The paper gives a detailed introduction to the Ceph file system. For the traditional file systems, data are usually saved to the different sectors in the storage volumes, this is fine to treat small data, for the large data the larger scalability makes it inefficient to manage. To solve this problem, Ceph is based on object-based storage- treat large data as object which makes it easier to manage. By replacing these objects across the node clusters, Ceph also has a high fault-tolerance, and can achieve high scalability while maintaining performance and scalability.

To achieve the performance and scalability, Ceph has three identities: decoupling data and metadata, dynamic distributed metadata management, and reliable Autonomic Distributed Object Storage. The key point of this paper is the second identity: Dynamic distributed metadata. The large journals can let each MDS stream its updated metadata to the OSD cluster efficiently. Also, to do the dynamic subtree partition, there are counters to monitor the operations that used inode. MDS will periodically check if the inodes are imbalanced and if imbalanced occurs MDS will reassign the work over some subtrees.

Then the paper talked about the distributed object storage. Every object is using a hashing function to map to the placement group and each placement group is connected to the OSD by the CURSH (Controlled Replication Under Scalable Hashing). The CURSH can distribute data pseudo-randomly to make sure OSD utilizations can be accurately modeled by a binomial or normal distribution. For the Synchronization, when the client writes data to primary, the primary will forward the data to the replicas, only if all the replicas return ack to the primary, the primary can return ack back to the client. For the security, each OSD will flush the cache to the Replicas. Once the replica received the update, it will send a commit message to the primary. The primary will commit to the client only if all the replicas returned commit message. By default, clients also buffer writes until they commit to avoid data loss. I think the buffer may cause some performance issue, but the paper doesn’t mention it.

Ceph also has failure detection and recovery mechanism. The OSD can only report specific failures like disk errors or corrupted data. If the failures make OSD unreachable, an explicit ping will be sent. At the same time, each OSD will monitor the last time it heard from each server, if a node breaks, the node will be marked down. If the broken node doesn’t recover quickly, it will be marked out and OSDs will join each affected PG to bring the backup.

In the end, the paper discussed the performance. The OSD throughput and metadata scaling are the two parts that interested me. If the write size is low, the latency between different replicas are almost the same. However, with a higher write size, the network transmission time will dominate the latency. For the metadata operation scaling, the efﬁciency only drops less than 50% below perfect linear scaling for most workloads. I think this is already a good performance for the large file system.

Overall, this paper gives detailed information about the Ceph, the most impressive part of the system is the Distributing load with a cluster-wide mapping function (CRUSH), it is really effective. But since Ceph doesn't address all data storage needs, the performance of Ceph to deal with a massive amount of data can be weak, the paper doesn’t tell enough story about this limitation. Also, in the page 11 Figure 11, the figure’s curves are 4MDS, 16 MDS, and 128 MDS, but the paragraph describes figure 11 is the comparison between 4MDS, 16MDs, and 64MDS. I think this is a typo and I feel confused while reading this paragraph. Such a paper shouldn’t make small mistakes like this.