PART-A

SHORT QUESTIONS WITH SOLUTIONS

Q1. Define file system. Model Paper-I, Q1(f)

Ans: File system serves as a source of operating system. It helps in identifying the data located on disk. It specify the way how data can be saved, stored and retrieved on a disk. It is considered as a data structure/method used by operating system.

A file system is basically used for,

1. Keeping records of files on disks/partitions.
2. Organizing the files on disks.
3. Partitioning the files stored on disk.
4. Accessing data present on disk.

A file system consist of collection of directories. Where each directory contains information about a group of files/multiple files. Modern high performance system opt the following three classes of file system.

1. Network File Systems (NFS)
2. Storage Area Networks (SAN)
3. Parallel File Systems (PFS).

Q2. Write short notes on,

(i) Cell storage

(ii) Journal storage

(iii) Database.

Ans:

(i) Cell Storage: Cell Storage defines storage of fixed sized cells. Each cell contains atleast one object in it. It describes the physical organization of primary and secondly storage media. In computer systems primary memory is arranged as array of memory cells whereas secondary memory is arranged in the form of block sectors. Read/write operations performed on each block is referred as a unit.

(ii) Journal Storage: Journal storage stores composite objects like records. These objects contain multiple field. This type of storage includes a journal manager and cell storage.

(iii) Database: A database is an integrated, shared collection of data. It consist of both 'End-users' data, (which includes raw-, facts of interest) and Meta-data (which includes data about data). It is usually so large that it has been stored on storage devices like disks/tapes.

Q3. List the two desirable properties of sprite network file system. Model Paper-IV, Q1(f)

Ans: The two desirable properties of sprite network file system are,

1. Delay-write-black
2. Write-through.

1. **Delay-write-back:** Delay - write back arises when there is a time delay (of seconds) while writing block of cache on disks. Such situation can lead to data loss at the time of disk failure
2. **Write-through:** Write through ensues reliability while writing data on the disk but at same time increases the write operation time.

Q4. Write short notes on general parallel file system.

Ans: Parallel file system substantiate concurrent access to single file. That is, parallel file system allow multiple clients to perform read/write operation on a single file simultaneously. This file system uses same Program, Multiple Data (SPDM) paradigm to allow Concurrency control is considered as one of the critical issue of parallel file system.

General parallel file system was designed based on the concept of parallel file system. This file system employs the characteristics of general purpose POSIX system which runs on single system.

This file system was developed for optimizing the performance of large clusters. It is designed in such a way that it can support large file system of 4PB having 4096 disks of 1 terabyte and maximum file size of about $(2^{63} - 1)$ bytes.

Q5. Define Big table storage system and Apache hadoop.

Ans:

Big Table Storage System: It is a distributed storage which allow users to store large amount of data. This storage system was developed by Google. It employs Google File System for storing users data and system information. In addition to this, it employs distributed lock service on directories/files in order to ensures atomic read/write operation.

Apache Hadoop: Apache Hadoop is a Java based software system. It is an open source software that are used in data-intensive applications like marketing analysis, machine learning, image processing, web crawling etc. It is capable of handling large volume of data across distributed applications.

Q6. Write short notes on Google File System and Megastore.

Ans: Model Paper-II, Q1(f)

Google File System: Google file system uses large storage system that are developed using inexpensive components. This storage system facilitates the storage of huge amount (i.e., perabytes) of data. This file system mainly focus on ensuring system reliability in case of hardware failure, software errors, application errors and human errors.

Megastore: Megastore is a highly scalable distributed storage system. This system provides various online services. This system is distributed across multiple data centers with a storage capacity of about perabytes. It is mostly used by Google for handling huge number of transactions (i.e., upto billion of transactions).

Q7. What is the role of a master in Google file system cluster. Model Paper-III, Q1(f)

Ans: A master is responsible for maintaining state information of all the system components. This information includes metadata about,

1. File name
2. Location of the replicated files on each chunk.
3. Access control information
4. State information of each chunk server

Besides this, it also manages multiple chunk servers. A master contain the location of each chunk control structure in its memory. Due to this, a master can have updated information regarding the chunk location. This updations are done each time the system gets started or when a new chunk server participates the cluster.

To maintain system reliability, an operations logs are maintained by the file system. This logs keeps a records of updated metadata information which are later reused by master in case of system failure. In addition to this, it also stores the replicated copies of metadata on persistent storage. To recover from fault tolerance and system failure. The operation logs are replayed However, the recovery time gets reduce when the master repeatedly check points it state. And once the last check point is made the recovery time replays the log records 1.

PART-B

ESSAY QUESTIONS WITH SOLUTIONS

6.1 EVOLUTION OF STORAGE TECHNOLOGY

Q8. Explain about the evolution of storage technology.

Ans: A cloud application demands huge amount of storage as well as computing cycles inorder to storage and process cloud data. The evaluation of storage technology begin in the year 1986. In this year, 2.6EB (A CD-ROM less than or equal to 730 MB) was used for storing data per computer. However in the year 1993, 15.8EB of storage (whose capacity was equivalent to 4CD-ROMS) was provided to the users. Later a storage of about 54.5EB (Whose capacity equal to 12 CD-ROMS) was provided to users in the year 2000. Whereas in the year 2007 a storage of 295EB (whose capacity was equal to 61 CD-ROMS) was provided to the users.

A recent study made in the year 2003 reveals that during the year 1980 – 2003 storage with respect to processor technology has increased the capacity of Hard Disk Drives (HDD) density by magnitude of four i.e., from 0.01 GB to 100 GB/in^2. Later in the year 2011 the HDD density was increased to 744 GB/in^2 and in the year 2016 it reached to 1,800 GB/in^2.

In the year 2003, Dynamic Random Access Memory (DRAM) increased from GB/in^2 to 100 GB/in^2. This lead to an increase in overall storage cost. Later in the year 2010 the Samsung introduce the 4 gigabit LPDDR2 DRAM ie.., Low-power, double-data-route DRAM which uses 30 nm process. Due to the rapid advancement in technology, huge variations where encounter between initial investment cost and the system management cost. Thus this leads to the development of centralized storage system for cloud. Eventhough the centralized storage have automated various storage management functions like data replication, backup recovery etc. to substantially minimize the cost of storage management it leads to performance degradation.

Hence to overcome the above issues, large scale distributed storage systems like network file system, Andrew file system, sprite file systems. Google file system and megastore where evolved in year 1980. This storage system uses off-the shell components and allow,

1. Storage of huge amount of data on cloud.
2. Process or manage large volume of wood data and
3. Perform intensive computation of huge amount of data.

The storage system was designed to ensure high performance (at any cost) as well as high reliability at low cost.

6.2 STORAGE MODELS, FILE SYSTEMS AND DATABASE

Q9. Explain the various storage models.

Ans: Model Paper-I, Q7(a)

Storage Models: Storage model defines data structure in physical storage like local disks, removable media and storage. Whereas a data model describes the logical aspects of data structure in a database.

The two abstract storage models are as follows,

1. Cell storage
2. Journal storage.

1. Cell Storage: Cell Storage defines storage of fixed sized cells. Each cell contains atleast one object in it. It describes the physical organization of primary and secondary storage media. In computer systems primary memory is arranged as array of memory cells whereas secondary memory is arranged in the form of block sectors. Read/write operations performed on each block is referred as a unit.

The two desirable properties of storage model especially of cell storage are as follows,

(i) Read/write coherence

(ii) Before-or-after atomicity.

(i) **Read/Write Coherence:** Read/write coherence arises when the result of read operation performed on memory cell does not match whit the recent write operation on memory cell.

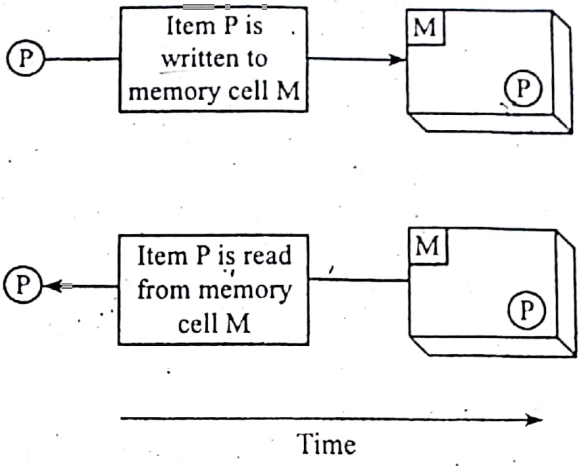


Figure: Read/Write Coherence

(ii) **Before-or-After Atomicity:** Before-or-after atomicity occur when the result of current read/write operation does not matches with the previous or next read/write operation.

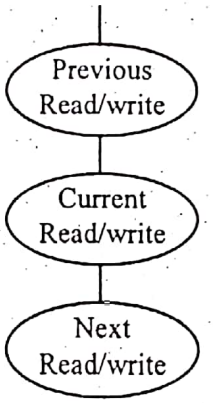


Figure: Before-or-After Atomicity

2. **Journal Storage:** Journal storage stores composite objects like records. These objects contain multiple field. This type of storage includes a journal manager and cell storage.

(i) **Cell Storage:** It is used for storing the history of variable instead of storing the current value.

(ii) **Journal Manager:** A journal manager allow users, to interact with the cell storage. To perform this interaction a user must send request to a journal manager inorder to

(a) Initiate a new action

(b) Perform read operation on cell value

(c) Perform write operation on cell value

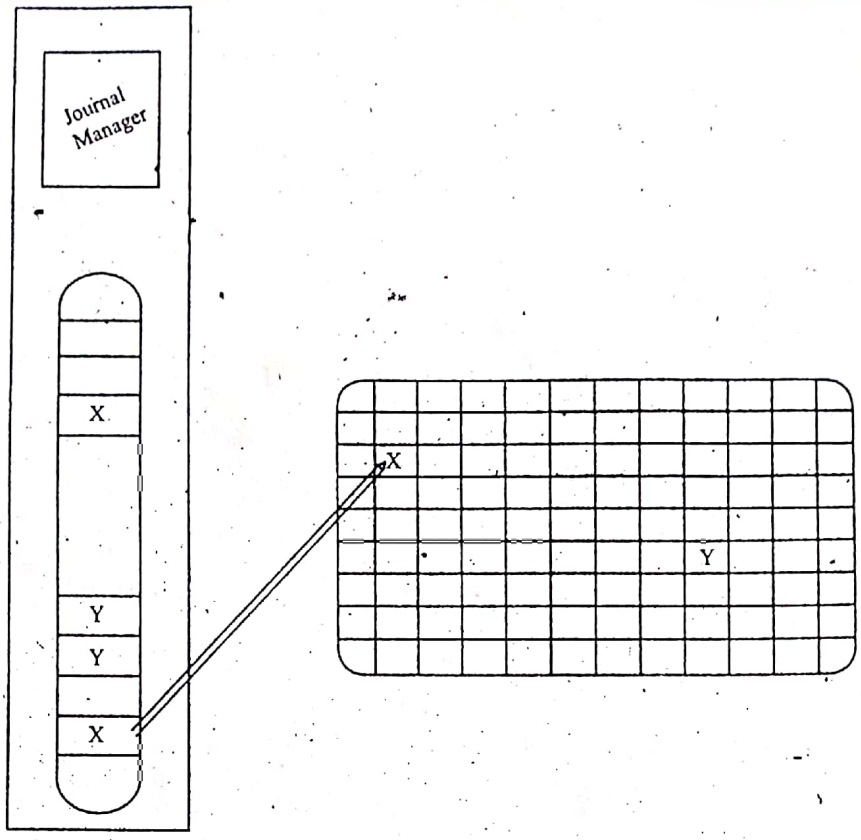
(d) Perform commit operation

(e) Stop a action.

Upon receiving a request, the journal manager, uses the following command inorder to allow users to interact with cel storage.

- **Read:** It allow users to perform 'Read' operation on a cell.
- **Write:** It allow users to perform 'write' operation on a cell.
- **Allocate:** It allow users to perform allocation on a cell.
- **Deallocation:** It allow users to perform deallocation.

In storage system, the cell storage maintains a 'log' for storing the history of variables. This log is basically present o the non volatile storage media of journal storage. This log contain a record which include information about updated data ite appended at the end of it. The information contained in the log can later be recovered/reused at the time of system failure. Th ensures that all the actions present in the log follow All-or-Nothing action. That is using this action helps in maintaining a ne value initially in a log of journal storage and later overwrite it on the cell storage. This sceneries is shown in figure below,



Figure

Q10. Write short notes on file system and database.

Ans:

File System: File system serves as a source of operating system. It helps in identifying the data located on disk. It specify the way how data can be saved, stored and retrieved on a disk. It is considered as a data structure/method used by operating system.

A file system is basically used for,

1. Keeping records of files on disks/partitions.
2. Organizing the files on disks.
3. Partitioning the files stored on disk.
4. Accessing data present on disk.

A file system consist of collection of directories. Where each directory contains information about a group of files/multiple files. Modern high performance systems opt the following three classes of file system.

(i) Network File Systems (NFS)

(ii) Storage Area Networks (SAN)

(iii) Parallel File Systems (PFS).

(i) Network File System: It is the most popularly used distributed file system developed by SUN Microsystem. This file system uses the basic client/server model inorder to share applications and also to share file systems among multiple clients connected via LAN.

However, this file system deals with various issues like scalability, reliability and system failure.

(ii) Storage Area Networks: It is a networking technology that connects storage systems with the computation servers. This storage system is said to be centralized storage as it maintains a storage pool which allow storage allocation based on the servers need.

A SAN-based file system is considered as expensive as it require additional hardware and software support for pooling.

(iii) Parallel File Systems: It is a distribute file system that allow multiple client to access same file. It make use of global name spacing mechanism to distribute a file across multiple I/O nodes. Here I/O nodes are used for forwarding data to all computation nodes. A parallel file system also maintains a metadata server inorder to maintain information about data stored on I/O nodes.

Database: A database is an integrated, shared-collection of data. It consist of both 'End-users' data, (which includes raw facts of interest) and Meta-data (which includes data about data). It is usually so large that it has been stored on storage devices like disks/tapes.

A Database Management System (DBMS) is a software that defines a database; stores the data, support a query language, produce reports and manages the process made to data in database.

Cloud applications used database as an interface to interact with file system.

The data-intensive cloud application supports database models like navigational model, relational model object oriented model. But it does not support SQL query language. The cloud database support NOSQL model. It does not ensures ACID properties.

6.3 DISTRIBUTED FILE SYSTEMS

Q11. Explain in detail about network file systems and UNIX file system. Model Paper-I, Q7(b)

Ans:

Network File System (NFS): It is the most popularly used distributed file system developed by Sun Micro System. This file system used basic client/server model in order to develop applications and also to share file system among multiple clients connected via Local Area Network (LAN).

Network File System (NFS) was designed based on the designing philosophy of UNIX File System (UFS). A Network file system offers the following benefits

1. It defines semantics similar to UNIX file system. This provides compatibility with the present file system.
2. It allows easy integration with existing file system.
3. It creates multi-clients environment. This guarantees the wide usage of all the system that connected to a network.
4. It accepts optimal level of performance degradation while providing remote access across the network containing high bandwidth.

Unix File System: Unix file structure (or Unix file system) plays a vital role in understanding most of the other Unix commands. Unix treats all of it's utilities as a 'file'. This file structure when pictorially represented, resembles, an upside down tree, with root directory claiming the peak position and various other sub-directories spreaded below it. A snapshot of such representation is as follows.

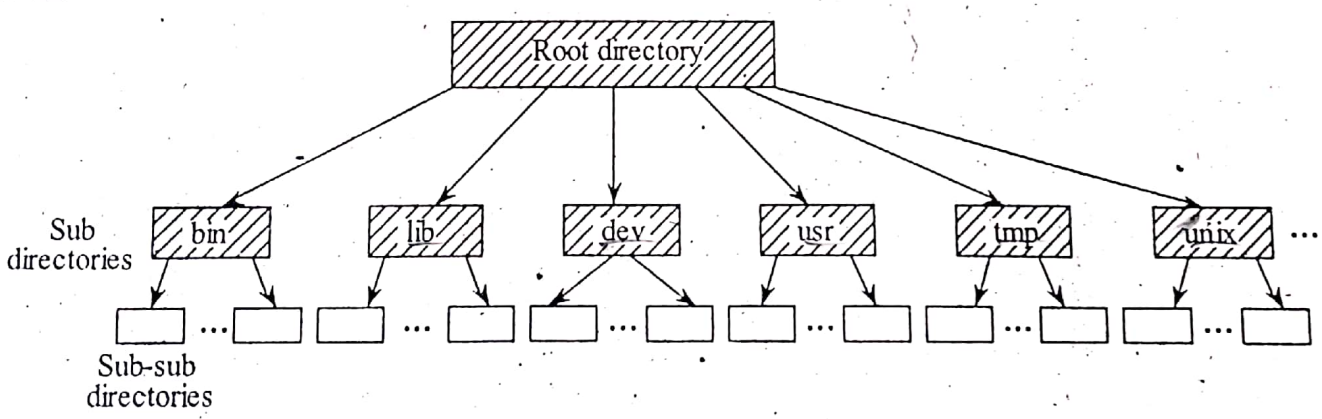


Figure: Unix Directory/File Structure (A Snapshot of Upside Down Tree)

A directory is a collection of related files hence a directory can be called as a file.

Main criteria behind such representation is to place all the relative files in one directory hence causing ease in management.

Features of Unix File System

(i) Supports hierarchical file structure

(ii) The dynamic grow of files

(iii) File access permissions

(iv) All devices are treated as files

(v) File sharing.

Characteristics of UNIX File System

1. The layered design of UNIX file system provides a flexible file system. The concept of layering minimizes the interaction between those modules that are necessary for implementing the system. Besides this, the use of unode layer in UNIX file system grants uniform excess right on both local and remote files.
2. The hierarchical design of UNIX file system facilitates a scable file system. The hierarchical design group multiple files into a single file called director. A hierarchal file system can have multiple (or) collection of files or directories.
3. The metadata contained in the UNIX file system describes the systematic view of the file system metadata contains information about the
   (i) Owner of the file
   (ii) Access rights granted to the user
   (iii) Time of file creation
   (iv) Last modifications made on the file
   (v) Size of the file
   (vi) Structure of the file
   (vii) Persistent storage device cells.
4. The inode layer present on the persistent storage media includes information about the files and directories.

Q12. Explain the layered design of UNIX file system.

Ans: Model Paper-II, Q7(a)

Layered Design of UNIX File System: The layered design of UNIX file system is shown in figure below,

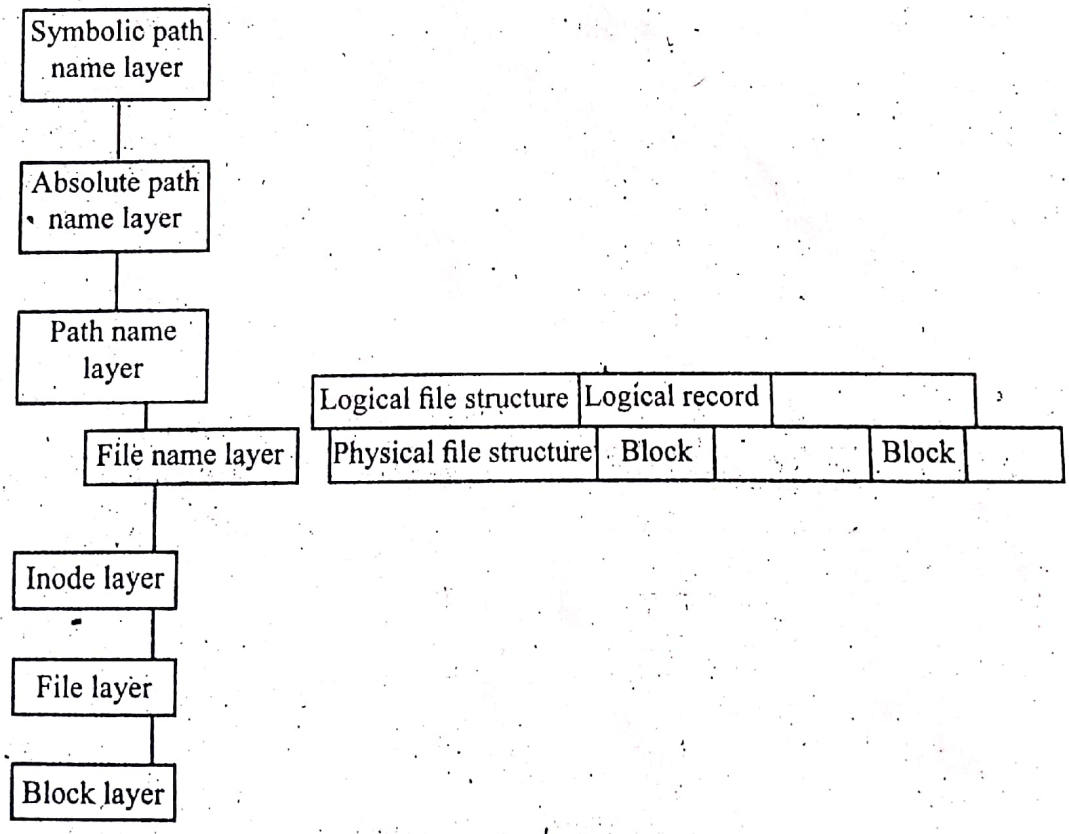


Figure: Layered Design of UNIX File System

In, layered design of UNIX file system the physical file structure is separated from that of the logical file structure.

Physical File Structure: The physical file structure of UNIX file system represent the storage model. This model provides description regarding the storage of file in a storage media. A physical files structure stores files on physical device called physical records/blocks.

Logical File Structure: The logical file structure of UNIX file system represent the data model. This model provides abstrac view of the data. A logical file include logical records which is nothing but a linear array.

The layered design of UNIX file system consist of the following seven layers.

1. Block layer
2. File layer
3. Inode layer
4. Path name layer
5. Absolute path name layer
6. Symbolic path name layer.

The block layer, file layer and inode's layer are the lower three layer of UNIX file system. These layer defines the physical organization of the file system. Where as the path name layer, absolute path name and symbolic path name are the upper three layers of the UNIX file system. These layer represent the logical organisation of file system. The file name act as a mediator between the upper and lower layers. The seven layers of UNIX file system are discussed below.

1. **Block Layer:** This layer is responsible for allocating physical records/block on physical devices.
2. **File Layer:** This layer is responsible for organizing blocks into files.
3. **Inode Layer:** This layer defines metadata for objects like files and directories.
4. **File Name Layer:** This layer act as a mediator between the upper and lower layers. It provides the machine oriented as well as the user oriented views of file system.
5. **Path Name Layer:** This layer depicts the files/directories location.
6. **Absolute Path Name or Full Path Layer:** This layer depicts the file location relative to root directory irrespective of the current directory/working directory of the process.
7. **Symbolic Path Name/Relative Path Name Layer:** This layer depict the file location in relation to current or working directory.

Beside this, the layered design of UNIX file system also maintains a file description as an index inorder to uniquely identify the local files, a file description table inorder to add new entries to the file system and a open file table inorder to meta data information.

Q13. Briefly discuss about network file systems architecture.

Ans: Network file system uses client-server paradigm to share file system among multiple clients connected by Local Area Network (LAN). In NFS client-server interaction, a client basically runs on local host whereas a server runs on remote file system. This file system uses local file systems Application Programming Interface (API) inorder to differentiate file operation of local file from remote file and later invokes the Remote Procedural Call (RPC) client as a response. An API calls are provided by users on remote file system and Network File System servers in response to RPC calls.

Figure below shows the NFS client-server interaction.

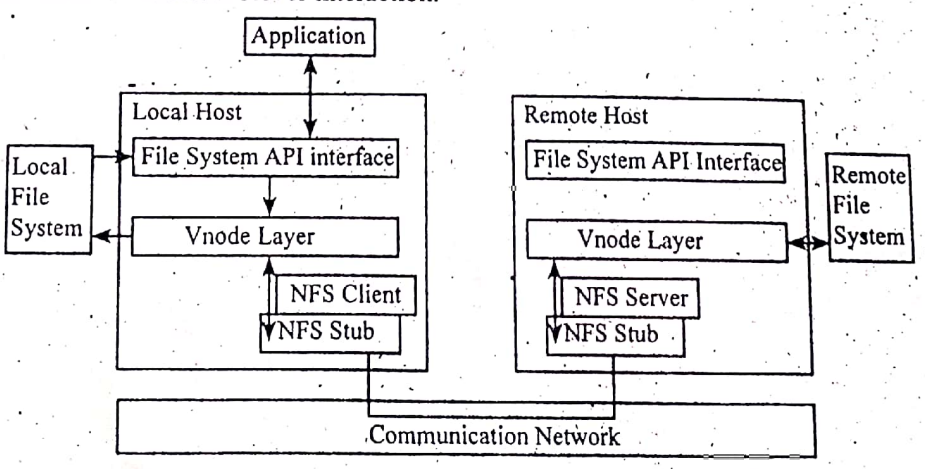


Figure : NFS Client – Server Interaction

In the above architecture, Vnode layer is responsible for implementing file operations on local or remote files and also for differentiating local files from remote file. A local file system includes all the operations of local file and remote file system includes all the operations of remote files and NFS.

A File Handle (FH) is used for uniquely identifying remote files. It contains a internal name of 32-bytes file system identification, inode number and a generation number. It allow system to identify remote file system and the corresponding file on the system. The generation number of the file handle facilitate the system in reusing the inode numbers. Apart from this, the generation also guarantees to provide accurate semantics when many clients simultaneously operates on a single remote file.

In NFS client-server package, a client basically forwards the information regarding the target to NFS server. The NFS server inturns forward this information to vnode layer present on the remote host. Which later directs it on remote file system.

Since, multiple RPC calls like 'read' are repeatedly perform. This may leads to communication failure. However, if network failure occur while delivering response to read RPC then in such cases calls can repeated. But if network failure occur while delivering response to RPC then in such cases repetition of calls delivers an error code to users.

Q14. Discuss in detail about Andrew File System (AFS). Model Paper-II, Q7(b)

Ans: Andrew File System is distribute file system which was designed to interconnect large workstations with relatively small servers. In this file system, the collection of server forms a "vice" framework. The large workstations runs on 4.2 BS UNIX operating system. This operating system is responsible for verifying file system calls and propagating it to user-level process. This process is called 'Venus'. It (venus) is responsible for accepting cached files from vice and for storing the modified file copies back on the requested servers. Venus can communication with the vice only when the open/close operations are performed on file. Where read/write operations can be performed directly on cached copy without any interaction of venus.

The design of Andrew file system significantly focus on performance, security and simple file management process. This file system used local disk of workstation as a persistent cache. Hence this helps in updating the masters file copy present on a single server when it is modified. This strategy helps in reducing the load on sever, enhances the system performance, ensure scalability and minimizes the response time.

To provide enhance security, this system employs a encyption mechanism between communicating clients and server. It also provides a secure network connection while performing file operations like read/write.

In addition to this, Andrew file system make use of Access Control Lists (ACLs) inorder to allow efficient data sharing. An access control lists helps in granting access rights to a single/multiple uses. Apart from this, the design of AFS provides location transparency. That is, it automatically allow users to access file from any location.

Q15. Explain about sprite network file systems.

Ans:

Sprite Network File System (SFS): Sprite Network File System (SFS) is a distributed file system. It is an essential component of Sprite Network operating system. It is basically designed to provide non-write-through file caching on both client and server systems.

Sprite network file system allow processes working on multiple workstation to use file access semantics similar to that of processes running on single system. This is due to the fact that cache consistency mechanism, frees the portion of cache memory and then disables the caching for shared files when read/write operation is being performed.

The concept of caching helps in,

(i) Decreasing network latency
(ii) Reducing the serves utilization
(iii) Improving the performance of the system.

Besides this, SFS also satisfy the file access request made by the clients process at various levels. This can be done by directing the request to local and remote servers.

Initially the file access request is passed to the local cache and local client file system. If the request is not satisfied locally then it is passed to the remote servers cache and then to serves file system.

Sprite file systems are said to be 30-35% faster than the network file system and Andrew file system. In this system, file cache is arranged as fixed size block of 4KB.

Each cache block is associated with a virtual address. The virtual address includes a unique file identifier and a block number in a file. Virtual addressing is done inorder to enable multiple clients to create new cache blocks without interacting with the server. Here, the role of server is to provide mapping between the virtual addresses and the physical disk addresses.

In sprite systems, cache size available of client and server system gets dynamically changed as needed. This is due to the fact that sprite operating system guarantee. The optimal usage of memory between virtual memory and file caching.

In addition to this, the SFS also uses physical memory pages inorder to keep tack of the last access made on the each block or page. Beside this, Least Recently Used (LRU) page replacement algorithm is also implemented inorder to record the time taken to perform read and write operation.

The two desirable properties of sprite network file system are,

1. Delay-write -black
2. Write-through.

1. **Delay-write-backs:** Delay-write back arises when there is a time delay (of seconds) while writing block of cache on disks. Such situation can lead to data loss at the time of disk failure.
2. **Write-through:** Write through ensues reliability while writing data on the disk but the same time increases the write operation time.

6.4 GENERAL PARALLEL FILE SYSTEMS

Q16. Explain about general parallel file systems.

Ans: Model Paper-III, Q7(a)

Parallel file system substantiate concurrent access to single file. That is, parallel file system allow multiple clients to perform read/write operation on a single file simultaneously. This file system uses Same Program Multiple Data (SPDM) paradigm to allow concurrent access to the single files. Concurrency control is considered as one of the critical issue of parallel file system.

General parallel file system was designed based on the concept of parallel file system. This file system employs the characteristics of general purpose POSIX system which runs on single system.

This file system was developed for optimizing the performance of large clusters. It is designed in such a way that it can support large file system of 4PB having 4096 disks of 1 terabyte and maximum file size of about $(2^{63} - 1)$ bytes.

In GPFS, a file is basically divided into fixed sized block which ranges between 16KB and 1MB. These files are striped on multiple disks. The configuration of general parallel file-system is shown below,

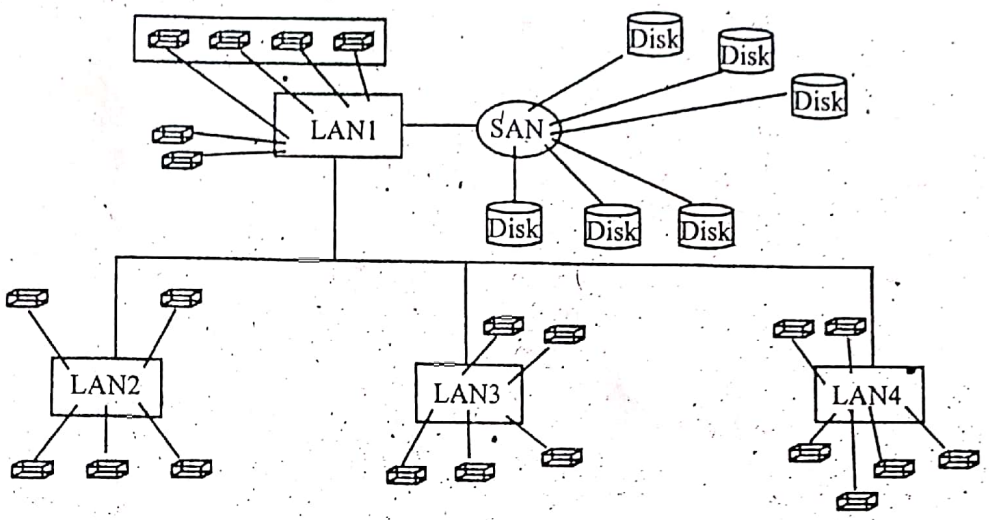


Figure: GPFS Configuration

Characteristics: General parallel File Systems posses the following characteristics.

1. It allow directories to make use of extensible hashing techniques to access rights to a file.
2. It make use of inodes and indirect blocks inorder to maintain file attributes and data block addresses.
3. It maintains a record of metadata updates in a write - ahead logs inorder to address system failure. A write a head log uses persistent storage media to maintain the updated log record.
4. It make use or RAID devices and dual attached RAID controller which stripes large files to fixed size block so as to allow concurrent access to a file.
5. It maintains a replicated copy of data files and meta data files on two physical disk to improve the fault tolerance of the system.
6. It uses distributed locking mechanism inorder to ensure data consistency and data synchronization. This mechanism employs a central lock manager which grants lock taken to every local manager running on their respective I/O node.
7. It offers lock granularity by applying the following two techniques.

(i) **Byte-range Tokens:** It is used for performing read/write operations on data file. This operation is carried out in the following manner.

Initially the first node/start node obtains a token [between 0 to ∞] and attempt to write to a file. This node can simultaneously perform all its subsequent read/write operations until the attempt to perform read/write operation is done by another node. On same file if such situation can restrict the token range provided to the first node. That is, if offset of first node for performing write operating is P_1 and if offset of the second node is P_2 then range of the token will be $[0, P_2]$ and $[P_2, \infty]$ respectively. Hence the two nodes are now allow to simultaneously access the file.

(ii) **Data Shipping Method:** This method offer a file-grained data sharing among multiple nodes to obtain similar files. This method uses round-robin algorithm to control the equal sized file blocks. Read/write operations are controlled by the target block.

Here, token manager is responsible for

(a) Maintaining token states
(b) Creating a token
(c) Distributing a token
(d) Collecting a token after the file gets closed.
(e) Upgrading a token when addition requests are made by node for file access.

8. It synchronizes the metadata access.
9. It employs disk mapping mechanism to manage the disk space. This mechanism partitions the disk maps into *n* regions and stores them on distinct I/O nodes. Thereby allowing several node to use the disk space at same time.

6.5 GOOGLE FILE SYSTEM

Q17. Discuss about storage system used in Google File Systems. Model Paper-III, Q7(b)

Ans: Google file system uses large storage system that are developed using inexpensive components. This storage system facilitates the storage of huge amount (i.e., perabytes) of data. This file system mainly focus on ensuring system reliability in case of hardware failure, software errors, application errors and human errors. Google file system was designed after analysis the following critical aspects of file and access models.

1. Scalability and reliability are the two major concerns that must be considered while designing a file system.
2. Range of the file size must be between Giga bytes to Terra bytes.
3. Append operation must be implemented on existing file system.
4. Sequential read operations must be performed on files.
5. Response time must be considered while processing large chunks of data.

Hence based on above analysis the following design decision were made,

1. Large files where segmented into huge chunks.
2. Append operation was implemented to allow multiple application to append the same file.
3. Caching was avoided at client side. Since maintaining cached copy at client side may lead to inconsistency and may include overhead thereby decrease the performance of system.
4. Large cluster where build across high bandwidth without using interconnected network of low-latency.
5. Guarantees data consistency by providing channeling on complex file operations via master. A master is a component of Google file system cluster. It is responsible for controlling the complete file system.
6. Reduces the 'masters' involvement while performing file access operations like read/write. This helps in eliminating hot spot contention problem which inturn guarantees scalability.
7. Checkpointing and fast recovery mechanism was introduced.
8. Garbage - collection mechanism was efficiently supported.

Q18. Briefly discuss the storage architecture of GFS.

Ans:

Architecture of Google File System: The architecture of Google File System (GFS) cluster is shown below,

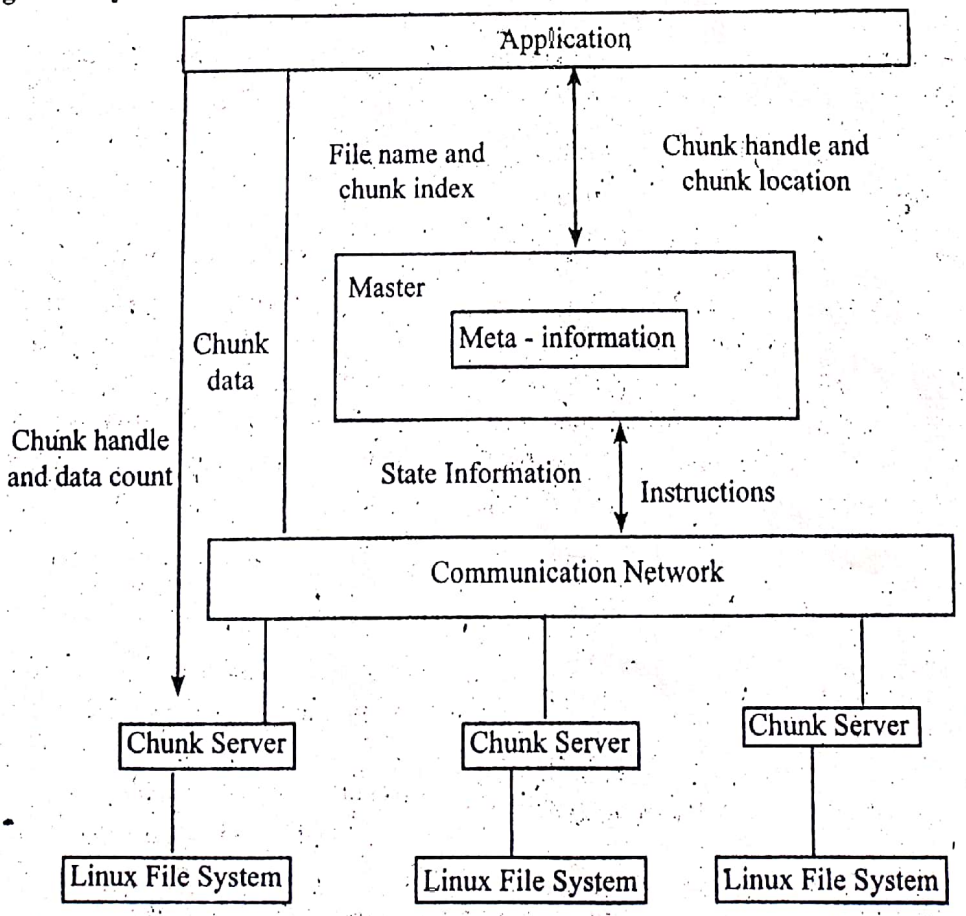


Figure: GFS Cluster

Google File System (GFS) files includes a set of fixed-sized segment of equal sized. These segments are referred to as chunk. Each chunk contains a 64 KB block where each block comprises a 32-bit checksum. Each chunk is associated with a unique chunk handler and are stored on Linux file system. In GFS, files are replicated and stored on multiple site.

GFS support a chunk size of about 64 MB. A large chunk size tend to,

1. Optimize the performance of large files.
2. Minimizes the storage of metadata on to the file system.
3. Maintains persistent network connection with chunk servers.
4. Minimizes the requests made for locating the chunk.

The Google File System cluster make use of the following two components.

(i) Master
(ii) Chunk Server.

Master: A master is responsible for maintaining state information of all the system components. This information includes metadata about,

1. File name
2. Location of the replicated files on each chunk.
3. Access control information
4. State information of each chunk server

Besides this, it also manages multiple chunk servers. A master contain the location of each chunk a control structure in its memory. Due to this, a master can have updated information regarding the chunk location. This updation are done each time the system gets started or when a new chunk server participates the cluster.

To maintain system reliability, an operations logs are maintained by the file system. This logs keeps a records of updated to metadata information which are later reused by masters in case of system failure. In addition to this, it also stores the replicated copies of metadata on persistent storage. To recover from fault tolerance and system failure, the operation logs are replayed. However, the recovery time gets reduce when the master repeatedly check points it state. And once the last check point is made the recovery time replays the log records 1.

Chunk Server: A Chunk Server is stored on Linux File System. It accepts the metadata information provided by the master to irectly communicate or interact with the applications inorder to perform desired file operation. It uses the status information to respond the Master.

In GFS, access to a file is provided in the following manner.

1. Initially a file name, chunk index and the offset of the file for performing read/write operation is send by client application to the master.
2. Upon receiving the request, the master responds to the application by specifying the chunk handle and location of the chunk.
3. Using the above information provided by the master, the application can directly interact with the chunk server to perform the desired file operation.

The GFS cluster architecture, is considered as a scable and effective consistency model. In this model, a master is responsible for carrying out atomic operations like file creation. This file system is said be scable, to make this architecture scalable, a master will contribute less in file mutation and also in frequently occurring operations like write/append. In such situations, a particular chunk is provided to primary chunk server. This server specifies the serial order for the updated chunk.

In this architecture data flow and control flow are de coupled. This helps in achieving efficiency while carrying out various file operations especially the write operation. The below steps illustrate how write request is carried out by decoupling the control flow from data flow.

1. Initially the client communicate with the master by allocating a lease to single chunk server for a specific chunk when there is no lease for that particular chunk. Upon assigning, the master generates response with ID of both primary/secondary chunk server which has the duplicate chunks. Thus the received information is cached by client.
2. The client sends the data as well the replicated copy of chunks to chunk server. Each chunk server uses internal LRU buffer to store the data and send an acknowledgment to the client.
3. The write request is forwarded from client to the primary chunk server after receiving the information from each chunk server that holds replicated copies of chunk. The primary chunk server finds the file mutation using the consecutive sequence number.
4. The write request is now forwarded for primary chunk server to secondary chunk servers.
5. The secondary chunk servers assign mutations to the file based on the sequence number and then sends an acknowledgment to primary chunk server.
6. Upon receiving an acknowledgment from secondary chunk server the primary chunk server responds to the client.

The GFS provides efficient check pointing procedure depending on the copy-on-write mechanism. It make use of lazy garbage collection in order to free the memory space after deletion.

Besides this, the master perform periodic scanning on the name spaces present on the file system and then deletes the metadata files of the less recently used/hidden older file name. In addition to this, a chunk server also maintains a checksum of locally store chunks. This ensure data integrity at the time of data loss and system future.

Google File System can be implemented on cloud store. It is an open-source software written using C++ programming language. It also supports to Java and Python programming language.