

Cloud based IT Infra with Central Identity

Phase II – Project Report

Project Guide

T. Chandra Shekar
Dept. of CSE – RGUKT Nuzvid
chandra.indra@gmail.com

Project Team

T. Aneesh Kumar	N090247
P. Nageswarao	N091030
P. Anesh	N090977
P. Jyothi Ram	N090990
K. Naresh Chowdary	N090331
N. Venkata Sateesh	N090935
M. Sanyasi Rao	N090891



Dept. of Computer Science and Engg.
R.G.U.K.T. - Nuzvid
Krishna Dt. - Andhra Pradesh - 521202

Jan 2015 – Apr 2015

Abstract

The main objective of “Cloud based IT Infra with Central Identity” Phase II is provide the implementation to our objectives.

Implementing the Web based central identity, network based central identity, achieve the combination and deploy all these in the private cloud

Contents

1	Introduction	1
1.1	Introduction	1
1.1.1	Private Cloud	1
1.1.2	Deploying Network Services	1
1.1.3	Central Identity	1
2	Phase I Work	2
2.1	Components	2
3	Phase II Work	3
3.1	Components	3
4	Web based Single Sign-On	4
4.1	OAuth Provider	4
4.1.1	OAuth Protocol	4
4.2	University User Profiles	5
4.3	REST API	6
4.3.1	Technologies Used	6
4.3.2	API Testing	7
4.4	Roles & Permissions	9
4.5	PHP Client Libraray	10
4.5.1	Initializing the Client Library	10
4.5.2	Get Authorization URL	10
4.5.3	Get Access Token	10
4.5.4	Initializing API with Access Token	10
4.5.5	Getting User Info from API	10
5	Network Single Sign-On	11
5.1	Introduction	11
5.2	LDAP Server	12
5.2.1	Installation and Configuration	12
5.3	phpLDAPAdmin	13
5.3.1	Installation and Configuration	13
5.3.2	Web Interface of phpLDAPAdmin:	13
5.4	LDAP Client:	16
5.4.1	Installation and Configuration:	16
5.4.2	Log In as an LDAP User:	17
5.5	HAProxy	18
5.5.1	Installing HAProxy	18

5.5.2	Configuring HAProxy	18
5.6	GlusterFS	19
5.6.1	Configure DNS solution	19
5.6.2	Install server components	19
5.6.3	Create a storage volume	19
5.6.4	Install and configure client components	19
5.6.5	Restrict access to the volume	20
5.7	XtreemFS	21
5.7.1	Installation, Creating & Mounting Filesystem	21
5.7.2	Distributed Step	21
6	Private Infrastructure Cloud	23
6.1	Openstack Architecture	23
6.2	Installation	24
6.2.1	NTP	24
6.2.2	MySQL	24
6.2.3	Rabbitmq-server	24
6.2.4	Keystone	24
6.2.5	Glance	24
6.2.6	Nova	24
6.2.7	Neutron	24
6.3	Virtual Machines	25
7	Conclusion & Future Work	26
7.1	Conclusion	26
7.2	Future Work	26
8	References	27

List of Figures

4.1	OAuth Protocol Work Flow Diagram	4
4.2	User Profile	5
4.3	Edit User Profile	6
4.4	CRUD Operations	6
4.5	List of Roles of User	9
4.6	Adding Permissions Option	9
5.1	phpLDAP	13
5.2	Complete category of LdapServer	13
5.3	User Creation	14
5.4	Group Creation	14
5.5	Adding User to Group	15
5.6	Groups information	15
6.1	Openstack Architecture.	23
6.2	Openstack Virtual Machines.	25
6.3	Openstack Resource Pool.	25

Chapter 1

Introduction

1.1 Introduction

“Cloud Based IT Infra with Central Identity” is a complete solution, based on private cloud to enhance and efficient utilization the IT Infrastructure of an emerging Universities and Organizations with Central Identity for all its users to access its services.

It is going to be developed in 3 phases

- *Private cloud*
- *Deploying Network Services*
- *Central Identity*

1.1.1 Private Cloud

Private Cloud establishment is targeted for hardware resource pooling, providing high computational and scalable virtual machines for deploying network based applications (smtp, proxy, ftp), web application and Network storage.

1.1.2 Deploying Network Services

Configuration of Uniform hardware experience over the complete university includes single sign on on every device, configuration of mail servers etc.

1.1.3 Central Identity

Essential part that combines normal network services(proxy, mail, etc.) and organizational web & native applications. In addition to that this central identity is available to thrid party developers as API with dynamic based role user authentication protocols.

Chapter 2

Phase I Work

As part of Phase I, we have done literature survey and analyzed feasibility of the several components

2.1 Components

- Central Identity
 - Single Sign-On with REST API
 - Identity Management
 - Dynamic Role Based Access Control
- Network Based Central Identity
 - LDAP Servers
 - NFS Servers
- Cloud Computing
 - Cloud Characteristics
 - Service Models
 - Deployment Models
- Private Clouds
 - Introduction
 - Open Source Tools

Chapter 3

Phase II Work

As part of Phase II, we have tried to implement some of the above mention components

3.1 Components

- Web based Signle Sign On
 - OAuth Provider
 - University Users Profiles
 - REST API
 - Support of assign roles to users with their permission set
 - Testing oauth client library in PHP using php-curl
- Network Components
 - LDAP Server
 - NFS Server
 - Haproxy
 - GlusterFS
 - XtreamFS
- Private Infrastructure Cloud
 - Openstack Architecture
 - Installation
 - Virtual Machines

Chapter 4

Web based Single Sign-On

4.1 OAuth Provider

Here in this section the overall workflow of Authentication in a Single Sign-On System is explained. In order to Authenticate users with the given credentials we must use a robust and stable protocol. And many Single Sign-On systems uses OAuth protocols for this purpose. Single Sign-On uses OAuth protocol for both Authentication & Authorization.

4.1.1 OAuth Protocol

OAuth is an authentication protocol that allows users to approve application to act on their behalf without sharing their password. The below figure explains the flow of Authentication in OAuth Protocol.

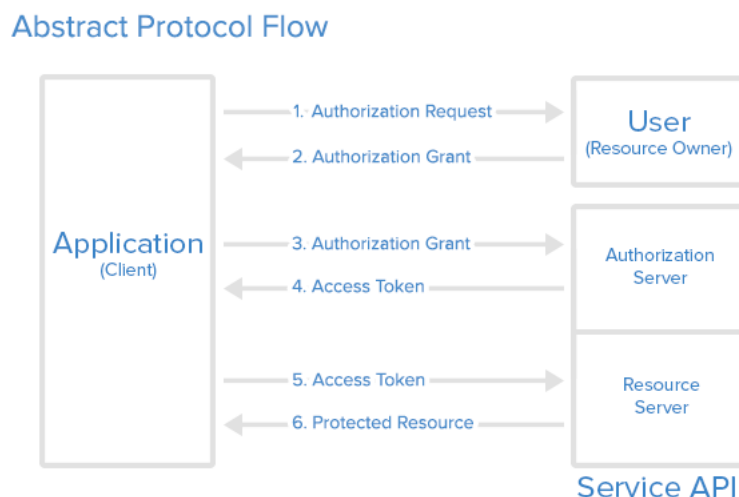


Figure 4.1: OAuth Protocol Work Flow Diagram

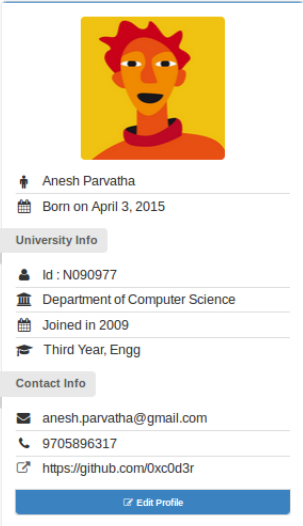
Steps invloved in the above flow diagram


- The application requests authorization to access service resources from the user
- If the user authorized the request, the application receives an authorization grant

- The application requests an access token from the authorization server (API) by presenting authentication of its own identity, and the authorization grant
- If the application identity is authenticated and the authorization grant is valid, the authorization server (API) issues an access token to the application. Authorization is complete.
- The application requests the resource from the resource server (API) and presents the access token for authentication
- If the access token is valid, the resource server (API) serves the resource to the application

4.2 University User Profiles

Online Portal v2.0
Notice Board
Departments
Developers
N090977





Anesh Parvatha

Born on April 3, 2015

University Info

Id : N090977

Department of Computer Science

Joined in 2009

Third Year, Engg

Contact Info

anesh.parvatha@gmail.com

9705896317

https://github.com/0xc0d3r

Edit Profile


Summary

Well I'm very passionate about secure coding and cyber security. As we all know that, India is one of the cyber effected countries. So I want to bring awareness in cyber security to people. I'm participating in Capture The Flag hacking contests and learnt different attack vectors. So I've got experience in Crypto, Stegano, Web and Forensics.

Education

Period	School / College / University	Degree	Area of Study	Percentage
2004-2009	ZPHS Sankhavaram	SSC	ALL	91.0
2009-2011	RGUKT Nuzvid	PUC	M.Bi.PC	89.2
2011-2015	RGUKT Nuzvid	B.Tech	CSE	85.0

Experiences


Internship
June - July 2014

Developed a steganograhpy tool

@ CMC Ltd - Hyderabad

Positons & Designations

DBA True

Figure 4.2: User Profile

Online Portal v2.0
Notifications
Departments
Developers
N090247

Edit Profile » Contact Info
Back to Profile

Website / Blog URL

Email Address

Mobile No

Update

Edit Profile
Contact Info
Profile Pic
Education
Experiences
Position & Roles
Interested Areas
Technical Skills
Achievements
Summary

Figure 4.3: Edit User Profile

4.3 REST API

REST stand for **RE**presentational **S**tate **T**ransfer. It's a collection of simple URIs, and HTTP calls like GET,PUT,POST,DELETE to those URIs to get some Protected data from a Resource Server. Once User provides a set of valid credentials to OAuth Provider it will generate an access token to that particular user, with that token user can fetch protected resources from the resource server by making basic HTTP calls through REST API. REST API provides users a flexibility to perform Basic CRUD(Create,Read,Update,Delete) on resource server.

operation	HTTP verb	action
Create	POST /posts	create
Read	GET /posts/1	show
Update	PUT /posts/1	update
Delete	DELETE /posts/1	destroy

Figure 4.4: CRUD Operations

4.3.1 Technologies Used

- django >= 1.6.0
- SQLite3
- django-rest-framework
- oauth-tool-kit
- semantic ui 2.0

4.3.2 API Testing

/api/basic_info/?access_token=<token>

```
1 {
2   "rid": "N090977",
3   "first_name": "Anesh",
4   "last_name": "Parvatha",
5   "date_of_birth": "2015-04-03",
6   "gender": "M"
7 }
```

/api/contact_info/?access_token=<token>

```
1 {
2   "mobile": "9705896317",
3   "url": "https://github.com/0xc0d3r",
4   "email": "anesh.parvatha@gmail.com"
5 }
```

/api/education/?access_token=<token>

```
1 [
2   {
3     "school": "ZPHS Sankhavaram",
4     "period": "2004-2009",
5     "degree": "SSC",
6     "stream": "ALL",
7     "grade": 91.0
8   },
9   {
10    "school": "RGUKT Nuzvid",
11    "period": "2009-2011",
12    "degree": "PUC",
13    "stream": "M. Bi.P.C",
14    "grade": 89.2
15  },
16  {
17    "school": "RGUKT Nuzvid",
18    "period": "2011-2015",
19    "degree": "B.Tech",
20    "stream": "CSE",
21    "grade": 85.0
22  }
23 ]
```

/api/skills/?access_token=<token>

```
1 [
2   {
3     "skills": [
4       {
5         "id": 1,
6         "title": "C"
7       },
8       {
9         "id": 2,
10        "title": "java"
11      }
12    ]
13  }
14 ]
```

```

11     },
12     {
13         "id": 3,
14         "title": "cloud"
15     },
16     {
17         "id": 4,
18         "title": "computing"
19     },
20     {
21         "id": 5,
22         "title": "python"
23     },
24     {
25         "id": 6,
26         "title": "mongodb"
27     }
28 ]
29 }
30 ]

```

/api/roles/?access_token=<token>

```

1 [
2   {
3     "role": {
4       "id": 2,
5       "title": "DBA",
6       "is_verified": false,
7       "permissions": [
8         {
9           "id": 4,
10          "title": "DB Delete",
11          "is_verified": true
12        },
13        {
14          "id": 5,
15          "title": "DB Edit",
16          "is_verified": true
17        }
18      ]
19    },
20    "is_verified": true
21  }
22 ]

```

4.4 Roles & Permissions

Online Portal v2.0

Notifications

Departments

</> Developers

N090247

New User Role added successfully

Edit Profile » Position & Roles

Back to Profile

#	Position & Role Name	Admin Verified	Actions
1	hod	False	Edit Delete
2	DBA	False	Edit Delete
3	HOD2	False	Edit Delete
4	hod	False	Edit Delete
5	Attender	False	Edit Delete

+ Add New

Edit Profile

Contact Info

Profile Pic

Education

Experiences

Position & Roles

Interested Areas

Technical Skills

Achievements

Summary

Figure 4.5: List of Roles of User

Online Portal v2.0

Notifications

Departments

</> Developers

N090247

Edit Profile » Position & Roles » Role » Add

Back to Profile

Role Title

HOD

Mark Role Permissions

add faculty x Add Notice x Update Notice x

+ New Position & Role

+ Add Role Permissions

Add Position & Roles

Edit Profile

Contact Info

Profile Pic

Education

Experiences

Position & Roles

Interested Areas

Technical Skills

Achievements

Summary

Figure 4.6: Adding Permissions Option

4.5 PHP Client Libraray

We developed a Client Library for PHP Applications. We used PHP-cURL to perform all the http calls and post requests to get protected data from API Server. And We developed it in a modular way with Object-Oriented approach. And all the function calls in the PHP library is self-explanatory.

4.5.1 Initializing the Client Library

```
1 <?php
2 include("Class.RIDOAuth.php");
3 $oauth=new OAuth("<ClientID>","<ClientSecret>");
4 ?>
```

4.5.2 Get Authorization URL

```
1 $url=$oauth->getAuthorizeURL("<RedirectURI>");
```

4.5.3 Get Access Token

```
1 $token=$oauth->getAccessToken("<AuthorizationCode>","<RedirectURI>");
```

4.5.4 Initializing API with Access Token

```
1 $api=new API("<Access Token>");
```

4.5.5 Getting User Info from API

```
1 $user=$api->get("<API Endpoint>");
```

Chapter 5

Network Single Sign-On

5.1 Introduction

Single sign-on (SSO) is a session/user authentication process that permits a user to enter one name and password in order to access multiple applications. The process authenticates the user for all the applications they have been given rights to and eliminates further prompts when they switch applications during a particular session.

Components Used:

- LDAP Server
- phpLdapAdmin
- LDAP Client
- HAProxy
- GlusterFS
- XtremFS

5.2 LDAP Server

LDAP, or Lightweight Directory Access Protocol, is a protocol for managing related information from a centralized location through the use of a file and directory hierarchy. It functions in a similar way to a relational database in certain ways, and can be used to organize and store any kind of information. LDAP is commonly used for centralized authentication.

5.2.1 Installation and Configuration

The OpenLDAP server is in Ubuntu's default repositories under the package "slapd". We have to install some additional utilities in order to use it in full pledged way.

- `sudo apt-get update`
- `sudo apt-get install slapd ldap-utils`

After the installation is complete, we actually need to reconfigure the LDAP package by the following

- `sudo dpkg-reconfigure slapd`

By following below steps we have to configure the LDAP

- Omit OpenLDAP server configuration? **No**
- DNS domain name? **reboot.org**
- Organization name? **reboot**
- Administrator password? **Password**
- Database backend to use? **HDB**
- Remove the database when slapd is purged? **No**
- Move old database? **Yes**
- Allow LDAPv2 protocol? **No**

5.3 phpLDAPAdmin

Its a web-based LDAP client. It provides easy, anywhere-accessible, multi-language administration for LDAP server. By this configuration and monitor of LDAP Server will be done in an easy way.

Its hierarchical tree-viewer and advanced search functionality make it intuitive to browse and administer your LDAP directory. Since it is a web application, this LDAP browser works on many platforms, making your LDAP server easily manageable from any location.

5.3.1 Installation and Configuration

- `sudo apt-get install phpldapadmin`

After the installation is complete configuration will be done by making following changes in the config.php file of phpLDAPAdmin.

- `sudo nano /etc/phpldapadmin/config.php`

```
1 $servers->setValue( 'server', 'host', '10.4.34.47' );
2 $servers->setValue( 'server', 'base', array( 'dc=reboot, dc=org' ) );
3 $servers->setValue( 'login', 'bind_id', 'cn=admin, dc=reboot, dc=org' );
4 $config->custom->appearance[ 'hide_template_warning' ] = true;
```

Listing 5.1: PHP Config file

5.3.2 Web Interface of phpLDAPAdmin:

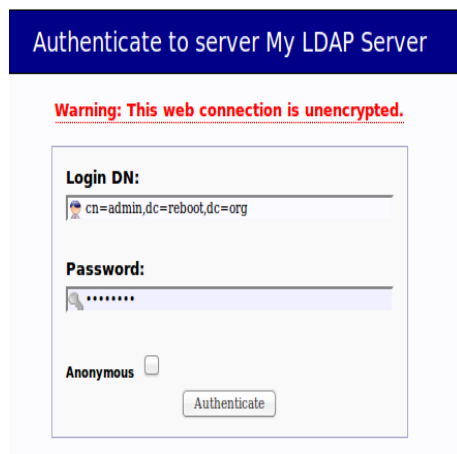


Figure 5.1: phpLDAP

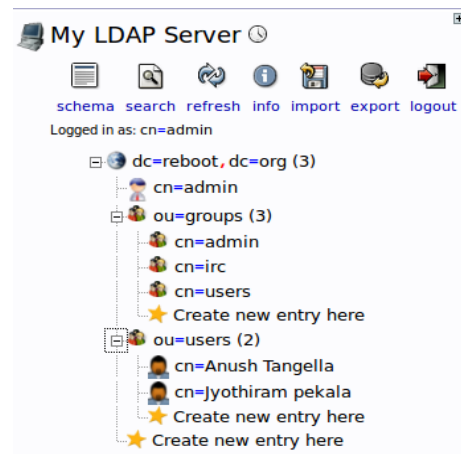


Figure 5.2: Complete category of LdapServer

cn required, rdn

Anush Tangella
(add value)
(rename)

gidNumber required

500
admin ()

givenName

Anush
(add value)

homeDirectory required

/home/users/atangella

loginShell

/bin/sh

objectClass required

inetOrgPerson (structural)
posixAccount
top
(add value)

Password alias

•••••••••••••••• md5
Check password...
(add value)

sn required

Tangella
(add value)

uidNumber required

1000

User Name alias, required

atangella
(add value)

Update Object

Figure 5.3: User Creation

New Posix Group (Step 1 of 1)

GID Number alias, required, hint, ro

500

Group alias, required, rdn

admin *

Users alias, hint

Create Object

Figure 5.4: Group Creation

cn required, rdn

admin *

[\(add value\)](#)
[\(rename\)](#)

gidNumber required

500

memberUid

Anush Tangella

[\(add value\)](#)
[\(modify group members\)](#)

objectClass required

posixGroup (structural)

top

[\(add value\)](#)

[Update Object](#)

Figure 5.5: Adding User to Group

ou=groups,dc=reboot,dc=org	
Entries found: 3 (0.01 seconds)	
export results Format: list table Base DN: ou=groups,dc=reboot,dc=org Filter performed: objectClass=*	
cn=admin	dn cn=admin,ou=groups,dc=reboot,dc=org cn admin gidNumber 500 memberUid Anush Tangella objectClass posixGroup top
cn=irc	dn cn=irc,ou=groups,dc=reboot,dc=org cn irc gidNumber 501 objectClass posixGroup top
cn=users	dn cn=users,ou=groups,dc=reboot,dc=org cn users gidNumber 502 objectClass posixGroup top

Figure 5.6: Groups information

5.4 LDAP Client:

LDAP, or Lightweight Directory Access Protocol, is one way of keeping authentication information in a single centralized location. We need another droplet to act as the client machine.

5.4.1 Installation and Configuration:

On the client machine, we need to install a few packages to make authentication function correctly with an LDAP server.

- `sudo apt-get install libpam-ldap nscd`

By following these below steps we need to configure the LDAP Client

- LDAP server Uniform Resource Identifier: `ldap://10.4.34.47/` from “`ldapi:///`”

Distinguished name of the search base: This should match our values in LDAP server’s `/etc/phpldapadmin/config.php` file.

- We have to replace “ ‘server’, ‘base’, array ” within the file to “`dc=reboot,dc=org`”
- LDAP version to use: **3**
- Make local root Database admin: **Yes**
- Does the LDAP database require login? **No**
- LDAP account for root:
 - This should also match with our values in your `/etc/phpldapadmin/config.php`
 - Search for: “ ‘ login ’ , ‘ bind_id ’ ” within the file
 - Our example was “`cn=admin,dc=reboot,dc=org`”

LDAP root account password: **Our-LDAP-root-password**

If made a mistake and need to change a value, we can go through the menu again by issuing this command:

- `sudo dpkg-reconfigure ldap-auth-config`

To configure client we adjust a few files that they can look to our LDAP server for authentication information. First, we have to edit the `/etc/nsswitch.conf` file. This will allow us to specify that the LDAP credentials should be modified when users issue authentication change commands

- `sudo nano /etc/nsswitch.conf`

The three lines we are interested in are the "passwd", "group", and "shadow" definitions. Modify them to look like this:

```
1 passwd : files ldap
2 group : files ldap
3 shadow : files ldap
```

Listing 5.2: Config file

We have to add the values to our PAM configuration.

PAM, or Pluggable Authentication Modules, is a system that connects applications that can provide authentication to applications that require authentication. When we installed and configured our LDAP PAM module, most of the needed information was added to the configuration files and we need to edit below file.

- `sudo nano /etc/pam.d/common-session`
- `sudo nano /etc/pam.d/login`
- `sudo nano /etc/pam.d/lightdm`

We have to add the below piece of code to each of the above PAM configuration files

- session required **pam_mkhomedir.so skel=/etc/skel umask=0022x**

The above will create a home directory on the client machine when an LDAP user logs in who does not have a home directory. We have to restart a service for these changes to be implemented:

- `sudo /etc/init.d/nscd restart`

5.4.2 Log In as an LDAP User:

We have now configured our client machine enough to be able to log in as one of our LDAP users. This user does not have to exist on the client machine. In order to connect to LDAP Client, we have to ssh into that particular machine.

- `ssh atangella@10.4.34.45`

5.5 HAProxy

HAProxy(High Availability Proxy) is an open source load balancer which can load balance any TCP service. It is particularly suited for HTTP load balancing as it supports session persistence and layer 7 processing. HAProxy can be configured as a front-end to load balance two VPS through private network connectivity.

5.5.1 Installing HAProxy

```
# to install haproxy
sudo apt-get install haproxy
# to get started by init script
edit /etc/default/haproxy, set ENABLED option to 1
# using haproxy
sudo /etc/init.d/haproxy start—stop—reload—restart—status
```

5.5.2 Configuring HAProxy

```
edit gedit /etc/haproxy/haproxy.cfg
```

```
frontend sunny
bind 10.4.34.250:8080
default_backend sunny-backend
backend sunny-backend
balance roundrobin
mode tcp
server sunny 10.4.34.250:80 check
server ram 10.4.34.242:80 check
server knc 10.4.34.245:80 check
```

Requests come to frontend PC will be send to any one of the backend PC's based on the algorithm. Here algorithm can be roundrobin, leastconn.

5.6 GlusterFS

GlusterFS is a unified, poly-protocol, scale-out file system serving many peta bytes of data. While many databases and other software allows you to spread data out in the context of a single application, other systems can operate on the file system level to ensure that data is copied to another location whenever it is written to disk. A clustered storage solution like GlusterFS provides this exact functionality.

We are writing by considering three systems. we are considering the two of our machines as cluster members and the third as a client. We will be configuring the computers we labeled as gluster0 and gluster1 as the cluster components. We will use gluster2 as the client.

5.6.1 Configure DNS solution

Go to **sudo nano /etc/hosts** and add below lines

```
#first_ip gluster0.droplet.com gluster0
#second_ip gluster1.droplet.com gluster1
#third_ip gluster2.droplet.com gluster2
```

5.6.2 Install server components

On our cluster member machines (gluster0 and gluster1), we can install the GlusterFS server package

```
sudo apt-get install glusterfs-server
```

#On one of the hosts, we need to peer with the second host.

```
sudo gluster peer probe gluster1.droplet.com
```

5.6.3 Create a storage volume

#Creating and enabling replication property for volume1

```
sudo gluster volume create volume1 replica 2 transport tcp gluster0.droplet.com:/gluster-storage gluster1.droplet.com:/gluster-storage force
```

#And we can activate the storage by command

```
sudo gluster volume start volume1
```

5.6.4 Install and configure client components

#On our client machine, we can install the GlusterFS client package

```
sudo apt-get install glusterfs-client
```

#for mounting our remote storage volume on our client computer

```
sudo mkdir /storage-pool
```

```
sudo mount -t glusterfs gluster0.droplet.com:/volume1 /storage-pool
```


5.6.5 Restrict access to the volume

```
#for restriction of storage volume for clients
sudo gluster volume set volume1 auth.allow *
#for removing restrictions
sudo gluster volume set volume1 auth.allow gluster_client1_ip,gluster_client2_ip
```

At this point, you should have a redundant storage system that will allow us to write to two separate servers simultaneously. This can be useful for a great number of applications and can ensure that our data is available even when one server goes down. But GlusterFS is failing in distributed environment at some situations. For that we moved an efficient one XtremFS, which works very well in distributed environment and tackles all errors.

5.7 XtreamFS

XtreamFS is a fault-tolerant distributed file system for all storage needs. It is simple to setup as it does not use any kernel modules. It's easy to integrate with clients for Linux and Windows. XtreamFS is also a parallel, object-based file system. You can stripe your files across many storage servers for high-performance parallel access. The stripe width can be configured per file.

It can stripe files within your cluster and can replicate your data across clusters. This allows you to have high-performance access within your cluster and to share your data in your virtual organization.

XtreamFS' replication is fault-tolerant. A broken hard drive or unhealthy storage server does not result in data loss or even degraded service quality.

This matters if you have 10 or 1000s of machines, as your jobs finish without interruptions and you can delay repairs to whenever you have time.

5.7.1 Installation, Creating & Mounting Filesystem

- Added the XtreamFS repo to our system and then installed the packages `xtreamfs-server` and `xtreamfs-client`.
- Starting the Directory Service: `$sudo /etc/init.d/xtreamfs-dir start`
- Starting the Metadata Server: `$sudo /etc/init.d/xtreamfs-mrc start`
- Starting OSD: `$sudo /etc/init.d/xtreamfs-osd start`
- Loading the FUSE kernel module: `$sudo modprobe fuse`
- We can check the registry by opening the DIR status page in our web browser at `http://localhost:30638`
- Creating a new volume with default settings : `$sudo mkfs.xtreamfs localhost/myVolume`
- Creating mounting point : `$sudo mkdir /xtreamfs`
- Mounting XtreamFS: `$sudo mount.xtreamfs localhost/myVolume /xtreamfs`
- We can unmount using : `$sudo umount.xtreamfs /xtreamfs`

5.7.2 Distributed Step

We assume a setup with two machines:

- One system running the directory service (DIR), metadata server (MRC), storage server (OSD).
- Another system running the only storage server (OSD).

First, installed the binary packages using repositories made available by the XtreamFS team. Once the repository file is registered, we can install these packages:

```
sudo apt-get install xtreamfs-client xtreamfs-server xtreamfs-tools
```

Now suppose one server to host the DIR and MRC service is called `osd1`, and my OSDs will be running on `osd1` and `osd2`. On `osd2`, we have to edit the following config files under `/etc/xos/xtreamfs` and set **`dir_service.host = kdc01`**. For the `mrc` and `osd` config files, we have to set up **`hostname = kdc01`**

- `mrcconfig.properties`
- `osdconfig.properties`

On `WKS01` where a second OSD will be running, edited the `osd` config file, point **`dir_service.host = kdc01`** and **`hostname = wks01`**

- `osdconfig.properties`

Then we have to start the services. On `KDC01`, the following services are started:

```
sudo /etc/init.d/xtreamfs-dir start
sudo /etc/init.d/xtreamfs-mrc start
sudo /etc/init.d/xtreamfs-osd start
```

On `WKS01`, OSD service

```
sudo /etc/init.d/xtreamfs-osd start
```

Then we created the XtreamFS filesystem using following commands:

```
sudo mkfs.xtreamfs kdc01/myVol # on KDC01
sudo mount.xtreamfs kdc01/myVol /data # on KDC01 and WKS01
```

By default, 1 replica will be stored on the entire XtreamFS volume. XtreamFS client is smart enough to contact the best OSD for the file resource. In case its desirable to maintain more than 1 replica of the same file, it can be done with the `xtfsutil` tool. For example, to configure XtreamFS to replicate a file to all available OSDs:

Chapter 6

Private Infrastructure Cloud

To support this central identity both the network and web network central identity we want to go for the private cloud deployment it includes creating the Private Infrastructure Cloud with openstack and creating Virtual Machines for installing these services and assign them the IP address.

6.1 Openstack Architecture

Openstack is a cloud operating system that provides the 3 main services for the Infrastructure clouds namely Storage, Compute, Networking and some other components are can be added later as addons

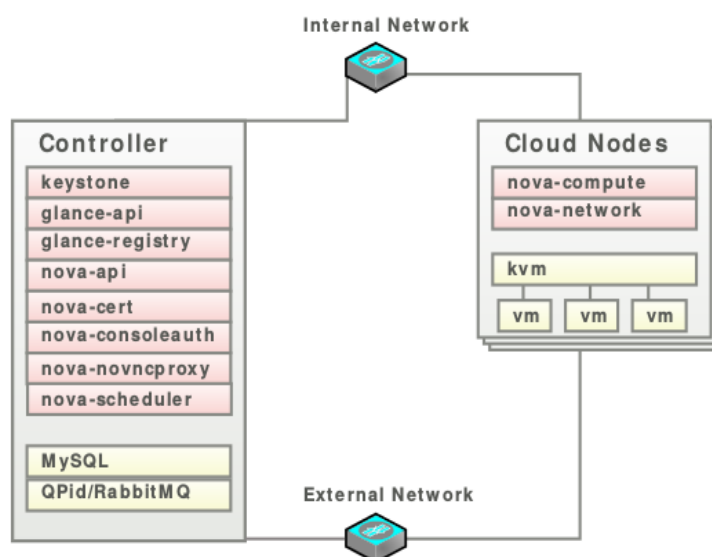


Figure 6.1: Openstack Architecture.

6.2 Installation

Installing openstack includes component wise installation namely NTP, MySQL, Rabbitmq-Server, Keystone, Nova, Cinder, Glance, Neutron

6.2.1 NTP

```
# apt-get install ntp
```

6.2.2 MySQL

```
# apt-get install mysql-server
```

6.2.3 Rabbitmq-server

```
# apt-get install rabbitmq-server
```

6.2.4 Keystone

```
# apt-get install keystone
```

6.2.5 Glance

```
# apt-get install glance python-glanceclient
```

6.2.6 Nova

```
# apt-get install nova-api nova-cert nova-conductor nova-consoleauth nova-novncproxy  
nova-scheduler python-novaclient
```

6.2.7 Neutron

```
# apt-get install neutron-server neutron-plugin-ml2
```

6.3 Virtual Machines

This Virtual Machines are created from the resource pool after successful installation openstack

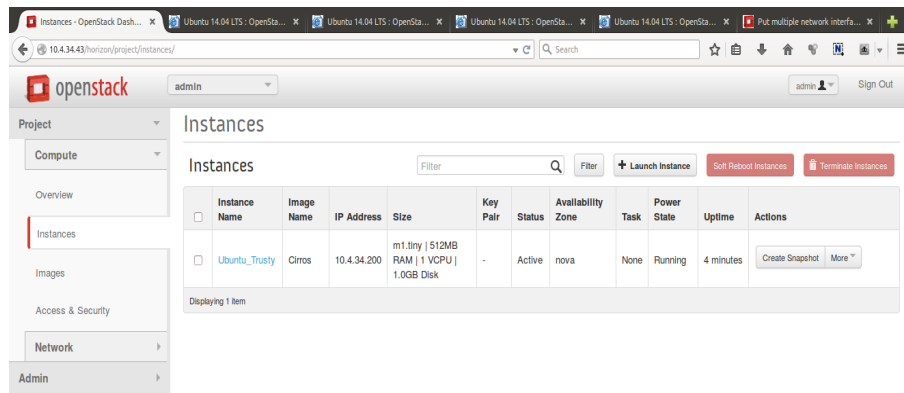


Figure 6.2: Openstack Virtual Machines.

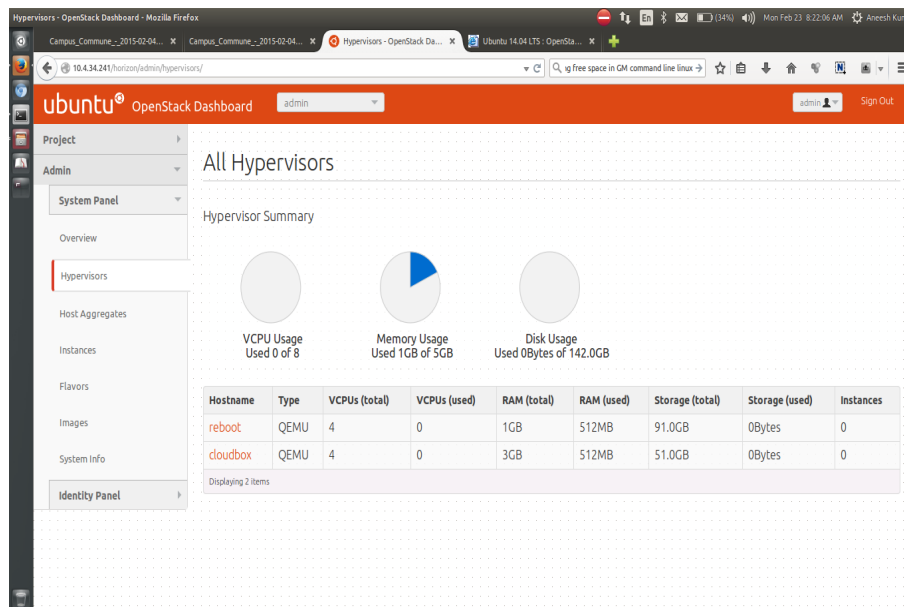


Figure 6.3: Openstack Resource Pool.

Chapter 7

Conclusion & Future Work

7.1 Conclusion

We tried GlusterFS for replication among systems, but its not working if any one of the system fails. Then we found that XtreamFS works well in distributed system and provides fault tolerant solution.

We developed network based sign on using LDAP and web based single sign on along with REST API using Oauth 2.0 and Django. We tried to create private cloud using openstack but lot of errors came because of proxy based internet and low configured PCs.

7.2 Future Work

We would like to combine Network single sign-on with Web based single sign on along with XtreamFS and HAProxy. Creating virtual machines and Private cloud is not possible with the available systems. But if we could provide systems with enough configuration, sure we can create better sophisticated solution

Chapter 8

References