MP3 : Simple Distributed File System

Group 28 : Abhishek Verma (averma11) Dominic Le (ddle2)

This mp consists of a distributed file system built on the SWIM failure detector. Each machine has a distribution of files and the system has five basic commands (1) Add a file to the SDFS from local machine (2) Delete a file from the SDFS (3) Getting a file into the local system from SDFS (4) Getting all the locations of a file in the SDFS (5) Getting a list of local files on a system. The design is similar to a ring, as it is built over the failure detector. The system was designed using GoLang, and can be run in the background using tmux. The program directory contains a file folder which can contains all the files in that machine and any while which that machine requests. The system also utilizes scp to send files around from one system to another. The main assumption is that the system at any point contains at least 4 machines for the failure detector to work.

Design

The system consists of a master node which contains the file metadata, which is stored in the form of a dictionary with the key as the file name and a info struct as value (contains file location, size etc). All the other nodes have knowledge of the master node. Adding a file adds the file from the local system to the sdfs files folder. The machine adding the file sends a message to the master with its ip and file info. The master then executes a replication series which replicates the file in the next three members of the systems membership list. The master adds the file to its metadata with the location of where the file is located and each machine where the file is located adds it to its local list.

For deleting a file, a machine sends a message to the master which finds all the occurrences (replicas) of the file in the system and sends a message to all the machine to delete the file from their storage as well as remove it from their local list. It also removes it from its metadata.

For getting a file, a machine sends a message to the master which in turn sends the file from one of the file replicas to the requestor using scp. This file is stored in the files folder of that machine and is added to the local list of a machine.

In all the above cases the master is executing the scp commands from the source to the target and handling all the file transactions. All the machines have access to the other machines files folder via an ssh key and hence the scp doesn't require authentication. The system does not account for failing of machines while transferring files, it simply sends the file via another machine. A machine can also get all the locations of a file in the sdfs. This is done by sending a message to the master which sends back a message consisting of all the location from its metadata dictionary. This only includes the replicas of a file and not all copies of the file like in the scenario where a machine gets the file. Similarly the command 5 simply lists all the files in a machine by iterating through the local file list present on each machine.

The SDFS also handles the situations involving machine failures and master failures. In the situation of a machine failure, the failure message from the machine detecting the failure is sent to the master which in turn creates replicas a new replica of each file present in that system in some other machine (on order of the membership list). The master executes a series of scp requests to do this. It also sends a message to the new replica containing machines to add the file to their local list as well as it updates its metadata dictionary. This failure detection is multithreaded and also handles multiple simultaneous machine failures.
In the scenario where the master fails, the next highest ID in the membership list is the new master. This is similar to the bully algorithm and the process is really trivial. On the occurrence of a master failure, each machine gets the message and makes the next higher IP of the master as the new master. Since the membership list is sorted,this works even for multiple master failures.
The system does extensive logging on each file operation of the SDFS. The previous MP implementation of distributed grep was used to debug the mp, to find if a file was added successfully, or the replication was successful at 3 nodes. It also helped to find if the master re-election and replication works during multiple simultaneous failures.

1) Re-replication time and bandwidth upon failure for a 30 MB file
   Avg Time : 11.4 seconds replicated at ~ 4.1MB/s
2) Time between master failure an re-election
3) Time to read and write one file
   1) 20 MB :
   2) 500 MB
4) Time to store wikipedia corpus
   1) with 4 machines
   2) with 8 machines