School of Computing

CA326 Year 3 Project Proposal Form

# Section A

Project Title: **Distributed Private Cloud Storage**

Student 1 Name: **Tomas Baltrunas** (ID Number **17350793**)

Student 2 Name: **Bartosz Śwituszak** (ID Number **17437072)**

Staff Member Consulted: **Ray Walshe**

# Project Description

Your own "Google Drive/Dropbox/OneDrive…". Store your data in a distributed and redundant manner on a network of your own machines/servers, known as nodes. Cut out the middleman and enjoy privacy and control of your data on a private cloud.

Firstly we will create a portable Go library, software that will allow nodes to participate in the cloud.

Data will be distributed between nodes in an efficient way to maximize redundancy and provide the quickest access by factoring in the nodes’ space availability, region, latency and bandwidth.

Having each chunk of data exist on at least two machines will be prioritized at first, to avoid loss of data if a node fails. Additionally, the system will attempt to have each chunk of data exist in every region and on high availability nodes. Each node in the cloud will be benchmarked with other nodes to measure it’s latency and available bandwidth. Nodes with highest bandwidth will be preferred for hosting large content of data that’s frequently utilized. Nodes with lowest latency will be preferred for files with the highest rate of access

It is not guaranteed for a node to have a copy of an entire file. Data will be stored in chunks, and it is possible for a node to only have few chunks of a file. Each node will contain a list of files and a list of other nodes that contain the data.

Because the network is distributed (and decentralized), there is no master server. There is a potential problem of two nodes attempting to modify the same data at the same time. To solve this, each change in the network requires at least 51% of the nodes to agree. The “change” that gets 51% first, will be the one implemented.

Encryption will be utilized to provide security of the data, ensuring that only users with private keys will be able to access the data. Compression may be utilized to maximize space efficiency.

After implementing the Go library for nodes, we will implement clients for PC, web and mobile platforms.

Users will be able to access their files, add new files for storage, and possibly have a cache of most recently modified files for live updates and quick editing.

This is an idea with a core concept and many possible extensions.

# Division of Work

Both will work on the core Go library. Afterwards we will split on implementing the clients for PC, mobile and web.

# Programming language(s)

* Go (for library and PC client).
* React Native (mobile).
* HTML/CSS/JavaScript (web).
* SQL or other database language.

# Programming tool(s)

* Go compiler and debugger.
* Client-to-client networking.
* Encryption library.
* Compression library.
* GUI.
* Web application.
* Mobile application.

# Learning Challenges

* Further develop our Go programming skills.
* Learn about networking - efficiently and securely transferring large data across networks.
* Learn about file systems - how to store data, how to implement redundancy, how to handle concurrent modification.
* Encryption - storing the data in a secure format.
* Learn mobile app development.
* Expand our knowledge on web development.

# Hardware / software platform

* PC (Windows, Linux, Mac).
* Web client can be used to access the cloud.
* Mobile app (Android, iOS).

# Special hardware / software requirements

* Machines for testing the app, i.e. private servers/PC’s at remote locations.