INVESTIGATION OF AMAZON AND GOOGLE FOR FAULT TOLERANCE STRATEGIES IN CLOUD COMPUTING SERVICES

SHEREEN AL-RAHEYM

JANUARY 2015

INVESTIGATION OF AMAZON AND GOOGLE FOR FAULT TOLERANCE STRATEGIES IN CLOUD COMPUTING SERVICES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES OF
ÇANKAYA UNIVERSITY

BY
SHEREEN AL-RAHEYM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGRE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF
MATHEMATICS AND COMPUTER SCIENCE / INFORMATION TECHNOLOGY PROGRAM

JANUARY 2015

Title of the Thesis: **Investigation of Amazon and Google for Fault Tolerance Strategies in Cloud Computing Services.**

Submitted by **Shereen AL-RAHEYM**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.

[illegible]

Prof. Dr. Taner ALTUNOK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

[illegible]

Prof. Dr. Billur KAYMAKÇALAN
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

[illegible]	[illegible]
Dr. Sinan Can AÇAN Co-Supervisor	Assist. Prof. Dr. Ö. Tolga PUSATLI Supervisor

Examination Date: 19/01/2015

Examining Committee Members

Assoc. Prof. Dr. Fahd JARAD	(UTAA)	[illegible]
Assist. Prof. Dr. Dr. Ö. Tolga PUSATLI	(Çankaya Univ.)	[illegible]
Assist. Prof. Dr. Abdül Kadir GÖRÜR	(Çankaya Univ.)	[illegible]

STATEMENT OF NON-PLAGIARISM PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name	:	Shereen, AL-RAHEYM
Signature	:	[illegible]
Date	:	19.01.2015

ABSTRACT

INVESTIGATION OF AMAZON AND GOOGLE FOR FAULT TOLERANCE STRATEGIES IN CLOUD COMPUTING SERVICES

AL-RAHEYM, Shereen
M.Sc., Department of Mathematics and Computer Science
/ Information Technology Program
Supervisor: Assist. Prof. Dr. Ö. Tolga PUSATLI
Co-supervisor: Dr. Sinan Can AÇAN

January 2015, 53 pages

Cloud computing has recently become an attractive topic due to its ability to offer information technology solutions through virtual machines as on-demand services to share and consume resources over the Internet. As a result of rapid development in such services, the necessity of fault tolerance in the cloud is a major concern with reliability, availability and dependability is more critical to this new service type. This thesis investigates techniques and means of tolerating cloud services as well as cloud customers' systems/enterprises execution over the cloud safe from failures. As one of this findings shows, failures in cloud enabled services should be expected to occur hence they should be handled. The essential features of implementing fault tolerance strategies guarantee the business continuity, avoid financial lost, recovering systems from failures, and provide disaster recovery as well. The specific focus is to explore scenarios of avoiding/recovering from failures through redundancy, checkpoint and replication. Commercial IaaS providers such as Amazon's AWS and

Google’s GCE are taken as examples as they tolerate their infrastructure from failures; in this way a robust architecture with fault tolerance property could be proposed for a system/enterprise. Hence, a general conceptual model with fault tolerance considerations has been proposed. With this basis, addressing potential failures in detail, implementing fault tolerance techniques, organizing teamwork to setup business policies are left for future work. Such limitations can be addressed through online questionnaires to collect information for case studies.

Keywords: Cloud Computing, Fault Tolerance, Reliability, Availability, Dependability, Redundancy, Checkpoint, Replication, AWS, GCE.

ÖZ

AMAZON VE GOOGLE'IN BULUT BİLİŞİM SERVİSLERİNDE HATA TOLERANS STRATEJİSİ KAPSAMINDA İNCELENMESİ

AL-RAHEYM, Shereen
Yüksek Lisans, Matematik-Bilgisayar Anabilim Dalı
/ Bilgi Teknolojileri Bölümü
Tez Danışmanı: Yrd. Doç. Dr. Ö. Tolga PUSATLI
Eş Danışman: Dr. Sinan Can AÇAN

Ocak 2015, 53 sayfa

Bulut bilişim, sanal makinalarla İnternet üzerinden kaynak paylaşımı ve kullanımı sunarak istek-üzeri bilişim teknolojileri çözümleri sunmaktadır; bu sayede, son zamanlarda çekici bir araştırma başlığı olmuştur. Bu tür servislerin hızlı gelişimleri sonucunda, bulutta hata toleransı gerekliliği ana konulardan birisi olmuştur. Bu gereklilik, bu yeni servislerin güvenilirliği ve her zaman hazır olması şartlarıyla daha da titizlik gösterilmesiyle öne çıkmaktadır. Bu tez, bulut servislerinde ve kullanıcıların sistemlerinde ve işyerlerinde işleyiş hatalarından koruyacak teknikleri ve olanakları incelemektedir. Bulgulardan birisinin de gösterdiği üzere, bulut kullanan servislerde hatalar olasıdır, beklenmelidir ve bu yüzden çözüm yolları bulunmalıdır. Hata tolerans stratejilerinin esas nitelikleri işin devamlılığını, finansal kayıpların önlenmesini, sistemin tekrardan ayağa kaldırılmasını ve felaketten kurtulmayı, güvence altına almalıdır. Tezin kapsamında, işleyiş bozukluklarının, tekrarlamalarla, kontrol noktalarıyla ve yedeklemeyle önlenmesi ve düzeltilmesi senaryoların incelenmesi vardır. Altyapıları hatayı tolere edebilen Amazon'un ticari

AWS'i ve Google'ın ticari GCE'si gibi servis olarak altyapı sağlayıcılar örnek olarak incelenmiştir ki bu sayede hata tolerans özelliği olan sağlam mimariler, sistem ve işyerlerine önerilebilsin. Bunlar temel alınarak, olası çöküşlerle nasıl ilgilenilebileceğini, hata tolerans tekniklerini, takım çalışması düzenlenmesini içeren iş politikaları olşturulması ileriki çalışmalara bırakılmıştır. Bu kısıtlamalara yönelik araştırmalar, çevrim içi anketlerle, daraltılmış kapsamlar için bilgi toplanarak yapılabilir.

Anahtar Sözcükler: Bulut Bilişim, Hata Toleransı, Güvenilirlik, Hazır Olma, Tekrarlama, Kontrol Noktası, Yedekleme, AWS, GCE.

ACKNOWLEDGEMENTS

At the very outset, I would like to start my sincere gratitude and grateful to both my supervisor Assist Prof. Dr. Dr. Ö. Tolga PUSATLI and co-supervisor Dr. Sinan Can AÇAN, thank you for your guidance, commitment, interest, and support during my study.

Many thanks, to my friends: Eman thank you for your support and encouragement, Ruaa, thank you for bringing smile to my face, and I wished if we could study together, my friends and brothers Ashraf and Hassan thank you for your friendship and support since my first minute when I arrived in Ankara.

Last but not least, my deeply grateful and gratitude goes to my family, without your love, encouragement, support, and understanding I could not complete my study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBRIVATIONS

EC2	Amazon Elastic Compute Cloud
RDB	Amazon Relational Database Service
S3	Amazon Simple Storage Service
VPV	Amazon Virtual Private Cloud
AWS	Amazon Web Services
AZ	Availability Zone
DM	Decision Maker
DR	Disaster Recovery
ELB	Elastic Load Balancing
FT	Fault Tolerance
GCE	Google Compute Engine
Haproxy	High Availability Proxy
HA	High Available
HDFS	High Distributed File System
HPC	High Performance Computing
IT	Information Technology
IaaS	Infrastructure As A Service
MTBF	Mean Time Between Failure
MTTF	Mean Time to Failure
MTTR	Mean Time to Repair
NIST	National Institute of Standards and Technology
PD	Persistent Disk
PaaS	Platform as a Service
QoS	Quality of Service
SLA	Service Level Agreements
SOA	Service-Oriented Architecture
SaaS	Software as a Service
SPE	Stream Processing Engines
TMR	Triple-Modular Redundancy
VMI	Virtual Machine Image
VLAN	Virtual Local Area Network
WS	Web Services

CHAPTER 1

INTRODUCTION

1.1 Motivation

Cloud computing is relatively young, and has manifested as a novel model over conventional systems. Cloud service providers supply information technology (IT) solutions such as on-demand services over the Internet. Herein, this service has a large influence on IT companies. The main centric point is that the cloud promise to reduce the entrance barrier to the market as well as minimizes resources' wasting. In reality, modern day cloud datacenters host a number of servers which are connected via mass switches and routers as well as run over virtualization technology so as to share resources, software and information. Due to the large infrastructure, the complexity, and the rapid exponential demand of cloud services, the reliability and guaranteed availability of resources become a critical challenge in cloud computing. For example, in 2012 Amazon and Azure which are considered as the biggest cloud providers in the market suffered service interruption which cause down in some customers' services. Basically, customers before transfer their critical enterprise systems to the cloud ask their cloud service providers a question" Do the cloud services ensure 100 percent of availability?" the answer is no, the cloud applications' dependability tile now has a hurdle to guarantee all the required or desirable elements. Failures in the cloud happen all the time because of the complicated underlying infrastructure and cloud applications are commercial, so that it is usually not designed to fault tolerate. Investigating in this field gives an awareness to the companies entering the cloud how they can survive their business.

1.2 Thesis Purpose

In basic technological term, the clouds are desired to have fault tolerance property to reduce systems risk of failures and to avoid costly unavailability. As a result, employing an effective fault tolerance is not an easy task within the cloud because the physical underlying level is the duty of cloud providers and under their authorization. However, we see that deploying solutions over the cloud need a comprehensive understanding of cloud infrastructure components, techniques, services, and limitation. This thesis has trend toward offering a general conceptual model in which customers establish application strategy scenario on how to design a solution with fault tolerance architecture and deployment. This is the purpose of this master thesis.

1.3 Scope Boundaries

This master thesis essentially focuses on fault tolerance characteristic of cloud computing and it splits into two facets. The first one presents a survey of cloud computing, its definition, roots, underlying technologies, features, taxonomy, benefits, and challenges as well. A clear understanding of this background helps realizing faults within the cloud, fault tolerance approaches, methods, policies, procedure, and how to gain dependability in a system. Moreover, this part reserves as a literature related works that clearly offer fault tolerance vision within the cloud infrastructure.

The second facet of this work is specifically focusing on evaluating use cases of real life commercial companies which provide infrastructure as a service. Their infrastructure components that assist providing availability are clearly discussed. This investigation builds upon the information collected from Amazon and Google Cloud official documents, blogs, presentations, webinars, and from their customer success stories published. The use cases enhance us to develop a conceptual method of deploying a fault tolerance solution across the cloud virtual infrastructure layer.

1.4 Research Question

In this thesis the following points will be covered in chapter two and three:

- From a business perspective, what does cloud computing mean? Chapter2
- What is the formal taxonomy of cloud computing? Chapter 2
- What are the cloud benefits and challenges? Chapter 2
- What is the essential definition of fault tolerance? Chapter 2
- How fault tolerance polices implementing across the cloud? Chapter 2
- How can businesses achieve high-availability assurance within the cloud? Chapter 3
- How can Amazon and Google survive their infrastructure as service components from failures? Chapter 2

The main thesis question addressed in this chapter is:

- *How to enhance architectural IT investment throughout cloud computing services to safeguard continuity of business from failure?*

CHAPTER 2

BACKGROUNG AND LITERATURE REVIEW

2.1 Background and Literature Survey

2.1.1 Cloud computing definition

Nowadays, cloud computing is a popular paradigm; it can be defined as a new style and mode of using information technology as a service over the Internet. Everyone not agrees on what it is. In fact, it seems that cloud definition varying from expert to cloud provider. Here we quote some definitions for cloud computing:

"National Institute of Standards and Technology" (NIST) team produced the more precise and technical definition:
"*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.*" [1]

Don Dodge definition:
"*Today's combination of high-speed networks, sophisticated PC graphics processors, and fast, inexpensive servers and disk storage has tilted engineers toward housing more computing in data centers. In the earlier part of this decade, researchers espoused a similar, centralized approach called "grid computing." But cloud computing projects are more powerful and crash-proof than grid systems developed even in recent years.*"[2]

Oracle’s Ellison definition:
“*All the Cloud is computers in a network… Our industry is so bizarre. I mean, they just change a term and they think they’ve invented technology*”. [3]

Salesforce.com’s Marc Benioff defines cloud as:
“*Our definition of Cloud Computing is multi-tenant, it’s faster, half the cost, pay as you go, and it grows as you grow or shrinks as you shrink. It is extremely efficient*”. [3]

With cloud computing described as above, cloud computing can be identified as a business model that offers online computing resources. It aims to deliver on-demand large amount of IT infrastructure such as (servers, storage applications, network, services) for multi-users to share and consume as a utility in a virtualized manner and highly abstracted from cloud service providers (e.g. AWS, Azure, GCP).

2.1.2 Roots OF cloud computing

From one perspective, cloud computing can be realized as a gathering of the combined evolution of computing technologies, particularly in hardware virtualization, Internet technology such as (WS, SOAs, Web2.0), distributed computing (clusters, grids), system management (datacenter automation, autonomic computing). From an alternative perspective, cloud computing mutates, how we can invent, deploy, scale, maintain, likewise pay for applications and infrastructure. [4]

- **From mainframes era to clouds**

Earlier, many users shared robust mainframes that were used by companies and governmental organizations for critical applications. In reality, mainframes were difficult to scale, users did not have full control over their applications, more expensive as well. As such, stand-alone PCs and mini computers were the next computing model. They gave users full control of their resources. The story of computing continued with a new challenge which was: how to share data and resources? Throughout connecting computers together, which form local network. As a result, a business computing needs grew, the LAN was connected to each other's constructing a broad and global network "The Internet" to gain remote applications' utility and resources [5]. Hence, the complexity and high cost management of IT infrastructure increased and the evolution seeks to share, select, and aggregation of geographically distributed resources to work on a single problem by using grid model. [6]

Furthermore, clustering model emerged with the ability of interconnecting a group of stand-alone computers that work as a single system.[7]. As though, comparing these models, one can recognize that cloud is just a return to the original mainframe with several important differences such as management, requiring lower space, and less energy consuming [5]. As well, the cloud can be seen as an evolution from isolated computers over cluster and beyond grids.

- **Web services, SOA, Web2.0**

Web Services (WSs) and Service-oriented Architecture (SOA) represent the essential technologies in cloud computing. Basically, cloud services are constructed as Web Services [5]. Web services technology is a practical and cost-effective solution with the capability of gathering and uniting information distributed among different applications over various OSs and platforms [8]. Moreover, they can be accessed through standard-based Internet protocol such as HTTP and SMTP. Likewise, this technology is based on standardized XML, SOAP, WSDL, and UDDI technologies [9]. Additionally, managing and monitoring web services within cloud can be gained by SOA. [6]

Another beneficial web-based technology for cloud computing is Web 2.0 which allow users to interact, collaborate, share resources, and enhance creativity with each other in a virtual community. [5]

2.1.3 Virtualization as the underlying technology

Virtualization is the backbone of cloud computing [3] in which cloud technology could never be without it. The concept of virtualizing a computer system's resources is based on adding a layer between the hardware and operating system that allows diverse operating system instances working simultaneously upon a single server. Physical resources, including: processors, memory, storage, network, and I/O devices are dynamically partitioned and shared. [10]

Generally, two virtualization architectures are found: hosted architecture where is installed as an application upon the operating system. Employing this technique is flexible for servers, workstations and individual use as well. In contrast, hypervisor (bare-metal) architecture could be installed directly on x86-based system. Therefore cloud computing's datacenter basically relies on this concept that has direct access to physical resources. So that, virtualization layer runs directly on physical hardware of the server, and hence only virtualization layer is considered to be a part of the physical server. [11]

Virtualization has advantages as many OSs can run on a single computer, saves the quantity of purchase servers, minimize the management and maintenance cost, as well as decrease the consuming of electricity and cooling system. [12]

2.1.4 Essential features of cloud models

The Cloud computing model enables a number of new certain features that satisfy clients/consumers. These characteristics are appeared as interpreted solutions and vary according to the class or type of cloud model that is employed [4]. As stated in the NIST definition of cloud computing, cloud model is formed of these five main characteristics according to [1]:

- **On-demand self-services**

Consumers can access cloud resources such as server or storage by using websites or web services interface whenever they need them. Automatically, they can order, customize, pay without interaction with the cloud provider’s personnel. [17]

- **Broad network access**

Because cloud computing services are web-based technology, a consumer can gain resources over the Internet using standard methods such as heterogeneous OSs, or thick and thin platform as laptops, and smart phones [13]. Therefore cloud computing is device independent. [29]

- **Resource pooling**

Cloud computing supports multi-tenant resource usage. In other words, a pool of provider’s computing resources is shared among a large number of consumers. These virtual resources allocate and reallocate relying on consumer demand [1].

In fact, resource pooling in cloud computing based on abstraction concept, this mean that the exact location of resources is not stated (e.g. VMs, processing, memory, storage, or connectivity). However, it may be able to specify location at a higher level of abstraction as country name, state, or datacenter. [1]

- **Rapid elasticity**

Resources can be supplied or released quickly and elastically so that consumers capable to scale up/add or scale down resources in an automatic or manual method, as well as in various quantities and at different time [17]; similar to much or less electricity we need from the power grid.

- **Measured services**

A metered system is used which makes consumers pay-per-use billing, for example the amount of storage billed by the day, the amount of processing power billing by the hour, as well as network I/O bandwidth and number of transactions. [5]

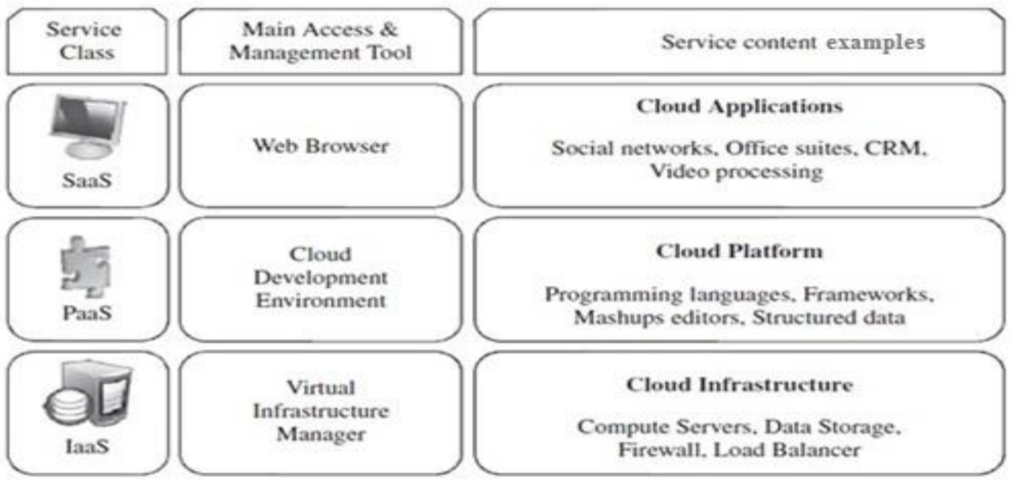
Moreover, cloud provider control, monitor, optimize, and record the resources that are used automatically. There is neither need up-front an investment nor capital expenditure. [1]

2.1.5 Cloud computing taxonomy

Mainly, two different taxonomies of clouds can be determined: these ones which are built on the service model and those which are built on the deployment/underlying infrastructure model. [14]

2.1.5.1 Cloud computing service models

In terms of business model requirements, cloud computing services are partitioned into three architectural layers, according to the abstraction level of the functionality provided and the providers' service models which are commonly known as SaaS, Paas, IaaS[16,5]. Together cloud computing services stack shown in figure 1



Service Class	Main Access & Management Tool	Service content examples
SaaS	Web Browser	**Cloud Applications** Social networks, Office suites, CRM, Video processing
PaaS	Cloud Development Environment	**Cloud Platform** Programming languages, Frameworks, Mashups editors, Structured data
IaaS	Virtual Infrastructure Manager	**Cloud Infrastructure** Compute Servers, Data Storage, Firewall, Load Balancer

Figure 1 The cloud computing stack [4]

Additionally, in a literary point of view, the service model format is XaaS/"<*something*> *as a service*". So there are NaaS: network as a service, DaaS: [Database, Development, Desktop] as a service, BaaS: Business as a service, OaaS: Organization as a service, and so on [14].

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

The easy way to online access software libraries, standard business applications, and services via web portals. Consumers' responsibility starts and finished with entering and managing its data and user interface. In other words, neither manages nor control over the cloud resources as well as the applications they use, but may be given a limited privilege to configure the application's setting. [1]

In addition, all SaaS shares the following characteristics:

- On-demand purchasing access to applications over WWW.
- Allow business saving money, no license fees.
- Low barrier to enter the market
- Automatically upgrade applications by the cloud service provider.
- Securing access "SSL" to applications.[18]
- Good for collaborative working such as Google Docs which allows multiple users to share documents and to work on them concurrently. [29]

In contrast, SaaS's consumers just use the applications. In other word, SaaS essentially works in "*take-it-or-leave-it*" form of cloud computing [29]. Popular and familiar examples of SaaS are e-mail services, word processing and spreadsheet. Furthermore, Salesforce.com is a well-known cloud application provider that offers business productivity application CRM, and Google offers basic business services such as Gmail and word processing programs.

- **Platform as a Service (PaaS)**

PaaS is another software platform model provides a stable online environment (e.g. VMs, OSs, applications, services, development framework, transactions, and control structure) on which consumers/developers can readily create, test, and deploy web application, and develop new software or even expand the present solution throughout browser-based development tools. [18, 19]

The cloud infrastructure, OSs, and software are entirely managed by the cloud service provider while the consumers are just responsible for installing and managing what they deploy [14].

Ideally, PaaS has all these features:

- Web-based development environment with the capability of creating a database and editing application code.
- Built-in scalability, security, access control, and web service interfaces.
- Can be integrated with other applications on the same platform.
- Including tools for designing web forms, defining business rules and creating workflow as well. [13]

In other side, PaaS limits developers to provider's languages and tools [29]. Additionally, the notable commercial examples of PaaS are Google App Engine, Microsoft Windows Azure, and Force.com.

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

IaaS provides fundamental computing resources as (VMs, virtual storage space, virtual network, memory, and CPU cycle) [14]. Thus, organizations/consumer can put and provision whatever they want on-demand. We can note that sometime IaaS is also called hardware as a service (HaaS). [18]

Significantly, IaaS service's providers/vendors own the equipments and manage all the infrastructure while consumers, which are mainly developers are provided with administrative responsibilities for all other sides, including creating virtual servers, storage, network, deploying OS, application and user interaction with the system as well [14]. Dynamically, these virtualized hardware resources can be scaled up/down. Furthermore, some IaaS service providers able to store copies of specific and hot data in different locations. [13]

Additionally, IaaS includes many parts such as SLAs, network hardware and software, utility computing billing, and platform virtualization environments. A well-known example is Amazon EC2 cloud hardware provider [18]. Figure 2 highlights the functionality provided by the cloud provider to the consumers, which generally depends on the consumer's requirements and on-premises responsibility as well.

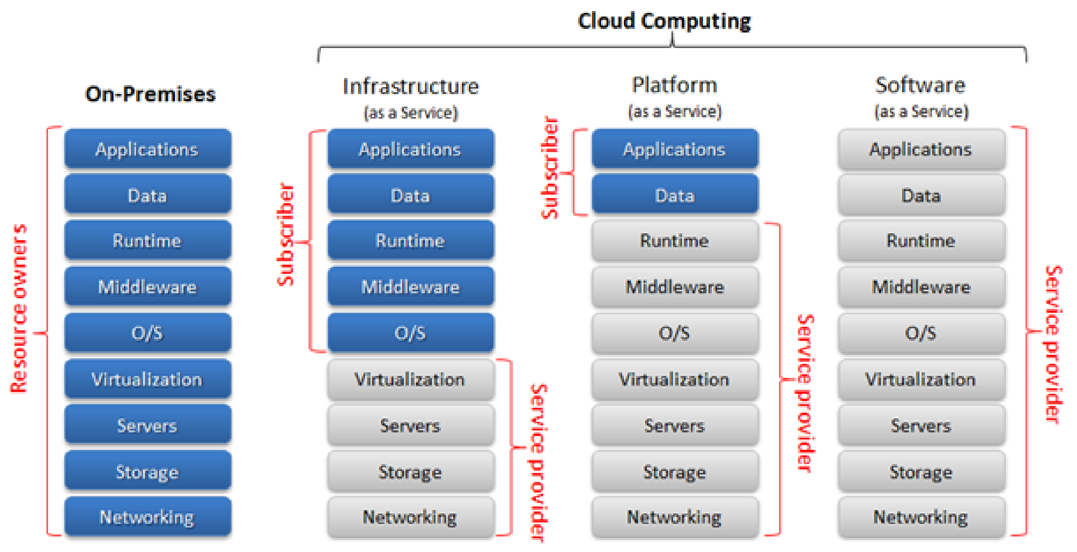


Figure2 The whole cloud server stack responsibilities [20]

2.1.5.2 Cloud computing infrastructure models

Clouds can be categorized based on the implied infrastructure deployment model. Depending on requirements, IT organization can choose to deploy their applications on a particular model. Each deployment model specifies where the cloud is located, for what purpose and who manages the cloud infrastructure. [14]

The servers are the essential components of cloud computing infrastructure that are located in the cloud datacenters. Typically, they can be real or virtual servers. Dedicated or real server is also called “blades-mounted”, whereas the virtual server is known as “virtual server instances”. Both servers can carry out the same task, and with some variation such as cost and security. These causes help emerging four categories of IaaS [29]:

- **Public cloud “external cloud”**

Services and utilities are available for the general public, and are used in pay-per-use consuming model. Public clouds are run by the third party providers and their physical infrastructure and resources are often hosted off-premises from consumers. Such cloud reduces consumer risk and cost throughout, providing elastic and even provisional extension to enterprise infrastructure [15]. Common examples of public clouds are Amazon AWS, such as EC2, S3, Microsoft Azure, and Rackspace Cloud Suite.

- **Private cloud "internal cloud"**

A private cloud is built for the unique use of a specific organization or business that providing complete authority over data, maintenance, security, and quality of services. A business might be built and managed its own private cloud in-house or the operation may be fully done by third party providers on the premises [16]. As a result, enterprises and projects can share a pool of computing resources across applications, departments or business units. Unlike the public cloud, this model needs considerable up-front costs, continuous maintenance, hardware, software, datacenter, and internal expertise also [13]. Therefore, private clouds are considered as the secure style of IaaS. [29]

- **Community cloud**

A number of organizations share the cloud infrastructure to serve a common function or purpose alternatively share similar concerns such as security requirements, policies, missions, regulatory compliance needs, and so on [16]. Additionally, a constituent or a third party can manage the community cloud. [14]

- **Hybrid cloud**

Finally, combining multiple public and private cloud models by standardized technology to distribute applications across them is known as hybrid cloud. The fact that hybrid clouds have the ability to spread applications and data from one cloud to another [17]. This model can be applied by many enterprises that use public cloud for general computing while customers' data is kept within a private cloud [13]. Amazon Virtual Private Cloud, Skylap Virtual Lab, as well as CohesiveFT are the most popular examples of hybrid cloud and offer hybrid cloud services.

2.1.6 Cloud computing benefit

- **Reducing investments and proportional costs:**

From a financial point of view, using in cloud-based IT resources minimize up-front capital cost because consumers rent operational resources such as hardware, software, and ownership cost instead of purchasing them. In reality, decreasing costs are derived from the deployment and operation of a large-scale datacenter cloud provider. Cloud datacenters normally exist in a real geographical location with IT professional. Low cost network bandwidth leads to decrease the costs and operational saving as well. Likewise, multi-tenancy sharing of infrastructure has great effect, for example, operating systems, and platform and application software. [7]

Additionally, “pay-as-you-go/use” method offer significant benefit to organizations, it is cheaper to pay just for what they use and require. Moreover, cloud services reduce the need for business to maintain in-house experts IT management cost, no need for technical administrative personnel. [13]

- **Increased scalability:**

From a technical point of view, built-in feature in the cloud is offering a flexible level of scalability in which IT resources are allocated on-demand to consumers dynamically, and similarly they can be scaled up/down or released automatically or manually. So, your cloud computing system grows or shrinks relying on your necessity. Figure 3 gives a simple example of changing demands of a cloud company during 24 hours a day.

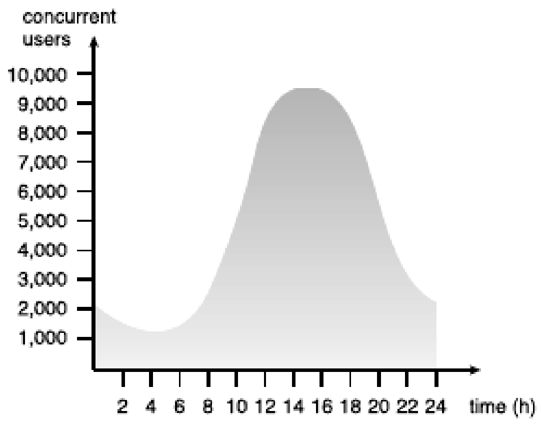


Figure 3 An example of a cloud company changing IT resources demand [7]

- **Increased availability and reliability:**

Providing cloud services from different geographical location help supporting business continue and disaster recovery. Moreover, load balancing and failover make IT resources highly available and reliable. So, cloud providers are capable to increase quality of service guarantees to consumers [14]. Other benefits can be gained, such as mobility, and reduce run and response time.

2.1.7 Risks and challenges in cloud computing

- **Security, privacy, and trust**

Information security is a primary affair in cloud computing. Cloud solutions are fundamentally public, therefore the systems expose to more attacks. As stated in the Berkeley technical report about cloud computing obstacles "*we believe that there are no fundamental obstacles to making a cloud-computing environment as secure as the vast majority of in-house IT environments*", however, the cloud may use technologies such as encrypted storage, VLAN virtual local area network, firewalls and packet filtering. [21]

Beside security, the trusted cloud provider is another fundamental issue which insures the applications' level of privacy, for example, Amazon web services (AWS) provides more secure plane to sensitive data encrypt and protect. [22]

Additionally, legal and regulatory need attention, where data are geographically located determines the ability of nations to access data. Also, some countries do not authorize some cryptography techniques; as a result, some consumers such as a bank or healthy care companies would not be comfortable putting their hot data in the cloud [4]

- **Data lock-in and standardization**

Consumers face many obstacles such as the difficulty of extracting or moving their data and application from one provider to run on another, because of the lack of standards and interoperability among providers. Till now, the standardization environment for cloud computing is just beginning to develop. Many efforts in USA, Europe and German institutes try to open standards. Figure 4 shows standardization environment in cloud computing, and a summary of the rollout of standards from 2010 is shown in Figure 5

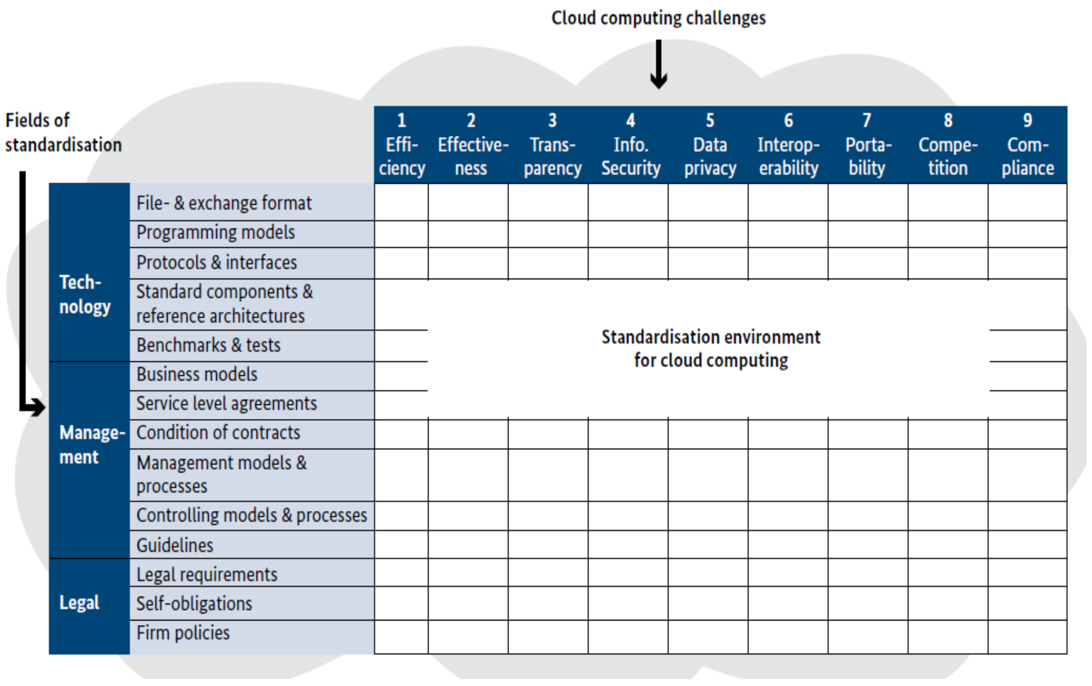


Figure 4 Cloud computing standardization environments [23]

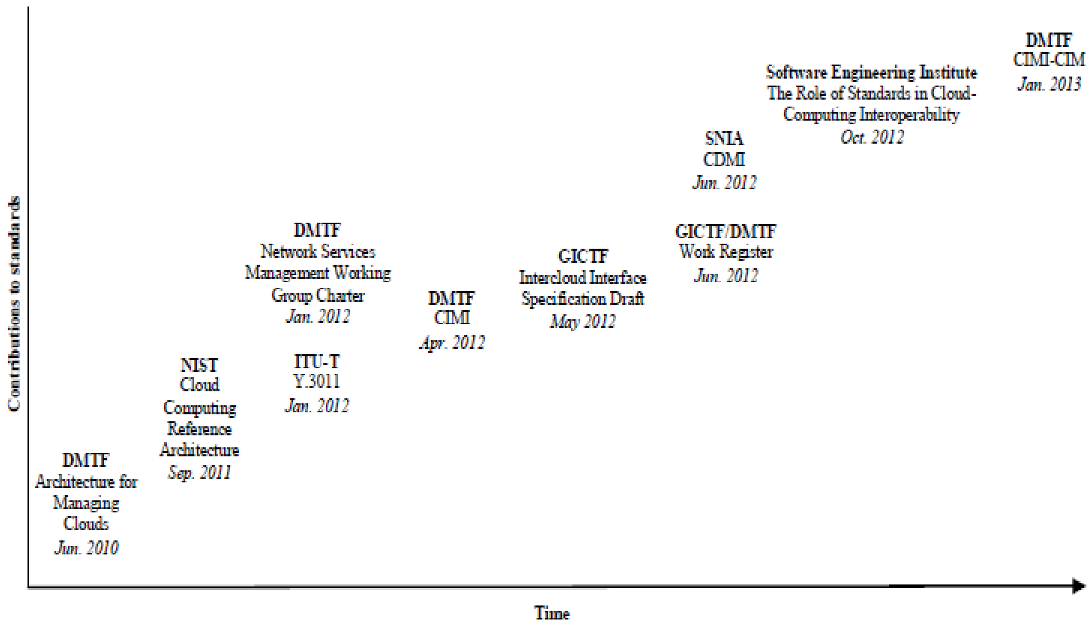


Figure 5 Cloud computing standardization rollout [24]

- **Availability, reliability, and fault-tolerance**

The Cloud computing based on the reliability and availability of open and access to computing power. These aspects consider as advantages; however, they arise open discussion because failure and outage in cloud datacenter lead to financial lost [25]. In addition, practical and efficient management of virtualized resource pools, physical resources and economic value should be carefully considered. Figure 6 distinguishes some of them.

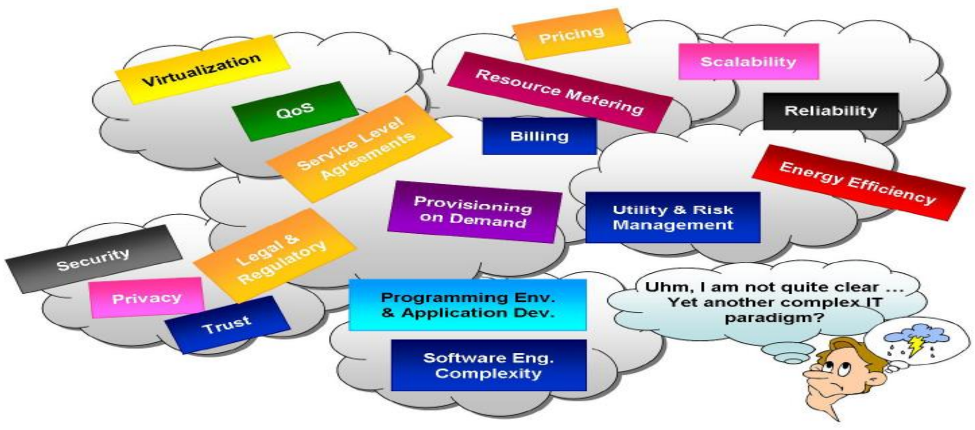


Figure 6 Cloud computing challenges, [28]

2.1.8 What is service level agreement (SLA)?

SLA represents a formal business contract which is established between consumers and cloud providers to define specifications and requirements of the cloud outsourcing services. The SLA describes the basic facts about the services to be provided such as the price of the services, level of security, resource metering, billing, as well as penalties for breaking expectations. However, SLA plays as guarantee for consumers [79, 82]. SLAs identify and guarantee the performance level in term of availability of the cloud services. In reality, the quality of service (QoS) within traditional web service is fixed, whereas the cloud QoS is usually unstable over the long run of time due to failure within the cloud environment, weak management of SLA, or monitoring of the resources. [27]

Generally, SLAs guarantee fill into two types [26]:

- Cloud infrastructure SLA (I-SLA): cloud service provider responsible for ensuring the quality level of cloud service for cloud customer, such as availability, storage capacity, network, and resource performance.
- Cloud-hosted application SLA (A-SLA): the guarantee and the amount of quality of the customers' solution for the end users.

Three main sides affect the guarantee of the cloud service availability and can help customer trusting the cloud service provider [81] :

- Fault tolerance techniques of the cloud services.
- Redundancy of the cloud service.
- Size of the cloud datacenter.

The major challenges affect SLAs are [79]:

- The way to monitor SLA dynamically.
- Setup trust level is very complex.
- The customer trust and selection of the cloud service provider is difficult to be defined.

2.2 Basic Fault Tolerance Concepts in Cloud Computing and Related Works

2.2.1 Definition of fault tolerance

Fault tolerance is an important key issue in cloud computing. It is the means to guarantee the availability and reliability (dependability) of critical cloud services as well as applications' executions and fulfilling their functions correctly even in the presence of failure affecting a system resources (e.g. hardware, software, network, overflow, timeout, power, or database lose). Basically, fault tolerance techniques are employed through the procurement or the development level of the system, so that, it is a survival attribute of cloud computing systems to satisfy the QoS requirement which are offered in SLA. [30, 31, 32]

2.2.2 Achieving dependable system

Fault tolerance aims to achieve dependability in a system. Therefore, system dependability is an essential objective of fault tolerance. Generally speaking, dependability is justified the trustworthiness of a system to deliver services to its customers. Figure 7 clarifies the tree of the major dependability characteristics: [33]

- **Dependability attributes**

The dependability tree illustrates its attributes; two primary attributes are reliability, and availability:

- **Reliability**: is the continuity of delivering correct services without disruption like loss of data or code reset during execution; it is a function of time so that it is related to the MTTF Mean Time to Failure and the MTBF Mean Time Between Failure. Moreover, MTTR Mean Time to Repair is the difference between MTTF and MTBF. As a result, MTBF=MTTF+MTTR. [30]. Inherently, reliability in the cloud is often covered by a service level agreement (SLA) as well as it is software-based solution not hardware-based.

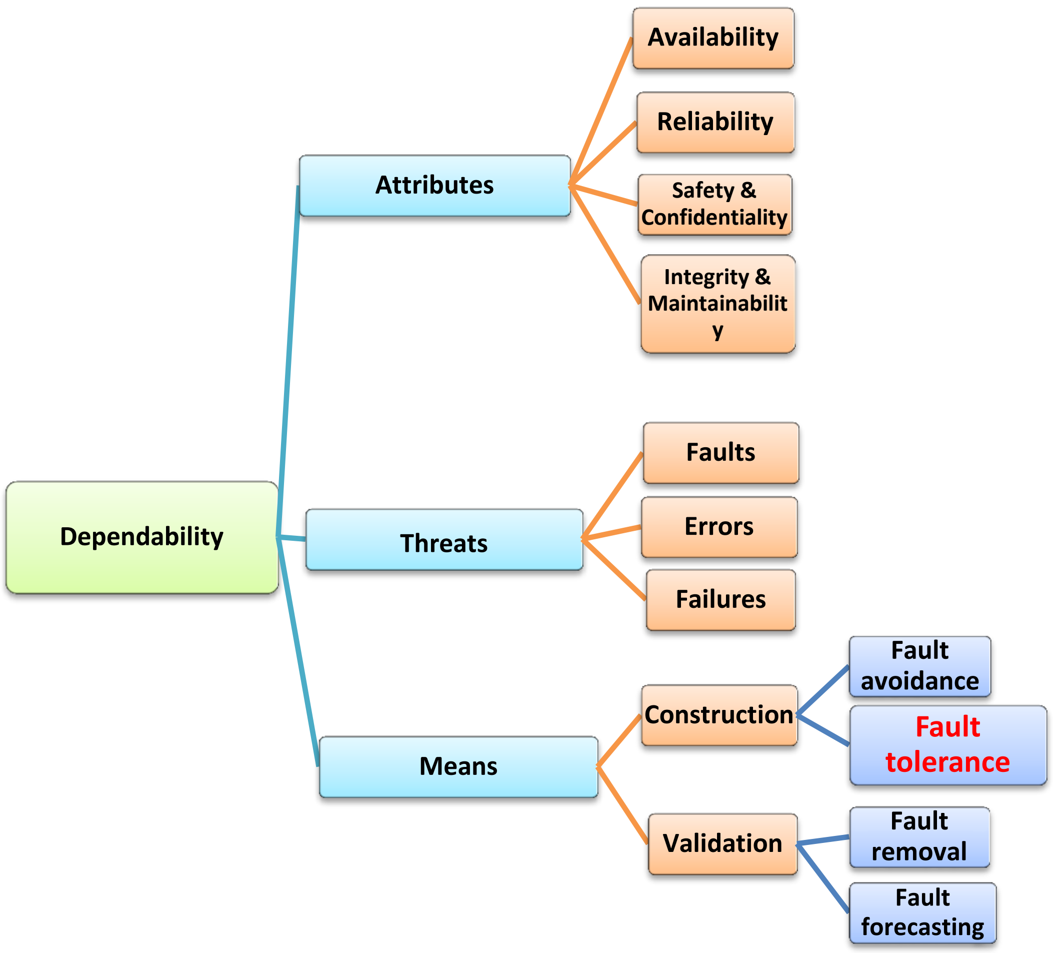


Figure 7 Dependability tree [30, 32]

- **Availability**: can be defined as the immediate readiness of the system to perform the services or the tasks when it is asked to do it. Availability = (MTTF) / (MTTR + MTTF) [34]. Basically, availability in the cloud can be attained by redundant throughout the replication of services or data and spreading them across various resources [35]. System availability is measured by downtime per year. Table 1 shows standard values of availability and their corresponding downtime [33]

Availability	Downtime/ year	Downtime/ month
90%	36.5 days	72 hours
95%	18.25 days	36 hours
97%	10.96 days	21.6 hours
98%	7.30 days	14.4 hours
99%	3.65 days	7.20 hours
99.5%	1.83 days	3.60 hours
99.8%	17.52 hours	86.23 minutes
99.9%	8.76 hours	43.2 minutes
99.95%	4.38 hours	21.56 minutes
99.99%	52.56 minutes	4.32 minutes
99.999%	5.26 minutes	25.9 seconds
99.9999%	31.5 seconds	2.59 seconds
99.99999%	3.15 seconds	0.259 seconds

Table 1 Standard Availability Values

Clearly, to achieve High-availability assurance of the cloud services, datacenters should characterize by:

Failure-isolated zone: is called Availability Zones in Amazon EC2. Users who practice IaaS can easily know where their application instances are located. These geographical locations of cloud datacenter are known as zones, so that failure in one zone is isolated from the other. Hence, distributing the users' application instances among the multiple zone improve the availability.

Automatic scale-up: depends on to the users' request, the cloud automatically starts and stops VM instances. This function is useful to gain the high-availability of applications in case of running server process failures.

Configurable load balancer: dynamically configuration of load balancer in distributing the requests to a different zone assists fulfilling high-availability. [36]

- **Dependability Threats**

On one hand, a fault is software or hardware failing, defect, shortcoming, or flaw that leads to incorrect or inaccuracy value on the computational level which is called an error. On the other hand, a failure is a system problem that is caused by the error. In other words, Faults are sources of errors and errors are sources of failures. So that, a fault is the root of failure. [37]

Possibly, faults in cloud computing may exist in four main places:

- **Provider-inner faults**: redundancy, backup or stop and restart services are prevalent methods recovering services from failure.

- **Provider-user**: network congestions, hacker attack, browser collapse, time out of the request, or malicious errors can cause faulty nodes.

- **User-across**: sharing critical resources among users throughout may cause chaos in a cloud computing system due to unsafe access to the resources. [38]

According to the cloud architecture, common faults can be occurred due to:

- **Datacenter hardware failures**: processor, hard disk drive, integrated circuit socket and memory [39].
- **Datacenter software faults:** lead to application failures

  From the high level model, faults in cloud datacenter can be classified into: power supply failure, network failure, server failure, and management software failure (e.g. VM management, virtual infrastructure management, cloud management) [40]

A different way to classify faults based on how a failed component behaves in a cloud computing environment, typically, faults in the cloud can be classified into: [37]

- **Crash faults**: totally due to either stop functioning of the system components or never return to a right condition (e.g. hard disk crash, power outage).

- **Byzantine faults**: it is malicious fault that leads the system components behave arbitrary or maliciously as well as producing incorrect and different output values.

- **Dependability Means**

Four techniques are used to achieve dependability. On one hand, means that are deployed during the process of software configuration or construction with the goal of providing trust services to customers [41], they are:

- **Fault tolerance**: it is the subject of this thesis that aims to avoid an application or service failures' events if a fault happens [42].

- **Fault avoidance/prevention**: aims to prevent or reduce faults from occurring or introduce as possible. [43]

On the other hand, techniques are deployed after the software development with the ability to reach confidence [32]:

- **Fault removal**: detects and eliminates the existence of fault during the development and operational life of the system.

- **Fault forecasting**: Evaluates estimates or ranks the system behavior during the present or activation of faults.

2.2.3 Approaches to fault tolerance in the cloud computing

Fault tolerance solutions base on redundancy. So that, redundancy is one of the keynote principle for supporting availability and reducing the risk of single points of failure. [44]

A diversity of resource redundancy techniques and structures exist for fault tolerance mechanism which can be categorized into four essential categories: [45]

Spatial redundancy: extra copies of the hardware computing resource can be added to find out and overcome the impact of component failure. In advance, static, dynamic, or hybrid hardware redundancy can be used depending on the system structure complication. [30,46]

Temporal redundancy: also it calls time redundancy, by which additional time can be used to re-execute the same task several times on the same resources in the event of a failure. Whilst time redundancy has an advantage for detecting fault effectively with a low hardware cost. [45]

Information redundancy: additional information which is called check bits are added to the original data help protecting from soft error. This form of redundancy needs hardware redundancy to process check bits.

Software redundancy: extra software is added to override software failure.

2.2.4 Redundancy strategies in cloud computing

Interestingly, managing and recovering failure in cloud form is different from what is happening in traditional datacenters. The redundancy in the cloud is managed and formulated in the software instead of hardware components. For example, Amazon EC2 offers a wide variety options such as direct failed virtual machine to another virtual machine image VMI , replications ,or live migration and all failed virtual machine configurations are same to original. [47]

In most current cloud, checkpointing and replication are the two common fault tolerance redundancy strategies which are used in case of a system outage.

- **Checkpointing/Restart (C/R) recovery:**

C/R is the typical technique to tolerate failure on unreliable systems. By saving a snapshot of running application on a stable storage periodically so as to restart the application from a latest checkpointing image in case of a crash. [48]

Various types of checkpoiting strategies had been investigated by researchers, but there are three popular checkpointing fault tolerance strategies which are used in cloud computing as shown in figure 8[49]:

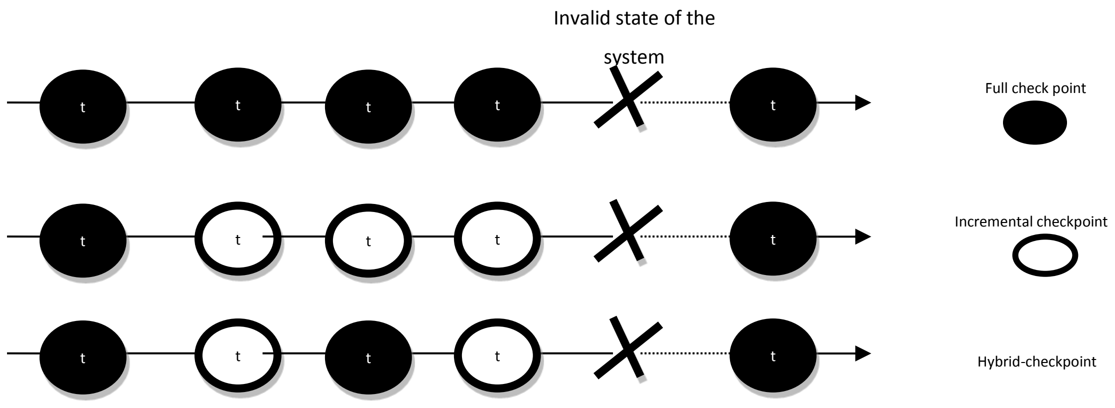


Figure 8 Checkpointing types [49]

Full checkpointing: is a traditional mechanism which saves the total state of the application or the system periodically to a storage platform. The drawback of this mechanism is the time which is consumed to make a snapshot of a whole system. And also the consumed of a large storage to save the whole system running states. [50]

Incremental checkpointing: typically the first checkpoint is full while the subsequent checkpoints only save pages that have been modified. This procedure produces a large recovery overhead due to the system must recover from the starting checkpoint. [51]

Hybrid checkpointing: is a combination between the full checkpointing and the incrementing strategies. Therefore a balance between the checkpointing overhead and the fault recovery overhead should achieve. [49]

As a result, the efficiency of the checkpointing mechanism depends on the length of checkpointing time [50]. Hence, "*How often should one insert checkpointing and saves system running state?*" as well as "*Which checkpointing strategies should be* *selected in the cloud?*” [49] “*Are a complicated decision that should be solved*”. It is based on the knowledge about the application and the system as well.

- **Replication**

The availability of replicated resources is a key requirement for the forming of fault tolerant systems in the cloud. Simply, replication means several copies of an application with the same input-set are executed simultaneously on alternative sites [52]. For example, proxy server and the caching in web browser can be considered as a form of replication. The essential goal of replication is to guarantee that at least one replica can complete the task correctly in case of the others fail [30]. More than one replication mechanisms such as active, passive, or semi-active have been used in the cloud computing.

Basically, Hadoop, Haproxy(high availability proxy), and Amazon Ec2 are the common tools that are implemented in order to manage replicas on different locations[53]. Moreover, three important problems should be addressed to achieve efficient replication:” *Which data should be replicated?*”, “*When to replicate in cloud systems*”, “*How many suitable new replicas should be created in the cloud?*”, and “*Where the new replica should be placed?*” [49]

2.2.5 Fault tolerance policies in the cloud

Typically, fault tolerance policies in cloud computing depend on various fault tolerance techniques that are implemented either during task-level or workflow level failure. Cloud fault tolerance planning can be classified into two groups:

- **Reactive/Post-Active Fault Tolerance Policies**

Reactive fault tolerance is a readily responsive to failure of application execution when it effectively occurred so as to minimize the impact of failure [54]. This approach is often selected by fault tolerate systems and different techniques are implemented on this approach: [55]

- **Checkpointing/ Restart:**

Is outlined above in section 2.2.4

- **Replication**

Is outlined above in section 2.2.4

- **Job Migration**

Moves a job’s state from a particular machine (node) to another when a node cannot be completely executed or processed the tasks. However tasks can be migrated by using HAproxy tool and load balancing. [56, 54]

- **S-Guard**

A fault tolerance technique is used for distributed stream processing engines (SPEs) such as email, online games, e-commerce, instance messaging, and search. It is relied on a rollback recovery that capable of checkpointing the state of stream processing nodes recurrently and restart failed nodes from last checkpoint. [57]

- **Retry-It**

Repeatedly, tries the failed task on the same cloud resources. [58]

- **Task Resubmission**

A new technique tries to re-execute the same task whenever a failed task is detected and is utilized during the workflow execution phase. However, a task may resubmit either to the same or to the different resources low at run time without interrupting the workflow of the system. [59]

- **User-Defined Exception Handling**

The ability of allowing users to specify the defined exceptions to handle task specific failures relying on the context of the task, as well as, the ability to define customs procedures to handle these errors as well. [60]

- **Rescue Workflow**

  This technique neglects the failed tasks and continues to execute the rest of the workflow tile nothing else can be made after that a rescue workflow description identifies failed nodes. [61]

- **Timing Checks**

  An error detection technique which supervises the liveness of critical functions as speed monitoring, and obstacle avoidance timing check are implemented by watchdogs in order to detect just timing error not value errors. [43]

- **Proactive Fault Tolerance Polices**

Proactive fault tolerance is a recovery service that is able to tolerate up faults and errors by using an avoidance mechanism throughout predicts the failure and proactively replacing doubtful resources with other. In other words, it is a migration of running applications from nodes that are predicted to fail to other resources and continue execution from a last checkup. [53]. This can be attained by depending on the system log (e.g. reliability and availability) and health monitoring using sensors which provide information about hardware or software status. [62]

Many challenges can be created by this approach such as fault prediction, prediction accuracy, or implementing efficient techniques as well as it requires a deep knowledge and tracks the system’s data [55]. Even so, proactive polices is important because reactive fault tolerance is not appropriate for some types of execution platforms such as large scale systems that read/ write a large amount of data and need computes’ speed (e.g. financial trading, health care, air traffic control systems, large enterprises, extra). Some of the mechanisms which are based on proactive policies are:

- **Preemptive Migration**

  Preemptive migration is an avoiding failure technique base on a feedback-loop control mechanism where an application is constantly monitored and preventative action is taken to avoid application failure. However, not all failures can be expected and covered by preemptive, so that combination of pro/ post-active fault tolerance technology will provide a sophisticated mechanism. [63]

- **Software Rejuvenation**

Rejuvenation is a cost effective and proactive fault management technique for a software application that is continuously running for a long period of time which increases the risk of application failure or degrades reliability and performance. So that, a software application is called software aging. Rejuvenation solution scheduling periodic stopping the running of a software and reboot it so as to cleaning and refreshing its internal state. [64]

- **Self-Healing**

Self-healing is the ability of a system that recognizes problems throughout automatically monitoring and detecting improper operation of software applications and transactions as well as making a decision without human intervention. [65]

2.2.6 Fault tolerance procedure

The process of dealing with faults in the cloud can be grouped into two distinct phases or steps:

- **Detecting Phase**

Detecting is the first, foremost, and starting point of the fault tolerance using checking technique of the application system state in order to detect all possible errors in the behavior of the system. Moreover, this step ensures that the effect of the errors is limited and failed components should be isolated so that its effects are not propagated further [66]. The most common error detection techniques are:

- **Replication Checks**: comparing the output of application replicas help detecting of an error in one or more components. This checking is often used in hardware in form which is called Triple-Modular Redundancy (TMR) or in software form such as N-version Programming, distributed recovery block. [34]

- **Temporal Checks**: this is used for detecting timing faults. Typical techniques are timing checks, and safety-bag checks

- **Recovery/Repair Phase**

Recovering is the procedure of restoring the system from an error to a valid state. There are two general forms:

- **Backward Error Recovery**: bringing the system back to a previous consistent valid state. In other words, undoes the execution action and comes back to a known system state. Checkpointing mechanism is a common backward mechanism. [66]

- **Forward Error Recovery**: directing the system from the erroneous state to a new valid state. [34]

2.2.7 Related works

DAFT: "*dynamic adaptive fault tolerance*" [49] is fault tolerance architecture ensured the fault tolerance service in cloud datacenter, which improve the cloud serviceability and achieve FT enhancements. Checkpointing and replication are the most current fault tolerance strategies that are used in this model. Analyzing the main aspects of checkpointing, balancing and combining its classes together help to construct an efficient dynamic adaptive checkpointing strategy. As well as replicating the data and considering the number of replicas and their locations determine a dynamic adaptive replication strategy. Merging the two models together raises the cloud serviceability and meets SLO. A dynamic adaptive fault tolerance strategy acts as an interface between Fault tolerance cloud environments and customers of the cloud.

VFT: *“A Virtualization and Fault Tolerance Approach for Cloud Computing*” [71] discussed a model which addresses fault tolerance in a hypervisor virtualization architecture inside a cloud datacenter. The VFT has the capability of decreasing the service time, also ensuring and raising the system availability. The Reactive fault tolerance mechanism with the replication and redundancy techniques is used to build the model. The models’ scheme implicates two stages. Cloud manager (CM) stage carries out hypervisor virtualization, load balancing, and performance recording (success rate parameter), then detect and repair faults throughout fault handler. The second stage is the decision maker (DM) which responsible for checking the nodes status and task deadline, then the algorithm will give the final decision to create checkpoint if the all checking gives correct result. The proposed model was evaluated throughout the analyzing of the success rate SR metric; whenever, SR is increased the scheme will achieve perfect performance.

Gomeza et al. [69] suggested virtual cluster architecture to tolerate faults in the cloud IaaS platform. Cloud sites failure can be recovered through deploying of virtual clusters on two different geographical locations. It is a significant method in high performance computing (HPC) applications, in case the whole or part of the site fails. This architecture is suitable for executing a particular application of a customer. It is totally independent, and can be deployed in one or several cloud providers. Another technique is added which monitoring the application performance periodically.

Feng et al. [67] proposed Magicube, A cloud storage architecture model has the ability to reduce the space cost of redundancy, improve performance and guarantee the reliability of the system. Unlike Amazon S3, high distributed file system (HDFS), and Google GFS storage system that needs 3 replications of each file as a default, Magicube system is conserved just 1 replica of a file.

The model architecture based on HDFS for storage, retrieve files as well as managing and storing metadata of files. Moreover, it provides FT mechanism by adaptive encoding algorithm that has the ability to encode and splits the source file into many parts. In case the main file lost the algorithm use the metadata and number of splits to recover the file. This paper tries to prove that Magicube performance is better than HDFS, WordCount and Grep benchmark is used to test the performance.

FTCloud [68] introduced a framework that provides fault tolerance software system (cloud application) in the cloud using algorithms. The first algorithm ranks critical and significant components in the cloud whose failures have an influence on the entire system. The ranking algorithm is based on the available information such as application structure, components relationship and characteristics. The second algorithm has the ability to select the optimal software fault tolerance strategy among (Recovery Block, N-Version programming, parallel). The selection of optimal strategy relies on basic metrics like response-time, required resources and fault tolerance. FTCloud framework constructs a high reliable cloud application which can tolerate unexpected system crash and value faults.

Bala and Chana [58] addressed fault tolerance policies, techniques and challenges. which are:

- How to employ independent fault tolerance (FT) techniques that can work for multiple VMs to run and execute an application.
- How can integrate various cloud components to find a reliable system.
- The capability of developing workflow scheduling algorithm to implement FT method.
- Evaluating the fault tolerance's performance via comparison with another model.
- Challenges of guarantees dependability.

In addition, the survey proposes Cloud Virtualization System Architecture that can tolerate faults using the reactive policy. Three VMs is used, Haproxy server works as fault monitoring through directs the request to the other VMs which execute the same application. If the master server goes down, the customer request will be directed to the backup server.

Egwutuoha et al. [70] defined a new cloud computing capability of employing fault tolerance for computation intensive application which needs a lot of computations and acceleration to perform the task such as medical imaging, bioscience, or financial trading. A framework of a high performance computing system which depends on a proactive FT strategy to predict the fault, reactive FT uses checkpointing to reduce the cost of execution time, live migration and FT protocol for communication as well.

AFTRC: "*Adaptive Fault Tolerance in Real Time Cloud Computing*" [72] introduced a fault tolerance model that offers a recovery solution for real time cloud application. Nowadays, real time audio (RTA) applications are widely used such as VoIP, e-commerce transaction which requires a specific period of time to be executed and originally need fault tolerance method to ensure that the application continues working even if a fault occur. A scheme that tolerates fault using redundancy method (nodes replications, snapshot of valid state) and a software to monitor the processes, is applied. The FT fulfils based on the VMs reliability. RTA FT in the cloud is important because any fail leads to financial loss. The characteristic of this model provides automatic forward recovery by reliable nodes as well as backward recovery throughout snapshot technique.

Fault Tolerance as a Service (FTaaS) [73] proposed a formal model that presents a service by a cloud vendor to provide fault tolerance guarantees to customers' applications. FTaaS functions as an agent to the lowest level of service in cloud computing, which is IaaS. It is provided as a part of SLA with two fault tolerance strategies: spatial redundancy which depends on majority-voting and temporal redundancy, which is based on checkpointing mechanism.

FtaaS allows customers to specify their requirements, such as the number of VMs, FT strategy as well. The customer must assign FT strategy as a function of time. This means that SLA for example divided a day into small time-slots and gives choice to the customer selection the fault tolerance method of each slot. In summary, FTaaS aides cloud services to be adapted to customer requirements, resulting in the possibility of attracting a large number of users and maximize revenue as well.

The authors in [74] present a framework that reactively creates and manages fault tolerance at system-level with the ability to detect and overcomes crash failure amongst VM instances as a result of network, server, or power failures. This work offers fault tolerance properties to the users' applications as an on-demand service by a cloud computing service provider or a third party to supply fault tolerance by replication, live migration, detection, masking of failures, recovery mechanism, messaging monitoring protocols, as well as a client/admin interface which even helps users with non-technical knowledge to easily configure the system properties.

Zhang et al. [75] proposes a Byzantine fault tolerance (BFT) framework which is built under voluntary-resource cloud infrastructure consisting of 257 providers located in 26 counties not a well-provisioned and well-managed infrastructure a large service provider. Those types of cloud resources are heterogeneous, so that crashing, network faults, and byzantine faults could be common due to the different geographical locations, operating systems, network environments, and implementing software among nodes. BFT offers a practical and reliable fault tolerance system in tolerating all these failures. Replication techniques are employed for requests execution. Whilst, a monitor is deployed as a middleware for controlling the QoS's performance and probable nodes' failure.

In workflow level, **Fault Tolerant Workflow Scheduling algorithm (FTWS)** in [76] tolerates up faults in users' applications that are represented as workflows executing in the cloud computing environments. However, failures in workflow like link faults, server faults, malicious code (byzantine) in the executing nodes, or datasets required are looked by other tasks, etc., minimize the system robustness and reliability.

Typically, a workflow is a sequence of tasks processed in a specific order, therefore scheduling tasks of the workflow within a deadline in spite of failures is the main goal of this work. It provides fault tolerance by making use of replication and resubmission of task as well. If all replicas fail, then workflow resubmits the task. Additionally, scheduling tasks to meet deadline balances between replication and resubmission mechanisms, throughout reducing resource wastage because of replicas as well as decrease the makespan or resubmission the workflow.

Remus [77] is considered as a system that provides OSs high-availability as a service which are based on virtualization infrastructure platform. Virtualization guaranties the capability of creating a copy of a running machine as well as migrates running VMs to other hosts. Remus offers protection similar or better than commercial providers' cost throughout replicates snapshot of the entire running operating system approximately every 25ms to another location.

2.2.8 Summary

Firstly, The background information about cloud computing offered in this chapter, how experts and provider define cloud computing, origin of the cloud, the cloud underlying technology, cloud taxonomy, the benefit of cloud computing as well as the cloud risks and challenges. This survey considers as beneficial material for novice researchers who may not be well known the cloud phenomenon and it encourage them to deeply investigation. Secondly, the concept of fault tolerance, system's dependability, and how to manage and recover system failure were presented. Then the fault tolerance policies and techniques were introduced. After defining these concepts, the process of dealing with fault was discussed.

Finally, the relevant related works were presented in this chapter, which addresses the main investigating field in this thesis. First, a dynamic adaptive fault tolerance [49] service by cloud datacenter is introduced, VFT [71] address FT within virtualization architecture inside cloud datacenter have been discussed, FTaaS formal model adapted by cloud provider that guarantee customer application was presented [73]. Another fault tolerance property is discussed as on-demand service [74]. [77] Remus project is a high-available OS service based on virtualization that offers protection to customer’s application was introduced. A common architecture application over the cloud that uses load balancing (LB) and high availability proxy (Haproxy) technologies is discussed [58]. Magicube [67] a cloud storage architecture with replication tool was examined. [69, 70] and offer fault tolerance strategy for HPC applications were also discussed in this chapter. AFRC [72] provide a fault tolerance model that help serving the new generation of applications such as VoIP was also introduced. Tolerating byzantine fault of a system was discussed [75]. At last, providing fault tolerance within a workflow level of executing application over the cloud was presented.

CHAPTER 3

FAULT TOLERANCE COLLECTIVE STUDIES FOR COMMERCIAL CLOUD IaaS PROVIDERS

3.1 Introduction

To understand how fault tolerance nowadays operates in cloud computing services, a real-life study that explores how well it perform in Amazon web services (AWS) and Google Compute Engine (GCE) infrastructure as a service is discussed in this chapter. Both of these commercial cloud service providers consider as leading players in the market. Based on the knowledge gained by chapter two, we investigate in this chapter their critical fault tolerance strategies that can help enterprises and business applications optimally use cloud computing infrastructure services as well as support providing a general view of how to build high available application over the cloud.

3.2 Amazon Web Services AWS IaaS Fault Tolerance Architectures

2006 is the start point of Amazon Web Services. AWS infrastructure is shown in figure 9, which provides two different strategies that are suited for building and maintaining fault tolerance in distributed systems/applications. Many Tools and feature are provided within the service that capable of designing fault tolerance (FT), disaster recovery (DR), and high available (HA) systems. In other side there are Further infrastructure building blocks within the framework of Amazon services include fault tolerance solutions by default.

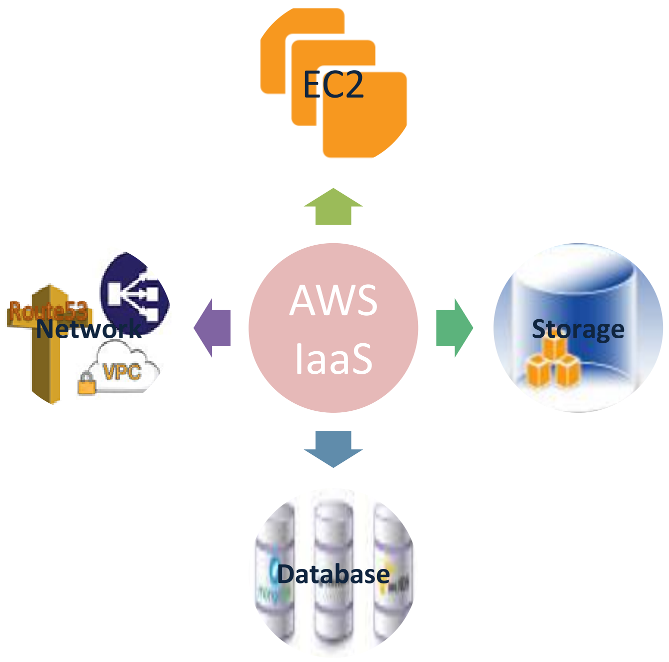


Figure 9 AWS infrastructure building block

- **Amazon Elastic Compute Cloud EC2**

  - At the core of Amazon virtual computing resources is an EC2 instance "virtual machine server". Amazon EC2 is constructed over multiple (eleven) geographical locations known as regions. Figure 10 illustrates global Amazon infrastructure.

  - To overcome failure in a region, each region splits into two or more Availability Zone (AZ). Table 2 clarifies AWS the number of AZs within regions. AZs are geographically isolated from each other in the same region, but they are connected with low latency network connection. As what is stated in chapter two, section 2.2.2 this feature increases reliability and availability.

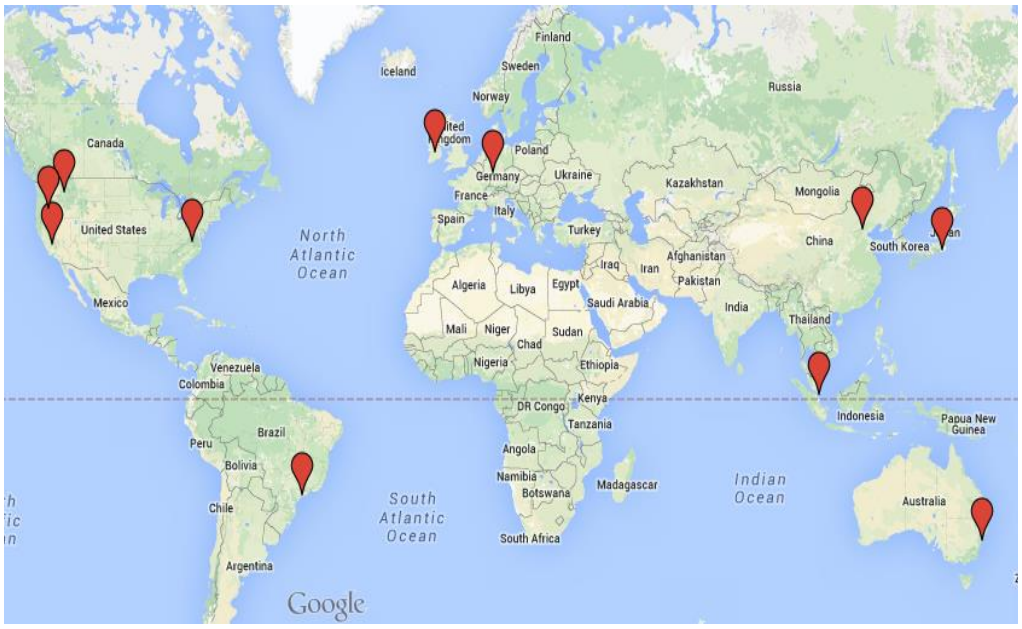


Figure 10 Amazon web services global infrastructure

Region	**Description**
US East (Northern Virginia)	5 AZs launched 2006
EU (Ireland)	3 AZs launched 2007
US West (Northern California)	3 AZs launched 2009
Asia Pacific (Singapore)	2 AZs launched 2010
US West (Oregon)	3 AZs launched 2011
AWS GovCloud (US)	2 AZs launched 2011
São Paulo	2 AZs launched 2011
Asia Pacific (Tokyo)	3 AZs launched 2011
Asia Pacific (Sydney)	2 AZs launched 2012
EU (Frankfurt)	2 AZs launched 2014
China (Beijing)	1 AZs Coming Soon

Table 2 AWS Availability Zones

- Each AZ has independent infrastructure (power, cooling, network and security) Hence, failure behind one AZ will not affect the others.
- Service Level Agreement (SLA) of EC2 claims offering 99.95% uptime for each Amazon EC2 region.
- The first step in EC2 is to launch a VM instance (23 types of instances) throughout, creating an Amazon Machine Image (AMI), it is a master template that helps to define instances such as (web server, application server, etc...).
- From one AMI multiple instances can be established. Moreover, AMI's instances can be scaled up/down.
- VM instances are not replicated automatically in the same or different regions, it is customers' mission and need their interaction.
- It is safety for critical application/system to keep a spare of an instance in different AZ and keeps it running, in case a hot instance fails, the activity is taken by just redirecting users' requests to a new instance. This action is done by using AWS other utilities which are discussed in chapter two, section 2.2.2, such as Elastic IP and Load balancer, in detail, they will discuss later in this chapter.
- For more availability AMI can be replicated in other regions to avoid failure in one region throughout using snapshot technique.
- An instance local storage is not persistent, so that other AWS infrastructure storage component such as Amazon Elastic Block Storage (Amazon EBS) is required. As a result the root device volume of an instance is Amazon EBS volume. Also, an instance hot data should store in Amazon EBS when the instance dies a customer can replace it by attaching the Amazon EBS volume to a new EC2 instance.
- Incremental checkpointing "snapshot" strategy is Amazon EBS redundantly feature that reduces the possibility of failure throughout automatically take an image of the instance volume.
- Replication feature is implemented in Amazon EBS, EBS volume data can be replicated with ease across multiple servers in a region AZs.
- Automatically, Amazon EBS snapshots are reserved in Amazon S3.

- It is recommended to create a snapshot frequently, so when the instance volume fails, a new volume is created from the last snapshot. Also, snapshot of a volume is at hand across all the availability zones AZs of a region.
- From time to time customer should recurrently renew their instances with new server instances to avoid resource wasted, software memory lost, hardware degradation, file system fragmentation, etc….
- Disaster tolerance for systems/applications is easily designed by using Geo-replication over multiple AZs.
- Customer pays for what they spend in hourly billing mode.

- **Amazon Relational Database Service (RDS)**

  - Amazon RDS is an infrastructure component that provides fully featured virtual database server on the cloud with the capability to online access and use customers' databases such as MySQL, Oracle, SQL server, or Postgre SQL).

  - Amazon RDS automatically snapshot data depending on retention period on scheduling time. The snapshot files are stored on Amazon S3 for more protection and availability.

  - A customer's database can be replicated in multi- Availability Zone AZs within the same region. There is no direct service to replicate in another region.

  - 99.95% uptime is stated in RDS's SLA.

- **Amazon Simple Storage Service S3**

  - Cloud storage solution that stores any volume (unlimited) of data in buckets from any place and at any time on the web such as static websites (media, images, HTML files, etc.…).

  - Amazon S3 depends on the replication method in which each bucket is automatically replicated at least in three Availability Zones AZ within the same region, this operation in Amazon the S3 is called standard redundancy, but if customers want to reduce the cost, they could use reduced redundancy class, however this action reduces availability.

  - SLA of Amazon S3 stated that S3 provides 99.9% monthly uptime.

- **Amazon SimbleDB**

  - Non-relational database functionality that creates, stores, and queries varied sets of data, with automatic administration infrastructure, hardware, software, etc.…

  - Geo-replication of data among multiple Availability Zones AZs within the same region is done automatically.

- **Elastic Load Balancing ELB**

  - Amazon ELB could distribute concurrently arriving traffic from users to multi-EC2 instances.

  - Better fault tolerance architecture in AWS can be achieved throughout ELB, which has the capability of monitoring the healthiness of Ec2 instances. If one server fails, ELB will route the traffic to the other running instances.

  - Amazon ELB route traffic across different Availability Zones AZs within specific region.

- **Amazon Virtual Private Cloud VPC " Virtual Datacenter"**

  - Amazon VPC supports launching AWS resources in a virtual network topology which is constructed by customers with entire governing over this network (IP address ranges, subnets, route table configuration, ports, VLANs, and gateway).

  - Cross-region recovery of customer's application is provided.

  - With all these critical features Amazon VPC assists on-premise datacenters at the time of disaster to switch and direct their enterprises in AWS.

  - Supports in-build datacenter with multiple geographically isolated availability zones tolerating faults.

- **Amazon Route53**

  - Amazon Route53 is the cloud base DNS server, which automatically answers users request to cloud EC2 instances, ELB, Amazon S3, or routes users to infrastructure outside AWSs.

- With Geo DNS, strong fault tolerance architecture can be configured, the healthy-check services which globally monitor and manage traffic to the just healthy and reachable resources by returning the IP address of healthy resources.
- Amazon Route53 SLA states that Route53 guarantees 100% uptime and availability.

3.3 Google Compute Engine GCE IaaS Fault Tolerant Architectures

Google Cloud Computing products released in 2008 which provides technology and services over the cloud. Generally, in December 2013 Google Cloud launched Google Compute Engine GCE as a new entry to the market providing IaaS. Figure 11 Clarifies GCE Infrastructure building blocks.

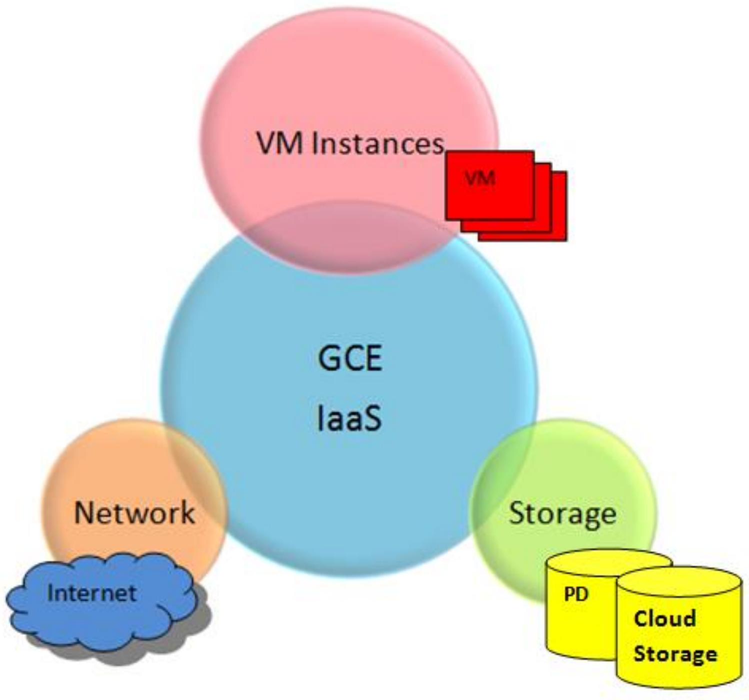


Figure 11 Classification of GCE infrastructure building block.

- **Google CE Instance**
  - Five different virtual machine servers can be launched for constructing systems and applications.

- Google group datacenters are distributed in regions worldwide, and each region consists of geographically isolated Zones which are connected with low-latency connection. The following figure 12 shows available regions and zones.
- All resources within a region can be accessed by zones within it throughout static IP address that is provided by default.

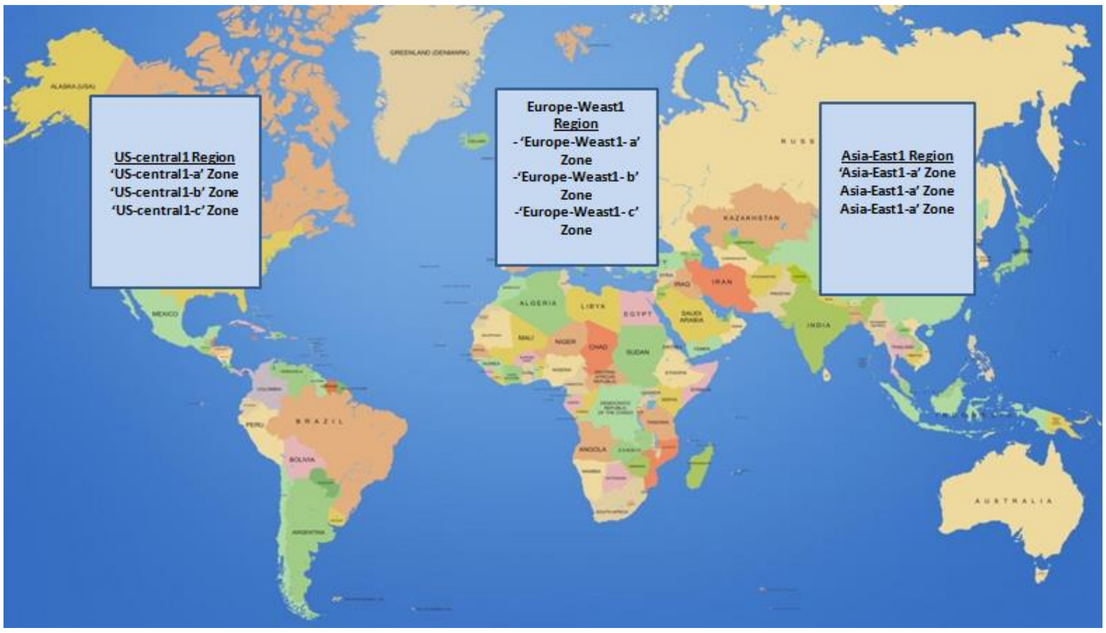


Figure 12 Google CE regions and zones

- Robust systems and applications need to distribute VM instances across zones so as to avoid unexpected component failures. In other side, putting instances in multiple regions achieve a high level of fault tolerance application.
- VM servers need constant disk to save data.
- Live migration feature is automatically set to all instances as infrastructure maintenance behavior.
- The Flexibility of using the Google Cloud Storage platform to back up hot data.
- Each month customer will pay just for what they use in minute billing mode.

- **Persistent Disk PD "Virtual Volume"**
  - Highly reliable, consistent, and high performance infrastructure solution with two types (standard, SSD) keeps data living.

- PD attaches to VMs so as to store boot device and critical data. In other words, it works as a physical HD.
- VM instance should attach to a PD in the same zone.
- Redundant strategy is applied by using different snapshot methods.
- A snapshot is a global resource, this mean PD snapshot can be applied to new PD volume in the same zone, different zone, or different region.
- Any PD type can practice a snapshot.
- It is preferred to save VM's data in more PD volumes.
- Many VM instances can be connected to one PD volume with read-only mode.

- **Advanced Networking Feature**
  - Per-region static IP address for an instance is specified by default.
  - Reserved static IP feature, so when a zone fails, it is easy to route traffic to another zone.
  - Global resource (public IP) determines how VM instances communicate with each other, with other network and with the outside world as well.
  - The Firewall can be applied and used by any network in the same system.
  - Service-side load balancing technology (Network load balancing in a single region, HTTP such as cross-region LB and content-based LB) support heavy traffic, detect unhealthy VM instances, and route request to the closet VM
  - Advance Routing is a new feature that helps connect in-premise datacenter and the cloud, build multi-cloud deployment, and VPNs.

3.4 Discussion of Amazon and Google IaaS Cloud Service

Investigate and explore Amazon and Google IaaS architectures is one of the objectives of this chapter. A clear understanding can be built from their profiles: VMs instances, the way of reserving customers' data, what is the advantage of the regions and the zones, network features, SLAs uptimes, billing mode, and so on. Comparing the efficiency of these features make sense which provider covers a customer requirement. It will be interesting to see that Amazon AWS, Google

Compute Engine, and if they compare to other IaaS cloud market providers such as Microsoft Azure, HP, IBM, Rockspace, they will be similar in their main infrastructure as a service components. AWS and GCE try to offer high-quality and high-availability service, resilient infrastructure, redundant infrastructure, easy accesse to resources, independent storage, create VMs snapshot, and high-level of network options, at least offer 99.95% monthly uptime as well. Note, however, that they use common fault tolerance strategies which are checkpointing and replication strategies to survive customers' applications and enterprises. Some differences are in their efficiency of infrastructure resources, for example (Google PD volume has the capability to be attached to multiple VM instances billing scheme mode is applied in Google CE versus hourly billing scheme within AWS, live migration of VM instances in case of a datacenter maintenance is done automatically in Google CE, AWS datacenters spreads across eleven regions, while GCE has three regions as clarifies in figure 3.3, and figure 3.5. Google Compute Engine tries to compete with AWS to offer new features and capability such as database and machine image.

Clearly, failure happens all the time and the cloud vendors design geo-location to tolerate regions' failure, more facilities help surviving AZs, network problem can be solved by load balancing technique, IPs, DNS routing, and VPNs, and deploy multiple instances overcome instance failure. Therefore, the critical key point to success deploying services/enterprises over the cloud is that how can customers design and deploy effective architecture that faults mitigate and avoid real risk for their businesses. Accordingly, in the following section a general conceptual model is proposed to help designing and architecting a solution with fault tolerance consideration for customers' applications over the cloud infrastructure.

3.5 Fault Tolerance Consideration for IaaS Cloud Computing Application

In reality, few works concentrate on a good understanding to build and setup cloud strategy of a solution in the cloud computing environment. In the cloud context, design a solution strategy is a complex decision- making process which needs strategic and well-ordered methods and consideration to be taken into account figure 13 showes the main critical point of fault tolerance concideration :

- **Build Fault Tolerance Strategy within a Workload Definition**

The workload is a key process for understanding service/enterprise architecture and describing exactly what will be deployed over the cloud virtual infrastructure. According to NetApp research [78], a successful solution needs to address the real application's workload elements, requirements, technical sides, and metrics such as:

- Availability,
- Design solution costs such as resource cost.
- Security, regulatory, and privacy demands.
- capacity,
- Infrastructure resources are the essential requirement to run the enterprise in pay-as-you-go method. As stated in chapter two, SMEs need to specify which right deployment model is suitable for their solution cloud strategy.
- Office suites those are delivered and accessible over the internet. For example e-mail, calendar, document editing service, communication features such as instance messaging, audio/video calling, web conference, and spreadsheeting as well.
- Financial consideration affects workload, such as the IT capital budget.
- Business services such as Enterprise Resource Planning (ERP), finance and accounting, Customer Relationship Management (CRM), Human Resources (HR), Payroll, as well as Project Management System.
- Data structure as SQL or NoSQL.
- Setup fault tolerance policy and strategy which was Discussed in detail in chapter two, section 2.2. This characteristic depends on the type of the enterprise. Is it traditional solution or a new generation solution?

In essence, the workload should be built on the idea that a web application failure happens all the time, thus enterprise developers should think about how can they choose the convenient methods that fit the workload as well as they should determine the workload type is it for a client-server/traditional application or a new generation application (gaming, mobile app, HPC, social app). Moreover, a cloud application has other sub-workloads, for example, a web site may have search and browse workload, and registration workload.

- **Build up Life-cycle Development Model**

Clearly, after establishing an application workload operational side should be designed within a specific period of time. According to Marks and Lozano research in their book [80] life-cycle describes the process of plan activity in a systematic manner throughout establish the milestones in which key scope decision are made for start and end point of the system, team responsibilities, gives direction, and identifying the availability of resource demands. As a result, the life-cycle includes tracking expected behavior of an application at different time and life-cycle stages such as planning, analysis of the system, system design, implementation, testing, and maintenance as well. Next step is to draw charts (weekly, annually) for a time against usage, availability, or capacity by taking entities, aspects, and conditions in their consideration. In this way, recognizing the life cycle will help determine the next step.

- **Identify Availability Plan**

The major important part is to assess availability plan in business infrastructure. What is the level of availability demanded over the life cycle of the workload should be defined. In terms of the SLA, studying carefully the cloud provider's SLA and identifies the applicable SLA which is needed. Which resource requires 99.9999% or 99.95% uptime because 99.9999% availability requires high-cost investment. The main key points of the SLA were clarified in chapter two, section 2.1.8

- **Recognize and Identify Failure Mode**

The availability of workload is affected by the failure. Therefore, understanding, identifying, and documenting failure will precisely insure building the map of failure points.

- **Understand the Architecture Patterns**

Establishing robust architecture help determining strategy for high availability so as to deliver a health application throughout many facets:

- Communication between and within different workload parts result in handle and manage failures.
- Automated everything is valuable, so as to minimize the developers' mistake which minimize business risk, improve efficiency. For example, data automated mainly depends on the database technology and query type as well which help remove unnecessary data and replicate critical data that is needed for the core of the functionality.
- Identify critical resources expose to fail, which components need to replicate, snapshot, change/modify software, improve, or upgrade such as OSs.
- Trust cloud services that are provided by third parties.
- Implement right redundancy strategies. Several scenarios are clearly architected by Amazon Web Services and Google Compute Engine. Either provid full redundancy solution, however, it increases 100% of the cost, or partial redundancy in another region (cost-effective) solution.
- Traffic management is valuable to ensure that a request is directed to the appropriate node.

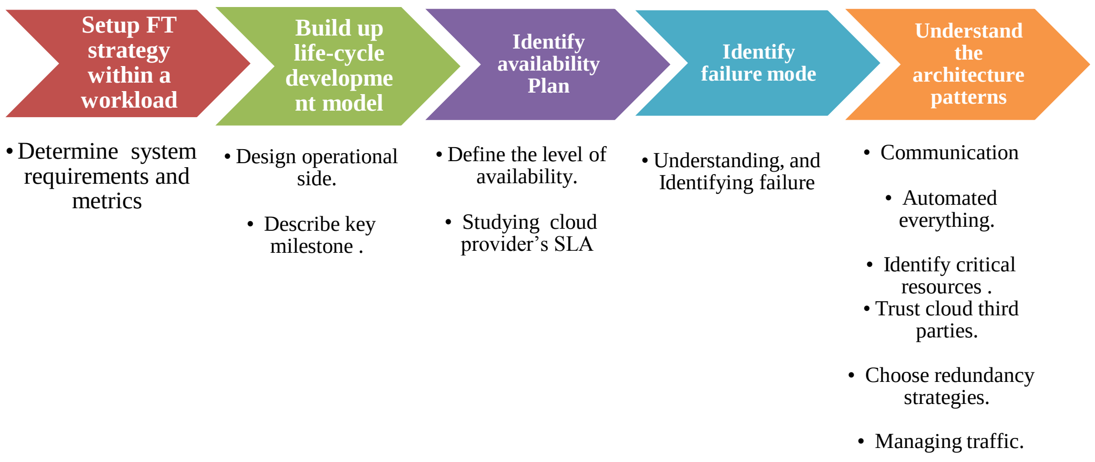

Setup FT strategy within a workload
Build up life-cycle developme nt model
Identify availability Plan
Identify failure mode
Understand the architecture patterns
• Determine system requirements and metrics
• Design operational side.
• Describe key milestone .
• Define the level of availability.
• Studying cloud provider's SLA
• Understanding, and Identifying failure
• Communication
• Automated everything.
• Identify critical resources .
• Trust cloud third parties.
• Choose redundancy strategies.
• Managing traffic.

Figure 23 Business fault tolerance consideration

3.6 Summary

In this chapter, we have learned the main feature of two IaaS cloud commercial providers. The first one is the oldest “AWS” which is considered as the shark of the market and the second is a new entry to IaaS market “GCE”. The focus of this chapter summarizes the infrastructure nature of those companies and how well they guarantee availability of their resources that assist systems and application deployment over the cloud.

To understand how to build fault tolerance service within the cloud a conceptual model is proposed in this chapter, which is based on understanding the main five critical points that help align the architecture and the strategy together to build a lifecycle model.

CHAPTER 4

CONCLUSION

4.1 Research Findings

- **Set up online solutions**

Generally, clouds computing introduce new online IT solution options for small and medium enterprises. It is an option to reduce entrance barrier to the market.

- **Low-level of abstraction**

Exploring Amazon and Google IaaS gives clear understanding that just the physical hardware underlying resources are abstracted whilst the level of abstraction is increased in PaaS, more and more high level "complete" abstraction in SaaS.

- **Manageability of the cloud infrastructure resources**

The cloud IaaS focuses on managing infrastructure resource in a perspective way so as to ensure service availability that helps enterprises to continue working.

- **Failure expected to happen**

From a cloud provider perspective, failures commonly happen all the time (hardware, software, outage, etc…..). Thus, the cloud SLAs provide an average value of availability which means that downtime can be happened.

- **Business continuity need detection from failure**

One of the cloud technical gaps is that business continuity is the customer's responsibility therefore there is a pressing need of fault tolerance strategy for enterprise IT components within the cloud to overcome the failures of VM instances, network traffic, and so on.

- **Typical use of redundancy strategy**

From an application/enterprise perspective, fault tolerance typically deals with checkpointing, replication, and live migration strategies to survive the business. Moreover, enterprise replication is not automatically done.

- **Map design consideration**

Ensuring a business design consideration may increase satisfactory value of availability and reliability.

4.2 Research Limitations

The goal of this work is to recognize the basic commercial cloud provider infrastructure architecture in term of failures and disasters detection, and the other objective is to look into the existing techniques and skills acquired that support businesses to build their application/system/service/enterprise with fault tolerance implementation at the time of development. Our proposed solution is not dealt with many technological details such as:

- Configuration failures, bug failure, and software failure are not taken into account, just the general basis of faults investigate.
- Scalability property is not discussed.
- How to manage and synchronize the distribution of enterprise infrastructure components across multiple-zones.
- The overhead of fault tolerance implementation is not counted.
- Limited time to examine more well-known commercial cloud providers.
- Teamwork is needed to fulfill an online questionnaire to analyze and evaluate how Amazon and Google CE case studies dealt with failure in their enterprises.

4.3 Future Work

The Future research path in the direction of fault tolerance will introduce possible evaluation and analyses the cost of implementing FT within Amazon Web Service AWS on-demand infrastructure.

4.4 Conclusion

In conclusion, cloud computing provides many features to small and medium business enterprises such as cost-effective of infrastructure resources, managing infrastructure, availability, and scalability. From a technical point of view, cloud services are commercial and their services are not designed to ensure the continuity of customers' application. In fact, they ensure the availability of their infrastructure and components which are offered to customers. As a result, fault tolerance property should be addressed so as to guarantee customer system continuity over the cloud. Based on the material of chapter two and the discussion of use cases a general conceptual model shows a pre-define strategy in which how an enterprise can be appointed into a workload so as to specify getting availability and failover. Accordingly, a life-cycle model and milestone could be established. After this step, specifying the nature of availability plan and establishing the map of expected failures are needed. All these key points should be considered to design a highly available and resilient architecture, where the customer could achieve a better fault tolerance plan. Section 3.5 discuss in detail the key lessons learned.

REFERENCES

1. **Mell P., Grance T., (2011)**, "*The NIST Definition of Cloud Computing*", http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf, (Data Download Date: 5 April 2014).

2. **Geelan J., (2009)**, "*Twenty One Experts Define Cloud Computing*", http://virtualization.sys-con.com/node/612375, (Data Download Date: 2 May 2014).

3. **Diversity Limited, (2011)**, "White paper: *Revolution Not Evolution How Cloud Computing Different from Traditional IT and Why it Matters*", CloudU.

4. **Voorsluys W., Broberg J., Buyya R., (2011)**, "*Cloud Computing Principles and Paradigms*", John Wiley , New Jersey, pp 33-60.

5. **Furht B., Escalante A., (2011)**, "*Handbook of Cloud Computing*", Springer Science Business Media, New York, USA, pp 381-476.

6. **Erl T., Mahmood Z., Puttini R., (2013)**, "*Cloud Computing Concepts, Technology & Architecture*", Arcitura Education, Westford, USA, pp 41-50.

7. **Buyya R., Yeo C. S., Venugopal S., Broberg J., Brandic I., (2008)**, "*Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as The 5th Utility*", ScienceDirect, Future Generation Computer Systems, vol. 25, pp. 599-616.

8. **Cavanaugh E.**, "White paper: *Web Services: Benefits, Challenges, and A Unique Visual Development Solution*", Altova, USA.

9. **Chappell D., Jewell T., (2002)**, "*Java Web Services*", O'Reilly, USA, pp. 6-15.

10. **Carlin S., Curran K., (2012)**, "*Cloud Computing Technology*", International Journal of Cloud Computing and Services Science (IJ-CLOSER), vol. 1, no. 2, pp. 59-65.

11. **VMware, (2007)**, "White paper: *Understanding Full Virtualization, Paravirtualization, and Hardware Assist*", USA.

12. **Li S., Yen D. C., Hu C., Lu W., Chiu Y. C., (2012)**, "*Identifying Critical Factors for Corporate Implementing Virtualization Technology*", SciVerse ScienceDirect, Computers in Human Behavior, vol. 28, pp. 2244–2257.

13. **Williams M., (2010)**, "*A Quick Start Guide to Cloud Computing*", Kogan Page, USA, pp. 21.

14. **Sosinsky B., (2011)**, "*Cloud Computing Bible*", Wiley, Indiana, pp. 30-115.

15. **Sun Microsystems, (2009)**, "White paper: *Introduction to Cloud Computing Architecture*", 1st Edition, Sun Microsystems, Santa Clara, USA.

16. **Dialogic Corporation**, "White paper: *Introduction to Cloud Computing*", Montreal, Canada, www.dialogic.com, (Data Download Date: 6 May 2014).

17. **Krutz L., Vines R., (2010)**, "*Cloud Security: a Comprehensive Guide to Secure Cloud Computing*", John Wiley, http://books.google.com.tr /books?id=f8AVWUk8lb4C&pg, (Data Download Date: 12 May 2014).

18. **Velte A., Velte T., Elsenpeter R., (2010)**, "*Cloud Computing: a Practical Approach*". The McGraw-Hill Companies, USA, pp. 32-35.

19. **Vaquero L. M., Rodero-Merino L., Caceres J., Lindner M., (2009)**, "*A Break in the Clouds: Towards a Cloud Definition*", SIGCOMM Computer Communication Review, vol. 39, no. 1, pp. 50-55.

20. **Chou Y., (2010)**, http://blogs.technet.com/b/yungchou/archive/2010/12/17/ cloud-computing-concepts-for-it-pros-2-3.aspx, (Data Download Date: 15 May 2014).

21. **Armbrust M., *Fox A., Griffith R., Joseph A. D., Katz R., Konwinski A.,* (2009),** "*Above the Clouds: A Berkeley View of Cloud Computing*", Technical Rept. UCB/EECS-28, http://www.eecs.berkeley.edu /Pubs/TechRpts/2009/ EECS-2009-28.html, (Data Download Date: 2 May 2014).

22. **Amazon Web Services, (2012)**, "*Creating Healthcare Data Applications to Promote HIPAA and HITECH Compliance".*

23. **Federal Ministry of Economics and Technology (BMWi)**, "*The Standardisation Environment for Cloud Computing*", Hansa Print Service GmbH, Berlin.

24. **Oredope A., (2013)**, "*The Standardisation of Cloud Computing: Trends in the State-of-the-Art and Management Issues for the Next Generation of Cloud*", Science and Information Conference, London, UK, pp. 7-9.

25. **Avram M. G., (2013)**, "*Advantages and Challenges of Adopting Cloud Computing from an Enterprise Perspective*", ScienceDirect, The 7th International Conference Interdisciplinary in Engineering (INTER-ENG), pp. 529-534.

26. **Suleiman B., Sakr S., Jeffery R., Liu A., (2011)**, "*On Understanding the Economics and Elasticity Challenges of Deploying Business Applications on Public Cloud Infrastructure*", Springer, J Internet Serv. Appl, vol. 3, no. 2, pp. 173-193.

27. **Qi L., Dou W., Ni J., Xia X., Ma C., Li J.,(2014)**, "*A Trust Evaluation Method for Cloud Services with Fluctuant QoS and Flexible SLA*", IEEE , International Conference on Web Services, pp. 345-352.

28. **Buyya R., (2009)**, "*Cloudbus Toolkit for Market-Oriented Cloud Computing*", Springer-Verlag, CloudCom, Berlin Heidelberg, pp. 24–44.

29. **Christopher B., (2010)**,"*A Brief Guide to Cloud Computing*", Constable and Robinson Ltd., UK, pp. 7-24.

30. **Latchoumy P., Khader P. S. A., (2011)**, "*Survey on Fault Tolerance in Grid Computing*", International Journal of Computer Science and Engineering Survey (IJCSES), vol.2, no.4, pp. 97-110.

31. **Ganga K., Karthik S., Christopher A., (2012)**, "*A Survey on Fault Tolerance in Work flow Management and Scheduling*", International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), vol. 1, pp. 176-179.

32. **Pullum L. L., (2001)**, "*Software Fault Tolerance Techniques and Implementation*", Arteh House, USA, pp. 24-80.

33. **Dubrova E., (2013)**, "*Fault Tolerant Design*", Springer, Science Business Media, New York, pp. 8-16.

34. **Selic B., (2001)**, "*Fault Tolerance Techniques for Distributed Systems*", Rational Software, Canada, pp. 30-73.

35. **Patel A., Taghavi M., Bakhtiyari K., Junior J. C., (2013)**, "*An Intrusion Detection and Prevention System in Cloud Computing: a Systematic Review*", ScienceDirect, Journal of Network and Computer Applications, vol. 36, pp. 25–41.

36. **Machida F., Andrade E., Kim D. S., Trivedi K. S., (2011)**, "*Candy: Component-based Availability Modeling Framework for Cloud Service Management Using SysML*", IEEE, International Symposium on Reliable Distributed Systems, pp.209-218.

37. **Dave S., Raghuvanshi A., (2012)**, "*Fault Tolerance Techniques in Distributed System*", International Journal of Engineering Innovation and Research, vol. 1, no.2, pp.124-130.

38. **Gong C., Liu J., Zhang Q., Chen H., Gong Z., (2010)**, "*The Characteristics of Cloud Computing*", IEEE, International Conference on Parallel Processing Workshops, pp. 275-279.

39. **Amal Ganesh A., Sandhya M., Shankar S., (2014)**, "*Study on Fault Tolerance methods in Cloud Computing*", IEEE, International Advance Computing Conference (IACC), pp. 844-849.

40. **Undheim A., Chilwan A., Heegaard P., (2011)**, "*Differentiated Availability in Cloud Computing SLAs*", IEEE, Grid Computing (GRID), pp. 129-136

41. **Avi_zienis A., Laprie J., Randell B., Landwehr C., (2004)**, "*Basic Concepts and Taxonomy of Dependable and Secure Computing*", IEEE Transactions on Dependable and Secure Computing, vol. 1, no. 1, pp. 11-33.

42. **Laprie J. C., (1996)**, "*Dependable Computing Concepts and Fault Tolerance: Terminology*", IEEE Conference Publications, vol. 3, pp. 131-135.

43. **Lussier B., Lampe A., Chatila R., Guiochet J., Ingrand F., Ingrand D., (2005)**, "*Fault Tolerance in Autonomous Systems: How and How Much?*", http://homepages.laas.fr/guiochet/telecharge/Lussier-IARP05.pdf, (Data Download Date: 4 July 2014).

44. **VMware, (2009)**, "White Paper: *Protection Mission-Critical Workloads with VMware fault Tolerance*", USA.

45. **Li H., Shang L., Dang J., Jin H., (2009)**, "*A Fault Recovery Approach in Fault-Tolerant Processor*", IEEE, International Conference on Embedded Computing, pp. 52-57.

46. **Runge A., (2012)**, "*Reliability Enhancement of Fault-prone Many-core Systems Combining Spatial and Temporal Redundancy*", IEEE, International Conference on High Performance Computing and Communications, pp. 1600-1605.

47. **Babcock C., (2010)**, "*Management Strategies for the Cloud Revolution*", McGraw-Hill Companies, pp 166-167.

48. **Li Y., (2011)**, "*FREM: A Fast Restart Mechanism for General Checkpoint/ Restart*", IEEE, Transactions on Computers, vol. 60, no. 5, pp. 639-652.

49. **Dawei Sun D., Chang G., Miao C., Wang X., (2013)**, "*Analyzing, Modeling and Evaluating Dynamic Adaptive Fault Tolerance Strategies in Cloud Computing Environments*", Springer, J Supercomput , Science and Business Media, New York, pp. 193-228.

50. **Gokuldev S.,Valarmathi M ., (2013)**, "*Fault Tolerant System for Computational and Service Grid*", International Journal of Engineering and Innovative Technology (IJEIT), vol. 2, no. 10, pp. 236-240.

51. **Garg R., and Singh A. K., (2011)**, "*Fault Tolerance in Grid Computing: State of the Art and Open Issues*", International Journal of Computer Science and Engineering Survey (IJCSES), vol. 2, no.1, pp. 88-97.

52. **Ghoreyshi S. M., (2013)**, "*Energy-Efficient Resource Management of Cloud Datacenters Under Fault Tolerance Constraints*", IEEE, Green Computing Conference, pp. 1-6.

53. **Ganga K., Karthik S., Paul A. C., (2012)**, "*A Survey on Fault Tolerance in Work flow Management and Scheduling*", International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), vol. 1, no. 8, 176-179.

54. **Patra P., Singh H., Singh G., (2013)**, "*Fault Tolerance Techniques and Comparative Implementation in Cloud Computing*", International Journal of Computer Applications, vol. 64, no.14, pp. 37-41.

55. **Vallee G., Charoenpornwattana K., Leangsuksun C., (2008)**, "*A Framework for Proactive Fault Tolerance*", IEEE, International Conference on Availability, Reliability and Security, pp. 659 - 664.

56. **Lin S., Huang M., Lai K., Huang K., (2008)**, "*Design and Implementation of Job Migration Policies in P2P Grid Systems*", IEEE, Asia-Pacific Services Computing Conference, pp. 75-80.

57. **Kwon Y., Balazinska M., Greenberg A., (2008)**, "*Fault Tolerant Stream Processing using a Distributed, Replicated File System*", Auckland, New Zealand, vol. 1, no. 1, pp. 574-585.

58. **Bala A., Chana I., (2012)**, "*Fault Tolerance- Challenges, Techniques and Implementation in Cloud Computing*", IJCSI International Journal of Computer Science Issues, vol. 9, no. 1, pp. 288-293.

59. **Plankensteiner K., Prodan R., Fahringer T., (2009)**, "*A New Fault Tolerance Heuristic for Scientific Workflows in Highly Distributed Environments Based on Resubmission Impact*", EEE, International Conference on e-Science, pp. 313-320.

60. **Magoules F., (2009)**, "*Introduction to Grid Computing*", Taylor and Francis Group, USA, pp. 37-40.

61. **Buyya J. Y. R., (2005)**, "*A Taxonomy of Workflow Management Systems for Grid Computing*", Springer, Journal of Grid Computing, pp. 171-200.

62. **Egwutuoha I. P., (2012)**, "*A Proactive Fault Tolerance Approach to High Performance Computing (HPC) in the Cloud*", IEEE , International Conference on Cloud and Green Computing, pp. 268-273.

63. **Engelmann C., Vallee G. R., Naughton T., Scott S. L., (2009)**, "*Proactive Fault Tolerance Using Preemptive Migration*", IEEE, Euromicro International Conference, pp. 252-257.

64. **Machida F., Kim D. S., Park J. S., Trivedi K. S., (2008)**, "*Toward Optimal Virtual Machine Placement and Rejuvenation Scheduling in a Virtualized Data Center*", IEEE, Software Reliability Engineering Workshops, pp. 1-3.

65. **Park J., Yoo G., Lee E., (2008),** "*A Reconfiguration Framework for Self-Healing Software*", IEEE, International Conference on Convergence and Hybrid Information Technology, pp. 83-91.

66. **Ramos B. L. C., (2007)**, "*Challenging Malicious Inputs with Fault Tolerance Techniques*", Black Hat Europe, (Data Download Date: 10 June 2014).

67. **Feng Q., et al., (2012)**, "*Magicube: High Reliability and Low Redundancy Storage Architecture for Cloud Computing*", IEEE, Networking, Architecture and Storage (NAS), pp. 89-93.

68. **Zheng Z., et al., (2010)**, "*FTCloud: A Component Ranking Framework for Fault-Tolerant Cloud Applications*", IEEE, Software Reliability Engineering (ISSRE), pp. 398-407.

69. **Gomeza A., Carril L. M., Valin R., Mourino J. C., Cotelo C., (2014)**, "*Fault-tolerant virtual cluster experiments on federated sites*", ScienceDirect, Future Generation Computer Systems, vol. 34, pp. 17–25.

70. **Egwutuoha I. P., Chen S., Levy D., Selic B.**, **(2012)**, "*A Fault Tolerance Framework for High Performance Computing in Cloud*", IEEE, Cluster, Cloud and Grid Computing (CCGrid), pp. 709-710.

71. **Das P., Khilar P. M., (2013)**, "*VFT: A Virtualization and Fault Tolerance Approach for Cloud Computing*", IEEE, Information and Communication Technologies (ICT), pp. 473-478.

72. **Malik S., Huet F., (2011)**, "*Adaptive Fault Tolerance in Real Time Cloud Computing*", IEEE, Services (SERVICES), pp. 280-287.

73. **Nandi B. B., Paul H. S., Banerjee A., Ghosh S. C., (2013)**, "*Fault Tolerance as a Service*", IEEE, Cloud Computing (CLOUD), pp. 446-453.

74. **Jhawar R. , Piuri V., Santambrogio M., (2012)**, "*A Comprehensive Conceptual System-Level Approach to Fault Tolerance in Cloud Computing*", IEEE, Systems Conference (SysCon), pp. 1-5.

75. **Zhang Y., Zheng Z., Lyu M. R., (2011)**, "*BFTCloud: A Byzantine Fault Tolerance Framework for Voluntary-Resource Cloud Computing*", IEEE, Cloud Computing (CLOUD), pp. 444-451.

76. **Jayadivya S. K., Nirmala J. S., Bhanu M. S., (2012)**, "*Fault Tolerant Workflow Scheduling Based on Replication and Resubmission of Tasks in Cloud Computing*", International Journal on Computer Science and Engineering (IJCSE), vol. 4, no. 6, pp. 996-1006.

77. **Cully B., Lefebvre G., Meyer D., Feeley M., Hutchinson N., Warfield A., (2008)**, "*Remus: High Availability via Asynchronous Virtual Machine Replication*", USENIX Association, Berkeley, CA, USA, pp. 161-174.

78. **NetApp, (2012)**, "White Paper: *Creating an Enterprise-Wide Cloud Strategy*", Theresa Villatore-Silva, USA.

79. **Jules O., Hafid A., Serhani M. A., (2014)**, "*Bayesian Network, and Probabilistic Ontology Driven Trust Model for SLA Management of Cloud Services"*, IEEE, Cloud Networking (Cloud Net), pp. 77-83.

80. **Mark E. A., Lozano B., (2010)**, "*Executive's Guide to Cloud Computing*", John Wiley, New Jersey, pp.144-148.

81. **Gonzalez A. J., Helvik B. E., (2012)**, "*System Management to Comply with SLA Availability Guarantees in Cloud Computing*", IEEE, Cloud Computing Technology and Science (CloudCom), pp. 325-332.

82. **Stamou K., Kantere V., Morin J., Georgiou M., (2014)**, “*SLA Information Management through Dependency Digraphs: the Case of Cloud Data Services*”, IEEE, System Sciences (HICSS), pp. 5038-5047.

APPENDICES A

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Al-Raheym, Shereen

Date and Place of Birth: 18 December 1980, Basrah/Iraq

Marital Status: Single

Phone: +9647706711077

Email: sherijma@gmail.com

EDUCATION

Degree	**Institution**	**Year of Graduation**
B.Sc.	Basrah University, Collage of Science, Computer Science Department	2002
High School	Al-Najaah High School	1998

WORK EXPERIENCE

Year	**Place**	**Enrollment**
2003- 2012	Basrah University Collage of Science, Computer Science Department	Lab. Assistant

FOREIN LANGUAGES

Advanced English