CloudPi: Private Cloud Storage Using Raspberry Pi and Open Source Software

by

Kyle Muenker and Shane Vallance

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Kyle Muenker	3/5/17 Date
Shane Vallance	3/5/17 Date
Brian Verkamp, Faculty Advisor	4/17/17 Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 2017

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

ABSTRACT

Cloud storage subscription services don’t always provide the right amount of storage for the right price. Our goal was to create a system that is budget friendly, yet capable enough to suit a variety of storage needs. CloudPi combines the use of low cost Raspberry Pi’s and available open source storage software to provide a flexible and private cloud storage alternative. CloudPi allows users to safely store and access their files in a high availability cloud storage system. Configuring the Raspberry Pi’s as a load balancer, two web servers, and a distributed file system, we provided peace of mind with data and infrastructure redundancy. The modularity of CloudPi gives the user room for scaling the system up or down, depending entirely on their storage needs.

INTRODUCTION

Introduction

Many cloud storage and sharing providers are setup using a one-size-fits-all subscription based model. This pricing model doesn't allow the user to get the right amount of space for what he/she needs. Often, the subscription based storage provides either not enough space or way more than they truly need. If the user needs an amount of storage that falls between these two categories, they have no option other than to subscribe and pay for the plan with more storage space than they need. CloudPi aims to create a cost-effective solution for cloud storage, that users can host at home. Considering the ease of currently existing cloud solutions, CloudPi must be easy to setup as well as easy to use. Through the use of Raspberry Pi's and open source solutions, CloudPi can provide a cost effective, yet robust private cloud storage alternative to subscription cloud storage providers.

Project Description

The purpose of this project is to create a private cloud infrastructure with expandable storage at a low cost using Raspberry Pi devices as servers with any type of USB attached storage. Open source software and products will be used to maintain a tight budget on this project. The intent for CloudPi is to alleviate concerns users may have with subscription based cloud services when it comes to cost, privacy, and security of their data. Multiple Raspberry Pi's are added to provide data redundancy for the user's storage.

Problem[1]

The cost of subscription based cloud storage adds up over time. These costly subscriptions often don't offer specific amounts of storage and instead have tiered pricing models with large gaps, often jumping from 1TB to 10TB as the lowest cost solutions.[1] These gaps are reflected in the pricing, meaning that if users need more than 1TB of cloud storage he/she must pay for 10TB, even if he/she are not using all of it.

Solution

CloudPi aims to meet three criteria: low cost, private, and customizable storage. By using inexpensive Raspberry Pi units, CloudPi opens the user's budget to allow more flexible storage options. Also, by owning the hardware and configuring it, the user will have full control over the privacy and integrity of their data. By configuring multiple web servers with a load balancer, CloudPi provides redundant access to the user's storage.

Overview

Our goal is to introduce users to an alternative solution to cloud storage subscriptions. The remainder of this report outlines the audience for our project, how we plan to complete steps along the way, the obstacles we encounter, and the observations we make along the way. A schedule of deliverables is included along with our Gantt chart for the fall semester and the tentative spring semester timeline. Finally, our budget will be discussed with the actual cost for the equipment and what it cost us to create.

[1] 1. "Google Drive Storage Plans and Pricing," Google, 2016, accessed November 5, 2016. https://support.google.com/drive/answer/2375123?hl=en

DISCUSSION

User Profile

The User Profile describes the CloudPi system used by students, IT professionals, private and small business owners, and covers anyone else who may be looking for an alternative to the currently offered subscription services. Table 1 shows the complete User Profile.

Table 1. User Profile.

Application: CloudPi private cloud storage solution with backup
Potential Users: -Students -IT professionals -Private business owners -Those seeking alternatives to subscription services
Software and Interface Experience: This project is geared towards users with low to moderate technical experience. He/she will have had exposure to other cloud storage solutions that are available. He/she will be able to understand the concept of private versus public cloud storage, as well as understand how the Web portal is accessed remotely. The user is meant to set this device up on their own.
Experience with Similar Applications: The user will have had prior experience working with other Web based cloud storage services such as Google Drive, Apple iCloud, or Dropbox. The storage within the CloudPi setup will be familiar to other services to make it easier for the user to learn.
Task Experience: Setting up and configuring the OwnCloud software will take Linux OS and command knowledge. The user must also be able to set the security permissions on the device to their preferences. Once configured, the user will be able to share and store data on the device.
Frequency of Use: CloudPi will be used as often as the user wants to store and share data. This can be done from directly accessing the Raspberry Pi device and its storage, or remotely through the Web portal. This is designed to be a substitute for services such as Google Drive and Apple iCloud, so the it will be utilized differently depending on the user's habits.
Key Project Design Requirements that the Profile Suggests: The data should be accessible from the Web portal as long as the device is connected to a network. The second Raspberry Pi will have ownCloud installed and be configured as part of the GlusterFS file system to provide data redundancy to the user.

Hardware

The hardware that CloudPi uses is separate Raspberry Pi 3B devices for the ownCloud servers and HAProxy/MySQL server, and a flash drive as the storage for the Raspberry Pi web servers. The hardware is identical for each web server used. The storage size can be determined by the user to fit their needs.

Software

CloudPi uses open source software to provide the user with secure, private cloud storage. The software frontend that end users interact with is ownCloud, which has many options to be customized to the user's preference. The Raspberry Pi units run on the Raspian Linux operating system. The units that have ownCloud installed will use Apache as the web server and will be using the MySQL database installed on the load balancer for the database. HAProxy is the load balancing software configured to direct web traffic in a round robin fashion. It is installed on a separate server so that in the event a web server is unavailable; the load balancer will fail traffic over to the available web server.

Data Redundancy

The Raspberry Pi web servers uses MySQL as the database and GlusterFS for a distributed file system. MySQL is set on each web server with a master-master replication relationship so the data is copied and shared to both servers. GlusterFS allows any changes made to the storage on one web server to be mirrored on the storage of the other web server. GlusterFS is installed and configured as a node on both or all the Raspberry Pi devices with ownCloud installed. GlusterFS is configured to replicate data changes to and from all other nodes set up on the other web servers allowing each device to have a copy of the data available in the event of failover.

Use Case Diagrams

The user will be able to access ownCloud on the CloudPi device through a web browser. There is a portal to log into that allows the user to transfer files back and forth to the CloudPi storage. This is shown in Figure 1. To manage the CloudPi system, the user will be able to either connect to the Raspberry Pi's through SSH or by plugging the Raspberry Pi into a monitor and performing the task directly on it. This allows the user to perform necessary updates, storage changes, etc. This is shown in Figure 2.

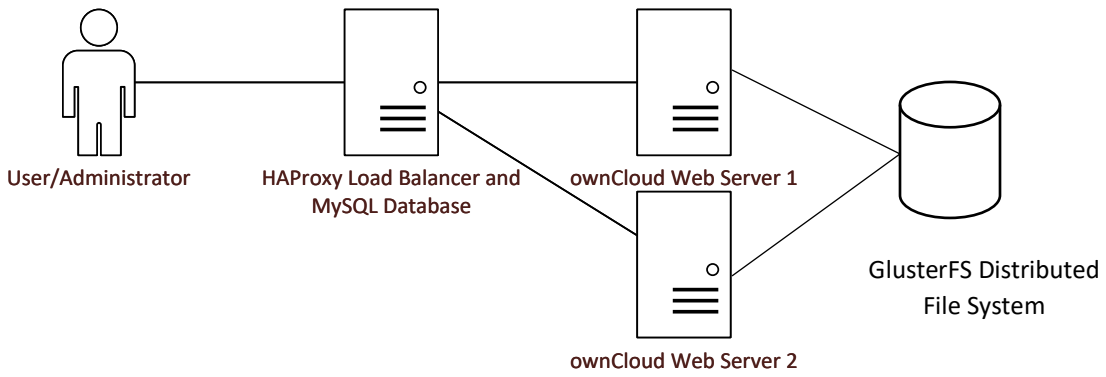

Figure 1. User and administrator access to ownCloud web portal.

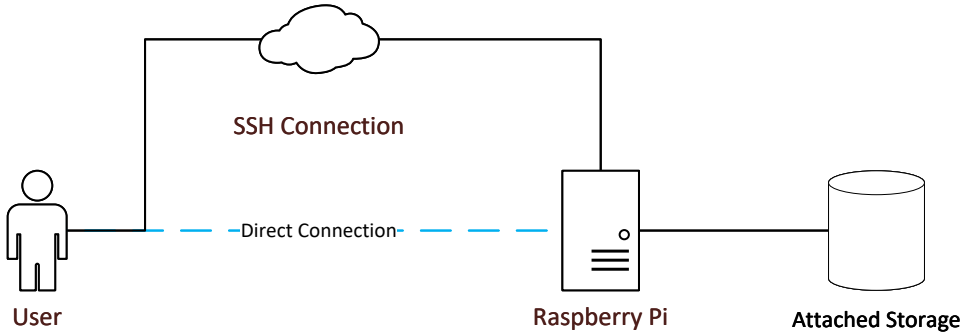

Figure 2. Management of Raspberry Pi device

Objectives/Deliverables

The project is planned to take place over the course of the fall and spring. Most of the fall consists of research and early implementation. Two Raspberry Pi's will be configured as the ownCloud web servers and the third will be the load balancer. The schedule for the deliverables and tasks is outlined in Table 2, along with the first and second semester Gantt charts in Figure 3 and Figure 4, respectively. The task number in the first column in Table 2 correspond to the tasks in Figures 3 and 4.

Table 2. Deliverable Schedule.

CloudPi - First Semester Schedule			
Task	**Task Description**	**Start Date**	**End Date**
1	Brainstorm project ideas	8/22/2016	9/12/2016
2	Research hardware requirements / costs	9/12/2016	9/19/2016
3	Research software solutions	9/12/2016	9/19/2016
4	Purchase hardware	9/19/2016	9/26/2016
5	Research Apache vs. NGINX web server solutions	9/26/2016	10/3/2016
6	Begin loading Raspian OS onto first SD card	10/3/2016	10/10/2016
7	Configure web server and other prerequisites to ownCloud	10/3/2016	10/17/2016
8	Download and configure ownCloud software	10/17/2016	10/31/2016
9	Research potential security solutions	10/24/2016	11/7/2016
10	Configure Raspberry Pi to use USB attached HDD	10/24/2016	10/31/2016
11	Research on setting second Raspberry Pi as backup client	11/1/2016	11/8/2016
12	Begin testing usability of ownCloud web portal	11/1/2016	11/8/2016
13	Configure second Raspberry Pi as backup client	11/8/2016	11/22/2016
14	Work on Rsync automation	11/8/2016	11/22/2016
15	Test Rsync from server device to backup	11/22/2016	11/30/2016
CloudPi - Second Semester Schedule			
16	Configure Haproxy Server	1/9/2017	1/16/2017
17	Configure MySQLDatabase	1/9/2017	1/23/2017
18	Configure both ownCloud Servers	1/9/2017	1/23/2017

19	Configure GlusterFS File Replication	1/9/2017	1/23/2017
20	Pair ownCloud Servers with LB and DB	1/16/2017	1/23/2017
21	Prepare Deliverables Presentation	1/23/2017	2/6/2017
22	Test Multiuser Setup and File Sync	1/30/2017	2/6/2017
23	Test LB Failover	2/6/2017	3/6/2017
24	Test LB Cookie for Persistent Sessions	2/6/2017	3/6/2017
25	Test File Replication via GlusterFS	2/6/2017	3/6/2017
26	Configure New Login Screen Image	2/6/2017	3/6/2017
27	Research Scale-up for Design	2/13/2017	3/6/2017
28	Research ownCloud App Usability	2/20/2017	3/13/2017
29	Test Remote Access to CloudPi	2/27/2017	3/20/2017
30	Prepare CloudPi for Expo	3/13/2017	4/10/2017

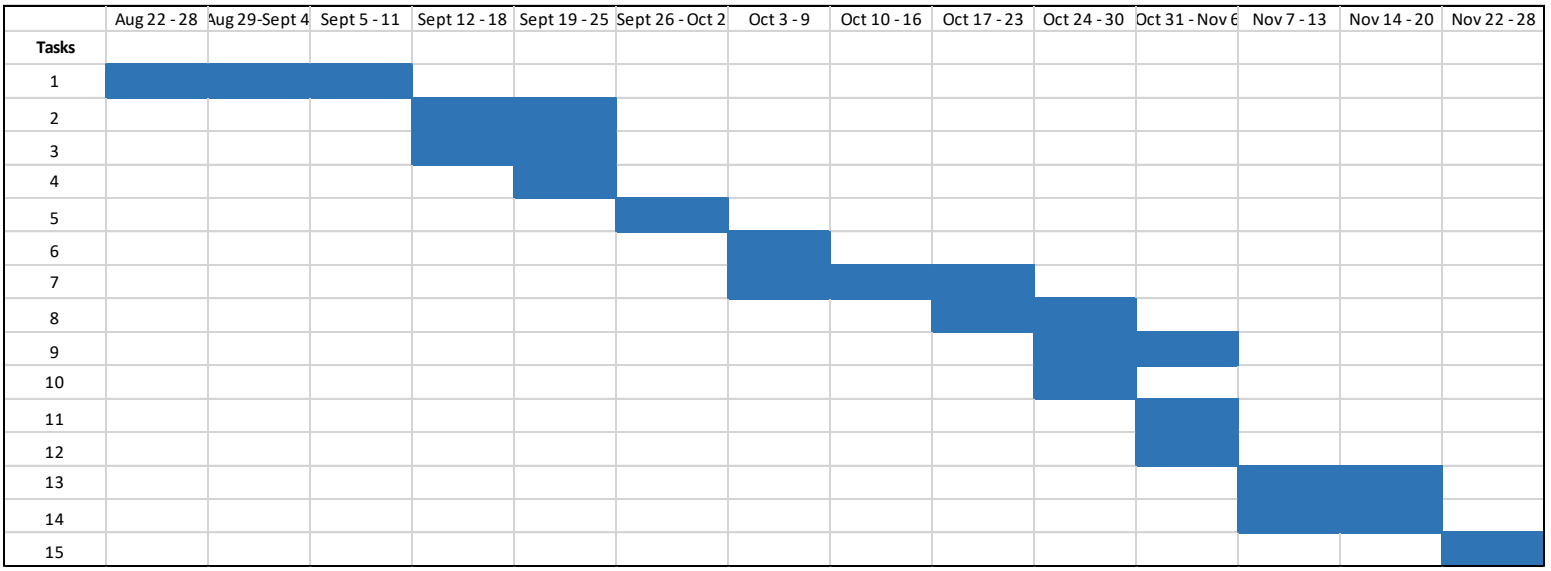

Figure 3. First Semester Gantt Chart

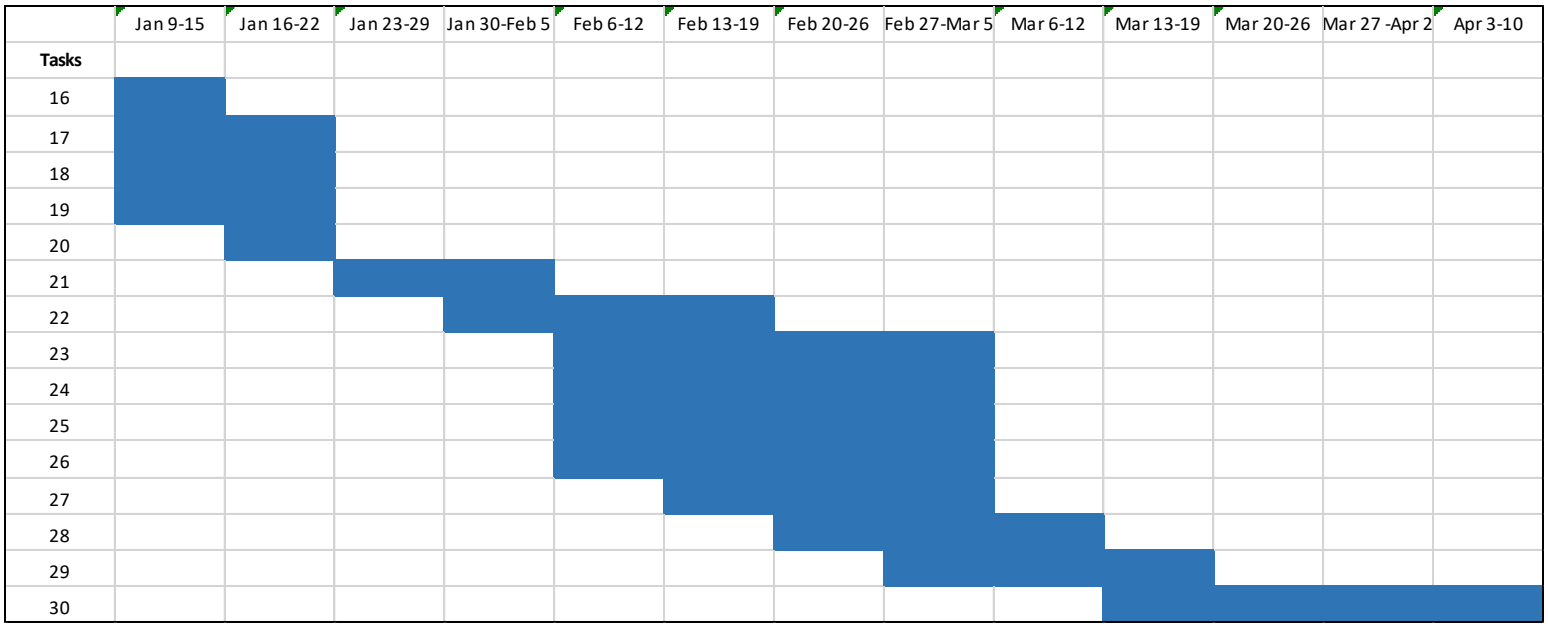

Figure 4. Second Semester Gantt Chart.

Budget

Our budget reflects both the actual cost of the CloudPi solution and how much it cost us to create, as shown in Table 3. Some of the hardware was already owned, so our project cost is much lower than the actual cost. Additionally, the storage size will vary based on the user's preference, so the actual cost will change depending on the user's storage. For the purpose of the project, the actual cost will represent what it would cost to purchase the components we already own.

Table 3. Project Budget.

Quantity	Item	Unit Cost	Project Cost	Actual Cost
HARDWARE				
3	Raspberry Pi 3 B	$34.99	$104.97	$104.97
2	16Gb Flash Drive	7.99	-	14.98
3	SD Card	9.99	9.99	28.97
SOFTWARE				
2	OwnCloud	-	-	-
TOTALS				
Actual Total Cost				$148.92
Project Total Cost				$114.96

End User Configuration

For end users to be able to configure CloudPi for personal use over an internet connection, he/she will need to have admin access to their personal network router and modem. Information on port-forwarding or information on how to find it will need to be provided to end users to ensure he/she is able to configure network access correctly.

Testing

Components of the CloudPi design will be tested in three categories: Load Balancing, Distributed File System, and the User Capabilities. Each category contains tests that prove the functionality of the core systems that make up the CloudPi design; redundancy of data, flexibility of storage, and privacy. Table 4 takes these tests and outlines the actions that were taken, the results of these actions, and whether they were deemed a pass or fail, according to the intended design of the system.

Load Balancing Requirements

A. Load balancer must direct the user to one of the web servers

B. Load balancer must leave a browser cookie to direct the user to the same web server in the future

C. If one web server is inaccessible, the load balancer must direct traffic to the other accessible web server

GlusterFS Distributed File System Requirements

D. Admin user must be able to log into both web servers

E. Admin user must be able to create new user accounts and set permissions on them

F. Admin user must be able to delete user accounts

G. Non-admin user must be able to log in to both web servers

H. Admin and non-admin users must be able to create and delete their own files

User Capabilities

I. File system must replicate created files to the storage device on the other web server

J. Deleted files must be removed from both storage devices

Test Results

Table 4 describes the above testing plan and shows the result of each task.

Table 4. Test Results.

Test	*Action*	*Result*	*Pass/Fail*
Load Balancer Testing			
A	Browse to Load Balancer	Directed to ownCloud Server 1	Pass
B	Close web browser, reopen and browse to LB address	Directed to ownCloud Server 1	Pass
C	Shutdown ownCloud Server 1, close browser, reopen and browse to LB address	Directed to ownCloud Server 2	Pass
GlusterFS Distributed File System Testing			
D	Admin user log into ownCloud server 1, log out, and log into ownCloud server 2	Successful authentication on both servers	Pass
E	Admin user creating user 'Kyle'	User 'Kyle' created and stored in MySQL DB	Pass
F	Admin user delete user 'Kyle'	User 'Kyle' deleted and removed MySQL DB	Pass
G	Non-admin user 'Shane' log into ownCloud server 1, log out, and log into ownCloud server 2	Successful authentication on both servers	Pass
H	Admin user creates and deletes text file; user 'Shane' creates and deletes text file	Admin user was able to create a file and then delete it; 'Shane' was able to create a file and delete it	Pass
User Capabilities Testing			
I	User 'Shane' creates a file. File created by user 'Shane' must be replicated onto the storage of both ownCloud web servers	File was created on ownCloud server 1 and was subsequently mirrored to the storage on ownCloud server 2	Pass

J	User 'Shane' deletes the file recently created by them	File owned by user 'Shane' must be removed from the storage devices on both ownCloud 1 and ownCloud 2	Pass

Problems Encountered

Over the course of the school year, we encountered several obstacles; some we could resolve, and others were not. The obstacles that we could not resolve paved the way for us to make changes to our design that ultimately shaped our final deliverable. Our initial design was to use two Raspberry Pi's with ownCloud on each and to have an rsync script copy the data back and forth. We found that this was not efficient, nor was it what we really wanted to do with the design. At the end of the first semester, we went back to the drawing board and made changes to our approach. The first two weeks of the second semester saw the addition of a third Raspberry Pi, which became the load balancer for the traffic to the two other ownCloud Raspberry Pi's. The two ownCloud Raspberry Pi's were also to host the MySQL database with a master-master replication relationship between the them. This new design was much closer to what we envisioned doing; however, there were still a few challenges that came with it. The first was getting enough power through the Raspberry Pi to the HDD without the use of a powered USB hub. To keep costs down, we replaced the HDD storage with flash drives that will still maintain the integrity of the concept, while resolving the power issue. The second conflict was automounting the data points with GlusterFS. On device startup, GlusterFS would immediately begin to look for the partner data point on the other web server. This action would take place before networking on the device started, so GlusterFS would fail to find the other data point. We would have to manually mount it and locate the other data point once the networking started on the Pi. We worked on setting up a cron job to attempt to run GlusterFS again after networking was active, but this still did not resolve the issue. Our third obstacle was with setting up MySQL with the master-master relationship on the web servers. There were problems with the databases not being in sync, and to have a functioning design, we made the decision to move the

MySQL database to the load balancer Pi. We understand this it is not ideal to have the database residing on the frontend device, and we would like to have been able to resolve the replication problem in time for the expo.

Conclusion

Future Considerations

We intend to use this project for personal use after the final product is completed. By creating snapshots and taking images of the devices along the way, we are able to quickly reconfigure and deploy more nodes if we would like to expand the system. Figure 5 shows the scalability of the CloudPi system by adding additional devices for a standalone database as well as an HAProxy failover. Web servers can also be added to increase the availability of the system.

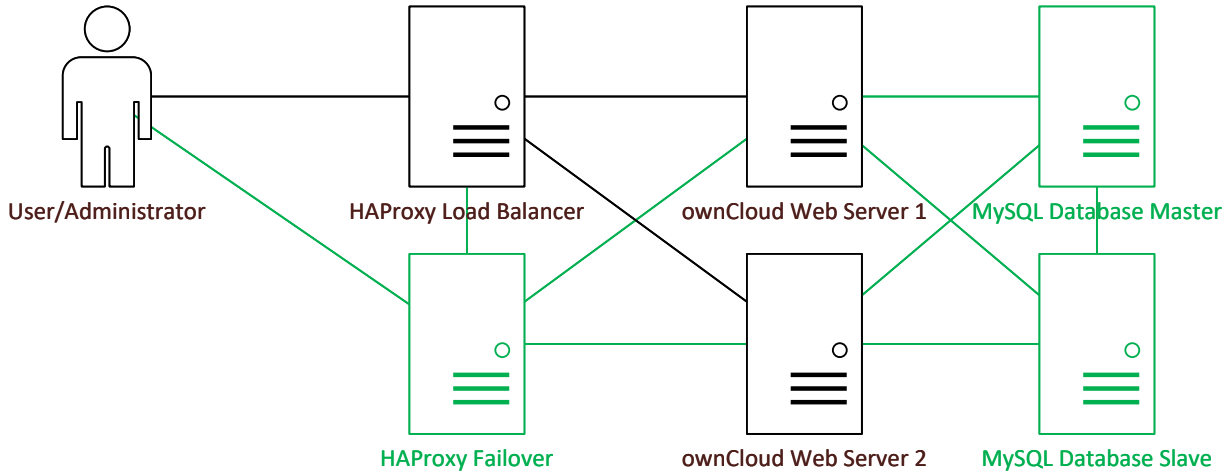

Figure 5. CloudPi Scalability

Conclusion

We noticed a gap in the available solutions for cloud storage and decided that we would look for a way to deliver a highly available cloud storage solution. One which would allow users to pick the amount of

storage their solution should provide instead of falling within the tiered storage levels of the larger cloud storage providers. We had an idea of what we intended to do, and through research and implementation of different available options and tools we were able to design and redesign CloudPi to fit our vision for a private cloud storage alternative. CloudPi fills the void of scalable and private cloud storage in a market dominated by non-flexible subscription services.

BIBLIOGRAPHY

"Google Drive Storage Plans and Pricing," Google, 2016, accessed November 5, 2016. https://support.google.com/drive/answer/2375123?hl=en.