Assignment (Week 11/30-12/4): Distributed Operating System Principles (COP5615)

Name: Vikas Chaubey, **UFID:** 3511 5826, **Email:** vikas.chaubey@ufl.edu

1) **Distributed File System (DFS):** A distributed file system is a system which is distributed across multiple machines in a cluster in different locations. DFS system allows the programs to store or access a file stored across the machines in cluster network. This system allows users from physically distributed systems to share their data and resources by using a common file system. Location transparency and redundancy are the main two components of distributed file systems. In case of any failure these components provide data availability by sharing data in different locations to be logically grouped under one folder, this folder locations are called "DFS root". The main features of DFS systems are transparency, high availability, high performance and simplicity and ease of use.

2) **Network File System (NFS):** NFS is a networking based distributed file system developed by sun microsystem which consists of server-side file system machine that host the data on its disks and many client-side file system machines which can request data from the servers using certain protocol messages. The main advantage of using NFS is that it provides centralized administration in the system, provides security and most importantly it allows easy sharing of data among different machines. When a client needs to access any file in the system then it passes a system call to the server like read, write or open etc. which in turn retrieves data from the server. These calls are not required to have special APIs, but they are made as normal system calls which is similar to requesting data from the physical disks, this is called "Transparency" in NFS.

3) **File Handles, Semantics of File sharing in NFS:** NFS makes use of file handles in order to identify a file which is currently being accessed, or the directory where the file is located. The file handle consists of a "volume identifier" which identifies the volume or partition where the file is placed in the server, in NFS a serve machine can contain various file systems hence a volume identifier helps in identifying which file system the requested file belongs to. Another important identifier is inode number which is used to locate and identify a file within the partition. While reusing the inode number generation number is used. File attributes is another terminology used in NFS. File attributes is a term used for metadata of a file such as creation time, last access time, size of the file etc.

4) **Some of the important protocols used in file sharing in NFSV2:** NFSPROC_GETATTR (protocol used to obtain file attributes given a file name) , NFSPROC_SETATTR (protocol used to set or update file attributes) , NFSPROC_LOOKUP (protocol the protocol used to return file handle) , NFSPROC_READ (protocol used to reads the data), NFSPROC_WRITE (protocol used to write data into the file) , NFSPROC_CREATE (protocol used to create a file) , NFSPROC_REMOVE(protocol used to remove the file from file system) , NFSPROC_MKDIR (creates a new directory)

5) **Client-side caching in NFS:** In order to improve the performance, the distributed file system in NFS caches the data which is requested by client from the server on the client side. So, whenever a similar request is made from the client side the cached data could be returned without following the full request cycle to get the data from the server. This is called client-side Caching. This reduces the process time involved in making subsequent calls to client. This also improves efficiency in the system because all the data is written on the server at once.

6) **Structured Peer-to-Peer Systems:** In a peer to peer system, the participating nodes are connected with each other with a network, and participant of the network work together in order to achieve a common goal, this goal could be file sharing, distributed computing, storage or communication etc. The main characteristics of the peer to peer system is that it does not have a centralized entity to control, administer and manage the whole system. Instead of having a single centralized controlling unit these tasks are divided among peers connected to the system. Peers can share resources such as CPU, storage, bandwidth etc. in order to perform system functions effectively. There are various advantages of implementing a distributed system using a peer to peer paradigm , these advantages mainly involve reduced reliance on central servers hence it removes the possibility of single point of failure in the system, it improves system scalability because any number of participant can be connected to the system and improve its capacity to perform a task, it is cost effective because the system utilizes already deployed resources and does not require to add any extra server for system to function , deployability of the system also improves with p2p networks because the system performs all the tasks at the endpoint nodes.

7) **REST framework:** Representational State Transfer (REST) is an architecture model which defines certain constraints to implement and access the web services. The Rest model makes the access of web service simple and flexible by reducing the processing. It is more robust compared to simple object access protocol(SOAP) based web services. Making the web services call using REST is simple. All communication made with REST models uses HTTP. In order to access the web services hosted on a server, a request is sent from client to server in the form of web URL as HTTP request, HTTP request could be any of the GET , POST , PUT or DELETE calls. Once the server receives the request it responds to the client in the form of resource which could be HTML, XML, Image or JSON.

8) **Microservices:** A microservice architecture is an architectural style which structures a software application as a collection of microservices. This architecture solves many problems associated with multi-tier software system. a microservice is a small scale, stateless, self-contained process which run and function separately and have a single responsibility.

   Advantages over multi-tier software architecture:

   1. A microservice architecture is not a layered architecture all microservices exist as separate entities hence this type of software architecture is highly maintainable and testable.
   2. Another advantage is that when a change in the software system has to be made then only the respective microservice has to be updated not the whole system.
   3. Scalability is another important advantage, sometimes software systems are required to scale a specific functionality in the system hence in this scenario multi-tier architecture has to scale up the whole system but in case of microservices architecture a specific microservice could be scaled up without changing the remaining system.
   4. A microservice architecture is not a layered architecture hence it does not have single point of failure as opposite to multi-layer architecture where if one layer fails the whole system is affected,

in case of microservice architecture if one service fails other services can still function without any issue. Hence this architecture is more reliable.

5. Loose coupling of microservices makes the software system more modular and flexible.
6. Microservices are containerized hence they are isolated from other services and components which increases security. that also means those services could started and stopped without effecting other components in the system.

9) **Docker:** It is an open runtime and packaging tool which can be used for building, distributing, and running applications in a portable, lightweight manner. Docker allows the software developers to pack the software application along all of its dependencies into a single unit called containers. Docker makes use of containers in order to run the application on host system. Software developers can create the images of the package and ship it for deployment. On the deployment servers these docker images could be used to create instances of containers which contain the application and all its dependencies. Docker containers are very light weight and can eliminate the use of virtual machine for the software deployment on the host machines which are very taxing on server resources

Refrences:

1) https://www.geeksforgeeks.org/network-file-system-nfs/
2) https://en.wikipedia.org/wiki/Clustered_file_system
3) https://docstore.mik.ua/orelly/networking_2ndEd/nfs/ch11
4) https://www.cse.iitk.ac.in/users/dheeraj/cs425/lec37.html
5) https://www2.cs.sfu.ca/~mhefeeda/Papers/p2pSurvey.pdf
6) https://www.geeksforgeeks.org/rest-api-introduction