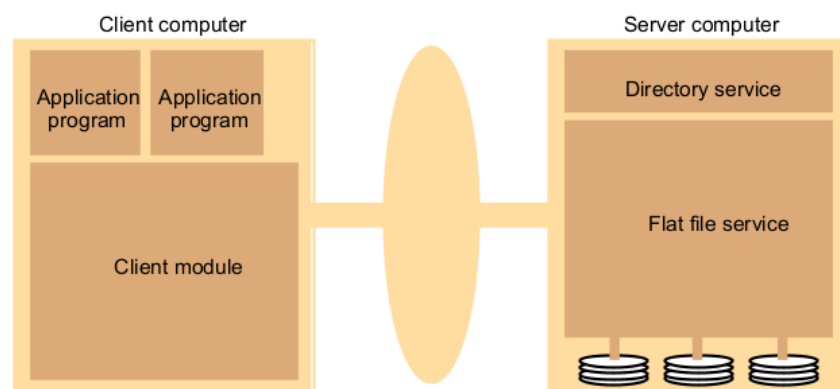# Distributed File System

1. Discuss techniques for achieving high-performance in distributed file systems.

7.

2. Discuss the model architecture of distributed file system and its components.

> 💡 Same as Q.3

9.

3. With a neat diagram explain the components of file service architecture in brief w. r .t. following: FFS, DS, CM.

File service architecture is an architecture that offers a clear separation of the main concerns in providing access to files is obtained by structuring the file service as three components:

- **Flat File Service:**

- The *Flat File Service* is concerned with the implementation of operations on the contents of the file.

- Unique File Identifiers (UFIDs) are used to refer to files in all requests for flat-file service operations.

- UFIDs are long sequences of bits chosen so that each file has a UFID that is unique among all of the files in a distributed system.

- When the *flat file service* receives a request to create a file, it generates a new UFID for it and returns the UFID to the requester.

- **Directory Service:**

  - The *Directory Service* provides a mapping between text names for the files and their UFIDs.

  - Clients may obtain the UFID of a file by quoting its text name to the directory service.

  - The directory service provides the functions needed to generate directories, to add new files to directories, and obtain UFIDs from directories.

- **Client Module:**

  - A *Client Module* runs in each client computer, integrating and extending the operations of the *flat file service* and the *directory service* under a single application programming interface that is available to user-level programs in client computers.

  - It holds information about the network locations of flat-file and directory server processes.

  - It helps to achieve better performance through the implementation of a cache of recently used file blocks at the client.

10. # 4. List out file system modules.

File system modules

Directory module:	relates file names to file IDs
File module:	relates file IDs to particular files
Access control module:	checks permission for operation requested
File access module:	reads or writes file data or attributes
Block module:	accesses and allocates disk blocks
Device module:	disk I/O and buffering

11. **5. Sketch the file attributes and record structure.**

File length
Creation timestamp
Read timestamp
Write timestamp
Attribute timestamp
Reference count
Owner
File type
Access control list

- File systems are responsible for the organization, storage, retrieval, naming, sharing, and protection of files.
- Files contain both data and attributes.

12. **6. List out the transparencies in the file system.**

- **Access Transparency:** Client programs should be unaware of the distribution of files. A single set of operations is provided for access to local and remote files. Programs written to operate on local files are able to access files without modification.

- **Location Transparency:** Client programs should see a uniform file namespace. Files or groups of files may be relocated without changing their pathnames, and user programs see the same namespace wherever they are executed.

- **Mobility Transparency:** Neither client programs nor system administration tables in client nodes need to be changed when files are moved. This allows file mobility. Files, or more commonly, sets or volumes of files may be moved, either by system administrators or automatically.

- **Performance Transparency:** Client programs should continue to perform satisfactorily while the load on the service varies within a specified range.

- **Scaling Transparency:** The service can be expanded by incremental growth to deal with a wide range of loads and network sizes.

7. List the directory service operation.

Lookup(Dir, Name) -> FileId Locates the text name in the directory and
-throws NotFound returns the relevant UFID. If *Name* is not in
the directory, throws an exception.

AddName(Dir, Name, File) If *Name* is not in the directory, adds(*Name,File*)
-throws NameDuplicate to the directory and updates the file's attribute
record.

exception. If *Name* is already in the directory: throws an

UnName(Dir, Name) If *Name* is in the directory, the entry containing
Name

is removed from the directory.
If *Name* is not in the directory: throws an exception.

GetNames(Dir, Pattern) -> NameSeq Returns all the text names in the directory that
match the

regular expression *Pattern*.

- The primary purpose of the directory service is to provide a service for translating text names to UFIDs.

- For each operation, a UFID for the file containing the directory is required.

- The **Lookup** operation performs a single *Name* → *UFID* translation. It is a building block for use in other services or in the client module to perform more complex translations, such as hierarchic name interpretation found in UNIX.

- There are two operations for altering directories:

  - **AddName:** adds an entry to a directory and increments the reference count field in the file's attribute record.

  - **UnName:** removes an entry from a directory and decrements the reference count.

  - If this causes the reference count to reach zero, the file is removed.

- **GetNames:** is provided to enable clients to examine the contents of directories and to implement pattern-matching operations on file names such as those found in the UNIX shell.

8. Explain the name service in detail.

- In a distributed system, names are used to refer to a wide variety of resources such as:

  - Computers, services, remote objects, and files, as well as users

- Naming is a fundamental issue in DS design as it facilitates communication and resource sharing

- In a Distributed System, a Naming Service is a specific service whose aim is to provide consistent and uniform naming of resources, thus allowing other programs or services to localize them and obtain the required metadata for interacting with them.

- The role of names and name services:

  - Resources are accessed using *identifier* or *reference*

    - An identifier can be stored in variables and retrieved from tables quickly

    - The identifier includes or can be transformed to an address for an object. Eg: NFS file handle, Corba remote object reference.

    - A name is a human-readable value that can be resolved to an identifier or address. Eg: An Internet domain name, file pathname, process number.

- For many purposes, names are preferable to identifiers

  - because the binding of the named resource to a physical location is deferred and can be changed.

  - because they are more meaningful to users

- Resource names are resolved by name services to give identifiers and other useful attributes.

9. Describe the characteristics of the file system.

File length
Creation timestamp
Read timestamp
Write timestamp
Attribute timestamp
Reference count
Owner
File type
Access control list

File attribute record structure

Directory module:	relates file names to file IDs
File module:	relates file IDs to particular files
Access control module:	checks permission for operation requested
File access module:	reads or writes file data or attributes
Block module:	accesses and allocates disk blocks
Device module:	disk I/O and buffering

Layered Module structure for the implementation of a non-distributed file system

- File systems are mainly responsible for the organization, storage, retrieval, naming, sharing, and protection of files.

- They provide a programming interface that characterizes the file abstraction about the details of storage allocation & layout.

- **Data and Attributes:**

  - **Data:** The data consists of a sequence of data items used for operations to read and write any portion of the sequence.

  - **Attributes:** It is a single record for holding information such as length of the file, timestamp, owner's identity, access control list.

- **Directory:** It provides a mapping from text names to internal file identifiers.

- **Metadata:** It contains information stored by the file system that is required for file management.

- Ex: File attributes, directories & all other persistent information

10. Discuss the distributed file system design requirements.

- **Transparency:** Some aspects of a Distributed system are hidden from the user.

  - **Access:** Client programs can be unaware of the distribution of files. The same set of operations is provided for access to remote as well as local files.

  - **Location:** The client program should see a uniform namespace.

  - **Mobility:** Client programs need not change their tables when files are moved to any other location.

  - **Performance:** The client program should continue to perform satisfactorily while the load on the service varies.

  - **Scaling:** The service can be expanded to deal with a wide range of load and network sizes.

- **Concurrent file updates:** Changes to the file by one client should not interfere with the operation of other clients simultaneously accessing or changing the same file.

- **File replication:** A file may be represented by several copies of its contents at different locations. It has the following benefits:

  - Load balancing to enhance the scalability of the service.

  - Enhances the fault tolerance.

- **Hardware and OS heterogeneity:** The service interfaces should be defined so that client and server software can be implemented for different operating systems and computers.

- **Fault tolerance:** To cope with transient communication failures, the design can be based on *at-most-once* invocation semantics.

- **Consistency:** Maintaining the consistency between multiple copies of files.

- **Security:** In distributed file systems, there is a need to authenticate client requests so that access control at the server is based on correct user identities and to protect the contents of the request and reply messages.

Security In Distributed Systems

1. Write the steps of the RSA Algorithm. Illustrate with an example given P=3 & Q=11.

15.

2. Analyze the following uses of Cryptography with suitable scenarios:
a) Secrecy and Integrity b) Authentication

16.

Secrecy and Integrity

- Cryptography is used to maintain the secrecy and integrity of information whenever it is exposed to potential attacks.

- It maintains the secrecy of the encrypted message as long as the decryption key is not compromised.

- It also maintains the integrity of the encrypted information, provided that some redundant information such as a checksum is included and checked.

Scenario:

Secret communication with a shared secret key:
Alice wishes to send some information secretly to Bob. Alice and Bob share a secret key KAB.

- Alice uses KAB and an agreed encryption function E(KAB, M) to encrypt and send any number of messages {Mi}KAB to Bob.

- Bob decrypts the encrypted messages using the corresponding decryption function D(KAB, M).

Authentication

- Cryptography is used in support of mechanisms for authenticating communication between pairs of principals.

- **Key** is known only to two parties.

Scenario:

Bob has a public/private key pair <KBpub, KBpriv>

- Alice obtains a certificate that was signed by a trusted authority stating Bob's public key Kbpub.

- Alice creates a new shared key KAB, encrypts it using Kbpub using a public-key algorithm, and sends the result to Bob.

- Bob uses the corresponding private key KBpriv to decrypt it.

17. # 3. Discuss asymmetric (public/private key pair-based) cryptography technique and how it can be used in supporting security in distributed systems.

When a public/private key pair is used, one-way functions are exploited in another way. The feasibility of a public-key scheme was proposed as a cryptographic method that eliminates the need for trust between communicating parties. The basis for all public-key schemes is the existence of a trap-door function. A trap-door function is a one-way function with a secret exit- it is easy to compute in one direction but infeasible to compute its reverse unless the secret is known.

The pair of keys needed for asymmetric algorithms are derived from a common root. The derivation of the pair of keys from the root is a one-way function. In the case of the RSA algorithm, the large numbers are multiplied together, this computation takes only a few seconds, even for very large primes used. The resulting product, N, is computationally infeasible to derive the original multiplicands.

One of the pair of keys is used for encryption. For RSA the encryption procedure obscures the plaintext by treating each block as a binary number and raising it to the power of key, modulo N. The resulting number is the corresponding ciphertext block. The size of N and at least one of the pair of keys is much larger than the safe key size for symmetric keys to ensure that N is not factorizable. For this reason, the potential attacks on RSA are small, its resistance to attacks depends on the infeasibility of factorizing N.

4. What is a distributed denial-of-service attack and how does it work?

18

Distributed denial-of-service attacks target websites and online services. The aim is to overwhelm them with more traffic than the server or network can accommodate. The goal is to render the website or service inoperable. Flooding a channel or other resource with messages in order to deny access to others.

Botnets

- The primary way a DDoS is accomplished is through a network of remotely controlled, hacked computers or bots. These are often referred to as "zombie computers."

- They form what is known as a "botnet" or network of bots. These are used to flood targeted websites, servers, and networks with more data than they can accommodate.

- The botnets may send more connection requests than a server can handle or send overwhelming amounts of data that exceed the bandwidth capabilities of the targeted victim.

- Botnets can range from thousands to millions of computers controlled by cybercriminals. Cybercriminals use botnets for a variety of purposes, including sending spam and forms of malware such as ransomware.

5. What is the goal of security? List the three broad classes of security threats?

19.

Classes of Threats

- **Leakage:** Refers to the acquisition of information by unauthorized recipients.

- **Tampering:** Refers to the unauthorized alteration of information.

- **Vandalism:** Refers to interference with the proper operation of a system without gain to the perpetrator.

6. What is cryptography? What is the use of it?

Cryptography

Cryptography is associated with the process of converting ordinary plain text into unintelligible text and vice-versa. It is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it.

Uses

- Secrecy and Integrity

- Authentication

> 💡 Same as Answer 2

7. Write a note on digital signature.

- Cryptography is used to implement a mechanism known as a digital signature.

- This is similar to the conventional signature, verifying to a third party that a message or a document is an unaltered copy of one produced by the signer.

- This can be achieved by encrypting the message called a digest – using a key that is known only to the signer.

- A digest is a fixed-length value computed by applying a secure digest function.

- The resulting encrypted digest acts as a signature that accompanies the message.

- The originator generates a signature with their private key, and the signature can be decrypted by any recipient using the corresponding public key.