and L. Joonwon, 2005. Closed P2P system for PVR-based file sharing. IEEE Transact. Consumer Elect., 51(3): 900-907.

Shun-Fa, Y. and W. Chin-Hsien, 2009. A low-memory management for log-based file systems on flash memory. Proceedings of the 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, (CSA' 2009), IEEE Computer Society Washington, DC, USA, pp: 219-227.

Song, J., Z. Xuechen, L. Shuang and K. Davis, 2010. Improving networked file system performance using a locality-aware cooperative cache protocol. IEEE Transact. Comput., 59(11): 1508-1519.

Taizhong, Q., Y. Dylan and W. Youjip, 2010. CMFS: Compressed Metadata File System For hybrid storage. IEEE International Conference on Network Infrastructure and Digital Content, 24-26 Sept. 2010, pp: 1030-1034.

Thanos, M., K. Yannis, M. Manolis, D.F. Michail and B. Angelos, 2010. ZBD: Using transparent compression at the blocklevel to increase storage space efficiency. 2010 International Workshop on Storage Network Architecture and Parallel I/Os (SNAPI), 3-3 May 2010, Inst. of Comput. Sci. (ICS), Found. for Res. Technol.-Hellas (FORTH), Heraklion, Greece, pp: 61-70.

Vijayan, P., A.C. Arpaci-Dusseau and R.H. Arpaci-Dusseau, 2005. Model-based failure analysis of journaling file systems. Proceedings of the 2005 International Conference on Dependable Systems and Networks, (DSN '05), IEEE Computer Society Washington, DC, USA, pp: 802-811.

Wei, L., M. Wu, X. Ou, W. Zheng and M. Shen, 1999. Design of an I/O balancing file system on web server clusters. 2000 International Workshops on Parallel Processing, Dept. of Comput. Sci., Tsinghua Univ., Beijing, PP: 119-125.

Weisong, S., S. Sharun and L. Hanping, 2004. An adaptive distributed file system for heterogeneous network environments. Proceedings of the Tenth International Conference on Parallel and Distributed Systems (ICPADS), Wayne State Univ., Detroit, MI, USA, pp: 145-152.

Won, Y., S. Yang and S. Kang, 2006. Harmonic Data Placement: File System Support for Scalable Streaming. IEEE Trans. Consumer Electr., 52(3): 811-818.

Xiong, J., Y. Hu, G. Li, R. Tang and Z. Fan, 2011. Metadata distribution and consistency techniques for large-scale cluster file systems. IEEE Transact. Parallel Distribut. Syst., 22(5): 803-816.

Youngwoo, P. and P. Kyu Ho, 2011. High-performance scalable flash file system using virtual metadata storage with phase-change RAM. IEEE Transact. Comput., 60(3): 321-334.

Yui-Wah, L., L. Kwong-Sak and S. Mahadev, 2002. Operation shipping for mobile file systems. IEEE Transact. Comput., 51(12): 1410-1422.

Yurong, X.U., S. Shouqian and P. Yunhe, 2001. Research of The Mobile-Code-Based File System for Portable Information Device. Proceedings of the 2001 International Conference on Computer Networks and Mobile Computing, (ICCNMC'01), IEEE Computer Society Washington, DC, USA, pp: 441.

Zhang, H., T. Jiang, Z. Gu and S. Zheng, 2004. Design and implementation of broadcast file system based on DSM-CC data carousel protocol. IEEE Trans. Consumer Electr., 50(3): 929-933.

Zuji, M. and D. Christos, 2004. A distributed database architecture for global roaming in next-generation mobile networks. IEEE/ACM Transact. Network., 12(1): 146-160.