Project Proposal: Big Box

Team members

Peng Wang (pwang1)

Di Zhu (dzhu1)

Motivation

People use all kinds of cloud storage services, many of them free, but they are not always perfect:

- Free users get limited space, e.g. 2GB for Dropbox, 5GB OneDrive, and 15GB Google Drive.
- Most providers don't always give you optimal performance. For example, try uploading 1000 very small files through Google Drive's client program?
- Although rare, users may suffer from temporary unavailability or permanent data loss.

Project description

We want to create Big Box, a web-based storage service to address these problems. Users may register multiple accounts on different cloud drives, and Big Box uses them like a RAID. For example, if we want to upload a file of 3GB, how about putting 1GB of it on each of my three accounts? Not only will you get larger capacity, but you could possibly also better utilize network bandwidth.

For the first stage of this project, we'd like to achieve RAID-0-like behavior. First, the server assists the user to authenticate and authorize our app, and store credentials in the database. Then, when the user uploads a large file, it is divided into chunks of maybe several MBs and uploaded to different locations in parallel. This involves both server, which keeps track of relevant metadata, and the script on the client side, because we want data flows directly to the net disks, instead of going through the server. Finally, the server provides information for the client to download and stitch together chunks to get files back.

Now we have credentials to access files from different users, they may be allowed to share files with each other, just like other major providers. We want the client to fetch chunks from net disks directly, so we'll have to carefully inspect APIs to avoid leaking one's tokens to another.

If time permits, we will also explore different techniques that accelerate file transfer. For example, Dropbox's client program does smart things such as compression, delta encoding, and batch transfer; while Google Drive does none. Our system structure is ready to adopt these techniques, because files are stored as chunks and compression, for example, only requires an additional step (before uploading a chunk / after downloading a chunk) on the client side.

Finally, it's possible to support a RAID-5-like deployment for better failure recovery.

Technologies

We plan to user Python + Django for backend, and Bootstrap + jQuery for frontend. Since the app interacts with cloud drives, we will use their SDKs. Some libraries (such as compression in JavaScript?) might be required along the way.