Distributed Cloud Storage – Technical Manual

0. Table of contents

1. Introduction

1.1 Overview

Distributed Cloud Storage – a set of programs that can turn your private servers into a cloud storage platform (think "Google Drive", "iCloud", or "Dropbox"). Our "node software" uses the Internet to connect your servers ("nodes") into a cloud designed for storage. Use one of our "client programs" to connect to your network and upload/download files, all as if the network was a single cloud entity.

Distributed (de-centralised), secure, intelligent.

- Leverages the nodes' underlying Operating Systems for persistent storage.
- Intelligent routing of files to the most optimal node in terms of storage load and network benchmarks.
- Reliability and privacy of storage at all times through redundancy and encrypted communications.
- Minimised single points of failure. Each node acts both as a client and as a server (a distributed system).

Portable (cross-platform), easily installable "node software" for technical/industry users requiring off-the-shelf private cloud storage solutions. Configure through a graphical or command-line interface.

"Client programs" including a mobile friendly website client and graphical desktop client for the end-users of storage. Modern file explorer UI/UX to interact with the cloud storage platform.

1.2 Glossary

1.2.1 Project-specific Terms

Node - a server or computer system capable of participating in a storage cloud (capabilities: network stack, persistent file system, etc.)

Node software - a program that joins the computer it is running on into a storage cloud, intended to be used by technical users.

Client program - a program or interface that connects the user to the storage cloud and allows them to store and download their files, intended for end-users that may not be as technical.

Node administrator - a user that interacts with the cloud storage in a technical way, ensuring set up of "node software" and some "client software" such as the website.

End user - a user that interacts with the cloud storage non-technically, to upload and download files.

1.2.2 General Terms

Cloud - network of computers connected via the Internet that expose some interface to the outside world.

Storage Cloud - a cloud designed to expose file storage.

Go, Golang - performant, concurrent, C like general-purpose programming language [https://golang.org].

RPC (Remote Procedure Call) - executing a function on a different computer.

Gob - Go standard library package for encoding/decoding variables into binary and vice versa, used for RPC [https://golang.org/pkg/encoding/gob/].

Binary executable - a single file that can be distributed and executed as a complete program (for example, .exe on Windows).

REST API - a style for web (HTTP) API's, important aspects include a client-server architecture and stateless requests (server treats each request as if the request had everything that was needed to serve it).

GCP (Google Cloud Platform) - cloud services provided by Google, including ability to rent a virtual machine with an external IP [https://cloud.google.com/free/].

Fyne - Go third party library for desktop-based portable GUI's [https://github.com/fyne-io/fyne].

React.js - JavaScript front-end web development library, declarative and stateful [https://reactjs.org/].

Bootstrap - CSS front-end library for mobile-friendly user interfaces [https://getbootstrap.com/].

PostgreSQL - a relational (SQL) database [https://www.postgresql.org/].

Ansible - an automation tool for deploying software onto machines via SSH using a declarative configuration [https://www.ansible.com/].

Make - a Unix tool for automatically building software via a set of rules [https://www.gnu.org/software/make/].

2. System Architecture

Go library. TCP.

Desktop GUI.

Desktop CLI.

Web app. Website. HTTPS

Secure communications.

3. High-Level Design

Class diagram.

Communications diagrams. TCP. HTTP (auth). web frontend <-> web backend <-> go library

4. Problems and Solutions

Data structure design (files, network).

Distribution algorithm (Calculating node benchmaks).

Frontend - bootstrap

Secure comms - HTTPS certs. Auth - JWT. Download. Auth middleware. DL. Login. Postgresql

Need Go. Go deps.

5. Installation Guide

Simple steps to install our node software and clients.

Where there are command-line examples, it is assumed that the environment is Unix (corresponding commands can be found for Windows).

See the User Manual (Node Administrator section) for more details on set up of the software.

All our software is cross-platform and compatible with most modern operating systems, including Windows, Linux, Mac OS X https://golang.org/cmd/go/#hdr-Compile_packages_and_dependencies. Some clients such as the web client work anywhere where there is a web browser.

We do not require special hardware. The software can manage both powerful and less powerful machines.

In the case of executable binaries we provide precompiled releases on our GitLab for different architectures. Another option is to compile from source.

5.1. Node Software

Obtain a binary distribution of our node software (named "cloud").

It is recommended for the node machine to have enhanced storage hardware (in storage space, RAID, etc.) and good or excellent network connectivity.

5.1.1. Obtain from a release.

See our GitLab releases.

5.1.2. Compile from source.

Clone the project's GitLab repository

```
git clone https://gitlab.computing.dcu.ie/baltrut2/2020-ca326-tbaltrunas-cloudstorage.git
```

Change directory into the node software

```
cd 2020-ca326-tbaltrunas-cloudstorage
cd code/cloud
```

Compile the software into a binary `go build cloud`

Optionally with the `Make` tool:

```
make
```

Find the node software executable binary under the name `cloud`.

5.2. Desktop GUI Client

Obtain a binary of the desktop GUI client.

The client requires the machine to have a graphical monitor.

5.2.1. Obtain from a release.

See our GitLab releases.

5.2.2. Compile from source.

Clone the project's GitLab repository

```
git clone https://gitlab.computing.dcu.ie/baltrut2/2020-ca326-tbaltrunas-cloudstorage.git
```

Change directory into the desktop client's directory

```
cd 2020-ca326-tbaltrunas-cloudstorage
cd code/cloud/des
```

Compile the software into a binary

```
go build
```

Optionally with the `Make` tool:

```
make
```

Find the binary.

5.3. Web Client

The web client runs as a website. See the user guide for in depth details of how to set up the web client from a node administrator's point of view.

6. Testing

Unit and integration tests.

System tests.

User testing.