EC504 Project: Proposal 2 - Dropbox-Like Storage

Team: The Box Droppers

(Yangruirui Zhou, Jie Lu, Yanqiu Chen)

Data Structures

- Chunk
  - (1KB)-sized data segment
  - Hash value to determine whether a newly-added file have this chunk already and serves as the name of chunk (e.g. $hashvalue.chunk)
- File
  - Name/hash value of every chunk that this file contains, in the form of a linked list
  - Maybe something extra: file size, timestamps, etc.
- A filename dictionary that corresponds actual filename strings to ID numbers
- A chunkname dictionary that corresponds chunk names to the number of files that contains it

- Function Outlines
- File Loading
  - Register a new ID in filename dictionary.
  - Generate a new file structure with ID.
  - Splitting file into chunks and padding the last chunk.
  - Hash each chunk and use the hash value as chunk name.
  - Find chunk name in chunkname dictionary, add new entry if needed. Add filenumber by 1.
  - Save the chunks, overwrite if needed.
- File Listing
  - Just list the dictionary of filenames, IDs should be hidden from display. Supports alphabetical order; with extra timestamp/file size it is possible to have these orders as well.

Function Outlines

- File Retrieval
  - Search by filename in the filename dictionary.
  - If filename is found in dictionary, find the corresponding file structure.
  - Go through the entire linked list, and for every node in the linked list copy the chunk.
  - Merge all chunks.
  - Return the file.
- File Deletion
  - Search by filename in the filename dictionary.
  - If filename is found in dictionary, find the corresponding file structure.
  - Go through the entire linked list, and for every node in the linked list decrease the item of chunkname dictionary by 1.
  - If now the item is 0, remove the chunk file and chunkname dictionary entry.
  - Remove the file structure, and entry in the filename dictionary.