**NUBISAVE-CLOUD STORAGE CONTROLLER**

A PROJECT REPORT

Submitted by

**SAKETH PATIBANDLA (10BCE0495)**

*in partial fulfillment for the award of the degree*

*of*

**B.TECH**

Degree in

**Computer Science and Engineering**

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**





**School of Computing Science and Engineering**

**DECLARATION**

I hereby declare that the project entitled **“Nubisave-Cloud Storage Controller”**

submitted to the School ofComputing Science and Engineering, VIT University, Vellore-14 in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in** **Computer Science and Engineering** is a record of bonafide work carried out by usunder the supervision of **Prof. Gopinath M.P Assistant professor(Senior)** and **Dr.Ing.Josef Spillner.** I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or university.

Signature

**SAKETH PATIBANDLA (10BCE0495)**



**School of Computing Science and Engineering**

**CERTIFICATE**

The project report entitled “**Nubisave-Cloud Storage Controller**” is

prepared and submitted by **SAKETH PATIBANDLA(10BCE0495)** as a part of

an internship team at **Technische Universitat Dresden.** It has been found satisfactory

in terms of scope, quality andpresentation as partial fulfillment of the requirements

for the award of the degree of **Bachelor of Technology in Computer Science and**

**Engineering** in VIT University,India.

**Extenal Guide Internal Guide**

**Internal Examiner** **External Examiner**

**ACKNOWLEDGEMENT**

It is my pleasure to express with deep sense of gratitude to **Prof. Gopinath M P**, SCSE, VIT, for his constant guidance and continual encouragement.

I wish to express my sincere gratitude to **Prof. Hari Seeta,** Programme Chair of B.Tech CSE, School of Information Technology and Engineering for providing an opportunity to do my project work.

Sincere gratitude to **Dr. Venkateswaran S**, Dean, School of Computing Science and Engineering [SCSE], for giving the opportunity to undertake the project.

My deep thanks to **Dr Vijaykumar V**,Director,International Relations for letting me pursue my semester abroad in Technische Universität DresdenGermany) and let me improve my skill in this field of engineering.

Regards to **Dr. Ing. Josef Spillner**,Facultat fur Informatik,for guiding me in all my queries and letting me be a part of developing this wonderful application for universities and organization.

At last but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place : Vellore Platz : Dresden

Date : Datum :

VIT University Technische Universität Dresden



|  |  |
| --- | --- |
| **Table of Contents** |  |
| [Abstrac](#page8)t | 8 |
| [1 Introductio](#page9)n | 9 |
| [1.1 Genera](#page9)l | 9 |
| [1.2 Motivatio](#page9)n | 9 |
| [1.3 Ai](#page9)m | 9 |
| [1.4 Objective](#page9)s | 9 |
| [1.5 System Requirement](#page9)s | 9 |
| [1.5.1 Hardware Requirement](#page9)s | 9 |
| [1.5.2 Software Requirement](#page9)s | 9 |
| [1.6Report Organisatio](#page10)n | [1](#page10)0 |
| [2 Overvie](#page10)w | [1](#page10)0 |
| [2.1 Introductio](#page10)n | [1](#page10)0 |
| 2.1.1 FUSE  2.1.2 Cache Parallelism | [1](#page10)0 |
| [2.2](#page12) Cloud Storage Controller | [1](#page12)0 |
| 3 Phase I : Creation and demonstration | [1](#page13)1 |
| [3.1](#page13) Purpose | [1](#page13)1 |
| [3.2 Introduction](#page17) | [1](#page13)1 |
| 3.3 Working | [1](#page14)2 |
| 3.4 Configuration | [1](#page14)4 |
| 3.4.1 Graphical | [1](#page15)4 |
| 3.4.2 Editor | [1](#page16)6 |
| 3.5 Activity:Model Working | [1](#page17)8 |
| 4 System Check | [1](#page17)9 |
| 4.1 Problem with existing system | [1](#page18)9 |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| [4.2](#page18)Proposed system | [1](#page18)9 |
| [5 Phase II :Architecture and Performance testing](#page21) | 19 |
| [5.1](#page21) Working | [2](#page21)2 |
| [5.2](#page22) Performance testing | [2](#page22)4 |
| [5.2.1](#page26)Rewrite test on hard disk | [2](#page26)4 |
| [5.2.2 Rewrite Test on cache](#page26) | [2](#page26)5 |
| 5.2.3 Rewrite test on the network | [2](#page27)6 |
| [6 Actual Protot](#page28)ype and Functioning | [2](#page28)6 |
| [6.1 Implementatio](#page28)n | [2](#page28)6 |
| [6.1.1](#page29) Cloudfusion | [2](#page29)7 |
| 6.1.1.1 Mounting Cloudfusion  6.1.1.2 Unmounting  6.1.2 Rebuild Nubisave  6.1.3 An example of a .ini file of dropbox  6.1.4 Understanding Gitorious  6.2 Demonstration  7 Conclusion and Result  [8 Reference](#page29)s | 33 |

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fig no.** |  | **Title** | **Page no.** |
|  |  |  |  |
| 3.5 | Activity diagram |  | 18 |
|  |  |  |  |
| 5.2.1 | Graph: Rewrite test on hard disk |  | 24 |
|  |  |  |  |
| 5.2.2 | Graph: Rewrite test on cache |  | 25 |
|  |  |  |  |
| 5.2.3 | Graph: Rewrite test on Network |  | 26 |
|  |  |  |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table no.** | **Title** | **Page no** |
|  |  |  |
| 5.1.1 | Services | 22 |
|  |  |  |
| 5.1.2 | Authentication | 23 |
|  |  |  |
| 6.2.1 | Storage Selection | 30 |
|  |  |  |
|  |  |  |
| 6.2.2 | Configuration Element | 31 |
|  |  |  |
|  |  |  |
|  |  |  |
| 6.2.3 | Non-Functional Properties | 31 |
|  |  |  |

Abstract

In recent days, cloud computing has become a major requirement and an asset for the world of computing in general and for specific organizations as a whole. Although storage of data has been highly being performed in major data centers, with it comes a requirement to store it in an interior and an organized storage called the cloud storage.

This cloud storage has to be highly feasible in accordance with the acceptance and the adoption rates of the server and the inbuilt system. All the non-functional properties such as redundancy, security (in the form of encryption). So, Nubisave is presented on this front to make it a user friendly application to desktop users and maintain their data through this university cloud which is maintained and distributed throughout for quick computing and worldwide sharing.

We use this way of computing not only to perform direct saving of data but to save it in a secured manner and to keep it protected from any threat and the major RAIC (Redundant Array of Inexpensive Clouds) which are available to develop this application. It selects the data, splits it into fragments adding redundancy to it and then preforming a secure partitioning of the data stored into it.

In this project, as a part of the FlexCloud Research Project we demonstrate the usage and maintenance of public data as a group through C++(code) whose front end is shown in the application to connect between devices and deploy storage access through the Nubisave feature and maintenance which is a part of real time hardware department and testing its vitality on data end users who would like to store their data and upload or connect their cloud online data drives to their desktop system through data sharing over the internet.

First, we connect the likes of Dropbox with Nubisave and check its compatibility with the system in use and write a result on the basis of its performance. Then, we demonstrate a connection with the online cloud and all the parameters that can be used over it to check its vitality over the data and finally we conclude by testing the data over the encrypted modules that are kept after connection.

1. Introduction
   1. General

Cloud storage (enterprise/customer) level is something that involves logical pooling of really digital data. Like all other physical data and servers owned and maintained by companies, this is one such development to ensure safety and security which are the high necessities of cloud storage achieved through Nubisave (frontend) to visualize and parameterize the data effective to it.

1.2 Motivation

There is lot of research going on about cloud data storage throughout the world and unlike a production level software Nubisave offers new talent to be heard in areas of research in cloud storage integration and data distribution. All such sides of cloud computing offer a wide role to save and store data in one such facility and transfer it towards regular use on requirement.

1.3 Aim

To satisfy the major aims of cloud computing and distributed systems, to save money (cost benefits),perform efficient and user friendly data storage and build a system that performs encryption on a whole.

1. Objectives
   1. To demonstrate the working and development of Nubisave (the cloud storage system) and connections to cloud servers
   2. To compare its architecture to a real distributed system architecture and perform performance analysis and take the results through its data unit.
   3. To let out an actual prototype to show its functioning, process and the storage of the file system that makes data storage and validation possible in this unit.
2. System Requirements
   * 1. Hardware Requirements
        1. RAM: At least 2 GB recommended(4GB – 8GB expected)
        2. Processor: Any Intel or AMD x86 processor supporting SSE2 instruction set
        3. Memory Requirements: 1 GB for MATLAB only, 3–4 GB for a typical installation.

d. Data Servers : 4-6 chip servers depending on the data stored and

memory used.

* + 1. Software Requirements
       1. Operating System: Debain or Ubuntu
       2. File Systems: NTFS
       3. Software: Git code repository with C++(package) and installed Nubisave

1.6 Report Organisation

The next part contains an introduction to the project. Following that, will be design, implementation, testing of the project. Later, conclusions, results and future enhancements will be discussed.

2 Overview

2.1 Introduction

Let us learn about the software being developed and its features to make it usable by the standalone users or enterprises.

It is one of those file systems built on FUSE which integrates with many cloud storage related modules. It offers tree like structures and interfaces for nested splitters, deduplication, compression and especially encryption. It performs technical features like optimal redundancy, cache parallelism, streaming, chunking and sessions data receival.

2.1.1 FUSE

In  [cloud computing,](http://en.wikipedia.org/wiki/Image_processing)  [FUSE (FileSystem in UserSpace) is an operating system that allows non privileged users create their own file systems without the need to alter the already existing Kernel code. It plays its role as a bridge to the real or the direct Kernel interfaces by running the file code in the user space. It is majorly used for creating VFS (virtual file systems), they don’t actually save data rather translate the existing file system into a device.](http://en.wikipedia.org/wiki/Computer_vision)

2.1.2 Cache Parallelism

Memory level Parallelism or cache parallelism is the system’s ability to have

Memory operations left over, more cache misses can occur at the same point of time.

Much cache parallelism is reflected in multiprocessor or multithreaded system environment, but actually happens over a single process.

2.2 Cloud Storage Controller

Now, pushing ourselves deep into the subject cloud storage controllers are ones used to store data and ease anxiety or any kind of pressure on actual data clouds.It means that we spend less money and store more data. The main advantage of the cloud storage controller is its file system. Its payload data and the metadata are separated with a logical and a physical separation between them which makes the extraction of metadata very easy and transfer it in the most simplest ways possible.

Navigation is also made simple with the existence of Cloud storage controller. In this the user can easily retrieve his data irrespective of what the person maintaining it follows and Nubisave is also an example of the cloud storage controller which makes work for the user as easy as it can get. Let us go into the first phase of the project where we learn and demonstrate the creation and working of this application.

3. Phase I : Creation and demonstration

* 1. Purpose

This part consists of installation of Nubisave into your Linux or Debian system and mounting the cloud storage controller onto the online cloud platforms like Dropbox or Sugarsync.

* 1. Introduction

We use OS Ubuntu 12.04 to run this cloud storage controller called Nubisave which allows us to use dispersion and other similar techniques to secure the data past encryption and safely store the data onto the cloud.

Let us first see a few steps to program nubisave on the file system.

a) Open the editor and install x86\_64 specific packages that run in the latest version of Ubuntu and Debian operating systems.

b) They consist of the following steps-

🡪 wget -O- http://nubisave.org/downloads/nubisave.key|sudo apt-key add

(connects to the main system and downloads the key and adds it onto the sub system)

🡪 echo "deb http://nubisave.org/packages/ ./" | sudo tee -a /etc/apt/sources.list

(compresses all the packages and connects to the debian server)

🡪 sudo apt-get update

(Installs the latest updates and variations of nubisave)

🡪 sudo apt-get install nubisave # complete set

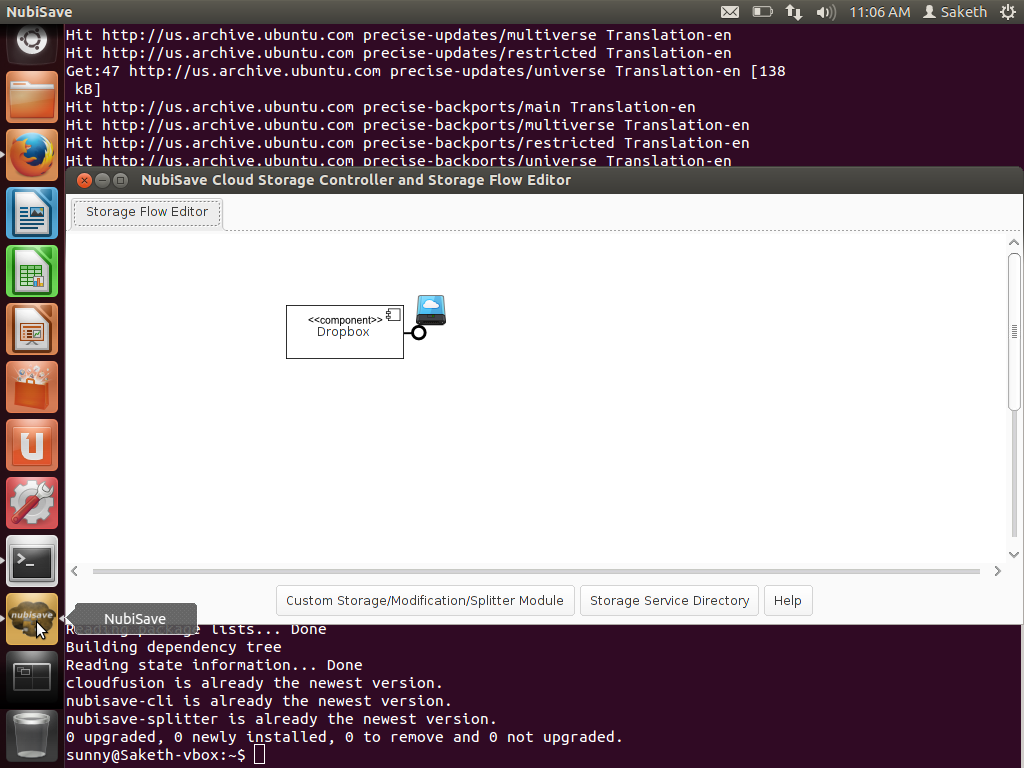
(installs the complete set of the Nubisave package directory and automatically upgrades any data to the next level)

🡪 sudo apt-get install nubisave-cli nubisave-splitter cloudfusion # headless setup

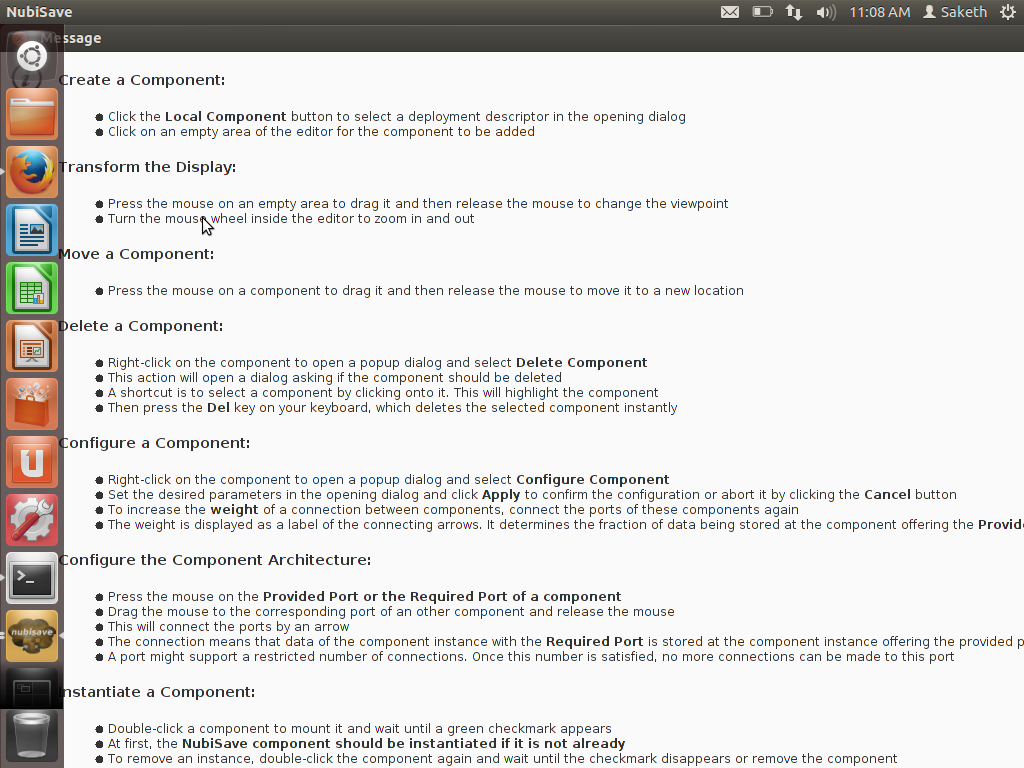
(installs the cloudfusion splitter module which enables users to mount their cloud onto an online server like dropbox or sugarsync through mountpoint).

* 1. Working

3.3.1 The Nubisave controller works like the following-



3.3.2For getting help you can refer to the help message bar in the NubiSave cloud storage controller flow editor.



Now that you have added the component on to the NubiSave controller, it’s activated when you right click on it as mentioned in the above Help message bar.

So, after you do it, it shows a green arrow and a marking on it to specify that it has been mounted. Since, we use dropbox for cloudfusion , we have to configure it and add it to the web cloud server space. To really configure and activate it onto the system, you need to provide your Cloud server’s authentication details and configure it for it to work on the Nubisave Controller.

3.4 Configuration

Cloudfusion can be configured in a couple of ways,

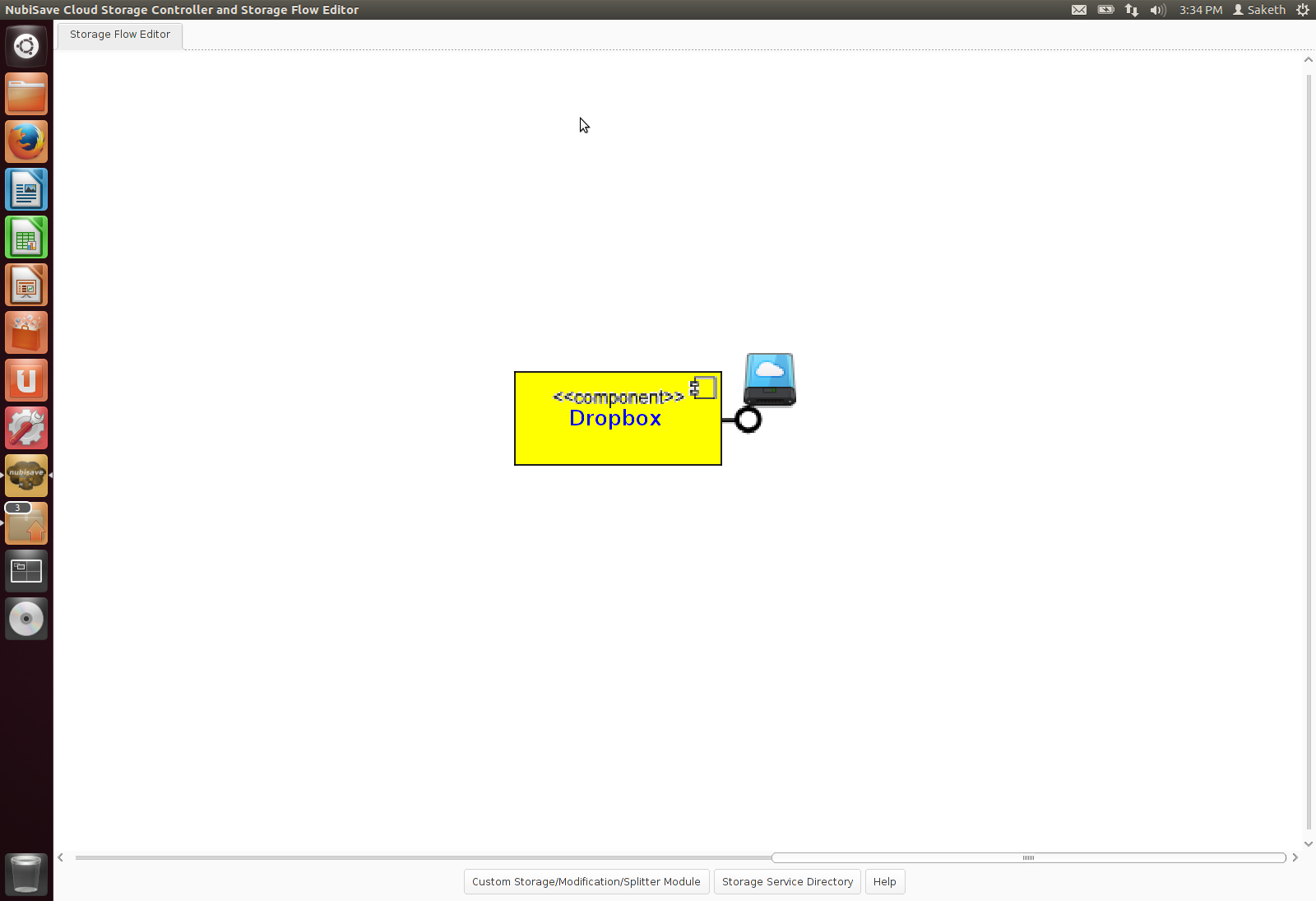
1. Graphically from Nubisave using an authentication on to the server.
2. From the editor to permit it to access it into a local device in your computer.

First, let us understand how it can be configured from your graphical NubiSave tool.

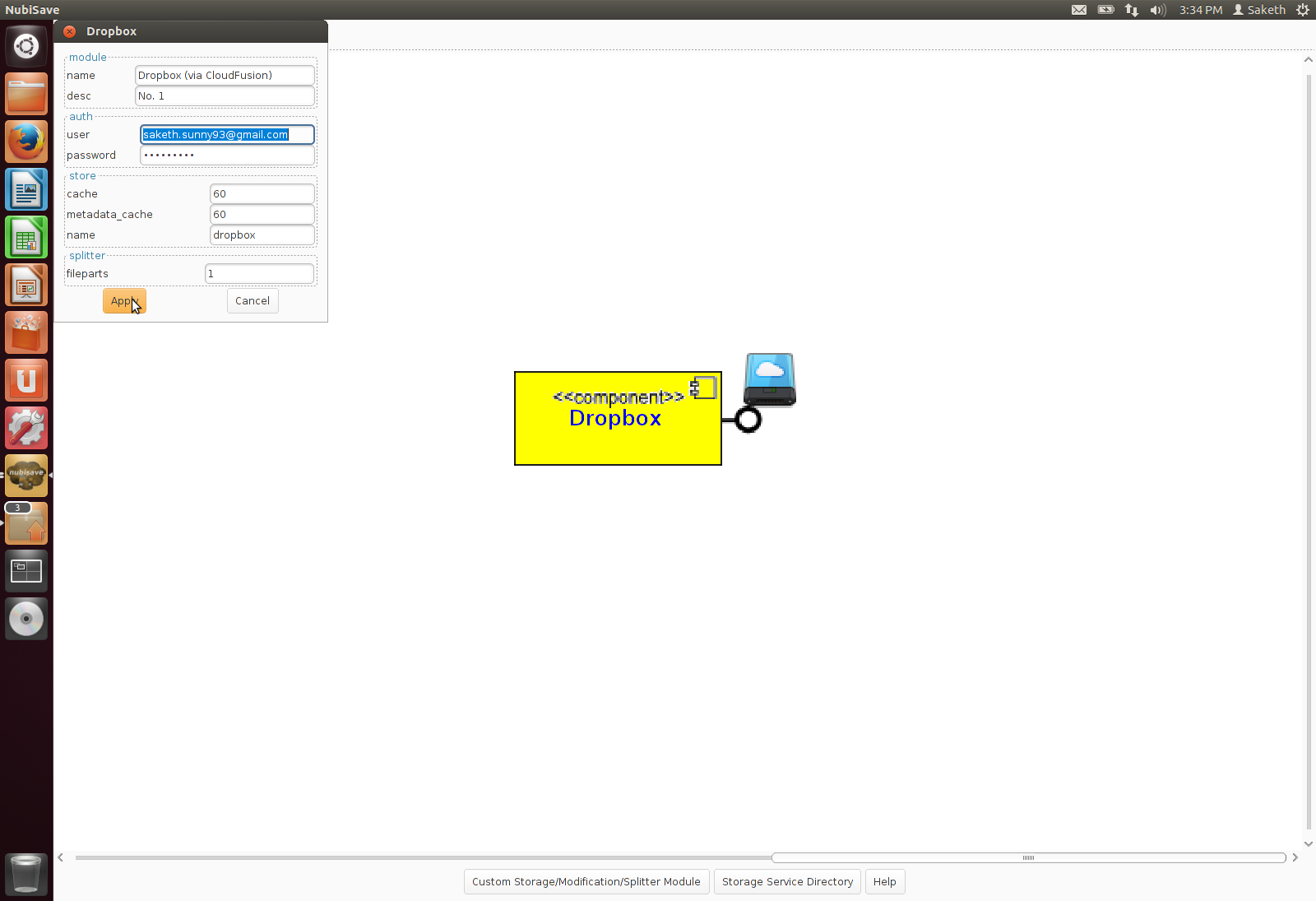
The following are the steps to configure it along with the authentication.

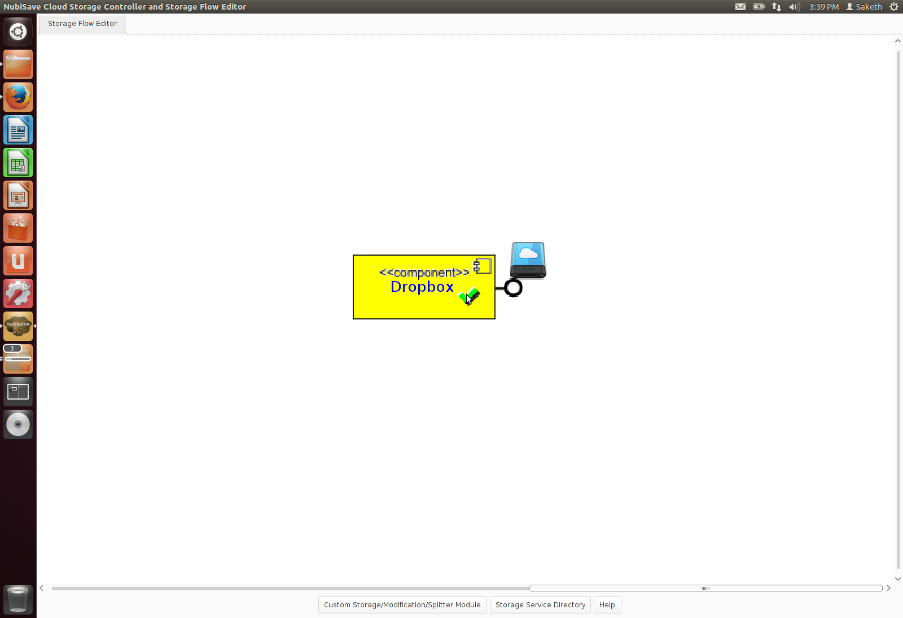
3.4.1 Graphical

1. Firstly, you must select the dropbox item from the splitter and then drag it on to anywhere on the screen to select it. Do not mount it yet.



1. Now, to configure the dropbox on to the server you have to right click on it and update your details of the actual dropbox account and let it authorize to successfully configure it.



1. Now that it has been configured, mount it.To mount it, you can either right click and select mount or double click, you will understand that it has been mounted once you get a tick mark on it.

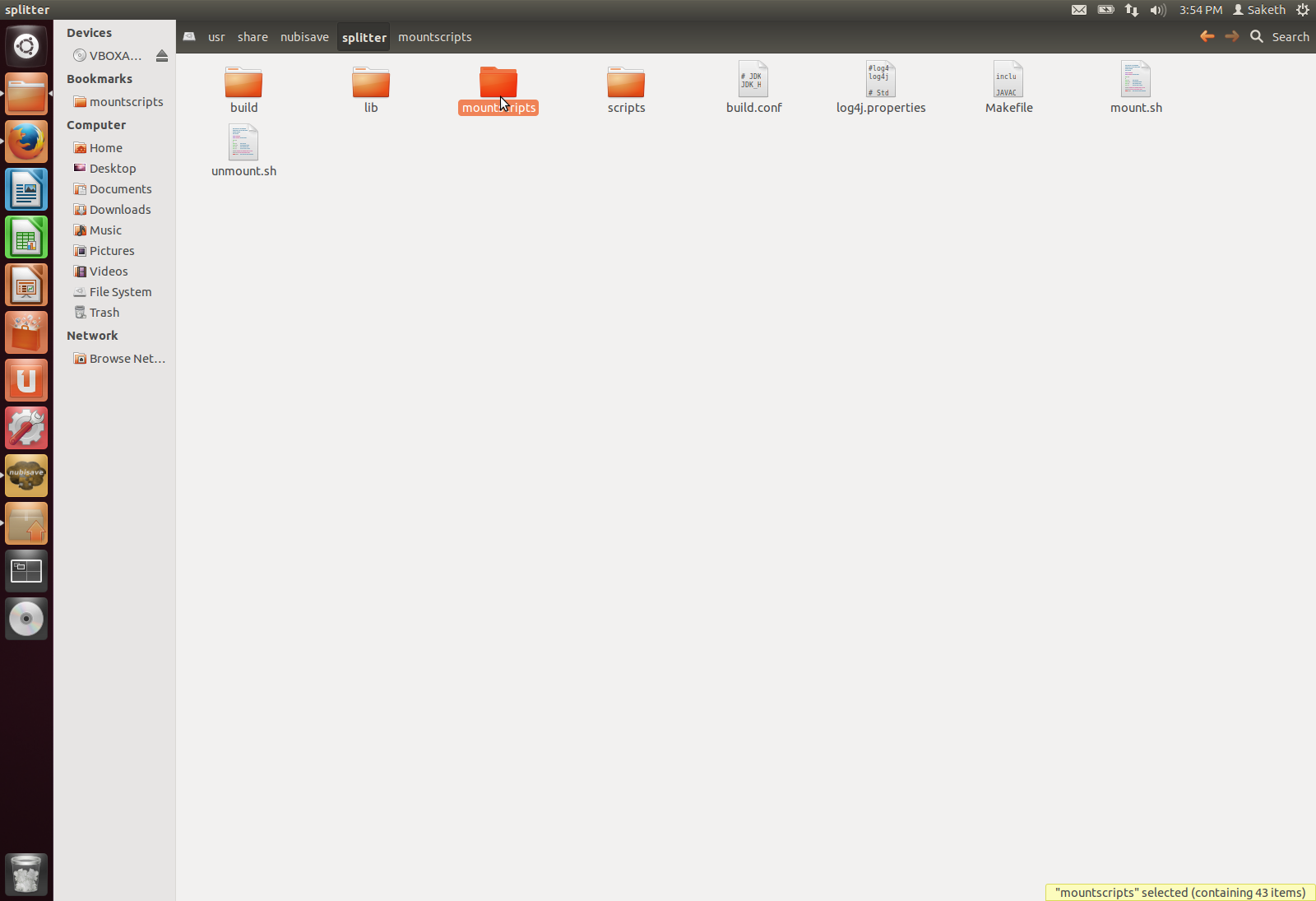
Once you update into the real system, it saves ini files and those are the ones that store the information for the particular server. Let us see the file for dropbox.

To access it in the editor via cloudfusion and mount it, we should use the following command-

cloudfusion --config/usr/share/nubisave/splitter/mountscripts/dropbox.ini

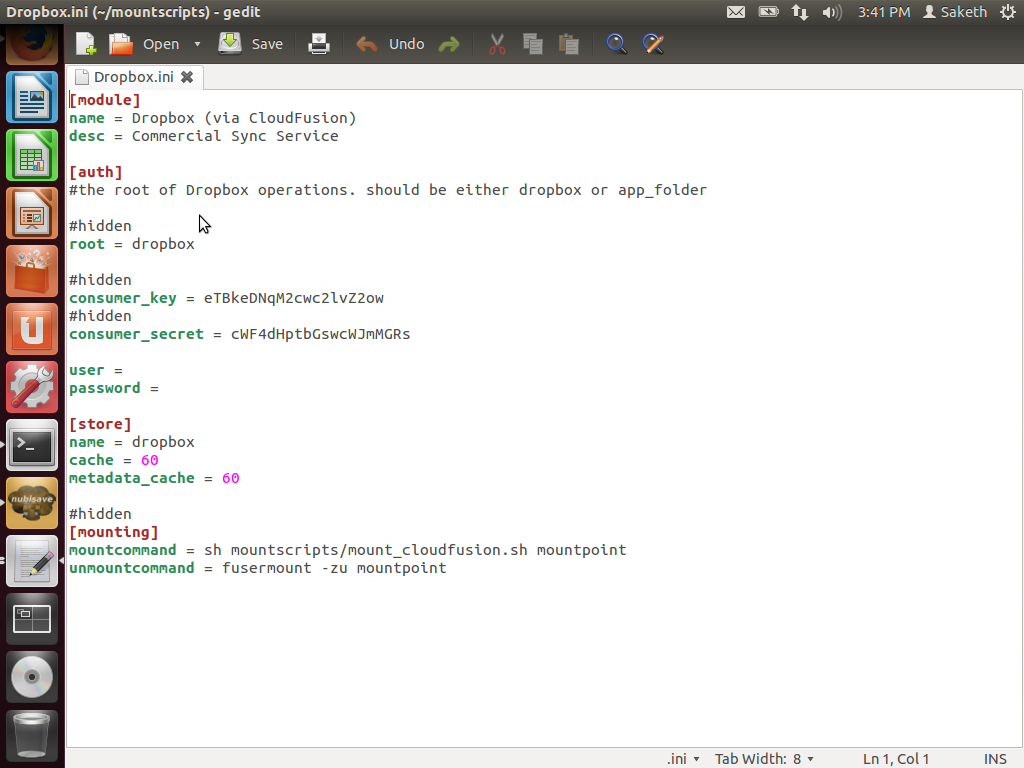
The above command is used to check the .ini files in the main directory.

To access it from your home directory you must follow the understated procedure.

1. Copy the folder mountscripts from your nubisave global storage and paste it in the home folder.
2. Now that you have done so, call the .ini files from that and configure them to make the dropbox run.

Now, access the .ini files of the home folder’s mountscript from the editor.

The .ini file of dropbox can be viewed as the following.



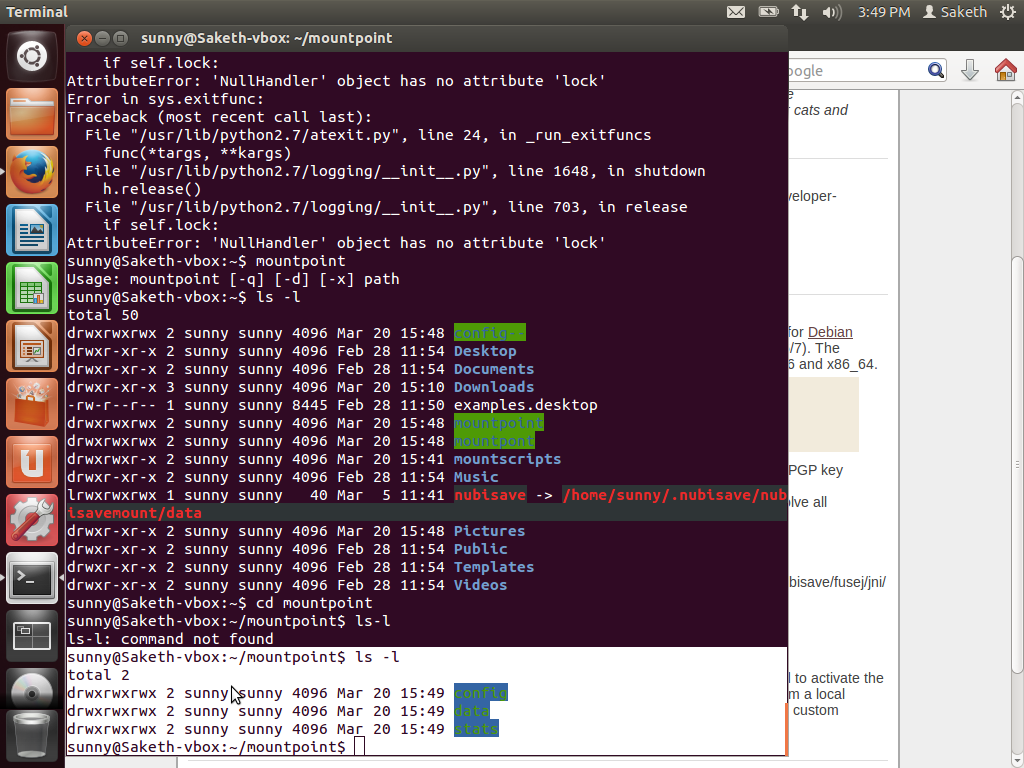
3.4.2 Editor

The next step that follows is configuring it on the editor.

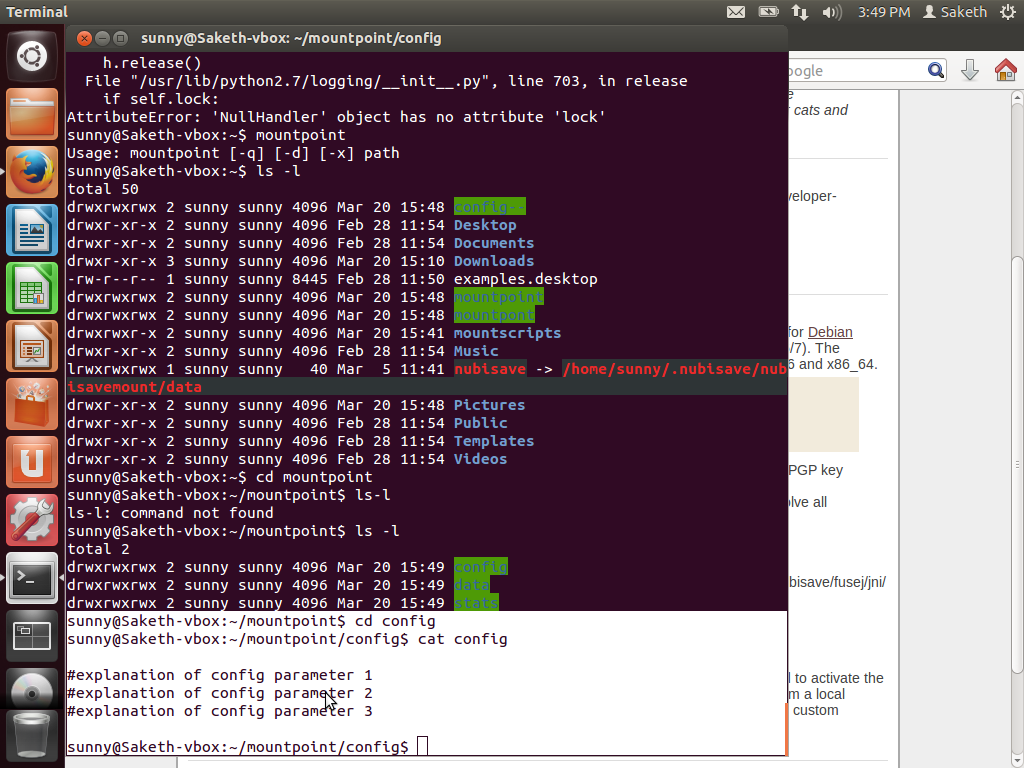
Once all the installations are done and it is ready to be configured, you open mountpoint to find three different files called Data, Stats and config.

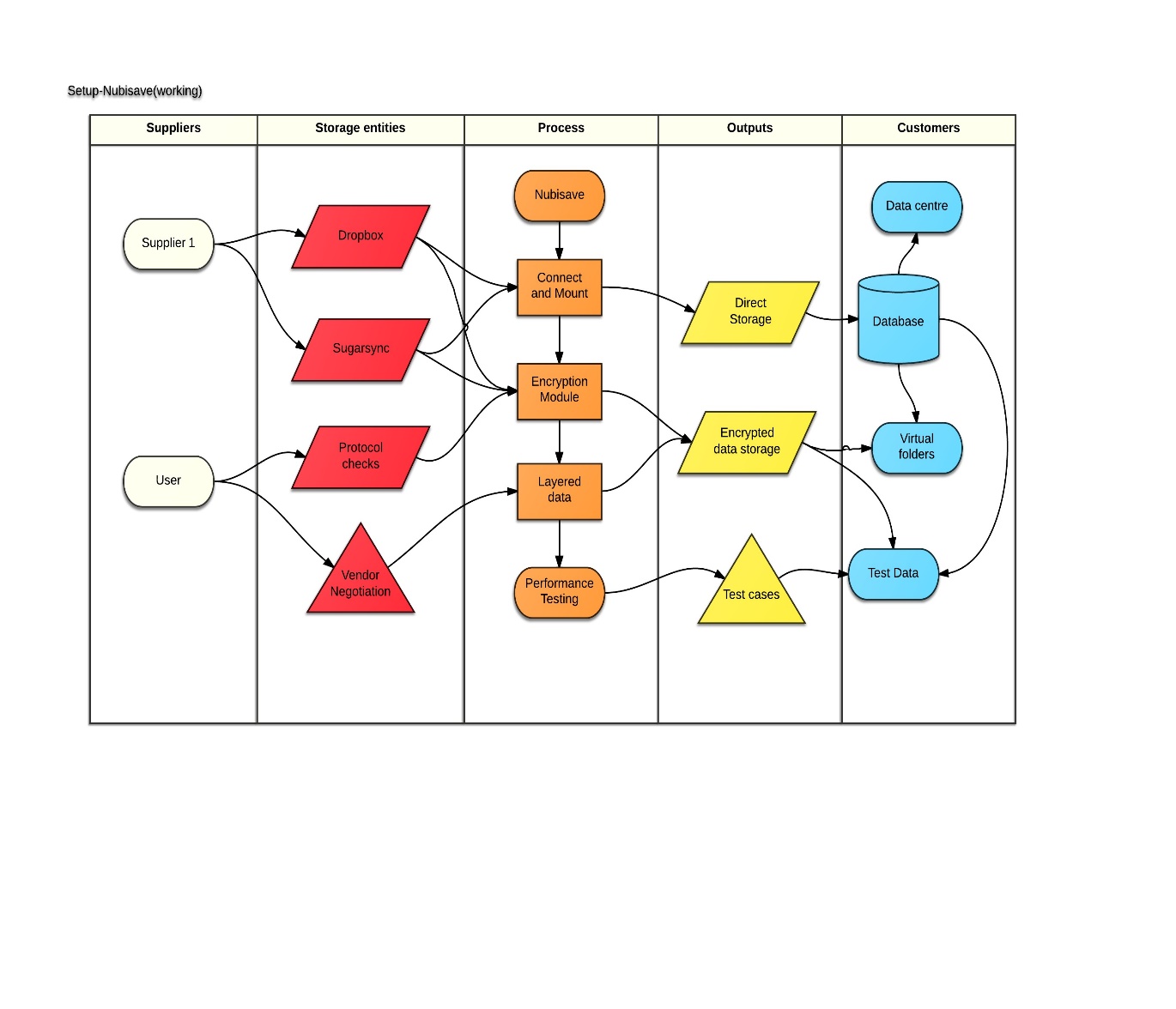
a. Data consists of any data that is actually stored in the file.

b. Config has all the messages about the file like a metadata.

c. Stats is something that has a list of all the uploaded data, errors occurred and the virtual data that is present in the model.

Now, let us see what the file config contains through a pictorial demo.



3.5 Activity Diagram on the working model

4 System Check

4.1 Problem with the existing system

In a normal cloud computing system, there are a few redundancy issues to be addressed. So, we try to move on to a cloud storage controller.

1. There is no organized way of storing, rather only chunks of data is directly stored.
2. When we try to separate the normal data from the system, it has to involve a tinker in the metadata and through this we can easily retrieve any data we like without altering any other interior storage.
3. The response time to reach or save the data is very high in a normal cloud system.

4.2 Proposed system

With this new application Nubisave being designed, we can thoroughly address all

Problems mentioned above and can comprehensively solve then to reach the data and

fix performance issues never to occur while data retrieval. With its high storage

facility and the use of the servers, it even offers distributed data visualization and

browsing with the support of another web application called as NubiVis.

5 Phase II : Architecture and Performance testing

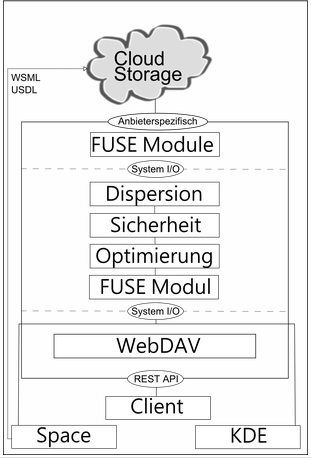
To get an idea about this architecture, let us know more about it from a user

perspective with more archictectural views and examples on how it works and the

performance of its tasks.

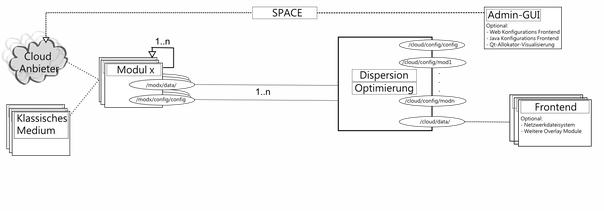
The task of NubiSave team is to develop an integrated desktop, cloud provider-wide tool so that the backup on heterogeneous cloud storage services are controlled with regard to the nature backup data. This requires a cloud abstraction and Selection component and a graphical front-end to be implemented for providers across to select the appropriate service storage. Through this service platform, a user-friendly manner is being created, its data redundantly at different storage provides stores, so that a significantly larger security server failure doesn’t arise. By splitting the files and distribution at different Company is reducing the chances of anyone to see the data getting stored or used. For the user NubiSave offers the advantage of his data being backed in the Cloud. Data is no longer manually uploaded to the different storage providers that are available, and still keeps track of where the data is saved in the server. It may increase its files in a virtual folder, which are then hoisted in the background the data redundancy on the various provider is typically distributed.

Let’s take this chance to look at its architecture which has been developed over various modules of vigorous improvisation and constant development.



INTERNAL ARCHITECTURE OF THE SYSTEM

After the above model prototype architecture, we arrive at the actual base of architecture



DISPERSION OPTIMISATION / DISPERSION OPTIMIERUNG

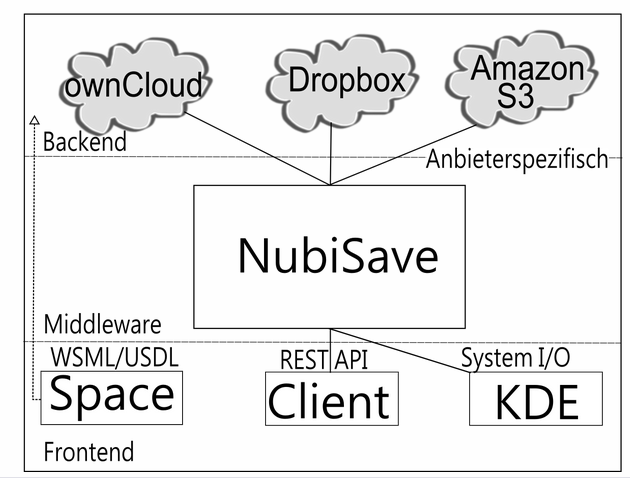
Now, we have understood the architecture from a developer point of view, let

us see how a user or an external agent sees the system with Nubisave in place.

This incorporates the client, the kernel and the user space to make a connection and

store their data and large or paid cloud data storage or saving facilities like dropbox,

cloudstream or Amazon.



EXTERNAL ARCHITECTURE

5.1 Working

Now, let us assume that a user or a client has successfully registered into the

application and connected to the cloud device. Nubisave incorporates a frontend that

lets the users who are registered in the system as Cloud storage providers to set

certain service parameters based on quality and requirement of their wish. This is also

a variant of the green cloud which is cost saving and less expensive developed in

FlexCloud in the European research where we don’t throw every possible data but

only the necessary data is pushed from the other side. For example, while dealing with

an organization through Nubisave one such encountered case will be demonstrated

below. When a provider is willing to pay a certain amount of price for the storage service he has to fill in details in the box that follows.

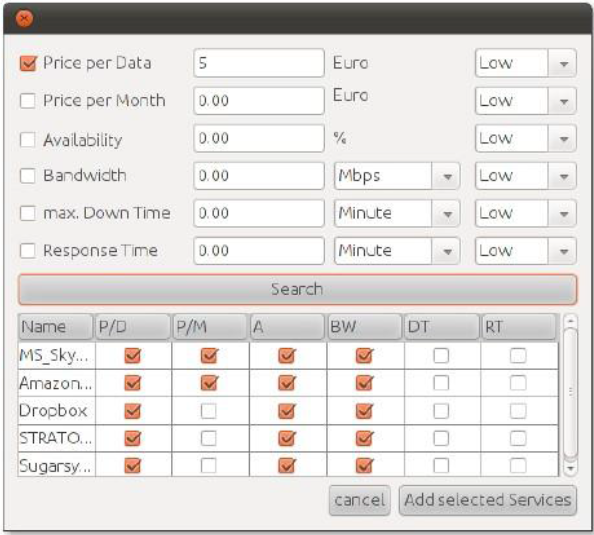


TABLE 5.1.1

It allows the selection of various parameters shown above and once you try to search with the parameters required to the user, you will find all services that host your data and offer a similar approach. In the next step, the provider is chosen manually and you have the details of the provider where he can alter his username/password and create other logins for his application. These providers also include the ones that have an account created directly from the supplier or through the software and can change the details to their willingness. One condition is that should be satisfied is the fact that these credentials must be known or stored in the Nubisave interface to be present on the front.

Below shown is a data element that clearly explains the above mentioned entity through the software.

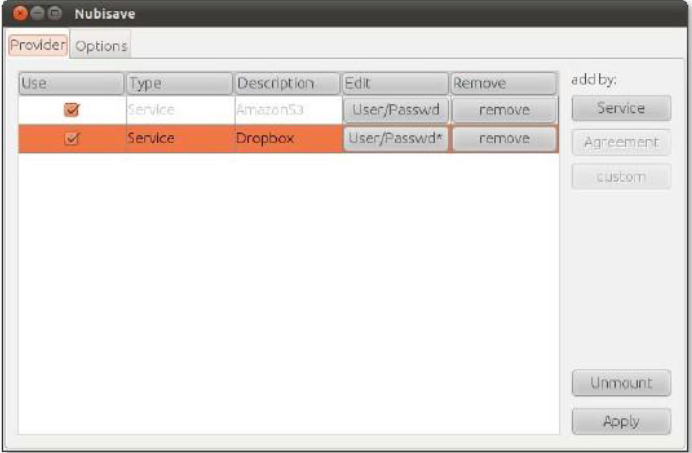
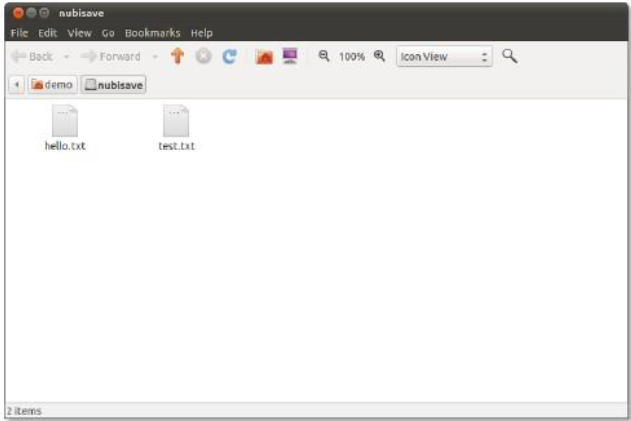


TABLE 5.1.2

Once the Apply is clicked, it is configured for users and will start creating Virtual Folders in the Background.

A virtual folder can also be created in the options tab of the GUI, you can perform changes like altering the percentage of the redundancy and how much it affects the situation. Higher the value more redundant data is stored. As the virtual folders are ready, files can easily be copied into it and are distributed automatically to different providers

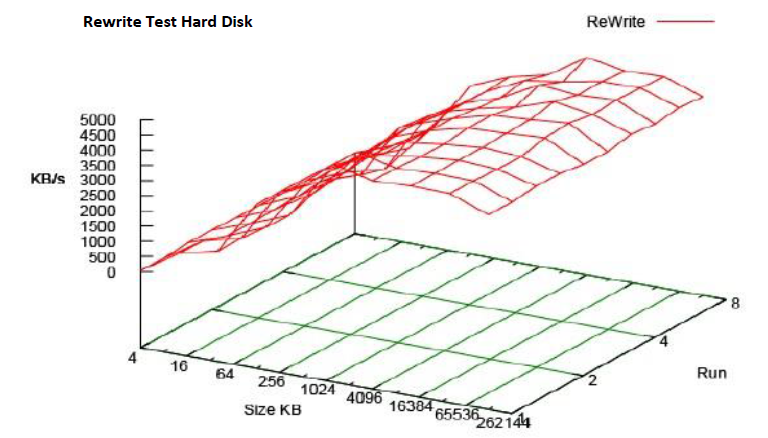


5.2 Performance Testing

To understand the progress of the system is to allow the performance of it and see how efficiently it works in overhead and faulty cases. In the following example we perform speed testing, the access of the data in rewrites on a memory block can be understood here from these cases.

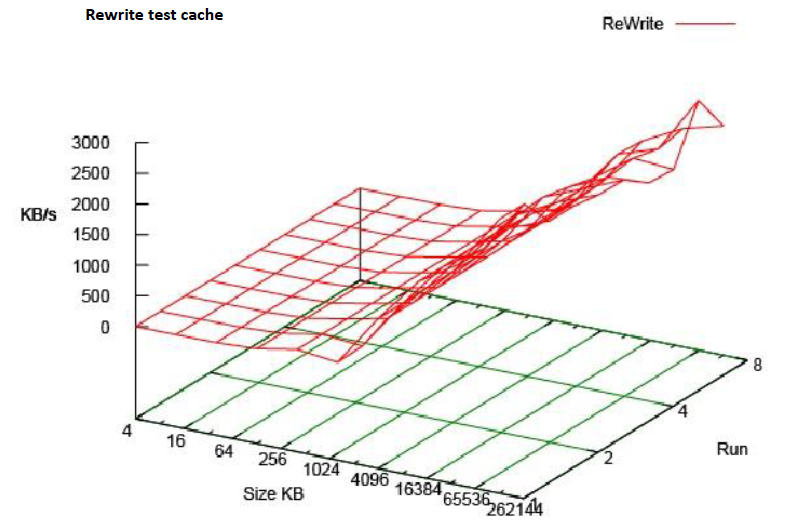
5.2.1 Rewrite test on hard disk

A rewrite is something that first requires a read access and then a write access on the data, the following graph shows how much data is being run on the system. It shows the speed of rewrite with which you access the hard disk with different sizes of data and it is observed that a hard disk gives results in an average speed increasing for increase in data. The number of passes has barely minimum effect on the data set.



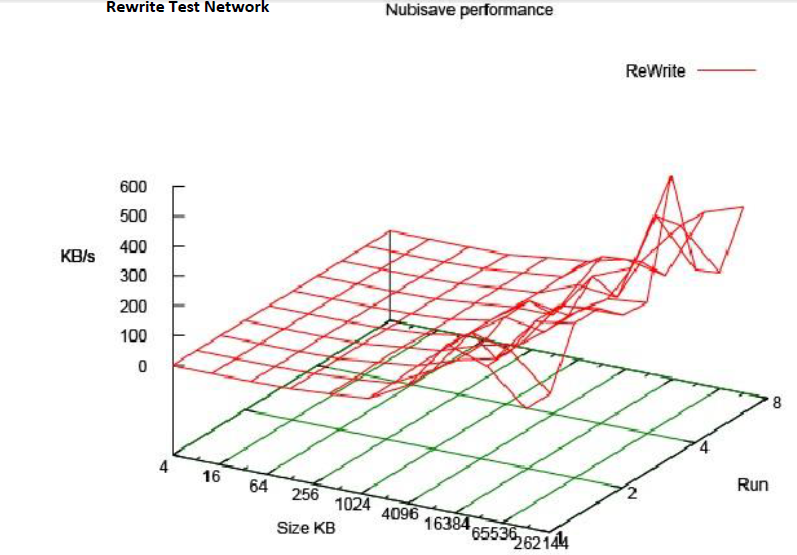
5.2.2 Rewrite test on cache

A step prior to the data entrance into Nubisave the upload of data actually doesn’t take place, it splits all the necessary files and keeps them in the data cache. The speed is very slow for small files as the data is not actually considered. Once the size crosses 1 MB of data then it increases from a rapid 3200 KB /s and keeps on growing. It is majorly used by private or home users who have a certain amount of data and not a wide amount to be used.

 The following graph shows the invariant increase in speeds after 1Mb and a constant pattern after reaching a certain limit.

5.2.3 Rewrite test on the network

The final and most unpredictable test is the one when a rewrite test is run on a network. Here the speed is varied randomly without a control over a bottleneck. It has a relatively slower rate compared to the others but no stability on itself. Here the only thing that matters is a hardware configuration and no amount of data or structure effect the speed in any way. It has to be executed between the server and the user.



6 Actual Prototype and Functioning

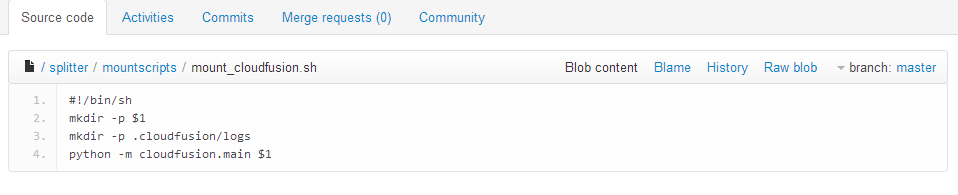
6.1 Implementation (Presentable modules)

In this we see some important features that let us understand how to

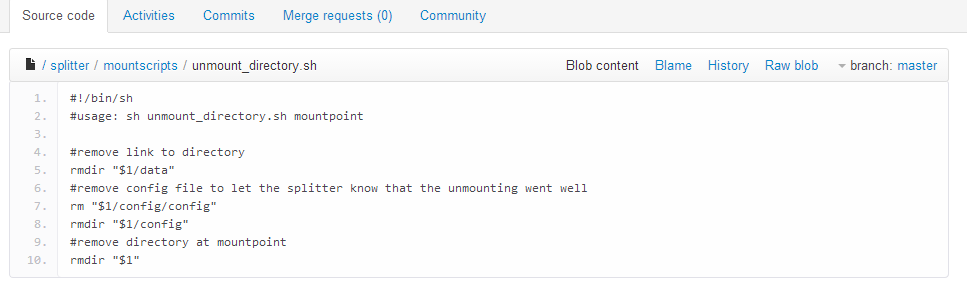
1. Mount/Unmount cloudfusion
2. Rebuild Nubisave on to the system.
3. An .ini file of dropbox
4. Understanding Gitorious(a complete maintenance and collaboration system)

6.1.1 Cloudfusion

6.1.1.1 Mounting cloudfusion



6.1.1.2 Unmounting the directory mounted cloudfusion



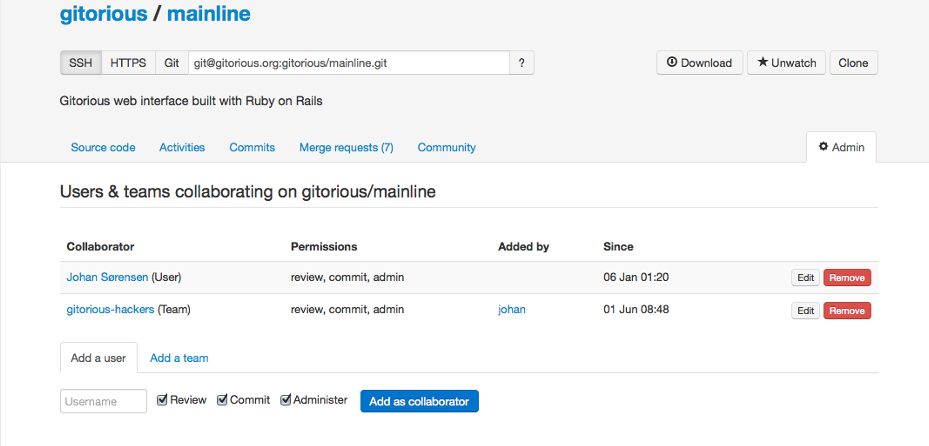
6.1.2 Rebuild Nubisave

1. #!/bin/sh
2. if [ ! -d debian ]; then
3. echo "Error: Needs to be called from top-level directory as ./debian/rebuild" >&2
4. exit 1
5. fi
6. if [ "$1" = "--prep" ]; then
7. apt-get install devscripts dh-make
8. apt-get install openjdk-6-jdk python-setuptools libfuse-dev
9. exit 0
10. fi
11. which debuild >/dev/null && which dh\_make >/dev/null
12. if [ $? != 0 ]; then
13. echo "Error: Needs build tools and dependencies. Call script with --prep first."
14. exit 1
15. fi
16. if [ ! -d CloudFusion ]; then
17. # git submodule init
18. # git submodule update CloudFusion
19. git clone git://github.com/joe42/CloudFusion.git
20. fi
21. if [ "$1" = "--force" ]; then
22. rm -f ../nubisave\_\*
23. git clean -df
24. fi
25. stamp=`date +%Y%m%d`
26. rm -rf debian/nubisave
27. echo y | dh\_make -i --createorig -p nubisave\_0.0~git$stamp
28. if [ ! -f ../\*.orig.tar.gz ]; then
29. # Workaround for old squeezy dh\_make
30. origdir=$PWD
31. cd ../\*.orig
32. tarname=`basename $PWD`
33. tarname=`echo $tarname | sed -e 's/-/\_/'`.tar.gz
34. tar cfz ../$tarname .
35. cd $origdir
36. fi
37. debuild

6.1.3 An example of a .ini file of dropbox

1. **[auth]**
2. #hidden
3. server = api.dropbox.com
4. #hidden
5. content\_server = api-content.dropbox.com
6. #hidden
7. port = 80
8. #hidden
9. request\_token\_url = https://api.dropbox.com/0/oauth/request\_token
10. #hidden
11. access\_token\_url = https://api.dropbox.com/0/oauth/access\_token
12. #hidden
13. authorization\_url = https://www.dropbox.com/0/oauth/authorize
14. #hidden
15. trusted\_access\_token\_url = https://api.dropbox.com/0/token
16. # the root of Dropbox operations. should be either dropbox or sandbox, depending on your app's setup
17. #hidden
18. root = dropbox
19. # key and secret granted by the service provider for this consumer application - same as the mockoauthdatastore
20. consumer\_key =
21. consumer\_secret =
22. user =
23. password =
24. **[store]**
25. name = dropbox
26. cache = 60
27. metadata\_cache = 60
28. #hidden
29. **[mounting]**
30. mountcommand = sh mountscripts/mount\_cloudfusion.sh mountpoint
31. unmountcommand = fusermount -zu mountpoint

6.1.4 Understanding Gitorious

 When you act as a collaborator to the team of the development of a system, Gitorious allows you to maintain it as code repository or a GIT repository for all your files which can be hosted on for the others to view and make changes on them. Let us see a part of the other team members sharing it over the software to access easily for a personal or professional use.

6.2 Demonstration

To demonstrate more about Nubisave users should understand it uses standard servers and **π-box** for its database storage work which is an orientation of PHP and My SQL lite and it contains all the servers and storage unit on the cloud as a backend and the Graphical frontend is the result i.e. Nubisave.

You can perform various operation like checking whether it is an external or an internal layer, since it is a cloud you have to check whether it is a public service or a system service, the final and most important prototype is to check whether it is an encryption layer or not. If yes how is the data stored?

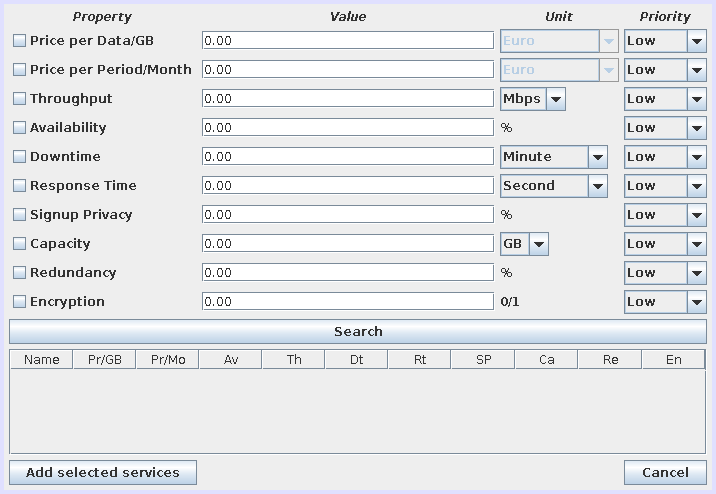
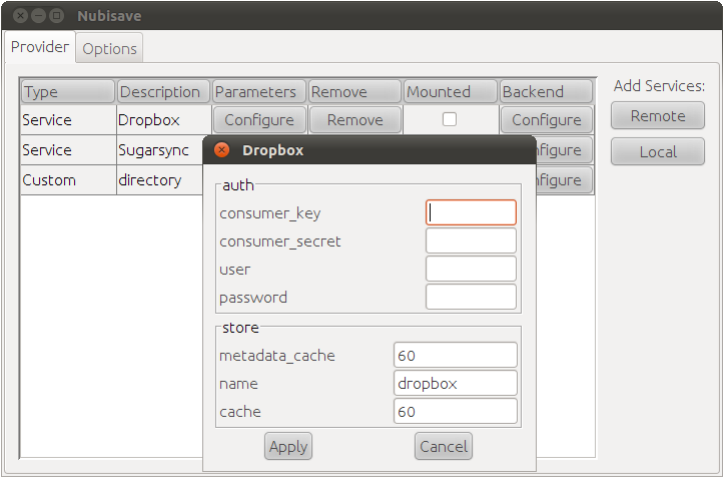


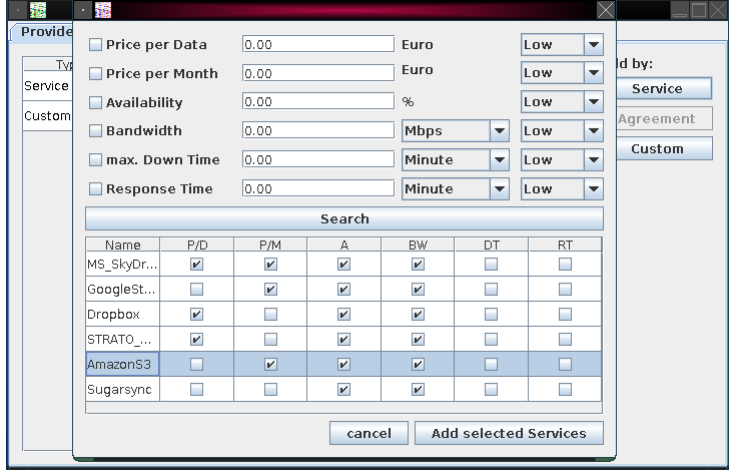
TABLE 6.2.1

The above table shows all the factors that can be varied by the user to make his storage efficient.

Specific details can be added like throughput and combination of its technical power and much more. Users can also provide authentication details to their dropbox account so that it is an easy task for Nubisave to control and authorize such that it becomes less complicated to a layman and people who have no or less knowledge of the software.



6.2.2 Yet Another Configuration Element



6.2.3 Non Functional Properties are also satisfied in this

7 Conclusion and Result

Through these results we can understand that it has been a very comfortable layout for its data on various storage providers. As it follows a modular system, it is provided with more functionality. It is important to once again note that there can be direct contact to the vendors through the GUI or Nubisave itself. When you register yourself separately you can store data in Nubisave through that. A protocol that can allow the user and the company direct access to the system through Nubisave is the WS-Information Protocol.

An encryption or a steganography module would be interesting to Security of its data when it continues to bind data between certain companies. Presently, the data is in Plain text and stored unencrypted on the provider - only some fragmentary, such that a few parts of the module of a single company is not completely necessary and will provide all the information together. The best idea in development would be a multi-PC server module which expands Nubisave to work on more than one module at a time. The tables for each PC can be encrypted so that they provide the details to join them on to another PC.

7 References

1. Josef Spillner, Alexander Schill, and Cordelia Schmid. :Toward Dispersed Cloud Computing
2. Josef Spillner, Johannes Schad, Stephan Zepezauer . :Personal and Federated Cloud Mangement Cockpit. Praxis der Informationsverarbeitung und Kommunikation 36(1):44(2013)

[3] Josef Spillner, [Alexander Schill](http://www.informatik.uni-trier.de/~ley/pers/hd/s/Schill:Alexander.html): Orchestration of Distributed Storage Targets through Storage Flows. [CloudCom (2) 2013](http://www.informatik.uni-trier.de/~ley/db/conf/cloudcom/cloudcom2013-2.html#SpillnerS13): 349-354

[4] Josef Spillner, [Johannes Müller](http://www.informatik.uni-trier.de/~ley/pers/hd/m/M=uuml=ller:Johannes.html), [Alexander Schill](http://www.informatik.uni-trier.de/~ley/pers/hd/s/Schill:Alexander.html): Creating optimal cloud storage systems. [Future Generation Comp. Syst. 29](http://www.informatik.uni-trier.de/~ley/db/journals/fgcs/fgcs29.html#SpillnerMS13)(4): 1062-1072 (2013)

[5] Josef Spillner, [Alexander Schill](http://www.informatik.uni-trier.de/~ley/pers/hd/s/Schill:Alexander.html): π-Control: A Personal Cloud Control Centre. [CoRR abs/1202.0970](http://www.informatik.uni-trier.de/~ley/db/journals/corr/corr1202.html" \l "abs-1202-0970" \t "_self) (2012)

[6] J. Spillner, G. Bombach, S. Matthischke, J. Müller, R. Tzschichholz, A. Schill: Information Dispersion over Redundant Arrays of Optimal Cloud Storage for Desktop Users. 4th IEEE/ACM International Conference on Utility and Cloud Computing ([UCC](http://www.cloudbus.org/ucc2011/)), pp. 1-8, Melbourne, Australia, December 2011.

[7] J. Spillner, M. Quellmalz, M. Friedrich, A. Schill: peaCS - Performance and Efficiency Analysis for Cloud Storage. Workshop of Cloud Storage Optimisation ([CLOUSO](http://clouso.unime.it/)) at the 2nd European Conference on Service-Oriented and Cloud Computing ([ESOCC](http://esocc2013.lcc.uma.es/)), Málaga, Spain, September 2013.

[8] When Clouds Disperse C.Kauba , S. Mayer{seminar : Seminar aus Informatik –SS2013} Salzburg ,Austria 25.05.2013

[9] Josef Spillner, Gerd Bombach, Steffen Matthischke, Johannes Müller, Rico Tzschichholz and Alexander Schill: Data Processing systems and security in the cloud.

2011 Fourth IEEE International Conference on Utility and Cloud ComputingCFP11UCC-PRT 978-1-4577-2116-8

[10] Article Josef Spillner (Cloud Computing –A complete storage) Personal and Federated cloud management cockpit.CloudRemix and FlexiSource