Assignment Repository Using OpenStack

Sanket Dhami
Computer Engineering
San Jose State University
San Jose, USA
sanket.dhami@sjsu.edu

Anuj Sanghvi
Electrical Engineering
San Jose State University
San Jose, USA
anuj.sanghvi@sjsu.edu

Akash Patel
Electrical Engineering
San Jose State University
San Jose, USA
akash.patel@sjsu.edu

***Abstract*— The main goal of our project is to efficiently manage the assignment repository using the features provided by OpenStack. OpenStack is an open source platform that helps us manage compute, storage and networking resources which can be simply done through a dashboard that provides administrator level access. We will be using AWS EC2(Amazon Elastic Compute Cloud) service which will help us to create servers as per our need and configure the networking, security and storage aspects of it. Our project aims at providing a centralized managing capability using one of the components of OpenStack called Swift which is a highly available and consistent object store.**

Keywords*— *OpenStack, Swift, Object, Container, Proxy, AWS-EC2, REST

I. Introduction

The software-defined storage market is expected to grow from 4.72 billion in 2016 to 22.56 billion by 2021 at a Compound Annual Growth Rate (CAGR) of 36.7% [1]. That shows us that there is an exponential rise for the software-defined storage. The traditional proprietary storage systems are migrating towards software-defined storage systems which are much more efficient and cost effective. OpenStack provides enterprises a platform that controls large pools of compute, storage and networking resources throughout a datacenter, which are managed through a centralized console which gives administrative level control [3].

Our project is based on OpenStack where we are building an assignment repository which can be managed centrally through the features provided by OpenStack Swift. Most of the education systems around the world have shifted to digital/virtual education and this digitalization is increasing every day. The traditional education systems where books and assignments were pen-paper based have been replaced with soft copies where students submit the assignment files online. And these assignments are then graded by the professor and the grades for the same are posted online. As we know the assignments or the work submitted by the students can contain Microsoft word files, pdfs, videos ranging from some megabytes to few terabytes. The data is unstructured and the files can be of various sizes. And storing them with the traditional approach would consume lots of storage and networking resources. Therefore, to address this problem we have implemented an assignment repository system using one of the components of OpenStack which is Swift. Swift is a highly available and consistent object store. Swift provides a cloud storage software that can be used to store and retrieve the data with simple API's. Assignment files can be considered as unstructured data which can be of different sizes. Swift is ideal for this scenario to store the data which can grow without any bounds.

II. Background

A. OpenStack

OpenStack is a free open-source cloud operating system that controls large pools of compute, storage and networking resources throughout a datacenter, which are managed through a centralized dashboard which gives administrative level control [4]. It lets users/developers deploy virtual machines and instances that handle different functions for managing the cloud environment on the go. Users can manage the instances through a web-based dashboard, through command-line tools or RESTful API's. OpenStack makes horizontal scaling easy i.e. scaling-out which means that the tasks that benefit from running parallel can easily serve more users by running more instances. It divides the loads easily and helps manage it efficiently. Since OpenStack is open-source developers around the world can develop a robust and secure.

B. OpenStack-Swift

There are several components of OpenStack which Nova, Swift, Cinder, Neutron, Horizon, Keystone, Glance, Ceilometer etc. Swift uses a distributed architecture with no central point of control, providing greater scalability, redundancy and high performance. Enterprises can use swift to store lots of data

efficiently, safely and cheaply. Data can be stored in clusters of servers to store petabytes of data. We can create, modify and get objects and their metadata by using the REST API's (Representational State Transfer). OpenStack is responsible for ensuring data replication and integrity across the cluster. All the objects stored are replicated three times is as unique as possible zones. The storage nodes can be scaled horizontally by adding new nodes. If in case a node fails, the data on that storage node is replicated to another node. We use language specific API's i.e. RESTful HTTP API, to interact with object storage. Using REST API makes it easier to integrate into our application.

Object storage consists of account, containers and objects. Storage accounts API are used to perform account-level operations like listing containers for an account. Storage containers API are used to perform container-level operations like listing objects in the specified container, also creating, deleting, updating the containers. Storage objects API include performing object-level tasks like creating, replacing and viewing details of the object [3]. The Proxy server sums up the swift architecture. For all the requests made, swift looks up the location of the account, container or the object in the ring and direct the request accordingly. Proxy server is also responsible for handling failures. The object server is the place where actual data is stored. The data can be retrieved using REST APIs. All the file names of the objects stored are stored in form of a hash.

III. System Architecture

The system architecture for the project Assignment Repository using OpenStack is shown in Figure 1.

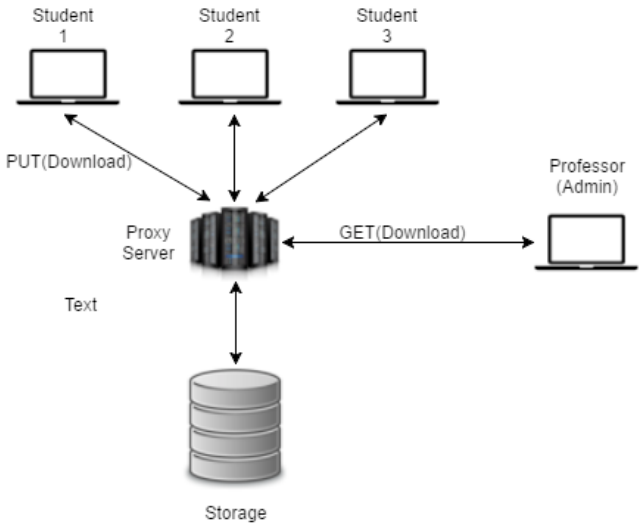

Figure 1: System Architecture

A. Setup

Main goal of our project is to implement assignment repository system using OpenStack Swift. For this we have created total of six instances on Amazon EC2 of which three are the instances that represent student and one instance represents admin i.e. professor. One instance is proxy server and another is object server where the assignment files are stored. All the files uploaded and downloaded go through the proxy server. We use REST APIs GET to download and PUT to upload files. Along with that we have used several other curl commands. The admin node has all the privileges i.e. it can access the files in all the containers of all the students. The three student nodes can access files in their own container only, they cannot access the files from the other student containers.

B. Tools

Our project is setup on Amazon EC2 where we have created six instances, three for the student nodes, one for professor (admin), storage node where actual files (objects) are stored and the proxy node. Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It helps to scale up and scale down easily within short time. And is highly reliable and flexible.

IV. Project Implemtation

Before starting the actual implementation, proxy configuration file was edited which includes certain student accounts having admin privileges and one professor account which has reseller admin privileges. In swift, a user with admin privileges can create, delete modify objects in its container. Whereas, a user with reseller admin privileges can create, modify and delete any object from any account. Also, a reseller admin user can give admin privileges to non-admin users in swift.

The implementation of the architecture described, starts with a python script in which a professor node having the reseller admin privileges creates containers and publishes this container to all the student nodes who are enrolled in the course per say. Later, the student nodes having admin privileges of their own account can see a container in his/her account. Now through python script for student nodes, the students having a file or assignment (object) can upload to the object store node. With the same script, a student can also delete a file from the object store node such that the file won't be able to retrieve to the professor node. After the assignment deadline, the professor can download all the files that are uploaded by the student in their account. All of these upload and download happens using basic APIs like GET and PUT. Once the student node upload files and documents, the

professor node can view/list all the files uploaded by all the students and is able to download all of them. We have also added a technique by which the name of the file is appended by the name of the account user so that the professor node comes to know which student has uploaded which files and this avoids ambiguity if all students upload files with the same name. Deleting the files from each of these nodes is also an option in our script. Authentication of our http requests is done by a feature in Swift called 'TempAuth' and generates a key for every session.

FUTURE SCOPE

We have implemented assignment repository where students can upload, view and download their assignment. We are using the swift object store as the storage. The scope of this project can be extended to a great extent. Graphic user interface can be added to it which will make it look more user-friendly. Several other functionalities can be added to our project. Currently, students can just upload, download and view their own files and the admin(Professor) can see everyone's submission. Further, posting of grades and adding a calendar to it with the functionality of deadlines can be added. Also, many other swift middleware like object versioning, rate-limit, Recon-used for monitoring and returns various system metrics. Server-side copy feature which enables users to copy objects between accounts and containers without the need to download and re-upload the objects eliminating bandwidth usage and saving time. Keystone authentication can be added so that the users can be authenticated for a specific type of service.

RELATED WORK

The closest related work to what we have implemented in our project is Canvas by instructure which is a learning management system built using Ruby on Rails as the web application framework backed by a PostgreSQL database. It consists of JQUERY, HTML5 and CSS3 which provides a modern graphic interface. Canvas operates as a software as a service using AWS [2]. We have implemented a similar application with more focus on the storage part which addresses the major problem of consistently increasing unstructured data.

CONCLUSION

Our project is an effective way of exploring the features of Swift and enhancing them by our application. Swift with all its features proves very efficient for storing inconsistent data and as we saw in out command line interface based interactive application that how Swift makes it easy to upload, download and manage files stored as objects. Swift gives a boost in performance which can be measured by the upload and download speeds, latency and bandwidth utilization. A major part of our approach was to emphasis on building metrics to measure the mentioned performance parameters.

ACKNOWLEDGMENT

The authors would like to thank professor Younghee Park and Aditya Fotedar, CIO Nexenta for their teachings and discussions in the class which made this project possible.

REFERENCES

[1] "Market Analysis-Software-Defined Storage". *Marketsandmarkets.com*. N.p., 2016. Web. 12 Dec. 2016.

[2] "Instructure". *En.wikipedia.org*. N.p., 2016. Web. 12 Dec. 2016.

[3] "Openstack Docs: Object Storage". *Docs.openstack.org*. N.p., 2016. Web. 12 Dec. 2016.

[4] "Software &Raquo; Openstack Open Source Cloud Computing Software". *OpenStack*. N.p., 2016. Web. 12 Dec. 2016.

[5] "Instructions For A Multiple Server Swift Installation (Ubuntu) — Swift 1.13.2.Dev9 Documentation". *Docs.openstack.org*. N.p., 2016. Web. 8 Dec. 2016.