Important Topics (questions could come in form of write short note)

- File sharing semantics
- File caching schemes
- File replication
- Fault tolerance
- Atomic transaction

1. File sharing semantics

A shared file may be simultaneously accessed by multiple users. Thus most important aspect while designing the DFS is define good file sharing semantic s among multiple user a accessing same file at the same time.

According to Levy and Silberschatz, 4 types of semantics are defined:

i. Unix semantics – this semantics enforce s an absolute time ordering on all operation and ensures that every read operation on the file see the effect of all previous write operation performed on that file. It means any write operation on open file by one user immediately become visible to the open file used by another user at the same time.

   Unix semantics is basically used in single processor system. In distributed file system, a more relaxed method is used. It uses lock mechanism while sharing files.

ii. Session semantics – In this semantics, generally client opens a file perform a series of read/write operation on the file and close the file. A session is series of file access made between open and close operation. In this semantics, updates made by one client A is visible to client A with open session while others cannot see the changes. Once

the client A who has made update closes the session, then other client who opened the file from start session then they will get the updated file. Already opened file will not reflect the changes. Multiple clients are allowed to read/ write file simultaneously. But for updates once they closes the session, then only the files are updated. Else every client has its own stale copy of file.

iii. Immutable shared files semantics – this semantics uses the immutable file model. Immutable files cannot be updated but can be only delete d once created. Hence if any files are changed, then the new version of file is created. Hence this semantics allows files to be shared only in read mode.

iv. Transaction like semantics – this semantics uses transaction like mechanism which is high level controlling concurrent access to shared mutable file. The transaction mechanism is a set of operation between begin- transaction and end- transaction. Partial modification made by client A is not visible to client B before end- transaction of client A.

2. File Caching Schemes

File caching has been implemented in centralized time sharing system to improve I/O performance. The file is cached in the main memory so that repeated accesses to the same information can be handled without additional disk transfer. In distributed file system, it also helps in scalability and reliability. There are certain schemes for distributed file caching system.

i. Cache location

Cache location refers to the place where the cached data is stored. There are certain 3 possibility of cache location considering original file is at server disk.

a. Server's main memory – when no caching is used, generally the file is first transferred from the server's disk to the server's memory. Then file is transferred to client main memory.

b. Client's disk – the second option is to have cache in the client's disk. Keeping file cached at client's location has many advantage s – it eliminates network access cost, improve reliability, doesn't get effected by data lost in a crash, also has more memory as compared with the main memory cache. Disadvantages can be – it will not support diskless workstation, will cost disk access cost.

c. Client's main memory – the third option is to have the cache in the client's main memory. This option eliminate s both the network cost and disk access cost.

ii. Modification Propagation

In distributed file system, when the cache is located at the client 's side, it may be cached on multiple nodes. In such situation, when the caches all the nodes needed to be consistent if any node modified the cached files. Then same needed to be replicated at all the nodes. Hence there are certain modification propagation followed by distributed file system. It has 2 schemes:

a. Write- through schemes – in this scheme, when a cache entry is modified, the new value is immediately send to the server to update the master copy of the file. This removes the lost updated file in crash case. But it does have poor write performance.

b. Delayed-write schemes – wrote through scheme helps in read but does increase network traffic. In this scheme, when cache entry is modified it is kept at client's cache and when all the entries are modify then the entire cache is send together. It also certain way as write on ejection from cache, periodic write and write on close.

iii. Cache Validation schemes

Now it's clear that multiple copies if cached files are kept by on multiple nodes of clients. Modifications propagation specify when to

update the master copy at server side when cache is modified at clients side. This scheme tells when the other copies at multiple Nodes should be updated. It has 2 approaches:

a. Client- Initiated Approach – In this approach client contact the server whether it has same copy as the master one. It again follows 3 ways to check.
    - Checking before every access
    - Periodic checking at regular interval of time
    - Checking whenever clients open the file – it uses session semantics.
b. Server-Initiated Approach – Since client initiated approach results in traffic congestion, hence server initiated approach is used. In this approach client informs the file server that it is opening a file in which mode(reading, writing, or both). A server keeps monitoring the file usages . When a client closes the file, an intimation is send to server along with any modifications made to the file. On receiving the intimation it update the file at its end.