

# **Cloud Computing: An Architectural Perspective**

A Thesis Presented to

The Faculty of the Computer Science Program

California State University Channel Islands

In (Partial) Fulfillment

of the Requirements for the Degree

Masters of Science in Computer Science

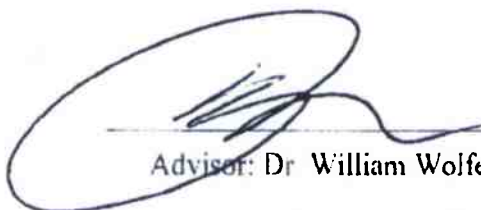
by


Hetalben Pandya

August 2012



APPROVED FOR THE COMPUTER SCIENCE PROGRAM

  
\_\_\_\_\_  
Advisor: Dr. William Wolfe      Date 8/22/2012

  
\_\_\_\_\_  
Dr. Andrzej Bieszczyk      Date 8/16/2012

  
\_\_\_\_\_  
Dr. Richard Wasniowski      Date Aug 16, 2012

APPROVED FOR THE UNIVERSITY

  
\_\_\_\_\_  
Dr. Gary A. Berg      Date 8/16/2012

## Non-Exclusive Distribution License

In order for California State University Channel Islands (CSUCI) to reproduce, translate and distribute your submission worldwide through the CSUCI Institutional Repository, your agreement to the following terms is necessary. The author[s] retain any copyright currently on the item as well as the ability to submit the item to publishers or other repositories.

By signing and submitting this license, you (the author[s] or copyright owner) grants to CSUCI the nonexclusive right to reproduce, translate (as defined below), and/or distribute your submission (including the abstract) worldwide in print and electronic format and in any medium, including but not limited to audio or video.

You agree that CSUCI may, without changing the content, translate the submission to any medium or format for the purpose of preservation.

You also agree that CSUCI may keep more than one copy of this submission for purposes of security, backup and preservation.

You represent that the submission is your original work, and that you have the right to grant the rights contained in this license. You also represent that your submission does not, to the best of your knowledge, infringe upon anyone's copyright. You also represent and warrant that the submission contains no libelous or other unlawful matter and makes no improper invasion of the privacy of any other person.

If the submission contains material for which you do not hold copyright, you represent that you have obtained the unrestricted permission of the copyright owner to grant CSUCI the rights required by this license, and that such third party owned material is clearly identified and acknowledged within the text or content of the submission. You take full responsibility to obtain permission to use any material that is not your own. This permission must be granted to you before you sign this form.

IF THE SUBMISSION IS BASED UPON WORK THAT HAS BEEN SPONSORED OR SUPPORTED BY AN AGENCY OR ORGANIZATION OTHER THAN CSUCI, YOU REPRESENT THAT YOU HAVE FULFILLED ANY RIGHT OF REVIEW OR OTHER OBLIGATIONS REQUIRED BY SUCH CONTRACT OR AGREEMENT.

The CSUCI Institutional Repository will clearly identify your name[s] as the author[s] or owner[s] of the submission, and will not make any alteration, other than as allowed by this license, to your submission.

Cloud Computing: An Architectural Perspective  
Title of Item

cloud, cloud-computing, cloud-comparisons, cloud-architectures  
3 to 5 keywords or phrases to describe the item

HETALBEN MAYANK PANDYA  
Author[s] Name (Print)

HPandya 08/24/2012  
Author[s] Signature Date

# **Cloud Computing: An Architectural Perspective**

by

Hetalben Pandya

Computer Science Program

California State University Channel Islands

## **Abstract**

Cloud Computing is a term heavily used in today's world. Not even a day passes by without hearing the words "Cloud Computing". It has become an increasingly important part of every person's life. However, it is still largely an unknown entity to the majority of people. Learning about what it is and how it works would be very beneficial to all who use it. My goal for this study is to take an architectural perspective towards the implementation of Cloud Computing.

On a high level, there are different kinds of cloud service available, and it needs a cloud client in order to access those services. Software-as-a-Service (SaaS) is a cloud delivery model, which is used to deliver the software application as a service over the Web. It eliminates the need to deploy or execute the application on the user's own machines. Since an application vendor is responsible for the maintenance and support of the application in the SaaS model, it allows the user to focus on using the application. Platform-as-a-Service (PaaS) is a cloud delivery model, which delivers a platform and a solution stack as a service to allow its users to develop and deploy the application, just by using the PaaS solution. It allows users to focus on the development and deployment aspect of the application and relieves them from the complexity of buying and managing the computing resources [48]. Infrastructure-as-a-Service (IaaS) is a delivery model that delivers computing infrastructure such as virtualized datacenters as a service. In this model, it provides users a hardware/software layer with computing resources, operating systems in the form of virtualized infrastructure.

A careful examination of different Cloud Computing offerings and their providers can suggest alternatives about the advantages and disadvantages of their services. This study analyzes various Cloud Computing services and their providers in order to enlighten users about choices that can be made in moving their application to the cloud or moving to the cloud in general.

### **Acknowledgements**

I would like to thank Dr. William wolf for his guidance and support to finish my thesis in time. I would like to thank Dr. Andrzej Bieszczad and Dr. Richard Wasniowski for evaluating my thesis. I am extremely grateful to Prof. Carolyn Asari for her help in proof reading my thesis work. I would like to whole heartedly thank my 1-year old son Het, who unknowingly supported me, and finally I would like to thank my husband for his guidance and support, without whom the Masters Program would have been only a dream for me.

## TABLE OF CONTENTS

<b>CHAPTER 1: INTRODUCTION.....</b>	<b>11</b>
1.1 BACKGROUND.....	11
1.2 DEFINING CLOUD COMPUTING .....	12
1.3 THE EVOLUTION OF CLOUD COMPUTING [2].....	13
1.3.1 Idea Phase.....	13
1.3.2 Pre Cloud Phase.....	15
1.3.3 Cloud Phase .....	15
1.4 CLOUD COMPUTING VERSUS GRID COMPUTING .....	16
1.4.1 Grid Computing .....	16
1.4.2 Cloud Computing.....	16
1.5 REMAINING CHAPTERS.....	17
<b>CHAPTER 2: CLOUD MODEL.....</b>	<b>18</b>
2.1 ESSENTIAL CHARACTERISTICS.....	18
2.1.1 On-Demand Self-Service.....	18
2.1.2 Broad Network Access.....	18
2.1.3 Resource Pooling.....	18
2.1.4 Rapid Elasticity.....	19
2.1.5 Measured Service .....	19
2.2 CLOUD DELIVERY MODELS.....	19
2.2.1 Software-as-a-Service (SaaS).....	19
2.2.2 Platform-as-a-Service (PaaS).....	20
2.2.3 Infrastructure-as-a-Service (IaaS).....	21
2.3 CLOUD DEPLOYMENT MODELS.....	22
2.3.1 Private Cloud .....	22
2.3.2 Community Cloud .....	23
2.3.3 Public Cloud.....	24
2.3.4 Hybrid Cloud .....	25
<b>CHAPTER 3: EVALUATION METHOD AND CRITERIA .....</b>	<b>27</b>
3.1 EVALUATION METHOD.....	27
3.1.1 Setup.....	27
3.2 EVALUATION CRITERIA .....	29
<b>CHAPTER 4: AMAZON WEB SERVICES.....</b>	<b>30</b>
4.1 AMAZON ELASTIC COMPUTE CLOUD (EC2) .....	31
4.1.1 Features of Amazon EC2 Based on my Analysis.....	31
4.1.2 Types of Amazon EC2 Instances.....	31
4.2 ELASTIC BLOCK STORAGE (EBS).....	32
4.2.1 Advantages .....	32
4.2.2 Disadvantages.....	32
4.3 AMAZON VIRTUAL PRIVATE CLOUD (VPC) .....	32
4.4 EC2 CLOUD WATCH .....	33
4.5 AMAZON SIMPLE STORAGE SERVICE (AMAZON S3) .....	33
4.5.1 Advantages.....	33
4.5.2 Disadvantages [17].....	33
4.6 ELASTIC MAPREDUCE (EMR).....	34
4.7 RELATIONAL DATABASE SERVICE (RDS).....	34

4.8 AMAZON ELASTIC BEANSTALK .....	34
4.9 EXPERIMENT .....	35
4.9.1 Background .....	35
4.9.2 Implementation .....	35
4.9.3 Deploying an Application .....	36
4.9.4 Creating MySQL DB Instance in RDS .....	38
4.9.5 Application Execution .....	40
4.10 ANALYSIS .....	42
4.10.1 Ease of Use .....	42
4.10.2 Reliability .....	42
4.10.3 Scalability .....	42
4.10.4 Security .....	43
4.10.5 Economics .....	43
4.10.6 Fault Tolerance .....	43
<b>CHAPTER 5: MICROSOFT WINDOWS AZURE PLATFORM .....</b>	<b>44</b>
5.1 WINDOWS AZURE PLATFORM ARCHITECTURE .....	44
5.2 WINDOWS AZURE .....	45
5.2.1 Compute .....	45
5.2.2 Storage .....	46
5.2.3 Fabric Controller .....	46
5.2.4 Caching .....	46
5.2.5 Connect .....	46
5.2.6 Content Delivery Network (CDN) .....	46
5.3 SQL AZURE .....	46
5.3.1 SQL Azure Database .....	47
5.3.2 SQL Azure Reporting .....	47
5.3.3 SQL Azure Data Sync .....	47
5.4 WINDOWS AZURE-BASED SERVICES .....	48
5.4.1 Service Bus .....	48
5.4.2 Access Control .....	48
5.5 EXPERIMENT .....	48
5.5.1 Background .....	49
5.5.2 Setup .....	49
5.6 ANALYSIS .....	52
5.6.1 Ease of Use .....	53
5.6.2 Reliability .....	53
5.6.3 Scalability .....	53
5.6.4 Security .....	54
5.6.5 Economics .....	54
5.6.6 Fault Tolerance .....	54
<b>CHAPTER 6: GOOGLE APP ENGINE PLATFORM .....</b>	<b>55</b>
6.1 FEATURES .....	55
6.2 BUILDING BLOCKS .....	56
6.2.1 Building Blocks .....	56
6.2.2 Data Storage .....	56
6.2.3 Google Accounts .....	57
6.2.4 App Engine Services .....	57
6.3 EXPERIMENT .....	58
6.3.1 Background .....	58
6.3.2 Setup .....	58



6.3.3 Performance Measures.....	61
6.4 ANALYSIS .....	62
6.4.1 Ease of Use .....	62
6.4.2 Reliability .....	63
6.4.3 Scalability.....	63
6.4.4 Security.....	63
6.4.5 Economics .....	64
6.4.6 Fault Tolerance.....	64
<b>CHAPTER 7: EUCALYPTUS CLOUD PLATFORM .....</b>	<b>65</b>
7.1 EUCALYPTUS ARCHITECTURE .....	65
7.2 CHARACTERISTICS OF EUCALYPTUS.....	67
7.3 EXPERIMENT .....	68
7.3.1 Background.....	68
7.3.2 Setup.....	68
7.3.3 Implementation.....	74
7.4 ANALYSIS .....	75
7.4.1 Ease of Use .....	75
7.4.2 Reliability .....	75
7.4.3 Scalability.....	75
7.4.4 Security.....	76
7.4.5 Economics .....	76
7.4.6 Fault Tolerance.....	76
<b>CHAPTER 8: CLOUD COMPUTING PLATFORM COMPARISONS .....</b>	<b>77</b>
8.1 COMPARISON BASED ON COMMON CHARACTERISTICS.....	77
8.2 COMPARISONS BASED ON FREE RESOURCES .....	78
8.3 COMPARISONS BASED ON PAID RESOURCES.....	79
8.4 CPU INSTANCES COMPARISON .....	80
<b>CHAPTER 9: CONCLUSION.....</b>	<b>81</b>
<b>CHAPTER 10: FUTURE WORK.....</b>	<b>83</b>
<b>REFERENCES.....</b>	<b>85</b>

## TABLE OF FIGURES

FIGURE 1 - EVOLUTION OF CLOUD .....	14
FIGURE 2 - CLOUD COMPUTING VS. GRID COMPUTING [1] .....	17
FIGURE 3 - A PRIVATE CLOUD DEPLOYMENT EXAMPLE [4] .....	23
FIGURE 4 - EXAMPLE OF PUBLIC CLOUD [4] .....	25
FIGURE 5 - EXAMPLE OF HYBRID CLOUD [4] .....	26
FIGURE 6 - PASSWORD MANAGER ARCHITECTURE.....	28
FIGURE 7 - AMAZON WEB SERVICES [12] .....	30
FIGURE 8 - AMAZON ELASTIC BEANSTALK APPLICATION CREATION .....	36
FIGURE 9 - LAUNCH NEW ENVIRONMENT FOR APPLICATION DIALOG .....	37
FIGURE 10 - SECURITY GROUP CONFIGURATION .....	38
FIGURE 11 - AMAZON RDS DB INSTANCE INFORMATION .....	39
FIGURE 12 - AMAZON RDS DB SECURITY GROUPS CONFIGURATIONS .....	40
FIGURE 13 - PASSWORD MANAGER APPLICATION.....	41
FIGURE 14 - PASSWORD INFORMATION PAGE ON PASSWORD MANAGER .....	41
FIGURE 15 - MICROSOFT WINDOWS AZURE PLATFORM [21].....	44
FIGURE 16 - WINDOWS AZURE PROVIDES COMPUTE AND STORAGE SERVICES IN THE CLOUD [21] .....	45
FIGURE 17 - SQL AZURE PROVIDES RELATIONAL DB SERVICES IN CLOUD [21].....	47
FIGURE 18 - WINDOWS BASED SERVICE PROVIDE INFRASTRUCTURE THAT CAN BE USED BY BOTH CLOUD AND ON-PREMISES APPLICATIONS [21] .....	48
FIGURE 19 - MANAGEMENT CONSOLE OF WINDOWS AZURE.....	49
FIGURE 20 - ENDPOINTS CONFIGURATION ON WINDOWS AZURE VIRTUAL MACHINES .....	50
FIGURE 21 - DASHBOARD REPRESENTS DIFFERENT GRAPHS FOR A VIRTUAL MACHINE IN WINDOWS AZURE PORTAL.....	51
FIGURE 22 - THE USAGE OVERVIEW OF RESOURCES ON WINDOWS AZURE PORTAL.....	51
FIGURE 23 - PASSWORD MANAGER LOGIN PAGE ON WINDOWS AZURE CLOUD .....	52
FIGURE 24 - OVERVIEW OF GOOGLE APP ENGINE FOR JAVA [31] .....	57
FIGURE 25 - CREATE APPLICATION USING GOOGLE APP ENGINE PORTAL.....	59
FIGURE 26 - MY APPLICATIONS PAGE IN GOOGLE APP ENGINE PORTAL .....	59
FIGURE 27 - PASSWORD APP AUTHENTICATION USING GOOGLE ACCOUNTS.....	60
FIGURE 28 - ONE PASS APPLICATION ON APP ENGINE CLOUD - FIRST TIME USER.....	60
FIGURE 29 - PASSWORD INFORMATION FOR RETURNING USER.....	61
FIGURE 30 - LOAD STATISTICS FOR PASSWORD APP .....	62
FIGURE 31 - EUCALYPTUS COMPONENTS ARCHITECTURE [35] .....	66
FIGURE 32 - EUCALYPTUS ADMIN CONSOLE LOGIN SCREEN.....	69
FIGURE 33 - EUCALYPTUS ADMIN CONSOLE - CREDENTIALS PAGE.....	70
FIGURE 34 - CONFIGURATION 1 - EUCALYPTUS ADMIN CONSOLE .....	71
FIGURE 35 - CONFIGURATION 2 - EUCALYPTUS ADMIN CONSOLE .....	72
FIGURE 36 - EUCALYPTUS INSTANCES TAB ON HYBRIDFOX PLUGIN FOR FIREFOX.....	73
FIGURE 37 - AVAILABLE IMAGES ON EUCALYPTUS SHOWED ON HYBRIDFOX PLUGIN.....	73
FIGURE 38 - GROUP PERMISSIONS FOR EUCALYPTUS USING HYBRIDFOX.....	74
FIGURE 39 - PASSWORD MANAGER APPLICATION RUNNING ON EUCALYPTUS CLOUD .....	74
FIGURE 40 - CPU SPEED INFO GRAPH PROVIDED BY CLOUD COMPUTING PLATFORMS.....	80

## TABLE OF TABLES

TABLE 1 - CPU & RAM SIZES PROVIDED BY GOOGLE APP ENGINE.....	62
TABLE 2 - CONFIGURATION OF MACHINES USED TO DEPLOY EUCALYPTUS CLOUD.....	68
TABLE 3 - CLOUD COMPUTING PLATFORMS COMPARISONS BASED ON COMMON CHARACTERISTICS [40] [41] [42].....	78
TABLE 4 - COMPARISON BASED ON FREE RESOURCES PROVIDED BY CLOUD PLATFORMS [43].....	79
TABLE 5 - COMPARISON BASED ON PAID RESOURCES PER CLOUD PLATFORM.....	80

# **Chapter 1: Introduction**

The first chapter introduces the motivation behind the thesis as well as the topic of Cloud Computing. This includes a discussion of the origins and evolution of Cloud Computing, including a brief comparison with Grid Computing.

## **1.1 Background**

During my previous courses, I came across different papers that were related to evaluations of different Cloud Computing platforms. However, I felt that it was difficult for me to conclude which platform to choose, in case I need to deploy my own Java based application. Then I decided to do more research in this area by evaluating different Cloud Computing platforms in regard to develop and deploy an application. So that at the end, I can provide the users with the suggestions on which platform to use based on their needs.

During this process I came across multiple Cloud Computing platform service providers, such as Amazon Web Services, Abiquo, Windows Azure, Salesforce, Rackspace, Nimbus, OpenNebula, Google App Engine, Eucalyptus, CloudStack to name a few. I went through their documentations on how to develop an application using their solutions and how to deploy on their architectures. However, I found that some platforms are very popular among different sets of users. They are also user-friendly and easy to use. They are considered as pioneers in Cloud Computing space. So I decided to evaluate four Cloud Computing platforms: Amazon Web Services, Windows Azure, Google App Engine and Eucalyptus.

Another reason I selected these four platforms is that they represent different classes of deployment models and service models that is offered in industry. Amazon Web Services and Windows Azure are public clouds and provides Infrastructure-as-a-Service, Google App Engine is a public cloud and provides Platform-as-a-Service and lastly Eucalyptus is a private cloud and provides Infrastructure-as-a-Service.

In order to evaluate these platforms, I have decided to use an application that I worked on during previous semesters. I will be using these platforms to deploy my application and in the process evaluate them using predefined criteria, which I will explain later in the thesis.

In the coming sections, I will describe more about deployment models, service models and Cloud Computing platforms.

## **1.2 Defining Cloud Computing**

The phrase "Cloud Computing" has become so common that it is often found in several news reports in a single day, but what does it actually mean? People use cloud in one form or another throughout the day. If you have used any of the popular mail services such as Gmail or Hot mail, then you have used the Cloud. Simply put, Cloud Computing is a set of pooled computing resources and services delivered over the Web [1].

With the PC revolution, mass storage and cheap CPUs became easily available to the average corporate desktop. Enterprising users started using a file server to share and archive the documents. PC clients were able to produce and consume the CPU cycles, required to do productive work [1].

The idea of a "Grid" started taking place, when there were a number of computers attached to each other using the Internet, even when it was going through the inception phase in the early 1990s. People who were deeply involved with the technology started thinking about how they could create a pool of resources that can be shared to harvest the large amount of computing power from it, which resulted in the idea of "Grid" [3].

According to Peter Mell and Tim Grance of NIST (National Institute of Standard and Technology):

“Cloud Computing is a model for enabling convenient, on-demand network access to a shared pool of configurable and reliable computing resources (e.g., networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal consumer management effort or service provider interaction” [5].

The Cloud Model has three fundamental aspects associated with it. It states five essential characteristics that every cloud model should have. It also mentions that there are three different service models and four different cloud deployment models. In this chapter, I will just mention the characteristics and the models, and I will elaborate more on these in chapter 2.

Every cloud should have following five characteristics:

- On-demand self-service
- Ubiquitous network access
- Resource pooling
- Location independence
- Rapid elasticity
- Measured service

Cloud service models are as follows:

- Software-as-a-Service or SaaS
- Platform-as-a-Service or PaaS
- Infrastructure-as-a-Service or IaaS

Cloud platform can be implemented or deployed using the following 4 different deployment models:

- **Public Cloud:** A Publicly available Cloud
- **Private Cloud:** It may be leased or owned by a private company or an enterprise
- **Community Cloud:** Available to a specific community which shares common concerns
- **Hybrid Cloud:** A combination of one or more clouds mentioned above

## **1.3 The Evolution of Cloud Computing [2]**

Is it a fair question to ask oneself, where the term “Cloud Computing” originated. Who are the people responsible for the idea of utility computing? When did the hype around Cloud start [2]? The evolution of Cloud Computing can be split into 3 phases:

- **Idea Phase:** The idea of utility computing and grid computing, evolved during this phase. This phase had begun in the 1960s. It lasted till the late 1990s.
- **Pre Cloud Phase:** The Internet as a medium began to provide Application as a Service. It spanned from 1999 to 2006.
- **Cloud Phase:** In this phase Cloud Computing became popular and cloud delivery models such as SaaS, PaaS and IaaS became formalized. This phase started in 2007, and it is still continuing.

Three phases of Cloud Evolution are discussing below in detail: the Idea Phase, the Pre Cloud Phase, and the Cloud Phase.

### **1.3.1 Idea Phase**

There were many concepts that were developed during the idea phase, such as the concept of Utility Computing and Grid Computing. This phase spanned from 1960s to late 1990s.

- J.C.R. Licklider brought the idea of Cloud Computing to the light.
- John McCarthy shared his opinion about “ Possible similarities about Computing may be organized as a public utility.
- “The Challenge of the Computer Utility”, a book published by Douglas Parkhill, which compares Cloud Computing characteristics with the electricity industry. This book also explored the use of electricity in public, private, government and community form.

- The concept of Grid came to life as people started thinking about creating shared pool or computing resources. Ian Foster and Card Kesselman published their book, “The Grid: Blueprint for a New Computing Infrastructure. They used the terminology that belongs to an electricity grid, where one can plug in to the grid and pay for the used utilities.

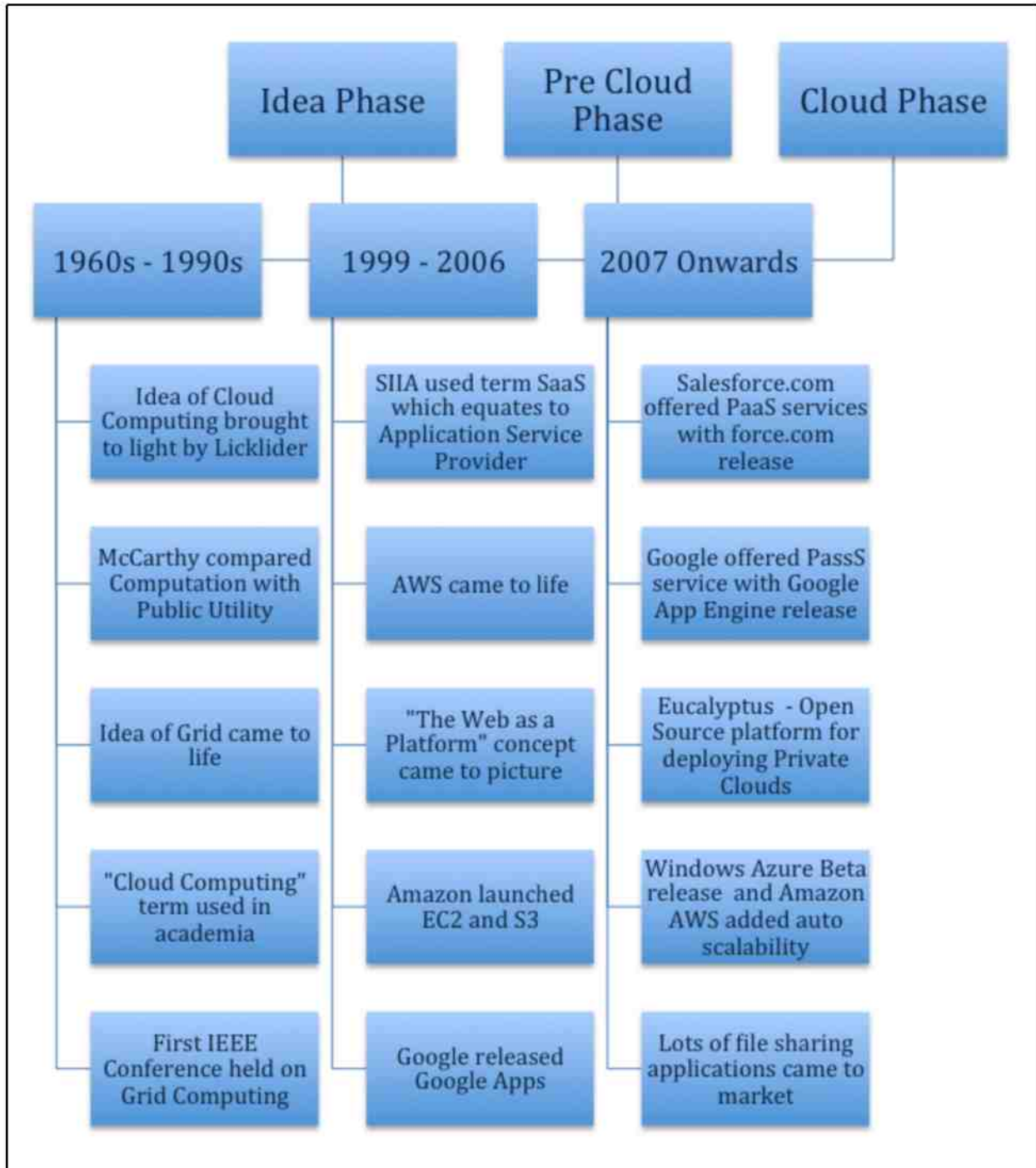


Figure 1 - Evolution Of Cloud

- The Indian professor, Ramnath Chellappa, used the term “Cloud Computing” in one of his lecture in 1997.
- Also the first IEEE conference held on Grid Computing in Bangalore, India during this phase.

### **1.3.2 Pre Cloud Phase**

In this phase, the idea of providing Application as a Service through the medium of the Internet evolved.

- The Software & Information Industry Association (SIIA) used the abbreviation SaaS and explained the term as Application Service Provider
- Amazon.com launched the Amazon Web Service, which allowed users to integrate with their online content
- In 2004, Tim O'Reilly and Dale Dougherty explained the concept of “The Web as Platform” in the first Web 2.0 conference, in San Francisco.
- The Amazon launched its Elastic Compute Cloud (EC2), which allowed users to rent the computing power to run their applications. S3 service launched as pay-per-use which was a unique offering at that time; it is quite common now.
- In the same year as EC2 and S3 release, Google released their online service offerings known as Google Apps

### **1.3.3 Cloud Phase**

In this phase, the Cloud Computing model becomes extremely popular. Lots of companies started offering different Cloud Services. The cloud service models are standardizing as IaaS, PaaS and SaaS.

- In the beginning of this phase, Salesforce.com launched their Platform-as-a-Service offering known as Force.com
- In 2008, Google released Google App Engine, which is a Platform-as-a-Service based offering. The pricing for this service is aggressive, and its entry-level plans were starting for free.
- The group of students with their professor at UCSB, Santa Barbara, developed an open-source platform for deploying Private Clouds, known as Eucalyptus.
- Microsoft released its PaaS service Windows Azure.



- In late 2010, a lot of services came to market, which can support a personal cloud option and allow users to store their documents in the cloud with universal availability.

## **1.4 Cloud Computing versus Grid Computing**

Cloud Computing and Grid Computing are remarkably similar concepts, but there are some fundamental differences that separated them from each other. Differences between these two entities are not commonly known to the public, which results in a lot of confusion.

### **1.4.1 Grid Computing**

Grid Computing is composed of multiple inter-connected computers. A piece of software is installed on all the connected computers, which is responsible for dividing the large problem in to several smaller pieces and executes them on thousands of computers on the grid using parallel processing.

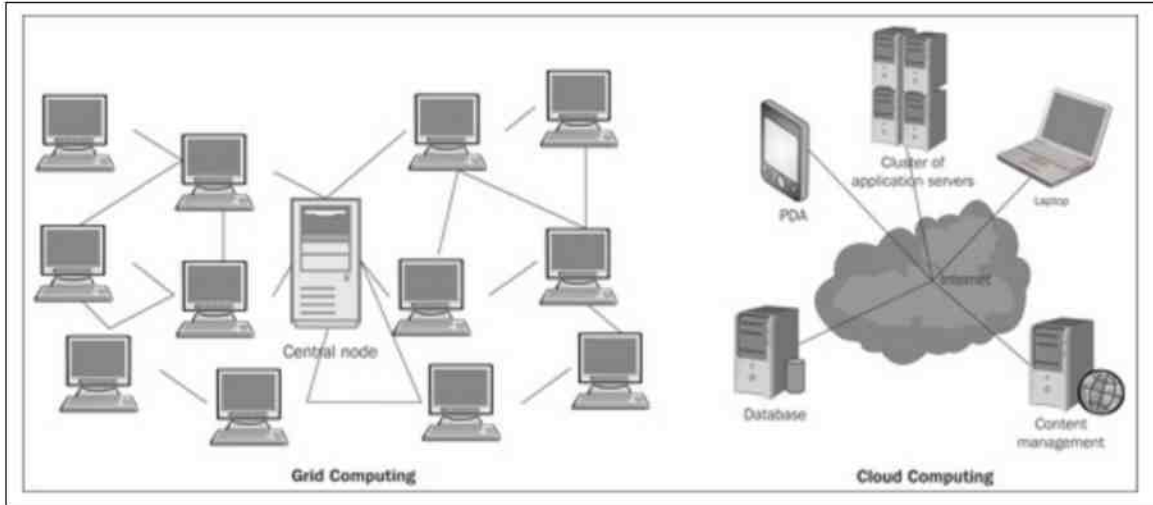
Failure of a single node on Grid Computing, sometimes results in complete system failure if processing of certain job depends on that failed node. This characteristic belongs to Grid computing and not the cloud [1].

In Grid Computing, we can provision the computing resources up front as utilities, and we pay for only what we use.

### **1.4.2 Cloud Computing**

Cloud Computing certainly evolved from Grid Computing, but there are considerable differences between them. In Cloud Computing a user can apply the computing resources on-demand.

Users do not need to invest in computing resources and can leverage Cloud Computing to deploy their applications. Cloud Computing instances are easily scalable; when the load on the system increases, the user can increase the computing capacity and reduce them once the load on the system decreases.



**Figure 2 - Cloud Computing vs. Grid Computing [1]**

## 1.5 Remaining Chapters

This study will include the following chapters in detail.

In chapter 2, I will be explaining Cloud models, their characteristics, cloud deployment models and cloud delivery models.

Chapter 3 will focus on Evaluation Method and Criteria.

Chapter 4 will focus on the Amazon Web Services.

Chapter 5 will focus on the Microsoft Windows Azure Cloud Computing platform.

Chapter 6 will focus on the Google App Engine Cloud Computing platform.

Chapter 7 will focus on the Eucalyptus Private Cloud Computing platform.

Chapter 8 will focus on comparisons of Cloud Computing platforms explained in the above-mentioned chapters.

Chapter 9 and 10 will focus on conclusions and future work considerations respectively.

## **Chapter 2: Cloud Model**

This chapter gives an overview of five essential characteristics for the cloud model, three cloud service models and four cloud deployment models.

### **2.1 Essential Characteristics**

There are 5 different characteristics of Cloud Computing based on the NIST definition of Cloud Computing.

#### **2.1.1 On-Demand Self-Service**

A user should be able to apply the computing resources such as computing power, storage, network etc., when required. A user should not have to receive help from the service provider or any third party to schedule for the use of the expected resources. A user should be able to scale up or scale down the computing resources at will [5]. User should have complete control over the scheduled resources so that he can perform his tasks, such as maintenance or troubleshooting, without someone intervening. Since there is no direct interaction between the user and the provider, it is easier and more cost effective to utilize.

#### **2.1.2 Broad Network Access**

Computing resources should be accessible over the network. The Cloud Computing platform should be accessible from the usual means of network access, such as from laptops, desktops, and mobile devices. It should have sufficient bandwidth available to it such that it can be accessed from anywhere regardless of the location of the resources versus clients.

#### **2.1.3 Resource Pooling**

Resource pooling is a fundamental characteristic of the Cloud Computing model. The cloud provider must have sufficient resources such that it can support the user's demand. It should allow users to scale their resources when required. Computing resources should be easily available to the applications and, it should be efficiently assigned to it when needed. Also, the resources should be available at multiple locations and should give users a feeling of independence.

The NIST definition of resource pooling states that, *"There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of provided resources but may be able to specify the location at a higher level of abstraction (e.g., country, state, or datacenter)"* [5].

#### **2.1.4 Rapid Elasticity**

Rapid elasticity allows users to provision the computing resources elastically. It should also allow users to release the resources when not required. Some time it should do provision and release of the resources automatically so that, the user application can scale up when demand is high and scale down when demand is less. Elasticity should also allow users to grow their resource without any limit; if a user requires unlimited storage then he should be able to acquire those resources and should not be capped.

#### **2.1.5 Measured Service**

Since the Cloud model is based on the Utility Computing, users can use the computing resources as per their needs. These resources are allocated to the user on demand and should be monitored by the service provider. A user is bound to pay for the resources that are used by the application.

As per NIST, measured service is defined as, *"Cloud systems that automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth and active user accounts). Resource usage can be monitored, controlled, and reported providing transparency for both the provider and consumer of the utilized service"* [5].

### **2.2 Cloud Delivery Models**

There are three different cloud delivery models: Software-as-a-Service, Platform-as-a-Service and Infrastructure-as-a-Service. They are described in the following subsections.

#### **2.2.1 Software-as-a-Service (SaaS)**

Software-as-a-Service (SaaS) is a service delivery model, in which an application provider can provide their applications using the Internet as a medium. In most cases, SaaS applications are available to users through licensing. The application provider is responsible for development and deployment of the application. The end user or client need not know the whereabouts of the application deployment or maintenance. The provider makes sure that the application is available for use at all times.

SaaS applications can be accessed through usual means through the Internet using a Web browser. Currently there are a lot of different ways to access the application. An application

provider should support different devices through which the application is accessible, such as laptops, mobile phones, tablet computers, etc. The availability of the SaaS application varies; some applications are available for free while some of them require a license or a subscription to the application for a pre-determined fee by the provider.

According to the NIST, the definition of Cloud Software as a Service (SaaS) is as follows:

*"The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based e-mail). The consumer does not manage or control the underlying cloud infrastructure, including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings" [5].*

### 2.2.1.1 Benefits

It enables organizations to outsource the application hosting to an independent vendor or the other service provider, which reduces the licensing cost, hardware management and other resources required to host the application locally.

SaaS allows the application provider to have better control over the application and use of software, by limiting the distribution of unlicensed copies and allows the vendor to have greater management control.

It also allows the application vendors to create and control multiple revenue streams with a one-to-many model.

The client of an application can access the application more readily through a browser or through supported apps on their mobile devices.

### 2.2.2 Platform-as-a-Service (PaaS)

Platform-as-a-Service allows users to develop and deploy the applications on their cloud solution. A user can develop an application using their virtual development platform and deploy the same application on their Cloud Computing platform.

Most of the PaaS service providers provide their users with an application development toolkit and a way to develop the application in their cloud. A user does not have to install anything on their local machines. They can deploy the application easily on the provider's platform.

NIST describes PaaS as follows:

*"The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure*

*including network, servers, operating systems, or storage, but has control over the deployed applications and possibly the application hosting environment configurations" [5].*

The PaaS service provider supplies the following services to the application developers:

- A software Development Kit to develop applications, which is custom made to work with the provider's virtual environment
- A set of APIs to support the different aspects of application development
- The ability to support multiple application development language platforms, i.e. Java, Python and Ruby.
- The ability to provide an easy and efficient way to test and deploy an application on the virtual environment.

### *2.2.2.1 Features*

PaaS services allow the user to start using the service for a small fee or sometimes for free. It is easy to start developing an application using PaaS framework.

PaaS solution can allow development, testing and deployment of an application, which is helpful to the application developers, as they do not have to worry about the hardware and software acquisition required for the application life cycle.

Some PaaS service providers give complete access to the deployment environment of its users, such as Amazon Elastic Beanstalk, where users can have complete control over the environment in which their applications run. On the other hand PaaS service providers, such as Google App Engine, does not allow users to have any kind of control over the deployment environment.

PaaS has a limited user base, such as application developers, and because of that there are few PaaS providers that offer these services.

As specified above, Amazon AWS offers PaaS services using a different set of services; Amazon Elastic Beanstalk is one of them. Google App Engine is another service which is PaaS based. Salesforce.com is also offering their PaaS based service through force.com.

### **2.2.3 Infrastructure-as-a-Service (IaaS)**

Infrastructure-as-a-Service (IaaS) is the cloud delivery model that provides their users with virtualized computing infrastructure, such as datacenters.

The NIST definition for Infrastructure-as-a-Service says:

*"The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems,*

*storage, deployed applications, and possibly limited control of select networking components” [5].*

The Infrastructure-as-a-Service cloud model allows corporations to, rapidly scale the infrastructure on demand in the cloud as compared to traditional IT infrastructure. Companies spend a lot of their budgets on procuring the hardware, software and other resources that need to manage those resources. Instead, using IaaS solutions allows companies to reduce their cost and also provides scalability, which is not possible in traditional IT infrastructure.

There are a wide range of IaaS providers, and they offer a variety of services. There are some open-source solutions available, such as Eucalyptus, which allow users to deploy a private cloud. Some provide more user oriented services such as cloud based storage service (i.e. Amazon S3); some provide on demand computation capability, such as Amazon Web Services.

IaaS allows pooling of resources efficiently so that computing power, storage, network devices and other components [4].

## **2.3 Cloud Deployment Models**

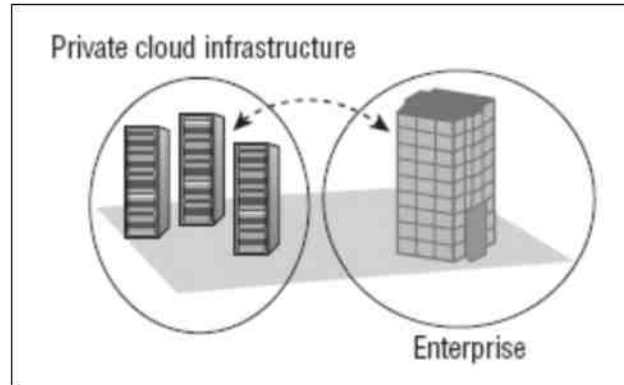
There are four deployment models: private, community, public and hybrid. All the deployment models are described below.

### **2.3.1 Private Cloud**

With the concept of datacenter virtualization, many organizations create the cloud infrastructure exclusively to use by the business and its customers. Such a cloud deployment model is known as A Private Cloud or enterprise cloud. A private cloud provides all the features of Public cloud, but it is exposed to limited users. It is still better than traditional IT infrastructure, because there is still an availability of cloud features such as resource pooling and automatic scalability.

According to NIST, private cloud is defined as

*“The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers. It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises” [5].*



**Figure 3 – A Private Cloud Deployment Example [4]**

### 2.3.1.1 Features

- There is a greater sense of security in a private cloud over a public cloud. Access to the cloud is highly restricted, so companies that handle sensitive customer data, such as banks, medical record-keeping companies, etc. feel more secure with a private cloud rather than a public cloud.
- The private cloud implements the virtualization to create datacenters, which save organizations a significant amount of money from buying physical hardware, software and network equipment, unlike traditional IT infrastructure. It is a significant advantage for the company, which decides to move to a private cloud.
- A private cloud can be combined with other cloud deployment models and leverage a lot of data for the other clouds. It can be combined with the public or community cloud to create a hybrid or government cloud.

### 2.3.2 Community Cloud

A community cloud is dedicated to people or organizations that belong to the community and that serve the same purpose and share the same concerns. An excellent example would be Nimbus Cloud, which is dedicated to the science community, as it is available to all users who are doing research in any field of science.

The NIST definition of Community cloud is as follows:

*“The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises” [5].*



### 2.3.3 Public Cloud

Public Cloud is a deployment model in which computing resources such as servers, storage and software is available to the general public over the Internet [8]. These services are offered for a free or using pay-per-use model [9].

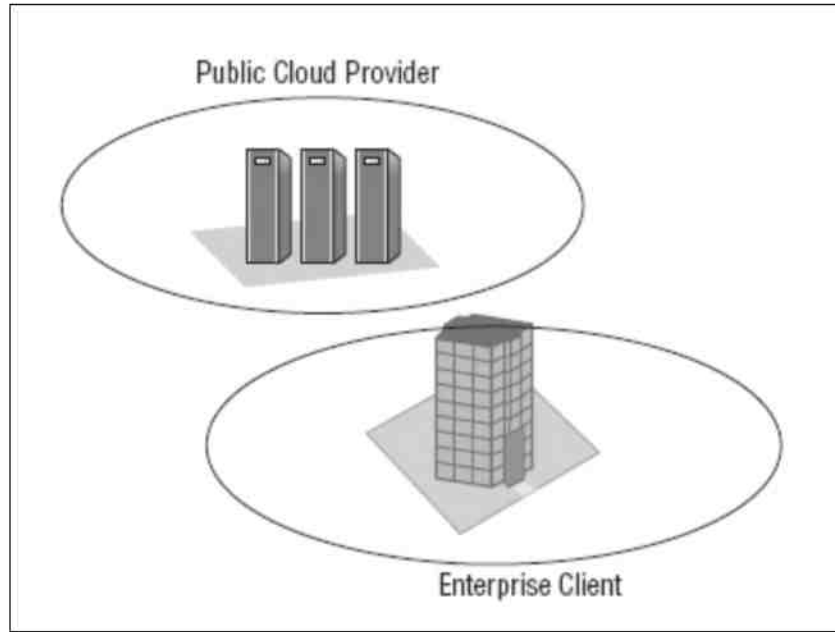
The general public can be defined as any cloud users, which ranges from individual application developer, a single cloud application user, a business user or an organization. The infrastructure is owned and operated by cloud providers. There are different application providers that implement public clouds and provide services, such as Google App Engine, Windows Azure, Salesforce.com and Amazon Web Services.

According to the NIST definition of Public cloud is as following

*“The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider”* [5].

#### 2.3.3.1 Features

- It is operated and managed by the cloud service provider; users of the cloud should not be concerned about the maintenance and compliance of the application. The cloud vendor or cloud service provider is responsible for system maintenance.
- The use of public cloud is remarkably economical for users, because they will be sharing the platform and they do not have to worry about investing in hardware resources. It is an immediate cost saving for users.
- Features include highly scalable computing resources, and it implements pay-as-you-use model.
- Security in the private cloud is extremely important, as computing resources are being shared. Most public cloud providers utilize the firewall for security in the cloud, but there is still hesitation about security in the cloud. Institutions such as banks, which are responsible for critical-sensitive customer data, are skeptical about moving their business to the cloud.



**Figure 4 - Example of Public Cloud [4]**

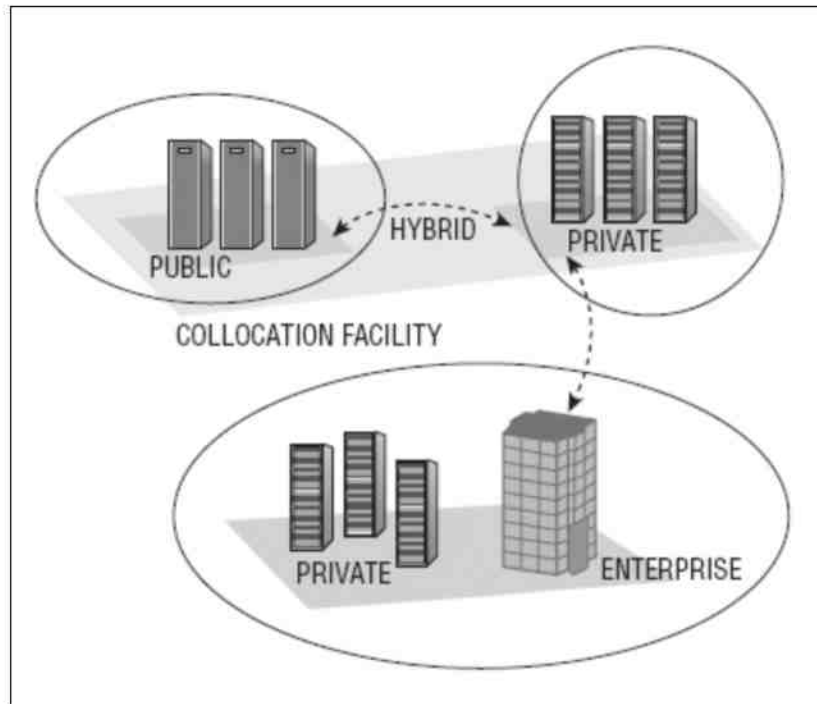
### 2.3.4 Hybrid Cloud

Any combination of private, community or public cloud deployment model is known as a hybrid cloud. NIST defines a hybrid cloud as:

*"A composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability" [5].*

Companies can leverage the public cloud platform by outsourcing non-critical functionality to public cloud and still keeping sensitive data that it is processing on a private cloud. **Figure 5** shows a typical hybrid cloud deployment, which includes public and private clouds.

Hybrid clouds usually implements the concept called “cloudburst”. It is a concept in which the application runs on a private cloud, can be dynamically deployed to public cloud if there is a spike in demand or if there is a failure of internal or private cloud [10].



**Figure 5 - Example of Hybrid Cloud [4]**

## Chapter 3: Evaluation Method and Criteria

In this chapter, I will be explaining about the evaluation method that I used to evaluate Cloud Computing platforms and the specific set of evaluation criteria, based on which, I decided to evaluate these platforms. This will provide more idea to the reader on how these platforms are evaluated.

### 3.1 Evaluation Method

As an evaluation method I decided to use a Java based application that I can deploy to Amazon Web Services, Windows Azure and Eucalyptus Cloud Computing platforms. These three platforms are providing Infrastructure-as-a-Service capabilities, which have support for creating virtual machines and use applications like application web server and a relational database, to run applications on them. Since Google App Engine is Platform-as-a-Service and has number of restrictions on their services, I will not be able to use my existing applications on it, so I will be creating a new application using Google App Engine's software development kit.

In my previous semester, I worked on a project called "Password Manager" that was based on Service Oriented Architecture and was implemented using Java Spring technology and MySQL relational database. Since it was implemented with SOA in mind, this is the perfect opportunity for me to test the application in the cloud with a proper service oriented set up. In the next section, I will be explaining the setup of my application that I would like to use on Amazon Web Services, Windows Azure and Eucalyptus.

#### 3.1.1 Setup

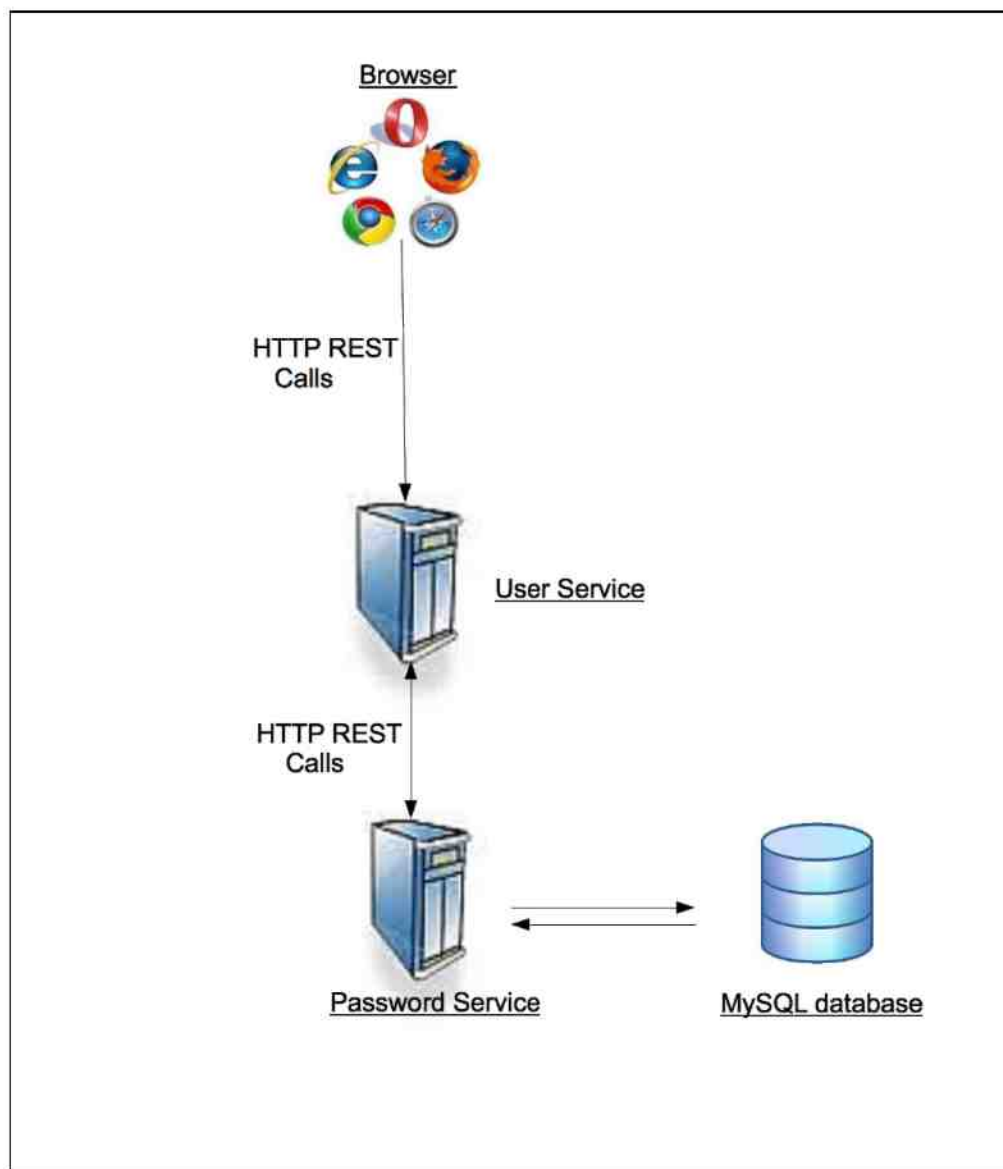
The application "Password Manager" consists of two services named "User Service" and "Password Service". "User Service" is responsible for the user interface and allows users to interact with the application. "Password Service" is the back-end service, which receives requests from the User Service and sends responses back to it. Password Service accesses the MySQL database where I store the user's data.

**Figure 6** depicts the Password Manager application's architecture. The browser is acting as a client of this application and it can be any browser running on any platform, such as a computer, a smart phone or a tablet. The user will be able to access the User Service with the browser, which is responsible for the user interface and allows users to interact with the application. User Service uses HTTP Rest calls, which support methods such as POST, GET, PUT and DELETE

in order to communicate with the Password Service. As figure 6 displays, Password Service is a back-end service and cannot be accessed directly by users. Password Service is responsible for all the data processing, which includes the creation of user, management of user passwords, and more. The Password Service uses JDBC connector in order to talk to MySQL database.

The following narrative is the ideal setup of the application that I wanted to implement. During this thesis, I was able to test the architecture of the application, which I had been considering. I tested this application on a single computer where both the services were running on different ports and built-in MySQL database that also resides on the same computer. I wanted to test these services as different applications deployed on their independent machines.

This thesis allowed me to meet that goal and test the application using different virtual machines and database servers provided by Cloud Computing platforms.



**Figure 6 - Password Manager Architecture**

## **3.2 Evaluation Criteria**

While using the evaluation method described above, I used the following evaluation criteria for the evaluation of the candidate Cloud Computing platforms. The evaluation criteria are as follows.

- **Ease of Use:** All the systems should be easy to use or the learning curve should be smaller so that users can start using the system faster.
- **Reliability:** Reliability is a very important criterion. The service should be reliable and trusted.
- **Scalability:** All the Cloud Computing platforms should provide easy, automatic or elastic scalability.
- **Security:** Businesses deploy their applications in the cloud, which they rely on. Security is the most important concern for them and cloud service providers should provide greater security measures for their services.
- **Economics:** Economics plays a very important role for the businesses in determining whether to move their applications to cloud or not. Cloud services should be competitive.
- **Fault Tolerance:** Another important criterion of evaluation is fault tolerance. Cloud services should have high fault tolerance and should take measures in case of failures.

## Chapter 4: Amazon Web Services

Amazon Web Services (also known as AWS) is the pioneer in Public Cloud development. There are a number of well-known online service providers that leverage AWS for their computing needs. The Initial offering from the Amazon was the cloud based Message Queuing service called Amazon Simple Queue Service (SQS). They eventually added services like Simple Storage Service (S3), Elastic Compute Cloud (EC2), and a flexible and distributed database service called SimpleDB. Amazon also provides MySQL and Oracle DB in Cloud through a service called Relational Data Service (RDS).

Amazon Web Services are categorized as Infrastructure-as-a-Service (or IaaS). It provides complete flexibility to its users. Users can select the operating system, application servers, and programming language of their choice. Amazon Web Services also provides standardized APIs for Java, Python, Ruby and Microsoft .NET. [12].

The Amazon Web Services provides great flexibility and less administration costs for reliable and scalable infrastructure to deploy web applications. AWS offers a variety of infrastructure services. The diagram below will introduce the AWS terminology and will help explain how the application can interact with different Amazon Web Services and how different services interact with each other [12].

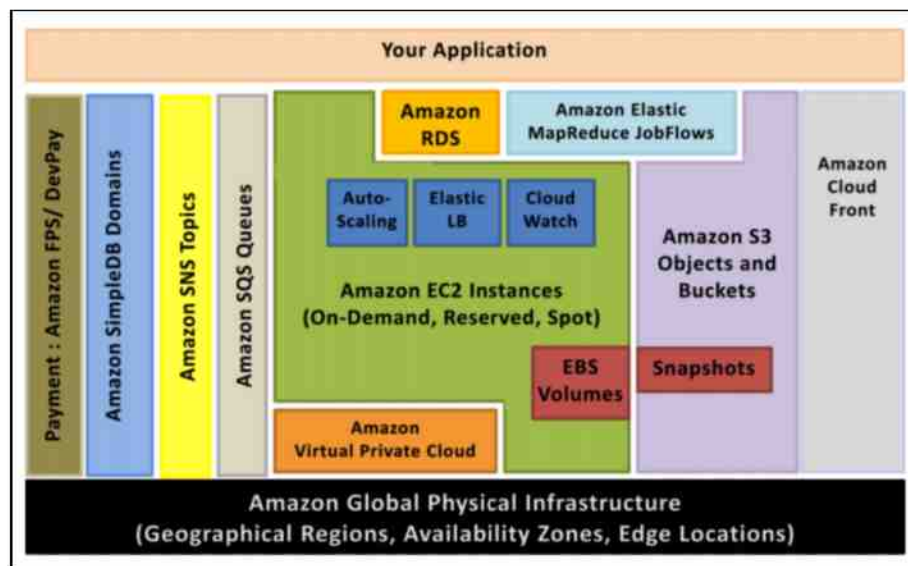


Figure 7 - Amazon Web Services [12]

## 4.1 Amazon Elastic Compute Cloud (EC2)

Amazon Elastic Compute Cloud (also known as Amazon EC2) is a service that provides a true virtual computing environment. Through Amazon Web Service Console, a user can launch a number of instances, manage network security for those instances, configure computing resources such as CPU capacity, storage capacity, RAM and bandwidth for those instances.

Amazon EC2 allows the user to use the Operating System desired as it is supporting different varieties of Linux such as Ubuntu, Red Hat, Suse, Amazon's trimmed down version of Linux and Windows Server 2008. The required operating system can be utilized and an instance can be launched. Amazon has different sizes of instances that range from micro to small to medium to large, which has a specific amount of computing resources configured for it. A predefined configuration can be used, followed by the instance, or selected resource configurations can be chosen. Since Amazon has a datacenter present in every major part of the world, instances can be launched in any part of the world based on the targeted audience of the application that supports them. Once launched, all the instances are associated with the account and, it remains in that condition until you shut them down or terminate them.

### 4.1.1 Features of Amazon EC2 Based on my Analysis

**Amazon Machine Image:** Another extensive functionality provided by Amazon EC2 is, user can create an image of the current environment with all applications and packages installed, which is known as Amazon Machine Image (AMI). Since it is the exact image of their environments, it can be used to launch multiple instances without requiring installation the same application again on all of them. This is a hugely beneficial utility if user has to install the same set of software or applications on multiple instances.

**Elasticity:** Because this service is elastic in nature, it can automatically scale the instances based on the load, if specified. It can launch additional instances if required and can shut them down as well if the load decreases on the system.

**Security:** By default, every Amazon EC2 instance is protected by a firewall and all communications to the instance are blocked. Amazon EC2 has the notion of the security group that needs to be associated with each instance when it is launched. You can specify access to certain ports on EC2 instance such as port 22 for SSH, 80 for HTTP, 3306 for MySQL. You can also manage the IP address ranges for clients who can access those allowed ports. This proves that it allows the user to control the instance with a highly granular level.

### 4.1.2 Types of Amazon EC2 Instances

There are three different kinds of Amazon instances available:

- **On-Demand Instances:** On-Demand instance allows user to use and pay for the computing resources per use per hour without hassle of long-term contracts.



- **Reserved Instances:** With reserved instances, user has an option to make one-time payment for each instance that is reserved and it gives you a hefty discount on the hourly charge for that instance. There are three reserved instance types possible, such as Light, Medium and Heavy Utilization Reserved instances.
- **Spot Instances:** Sometimes there are unused instances available; those instances are then converted to Spot Instances. “These instances allow customers to bid on unused Amazon EC2 instance capacity and run those instances for as long as their bid exceeds the current Spot Price” [13]. If the application has the flexibility of when to run, it can significantly reduce the EC2 costs.

## **4.2 Elastic Block Storage (EBS)**

Elastic Block Storage service provides the raw unformatted block storage to the user. User can create a file system on top of EBS, and format it with the desired format, and attached it to the EC2 instance. The charges for EBS volumes are based on the number of I/O requests and size of the volume.

### **4.2.1 Advantages**

- The storage capacity varies from 1 GB to 1 TB. So that users can use the space that is required.
- EBS comes with built-in redundancy; which prevents the failure of in case one single drive fails.
- It provides the ability to create a snapshot of drive, and store those snapshots to Amazon S3. User can use those snapshots to recreate a new volume or restore the volume in certain point-in-time. It is a gradual backup of the current volume and an excellent way to protect the data.

### **4.2.2 Disadvantages**

- Amazon has datacenters in different parts of the world. Availability zones identify those locations; certain services are zone specific only. Elastic Block Storage is one of them. It can only be attached to the EC2 instance that is present in the same availability zone as EBS.
- EBS is redundant but not as Amazon S3 as it is redundant across the availability zone, which is highly reliable in case of complete datacenter failure or natural calamities.

## **4.3 Amazon Virtual Private Cloud (VPC)**

Amazon Virtual Private Cloud is a service that creates a secure communication between any organizations IT infrastructure and the Amazon Web Services cloud. VPC allows enterprise

customers to use Amazon's cloud computing resources using Virtual Private Network (VPN) connection.

## **4.4 EC2 Cloud Watch**

The EC2 CloudWatch is a service that provides monitoring of resources within EC2. "It collects and store information about the performance (CPU load average, disk I/O rate, and network I/O rate) of each of your EC2 instances. The data is stored in to the Amazon Cloud for two weeks and can be retrieved for analysis or visualization" [13].

## **4.5 Amazon Simple Storage Service (Amazon S3)**

Amazon Simple Storage Service (or S3) is an implementation of storage infrastructure, which allows storage and retrieval of user data, using the Internet regardless of location of the user. Amazon S3 has revolutionized the data storage in the cloud [19]. It has given a new direction to the IT industry and provided the solution for universal data storage and data sharing. It adheres to the strategy of taking the data with user and accessing it when users want. It needs two pieces of information during data storage, first is the data itself and second is metadata that represents the information about data. Metadata helps S3, in identifying data type, size and more. It uses the bucket concept to store the data in S3.

Amazon S3 is capable of storing data in every format possible. The user can store documents, Amazon Virtual Machine Image, audio, video, serialized object, images and many more types of data on Amazon S3 [19]. These features make S3 an ideal candidate for storing a data in one place and accessing it anywhere using multiple clients.

### **4.5.1 Advantages**

- It is scalable; there is no limitation on how much space user can use. They do not have to decide it upfront. It can scale, as per your need, there is no configuration change required.
- Unlimited storage, pay what user utilize.
- It is available anywhere in the world that means user can access the data from anywhere.
- Inexpensive for the startups as no upfront infrastructure setup required.

### **4.5.2 Disadvantages [17]**

- It is not user friendly. User Interface is benign.
- Another key issue is trust. Not all businesses want to put their data in the cloud. Some businesses, which handle private and critical information, do not want to put their data on the public storage system.
- Availability is also an issue for Amazon S3 services. It suffered some serious outages in 2008.

## **4.6 Elastic MapReduce (EMR)**

Elastic MapReduce service allows users to launch multiple EC2 instances, which can run in parallel and perform, data processing on large-scale jobs. It uses Hadoop open-source framework based on MapReduce paradigm. MapReduce handles all the issues that arise when it is needed to launch, monitor, load and terminate hundreds of instances. Elastic MapReduce handles all kind of applications, which range from log file processing to gene sequencing.

Amazon Elastic MapReduce starts Hadoop implementation of the MapReduce framework on Amazon EC2. It divides the data into smaller sections so that, it can perform parallel procession on them (known as Map function), and eventually recombining all the processed data into the final solution (known as "reduce" function). Amazon S3 stores the data that needs to be processed as well as the final results.

## **4.7 Relational Database Service (RDS)**

Amazon RDS offers the relational database on the cloud. It is based on the popular MySQL database. RDS is useful when there is a need to move a traditional Line of Business application to the cloud and to maintain high fidelity with the existing systems. The advantage of RDS is that there is no need to install, configure, manage, and maintain the DB server. Routine operations like patching the server and backing up the databases are taken care of by RDS; the user can only focus on consuming the service. RDS is priced on the pay-as-you-go model, and there is no upfront investment required. It is accessible through REST and SOAP based API [14].

The Amazon RDS application is designed for users who are in need of a fully functional relational database and want to integrate their existing applications with it. RDS provides the capabilities of the MySQL or Oracle database running on AWS instance.

## **4.8 Amazon Elastic Beanstalk**

AWS Elastic Beanstalk allows users, to quickly deploy and manage the application in AWS Cloud. "User can simply upload the application and Elastic Beanstalk automatically handles the deployment details such as provisioning of capacity, load balancing, auto-scaling and application health monitoring" [15]. Along with all these functionalities, users can still have complete control over the AWS resources that are powering the application and can access the underlying resources.

Elastic Beanstalk takes advantages of Amazon Web Services such as, Amazon EC2, Amazon Simple Notification Service, Amazon S3, Elastic Load Balancing and Auto-Scaling to deliver the highly reliable, scalable, and cost-effective infrastructure.

Most existing platform-as-a-service providers, “while reducing the amount of programming required, significantly restricts the developer’s flexibility and control” [15]. Developers are forced to live with all the decisions and/or rules predetermined by the vendor. However, with Elastic Beanstalk service, the user retains full control over the AWS resources.

To allow the portability of the application, Elastic Beanstalk is built using a familiar software stack such as Apache HTTP Server for PHP or Apache Tomcat for Java.

## **4.9 Experiment**

Experimenting with the Cloud Computing Platform available for developing and deploying an application was the goal of this thesis, along with the study and analysis of different Cloud Computing Platforms.

### **4.9.1 Background**

As I mentioned in Chapter 3, I would like to deploy the Password Manager application using Amazon Web Services. This section will focus on different services that I used to deploy my application on Amazon Web Services platform.

### **4.9.2 Implementation**

To implement the architecture mentioned in the **Figure 7**, I have decided to use different services provided by Amazon Web Services. While evaluating Amazon Web Services, I found that Amazon Elastic Beanstalk service is the best match for deploying the application, as it supports the existing software stack, such as Apache HTTP Server for PHP and Apache Tomcat for Java. [15] As my application was built in Java and this service supports the Apache Tomcat application deployment server, this was the best match for me and this service can get my application running with a minimum set up. It also allows user to set environment variables, that you would like to send it to an application container, while loading the application such as JDBC URL for the database connection. So there is no need to update the code every time users want to change their database application.

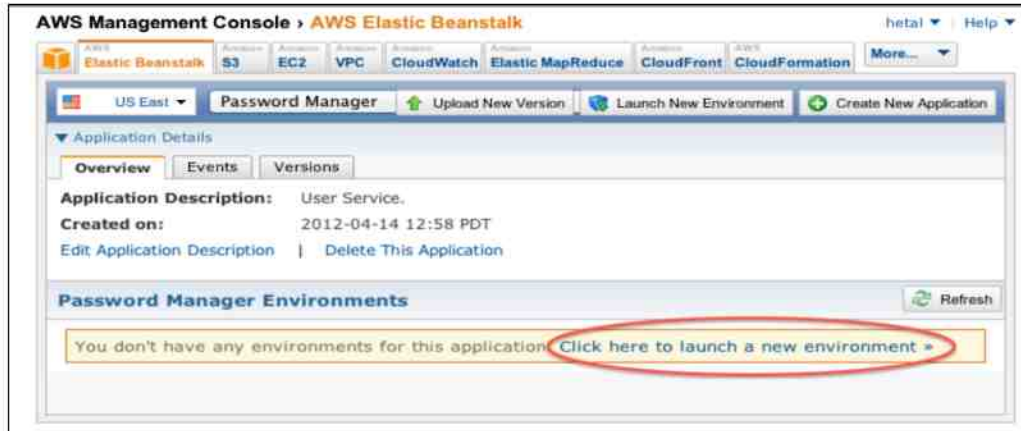


Figure 8 - Amazon Elastic Beanstalk Application Creation

### 4.9.3 Deploying an Application

In order to deploy the application in Elastic Beanstalk user have to go through the following steps:

- Create an application in Elastic Beanstalk. For multiple services in my case, I had to create 2 applications. I named them Password Manager and the Password Service.
- Users are required to provide URL for that application that they are creating. Those URLs can be used in order to use the application.
- Once users create an application, it allows them to upload the application. Password Manager is implemented in Java, so I have uploaded .war files for both services. Also, it allows user to select the container type in which user want to deploy the application. I have chosen Apache Tomcat for deploying the Java based application.
- Amazon Beanstalk uses Amazon EC2 instances in order to deploy the application.
- For security measures, Amazon EC2 provides Security Groups, which acts as a firewall for the application. Users can specify ports and IP addresses that they want to allow to talk to their application under Security Groups. They are required to add at-least one Security Group for the deployed instances.

Let's take a look at the figures that demonstrate the steps explained above.

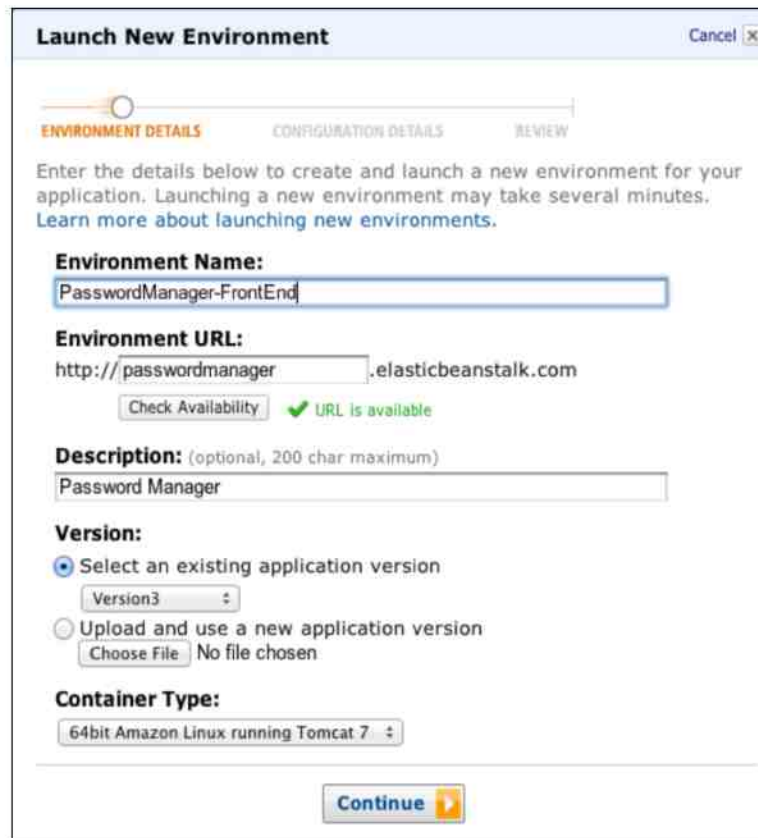
**Figure 8** depicts the Elastic Beanstalk, application details page on Amazon Web Services Console. User can see in the same image, that I have created an application named, Password Manager, and its description mentions the User Service that tells me that Password Service is actually User Service. Currently, the Password Manager application is not deployed to any instance, and because of that I am seeing a message with URL to Launch an Environment, which I highlighted with red oval.

Once users click that link to launch an environment, they are presented with a **Launch New Environment** dialog, which is depicted in **Figure 9**.

They are required to provide information for the new environment such as Environment Name, URL, Description, Version of an Application and Container Type. I have provided the necessary information, and the application will be accessed at the following URL: <http://passwordmanager.elasticbeanstalk.com>. Since my application was Java based, I selected Tomcat 7 as the container type for my application. I have also selected the version of my application, which is “version3” as mentioned in the **Figure 9**.

In order to deploy password Service I had to repeat the same process explained above. The URL for password service that I selected was <http://passwordmanager.elasticbeanstalk.com>. The User Service will have to know about this URL so that it can talk to the Password Service using the same on port 80 that is a default port for HTTP based requests.

Amazon Elastic Beanstalk eventually starts the Amazon EC2 instances. One or more Security Group is assigned to every EC2 instance. The security groups are responsible for granular level access to each EC2 instance. **Figure 10** displays the details of Security Groups.



The screenshot shows the 'Launch New Environment' dialog box with the following details:

- Environment Name:** PasswordManager-FrontEnd
- Environment URL:** http://passwordmanager.elasticbeanstalk.com
- Description:** Password Manager
- Version:** Version3
- Container Type:** 64bit Amazon Linux running Tomcat 7

A 'Continue' button is located at the bottom right of the dialog.

**Figure 9 · Launch New Environment for Application Dialog**

For my application, I had created a security group named “**elasticbeanstalk-default**”, as shown in the figure below. When I select the above-mentioned security group, it displays the details of that group below. Users can manage the incoming connections to their EC2 instances using **Inbound** tab, where they can specify the port range and a source IP address if they want to do granular restriction. On the right bottom of the Figure 12, user can see that I have allowed Port

22 (for SSH Connections), 80 (for HTTP Connections) and 3306 (for MySQL DB Access) for incoming connections. Source IP values 0.0.0.0/0 signifies that any device on the Internet can make a connection to your EC2 hosts using allowed ports. Users can provide specific IP addresses if they want to restrict the access to selected hosts/computers.

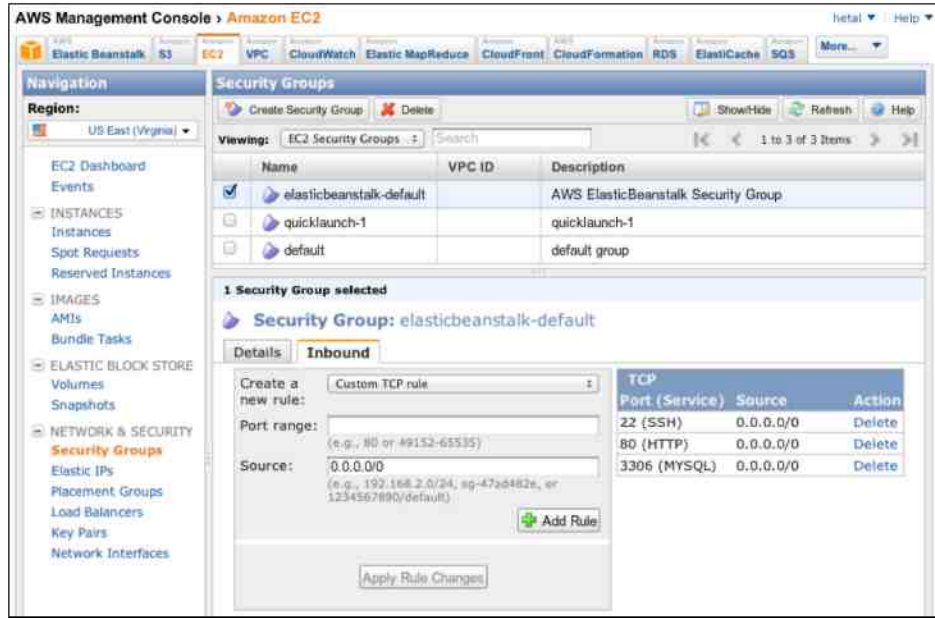


Figure 10 - Security Group Configuration

#### 4.9.4 Creating MySQL DB Instance in RDS

The second step was to deploy the application, by creating a relational database using AWS. Amazon Relational Database Service (RDS) allows you to create a MySQL and Oracle DB instances on demand. This service manages the maintenance of your database instance and also takes automatic backups of the database. Single RDS API call can help you scale the compute resources or storage capacity for your database instance(s) [16].

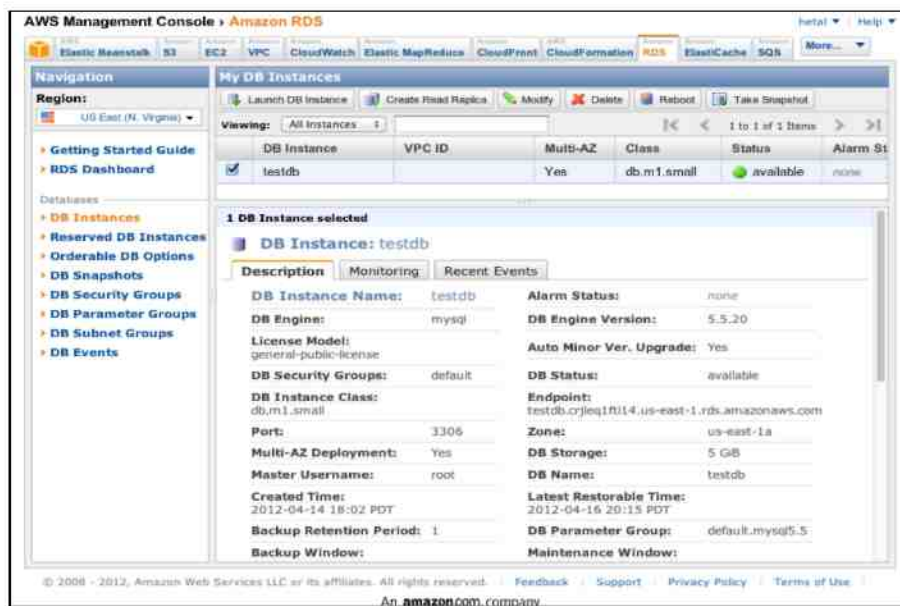
Like Amazon EC2 instances, RDS DB instances are also having the notion of Security Groups. By default, DB Security Groups do not allow any incoming or outgoing connections. In order to access the database, I had to add the same security group that I used for EC2 instances, along with that I also had to add CIDR/IP addresses so that DB instance can be accessed by EC2 instances on the same network. User can also specify an IP address that does not belong to the Amazon network.

The Password Service deployed on one of the EC2 instances in order to connect to the database use the following URL:

`"jdbc:mysql://testdb.crjleq1fti14.us-east-1.rds.amazonaws.com:3306/testdb?user=root&password=<password for root user>"`

**Figure 11** depicts the webpage to create Amazon RDS DB instance using Amazon Web Service Console. For “Password Manager” application, I created an instance name *testdb*. When user selects the DB Instance from My DB Instances List, the detailed description for DB instance shows up below. User can see all the different parameters configured for selected DB instance. They can see information about the DB as it is using MySQL DB engine with version 5.5.20. Also, the detailed information includes the zone where this instance is located, the amount of storage assigned to it, the port number to access the database, and most important of all, “Endpoint” value. The “Endpoint” value in the DB Instance info is used to connect to the database instance from the application. Once the instance is created user needs to use the following command to connect to the instance in order to create the database schema.

*“mysql -h testdb.crjleq1ft14.us-east-1.rds.amazonaws.com -P 3306 -u root -p”*



**Figure 11 - Amazon RDS DB Instance Information**

Upon executing the above command, users will be prompted for a password, given the correct password they can see the mysql command prompt. User can begin their schema creation process there.

**Figure 12** shows the configuration of the DB security group for “testdb” instance. User can add two different connection types under DB security group. Users can leverage EC2 security group that they have created for EC2 hosts, and they can explicitly provide IP addresses of the hosts in CIDR format that are allowed to communicate with DB instance. Here, user can see that I have selected ‘elasticbeanstalk-default’ EC2 security group as an allowed connection type and also provided CIDR: 10.0.0.0/8 so that all the EC2 instances in Amazon’s internal network can access this database instance with proper permission. I have also authorized a host outside of the Amazon network to access the database, which is identified as CIDR: 76.175.67.56/32.



#### 4.9.5 Application Execution

Besides deploying the different aspects of an application to Amazon Cloud, another important aspect of the application is the actual functionality of it. The goal of the Password Manager is to allow users to save their different sets of username and password combinations at one single place. This application also allows users to leverage the Twitter and Facebook authentication, so that they do not have to maintain one more set of username and password for this application.

After everything is completed, I can access the application using the Password Manager URL: <http://passwordmanager.elasticbeanstalk.com>. I can see the Sign Up Page of the Password Manager Application by clicking URL mentioned before.

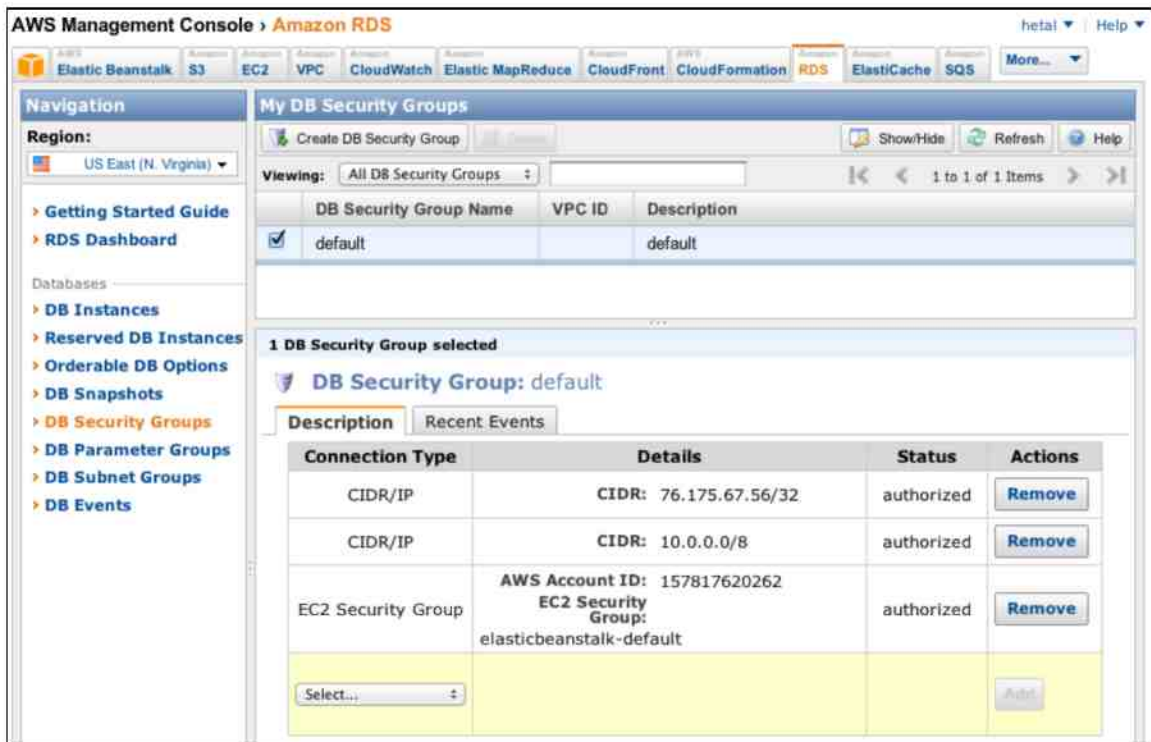
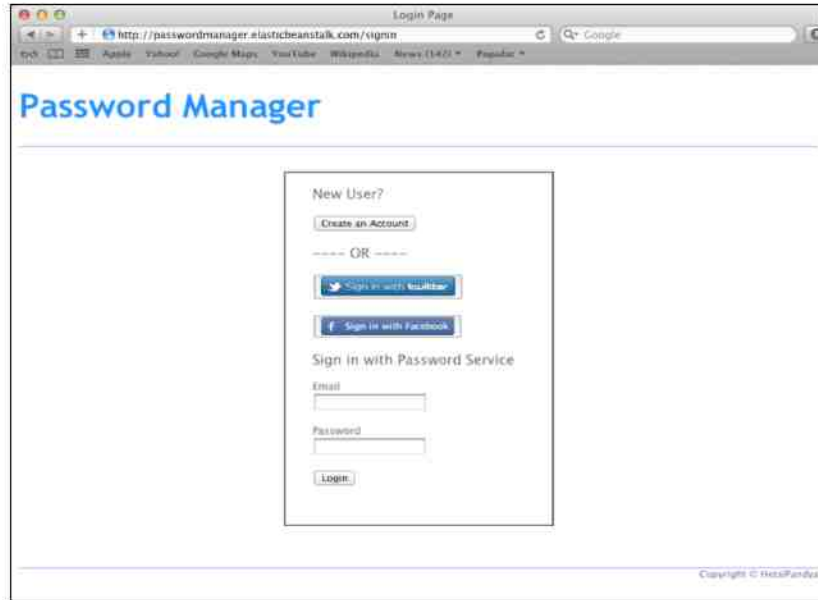


Figure 12 - Amazon RDS DB Security Groups Configurations

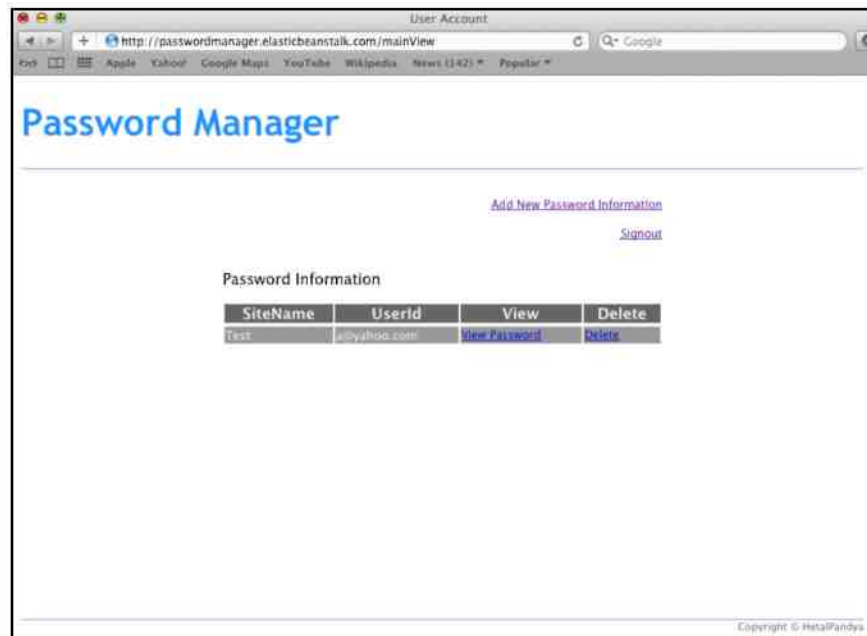
**Figure 13** shows the Sign Up/Login page for the application. I have knowingly made this page very simple. It provides three different ways of using the service. The user can click on “Create An Account” button to create an account with Password Manager, and use them with the Email/Password text boxes below to sign in to the service; or, they can use their existing Twitter or Facebook login information to authorize Password Manager, in order to allow users



**Figure 13 - Password Manager Application**

to participate in the service, utilizing Twitter or Facebook authentication. It uses OAuth protocol for that which is outside the scope of this thesis work.

Once the user is successfully logged in to the application, they are taken to the main page of the application. This page displays Password Information belongs to the logged in the user. Users can add new / update existing username/password information for the given site/application. They can also delete the entry from the database. User Service makes REST calls to the Password Service, to get the user specific data specified on the main page of an application.



**Figure 14 - Password Information Page on Password Manager**

**Figure 14** is the main page of the application that is being displayed to the user. This page has a tabular view, which contains all the data for that user. It includes Site Name, User Id and a way to view the password. I am not showing the password in clear text format right on this page, as it could be a security hazard. Users will have to click the “View Password” link to view the password related to that entry. They can delete the entry they want by clicking to delete link at the end of the entry.

Since the user is able to login to the system, and can see the Password Information, it is ensured that the application is deployed and functioning properly.

## **4.10 Analysis**

Amazon Web Services is the pioneer in the Cloud Computing world. They are the first who provided the complete suite of services that can be considered as Infrastructure-as-a-Service. Based on the predefined evaluation criteria, following is my analysis of the Amazon Web Service as a whole.

### **4.10.1 Ease of Use**

Easy to use and deploy the applications. It is an extremely helpful way to move the applications to cloud.

It provides complete access to the EC2 instances. Unlike other providers, user can access the hosts manually and perform a certain task or do the maintenance if required. User can perform operations on it, which is not provided by Web Services interface or Amazon Web Service Console.

### **4.10.2 Reliability**

Reliability is still an issue as there are sporadic occurrences of outages in the Amazon Cloud [18].

Not all businesses want to put their data in the cloud. Some businesses, which handle private and critical information, do not want to put their data on the public storage system.

### **4.10.3 Scalability**

It provides elastic scalability, as it is elastic in nature. It allows resources to be scaled automatically based on the load, if specified. It can launch additional instances if required and can shut them down if the load decreased on the system.

Amazon has datacenters in different parts of the world. Availability zones identify those

locations; certain services are zone specific only. Elastic Block Storage is one of them. It can only be attached to the EC2 instance that is present in the same availability zone as EBS.

#### **4.10.4 Security**

Higher level of security, as it allows users to control access to the host at the granular level using the port and IP address specification in security groups.

#### **4.10.5 Economics**

It is extremely helpful for the individual developers or companies in their inceptions, which do not have sufficient budgets to deploy or test their applications. They can take advantage of EC2 and their pay-per-use model.

With the pay-per-use model and automatic scalability of different instances, it is sometimes difficult to manage the expenses. You have to manage it carefully as these costs are per instance used and for storage; it is based on the data stored and read/write of data.

#### **4.10.6 Fault Tolerance**

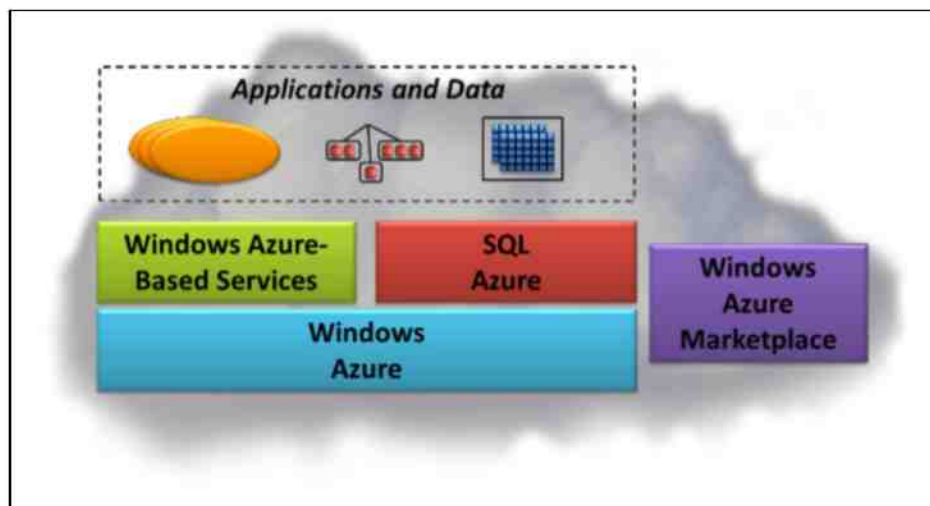
It is better to have multiple instances of your application running for redundancy as if you have only one instance running and somehow there is an outage, it will take time to get other instance(s) up and running.

## Chapter 5: Microsoft Windows Azure Platform

In 2008 during the Professional Developer's Conference, Microsoft announced that it is entering into the cloud services arena with the Windows Azure platform. Microsoft Online services, known as Business Productivity Online Suite, have been in use for a few years now. The Windows Azure platform is attempting to create an end-to-end cloud service offering in different categories such as the platform, middleware, enterprise services, and consumer services [20].

### 5.1 Windows Azure Platform Architecture

The Windows Azure platforms have four parts. **Figure 15** shows all 4 components below [21].



**Figure 15 - Microsoft Windows Azure Platform [21]**

**Windows Azure:** Microsoft Azure is a Windows environment responsible for executing applications and storing the data in computers in Microsoft data centers.

**SQL Azure:** SQL Azure provides relational database services in the cloud and it is based on SQL Server.

**Windows Azure-based Services:** These services provide Cloud infrastructure for running the applications in the cloud or locally.

**Windows Azure Marketplace:** It is an online marketplace for purchasing the cloud based application and data.

All four components reside in Microsoft data centers around the world. Users have control over where they want to deploy their applications. It gives them the ability to place their application close to their users.

## 5.2 Windows Azure

Windows Azure is easy to understand at a high level. It is responsible for running Windows applications and stores related data in the cloud. **Figure 16** shows its components.

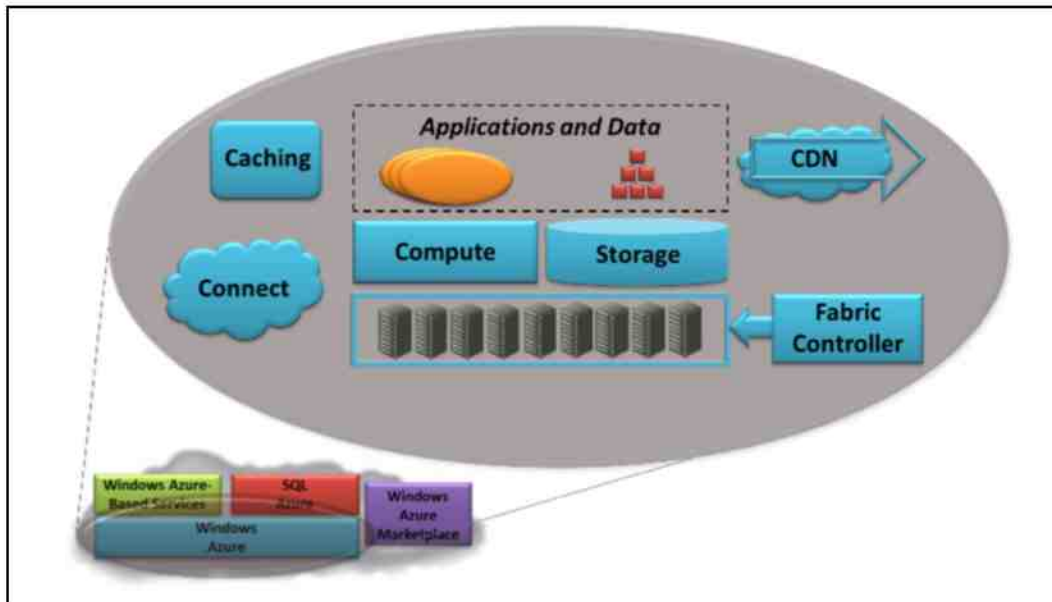


Figure 16 - Windows Azure provides Compute and Storage Services in the cloud [21]

### 5.2.1 Compute

Compute service runs the user developed applications on the Windows Server foundation. It supports an application prepared to use various frameworks, such as NET framework with languages such as C# and Visual Basic or languages such as C++ and Java. These applications can be developed using Visual Studio or any other development tools. Use of Microsoft technologies such as ASP.NET, Windows Communication Foundation (WCF) for free.

### **5.2.2 Storage**

Storage service stores the data in binary large objects (blob) format. It provides queues for interaction between different components of the Windows Azure application.

Data in storage service is accessible to both Windows Azure applications and on-premise applications. It supports HTTP Rest protocol for storing and retrieving data.

### **5.2.3 Fabric Controller**

Figure 18 displays that Windows Azure runs on a large number of machines. The fabric controller is responsible for keeping these machines together in the Windows Azure data center. Compute and Storage services are implemented on top of this large pool of resources.

### **5.2.4 Caching**

Most of the applications tend to access the same data again and again. It is cumbersome for service to keep hitting the database for the same data. One way to improve this situation is to cache the frequently accessed data. The Caching service provides this and caches the data, boosting the performance of an application running in the Windows Azure platform.

### **5.2.5 Connect**

Windows Azure Connect application allows the Windows Azure application to access on-premise database.

### **5.2.6 Content Delivery Network (CDN)**

Windows Azure Content Delivery Network service caches the frequently accessed blob data and maintains the cached copies of that objects at the sites around the world.

Instead of procuring and maintaining their own system, a user can rely on the cloud provider for those services. The customer only needs to pay for the computing power and storage they have used.

## **5.3 SQL Azure**

When we run applications in the cloud, most of the time there is a need for storing the data in relational database schemas. SQL Azure provides the solution to this problem by offering cloud-based service for relational data. **Figure 17** shows three components of SQL Azure.

The components of SQL Azure are as follows:

### 5.3.1 SQL Azure Database

SQL Azure Database is a solution that provides a cloud-based relational database. This service allows remote and cloud applications to store relational data on servers in the datacenters. It also follows the pay-per-use model. Users only have to pay for data storage and data I/O.

### 5.3.2 SQL Azure Reporting

It is an SQL Service Reporting Services running in the cloud. It is used with SQL database and it can create and publish reports in the cloud data.

### 5.3.3 SQL Azure Data Sync

As the name suggests, SQL Azure Data sync synchronizes the data between the database in the cloud and database installed remotely on-premises. It also allows data synchronization between SQL Azure databases, located across different data centers.

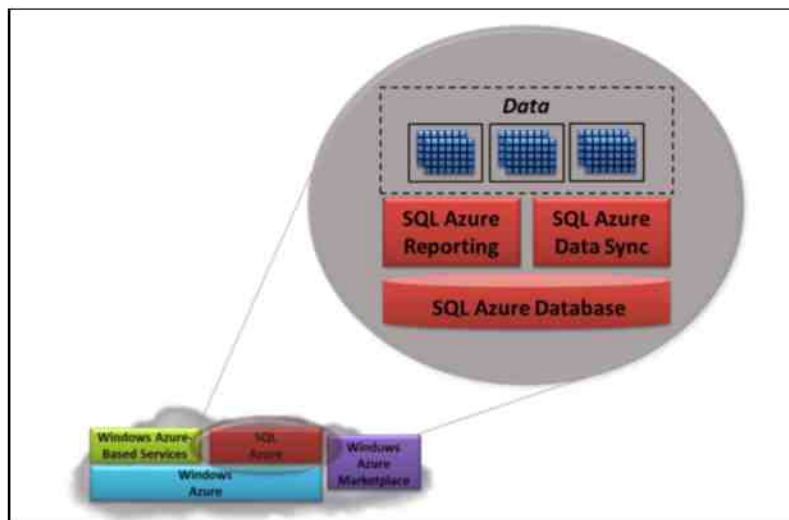


Figure 17 - SQL Azure provides Relational DB Services in Cloud [21]

It is developed using Microsoft SQL Server and provides similar operations as local RDBMS, such as use of stored procedures, defining views, defining triggers and more. There is a seamless integration between the database in the cloud and on-premises. Applications that use a local database will work as is with the database in the cloud.

Using the RDBMS with SQL Azure reduces the management requirement drastically. Instead of worrying about monitoring, and maintenance of the database, the SQL Azure user can just use the database and focus on their application and its data.

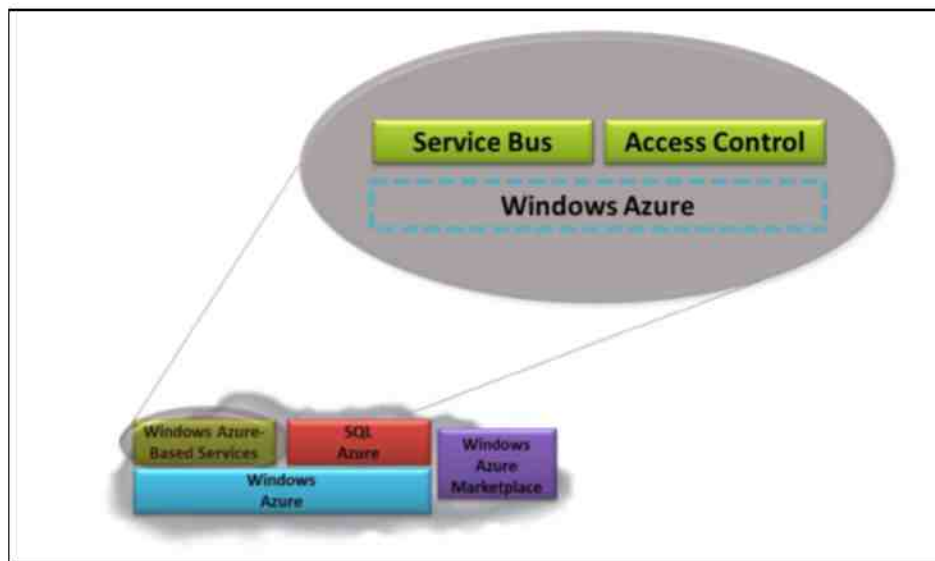


## 5.4 Windows Azure-Based Services

Executing applications and storing data in the cloud are fundamental aspects of Cloud Computing. It is possible to provide cloud infrastructure services to the application that runs locally on premises. There are two essential services provided by Windows Azure platform, and it is depicted in **Figure 18**.

### 5.4.1 Service Bus

Service Bus is responsible for exposing an application's services on the Internet by allowing applications to expose their endpoints or access points in the cloud. These endpoints are accessible by other applications. These applications may belong on-premises or in the cloud. Each endpoint is assigned with URI, which is used to access the service. Service Bus is responsible for translating the network addresses and getting through the firewalls.



**Figure 18 - Windows based service provide infrastructure that can be used by both cloud and on-premises applications [21]**

### 5.4.2 Access Control

Access Control service provides support for all the different ways of use of digital identity and a way to authenticate them. Be it Active Directory or Windows Live ID or Google Account or Facebook account, Access Control has built in support for all of them. There is a single place where application admin can define the rules, which control access to the service based on those rules.

## 5.5 Experiment

All of my earlier experiments with other Cloud service providers include interaction with Unix based systems. I was a little hesitant to use Windows Azure service to deploy my “Password

Manager” application, as I do not have much experience with running Java based applications on the Windows environment. By the time I start working on my experiment, I learned that Windows Azure has started supporting the VM creations using different Linux flavors such as, CentOS, Suse, Ubuntu and OpenSuse along with several Windows based VMs. That boosted my interest in using a Windows Azure Cloud platform for my test application. After going through some process of registration and waiting for their approval to use the “Preview” portal on Windows Azure, I was able to deploy my application successfully.

### 5.5.1 Background

My goal was to deploy the “Password Manager” application, which I created during my course work in previous semesters. I have used Amazon Web Services for deploying the same and now I wanted to use Windows Azure platform to deploy it.

### 5.5.2 Setup

Preview portal of Windows Azure makes creation of a Virtual Machine, Cloud Services and Virtual Databases extremely easy and user friendly. **Figure 19** depicts the elegant layout of the Windows Azure portal. As you can see I created 2 virtual machine instances and a storage account for block storage.

Along with the status of the service, it also provides information about the type of subscription for that service and location of the datacenter where those services are deployed.

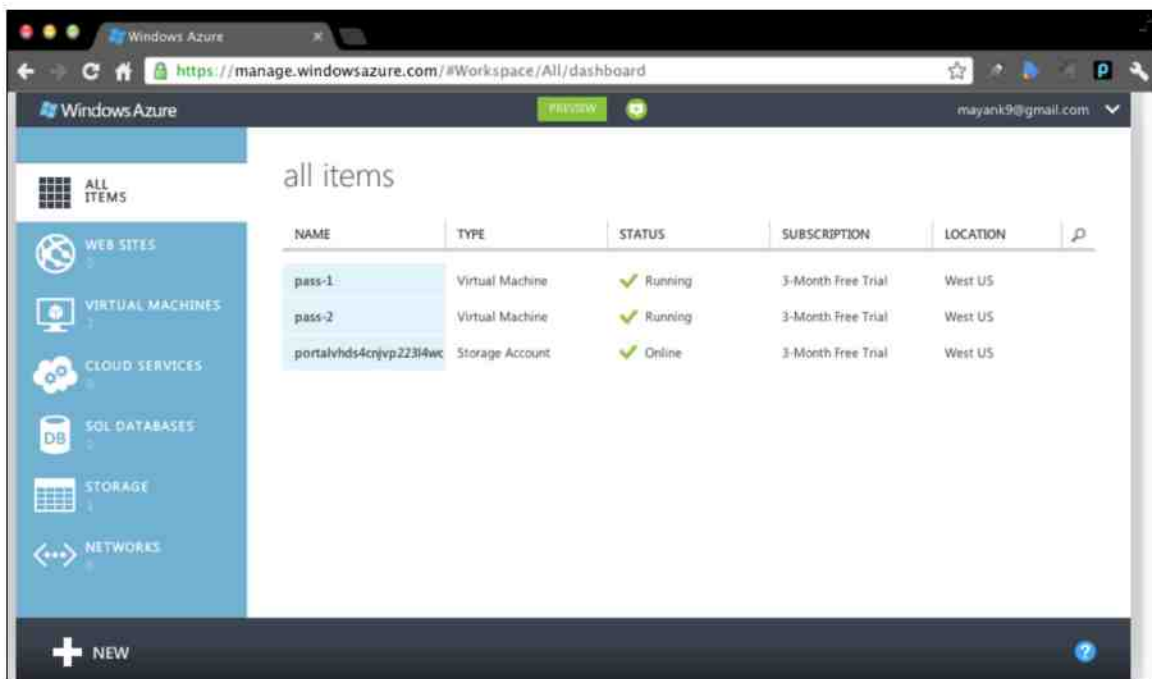


Figure 19 - Management Console of Windows Azure

I created two virtual machines pass-1 and pass-2 to deploy User Service and Password Service on them. Password Manager application contains these two services. Once I was done with creating the instances, I had to install the necessary applications to deploy Java based application. So I installed Java Runtime Environment, and Apache Tomcat Web Server for Java applications.

User Service is responsible for the user interface implementation of the Password Manager and Password Service is responsible for handling business logic, storing and retrieving data from the database. These two services communicate with each other using HTTP REST protocol. I am using MySQL database for my application. SQL Databases item in Windows Azure portal only creates Microsoft SQL servers, so I decided to install MySQL server locally on pass-2 virtual machine and run it along with the Password Service.



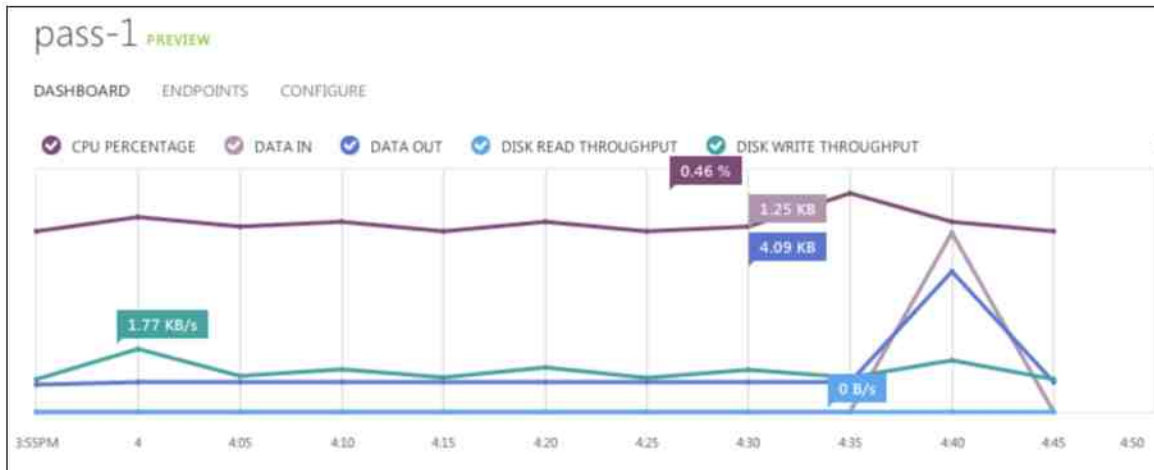
**Figure 20 - Endpoints Configuration on Windows Azure Virtual Machines**

Similar to the concept of security groups in Amazon EC2, Windows Azure has the concept called Endpoints. With Endpoints, user can specify different rules that can allow communication on certain port for certain protocol. We can also map public port with private port, so that it is hidden from the end user on which port the service is being executed.

In **figure 20**, I have allowed port 22, 3306 and 80 for SSH, MySQL and HTTP requests respectively on pass-2 VM. I have mapped public port 80 to internal port 8080, Apache Tomcat uses it to execute Password Service. These services are not using load balancers as of now. However, we can use them if we want to deploy multiple instances of the same service for scalability and reliability of service in applications, which require high availability.

**Figure 21** represents the Dashboard where it represents all the information about the running virtual machine instance. It contains configuration information about the host along with how to access the host and many more details that are not displayed here. I wanted to show the graphs of different parameters, which can be used to identify the performance of the system on

Windows Azure Cloud. Here we can see that it represents the CPU percentage, Data In, Data Out, Disk Read Throughput and Disk Write Throughput. This information gives an idea to the admin/user about the performance of the virtual machine.



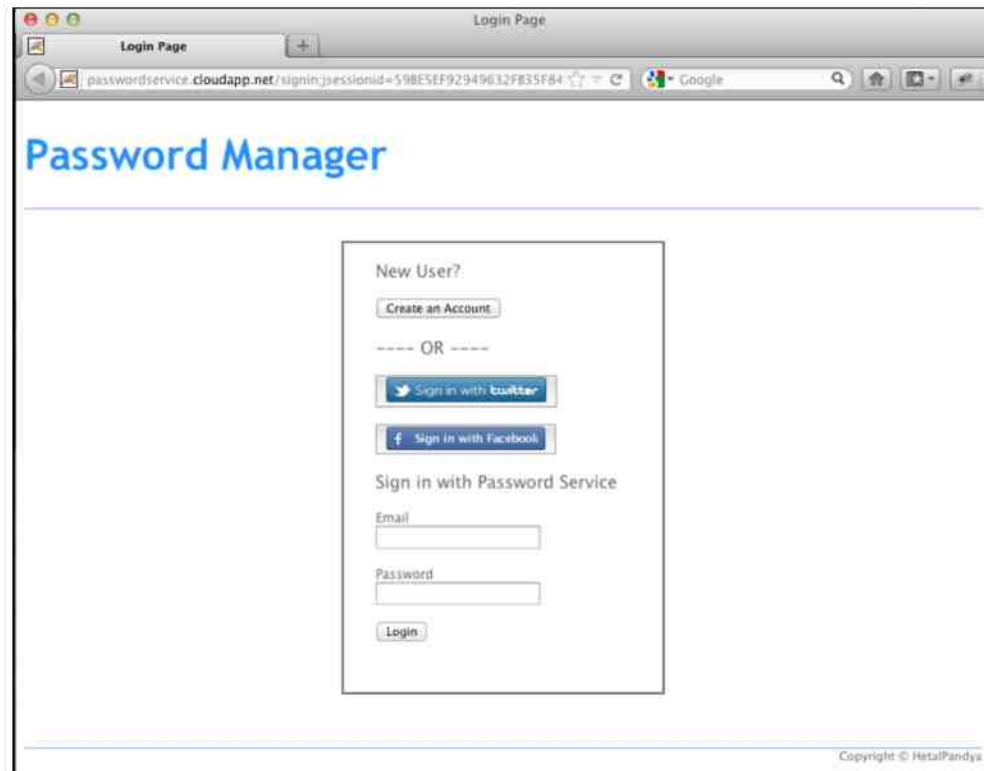
**Figure 21 - Dashboard represents different graphs for a Virtual Machine in Windows Azure Portal**

**Figure 22** also available on the Dashboard; it shows the Usage Overview of CPU and Hard Disk used by virtual machine. We can notice that Pass-1 VM is using 1 Core of the CPU right now and a disk is associated with that VM.



**Figure 22 – The Usage Overview of resources on Windows Azure Portal**

Once the virtual machine is launched and running, admin can login to the host using assigned username and password that was created at the time of VM creation. Then, I set up all the requirements necessary to run my application. Once that was completed, I installed the application, which is accessible at “<http://passwordservice.cloudapp.net/signin>”. Upon clicking this link user will be taken to the login page of Password Manager application that is displayed in **Figure 23**.



**Figure 23 - Password Manager Login Page on Windows Azure Cloud**

## 5.6 Analysis

Microsoft has emerged as a very big provider with different services in the cloud. Microsoft has started the Azure as Platform-as-a-Service, where they were offering services in the cloud. In Spring 2012 Microsoft has declared that they are now supporting virtual machines which can run selected Linux Images along with Windows Server 2008 and Windows Server 2012. With this release Microsoft has entered Infrastructure-as-a-Service space where now they can attract more developers from Amazon Web Services. Based on the predefined evaluation criteria, I have done analysis of Windows Azure platform, which is provided as follows.

### **5.6.1 Ease of Use**

Fabric Controller manages all the applications deployed in the Windows Azure Cloud. It is responsible for decision making on where to run it, monitor the application, handle the crash of an instance, and recover it with starting the new instance. It also manages timely updates to instances. Everything is handled by the fabric controller, which makes an application developer's life easy [24].

With PaaS and IaaS offerings, Windows Azure takes care of all the responsibilities of maintaining, provisioning, and hosting the cloud infrastructure. Users can just concentrate on development.

### **5.6.2 Reliability**

It is using fabric controller, which is responsible of handling the crash of an instance, which recovers with starting a new instance. Microsoft has datacenters across the world, and it also replicates the data throughout all the datacenters for redundancy. So in case of datacenter failure, the data is readily available from other datacenters and users can start their applications on it.

Windows Azure is the only provider of .NET framework based solutions. The user is locked in to the service with the vendor. There is no portability available. This seems to me as a reliability concern.

### **5.6.3 Scalability**

Windows Azure cloud allows users to integrate their Office Apps with the Apps in the Cloud. Users can easily switch from using the app in the local machine to the cloud and continue working on the document, spreadsheets or presentations.

Windows Azure has a Connect feature that allows application running in to the cloud to use resources, which is located on-premises. For example, users can use local SQL database with the application running into cloud, to store sensitive data. This feature is very important and enables businesses that deal with the customer's sensitive data to use Windows Azure cloud as they can leverage the flexibility of the cloud, but can still store the data locally [23].

Researchers are using Windows Azure platform to migrate eScience desktop applications to the cloud with certain modification in the process. It provides a great opportunity for the heavy-duty science processes to use the cloud platform to perform those operations easily compared to doing it on desktop computers [22].

Users have been waiting for Windows Azure to support Linux based VMs on its cloud. In Spring 2012, they have updated Windows Azure, which now allows users to create Linux VMs.

#### **5.6.4 Security**

Data security is a big concern in the cloud. There is no indication if the cloud data is accessed by someone, or deleted for any proper reason.

Since data stored in Cloud is replicated to the Microsoft datacenters, throughout the globe for redundancy, there are some concerns over storing the Government data out of the country. Currently, Microsoft does not provide the option to store data only at one location [25].

#### **5.6.5 Economics**

Windows Azure pricing for Cloud resource has become very competitive and with the introduction of virtual machine support, the prices really gone down and are matching the pricing of Amazon EC2 services. With lower prices and tight integration with other services provided by Microsoft, it becomes preferable choice for users who develop using Microsoft technologies.

#### **5.6.6 Fault Tolerance**

It has suffered some outages in recent years. Windows Azure provides the fault detection and it fails over the system in case of application, datacenter or database failures.

## **Chapter 6: Google App Engine Platform**

Google App Engine (also known as AppEngine or just GAE) is a Platform-as-a-Service (PaaS), Cloud Computing platform provided by Google. Google App Engine allows users to run web applications on Google's highly scalable, highly available and reliable datacenters [26].

“App Engine applications are easy to build, maintain and scale as the application gets more traffic and need for more storage increases” [26]. Since App Engine provides PaaS services, users should not be concerned about infrastructure maintenance. Like other PaaS providers, App Engine provides SDKs to develop, test and deploy the application on their servers. It makes things easy for the application developers.

Applications in App Engine are running in a secure environment where an application has limited access to the underlying operating system. Because of this limitation, App Engine can distribute web requests across multiple servers and scale the servers, to cope up with the traffic demands. The sandbox isolates an application in its own secure, reliable environment that is independent of the underlying computing resources such as OS, web server location, and computing hardware [26] [27].

### **6.1 Features**

App Engine makes it a breeze to develop an application that can execute reliably, under the heavy load and with a large amount of data. It includes the following features [26]:

- Supports persistent storage.
- Allows web applications to serve dynamic requests and provides complete support for common web technologies, such as Java, Scripting Languages and Python.
- Provides SDK (Software Development Kit) that allows users to develop the application in the environment that simulates GAE on user's computer.
- Supports Queue mechanism to provide asynchronous processing.
- Also provides scheduled tasks for triggering events at specific times and regular intervals based on application requirements.
- Supports three environments: Go environment, Java environment and Python environment.



## 6.2 Building Blocks

Google App Engine comprises different building blocks. It needs a different set of services to support all the features mentioned above. Some of the important architectural components of the Google App Engine are as follows:

### 6.2.1 Building Blocks

App Engine supports three different run time environments:

- **Go RunTime Environment:** Go runtime environment allows users to develop and deploy the web application's writer using Go programming language.
- **Java RunTime Environment:** User can develop application for Java runtime environment using common Java development tools and APIs. It can interact with environment using Java Servlet and JSP technologies.
- **Python RunTime Environment:** Python RunTime environment allows the user to implement an application using the Python programming language, and run it. Lots of feature rich APIs and tools are provided by App Engine for Python web application development [26].

### 6.2.2 Data Storage

App Engine provides a range of options for storing the application data.

- **App Engine Datastore:** App Engine provides a distributed NoSQL data storage service that features query engine and transactions. Distributed datastore grows with the data, just like the distributed web server grows with the application traffic [26].
- **Google Cloud SQL:** Google Cloud SQL is a service provided by Google that allows users to create, configure, and use relational databases to use with App Engine applications. Provider manages and administers the service. It provides functionalities of the MySQL database, which allows user portability by easily moving data, application and services in and out of the cloud. This service is in a limited preview currently [29].
- **Google Cloud Storage:** Google Cloud Storage is a service for storing and accessing user data on Google's infrastructure. It uses HTTP Rest protocol for communication. Features of this service include, high performance, scalability, advanced security and sharing capabilities [30].

### 6.2.3 Google Accounts

App Engine supports Google Accounts integration with an application for user authentication. The application can allow a user to sign in with Google account.

With the help of Users API, application can identify if the current user is a registered admin for the app. It allows admin to view/implement an admin-only area of application.

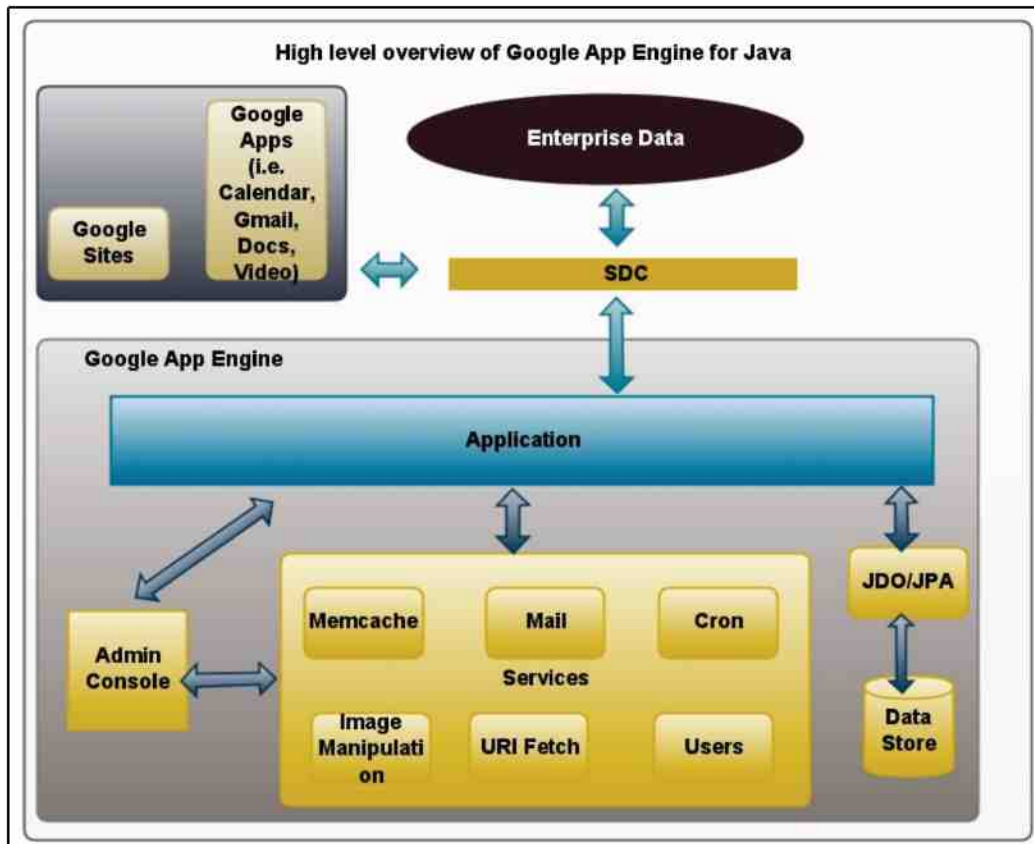


Figure 24 - Overview of Google App Engine for Java [31]

### 6.2.4 App Engine Services

App Engine provides a variety of services that enables the user to perform common operations when managing the application [26]. Available APIs are explained below.

- **URL Fetch:** With URL Fetch service, an application can access the resources on the Internet.
- **Mail:** Mail service can be used to send an email for an application. It uses Google infrastructure to send email messages.
- **Memcache:** Memcache is a service that allows an application to store that data in cached with a key-value pair. It is accessible by multiple instances of an application. It is

useful to store non-transactional data or data which is copied from the datastore for faster availability.

- **Image Manipulation:** Application can use the Image manipulation service to perform resize, rotate, crop and flip operation on images. It supports JPEG and PNG images.
- **Task Scheduling:** Allows the user to configure regularly scheduled tasks that operate at defined times or regular intervals [31].

**Figure 24** depicts how above-mentioned building blocks comprise the Google App Engine and how it is being used for applications built for Java RunTime environment.

## 6.3 Experiment

The easiest way to learn about the technology and services would be to create an application using the tools provided by the service. Since Google App Engine is providing platform-as-a-service, they control the environment in which users develop an application. They provide the tools to develop the application and also an environment where we can deploy the application.

### 6.3.1 Background

For Google App Engine platform (GAE), I create a very simple Java based application where users can store all of their passwords in one location and also can see them whenever they want. For this application, user doesn't have to create another set of username and password, as a user can use their Google user credentials with this application. My goal was to learn about some of the services Google App Engine provides for building an application. I ended up using UserService and DatastoreService in my application.

### 6.3.2 Setup

In order to get started with the application building, we need App Engine Java SDK provided by Google. It provides all the necessary APIs (Application Programming Indexes) to use different services. Next, I created an application using the Google App Engine portal. **Figure 25** shows the process of creating an application using the GAE portal. As per **Figure 26**, you can see that I have named my application "one-pass", which signifies one place to store all your passwords. I am using the domain provided by 'appspot.com', so my application is accessible at "one-pass.appspot.com". Here "one-pass" is also an application identifier, using which Google App Engine will identify my application and relate it to my Google account. It also provides different authentication options, which I can use as the authentication mechanism with my application, such as Google Account validation, Accounts with the Google Apps domain, or using id from OpenID provider.

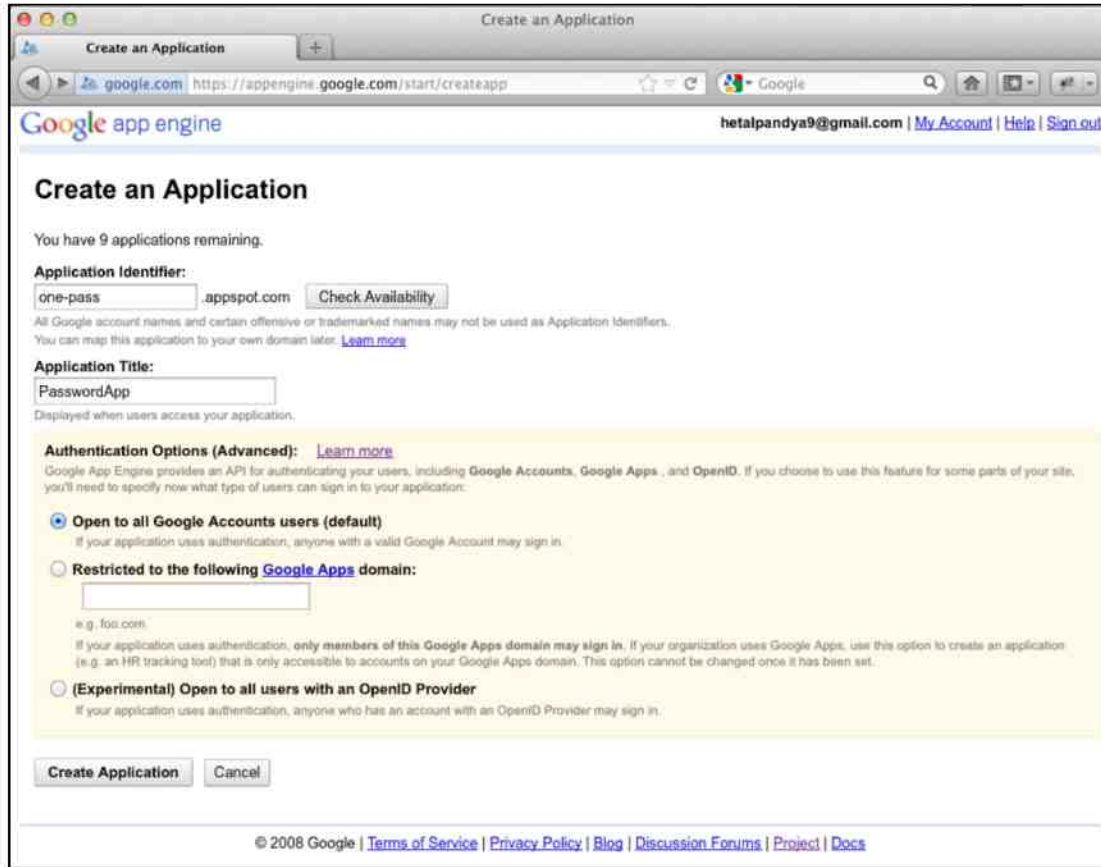


Figure 25 - Create Application using Google App Engine Portal

Figure 26 depicts the application I created based on the above flow. It mentions that I can create nine more different applications, as Google allows free users to create ten applications.

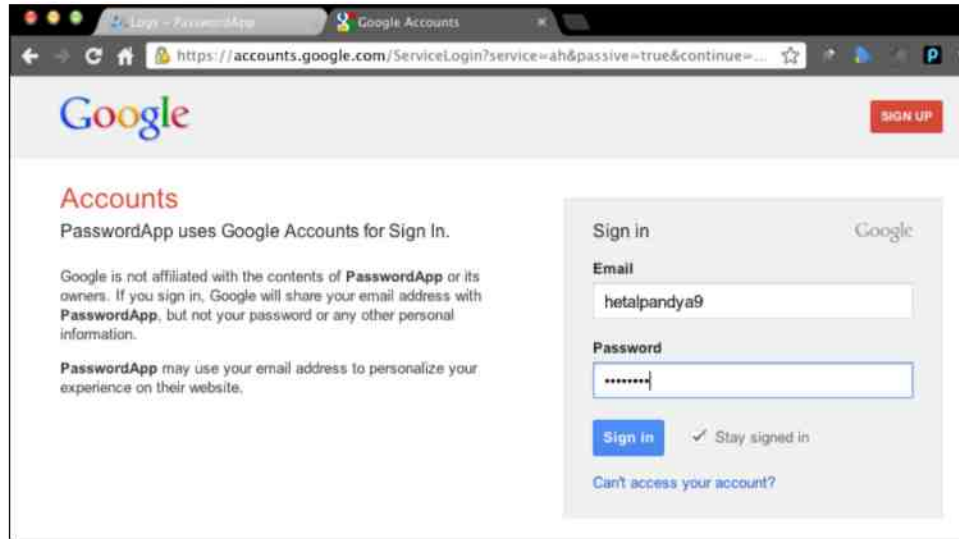


Figure 26 - My Applications Page in Google App Engine Portal

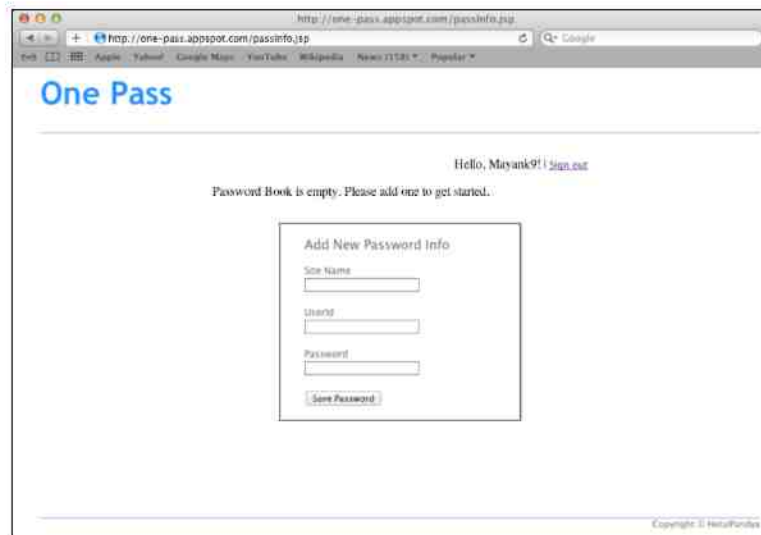
I used IDEA IntelliJ IDE and Google App Engine SDK to develop this application. SDK provides an environment like a sandbox, which implements the Google PaaS like functionality locally, where we can develop and test the application before deploying it to Google PaaS cloud. IntelliJ IDE has an integrated way of deploying an application to Google cloud. The

section below will explain more about the application in the cloud and different application monitoring.

**Figure 27** depicts that I am using Google Account authentication for this application. Whenever a user tries to login to my application, it verifies if that user was already logged in. If the user is not logged in, the application redirects the user to the Google Accounts login page.



**Figure 27 - Password App authentication using Google Accounts**



**Figure 28 - One Pass application on App Engine Cloud – First time user**

After logging in with Google Account credentials, the user gets redirected to my application. Currently, this application only contains one webpage developed in JSP. Users will be presented with the page displayed using **Figure 28**. As they are logged in for the first time, they will be asked to add passwords they want to store using this application.

Once the user starts adding new passwords, all the stored passwords will show up on the page in the table above “Add New Password Info” form. As you can see in **Figure 29**, all the passwords are displayed for that user. These passwords are stored using DatastoreService provided by Google App Engine; it uses no-SQL database for storing this as JPA entities.



**Figure 29 - Password Information for returning user**

### 6.3.3 Performance Measures

**Figure 30** shows different parameters for certain requests in my “one-pass” application. It shows the load on the system at a given time. As we can see that it shows values for parameters like requests/minute, average latency when it loads certain latency. For my application, “/passInfo.jsp”, “/add” and “/stylesheets/main.css” are most important requests.

Current Load ?				
URI	Req/Min current	Requests last 23 hrs	Runtime MCycles last hr	Avg Latency last hr
/favicon.ico	1.3	84	13	88 ms
/guestbook.jsp	0.3	71	58	124 ms
/passInfo.jsp	2.8	43	45	285 ms
/	0.3	9	385	598 ms
/add	0.0	8	85	155 ms
/stylesheets/main.css	0.0	5	0	41 ms
/passwordInfo/	0.0	1		

Figure 30 - Load statistics for Password App

Google App Engine has a concept of Front End Server, which is responsible for running the application and making the requests to fetch the data from distributed servers and highly scalable and replicated databases. Google App Engine provides three different configurations for Front End Servers. **Table 1** shows that configuration. You must have noticed that Front End Server names are F1, F2 and F4. By looking at the related configurations, we can see that CPU Speed and RAM for F2 is twice times of that of F1 and so on. So if we want to use the instance of F2 or F4 then the charges would simply be twice or four times that of the F1 instance.

Front End Server	CPU Speed	RAM Size
F1	600 MHz	128 MB
F2	1200 MHz	256 MB
F4	2400 MHz	512 MB

Table 1 - CPU &amp; RAM sizes provided by Google App Engine

## 6.4 Analysis

Google App Engine has been around for many years now, and it has been very successful with the users who are associated with startups or individual developers who cannot afford procuring servers for their applications. Following is my analysis based on the predefined evaluation criteria.

### 6.4.1 Ease of Use

Individual Users or startups can start developing their applications using Google App Engine SDK almost immediately. It allows users to create maximum ten applications associated with their App Engine accounts.

Users are not responsible for maintaining the servers and other resources, as Google handles everything. Users only have to develop and deploy applications.

Google App Engine allows users to deploy the application using plugins like Google App plugin for Eclipse and some other IDEs. It is a very straightforward process. As I was deploying my application for the first time, it took me no longer than ten minutes to understand the process of deployment.

#### **6.4.2 Reliability**

Reliability is a concern for Google App Engine as it has seen number of outages. Lot of time it is user error.

Google App Engine does not allow users to access the underlying computing resources. An application developer is unaware of the location where it is deployed. They only need to deploy their application using AppEngine portal; Google controls everything else. This is a big limitation over other providers, where they provide complete control of the environment to users.

Google provides number of services such as queue support, memcache, mail, URL fetch and many more, which is specifically provided by the Google App Engine using its SDK. It restricts the users from migrating their applications to platforms other than Google App. If applications are developed with no relational database, which was Google's original offering for persistent storage, it was almost impossible to migrate the database. Now, however, they have added the relational database support that can be used with Google App Engine

#### **6.4.3 Scalability**

Google App Engine is a highly scalable system. It scales the application automatically, based on the load on the application. This system is highly available as there are very few incidents of its outage.

Google App Engine recently started supporting relational database such as MySQL and Oracle using Google Cloud Database service, which is tightly integrated with Google App Engine. Developers were waiting for this feature for a long time. It makes the migration of the data from Google Infrastructure to anywhere else easier.

Google App Engine allows users to integrate different service provided by Google with their own applications. Users can integrate Google services such as YouTube, Gmail, Blogger and many more with their applications.

#### **6.4.4 Security**

It allows applications to use Google Account authentication as a way to authenticate users on their applications. It eliminates the implementation of an authentication mechanism by the application developer.



#### **6.4.5 Economics**

For applications with moderate traffic and resource use, it is almost free to use the Google cloud. Even for applications with higher traffic and resource use, the charges are really low compared to other providers such as Amazon EC2 and Windows Azure.

#### **6.4.6 Fault Tolerance**

Google App Engine has suffered sporadic outages over last 2-3 years. However it is very fast to recover.

## Chapter 7: Eucalyptus Cloud Platform

“Eucalyptus stands for **Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems**”. This project was started in the Computer Science department of University of California, Santa Barbara as a research project in the field of high performance computing. The main goal of Eucalyptus was to build an open-source private cloud platform. Once the Cloud Computing platform became successful and widely used, a company was founded called Eucalyptus Systems to support the commercialization of the platform [32].

Eucalyptus is the first open-source Cloud Computing platform that implemented the Amazon EC2 compatible business interfaces. It uses Linux and Xen hypervisors to implement virtualization [33]. Because of the compatibility of the interfaces of Eucalyptus and EC2, it is a perfect candidate for implementing private as well as a hybrid cloud [33].

Eucalyptus is compatible with Microsoft operating systems as well as all the latest Linux distributions. It supports a variety of virtualization technologies such as Xen, KVM hypervisors and VMware [11] [32].

As its name suggests, Eucalyptus is a utility computing, which is elastic in nature, and it is used to connect user programs to useful, important systems. “It is an open-source platform that uses clusters and number of workstations to implement elastic utility computing that allows users to rest the network for computing needs” [33].

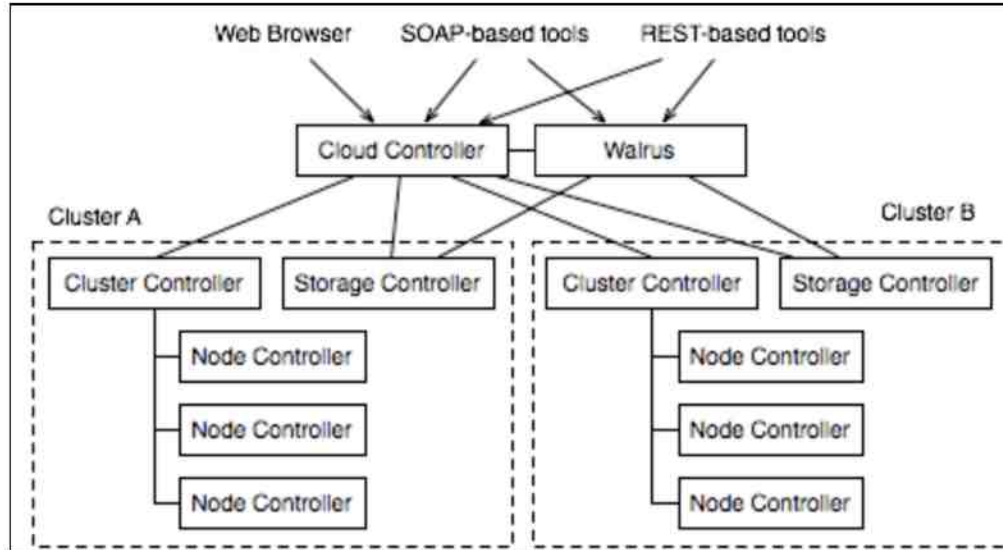
Eucalyptus allows the creation of private clouds on top of an organization’s existing IT infrastructure. It implements IaaS private cloud that is accessible using an API that is compatible with Amazon EC2 and S3. Because of the API compatibility, it is a perfect candidate to implement hybrid cloud, which makes it easier for the private cloud application to use resources on public cloud [34].

### 7.1 Eucalyptus Architecture

Eucalyptus Cloud Computing platform architecture consists of following five types of components.

- Cloud Controller (CLC)
- Walrus
- Cluster Controller (CC)
- Storage Controller (SC)
- Node Controller (NC)

**Figure 31** represents all the components that are part of Eucalyptus Cloud Platform. Each component is explained below in detail.



**Figure 31 · Eucalyptus Components Architecture [35]**

Each Eucalyptus Cloud contains one Cloud Controller and Walrus components:

- **Cloud Controller (CLC):** Cloud Controller is a program developed in Java and it is accessible using well-defined SOAP interface, which is compatible with EC2. It is also accessible via Web Interface and Rest based tools. It is responsible for handling incoming requests to the Eucalyptus cloud and resource scheduling.
- **Walrus:** Walrus is an implementation of storage service features that accompany the Eucalyptus Cloud platform. It is compatible with Amazon S3 APIs. Just like Amazon S3, Walrus allows users to store persistent data organized in buckets and objects [36].

**Figure 31** shows Cluster A and Cluster B. In Eucalyptus Cloud Computing platform there are thousands of clusters just like A and B. Each cluster has multiple controllers, which are responsible for various tasks in clusters.

- **Node Controller (NC):** Node Controller is responsible for performing execution of actions on the physical resources where the hypervisor and virtual machine are installed. It controls the VM operations such as the launching of instance, execution, inspection and termination [37].
- **Cluster Controller (CC):** The responsibility of Cluster Controller is to manage the collection of Node Controllers. Cluster Controller is installed at the head node of every cluster. It monitors the state of all the NCs and assigns or coordinates the flow of user requests.

- **Storage Controller (SC):** The storage controller is responsible for providing block-level network storage that can be dynamically attached to the virtual machines [32]. This is similar to Elastic Block Storage service provided by the Amazon Web Services. Storage Controller is also installed at the head node along with Cluster Controller.

## 7.2 Characteristics of Eucalyptus

Some characteristics that make Eucalyptus the most widely deployed Cloud platform for the private cloud are as follows [34]:

- Eucalyptus is an open source Cloud Computing platform. It can be updated or modified to satisfy user specific requirements.
- Design of Eucalyptus is modular. The Eucalyptus Components have well-defined interfaces, and because of that it can be easily swapped out for custom components.
- Eucalyptus allows its components to be installed strategically close to the needed resource. For example, Walrus can be installed close to the storage while Cluster Controller can be installed close to the Cluster it will be managing [34].
- Eucalyptus is highly flexible, as it can be installed on a particularly small set up and also on thousands of cores and terabytes of storage.
- It is compatible with Amazon EC2 and Amazon S3 APIs.
- It is designed to support most available hypervisors, which makes it hypervisor agnostic. It currently supports KVM and Xen.

Eucalyptus client interface allows users to access all the resources on the Cloud Computing platform. It is also the only way of communicating between the User and the Cloud Computing platform.

Ubuntu Linux distribution has included Eucalyptus platform and it is currently the core element of Ubuntu Enterprise Cloud.

Eucalyptus has provided an extraordinarily valuable solution that is extremely easy to deploy on top of the existing resources and allows users to create private cloud infrastructure with ease. It also provides the powerful features through Amazon EC2 and Amazon S3 compatible APIs [38].

## 7.3 Experiment

Understanding the working of Eucalyptus has provided me a great deal of knowledge about how cloud and different components in clouds work. I had fun learning different aspects of deploying the cloud and making things work, along with debugging for the issues and finding solutions. It has given me good insight of how different components of Eucalyptus work together.

### 7.3.1 Background

In this experiment, I wanted to deploy the Eucalyptus cloud locally using two of my older computers. I also wanted to deploy the “Password Manager” application that I have created as my previous class project, which was developed with keeping SOA (Service Oriented Architecture) in mind. It contains two different services, User Service and Password Service. User Service is responsible for user interface and Password Service is responsible for back-end and database operations. I have provided detailed information about this project in the Chapter 3.

### 7.3.2 Setup

For experiments I had 2 computers as mentioned below. Their configuration is presented in the following table.

	Machine 1	Machine 2
CPU	AMD Athlon(tm) 64 X2 Dual Core Processor 3800+	AMD Athlon(tm) 64 Processor 3800+
RAM	3 GB	2 GB

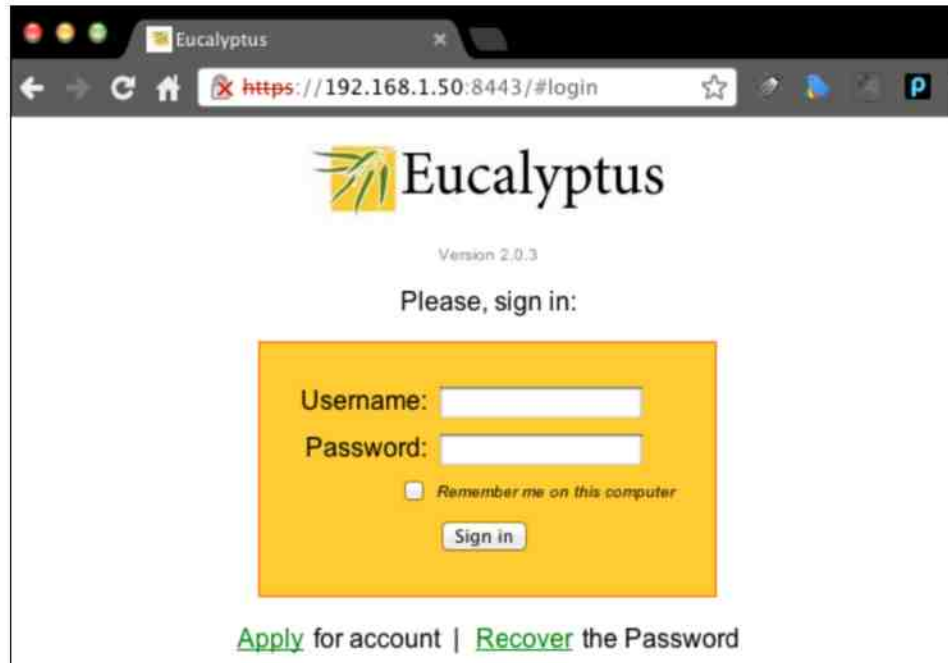
**Table 2 - Configuration of machines used to deploy Eucalyptus Cloud**

Since I wanted to experiment with installation of private cloud in my local network at home, I used faststart tool (<http://go.eucalyptus.com/Download-FastStart.html>) which was developed by the Eucalyptus team to test drive their private cloud offering. It provides an easy way to install a different component such as CLC (Cloud Controller), SC (Storage Controller), CC (Cluster Controller), Walrus and NC (Node Controller). I installed CLC, SC, CC and Walrus on Machine 2 (see **Table2**), and installed NC on Machine 1 (see **Table2**). The reason behind installing NC on Machine 1 is that, since it is a node controller, it is responsible for different VM related communications, and it creates VMs on the same node. I wanted to run two of my services on different instances and each instance needed at least one core of the CPU. Machine 1 has a dual core processor; I can create two instances with each core and run the application using them.

Once the complete configuration of the Cloud installation checked out fine, CLC had started the web application to control the basic aspects of the cloud, such as security credentials, configuration, user creation and installed images information to name a few.

I have installed Cloud Controller and its subordinate components on a host with IP 192.168.1.50. So I can access the Eucalyptus webapp at <https://192.168.1.50:8443>.

**Figure 32** depicts the login page of the Eucalyptus web app. The first time user will have to use default credentials “admin/admin” and is prompted to change the password on first login.



**Figure 32 - Eucalyptus Admin Console Login Screen**

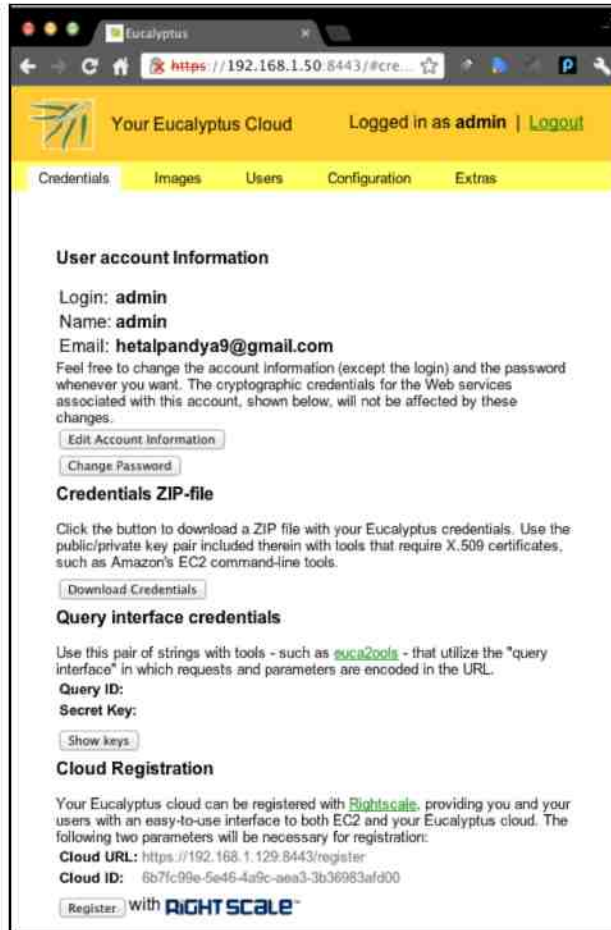


Figure 33 - Eucalyptus Admin Console - Credentials Page

After login, I was presented with Credentials page (see **figure 33**). This page contains information related to the account of the user who is logged in. It also includes the Credentials Zip-file that includes private/public key pair that is required to login to the cloud nodes from local computers; it can also be used to login with Amazon EC2 tools. Eucalyptus provides command line tools known as **euca2tools**, through which we can perform different operations such as registering the NCs, different cloud components, starting an instance, terminating an instance, adding security groups and many more. While performing these operations, it needs credentials to validate the requests, and in order to do that we need to provide Query Interface Credentials with it, which is available on the credentials page.

It also provides a way to integrate with Rightscale, which provides a graphical user interface, with which we can manage the Eucalyptus cloud. However, I was not able to use it, since I implemented this on a local network and did not have a public IP address.

The screenshot displays the Eucalyptus Admin Console Configuration page. The browser address bar shows `https://192.168.1.50:8443/#conf`. The page header includes the Eucalyptus logo, the text "Your Eucalyptus Cloud", and the user "admin" is logged in. The navigation menu includes "Credentials", "Images", "Users", "Configuration", and "Extras".

**Cloud configuration:**

- Cloud Host:
- Default kernel:
- Default ramdisk:
- Loaded configuration from server

**DNS configuration:**

- Domain name:
- Nameserver:
- IP:
- Loaded configuration from server

**Walrus Configuration:**

- Walrus host:
- Buckets Path:
- Space reserved for unbundling images (MB):
- Maximum buckets per user:
- Maximum bucket size (MB):
- Space reserved for snapshots (GB):
- Walrus configuration up to date

**Figure 34 – Configuration 1 - Eucalyptus Admin Console**

Another important page on web app is configuration; **Figure 34** depicts the different configuration of Cloud components. As indicated, Cloud Host is deployed on a host with IP address 192.168.1.50. You can also see the DNS configuration and Walrus configuration on the same image. Walrus is implemented using the same concept of Amazon S3, which is a block storage implementation.

**Figure 35** is the continuation of the configuration page mentioned above. It shows the cluster controller configuration. As you can see that Cluster Controller is installed on 192.168.1.50 and whenever a Node Controller is registered with the cloud, it is assigned to this cluster named "cluster00".



The screenshot shows the Eucalyptus Admin Console interface. The top section is titled "Clusters:" and contains configuration fields for a cluster named "cluster00". Below this is the "Cluster Controller" section with fields for "Host" (192.168.1.50), a checked "Dynamic public IP address assignment" checkbox, "Reserve for assignment" (10 public IP addresses), "Maximum of" (5 public IP addresses per user), "Use VLAN tags" (10 through 4095), "Storage Controller" section with "Disk space reserved for volumes" (50), "Max volume size" (10), a "Zero-fill volumes" checkbox, "Storage Interface" (eth0), and "Volumes path" (/var/lib/eucalyptus/volumes). At the bottom of the cluster section are buttons for "Register cluster", "Save cluster configuration", and a status "Clusters up to date".

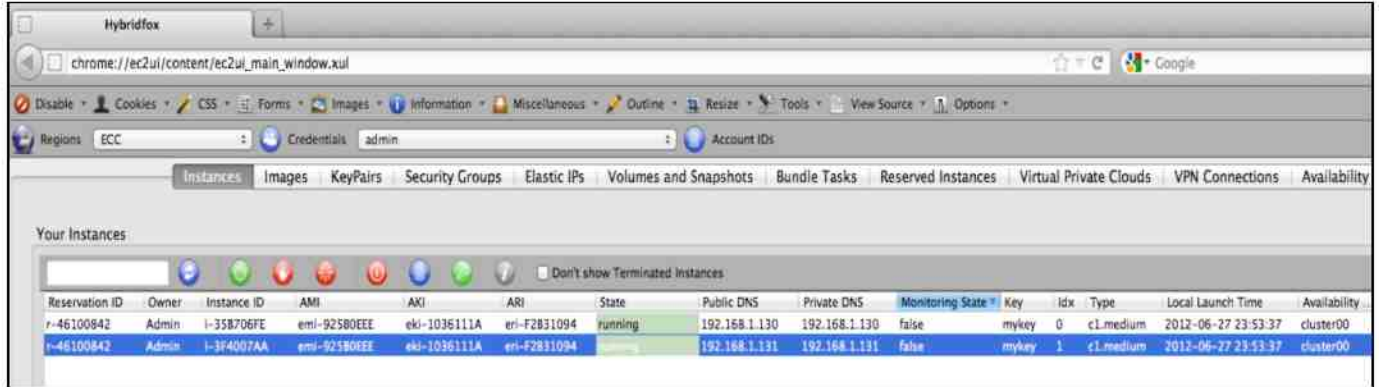
The bottom section is titled "VM Types:" and contains a table with columns "Name", "CPUs", "Memory (MB)", and "Disk (GB)". The table lists six VM types: m1.small, c1.medium, m1.large, m1.xlarge, and c1.xlarge. The rows for c1.medium and m1.xlarge are highlighted in yellow. Below the table is a "Save VmTypes" button.

Name	CPUs	Memory (MB)	Disk (GB)
m1.small	1	128	2
c1.medium	1	512	10
m1.large	2	768	15
m1.xlarge	2	1024	20
c1.xlarge	4	2048	20

Figure 35 - Configuration 2 - Eucalyptus Admin Console

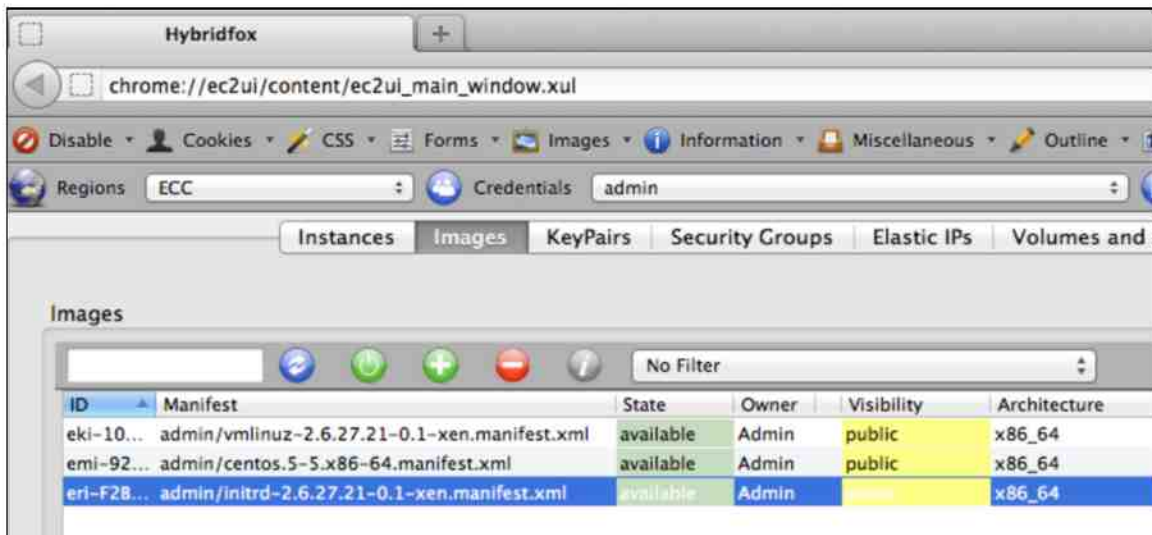
We can also see the different VM types that are preconfigured for us. It contains information about numbers of CPUs, Memory and Disk space, which needs to be allocated while creating an instance. We can make changes to it, but it should be incremental. It is required to provide instance type when we create a new instance.

Since I was not able to use Rightscale GUI to manage my instances on Eucalyptus cloud, I found an alternative to it. There is a plugin available for Firefox browser, named HybridFox, which can be used to visualize and manage Amazon EC2 and Eucalyptus cloud infrastructure. I used the same plugin to manage my private test cloud. **Figure 36** indicates that I have two instances running using IP addresses 192.168.1.130 and 192.168.1.131. It also mentions lots of other information such as instance type, which in my case is **c1.medium** and Availability in **cluster00**.



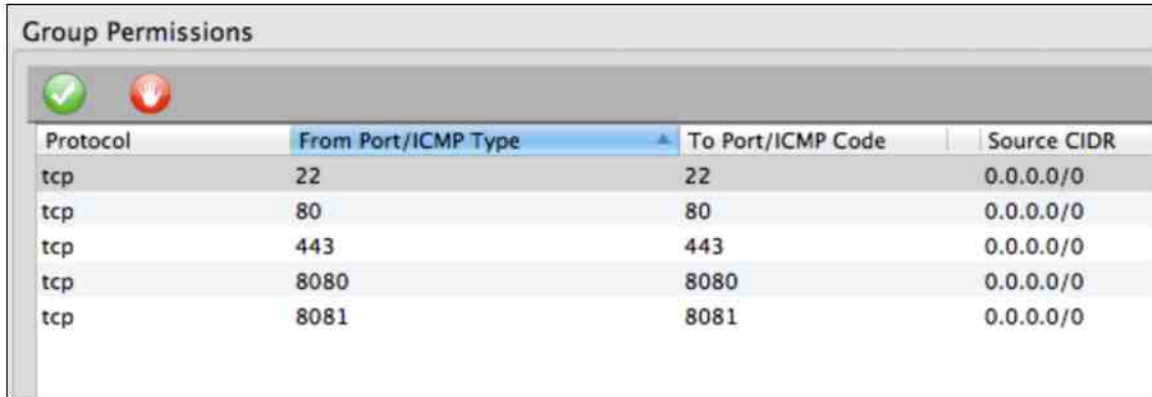
**Figure 36 - Eucalyptus Instances tab on Hybridfox plugin for Firefox**

We can also view and manage the number of available images, which are installed on the cloud. We can use those images and create an instance right from the Hybridfox plugin. It is a very cool free implementation. **Figure 37** depicts the images installed on my private test cloud right now.



**Figure 37 - Available Images on Eucalyptus showed on Hybridfox plugin**

Another very important aspect of cloud computing is security groups. By default all the access to instances created in the cloud is blocked. Eucalyptus also has the concept of Security Groups. By default, port 22 is open in Eucalyptus instances in order to connect to the virtual instances using valid private key. For any other access such as port 80, 8080 and 443, we need to add Group Permissions for it. Hybridfox plugin makes it easy for us. We can easily specify those specific ports and source CIDR, so that only those IP addresses can access the designated ports. **Figure 38** depicts the number of ports I have allowed to be connected from the outside of the cloud network. In my case, I have mentioned 0.0.0.0/0 as the source CIDR, which means a machine with any IP address on the Internet can connect to those 2 instances that I have running in my private test cloud.



Protocol	From Port/ICMP Type	To Port/ICMP Code	Source CIDR
tcp	22	22	0.0.0.0/0
tcp	80	80	0.0.0.0/0
tcp	443	443	0.0.0.0/0
tcp	8080	8080	0.0.0.0/0
tcp	8081	8081	0.0.0.0/0

Figure 38 - Group Permissions for Eucalyptus using Hybridfox

### 7.3.3 Implementation

With Cloud setup properly and 2 instances running, all I need to do is, to install the necessary application packages and run my application. In order to run my Java based web application, I installed Java Runtime Environment and Apache Tomcat6 web server to run the application.

As I mentioned earlier, I installed 2 services on 2 different instances. After installing Password Manager successfully, I was able to access the application using URL: <http://192.168.1.130:8080> and **Figure 39** depicts the login page of my application.

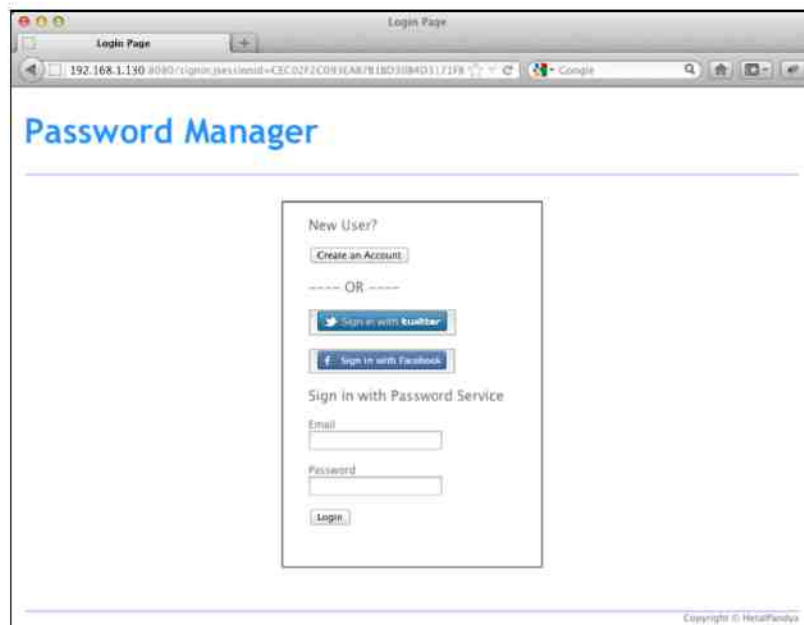


Figure 39 · Password Manager application running on Eucalyptus Cloud

## **7.4 Analysis**

Eucalyptus private Cloud Computing platform has become extremely popular since its emergence. It has a number of positive points that I found during my experiment with it. Every system has some sort of challenges and Eucalyptus has theirs as well. I have mentioned them later in this section.

### **7.4.1 Ease of Use**

It is an open-source cloud platform, available for use right out of the box for free. As its source code is also available to users, they can modify the software to satisfy their needs.

It is an open-source Cloud Computing platform, where users have access to the source code and it is fully customizable. If user thinks that it is lacking some functionality, which is useful to them, they can make changes to the code base and use it. Their source code is written mostly in Java and C.

Eucalyptus has tighter integration with Rightscale Cloud Management services, which is very useful for users. It provides a much better user-interface than Amazon Console and is very easy to use. It makes a Eucalyptus cloud admin's job much easier.

Compared to public clouds, in Eucalyptus, users will have to maintain the infrastructure. In small organizations, they will have to appoint a dedicated resource for management and maintenance.

Network setup in Eucalyptus is complicated at the beginning. Admin will have to take a lot of things in to consideration. Admin should have root access in order to access and maintain the physical hosts. Also admin has to make sure that it has proper network addresses assigned and has setup a dedicated subnet. [33]

### **7.4.2 Reliability**

Infrastructure clouds built with the Eucalyptus cloud platform resides within the boundary of an organization and it can be protected with a measure where only trusted users can use it. On Public clouds, sensitive data resides on a public network and user is not in its control and cannot guarantee its security

### **7.4.3 Scalability**

Eucalyptus APIs are designed to be compatible with the Amazon's EC2 and S3 services. This provides a great deal of flexibility and scalability to users, using that they can move the public cloud if they desire.

Another very important benefit of having APIs compatible with the Amazon's EC2 and S3 services is that, Eucalyptus can be used to create hybrid clouds. In which, users of the Eucalyptus cloud can leverage the power of EC2 and S3 services for large computation, data processing and many more operations, sometimes temporary, while keeping the sensitive data on the private cloud.

#### **7.4.4 Security**

Eucalyptus cloud is mostly established inside the organization's boundaries. Some institutions/organizations which handle sensitive customer data such as banks, the government, and many more do not want to put that sensitive data in public cloud. In such cases, eucalyptus cloud becomes the preferred choice as they can be sure that the information is not out there on the public platform and vulnerable.

It is one of the pros mentioned above that private cloud is secure; however, there are certain areas where it needs improvement. OS images uploaded to eucalyptus cloud become public automatically. Users can upload compromised images, which may open the door for hackers. [39]

#### **7.4.5 Economics**

Since it is a platform to create private cloud infrastructure, it becomes cost- effective for the business in the long run compared to using the resources on public cloud as they include charges per hour for numbers of resources used. This solution requires businesses to invest money to procure the infrastructures for the first time, but then after, it proves useful and saving money for businesses.

#### **7.4.6 Fault Tolerance**

In private clouds, organizations have greater control over the infrastructure. In public cloud offerings, provider is responsible for the infrastructure and sometimes highly capable resources are very expensive to use on public cloud, whereas in private cloud user can deploy high power, high capacity machines if they are required, with one-time cost. This functionality becomes helpful where better management can avert the outages and makes it tolerable to faults.

## Chapter 8: Cloud Computing Platform Comparisons

The goal of this work was to evaluate the architecture of different Cloud Computing platforms, along with analyzing their applications in the real world, in order to assess how users choose different platforms to satisfy varying needs. In previous chapters, I have also analyzed various platforms for their strengths and weaknesses.

Identifying different characteristics of the Cloud Computing platforms and comparing different platforms using the same criteria is another important task. In this chapter, I will compare all four platforms, Amazon Web Services, Microsoft Azure, Google App Engine, and Eucalyptus, using the same criteria.

### 8.1 Comparison based on common characteristics

The first set of comparison I would like to mention is based on the common characteristics that every cloud should have, or the features that users look for in cloud platforms to deploy their applications. Common characteristics that a cloud should have include: Scalability, Security, Cloud Type, Cloud Form, Ease of Use, Operating System Support, Supported Development Languages, Pricing Options, Reliability, Flexibility and many others. With some references from [40] [41] and [42] and my own analysis, I will be comparing the cloud platform mentioned above based on these criteria.

	Amazon Web Services	Microsoft Azure	Google App Engine	Eucalyptus
<b>Services</b>	IaaS	IaaS/PaaS	PaaS	IaaS
<b>Cloud Type</b>	Public	Public	Public	Private/Hybrid
<b>Computing Architecture</b>	Elastic Compute Cloud which supports Machine Images (AMI) which can be uploaded and executed	It is hosted in Microsoft datacenters and provides services that can be used together or individually	Uses Google's scalable and distributed architecture	Allows users to configure multiple clusters.
<b>Virtualization</b>	OS level virtualization with Xen Hypervisor	Windows Azure Hypervisor (based on Hyper-V)	Multitenant Architecture, runs on Java VM instances of Python runtimes	Designed to support any hypervisors, currently supports Xen and KVM

<b>Ease of Use</b>	Easy to use with GUI and supported APIs	Easy to use with pretty GUI	Easy to start with, moderate learning curve	Difficult to start with, can be easy if right tools are used, such as Hybridfox, Rightscale for management
<b>Reliability</b>	Reliability is a concern, frequent outages	Reliability is a concern	Reliability is a concern	Highly reliable
<b>Load Balancing</b>	Provides elastic load balancing	Hardware load balancing built in to the system	Automatic load balancing based on the load on the system	Can be configured to have a simple load balancing using the cloud controller.
<b>Storage</b>	Provides S3 for block storage and Simple DB for semi-structured datastore with querying capability	SQL Server Data Services, also supports blob data storage	Proprietary NoSQL Database	Walrus (similar to Amazon S3)
<b>Security</b>	Firewall, X.509 Certificates, SSL protected APIs, Security Groups	Security Token Service that creates Token according to the rule	Google Secure Data Connector, uses TLS based server authentication [41]	Web Service security for authentication, Public/Private Key generation by Cloud Controller.
<b>Deployment</b>	Using command line tools or Amazon Web Service Console	Through Visual Studio or using command line	Using IDE plugins or using AppCfg tool	Using command line tools or third party GUIs.
<b>User Interface</b>	Web Interface, Command Line, Supported Tools, Browser Plugin (Hybridfox)	Web Interface Only	Web Admin Console	Command line tools, A third party web based console (Rightscale), Browser Plugin (Hybridfox)
<b>Programming Framework</b>	Amazon MapReduce and Amazon Machine Images (AMI)	Microsoft.Net, Java, PHP, Ruby on Rails etc.	The MapReduce framework that supports Python, Java and other Java related technologies such as Servlet, JSP, JDO and JPA [41]	Axis2, C, Hibernate, Java

**Table 3 - Cloud Computing Platforms comparisons based on Common Characteristics [40] [41] [42]**

## 8.2 Comparisons based on free resources

Due to the increasing competition and the falling price of resources, all the providers are offering free services to some extent to entice customers to try or use their solutions and become familiar with them. This allows the consumer to test the services that are offered by the provider, without making a commitment. They don't have to spend money to test all the options that are available. I am going to compare three Cloud Computing platform providers that



include Google App Engine, Amazon EC2 and Microsoft Azure. I am not comparing Eucalyptus as it is an open source platform and available for free.

	Google App Engine	Amazon Web Services	Microsoft Azure
<b>CPU</b>	28 CPU Hours /Day	750 Instance Hours /Month	750 Small Compute Hours / Month
<b>Bandwidth – Outgoing</b>	1 GB / Day	15 GB across AWS	20 GB
<b>Bandwidth – Incoming</b>	1 GB/ Day	Unlimited	Unlimited
<b>Storage – Blob</b>	5 GB	30	35 GB
<b>Storage – Datastore</b>	1 GB	20 GB	1 GB

**Table 4 - Comparison based on free resources provided by Cloud Platforms [43]**

All these providers impose certain restrictions while allowing users to use their services. Some of the restrictions are mentioned below.

- Google App Engine's free quota gets reset at the end of every day. User will not incur any billing charges if their application satisfies these criteria. Free quota is available to all the users and does not restrict users at all [47].
- Amazon Web Services offers free trial with a number of limitations and it only allows micro instances to be used by users for free. These free resources are available to users for 1 year from the user signed up for the service [45].
- Microsoft Azure also restricts users to use extra small instances during the free trial period. Their free trial period is for 90 days only. After that, the user will start incurring charges for the resources they use [46].

### 8.3 Comparisons based on paid resources

Besides Google App Engine, all other provider starts charging money once the free trial ends. I wanted to find out what would be overage charges for some basic resources that every application uses while running in the cloud. So I did some research for all the three cloud providers that I analyzed above and found what are they charging for those resources.

Table 5 displays the statistics for the Google App Engine, Amazon Web Services and Microsoft Azure for resources such as CPU Hour, Outgoing and Incoming Bandwidth, Blob Storage and Database Storage [43].



	Google App Engine	Amazon Web Services	Microsoft Azure
<b>CPU</b>	\$0.08 – 0.32/ CPU Hour	\$0.08- 0.64/CPU Hour	\$0.08-0.32 / CPU Hour
<b>Bandwidth – Outgoing</b>	\$0.12/ GB	0.12/ GB	0.12/ GB per month
<b>Bandwidth – Incoming</b>	\$0.10/GB/month	Unlimited	Unlimited
<b>Storage – BLOB</b>	\$0.13/GB per Month	\$0.125/GB	\$0.125/GB per Month
<b>Storage- Datastore</b>	\$0.24/GB per Month	\$0.10/GB per Month	\$9.99 /GB for first GB and then \$3.99/GB additional per Month

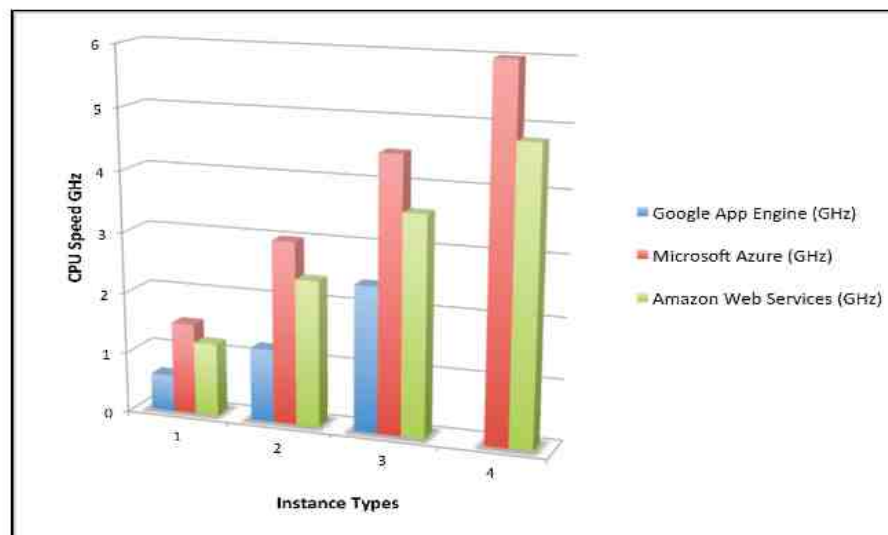
**Table 5 - Comparison based on paid resources per cloud platform**

## 8.4 CPU Instances Comparison

CPU is the most important resource in any Cloud Computing environment. Google App Engine, Amazon Web Services and Microsoft Azure provide different configurations of CPU speeds and RAM sizes. From the numbers I have noticed that CPU speeds are very much comparable for Amazon Web Services and Microsoft Azure as they provide a similar configuration. CPU speed of the Google App Engine is relatively slow.

If you have an application with a large user base and need more computing power, you might want to look at Amazon Web Services or Microsoft Azure. If you have a moderate user base and operations on your applications stay within the supported limit and you do not want to spend more, then Google App Engine is the solution that you should look in to.

**Figure 40** depicts the comparison of the CPU speeds for 3 Cloud Computing platforms.



**Figure 40 - CPU Speed Info Graph Provided by Cloud Computing platforms**

## **Chapter 9: Conclusion**

I was able to accomplish my goal of understanding different aspects of Cloud Computing platforms and their architecture. I learned the basics of Cloud Computing, its evolution, and its history. I was able to understand the difference between Grid Computing and Cloud Computing. I have also learned about the cloud delivery models such as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS) along with different cloud deployment models such as Private Cloud, Public Cloud and Hybrid Cloud.

Based on the analysis and evaluation of the Cloud Computing platforms now I can suggest to the users on which platform to use based their needs. I will be providing different features for each provider that stand out from each other. Users can choose the platform if the characteristics matches their needs.

If a user would like to start using cloud faster and easier with all these features and budget is not an issue then I would recommend using Amazon Web Services.

If a user is developing Windows technologies based application, it is an easy option for them to choose Windows Azure. It also provides a good alternative to Amazon Web Services. Since it is also having competitive prices, it is something you can try out.

If a user does not care about the control over the environment, have limited budget and used to the Google's software development kit then they can choose Google App Engine. It provides tools to develop and deploy the applications on their highly scalable and reliable architecture.

If a user does not want to use public clouds to deploy their applications and sensitive data or want to have total control over the underlying infrastructure and still have Cloud functionality then they can choose Eucalyptus Cloud Computing platform.

My observations for these Cloud Computing platforms do align with other evaluations that are explained below.

Amazon Web Services:

- It provides full control over the environment to the user. Supports easy development and deployment of an application.
- It is flexible and elastically scalable
- Since it provides virtual machines, user can install any application server and use any programming platform.

- It provides very competitive pricing.

#### Windows Azure:

- It is highly focused on Windows based services. However now it supports creation of Linux virtual machines as well. So it is competing with Amazon Web Services directly.
- It supports Microsoft SQL server, but does not provide support for Oracle or MySQL databases.
- If you are developing non-Windows applications, then deployment is cumbersome.
- It highly supports .NET framework. User can use Microsoft Visual Studio to develop and deploy their application as it has tight integration with Windows Azure.

#### Google App Engine:

- It is free to start and user can deploy 10 applications for free.
- Uses Google's highly scalable and reliable architecture.
- User only pays for the instance hour what is used, compared to Amazon and Windows Azure where user pays for the time their instance is running.
- User does not have control over the deployment environment. It is very restrictive in nature compared to Amazon Web Services and Windows Azure where user has complete control over the environment.

#### Eucalyptus:

- It is an open-source solution. So no cost to use. Eucalyptus can be implemented on the existing IT infrastructure.
- It allows users to create private cloud and also supports creation of hybrid cloud.
- It provides APIs compatible with Amazon EC2 and S3 services.

This thesis has given me much greater knowledge about how Cloud Computing works and how different platforms are implemented to supply various Cloud Computing services. It provided me with very significant insights about an extremely relevant and popular topic in the computer world today.

## **Chapter 10: Future Work**

During the course of this thesis, I was able to develop and deploy applications for Amazon Web Services, Microsoft Azure, Google App Engine and Eucalyptus. So overall, I got a chance to play around with all kind of Cloud services.

Besides these four Cloud Computing platforms, there are number of other providers such as Salesforce, Rackspace, Cloudstack, Nimbus, Abiquo, OpenNebula etc, which I did not get the chance to evaluate. I would like to evaluate them as some of them are open-source and some of them are proprietary platforms.

During my experiments, I used very few functionalities or services that is provided by Amazon Web Services, Windows Azure, Google App Engine and Eucalyptus. I would like to implement an application that is production ready, which I can deploy to any of those providers' platform, which can be used by actual user.

In Amazon Web Services, I only got the change to use EC2 (Elastic Compute Cloud), EBS (Elastic Block Storage), RDS (Relational Database Storage) and Elastic Beanstalk services. This is just the subset of services provided by Amazon. It also provides services such as S3, ElastiCache, Elastic MapReduce, Load Balancer, Monitoring, Simple Notification Service and many more. I did not get the chance to use these services, so in future, I would like to use them and experiment with them in my application.

While experimenting with Windows Azure, I only used their new service where I created Linux virtual machines and used them to deploy my application. In future, I would like to try other services that they have to offer such as, Windows based virtual machine, their PaaS solution to create an application using .Net framework, Windows SQL server etc. I would also like to try their service, which connects the cloud-based applications to MS-SQL server deployed on premise.

While using Google App Engine, I created simple PaaS application, which stores the passwords for a user in Google Datastore. Where I used only two services provides by Google App Engine, UserService and DatastoreService. Google App Engine offers lots of different services such as Mail, Memcache, URI fetch, Image Manipulation, and Cron, which I did not use in my previous application. I would like to create an application, which uses these services along with Google Cloud Storage and Google Cloud SQL services.

Study of Eucalyptus has provided me with vast knowledge of private cloud setup and lots of networking tactics. However, due to the limitation of the hardware for my experiment, I was only able to create a single cluster in the cloud with only one cloud node where I created two virtual machines to carry out my experiments. If possible, I would like to create multiple cloud

clusters with more than one node so that I can create powerful virtual machines and deploy applications, which can use more resources.

## References

1. Francesco, M. (2011). *JBoss AS 7 Configuration, Deployment, and Administration*. Packt Publishing.
2. The evolution of cloud computing: Important events in the life of cloud computing. Retrieved from <http://cloudcomputing.sys-con.com/node/1744132>
3. The benefits of cloud computing in IT intensive organizations. Retrieved from [oathesis.eur.nl/ir/repub/asset/10935/10935-Bakker.doc](http://oathesis.eur.nl/ir/repub/asset/10935/10935-Bakker.doc)
4. Krutz, R., & Vines, R. (2010). *Cloud security: A comprehensive guide to secure cloud computing*. John Wiley & Sons.
5. Mell, P., & Grance, T. (2009). *The NIST Definition of Cloud Computing. Recommendations of the National Institute of Standards and Technology*. NIST special publication, 2011 National Institute of standards and technology. 145(6).
6. Autonomic Computing. Retrieved from [http://en.wikipedia.org/wiki/Autonomic\\_computing](http://en.wikipedia.org/wiki/Autonomic_computing)
7. Hypervisor. Retrieved from <http://en.wikipedia.org/wiki/Hypervisor>
8. Public Cloud. Retrieved from <http://searchcloudcomputing.techtarget.com/definition/public-cloud>
9. Cloud computing. Retrieved from [http://en.wikipedia.org/wiki/Cloud\\_computing](http://en.wikipedia.org/wiki/Cloud_computing)
10. Cloud Computing Vocabulary. Retrieved from <https://sites.google.com/site/cloudcomputingwiki/Home/cloud-computing-vocabulary>
11. HTG explains: What is virtual machine hypervisor? Retrieved from
  - a. <http://www.howtogeek.com/66734/htg-explains-what-is-a-hypervisor/>
12. Architecting for the Cloud: Best Practices. Retrieved from [http://media.amazonwebservices.com/AWS\\_Cloud\\_Best\\_Practices.pdf](http://media.amazonwebservices.com/AWS_Cloud_Best_Practices.pdf)
13. Elastic Compute Cloud. Retrieved from <http://aws.amazon.com/ec2/>
14. Amazon Web Service Products. Retrieved from <http://aws.amazon.com/products/>

15. Amazon Elastic Beanstalk. Retrieved from <http://aws.amazon.com/elasticbeanstalk/>
16. Amazon RDS. Retrieved from <http://aws.amazon.com/rds/>
17. Amazon S3. Retrieved from <http://www.slideshare.net/parn09/intro-to-amazon-s3-presentation>
18. Advantages and disadvantages of private and public cloud computing solution. Retrieved from <http://blog.virtual.com/2011/the-advantages-and-disadvantages-of-private-and-public-cloud-computing-solutions>
19. Amazon S3. Retrieved from <http://aws.amazon.com/s3/>
20. Redkar, T., Windows Azure Platform. Apress Publications
21. Chappell D., (2011), Introducing The Windows Azure Platform
22. Yogesh Simmhan, Catharine van Ingen, Girish Subramanian, Jie Li, "Bridging the Gap between Desktop and the Cloud for eScience Applications", 2010 IEEE 3rd International Conference on Cloud Computing
23. Chia-Chi Teng, Jonathan Mitchell, Christopher Walker, Alex Swan, Cesar Davila, David Howard, Travis Needham, "A Medical Image Archive Solution in the Cloud"
24. Windows Azure Overview Part 2: Pros and Cons. Retrieved from <http://blog.monitis.com/index.php/2012/03/26/windows-azure-overview-part-2-pros-and-cons/>
25. Retrieved from <http://my.safaribooksonline.com/book/programming/microsoft-dotnet/9781430224556/windows-azure/disadvantages>
26. What is Google app engine? Retrieved from <https://developers.google.com/appengine/docs/whatisgoogleappengine>
27. Google app engine. Retrieved from [http://en.wikipedia.org/wiki/Google\\_App\\_Engine](http://en.wikipedia.org/wiki/Google_App_Engine)
28. What are the advantages and disadvantages of Google app engine? Retrieved from <http://www.quora.com/What-are-the-advantages-and-disadvantages-of-Google-App-Engine-against-Amazon-EC2-or-Windows-Azure>
29. Google cloud sql. Retrieved from <https://developers.google.com/cloud-sql/>
30. Google cloud storage. Retrieved from <https://developers.google.com/storage/>
31. Overview of java support in Google app engine. Retrieved from <http://www.byteonic.com/2009/overview-of-java-support-in-google-app-engine/>

32. Eucalyptus (Computing). Retrieved from [http://en.wikipedia.org/wiki/Eucalyptus\\_\(computing\)](http://en.wikipedia.org/wiki/Eucalyptus_(computing))
33. Peng, J., Zhang, X., Lei, Z., Zhang, B., Zhang, W., & L, Q. (2009). Comparison of Several Cloud Computing Platforms, 23-27
34. What is eucalyptus? Retrieved from <http://open.eucalyptus.com/learn/what-is-eucalyptus>
35. Installing Eucalyptus 2.0. Retrieved from [http://open.eucalyptus.com/wiki/EucalyptusInstallation\\_v2.0](http://open.eucalyptus.com/wiki/EucalyptusInstallation_v2.0)
36. Eucalyptus Storage. Retrieved from [http://open.eucalyptus.com/wiki/EucalyptusStorage\\_v1.4](http://open.eucalyptus.com/wiki/EucalyptusStorage_v1.4)
37. Overview of Eucalyptus. Retrieved from [http://support.rightscale.com/09-Clouds/Eucalyptus/01-Overview\\_of\\_Eucalyptus](http://support.rightscale.com/09-Clouds/Eucalyptus/01-Overview_of_Eucalyptus)
38. Daniel Nurmi, Rich Wolski, Chris Grzegorzczak, Graziano Obertelli, Sunil Soman, Lamia Youseff, Dmitri Zagorodnov "The Eucalyptus Open-source Cloud-Computing System", Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium
39. Oliver Popović, Zoran Jovanović, Nenad Jovanović, Ranko Popović "A comparison and security analysis of the cloud computing software platforms", Telecommunication in Modern Satellite Cable and Broadcasting Services (TELSIKS), 2011 10th International Conference
40. Bhaskar Prasad Rimal, Eunmi Choi, Ian Lumb "A Taxonomy and Survey of Cloud Computing Systems", 2009 Fifth International Joint Conference on INC, IMS and IDC.
41. Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities", The 10th IEEE International Conference on High Performance Computing and Communications
42. Panos Louridas, "Up in the Air: Moving Your Applications to the Cloud", IEEE Software July/August 2010
43. Google App Engine (GAE) versus Amazon Web Services (AWS), Retrieved from <http://notwastingtime.blogspot.com/2010/05/google-app-engine-gae-versus-amazon-web.html>
44. Oliver Popović, Zoran Jovanović, Nenad Jovanović, Ranko Popović "A comparison and security analysis of the cloud computing software platforms", Telecommunication in Modern Satellite Cable and Broadcasting Services (TELSIKS), 2011 10th International Conference



45. AWS product Information, Retrieved from [http://aws.amazon.com/search?searchQuery=pricing&searchPath=products\\_and\\_info&x=0&y=0](http://aws.amazon.com/search?searchQuery=pricing&searchPath=products_and_info&x=0&y=0)
46. Microsoft Azure Pricing Details. Retrieved from <https://www.windowsazure.com/en-us/pricing/details/>
47. Google App Engine Billing and Budgeting Resources. Retrieved from <https://developers.google.com/appengine/docs/billing>
48. Inspiring, Sharing & Giving, Blog Archive, CLOUD COMPUTING. Retrieved from <http://www.ucukomarudin.com/?p=107>