CRITIQUE - 3 [Lecture - 4 Network File system]

Summary:
In this lecture, we discussed the Network file system, a high-level protocol that allows you to perform operations on files such as read, write, copy or delete. NFS is a file storage protocol that can be used in NAS (network attached storage). We discussed some of the important motivations for the NFS like the collaboration between the groups without having to copy the files, sharing storage resources, centralized administration etc. Later discussions included NFS design walk-through, an example of a read operation, some of the complications faced such as how to recover server crashes and the future scope of NFS.

In the first part of the lecture, I learnt about the motivations behind the birth of NFS such as collaboration among teams, mobility, sharing of the resources and centralized management. Some of the design assumptions that were made while implementing NFS are the Unix-centric file system, all clients had to be connected to the network to share resources, support for heterogeneous hardware etc. The important thing that I learnt here is that NFS is a protocol to access the files over the network as if they were local however, it doesn't implement the file system per se. To access files, NFS has to make requests over the network, which can be addressed using RPC (remote procedure calls). Security was implemented using either "AUTH_UNIX" or "Kerberos", reliability, fault-tolerance (how to recover from server crashes) etc are some of the problems to be taken care of while designing the NFS protocol.

We dived into each of the design assumptions made for NFS. We talked about the UNIX file semantics like soft and hard links to files, file permissions, deletion etc. Since NFS adheres to UNIX-like semantics - file locking is optional and it's advisory. NFS requests are self-contained and this makes it a "Stateless" protocol (NFS v3.0 and lower). Because it's stateless it can resume servicing clients after a crash. This gave me a good overview of the existing problems, how they were mitigated and incorporated in the NFS design.

Later we were introduced to the different versions of NFS like v2.0, v3.0, v4.0 and v4.1. The major difference between NFSv3.0 and v4.0 is that the latter was made "Stateful", which means NFSv4.0 and above are aware of which users are working with what files. NFSv4.1 implemented RAID-0 concurrency called pNFS (parallel NFS) with support for the block, file, and object storage protocols. Access permissions and other metadata were present on a single "Metadata server". I think this would pose some bottleneck issues when a large number of file requests come in and has a single point of failure. The example of reading 100 bytes from a file, clearly illustrated the different calls made between NFS client and NFS server and it also captured the essential differences between different versions of NFS with respect to API. This was really good and useful. Maybe a sequence diagram(a visual representation), instead of verbose on the slides could have been more helpful.

I learnt about some of the complications of the NFS. Initially, NFS was implemented using UDP and later migrated to TCP. NFS v4.1 supports sessions so that the request is delivered exactly once. Discussion on where NFS was successful and where it failed or could have addressed with a clean solution was interesting. NFS addressed the file renaming by abstracting the file names however it didn't address the file removal effectively. I understood how the NFS server recover crashes by "Grace period". During this grace period, all the request are denied and it's responsibility of the NFS clients to restore the state of the server. NFSv4 followed UNIX-style permission and also had support for non-UNIX environments using "Access control list" (ACL). However, ACL was not a big success and wasn't included in the standards by IETF. It also had a slower performance when compared to the older versions due to statefulness. Security in NFS is implemented using Kerberos but very hard to configure in reality. I felt he could have explained a little more about the ACL and why Kerberos is hard to configure.

The last part of the lecture was about the future scope of NFS. The original implementation of pNFS had single metadata server but could be implemented using parallel(multiple) metadata servers. File permissions could be implemented using "Discretionary access control"(DACL). Minor improvement in server side copies, end-to-end checksum. Federated NFS and transparent migration of data etc were explained with clarity.

Although the overall lecture was informative and concise, I felt it was a bit slow-moving and less interactive. All the concepts presented were new to me and I liked how the lecture was organized and example presented was good.