I J C T A, 9(11) 2016, pp. 5383-5390
© International Science Press

MAILCHAMP – A SIMPLE MAIL SERVER FOR UNIVERSITIES AND COLLEGES

Asif Riyaz Ahmed[*]**, Saurabh Gaumat**[**] **, Dhruv Garg**[***] **, and Jagdish Chandra Patni**[****]

Abstract: Nowadays, email marketing campaigns are becoming a common occurrence. And for doing it with software for desktops or mail clients and servers, it's crucial to have access to a highly available, scalable as well as reliable cloud mail server i.e. a mail delivery service that will provide the optimum possible successful rate of delivery for all mails sent. Regrettably, many marketing professionals even now depend upon regular mail servers like the ones used by Yahoo or Gmail. Negligent of all the possible risks, they generally divide their list into little partitions to bypass these mail servers' delivery constraints but they don't perceive that the actual drawback lies just not in scalability of the service but also the success of the service. Actually, the guarantee of not only supporting unlimited mail delivery at once but also delivering with efficiency and consistency can only be ensured by a cloud email server. Such an e-mail server is developed precisely to handle this type of bulk messaging: supporting a tight link with internet service providers and appointing only white-listed IPs, it is a secure and dependable choice for transactional or marketing messages. Also, email delivery service based on a cloud platform presents an advanced tool to track and get valuable data regarding every delivered mail – their click-through rate and open rate; all spam objections, the no. of unsubscribers, etc. Actually anything that will help to improve the generic cloud emailing or newsletter campaign. So, basically we are building a cloud SMTP (Simple Mail Transfer Protocol) server by installing OpenStack IceHouse on Ubuntu 12.10 and deploying a private cloud. Thus, providing the scalable service. Then the private cloud will be used to host an SMTP server which will be configured with a mail monitoring tool/API.

Key Words: Infrastructure as a Service, Simple mail transfer protocol, Platform as a service, Software as a service, Application Program Interface, Virtualization, VMware, Openstack, Dashboard, Secure Socket Layer, Transport layer security

1. INTRODUCTION

OpenStack is an open-source and free collection of software projects for managing a cloud computing platform. It is mainly installed as an IaaS (Infrastructure as a Service) solution. The project is comprised of a set of interrelated softwares that manage pools of storage, networking and processing computing resources in a data center database/ library, able to be provisioned and managed on demand via a restful API, command-line tools, or A web-based dashboard.

1.1 Anatomy of OpenStack Cloud Project

According to the OpenStack community, OpenStack Compute is a software project to rapidly and elastically provision and operate massive clusters of private, community and virtual servers. The

[*] CIT, UPES, Dehradun Email: asif.riyaz.ahmed@gmail.com
[**] CIT, UPES, Dehradun Email: shgaumat@gmail.com
[***] CIT, UPES, Dehradun Email: dhruv.garg05@gmail.com
[****] CIT, UPES, Dehradun Email: patnijack@gmail.com

many assorted elements inside OpenStack Compute make us able to run instances, manage multiple networks of cloud deployments, and control the entry to the project via various projects and users. The virtualization solution provided by the OpenStack platform isn't its own. Alternatively, the OpenStack project provides architecture for the deployment of different virtualization procedures in a separate environment based on cloud, with fast and reliable access to the failover and scalability options unremarkably related to the cloud platform. The OpenStack provided API can even relate with various virtualization environments like Xen, KVM, QEMU, and HyperV.

OpenStack includes several options same as the Amazon net service. A project is assigned to a Virtual machine. Every user has a pair of private-public key to enter at least one or more projects/ nodes. Every node is especially provisioned a RAM, storage and processor core, and other resources. The selection of the Application Programming Interface is operated with the support of the RBAC (Role Based Access Control). The various elements of the OpenStack project provide a promisingly flexible and dynamic platform for cloud computing.

Cloud configuration related settings and information are all stored in a database library. The REST API provides the users the capability to communicate with any of the cloud components. The recent Juno update emphasizes the command line as the method for managing the API tools, but finally, the communication with the API will occur via web applications (web dashboard) or fat clients. The Auth Manager manages all the access to the API services based on role, thereby circumventing the users shutting down and starting each instance. The Scheduler relegates the computational time and access separately to each virtual component.

Not every OpenStack module is fully deployed in the Juno update. As an example, take the Object Store which as of now only maintains storing of 5GB or smaller objects, and role-based quotas are also missed. But the Juno update is definitely adequately finished for setting it up and getting started.

1.2 Components of OpenStack Framework (Required in our project)

OpenStack Dashboard (Horizon)

It is the dashboard for openstack, which is available in around 16 languages such as Serbian, German, Hindi, English etc. The interface is being upgraded with many latest navigation features and end-user experience enhancements like in-line editing.

OpenStack Compute (Nova)

Recent endorsement for rotating updates decreases the influence on continuous workloads throughout the update process. Analyzing and estimating the needs of the third-party drivers has become highly stringent, and the performance of the scheduler is much increased. Various improvements involve the better boot process reliability over various service platforms, new options available to the end-users via API upgrades such as targeted systems by affinity and more efficient gain to the data layer to better the performance, especially at scale.

OpenStack Image Service (Glance)

It enables the registration, discovery, and delivery services for the server and disk images. Stored images can be utilized as a template. The stored images can also be utilized to catalog and store an extensive number of backups. This utility stores the server and disk images in multiple back-ends, along with the OpenStack Object Storage. For querying information concerning the disk and server images, the APIs in Glance give a customary REST interface and let the users deliver the snapshots to new servers.

OpenStack Object Storage (Swift)

An important attribute is discoverability that drastically enhances the system workflows and decreases time by enabling the users to query any Object Storage cloud implementation regarding the features which are accessible via an API call. A latest replication process considerably increases the execution by launching the s-sync to more economically deliver the data.

1.3 SMTP Server

A SMTP mail server can be a computer or a system of computers that receives all the outgoing mail messages from the users and then dispatches them to their addressed recipients [6]. All the mail servers enforce a variant of the Simple Mail Transport Protocol, and much of these versions run on the UNIX operating system using the ‘sendmail’ program. Email messages will often pass through many servers to reach their destinations, and the SMTP protocol enables this.

2. LITERATURE REVIEW

In the month of July 2010, NASA and Rackspace Hosting together deployed an open-source cloud-software initiative called as OpenStack. The OpenStack project meant to assist organizations provide cloud-computing services operating on regular hardware. The society's 1st authentic release, designated as Austin, came into public 4 months afterwards, with decisions to release periodic updates of the project following each few months. The initial code was from Rackspace's Cloud Files platform as well as from NASA's Nebula platform.

It is observed that the OpenStack community has the biggest active population throughout the previous few months. And overall, it has the biggest number of users succeeded by Nebula and Eucalyptus. Bram et. al proposes a test procedure to determine the practicality and capital requirement for deployment in the cloud environment. The fundamental characteristics of the cloud mentioned are broad network access, self-service, on-demand, rapid elasticity, and resource pooling and measured service.

SMTP protocol is generally the most commonly adopted protocol for e-mail transportation [5]. Although, it lacks safety features for authentication of sending party, privacy, probity of e-mail message, nonrepudiation and consistency of e-mail container. For providing a private and secure e-mail delivery service, SMTP servers integrate many security attributes using add-on security and privacy protocols. The add-on security and privacy protocols present an acceptable security but have several drawbacks.

3. PROBLEM STATEMENT

In the case of mass e-mailing such as in an e-mail marketing campaigns, it is better to have a mail service which can scale-up for your vast needs. But the service should also be monitored to get valuable data and statistics about all the sent emails – their rate of opening and their click-through rate, the number of mail unsubscribes, all spam complaints, etc. to improve the newsletter campaign. But such services are not provided by common mail servers like the ones associated to Yahoo or Gmail as they are not scalable (limited to 500 accounts per mail) and have no such mail monitoring tool.

4. METHODOLOGY

The project involves creating our own private cloud using OpenStack and creating the SMTP server using a standard open source platform.

First we will install VMware workstation on the on a x64 architecture system and then create two virtual machines namely a Controller node and Storage node. On the Controller node we will install and configure the Keystone and the Horizon OpenStack projects [4]. Then we deploy our SMTP mail server on the Controller node. Next on the storage node we will install and configure the Swift OpenStack project. Then all the data in the mail server can be transferred to the Swift node using networking commands.

4.1 Creation of Controller Node

Before we installed and configured the Compute service, we created a SQL database (MariaDB), service admin credentials, and API endpoints [1]. Afterwards we installed the OpenStack Compute packages and edited the `nova.conf` file as per our configurations and populated the Compute database. Then we restarted the VM.

Afterwards we installed the OpenStack Identity Service (Keystone) and the OpenStack Dashboard (Horizon) on the Controller Node. Then we created a network with Controller Node, Storage Node and the Network adapter.

After that we setup a SMTP mail server on Controller Node following the below steps [2]:

1. First we configured the Postfix Package, and created and set all the required usernames and passwords.
2. Next we created a virtual user through which all email messages will be managed using the features of postfix.
3. Then we created multiple postfix files and configured them as per our need like main.cf and master.cf, we can change user names, passwords etc. as per our choices.
4. Next we created the virtual maps using files alias.cf, user.cf and domain.cf. Also we then set the privileges and ownership to these maps using chmod and chown commands.
5. Then we set the SASL authentication (SSL/TLS) in the files like smtpd.conf, saslauthd.conf as well as the authentication privileges to these files.
6. Next we activated the courier services using the files authdaemonrc, authmysqlrc, imapd, imapd-ssl and pop3d-ssl.
7. Also we entered the information about the certificate which was created as asked once editing was done.
8. Next we configured and activated the services like amavis, spam assassin, clamAV. We also configured the MySQL database.
9. Afterwards we rebooted the services like saslauthd, postfix, and courier to make them run.
10. Finally, we created the mail directories (if not made we will not be able to login the mailbox).

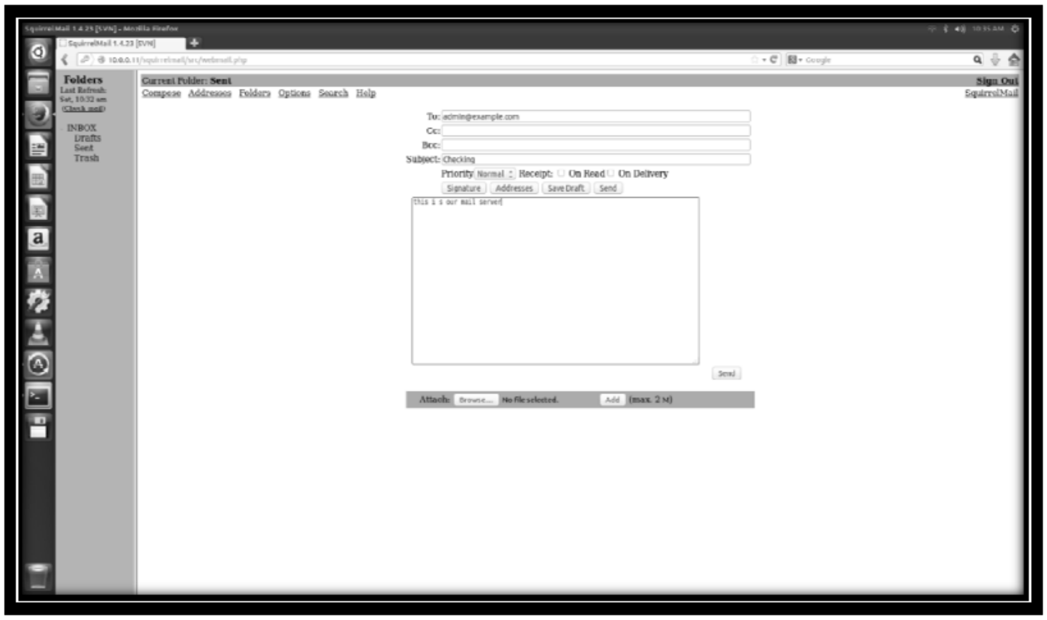
4.2 Creation of Storage Node

On Storage Node, first, we configured the network prerequisites [1]. Then we install the OpenStack Swift packages and obtain the accounting, container, and object service configuration files from the object storage source repository. Then we edited the account-server.conf, container-server.conf and object-server.conf files as per our configuration. Then we created a recon directory with proper ownership.

5. OBSERVATION

After observing and noting the results of the experiments conducted, we can conclude the following results:-

1. More Scalable than regular mail servers such as gmail, yahoo, etc.



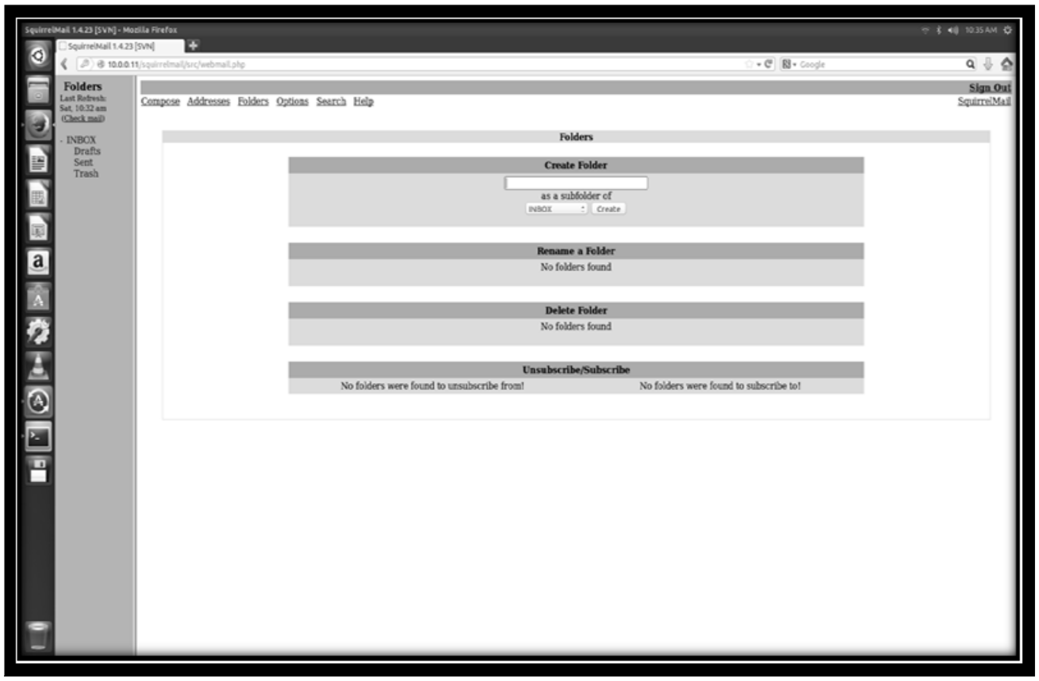
To: admin@example.com
Cc:
Bcc:
Subject: Checking
Priority: Normal Receipt: On Read On Delivery

```
this i s our mail server
```

Attach: Browse... No file selected. Add (max. 2 M)

Figure 1. Sending e-mail

2. Our mail server (MailChamp) is scalable as per the demand.



Folders

Create Folder

as a subfolder of INBOX

Rename a Folder

No folders found

Delete Folder

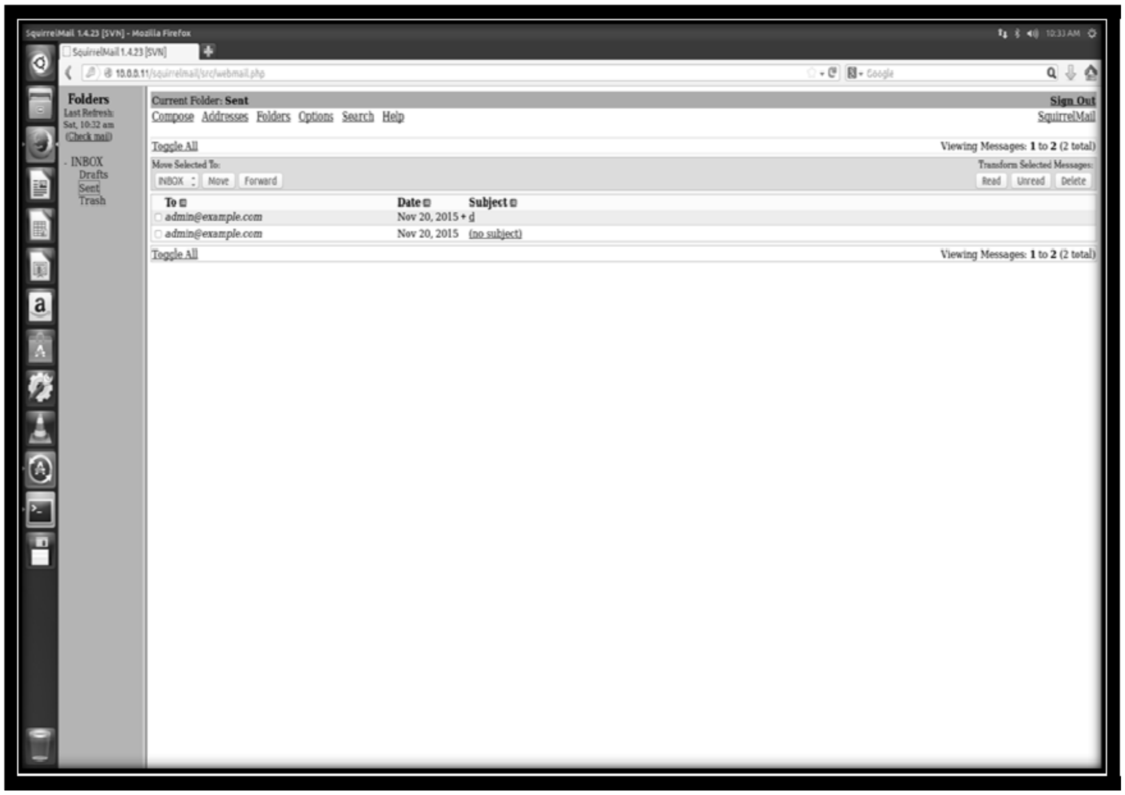
No folders found

Unsubscribe/Subscribe

No folders were found to unsubscribe from! No folders were found to subscribe to!

Figure 2: Monitoring e-mails

3. It is also capable of monitoring the sent mail, whether the mail is read or not.


SquirrelMail 1.4.23 [SVN] - Mozilla Firefox
SquirrelMail 1.4.23 [SVN]
10.0.0.11/squirrelmail/src/webmail.php
Folders
Last Refresh: Sat, 10:32 am
(Check mail)
INBOX
Drafts
Sent
Trash
Compose Addresses Folders Options Search Help
Sign Out
SquirrelMail
Toggle All
Move Selected To:
INBOX
Move
Forward
Transform Selected Messages:
Read
Unread
Delete

Current Folder: Sent

Viewing Messages: 1 to 2 (2 total)

To	Date	Subject
admin@example.com	Nov 20, 2015 +	d
admin@example.com	Nov 20, 2015	(no subject)

Viewing Messages: 1 to 2 (2 total)

Figure 3: Unlimited e-mailing

4. It also has no daily limit like in the cases of Gmail, Yahoo, etc.

Table 1
Results of the study

S. No	*Mail Server*	*Upper Messaging Limit (per account)*	*Mail Monitoring*
1	Gmail	100 [3]	Not Possible
2	Yahoo	100 [3]	Not Possible
3	Hotmail	100 [3]	Not Possible
4	MailChamp	Scalable	Possible

6. CONCLUSION

With the help of above methodology we were able to set up our own private cloud over a two node i.e. Storage (Swift) node and Controller (nova, keystone, horizon and SMTP mail server) node cluster of 0penStack, and thus finally deploying our SMTP mail server on 0penStack. After observing and noting the results of the experiments conducted, we can conclude that our mail server (MailChamp) is scalable as per the demand. We can increase the number of mails that can be sent at once over the server as per the demand (100 per message to 500 per message). It is also capable of monitoring the sent mail in whether the mail is read or not. A statistical result chart can be prepared for open read, click-through read, spam objections, and the numeral of the unsubscribers built on the basis of the outcomes to comprehend the accomplishment of any newsletter campaign or digital marketing campaign. Also that our mail server has no daily limits just like the other servers such as Gmail, Yahoo and many other servers.

References

[1] *OpenStack Installation Guide for Ubuntu 14.04*. (n.d.). Retrieved November 20, 2016, from http://www.docs.openstack .org/juno/install-guide/install/apt/content/

[2] Adam. (2013, September 8). *Creating a Mail Server on Ubuntu*. Retrieved from http://www.pixelinx.com/

[3] *Email Sending Limit and Send Rate – Gmail, Hotmail, Yahoo! Mail, AOL*. (n.d.). Retrieved November 20, 2016, from http:// www.yetesoft.com/free-email-marketing-resources/email-sending-limit/

[4] *Operator Training Guide*. (n.d.). Retrieved November 20, 2016, from http://docs.openstack.org/ icehouse/training-guides/con tent/bk_operator-training-guide.html

[5] I. Forster, J. Larshon, M. Masrich, A. Snoerean, S. M. Savage, and K. Levchenkro. *Security by many other names: On the effectiveness of the provider based email security*. In 22nd ACM Conference on the Computer and Communications Security, Oct. 2015.

[6] Klenshin, (2001) 'Simple Mail Transfer Protocol on Internet' IETF RFC 2821.