Cloud based IT Infra with Central Identity

Final Presentation

Project Guide

T. Chandra Shekar

Lecturer – Dept. of CSE

Presenting by

Team r3b00+

Dept. of CSE, RGUKT – Nuzvid

April 29, 2015

About us

We are from team *r3b00+* {reboot}

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

Outline

Objective

Our objective is to create a private cloud and availing access of all its services using central identity with single sign on through dynamic role based management along with REST API to third party for application developers and users.

This can be developed by using open source tools like OpenStack, NFS, LDAP, Ubuntu and etc

Expecting to serve with virtual machines to the research, virtual labs rather than dedicated lab hardware.

Outline

Motivation

- No Central Identity, Central Storage & High capacity hardware resource pool.
- Failed to maintain large user load web services like ONB, Exam servers, etc.
- Dedicated computer course labs like Matlab, VLSI, etc.
- No proper Web Application Security & Standards.
- Inadequate resource requirements for Research.

Outline

Users & IT Services

We are collaborating all IT Services that are required for University and identifing the users who is going to use them. All Users are catagorized into 4 groups [1]

- Studens, Developers, Staff, faculty & Researches

Figure : Simplified structure of the main users of IT services.

Cloud Infrastructures

All University IT Services are deployed in a private cloud, constructed over exsiting infrastructure, that can be broadly viewed as

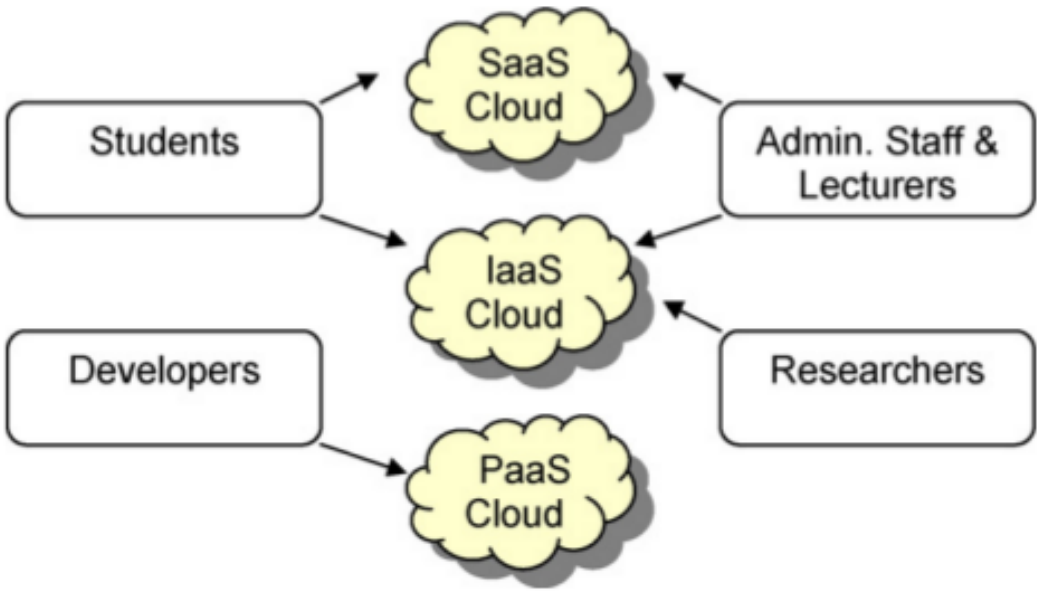

Figure : IT Services and Users in Cloud Computing

Outline

Phase I Review I

- Central Identity
  - Single Sign-On with REST API
  - Fedarated Identity Management
  - Dynamic Role Based Access Control
  - Network Based Central Identity
    - LDAP Servers
    - NFS Servers

Phase I Review II

- Cloud Computing
  - Cloud Characterstics
  - Service Models
  - Deployment Models
  - Private Clouds
    - Introduction
    - Open Source Tools

Project Timeline

Module	**Duration**
Literature Survey	Sep - Dec 2014
Central Identiy - Profile, Resourses, Login	Jan - Feb 2015
Central Identiy - Admin Portal	Jan - Feb 2015
Network Signle Sign on - Gluster FS	Jan - Feb 2015
Central Identiy - OAuth API	Feb - Mar 2015
Load Balancing - HAProxy & DOS Attacks	Feb - Mar 2015
Network Signle Sign on - LDAP	Feb - Mar 2015
Network Signle Sign on - NFS	Feb - Apr 2015
Network Signle Sign on - Xtream FS	Feb - Apr 2015
Central Identiy - Client Library	Mar - Apr 2015

Outline

Demo

How well we implemented OAuth Provider?

- To implement OAuth provider we used python-django and oauth-tool-kit
- When user requests the protected resource, oauth-tool-kit will generate client_id and client_secret
- By using those two things user will get access_token to access protected resource

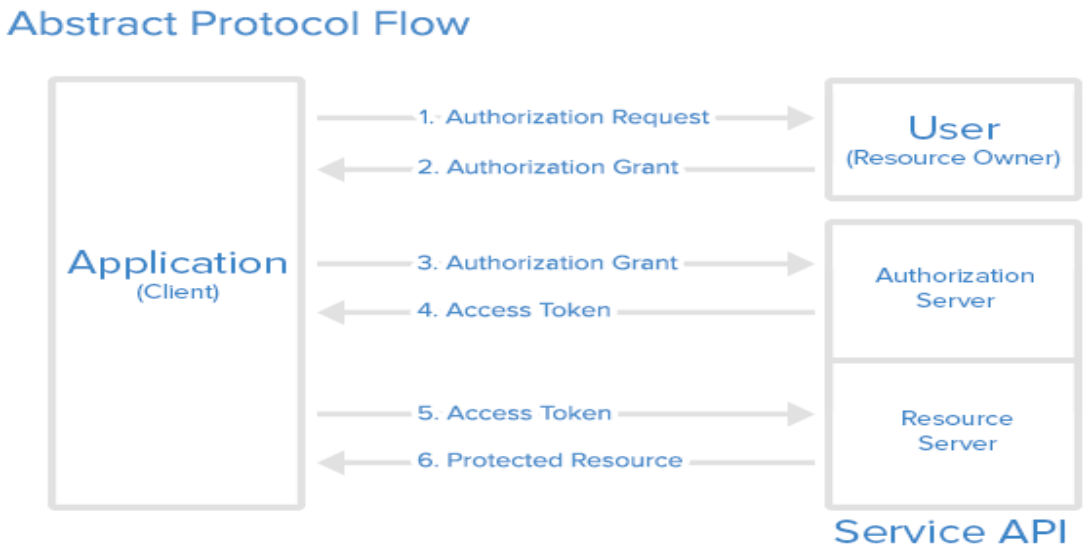

Figure : OAuth Protocol Work Flow Diagram

REST API

- REST stands for **RE**presentational **S**tate **T**ransfer
- A Collection of simple URIs, and HTTP calls to those URIs and some JSON resources
- We implemented REST API by using django-restframework

`/api/contact_info/?access_token=<token>`

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

PHP Client Library

- 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

PHP Client Library

Initializing the Client Library

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

Get Authorization URL

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

Get Access Token

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

Initializing API with Access Token

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

Getting User Info from API

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

Outline

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
- NFS Server
- NFS Client

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.
- LDAP is commonly used for centralized authentication.

phpLDAPadmin

- Its a web-based LDAP client which provides easy, anywhere-accessible, multi-language administration for LDAP server.
- Since it is a web application, this LDAP browser works on many platforms, making your LDAP server easily manageable from any location.

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

```
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 1: PHP Config file

LDAP Client I

- LDAP-Clinet is a another droplet to act as the client machine.
- 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 2: Config file

LDAP Client II

- PAM(Pluggable Authentication Modules), is a system that connects applications that can provide authentication to applications that require authentication.
- session required **pam_mkhomedir.so skel=/etc/skel umask=0022x**
- We have to add above piece of code to these files **common-session**, **login**, **lightdm** in /etc/pam.d/ directory
- In order to connect to LDAP Client, we have to ssh into that particular machine.
  - ssh atangella@10.4.34.45

NFS Server

Installation

apt-get install nfs-kernel-server

mkdir -p /var/nfs & mkdir -p /var/nfs-share

Edit /etc/exports

```
/home                   10.4.34.202(rw,sync,no_root_squash,
    no_subtree_check)
/var/nfs                10.4.34.203(rw,sync,no_subtree_check)
/var/nfs-share          *(ro,sync,root_squash,no_subtree_check)

# here the  ro - read only | rw - read and write
# ip and * means allowed hosts
```

Listing 3: /etc/exports

Exporting direcories & Restart Server

exportfs -a & # /etc/init.d/nfsserver restart

NFS Server

Installation

apt-get install nfs-client

Mounting NFS Shares

mount 10.4.34.201:/var/nfs-share /mnt

Outline

Introduction

- Maintaining fault-tolerant file systems in distributed environment is always challenging
- We can achieve it through replication of data among systems
- But adding a load balancing on distributed systems improves response time
- GlusterFS provides clustered storage solution when all server systems available
- XtreemFS along with HAProxy helps us to achieve the goal

HAProxy

- HAProxy(High Availability Proxy) is an open source Reliable, High Performance TCP/HTTP Load Balancer
- HAProxy can be configured as a front-end to load balance two VPS through private network connectivity.
- Installing the HAProxy – # apt-get install haproxy
- Configuring HAProxy

```
1  frontend sunny
2   bind 10.4.34.250:8080
3   default_backend sunny-backend
4    backend sunny-backend
5   balance roundrobin
6   mode tcp
7   server sunny 10.4.34.250:80 check
8   server ram 10.4.34.242:80 check
9   server knc 10.4.34.245:80 check
10 /etc/init.d/haproxy {start|stop|restart|status}
```

Load Balancing

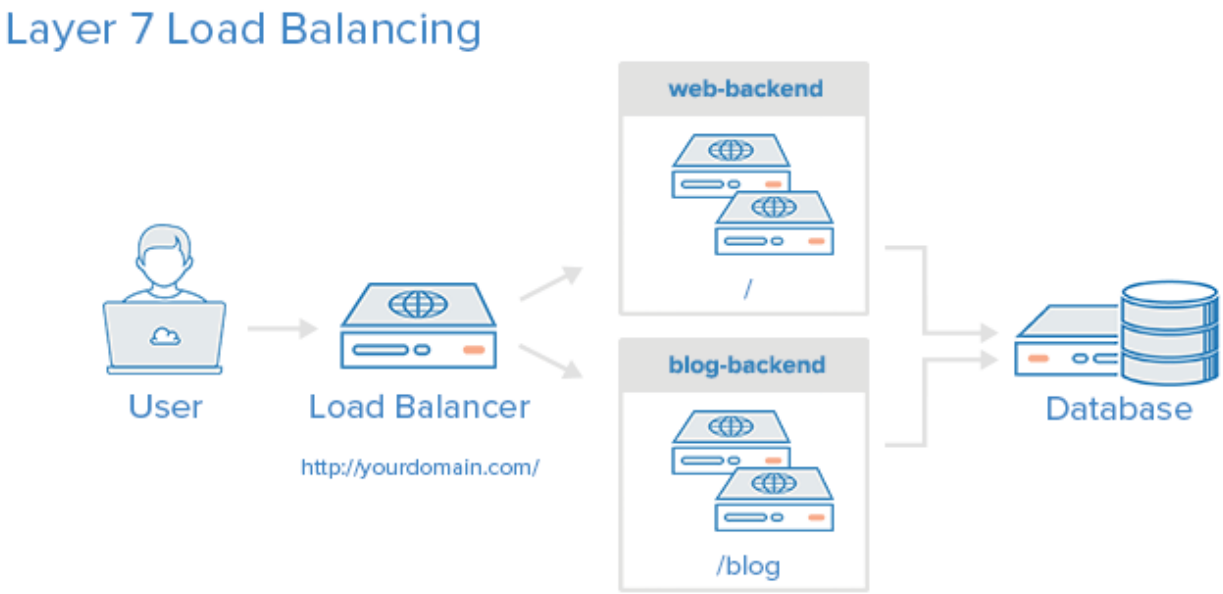

Figure : Load Balancing

GlusterFS

- GlusterFS is a clustered storage solution allows you to spread data 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
- Steps to be followed:
  - Configure DNS solution
  - Install server components
  - Create a storage volume
  - Install and configure client components
  - Restrict access to the volume
- This fails in a situation where all systems are available

XtreemFS

- Its a fault-tolerant distributed file system avails high-performance parallel access
- **Features:**
  - File Replication
  - Elasticity & Scalability
  - Cloud Storage
  - Asynchronous MRC Backup
  - Security
  - Stripping
- **Packages required:** xtreemfs-server, xtreemfs-client and xtreemfs-utils.

- We can add replica properties and permissions to the files using xtfutils command.

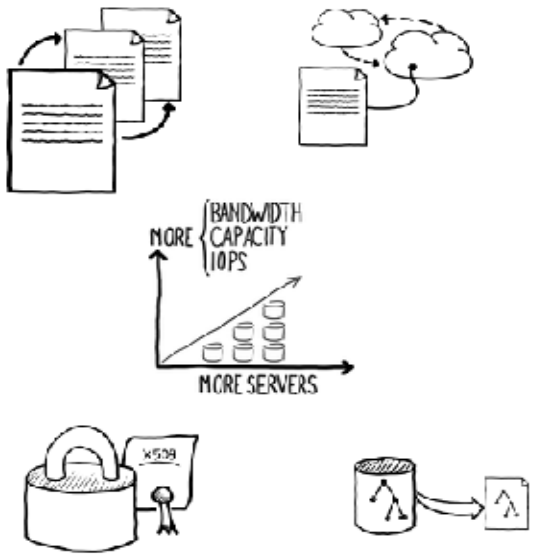

Figure : XtreemFS Features

XtreemFS Cont.

```
root@sunny-SVE1513BYNB:/# cd datapoint/
root@sunny-SVE1513BYNB:/datapoint# echo "hello" > hello.txt
root@sunny-SVE1513BYNB:/datapoint# xtfsutil -r WqRq hello.txt
Changed replication policy to: WqRq
root@sunny-SVE1513BYNB:/datapoint# xtfsutil -a auto hello.txt
Added new replica on OSD: 282779e9-c1eb-414c-851e-440734d67f5d
root@sunny-SVE1513BYNB:/datapoint# xtfsutil hello.txt
Path (on volume)     /hello.txt
XtreemFS file Id     ad9fdd23-66ae-480a-86f1-e07d680bbc33:6
XtreemFS URL         pbrpc://osd1:32638/Data/hello.txt
Owner                root
Group                root
Type                 file
Replication policy   WqRq
XLoc version         2
Replicas:
  Replica 1
     Striping policy     STRIPING_POLICY_RAID0 / 1 / 128kB
     OSD 1               7f0e8a09-de67-4be8-9a68-a878eec28bb2 (osd1:32640)
  Replica 2
     Striping policy     STRIPING_POLICY_RAID0 / 1 / 128kB
     OSD 1               282779e9-c1eb-414c-851e-440734d67f5d (osd2:32640)
```

Figure : XtreemFS Distrirbuted & Replicated Step

Outline

Additional Work

- Openstack Installation
- Building Private Cloud
- GlusterFS Replication
- DOS Attacks on deployed Application

References I

- Django Docs – https://docs.djangoproject.com/en/1.7/
- Django – https://djangoproject.com/
- Django OAuth Tool Kit – https://github.com/evonove/django-oauth-toolkit
- Django REST Framework – http://www.django-rest-framework.org/
- OAuth 2.0 – http://oauth.net/2/
- Semantic UI – http://beta.semantic-ui.com/
- GlusterFS – https://www.digitalocean.com/HowToCreateaRedundantStoragePoolUsingGlusterFSonUbuntuServers_DigitalOcean.htm
- HAProxy – www.digitalocean.com/HowToUseHAProxytoSetUpHTTPLoadBalancingonanUbuntuVPS_DigitalOcean.htm
- LDAP – https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-a-basic-ldap-server-on-an-ubuntu-12-0

References II

- NFS Server – http://www.server-world.info/en/note?os=Ubuntu_14.04&p=nfs
- NFS Client – http://www.server-world.info/en/note?os=Ubuntu_14.04&p=nfs&f=2
- Openstack – http://www.server-world.info/en/note?os=Ubuntu_14.04&p=openstack_icehouse
- XtreemFS - https://blog.headdesk.me/DistributedfilesystemwithXtreemFS_xpk'sblog.htm

End

Thank you and Any Queries ?