EFFECTIVE HYBRID APPROACH FOR STORAGE VIRTUALIZATION USING GENETIC ALGORITHM IN CLOUD COMPUTING

Gaurav Kumar
Magma Research and Consultancy Pvt. Ltd.
Ambala City, Haryana, India

gaurav.kumar@magmaconsultancy.com

CHAPTER 1
INTRODUCTION

Cloud computing [1] is a unique and effective way to use the computing infrastructure for effective communication and less overhead. The major technology behind grid as well as cloud computing is the virtualization technology. Using virtualization technology, the number of arbitrary and remote processors works in parallel for the execution of tasks in such a way that there is less complexity and higher integrity of the processes. In cloud and grid architecture, there are number of remote computing devices and processors which work and execute the processes in parallel so that there is minimum load on the single node or machine. By this way, the overall load is balanced. There are number of research areas in the domain of grid computing and cloud computing, still few areas are still in the preferences of the research community as these areas needs modifications and enhancements in regular basis. The major areas of research in grid and cloud architecture are -

- Load Balancing
- Power Optimization
- Energy Optimization
- Secured Routing
- Task Scheduling
- Confidentiality and Integrity Management
- Public and Private Key Cryptography ….. and many others

1.1 CLOUD SERVICE PROVIDERS

In the current instance, most of computing services are available on demands which are hosted by number of cloud service providers [2]. There are number of domains and key areas in cloud computing which are under research from a long time due to escalating need and applications of digital services. Cloud computing and related services are very frequently taken as the research domain by the research scholars as well as academic practitioners. As cloud services are having number of domains, deployment models and respective algorithmic approaches, there are huge scope of research.

1.2 CLOUD ARCHITECTURAL FRAMEWORK

Cloud Computing Architectural Framework [3] provides a conceptual framework for the rest of the Cloud Security Alliance's guidance. The contents of this domain focus on a description of Cloud Computing that is specifically tailored to the unique perspective of IT network and security professionals. Understanding the architectural framework described in this domain is an important first step in understanding the remainder of the Cloud Security Alliance guidance. The framework defines many of the concepts and terms used throughout the other domains.

Cloud computing ('cloud') is an evolving term that describes the development of many existing technologies and approaches to computing into something different. Cloud separates application and information resources from the underlying infrastructure, and the mechanisms used to deliver them. Cloud enhances collaboration, agility, scaling, and availability, and provides the potential for cost reduction through optimized and efficient computing. More specifically, cloud describes the use of a collection of services, applications, information, and infrastructure comprised of pools of compute, network, information, and storage resources. These components can be rapidly orchestrated, provisioned, implemented and decommissioned, and scaled up or down; providing for an on-demand utility-like model of allocation and consumption.

From an architectural perspective, there is much confusion surrounding how cloud is both similar to and different from existing models of computing, and how these similarities and differences impact the organizational, operational, and technological approaches to network and information security practices. There are many definitions today that attempt to address cloud from the perspective of academicians, architects, engineers, developers, managers, and consumers. This document focuses on a definition that is specifically tailored to the unique perspectives of IT network and security professionals. The earlier version of the Cloud Security Alliance's guidance featured definitions that were written prior to the published work of the scientists at the U.S. National Institute of Standards and Technology (NIST) [4] and their efforts around defining cloud computing.

NIST's publication is generally well accepted, and we have chosen to align with the NIST Working Definition of cloud computing (version 15 as of this writing) to bring coherence and consensus around a common language so we can focus on use cases rather than semantic nuance.

It is important to note that this guide is intended to be broadly usable and applicable to organizations globally. While NIST is a U.S. government organization, the selection of this reference model should not be interpreted to suggest the exclusion of other points of view or geographies.

1.3 CLOUD REFERENCE MODEL

Understanding the relationships and dependencies between Cloud Computing models [5] is critical to understanding Cloud Computing security risks. IaaS is the foundation of all cloud services, with PaaS building upon IaaS, and SaaS in turn building upon PaaS as described in the Cloud Reference Model diagram. In this way, just as capabilities are inherited, so are information security issues and risk. It is important to note that commercial cloud providers may not neatly fit into the layered service models. Nevertheless, the reference model is important for relating real-world services to an architectural framework and understanding the resources and services requiring security analysis.

IaaS includes the entire infrastructure resource stack from the facilities to the hardware platforms that reside in them. It incorporates the capability to abstract resources (or not), as well as deliver physical and logical connectivity to those resources. Ultimately, IaaS provides a set of APIs, which allow management and other forms of interaction with the infrastructure by consumers.

PaaS sits atop IaaS and adds an additional layer of integration with application development frameworks, middleware capabilities, and functions such as database, messaging, and queuing.

These services allow developers to build applications on the platform with programming languages and tools are supported by the stack.

SaaS in turn is built upon the underlying IaaS and PaaS stacks and provides a self-contained operating environment used to deliver the entire user experience, including the content, its presentation, the application(s), and management capabilities.

It should therefore be clear that there are significant trade-offs to each model in terms of integrated features, complexity vs. openness (extensibility), and security. Generally, SaaS provides the most integrated functionality built directly into the offering, with the least consumer extensibility, and a relatively high level of integrated security (at least the provider bears a responsibility for security).

PaaS is intended to enable developers to build their own applications on top of the platform. As a result, it tends to be more extensible than SaaS, at the expense of customer-ready features. This tradeoff extends to security features and capabilities, where the built-in capabilities are less complete, but there is more flexibility to layer on additional security.

IaaS provides few if any application-like features, but enormous extensibility. This generally means less integrated security capabilities and functionality beyond protecting the infrastructure itself. This model requires that operating systems, applications, and content be managed and secured by the cloud consumer.

The key takeaway for security architecture is that the lower down the stack the cloud service provider stops, the more security capabilities and management consumers are responsible for implementing and managing themselves.

In the case of SaaS, this means that service levels, security, governance, compliance, and liability expectations of the service and provider are contractually stipulated, managed to, and enforced. In the case of PaaS or IaaS, it is the responsibility of the consumer's system administrators to effectively manage the same, with some offset expected by the provider for securing the underlying platform and infrastructure components to ensure basic service availability and security. It should be clear in either case that one can assign/transfer responsibility but not necessarily accountability.

Narrowing the scope or specific capabilities and functionality within each of the cloud delivery models, or employing the functional coupling of services and capabilities across them, may yield derivative classifications. For example "Storage as a Service" is a specific sub-offering within the IaaS 'family'. For an excellent overview of the many cloud computing use cases, the Cloud Computing Use Case Group produced a collaborative work to describe and define common cases and demonstrate the benefits of cloud, with their goal being to "...bring together cloud consumers and cloud vendors to define common use cases for cloud computing...and highlight the capabilities and requirements that need to be standardized in a cloud environment to ensure interoperability, ease of integration, and portability."

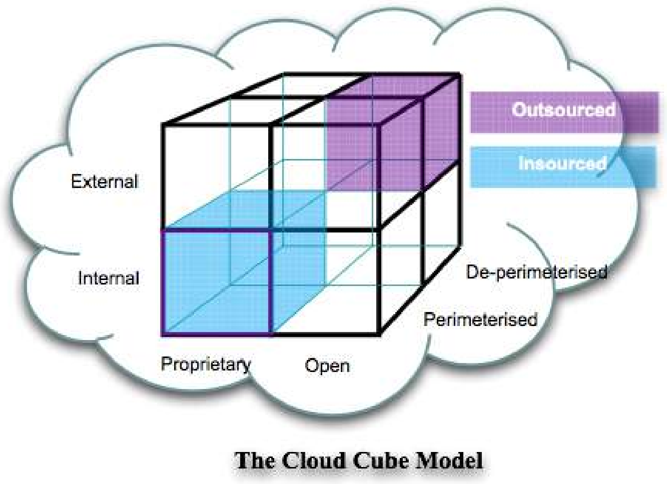


Figure 1.1 - Jericho Cloud Cube Model [6]

The Cloud Cube Model illustrates the many permutations available in cloud offerings today and presents four criteria/dimensions in order to differentiate cloud 'formations' from one another and the manner of their provision, in order to understand how cloud computing affects the way in which security might be approached.

The Cloud Cube Model also highlights the challenges of understanding and mapping cloud models to control frameworks and standards such as ISO/IEC 27002, which provides "...a series of guidelines and general principles for initiating, implementing, maintaining, and improving information security management within an organization."

The ISO/IEC 27002, section 6.2, "External Parties" control objective states: "…the security of the organization's information and information processing facilities should not be reduced by the introduction of external party products or services…"

As such, the differences in methods and responsibility for securing the three cloud service models mean that consumers of cloud services are faced with a challenging endeavor. Unless cloud providers can readily disclose their security controls and the extent to which they are implemented to the consumer, and the consumer knows which controls are needed to maintain the security of their information, there is tremendous potential for misguided decisions and detrimental outcomes.

This is critical. First one classifies a cloud service against the cloud architecture model. Then it is possible to map its security architecture as well as business, regulatory, and other compliance requirements against it as a gap-analysis exercise. The result determines the general "security" posture of a service and how it relates to an asset's assurance and protection requirements.

The figure below shows an example of how a cloud service mapping can be compared against a catalogue of compensating controls to determine which controls exist and which do not — as provided by the consumer, the cloud service provider, or a third party. This can in turn be compared to a compliance framework or set of requirements such as PCI DSS, as shown.

1.4 CLOUD SECURITY

Security controls in cloud computing are, for the most part, no different than security controls in any IT environment. However, because of the cloud service models employed, the operational models, and the technologies used to enable cloud services, cloud computing may present different risks to an organization than traditional IT solutions.

Cloud computing is about gracefully losing control while maintaining accountability even if the operational responsibility falls upon one or more third parties.

An organization’s security posture is characterized by the maturity, effectiveness, and completeness of the risk-adjusted security controls implemented. These controls are implemented in one or more layers ranging from the facilities (physical security), to the network infrastructure (network security), to the IT systems (system security), all the way to the information and applications (application security). Additionally controls are implemented at the people and process levels, such as separation of duties and change management, respectively.

As described earlier in this document, the security responsibilities of both the provider and the consumer greatly differ between cloud service models. Amazon’s AWS EC2 infrastructure as a service offering, as an example, includes vendor responsibility for security up to the hypervisor, meaning they can only address security controls such as physical security, environmental security, and virtualization security. The consumer, in turn, is responsible for security controls that relate to the IT system (instance) including the operating system, applications, and data.

The inverse is true for Salesforce.com’s customer resource management (CRM) SaaS offering. Because the entire ‘stack’ is provided by Salesforce.com, the provider is not only responsible for the physical and environmental security controls, but it must also address the security controls on the infrastructure, the applications, and the data. This alleviates much of the consumer’s direct operational responsibility.

One of the attractions of cloud computing is the cost efficiencies afforded by economies of scale, reuse, and standardization. To bring these efficiencies to bear, cloud providers have to provide services that are flexible enough to serve the largest customer base possible, maximizing their addressable market. Unfortunately, integrating security into these solutions is often perceived as making them more rigid. This rigidity often manifests in the inability to gain parity in security control deployment in cloud environments compared to traditional IT. This stems mostly from the abstraction of infrastructure, and the lack of visibility and capability to integrate many familiar security controls — especially at the network layer.

The figure below illustrates these issues: in SaaS environments the security controls and their scope are negotiated into the contracts for service; service levels, privacy, and compliance are all issues to be dealt with legally in contracts. In an IaaS offering, while the responsibility for securing the underlying infrastructure and abstraction layers belongs to the provider, the remainder of the stack is the consumer's responsibility. PaaS offers a balance somewhere in between, where securing the platform itself falls onto the provider, but securing the applications developed against the platform and developing them securely, both belong to the consumer.
Understanding the impact of these differences between service models and how they are deployed is critical to managing the risk posture of an organization.

The twelve other domains which comprise the remainder of the CSA guidance highlight areas of concern for cloud computing and are tuned to address both the strategic and tactical security 'pain points' within a cloud environment, and can be applied to any combination of cloud service and deployment model. The domains are divided into two broad categories: governance and operations. The governance domains are broad and address strategic and policy issues within a cloud computing environment, while the operational domains focus on more tactical security concerns and implementation within the architecture.

1.5 CLOUD COMPUTING SECURITY ARCHITECTURE

Security within cloud computing [6] is an especially worrisome issue because of the fact that the devices used to provide services do not belong to the users themselves. The users have no control of, nor any knowledge of, what could happen to their data. This is a great concern in cases when users have valuable and personal information stored in a cloud computing service. Users will not compromise their privacy so cloud computing service providers must ensure that the customers' information is safe. This, however, is becoming increasingly challenging because as security developments are made, there always seems to be someone to figure out a way to disable the security and take advantage of user information. Some of the important components of Service Provider Layer are SLA Monitor, Metering, Accounting, Resource Provisioning, Scheduler& Dispatcher, Load Balancer, Advance Resource Reservation Monitor, and Policy Management. Some of the security issues related to Service

Provider Layer are Identity, Infrastructure, Privacy, Data transmission, People and Identity, Audit and Compliance, Cloud integrity and Binding Issues. Some of the important components of Virtual Machine Layer creates number of virtual machines and number of operating systems and its monitoring. Some of the security issues related to Virtual Machine Layer are VM Sprawl, VM Escape, Infrastructure, Separation between Customers, Cloud legal and Regularity issues, Identity and Access management Some of the important components of Data Center (Infrastructure) Layer contains the Servers, CPU's, memory, and storage, and is henceforth typically denoted as Infrastructure-as-a-Service (IaaS). Some of the security issues related to Data Center Layer are secure data at rest, Physical Security : Network and Server.

Some organizations have been focusing on security issues in the cloud computing. The Cloud Security Alliance is a non-profit organization formed to promote the use of best practices for providing security assurance within Cloud Computing, and provide education on the uses of Cloud Computing to help secure all other forms of computing. The Open Security Architecture (OSA) is another organizations focusing on security issues. They propose the OSA pattern, which pattern is an attempt to illustrate core cloud functions, the key roles for oversight and risk mitigation, collaboration across various internal organizations, and the controls that require additional emphasis. For example, the Certification, Accreditation, and Security Assessments series increase in importance to ensure oversight and assurance given that the operations are being “outsourced” to another provider. System and Services Acquisition is crucial to ensure that acquisition of services is managed correctly. Contingency planning helps to ensure a clear understanding of how to respond in the event of interruptions to service delivery.

1.6 CHALLENGES IN CLOUD COMPUTING

Cloud Computing research addresses the challenges of meeting the requirements of next generation private, public and hybridcloud computing architectures, also the challenges of allowing applications and development platforms to take advantage of the benefits of cloud computing. The research on cloud computing is still at an early stage. Many existing issues have not been fully addressed, while new challenges keep emerging from industry applications.

Some of the challenging research issues in cloud computing are given below.

- Service Level Agreements (SLA's)
- Cloud Data Management And Security
- Data Encryption
- Migration of Virtual Machines
- Interoperability
- Access Controls
- Energy Management
- Multitenancy
- Server Consolidation
- Reliability and Availability of Service
- Common Cloud Standards
- Platform Management

Service Level Agreements (SLA's) : Cloud is administrated by service level agreements that allow several instances of one application to be replicated on multiple servers if need arises; dependent on a priority scheme, the cloud may minimize or shut down a lower level application. A big challenge for the Cloud customers is to evaluate SLAs of Cloud vendors. Most vendors create SLAs to make a defensive shield against legal action, while offering minimal assurances to customers. So, there are some important issues, e.g., data protection, outages, and price structures that need to be taken into account by the customers before signing a contract with a provider. The specification of SLAs will better reflect the customers' needs if they address the required issues at the right time.

Cloud Data Management : Cloud data Can be very large (e.g. text-based or scientific applications), unstructured or semi-structured, and typically append-only with rare updates Cloud data management an important research topic in cloud computing. Since service providers typically do not have access to the physical security system of data centers, they must rely on the infrastructure provider to achieve full data security.

Data Encryption : Encryption is a key technology for data security. Understand data in motion and data at rest encryption. Remember, security can range from simple (easy to manage, low cost and quite frankly, not very secure) all the way to highly secure (very complex, expensive to manage, and quite limiting in terms of access).

Migration of virtual Machines : applications are not hardware specific; various programs may run on one machine using virtualization or many machines may run one program. Virtualization can provide significant benefits in cloud computing by enabling virtual machine migration to balance load across the data center. In addition, virtual machine migration enables robust and highly responsive provisioning in data centers. Virtual machine migration has evolved from process migration techniques. More recently, Xen and VMWare have implemented "live" migration of VMs that involves extremely short downtimes ranging from tens of milliseconds to a second. The major benefits of VM migration are to avoid hotspots; however, this is not straightforward.

Interoperability : This is the ability of two or more systems work together in order to exchange information and use that exchanged information. Many public cloud networks are configured as closed systems and are not designed to interact with each other. The lack of integration between these networks makes it difficult for organizations to combine their IT systems in the cloud and realize productivity gains and cost savings. To overcome this challenge, industry standards must be developed to help cloud service providers design interoperable platforms and enable data portability.

Energy Resource Management : Significant saving in the energy of a cloud data center without sacrificing SLA are an excellent economic incentive for data center operators and would also make a significant contribution to greater environmental sustainability.

Multi-tenancy : There are multiple types of cloud applications that users can access through the Internet, from small Internet-based widgets to large enterprise software applications that have increased security requirements based on the type of data being stored on the software vendor's infrastructure. These application requests require multi-tenancy for many reasons, the most important is cost.

Server consolidation : The increased resource utilization and reduction in power and cooling requirements achieved by server consolidation are now being expanded into the cloud. Server consolidation is an effective approach to maximize resource utilization while minimizing energy consumption in a cloud computing environment.

Reliability & Availability of Service : The challenge of reliability comes into the picture when a cloud provider delivers on-demand software as a service. The software needs to have anetwork conditions (such as during slow network connections). There are a few cases identified due to the unreliability of on-demand software.

Common Cloud Standards : Security based accreditation for Cloud Computing would cover three main areas which are technology, personnel and operations. Technical standards are likely to be driven by organizations, such as, Jericho Forum1 before being ratified by established bodies, e.g., ISO2 (International Standard Organization).

Delivery Models for Cloud Computing

Cloud computing providers can offer services at different layers of the resource stack, simulating the functions that are performed by applications, operating systems, or physical hardware. Although some commentators categorize cloud computing services somewhat differently, the most common approach segregates services into a three-part taxonomy (see, for example, Foster et al., 2008; Vaquero et al., 2009; Mell & Grance, 2009).

- *Software as a Service* (SaaS) offers finished applications that end users can access through a thin client (typically, but not necessarily, a web browser). Prominent examples of SaaS include Gmail, Google Docs, and Salesforce.com. The end user does not exercise any control over the design of the application (aside from some minor customization and configuration options), servers, networking, and storage infrastructure.

- *Platform as a Service* (PaaS) offers an operating system as well as suites of programming languages and software development tools that customers can use to develop their own applications. Prominent examples include Microsoft Windows Azure and Google App Engine. PaaS gives end users control over application design, but does not give them control over the physical infrastructure.
- *Infrastructure as a Service* (IaaS) offers end users direct access to processing, storage, and other computing resources and allows them to configure those resources and run operating systems and software on them as they see fit. Examples of IaaS include Amazon Elastic Compute Cloud (EC2), Rackspace, and IBM Computing on Demand.

1.7 PROBLEM DEFINITION

The classical approach of scheduling and virtualization should be enhanced using metaheuristic approach. For this integration, the genetic algorithm is providing higher efficiency and cost optimization.

The existing approaches for storage virtualization which is directly associated with the job scheduling is not effective and cost aware. It is mandatory and required in the proposed approach to devise a new algorithm and approach that is able reduce the cost, time and improve the efficiency of overall cloud architecture so that the integrated multi-tenancy platform can be fully secured with higher performance using effective scheduling.

1.8 RESEARCH OBJECTIVES

- To investigate the drawbacks and shortcomings in the classical virtualization technology that is not secured and integrity based.
- To propose a new algorithm and architecture using genetic algorithm so that the overall efficiency, cost and time factors can be optimized.
- The storage virtualization in the classical approach is not fully secured and making use of classical single key implementation.
- The proposed approach is required to be implemented so that that dynamic key based exchange of packets and information in the data centers and virtual machines can be done.
- To propose and implement a novel technique for the simulation of genetic algorithm based architecture of cloud framework

1.9 RESEARCH METHODOLOGY

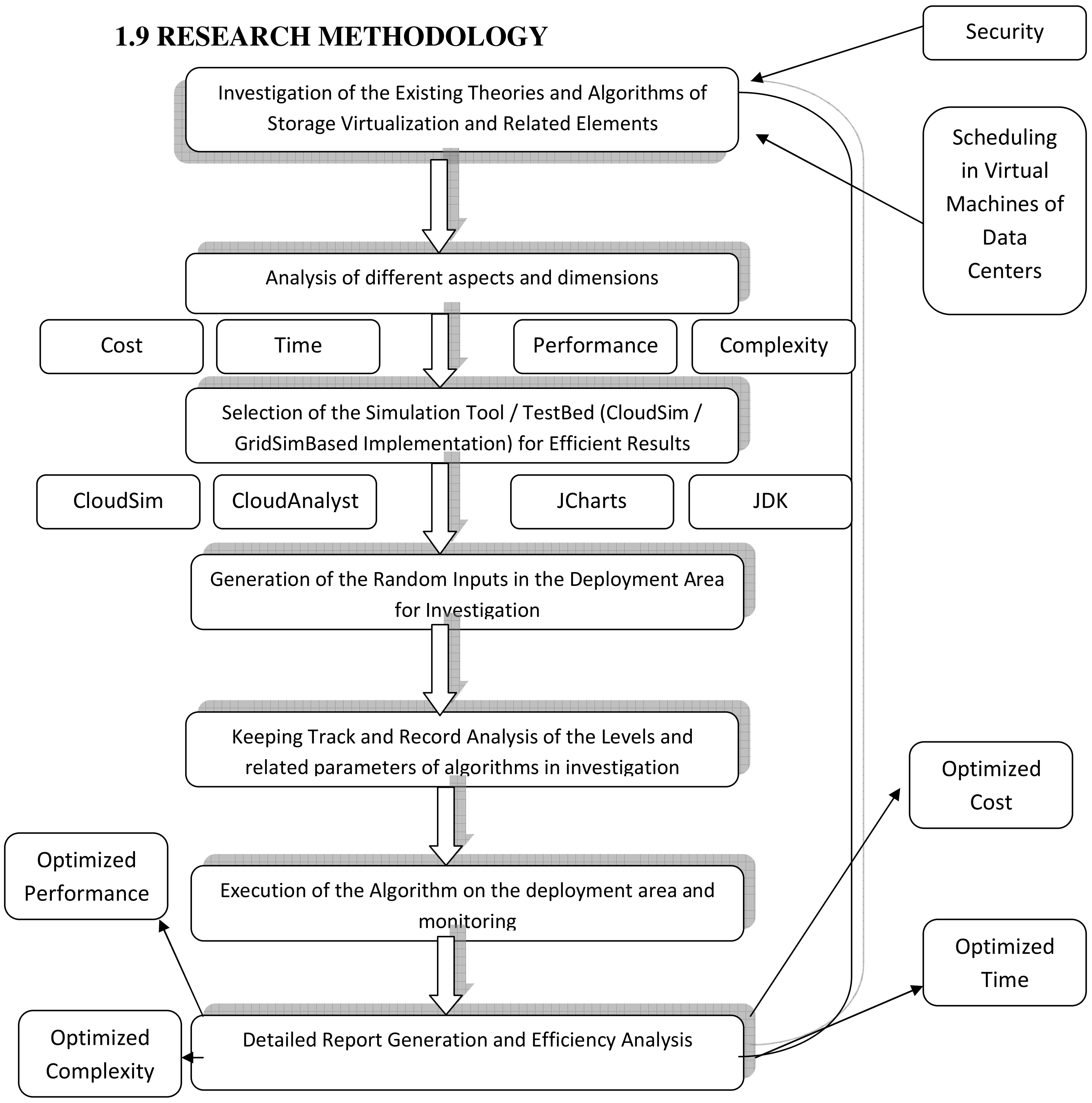

Security
Investigation of the Existing Theories and Algorithms of Storage Virtualization and Related Elements
Scheduling in Virtual Machines of Data Centers
Analysis of different aspects and dimensions
Cost
Time
Performance
Complexity
Selection of the Simulation Tool / TestBed (CloudSim / GridSimBased Implementation) for Efficient Results
CloudSim
CloudAnalyst
JCharts
JDK
Generation of the Random Inputs in the Deployment Area for Investigation
Keeping Track and Record Analysis of the Levels and related parameters of algorithms in investigation
Optimized Cost
Optimized Performance
Execution of the Algorithm on the deployment area and monitoring
Optimized Time
Optimized Complexity
Detailed Report Generation and Efficiency Analysis

Fig. 1.2 - Proposed Algorithm Flow

Our proposed approach is making use of improved storage virtualization using GA, it becomes mandatory to use higher security. Otherwise, the proposed system will be vulnerable. If we use GA with storage virtualization, there will be lots of vulnerabilities and loopholes because of number of iterations in GA. To avoid this issue, higher security based integration of dynamic hash key is used so that the data can be transferred with efficiency and security. The multiprocessor scheduling is used to work in parallel rather than sequential approach.

In virtualization, number of cloud resources including storage media and elements should be distributed without intimating the complexities to the users. In classical way, the user has to use the drives/medial on their own systems. In virtualization, all these hardware complexities are hidden and not known to the user.

To avoid the issue of degradation of the efficiency and need of higher availability of the storage media to the users, the GA will decide the chunk/block using its components (selection, crossover, mutation). Using GA, the best possible and best fit storage media will be allocated to be user. Now, again the question comes on security if same disk is being used by multiple users. For this, we have used and implemented security so that the data access will be secured and with higher integrity and privacy. Now again, the question come on the performance because many users are using same disk and multiple blocks/chunks. For this implementation, we have implemented job scheduling or simply multiprocessor scheduling.

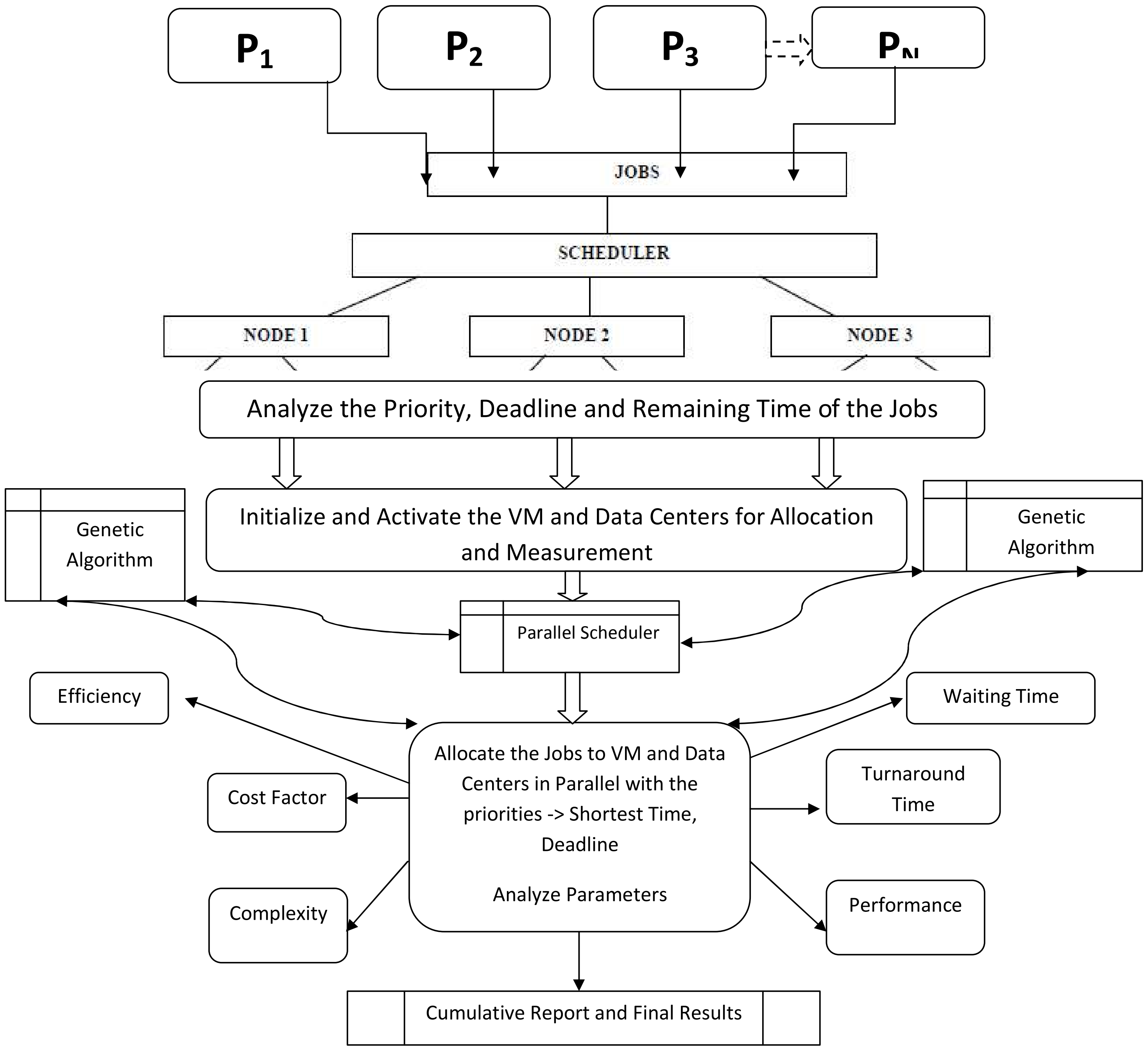

P1
P2
P3
PN
JOBS
SCHEDULER
NODE 1
NODE 2
NODE 3
Analyze the Priority, Deadline and Remaining Time of the Jobs
Initialize and Activate the VM and Data Centers for Allocation and Measurement
Genetic Algorithm
Genetic Algorithm
Parallel Scheduler
Efficiency
Waiting Time
Allocate the Jobs to VM and Data Centers in Parallel with the priorities -> Shortest Time, Deadline
Analyze Parameters
Cost Factor
Turnaround Time
Complexity
Performance
Cumulative Report and Final Results

Fig. 1.3 – Allocation of Tasks to VM and related aspects

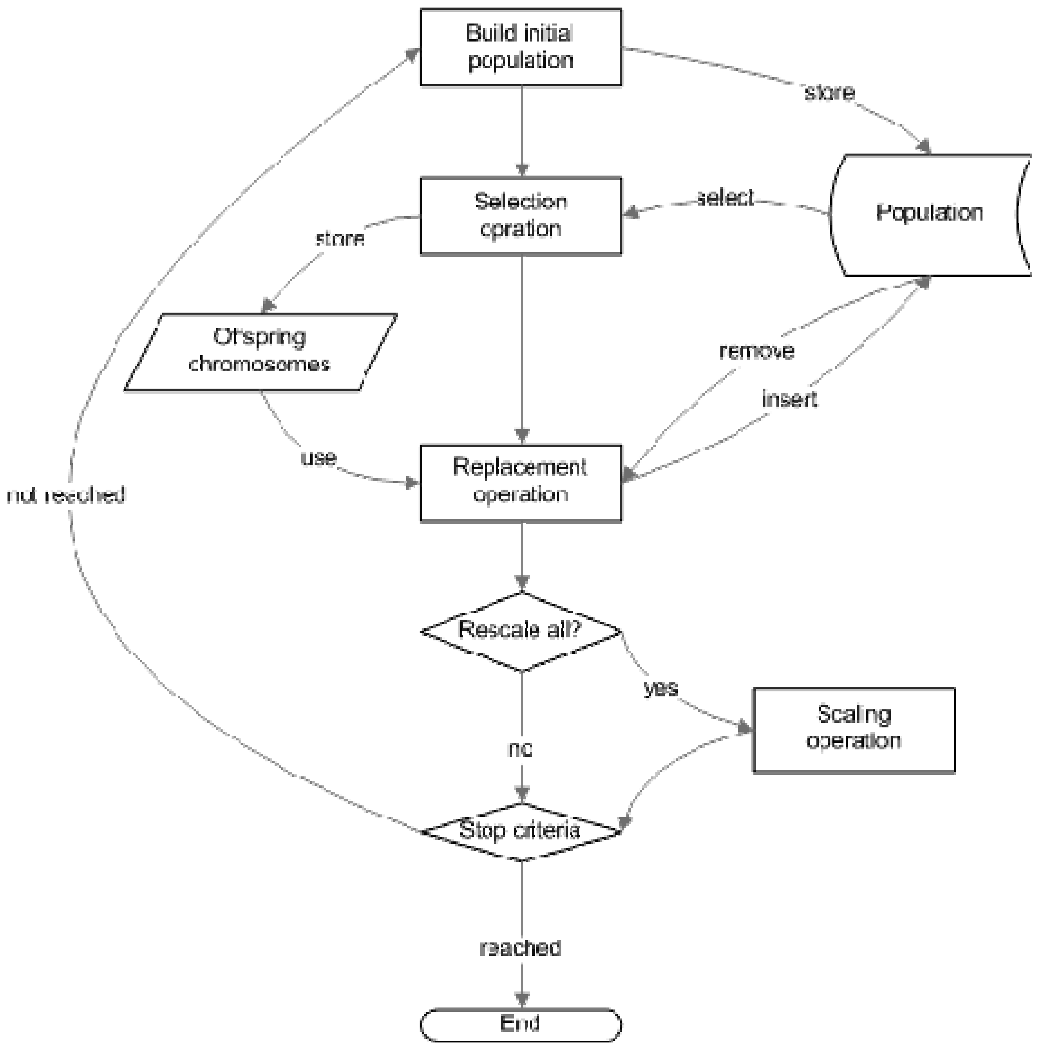


Fig. 1.4 - Modular Approach of Genetic Algorithm

1.10 ADVANTAGES OF THE PROPOSED APPROACH

- The proposed approach is effective as it is implemented and integrated using well known and high performance metaheuristic approach genetic algorithm that is used for optimization of the results including the cost and performance factor.
- The proposed algorithmic approach and system is giving effective and performance based results in terms of the optimal solution when executed using genetic algorithm.
- The proposed approach is efficient in terms of the cost, execution and turnaround time despite of the number of iterations
- The further enhancements and upgradations can be included regarding the proposed work in terms of its further enhancement using assorted metaheuristics.
- The proposed algorithmic approach and system is providing the efficient results when executed using genetic algorithm that is one of the outstanding metaheuristic techniques.

THESIS ORGANIZATION

Chapter 1 is having the major highlights of cloud computing and related technologies. In this chapter, the introduction is given in details on the assorted dimensions.

Chapter 2 Virtualization and cloud is having the details of interrelation of cloud computing and virtualization technology. In this chapter, the deep relationship of cloud and virtualization is underlined.

Chapter 3 is having the detailed information of Storage management and its associated aspect.

Chapter 4 Literature survey is having the extracts of different research papers, articles and literature.

Chapter 5 is having the details of Illustration of experimental setup. In this chapter, the tools and technologies are depicted with their relative introduction.

Chapter 6 is the Research methodology and proposed work in which the details of proposed research work is mentioned. This is providing the deep analysis of Genetic algorithm that is one of the prominent metaheuristic approaches. In this research work, this approach is used for optimization of the results.

Chapter 7 is having the Simulation Results and the graphical representation based on the multiple parameters

Chapter 8 is the concluding chapter with the scope of future work

CHAPTER 2
VIRTUALIZATION AND CLOUD COMPUTING

Virtualization is the key technology that works at the back end of cloud services and digital infrastructure. In actual implementation, the cloud is deployed using virtualization. Whenever there is need of a remote system by any user, the dynamically created virtual machines are provided to the end user or developer at other remote location with the credentials of machine at cloud data center.

Virtual Machine (VM) refers to the software implementation of any computing device or machine or computer which executes the instructions or programs as the physical machine. When a user or developer works on a virtual machine, the resources including all programs installed on the remote machine are accessible using a specific set of protocols. Here, for the end user of the cloud service, the virtual machine acts like the actual machine.

Widely used virtualization software includes VMWare Workstation, VMLite Workstation, Virtual Box, Hyper-V, Microsoft Virtual PC, Xen and KVM (Kernel Based Virtual Machine)

Virtualization technology is back bone in the grid and cloud platforms for effective management of the systems on demand for the jobs under execution. Using virtualization, there is no need of deployment of the dedicated systems or no0064es, rather the virtual machines take charge of all the operating system, platform and the resources to be used.

Virtual machines are classified into the two major taxonomies which depend on their usage and the degree of communication -

- **System Virtual Machine** – This system provides the full system environment to support the processes and execution of operating system in use.
- **Process Virtual Machine.** It is also called as Language Virtual Machine**.** Such machine is deployed and designed to process a single task or job or simply program. It means that it is dedicated for a single process. These VMs are very tightly associated

to one or more programming languages and built with the purpose of providing program portability and flexibility (amongst other things).

Key Virtualization Technology Based Software Suite in the Corporate World

- **Windows as Host OS**
  - VMWare Workstation (Any Guest OS)
  - Virtual Box (Any Guest OS)
  - Hyper-V (Any Guest OS)

- **Linux as Host OS**
  - VMWare Workstation
  - Microsoft Virtual PC
  - VMLite Workstation
  - VirtualBox
  - Xen

Hypervisor or simply virtual machine monitor (VMM) is the slot of computer software, hardware or firmware which creates and runs the virtual machines for effective usage. A system or node on which the hypervisor run one or more virtual machines refers to a host machine. Every virtual machine is known as a guest machine.

Hypervisor Type 1 Software Suite

- VMWare ESXi
- Citrix Xen
- KVM (Kernal Virtual Machine)
- Hyper-V

Hypervisor Type 2 Software Suite

- VMware Workstation
- VirtualBox

Hypervisors Used in Industry

Hypervisor	**Cloud Service Provider**
Xen	Amazon EC2 IBM SoftLayer Fujitsu Global Cloud Platform Linode OrionVM
ESXi	VMWare Cloud
KVM	Red Hat HP Del Rackspace
Hyper-V	Microsoft Azure

This research work is having the focus on development as well as implementation of a job scheduling algorithm for effective scheduling and prioritization. By this way and the proposed technique, the effective results in terms of less cost, higher performance and less execution time is obtained. In this research work, we have analyzed the performance of assorted algorithms based on new parameters and dimensions. There is need to analyze the performance of job scheduling algorithms on cloud as well as grid environment as priority, load balancing and related aspects can affect the performance of cloud and grid to a huge extent.

The work done in this research includes to propose the implementation of algorithms on new parameters so that the hidden aspects and attributes of the algorithms can be analyzed. The work includes to perform the implementation / simulation and analysis on algorithms on new parameters beyond the work in the base literature.

2.1 VIRTUALIZATION TECHNOLOGY AND CLOUD

Virtualization is the major technology that works with cloud computing. Actual cloud is implemented with the use of virtualization technology. In Cloud computing, the dynamic virtual machines are created to provide the access of actual infrastructure to the end user or developer at other remote location.

A virtual machine or simply VM is the software implementation of any computing device or machine or computer that executes the series of instructions or programs as a physical (actual) machine. When a user or developer works on a virtual machine, the resources including all programs installed on the remote machine are accessible using a specific set of protocols. Here, for the end user of the cloud service, the virtual machine acts like the actual machine.

The term VM was originally proposed and defined by Popek and Goldberg as "an efficient, isolated duplicate of a real machine".

Virtual machines are identified into two major classifications, depending on their use and degree of correspondence to any real machine:

- **System Virtual Machine** – System Virtual Machine provides a complete system platform that supports the execution of a complete operating system (OS). These emulate an existing system architecture, and are built with the purpose of either providing a platform to run programs where the real hardware is not available for use or of having multiple instances of virtual machines leading to more efficient use of computing resources, both in terms of energy consumption and cost effectiveness (known as hardware virtualization, the key to a cloud computing environment), or both.

- **Process Virtual Machine (Language Virtual Machine)** – This type of virtual machine is designed to execute a single program which means that it supports a single process. Such virtual machines are closely suited to one or more programming languages and built with the purpose of providing program portability and flexibility (amongst other things). An essential characteristic of a virtual machine is that the software running inside is limited to the resources and abstractions provided by the virtual machine - it cannot fragment or break out its virtual environment.

A hypervisor or virtual machine monitor (VMM) is a piece of computer software, firmware or hardware that creates and runs virtual machines. A computer on which a hypervisor is running one or more virtual machines is defined as a host machine. Each virtual machine is called a guest machine. The hypervisor presents the guest operating systems with a virtual operating platform and manages the execution of the guest operating systems. Multiple instances of a variety of operating systems may share the virtualized hardware resources.

In the implementation and deployment of the cloud service, type 1 hypervisors are used. Hypervisors of type 1 are associated with the concept of bare metal installation. It means there is no need of any host operating system to install the hypervisor.

By this technology, there is no risk of getting the host operating system corrupt. These hypervisors are directly installed on the hardware without need of any other operating system. On this hypervisor, multiple virtual machines are created.

A Type-1 hypervisor is a type of client hypervisor that interacts directly with hardware that is being virtualized. It is completely independent from the operating system, unlike a Type-2 hypervisor, and boots before the operating system (OS).

Currently, Type-1 hypervisors are being used by all the major players in the desktop virtualization space, including but not limited to VMware, Microsoft and Citrix.
The classical virtualization software or type 2 hypervisor is always installed on any host operating system.

If host operating system gets corrupt or crashed by any reason, the virtualization software or type 2 hypervisor will also be crashed and obviously all virtual machines and other resources will be lost. That's why the technology of hypervisor or bare metal installation is very famous in the cloud computing world.

Type 2 (Hosted) hypervisors execute within a conventional operating-system environment. With the hypervisor layer as a distinct second software level, guest operating-systems run at the third level above the hardware. A Type-2 hypervisor is a type of client hypervisor that sits on top of an operating system.

Unlike a Type-1 hypervisor, a Type-2 hypervisor relies heavily on the operating system. It cannot boot until the operating system is already up and running and, if for any reason the operating system crashes, all end-users are affected.

This is a big drawback of Type-2 hypervisors, as they are only as secure as the operating system on which they rely. Also, since Type-2 hypervisors depend on an OS, they are not in full control of the end user's machine.

Hypervisor Type 1 Products

- VMWare ESXi
- Citrix Xen
- KVM (Kernal Virtual Machine)
- Hyper-V

Hypervisor Type 2 Products

- VMware Workstation
- VirtualBox

2.2 STORAGE VIRTUALIZATION

Storage virtualization uses virtualization to enable better functionality and more advanced features in computer data storage systems.

A "storage system" is also known as a storage array or disk array or a filer. Storage systems typically use special hardware and software along with disk drives in order to provide very fast and reliable storage for computing and data processing. Storage systems are complex, and may be thought of as a special purpose computer designed to provide storage capacity along with advanced data protection features. Disk drives are only one element within a storage system, along with hardware and special purpose embedded software within the system.

Storage systems can provide either block accessed storage, or file accessed storage. Block access is typically delivered over Fibre Channel, iSCSI, SAS,FICON or other protocols. File access is often provided using NFS or CIFS protocols.

Within the context of a storage system, there are two primary types of virtualization that can occur:

- **Block virtualization** used in this context refers to the abstraction (separation) of logical storage (partition) from physical storage so that it may be accessed without regard to physical storage or heterogeneous structure. This separation allows the administrators of the storage system greater flexibility in how they manage storage for end users.

- **File virtualization** addresses the NAS challenges by eliminating the dependencies between the data accessed at the file level and the location where the files are physically stored. This provides opportunities to optimize storage use and server consolidation and to perform non-disruptive file migrations.

Block virtualization

Address space remapping

Virtualization of storage helps achieve location independence by abstracting the physical location of the data. The virtualization system presents to the user a logical space for data storage and handles the process of mapping it to the actual physical location.

It is possible to have multiple layers of virtualization or mapping. It is then possible that the output of one layer of virtualization can then be used as the input for a higher layer of virtualization. Virtualization maps space between back-end resources, to front-end resources. In this instance, "back-end" refers to a logical unit number (LUN) that is not presented to a computer, or host system for direct use. A "front-end" LUN or volume is presented to a host or computer system for use.

The actual form of the mapping will depend on the chosen implementation. Some implementations may limit the granularity of the mapping which may limit the capabilities of the device. Typical granularities range from a single physical disk down to some small subset (multiples of megabytes or gigabytes) of the physical disk.

In a block-based storage environment, a single block of information is addressed using a LUN identifier and an offset within that LUN - known as a logical block addressing (LBA).

Meta-data

The virtualization software or device is responsible for maintaining a consistent view of all the mapping information for the virtualized storage. This mapping information is often called meta-data and is stored as a mapping table.

The address space may be limited by the capacity needed to maintain the mapping table. The level of granularity, and the total addressable space both directly impact the size of the meta-data, and hence the mapping table. For this reason, it is common to have trade-offs, between the amount of addressable capacity and the granularity or access granularity.

One common method to address these limits is to use multiple levels of virtualization. In several storage systems deployed today, it is common to utilize three layers of virtualization.

Some implementations do not use a mapping table, and instead calculate locations using an algorithm. These implementations utilize dynamic methods to calculate the location on access, rather than storing the information in a mapping table.

I/O redirection

The virtualization software or device uses the meta-data to re-direct I/O requests. It will receive an incoming I/O request containing information about the location of the data in terms of the logical disk (vdisk) and translates this into a new I/O request to the physical disk location.

For example, the virtualization device may :

- Receive a read request for vdisk LUN ID=1, LBA=32
- Perform a meta-data look up for LUN ID=1, LBA=32, and finds this maps to physical LUN ID=7, LBA0
- Sends a read request to physical LUN ID=7, LBA0
- Receives the data back from the physical LUN
- Sends the data back to the originator as if it had come from vdisk LUN ID=1, LBA32

Capabilities

Most implementations allow for heterogeneous management of multi-vendor storage devices within the scope of a given implementation's support matrix. This means that the following capabilities are not limited to a single vendor's device (as with similar capabilities provided by specific storage controllers) and are in fact possible across different vendors' devices.

Replication

Data replication techniques are not limited to virtualization appliances and as such are not described here in detail. However most implementations will provide some or all of these replication services.

When storage is virtualized, replication services must be implemented above the software or device that is performing the virtualization. This is true because it is only above the virtualization layer that a true and consistent image of the logical disk (vdisk) can be copied. This limits the services that some implementations can implement - or makes them seriously difficult to implement. If the virtualization is implemented in the network or higher, this renders any replication services provided by the underlying storage controllers useless.

Remote data replication for disaster recovery

- Synchronous Mirroring - where I/O completion is only returned when the remote site acknowledges the completion. Applicable for shorter distances (<200 km)

- Asynchronous Mirroring - where I/O completion is returned before the remote site has acknowledged the completion. Applicable for much greater distances (>200 km)

- Point-In-Time Snapshots to copy or clone data for diverse uses
  - When combined with thin provisioning, enables space-efficient snapshots

Pooling

The physical storage resources are aggregated into storage pools, from which the logical storage is created. More storage systems, which may be heterogeneous in nature, can be added as and when needed, and the virtual storage space will scale up by the same amount. This process is fully transparent to the applications using the storage infrastructure.

Disk management

The software or device providing storage virtualization becomes a common disk manager in the virtualized environment. Logical disks (vdisks) are created by the virtualization software or device and are mapped (made visible) to the required host or server, thus providing a common place or way for managing all volumes in the environment.

Enhanced features are easy to provide in this environment:

- Thin Provisioning to maximize storage utilization
  - This is relatively easy to implement as physical storage is only allocated in the mapping table when it is used.
- Disk expansion and shrinking
  - More physical storage can be allocated by adding to the mapping table (assuming the using system can cope with online expansion)
  - Similarly disks can be reduced in size by removing some physical storage from the mapping (uses for this are limited as there is no guarantee of what resides on the areas removed)

2.3 BENEFITS

Non-disruptive data migration

One of the major benefits of abstracting the host or server from the actual storage is the ability to migrate data while maintaining concurrent I/O access.

The host only knows about the logical disk (the mapped LUN) and so any changes to the meta-data mapping is transparent to the host. This means the actual data can be moved or replicated to another physical location without affecting the operation of any client. When the data has been copied or moved, the meta-data can simply be updated to point to the new location, therefore freeing up the physical storage at the old location.

The process of moving the physical location is known as data migration. Most implementations allow for this to be done in a non-disruptive manner, that is concurrently while the host continues to perform I/O to the logical disk (or LUN).

The mapping granularity dictates how quickly the meta-data can be updated, how much extra capacity is required during the migration, and how quickly the previous location is marked as free. The smaller the granularity the faster the update, less space required and quicker the old storage can be freed up.

There are many day to day tasks a storage administrator has to perform that can be simply and concurrently performed using data migration techniques.

- Moving data off an over-utilized storage device.
- Moving data onto a faster storage device as needs require
- Implementing an Information Lifecycle Management policy
- Migrating data off older storage devices (either being scrapped or off-lease)

Improved utilization

Utilization can be increased by virtue of the pooling, migration, and thin provisioning services. This allows users to avoid over-buying and over-provisioning storage solutions. In other words, this kind of utilization through a shared pool of storage can be easily and quickly allocated as it is needed to avoid constraints on storage capacity that often hinder application performance

When all available storage capacity is pooled, system administrators no longer have to search for disks that have free space to allocate to a particular host or server. A new logical disk can be simply allocated from the available pool, or an existing disk can be expanded.

Pooling also means that all the available storage capacity can potentially be used. In a traditional environment, an entire disk would be mapped to a host. This may be larger than is required, thus wasting space. In a virtual environment, the logical disk (LUN) is assigned the capacity required by the using host.

Storage can be assigned where it is needed at that point in time, reducing the need to *guess* how much a given host will need in the future. Using Thin Provisioning, the administrator can create a very large thin provisioned logical disk, thus the using system thinks it has a very large disk from day 1. it presents a logical view of the physical storage

Fewer points of management

With storage virtualization, multiple independent storage devices, even if scattered across a network, appear to be a single monolithic storage device and can be managed centrally. However, traditional storage controller management is still required. That is, the creation and maintenance of RAID arrays, including error and fault management.

2.4 RISKS

Backing out a failed implementation

Once the abstraction layer is in place, only the virtualizer knows where the data actually resides on the physical medium. Backing out of a virtual storage environment therefore requires the reconstruction of the logical disks as contiguous disks that can be used in a traditional manner.

Most implementations will provide some form of back-out procedure and with the data migration services it is at least possible, but time consuming.

Interoperability and vendor support

Interoperability is a key enabler to any virtualization software or device. It applies to the actual physical storage controllers and the hosts, their operating systems, multi-pathing software and connectivity hardware.

Interoperability requirements differ based on the implementation chosen. For example, virtualization implemented within a storage controller adds no extra overhead to host based interoperability, but will require additional support of other storage controllers if they are to be virtualized by the same software.

Switch based virtualization may not require specific host interoperability — if it uses packet cracking techniques to redirect the I/O.

Network based appliances have the highest level of interoperability requirements as they have to interoperate with all devices, storage and hosts.

Complexity

Complexity affects several areas :

- Management of environment : Although a virtual storage infrastructure benefits from a single point of logical disk and replication service management, the physical storage must still be managed. Problem determination and fault isolation can also become complex, due to the abstraction layer.
- Infrastructure design : Traditional design ethics may no longer apply, virtualization brings a whole range of new ideas and concepts to think about (as detailed here)
- The software or device itself : Some implementations are more complex to design and code - network based, especially in-band (symmetric) designs in particular — these implementations actually handle the I/O requests and so latency becomes an issue.

Meta-data management

Information is one of the most valuable assets in today's business environments. Once virtualized, the meta-data are the glue in the middle. If the meta-data are lost, so is all the actual data as it would be virtually impossible to reconstruct the logical drives without the mapping information.

Any implementation must ensure its protection with appropriate levels of back-ups and replicas. It is important to be able to reconstruct the meta-data in the event of a catastrophic failure.

The meta-data management also has implications on performance. Any virtualization software or device must be able to keep all the copies of the meta-data atomic and quickly updateable. Some implementations restrict the ability to provide certain fast update functions, such as point-in-time copies and caching where super fast updates are required to ensure minimal latency to the actual I/O being performed.

Performance and scalability

In some implementations the performance of the physical storage can actually be improved, mainly due to caching. Caching however requires the visibility of the data contained within the I/O request and so is limited to in-band and symmetric virtualization software and devices. However these implementations also directly influence the latency of an I/O request (cache miss), due to the I/O having to flow through the software or device. Assuming the software or device is efficiently designed this impact should be minimal when compared with the latency associated with physical disk accesses.

Due to the nature of virtualization, the mapping of logical to physical requires some processing power and lookup tables. Therefore, every implementation will add some small amount of latency.

In addition to response time concerns, throughput has to be considered. The bandwidth into and out of the meta-data lookup software directly impacts the available system bandwidth. In asymmetric implementations, where the meta-data lookup occurs before the information is read or written, bandwidth is less of a concern as the meta-data are a tiny fraction of the actual I/O size. In-band, symmetric flow through designs are directly limited by their processing power and connectivity bandwidths.

Most implementations provide some form of scale-out model, where the inclusion of additional software or device instances provides increased scalability and potentially increased bandwidth. The performance and scalability characteristics are directly influenced by the chosen implementation.

Implementation approaches of Virtualization

- Host-based
- Storage device-based
- Network-based

Host-based

Host-based virtualization requires additional software running on the host, as a privileged task or process. In some cases volume management is built into the operating system, and in other instances it is offered as a separate product. Volumes (LUN's) presented to the host system are handled by a traditional physical device driver. However, a software layer (the volume manager) resides above the disk device driver intercepts the I/O requests, and provides the meta-data lookup and I/O mapping.

Most modern operating systems have some form of logical volume management built-in (in Linux called Logical Volume Manager or LVM; in Solaris and FreeBSD, ZFS's zpool layer; in Windows called Logical Disk Manager or LDM), that performs virtualization tasks.

Host based volume managers were in use long before the term *storage virtualization* had been coined.

Pros

- Simple to design and code
- Supports any storage type
- Improves storage utilization without thin provisioning restrictions

Cons

- Storage utilization optimized only on a per host basis
- Replication and data migration only possible locally to that host
- Software is unique to each operating system
- No easy way of keeping host instances in sync with other instances
- Traditional Data Recovery following a server disk drive crash is impossible

Storage device-based

Like host-based virtualization, several categories have existed for years and have only recently been classified as virtualization. Simple data storage devices, like single hard disk drives, do not provide any virtualization. But even the simplest disk arrays provide a logical to physical abstraction, as they use RAIDschemes to join multiple disks in a single array (and possibly later divide the array it into smaller volumes).

Advanced disk arrays often feature cloning, snapshots and remote replication. Generally these devices do not provide the benefits of data migration or replication across heterogeneous storage, as each vendor tends to use their own proprietary protocols.

A new breed of disk array controllers allows the downstream attachment of other storage devices. For the purposes of this article we will only discuss the later style which do actually virtualize other storage devices.

Concept

A primary storage controller provides the services and allows the direct attachment of other storage controllers. Depending on the implementation these may be from the same or different vendors.

The primary controller will provide the pooling and meta-data management services. It may also provide replication and migration services across those controllers which it is .

Pros

- No additional hardware or infrastructure requirements

- Provides most of the benefits of storage virtualization
- Does not add latency to individual I/Os

Cons

- Storage utilization optimized only across the connected controllers
- Replication and data migration only possible across the connected controllers and same vendors device for long distance support
- Downstream controller attachment limited to vendors support matrix
- I/O Latency, non cache hits require the primary storage controller to issue a secondary downstream I/O request
- Increase in storage infrastructure resource, the primary storage controller requires the same bandwidth as the secondary storage controllers to maintain the same throughput

Network-based

Storage virtualization operating on a network based device (typically a standard server or smart switch) and using iSCSI or FC Fibre channel networks to connect as a SAN. These types of devices are the most commonly available and implemented form of virtualization.

The virtualization device sits in the SAN and provides the layer of abstraction between the hosts performing the I/O and the storage controllers providing the storage capacity.

Pros

- True heterogeneous storage virtualization
- Caching of data (performance benefit) is possible when in-band
- Single management interface for all virtualized storage
- Replication services across heterogeneous devices

Cons

- Complex interoperability matrices - limited by vendors support
- Difficult to implement fast meta-data updates in switched-based devices
- Out-of-band requires specific host based software

- In-band may add latency to I/O
- In-band the most complicated to design and code

Appliance-based vs. switch-based

There are two commonly available implementations of network-based storage virtualization, appliance-based and switch-based. Both models can provide the same services, disk management, metadata lookup, data migration and replication. Both models also require some processing hardware to provide these services.

Appliance based devices are dedicated hardware devices that provide SAN connectivity of one form or another. These sit between the hosts and storage and in the case of in-band (symmetric) appliances can provide all of the benefits and services discussed in this article. I/O requests are targeted at the appliance itself, which performs the meta-data mapping before redirecting the I/O by sending its own I/O request to the underlying storage. The in-band appliance can also provide caching of data, and most implementations provide some form of clustering of individual appliances to maintain an atomic view of the metadata as well as cache data.

Switch based devices, as the name suggests, reside in the physical switch hardware used to connect the SAN devices. These also sit between the hosts and storage but may use different techniques to provide the metadata mapping, such as packet cracking to snoop on incoming I/O requests and perform the I/O redirection. It is much more difficult to ensure atomic updates of metadata in a switched environment and services requiring fast updates of data and metadata may be limited in switched implementations.

In-band vs. out-of-band

In-band, also known as *symmetric*, virtualization devices actually sit in the data path between the host and storage. All I/O requests and their data pass through the device. Hosts perform I/O to the virtualization device and never interact with the actual storage device. The virtualization device in turn performs I/O to the storage device. Caching of data, statistics about data usage, replications services, data migration and thin provisioning are all easily implemented in an in-band device.

Out-of-band, also known as *asymmetric*, virtualization devices are sometimes called **meta-data servers**. These devices only perform the meta-data mapping functions. This requires additional software in the host which knows to first request the location of the actual data.

CHAPTER 3
STORAGE MANAGEMENT

Cloud storage is a model of data storage in which the digital data is stored in logical pools, the physical storage spans multiple servers (and often locations), and the physical environment is typically owned and managed by a hosting company. These cloud storage providers are responsible for keeping the data available and accessible, and the physical environment protected and running. People and organizations buy or lease storage capacity from the providers to store user, organization, or application data.

Cloud storage services may be accessed through a co-located cloud computer service, a web service application programming interface (API) or by applications that utilize the API, such as cloud desktop storage, a cloud storage gateway or Web-based content management systems.

Cloud computing is believed to have been invented by Joseph Carl Robnett Licklider in the 1960s with his work on ARPANET to connect people and data from anywhere at any time.
In 1983, CompuServe offered its consumer users 128k of disk space that could be used to store any files they chose to upload.

In 1994, AT&T launched PersonaLink Services, an online platform for personal and business communication and entrepreneurship. The storage was one of the first to be all web-based, and referenced in their commercials as, "you can think of our electronic meeting place as the cloud." Amazon Web Services introduced their cloud storage service AWS S3 in 2006, and has gained widespread recognition and adoption as the storage supplier to popular services such as Smugmug, Dropbox, Synaptop and Pinterest. In 2005, Box (company) launched an online file sharing and personal cloud content management service for businesses.

3.1 KEY CONCERNS

Outsourcing data storage increases the attack surface area.

1. When data has been distributed it is stored at more locations increasing the risk of unauthorized physical access to the data. For example, in cloud based architecture, data is replicated and moved frequently so the risk of unauthorized data recovery increases dramatically. (e.g. disposal of old equipment, reuse of drives, reallocation of storage space) The manner that data is replicated depends on the service level a customer chooses and on the service provided. that encrypts data prior to uploading it to the cloud.
2. The number of people with access to the data who could be compromised (i.e. bribed, or coerced) increases dramatically. A single company might have a small team of administrators, network engineers and technicians, but a cloud storage company will have many customers and thousands of servers and therefore a much larger team of technical staff with physical and electronic access to almost all of the data at the entire facility or perhaps the entire company. Encryption keys that are kept by the service user, as opposed to the service provider limit the access to data by service provider employees. As for sharing the multiple data with multiple users in cloud, a large number of keys has to be distributed to users via secure channels for decryption, also it has to be securely stored and managed by the users in their devices. And storing these keys requires rather expensive secure storage. To overcome that Key-aggregate cryptosystem can be used.
3. It increases the number of networks over which the data travels. Instead of just a local area network (LAN) or storage area network (SAN), data stored on a cloud requires a WAN (wide area network) to connect them both.
4. By sharing storage and networks with many other users/customers it is possible for other customers to access your data. Sometimes because of erroneous actions, faulty equipment, a bug and sometimes because of criminal intent. This risk applies to all types of storage and not only cloud storage. The risk of having data read during transmission can be mitigated through encryption technology. Encryption in transit protects data as it is being transmitted to and from the cloud service. Encryption at rest protects data that is stored at the service provider. Encrypting data in an on-

premises cloud service on-ramp system can provide both kinds of encryption protection.

Supplier stability

Companies are not permanent and the services and products they provide can change. Outsourcing data storage to another company needs careful investigation and nothing is ever certain. Contracts set in stone can be worthless when a company ceases to exist or its circumstances change. Companies can:

1. Go bankrupt.
2. Expand and change their focus.
3. Be purchased by other larger companies.
4. Be purchased by a company headquartered in or move to a country that negates compliance with export restrictions and thus necessitates a move.
5. Suffer an irrecoverable disaster.

Accessibility

Performance for outsourced storage is likely to be lower than local storage, depending on how much a customer is willing to spend for WAN bandwidth

Reliability and availability depends on wide area network availability and on the level of precautions taken by the service provider. Reliability should be based on hardware as well as various algorithms used.

CHAPTER 4
LITERATURE SURVEY

To propose, defend and depict the novel research work, number of research articles, papers and conferences are investigated. Following is the list and explanation to the excerpts fetched from the multiple ad assorted research tasks done by the academicians and researchers.

Connor (2004) makes the point that server virtualization is moving from small markets to the mainstream and that the rate of implementation is steadily increasing. The article also addresses the race to keep up with processor technology by VMWare and others (Connor, 2004). As evidence for this, Connor (2004) says, "If you look at processor trends, both Intel and AMD have shifted from increasing the clock speed of their processors to, increasing the number of processor cores on a single chip," says Michael Mullaney, vice president of marketing for VMware. "Going forward, you are going to find out that even a two-CPU server actually has four processors" (p. 01) Connor (2004) says "Virtualization is moving from a niche market into the mainstream, especially since Microsoft entered the market" (p. 01) This makes it clear that this is a project topic that is worth pursuing. This is a technology that is just now permeating into the main stream and the future of the technology is leading in a direction of more and more implementation. This article is very relevant to my project topic because of the statement above and the varied range of topics discussed in the article like VMWare's partnerships with Citrix, Dell, HP, IBM, Oracle and Red Hat.

Spiegel (2006) makes the case that there is business benefits in application and server virtualization, "Application virtualization is the new fancy trendy name for server-based computing. However, instead of installing applications on desktops, the applications are installed in a server farm for secure, remote access. Server virtualization allows you to take multiple physical servers and create the same number of virtual servers, or "machines," on one host physical server" (p. 01). By simplifying the structure and management, the business gets a benefit, or multiple benefits (Spiegel, 2006) This too supports the assertion that there is a business benefit to application and server virtualization. This assertion is further supported by the point that centralizing servers and application onto hypervisors, or rather servers running a virtual server product, like VMWare's ESXi, can give the business several things that add value and support the ROI. Those things being; reduced maintenance costs, reduced hardware costs, reduced licensing cost, reduced administration costs, and increased disaster recovery security.

Hassell (2007) gives us a summary of his article in the abstract section when he says, "Virtualization, the move from real, physical hardware to virtual hardware is being seen as one of the "next big things" in IT. There are more virtualization options for IT departments than ever before, including XenSource Inc's and Virtual Iron Software Inc's open-source applications, Microsoft Corp's Virtual Server and VMware Inc's venerable products. But if you are new to this party, you might not know how to get started." (p. 01). Hassell (2007) gives a broad overview of the topic and a launching point to explore the topic of virtualization and some of the options that are available, phases of implementation. (Hassell, 2007). Hassell (2007) is experienced in this field and is therefore a valued source for this subject matter. Hassell (2007) breaks down the process step by step, "The first step in virtualization is determining if you have the right type of infrastructure to support it. Look for a lot of machines doing similar tasks, and make sure you have more than 10 of them. For 10 or fewer, the payoff is questionable." (p. 01). Reducing redundant hardware through virtualization as a strategy is the key point made by McAllister (2007). McAllister (2007) states, "In today's complex IT environments, server virtualization simply makes sense. Redundant server hardware can rapidly fill enterprise datacenters to capacity; each new purchase drives up power and cooling costs even as it saps the bottom line. With virtualization, you can dynamically fire up and take down virtual servers, each of which basically fools an operating system into thinking the virtual machine is actual hardware. The most popular method of virtualization uses software called a hypervisor to create a layer of abstraction between virtual servers and the underlying hardware." (p. 01). This is a valid statement, and the core idea of the process behind virtualization. This article also compares the state of virtualization now with the past and names the major players in the environment like VMWare and Microsoft (McAllister, 2007). McAllister (2007) validates not only one of my suggested methods of business benefit, which is cost savings, but it also provides context for the history and installation options of a virtual environment. McAllister (2007) also details many options that are available to administrators. One main point that is important to present is the time saving and options that are available to server administrators. Bele and Desai (2012) look at virtualization from a slightly broader perspective. Beyond the hypervisor platform alone, Bele and Desai (2012) look at how virtual host tie into the rest of your environment, like SAN with a look at storage virtualization. Bele and Desai (2012) detail how all of these components are related, “Server Virtualization plays key role to resolve such problems. It abstracts the resources from the applications that utilize them and can be applied to many platforms.

Virtualization Technology has been adopted at large level by many data centers in the industry. It provides benefits like server consolidation, live migration, data security, less power consumption etc. Same time Storage Virtualization abstracts physical storage (SAN, NAS) resources from front end applications running in the system. Storage virtualization is useful to maintain large volume of data with continuous backup facility” (p. 01). Bele and Desai (2012) look at the idea or concept of virtualization beyond mere server virtualization and gives us other lines to explore when looking at the value of virtualization on the whole. Any technology that really permeates the market typically is able to perform multiple functions, but is usually part of a larger technology. Schultz (2009) looks at the beginnings, present, and future of virtualization. Schultz (2009) gives this insight, "With information virtualization, an enterprise is able to assemble a single view, or profile, of a client by bringing together information stored in multiple repositories. Virtualizing the workspace is the next logical step, Bishop says. While leading edge enterprises are striving toward this virtual nirvana, the majority of companies are baby-stepping their way through current-generation virtualization projects. What's next for them is more about growing the virtual server environment, integrating virtualization across servers, storage and the network, extending virtualization to the desktop" (p. 01). Schultz (2009) shows some of the changes in virtualization technology in the past and is key to my report by helping to detail and shape the "where did we come from" perspective. However, Schultz (2009) also gives a sense of where we are going with the technology, specifically the subject of desktop virtualization. Norall (2007) gives this insight on the potential and advantages of storage virtualization, "In addition to creating storage pools composed of physical disks from different arrays, storage virtualization provides a wide range of services, delivered in a consistent way. These stretch from basic volume management, including LUN masking, concatenation, and volume grouping and striping, to data protection and disaster recovery functionality, including snapshots and mirroring. In short, virtualization solutions can be used as a central control point for enforcing storage management policies and achieving higher SLAs." (p. 01). As with other aspects of virtualization, Norall (2007) makes the point that there are clear advantages to virtualizing your storage. Norall (2007) shows that virtualization as a technology encompasses more than just server virtualization, which is the most common modern use of the overall technology. However, Norall (2007) shows that there are many aspects of virtualization and that it can be applied to many different technologies within the computer science spectrum

Peggy (2007) makes the connection between virtualization and cost savings as follows: "There are three key areas of potential benefits: space, time and money. Fewer systems deployed results in lower capital costs. If companies can put 10 applications on 10 machines onto one or two machines, not only is that less money spent on computers, but the amount of money spent on electricity to power and cool these boxes goes down, freeing up precious data center space as well." (p. 01). Peggy (2007) discusses lowering costs through virtualization, which is one of the main advantages of virtualization. Kontzer (2010) details a history and projected future for the technology, "More than a decade after VMware introduced the first software that enabled x86 virtualization, the question facing most IT executives is no longer whether they plan to virtualize, but how far they plan to go...The long-term potential of virtualization speaks to an issue that transcends server spread, budget concerns and any other barriers that might get in the way of a virtualization investment: The exponential growth of data is causing IT environments to burst at the seams." (p. 01). Kontzer (2010) explores not only the origin of the adoption of virtualization, but also the growth of virtualization in the industry, and what factors influence a company to make the investment in virtualization technology. Kovar (2008) reveals the research that shows the current growth pattern of the virtualization technology. Kovar (2008) writes, "Solution providers are turning server virtualization, one of the fastest-growing segments of the IT market, into their very own gravy train. According to two recent exclusive CMP Channel surveys, solution providers said server virtualization is becoming a larger part of their business and it's also quickly becoming the catalyst for a wide range of other service offerings, including disaster recovery and data center consolidation." (p. 01). Everyone is talking about virtualization and just how hot it is, but can that growth or interest be measured? Yes, and Kovar (2008) shows that measurement and quantifies the growth factor. Yoshida (2008) make the following revelation about storage virtualization," The initial approach to storage virtualization was to address virtualization in the storage-area network (SAN) because the SAN sat between the storage and servers and would cause the least disruption to these systems. However, after nearly a decade, this approach has not taken off while server virtualization has become widely accepted. The breakthrough came with the ability to virtualize physical logical unit numbers (LUNs) without the need to remap them by using a virtualization technique based on storage control units. This approach to storage virtualization is simple to implement. Storage virtualization will deliver significant efficiencies, cost savings, power and cooling benefits, as well as greater agility in aligning storage infrastructure to business requirements." (p. 01). One of the aspects of Yoshida's (2008) insight is to show that virtualization as a technology extends well beyond simple server virtualization. Dubie (2009) goes deep into the concept and value of desktop virtualization, "Successful server virtualization deployments lead many IT managers to believe desktop virtualization would provide the same benefits. While that is partly true, companies need to be aware of how the two technologies differ" (p. 01). Dubie (2009) rightly points out that desktop virtualization is a technology not to be thought out lightly. Many industry experts focus on server virtualization, but Dubie (2009) suggests that desktop virtualization is the next big movement for virtualization. Dubie (2009) delves deep into the concept of desktop virtualization and reveals it to be a valid path for the mainstream IT department to pursue, but with caution and understanding. Kennedy (2007) offers real solutions to the issue of desktop deployment, "Despite rumors to the contrary, virtualization is not just for the datacenter. From the most complex workstation applications to the simplest DLLs, virtualization is leaving an indelible mark on client computing. The idea behind application virtualization is to eliminate many of the support-draining configuration problems that plague conventional desktop implementations." (p.01) Kennedy (2007) takes on desktop virtualization from a slightly different perspective. Kennedy (2007) makes the point that desktop virtualization will continue to grow in the future. Hsieh (2008) lays out a strategy that is key to any virtualization project, as follows, "Virtualization has become one of the hottest information technologies in the past few years. Yet, despite the proclaimed cost savings and efficiency improvement, implementation of the virtualization involves high degree of uncertainty, and consequently a great possibility of failures. Experience from managing the VMware based project activities at several companies are reported as the examples to illustrate how to increase the chance of successfully implementing a virtualization project" (p.01). After all of the research, you need to be able to put it all together. Hsieh (2008) covers just that, a way to put it all together and implement a successful virtualization project.

CHAPTER 5
ILLUSTRATION OF EXPERIMENTAL SETUP

5.1 TOOLS / TECHNOLOGIES FOR THE IMPLEMENTATION

- JDK 8
- Advance Java Code and Snippets
- Eclipse IDE
- CloudSim
- CloudAnalyst
- GridSim
- JUnit
- JCharts
- Dia
- Apache Ant

Java

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented,and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2015, Java is one of the mostpopular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems'Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

One design goal of Java is portability, which means that programs written for the Java platform must run similarly on any combination of hardware and operating system with adequate runtime support. This is achieved by compiling the Java language code to an intermediate representation called Java bytecode, instead of directly to architecture-specificmachine code. Java bytecode instructions are analogous to machine code, but they are intended to be executed by avirtual machine (VM) written specifically for the host hardware. End users commonly use a Java Runtime Environment(JRE) installed on their own machine for standalone Java applications, or in a web browser for Java applets.

Standard libraries provide a generic way to access host-specific features such as graphics, threading, and networking.

The use of universal bytecode makes porting simple. However, the overhead of interpreting bytecode into machine instructions makes interpreted programs almost always run more slowly than native executables. However, just-in-time(JIT) compilers that compile bytecodes to machine code during runtime were introduced from an early stage. Java itself is platform-independent, and is adapted to the particular platform it is to run on by a Java virtual machine for it, which translates the Java bytecode into the platform's machine language.

Apache Ant

Apache Ant is a software tool for automating software build processes. It originally came from the Apache Tomcat project in early 2000. It was a replacement for the unix make build tool, and was created due to a number of problems with the unix make. It is similar to Make but is implemented using the Java language, requires the Java platform, and is best suited to building Java projects.

The most immediately noticeable difference between Ant and Make is that Ant uses XML to describe the build process and its dependencies, whereas Make uses Makefile format. By default the XML file is namedbuild.xml.

Ant is an Apache project. It is open source software, and is released under the Apache License.

shell commands which are specific to the platform on which Make runs. Different platforms require different shell commands. Ant solves this problem by providing a large amount of built-in functionality that is designed to behave the same on all platforms. For example, in the sample build.xml file above, the clean target deletes the classes directory and everything in it.

In a Makefile this would typically be done with the command:

```
rm -rf classes/
```

rm is a Unix-specific command unavailable in some other environments. Microsoft Windows, for example, would use:

```
rmdir /S /Q classes
```

In an Ant build file the same goal would be accomplished using a built-in command:

```
<delete dir="classes"/>
```

A second portability issue is a result of the fact that the symbol used to delimit elements of file system directory path components differs from one platform to another. Unix uses a forward slash (/) to delimit components whereas Windows uses a backslash (\). Ant build files let authors choose their favorite convention: forward slash or backslash for directories; semicolon or colon for path separators. It converts each to the symbol appropriate to the platform on which it executes.

CloudSim

Cloud computing is the leading approach for delivering reliable, secure, fault-tolerant, sustainable, and scalable computational services. Hence timely, repeatable, and controllable methodologies for performance evaluation of new cloud applications and policies before their actual development are reqruied. Because utilization of real testbeds limits the experiments to the scale of the testbed and makes the reproduction of results an extremely difficult undertaking, simulation may be used.

CloudSim goal is to provide a generalized and extensible simulation framework that enables modeling, simulation, and experimentation of emerging Cloud computing infrastructures and application services, allowing its users to focus on specific system design issues that they want to investigate, without getting concerned about the low level details related to Cloud-based infrastructures and services.

CloudSim is developed in the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, at the Computer Science and Software Engineering Department of the University of Melbourne.

Main features

- support for modeling and simulation of large scale Cloud computing data centers
- support for modeling and simulation of virtualized server hosts, with customizable policies for provisioning host resources to virtual machines
- support for modeling and simulation of energy-aware computational resources
- support for modeling and simulation of data center network topologies and message-passing applications
- support for modeling and simulation of federated clouds
- support for dynamic insertion of simulation elements, stop and resume of simulation
- support for user-defined policies for allocation of hosts to virtual machines and policies for allocation of host resources to virtual machines

Cloud Simulations

- Modeling and simulation of large scale Cloud computing data centers
- Modeling and simulation of virtualized server hosts, with customizable policies for provisioning host resources to virtual machines
- Modeling and simulation of energy-aware computational resources

- Modeling and simulation of data center network topologies and message-passing applications
- Modeling and simulation of federated clouds
- Dynamic insertion of simulation elements, stop and resume of simulation
- User-defined policies for allocation of hosts to virtual machines and policies for allocation of host resources to virtual machines

Scope of Simulation

• Data Centers	Load Balancing
• Cloudlets	Resource Provisioning
• Scheduling	Storage and Cost Factors
• Energy Optimization	

… and many others

Focus on specific system design issues that they want to investigate, without getting concerned about the low level details related to Cloud-based infrastructures and services.

CloudSim Plugins

• CloudSim	CloudAnalyst
• GreenCloud	iCanCloud
• MDCSim	NetworkCloudSim
• VirtualCloud	CloudMIG Xpress
• CloudAuction	CloudReports
• RealCloudSim	DynamicCloudSim
• WorkFlowSim	

5.2 METHODOLOGY USED

- Collection and Initialization of Training DataSets with dynamic initialization of the cloud elements for deep investigation and research
- Implementation of the Algorithmic Approach using Classical as well as Proposed Techniques

- Applying Genetic Algorithm on the end of proposed approach with the integration of dynamic key to improve the security in virtualization.
- Application of the proposed algorithmic approach and model on cloud elements for effective results
- Fetching of Results and storage in the file system
- Data Cleaning, Analysis, Interpretation and Predictions

5.3 IMPLEMENTATION ASPECTS

The implementation is done in Eclipse IDE with Cloud and Grid Simulators in the External JAR Libraries for compatibility with Cloud and Grid Infrastructure. It is found and concluded from the results that the proposed algorithm is having effective results in cumulative way if we consider all the parameters in investigation as a whole level.

- The existing and base approaches of storage virtualization with security and scheduling are not effective having huge pitfalls more execution time
- The basic solution obtained without swarm intelligence is not efficient in terms of the turnaround time and optimal results
- To investigate the drawbacks and shortcomings in the classical approach in the cloud environment.
- To propose and implement a novel technique for the simulation of genetic algorithm based integration.

In the complete implementation and research task, following aspects and reference is used regarding classical and proposed approach

CHAPTER 6
RESEARCH METHODOLOGY AND PROPOSED WORK

In the proposed work of improved storage virtualization, security and scheduling, we have built up a special and novel calculation making utilization of crossover and parallel methodology of taking care of and preparing the employments in succession. This is one of key process in the genetic algorithms.

We have proposed and actualized the measurements for the ideal use of the considerable number of processors in a cloud framework to propose and actualize the Hybrid and in addition parallel methodologies of booking calculation and assess the proposed cloud based virtualization which are taken from driving computational focuses.

The significant key idea or thought of the proposed calculations is to execute employments ideally with the goal that there is the best mix of normal holding up, turnaround and effectiveness and expense.

The classical virtualization techniques are having number of pitfalls that takes more execution time

The basic solution obtained without swarm intelligence is not efficient in terms of the turnaround time and optimal results

The classical results are not effective in terms of multiple parameters including

- Turnaround time
- Complexity
- Waiting time
- Throughput
- Cost factor
- Performance

6.1 AIMS AND PROPOSED WORK

This research work is having the focus on development as well as implementation of a job scheduling algorithm for effective scheduling and prioritization. By this way and the proposed technique, the effective results in terms of less cost, higher performance and less execution time is obtained.

In this research work, the implementations have analyzed the performance of assorted algorithms based on new parameters and dimensions. There is need to analyze the performance of job scheduling algorithms on cloud as well as grid environment as priority, load balancing and related aspects can affect the performance of cloud to a huge extent.

The work done in this research includes proposing the implementation of algorithms on new parameters so that the hidden aspects and attributes of the algorithms can be analyzed.

The work includes performing the implementation / simulation and analysis on algorithms on new parameters beyond the work in the base literature.

6.2 KEY POINTS OF THE PROPOSED WORK

- The proposed algorithm is implemented and integrated using genetic algorithm for optimization of the results including the cost and performance factor.
- The proposed system is generating efficient results in terms of the optimal solution when executed using genetic algorithm.
- The proposed technique is efficient also in terms of the execution and turnaround time despite of the number of iterations
- The limitations may be included regarding the proposed work in terms of its further enhancement using assorted metaheuristics.
- The proposed system gives better results in executed using genetic algorithm that is one of the prominent metaheuristic techniques.

6.3 PROPOSED APPROACH

- *Activate and Initialization of the cloud elements for analysis*
    - *Virtual Machines*
    - *Cloudlets*
    - *Data Centers*
    - *Keys*
- *A cloudlet is a mobility-enhanced small-scale cloud datacenter that is located at the edge of the scenario. The main purpose of the cloudlet is supporting resource-intensive and interactive mobile applications by providing powerful computing resources to mobile devices with lower latency.*
- *A data center is a facility used to house computer systems and associated components, such as telecommunications and storage systems. It generally includes redundant or backup power supplies, redundant data communications connections, environmental controls (e.g., air conditioning, fire suppression) and various security devices.*
- *A virtual machine (VM) is an operating system OS or application environment that is installed on software which imitates dedicated hardware.*
- *To design the blocks of the computation cost and related parameters as well as communication.*
- *In this panel and simulation block, the computational cost is initialized to null so that the dynamic values can be flooded after simulation*
- *Assignment of unique priorities to tasks in the virtual machine environment with the data centers.*
- *The dynamic data centers are set up and activated so that the simulation takes place and desirable results can be fetched.*
- *Calculation of the Parameters*
- *The simulation is associated with earliest start time and earliest finish time of the processes or cloudlets.*
- *Activation and Implementation of the Genetic Algorithm and its aspects*
    - *Activation of Selection, Crossover and Mutation*
    - *Selection of the Best Candidate for the Effective Virtualization Approach*

- *Effective and Pragmatic Comparative Analysis between the classical (existing) and proposed Approach*

6.4 GENETIC ALGORITHM - A PROMINENT METAHEURISTIC APPROACH

This section underlines and highlights the positive and key advantages of genetic algorithms due to which we have adopted this technique for our research work.

This section depicts the applications of genetic algorithm which motivated us to implement this approach in our research work.

Efficient and perfect solution to the combinatorial optimization problems in different streams have been an area of research from long time. Engineering, Industrial, Economical and Scientific problems such as Transportation, Bioinformatics, Logistics, Scheduling, Timetabling, Vehicle Routing, Resource Allocation and many other are tackled with various approaches such as Simulated Annealing, Tabu Search, Genetic Algorithms, Ant Colony Optimization, Harmony Search, Scatter Search or Iterated Local Search. These techniques known as metaheuristics presents itself as highly promising choice for nearly-optimal solutions in reasonable time where exact approaches are not applicable due to extremely large running times or other limitations. Metaheuristic is an excellent strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality. This section highlights the efficiency of a metaheuristic approach, Genetic Algorithm, which is being used by the researchers now days in several Engineering, Scientific, Business and Industrial applications.

Metaheuristics are used to solve Combinatorial Optimization Problems, like Bin Packing, Network Routing, Network Design, Assignment Problem, Scheduling, or Time-Tabling Problems, Continuous Parameter Optimization Problems, or Optimization of Non-Linear Structures like Neural Networks or Tree Structures as they often appear in Computational Intelligence.

Metaheuristics are generally applied to problems for which there is no satisfactory problem-specific algorithm or heuristic; or when it is not practical to implement such a method. Most commonly used Metaheuristics are focused to combinatorial optimization problems, but obviously can handle any problem that can be recast in that form, such as solving Boolean equations.

HEURISTICS AND METAHEURISTICS

Heuristic refers to “discover”. A Heuristic is used when

1. Exact method are not on any help, due to execution time
2. There are errors in input data or is unreliable
3. Improvement in the performance of exact methods is required
4. There is need of a solution after a limited period of time.
5. We have to choose between addressing a more realistic model and provide an approximate solution instead of a simpler, unrealistic model that we can prove that can solve to optimality.
6. There is need of good starting points for an exact method.

DISADVANTAGES OF USING HEURISTICS

1. In many cases, convergence is generally guaranteed
2. Optimality may be achieved but it is not proved
3. In many cases, they may not be able to generate a feasible solution.

Metaheuristics are said to be high level procedures which coordinate simple heuristics such as local search, to find solutions that are of better quality than those found by simple heuristics done.

COMMONLY USED METAHEURISTICS

- Tabu search [Glover, 90]
- Simulated Annealing [Kirckpatrick, 83]
- Threshold accepting [Deuck, Scheuer, 90]
- Variable neighborhood [Hansen, Mladenovi´c, 98]
- Iterated local search [Loren¸co et al, 2000]

- Genetic Algorithm [Holland Goldberg]
- Memetic Algorithm, Moscatto 1989
- Ant Colony Optimization, Dorigo 1991
- Scatter search, Laguna, Glover, Marty 2000

Main Features of a Good Metaheuristics

- Population intrinsic parallelism
- Indirect Coding
- Cooperation adapted crossover
- Local search in solution space
- Diversity need to be controlled
- Easy to implement the restarts
- Randomness

Countless variants and hybrids of these techniques have been proposed, and many more applications of Metaheuristics to specific problems have been reported. This is one of the active fields of research, with a considerable literature, a large community of researchers and users, and a wide range of applications.

Traditional methods of search and optimization are too slow in finding a solution in a very complex search space, even implemented in supercomputers. Metaheuristics consist of number of methods and theories having robust search method requiring little information to search effectively in a large or poorly-understood search space. There exists an extensive range of problems which can be formulated as obtaining the values for a vector of variables subject to some restrictions. The elements of this vector are denominated decision-variables, and their nature determines a classification of this kind of problems. Specifically, if decision-variables are required to be discrete, the problem is said to be combinatorial. The process of finding optimal solutions (maximizing or minimizing an objective function) for such a problem is called combinatorial optimization.

Combinatorial optimization problems have been traditionally approached using exact techniques such as Branch and Bound (Lawler and Wood, 1966). Finding the optimal solution is ensured with these techniques but, unfortunately, they are seriously limited in their application due to the so-called combinatorial explosion. As an example, consider the Traveling Salesman Problem (TSP). This problem (obtaining a minimal Hamiltonian tour through a complete graph of n nodes) is a classical example of NP-complexity: the work-area to be explored grows exponentially according with the number of nodes in the graph, and so does the complexity of every know algorithm to solve this problem. It is not only a good example of a combinatorial optimization problem, but also an approach to real problems like VLSI-design or X-ray Crystallography.

GENETIC ALGORITHM

Genetic algorithms (GAs) are search methods based on principles of natural selection and genetics (Fraser, 1957; Bremermann, 1958; Holland, 1975). We start with a brief introduction to simple genetic algorithms and associated terminology.

GAs encode the decision variables of a search problem into finite-length strings of alphabets of certain cardinality. The strings which are candidate solutions to the search problem are referred to as chromosomes, the alphabets are referred to as genes and the values of genes are called alleles. For example, in a problem such as the traveling salesman problem, a chromosome represents a route, and a gene may represent a city. In contrast to traditional optimization techniques, GAs work with coding of parameters, rather than the parameters themselves.

To evolve good solutions and to implement natural selection, we need a measure for distinguishing good solutions from bad solutions. The measure could be an objective function that is a mathematical model or a computer simulation, or it can be a subjective function where humans choose better solutions over worse ones. In essence, the fitness measure must determine a candidate solution's relative fitness, which will subsequently be used by the GA to guide the evolution of good solutions.

Another important concept of GAs is the notion of population. Unlike traditional search methods, genetic algorithms rely on a population of candidate solutions. The population size, which is usually a user-specified parameter, is one of the important factors affecting the scalability and performance of genetic algorithms. For example, small population sizes might lead to premature convergence and yield substandard solutions. On the other hand, large population sizes lead to unnecessary expenditure of valuable computational time. Once the problem is encoded in a chromosomal manner and a fitness measure for discriminating good solutions from bad ones has been chosen, we can start to evolve solutions to the search problem using the following steps:

Initialization. The initial population of candidate solutions is usually generated randomly across the search space. However, domain-specific knowledge or other information can be easily incorporated.

Evaluation. Once the population is initialized or an offspring population is created, the fitness values of the candidate solutions are evaluated.

Selection. Selection allocates more copies of those solutions with higher fitness values and thus imposes the survival-of-the-fittest mechanism on the candidate solutions. The main idea of selection is to prefer better solutions to worse ones, and many selection procedures have been proposed to accomplish this idea, including roulette-wheel selection, stochastic universal selection, ranking selection and tournament selection, some of which are described in the next section.

Recombination. Recombination combines parts of two or more parental solutions to create new, possibly better solutions (i.e. offspring). There are many ways of accomplishing this (some of which are discussed in the next section), and competent performance depends on a properly designed recombination mechanism. The offspring under recombination will not be identical to any particular parent and will instead combine parental traits in a novel manner (Goldberg, 2002).

Mutation. While recombination operates on two or more parental chromosomes, mutation locally but randomly modifies a solution. Again, there are many variations of mutation, but it usually involves one or more changes being made to an individual's trait or traits. In other words, mutation performs a random walk in the vicinity of a candidate solution.

Replacement. The offspring population created by selection, recombination, and mutation replaces the original parental population. Many replacement techniques such as elitist replacement, generation-wise replacement and steady-state replacement methods are used in GAs.

Goldberg (1983, 1999a, 2002) has likened GAs to mechanistic versions of certain modes of human innovation and has shown that these operators when analyzed individually are ineffective, but when combined together they can work well. This aspect has been explained with the concepts of the fundamental intuition and innovation intuition. The same study compares a combination of selection and mutation to continual improvement (a form of hill climbing), and the combination of selection and recombination to innovation (crossfertilizing). These analogies have been used to develop a design-decomposition methodology and so-called competent GAs—that solve hard problems quickly, reliably, and accurately—both of which are discussed in the subsequent sections.

Basic Genetic Algorithm Operators

In this section we describe some of the selection, recombination, and mutation operators commonly used in genetic algorithms.

Selection Methods

Selection procedures can be broadly classified into two classes as follows.

Fitness Proportionate Selection

This includes methods such as roulette-wheel selection (Holland, 1975; Goldberg, 1989b) and stochastic universal selection (Baker, 1985; Grefenstette and Baker, 1989). In roulette-wheel selection, each individual in the population is assigned a roulette wheel slot sized in proportion to its fitness. That is, in the biased roulette wheel, good solutions have a larger slot size than the less fit solutions. The roulette wheel is spun to obtain a reproduction candidate.

The roulettewheel selection scheme can be implemented as follows:

1 Evaluate the fitness, f_i , of each individual in the population.

2 Compute the probability (slot size), p_i , of selecting each member of the population: $p_i = fi/\sum^{n}_{j=1} f_j$, where n is the population Size.

3 Calculate the cumulative probability, qi , for each individual: $q_i = \sum^{i}_{j=1} p_j$.

4 Generate a uniform random number, $r \in (0, 1]$.

5 If $r < q1$ then select the first chromosome, x1, else select the individual xi such that $qi-1 < r \leq qi$.

6 Repeat steps 4–5 n times to create n candidates in the mating pool.

To illustrate, consider a population with five individuals (n = 5), with the fitness values as shown in the table below. The total fitness, $\sum^{n}_{j=1} f_j = 28+18+14+9+26 = 95$. The probability of selecting an individual and the corresponding cumulative probabilities are also shown in the table below.

Cumulative Probabilities of Selecting an Individual

Chromosome #	1	2	3	4	5
Fitness, *f*	28	18	14	9	26
Probability, p_i	28/95 = 0.295	0.189	0.147	0.095	0.274
Cumulative probability, *Qi*	0.295	0.484	0.631	0.726	1.000

Now if we generate a random number r, say 0.585, then the third chromosome is selected as $q2 = 0.484 < 0.585 \leq q3 = 0.631$.

Ordinal Selection

This includes methods such as tournament selection (Goldberg et al., 1989b), and truncation selection (M¨uhlenbein and Schlierkamp-Voosen, 1993). In tournament selection, s chromosomes are chosen at random (either with or without replacement) and entered into a tournament against each other. The fittest individual in the group of k chromosomes wins the tournament and is selected as the parent. The most widely used value of s is 2. Using this selection scheme, n tournaments are required to choose n individuals. In truncation selection, the top (1/s)th of the individuals get s copies each in the mating pool.

Crossover

After selection, individuals from the mating pool are recombined (or crossed over) to create new, hopefully better, offspring. In the GA literature, many crossover methods have been designed (Goldberg, 1989b; Booker et al., 1997; Spears, 1997) and some of them are described in this section.

k-Point Crossover

One-point, and two-point crossovers are the simplest and most widely applied crossover methods. In one-point crossover, a crossover site is selected at random over the string length, and the alleles on one side of the site are exchanged between the individuals. In two-point crossover, two crossover sites are randomly selected.

Uniform Crossover

Another common recombination operator is uniform crossover (Syswerda, 1989; Spears and De Jong, 1994). In uniform crossover, every allele is exchanged between the a pair of randomly selected chromosomes with a certain probability, pe, known as the swapping probability. Usually the swapping probability value is taken to be 0.5.

Uniform order Based Crossover

The k-point and uniform crossover methods described above are not well suited for search problems with permutationcodes such as the ones used in the traveling salesman problem. They often create offspring that represent invalid solutions for the search problem.

Order Based Crossover

The order-based crossover operator (Davis, 1985) is a variation of the uniform order-based crossover in which two parents are randomly selected and two random crossover sites are generated.

Partially Match Crossover (PMX)

Apart from always generating valid offspring, the PMX operator (Goldberg and Lingle, 1985) also preserves orderings within the chromosome. In PMX, two parents are randomly selected and two random crossover sites are generated. Alleles within the two crossover sites of a parent are exchanged with the alleles corresponding to those mapped by the other parent.

MUTATION

If we use a crossover operator, such as one-point crossover, we may get better and better chromosomes but the problem is, if the two parents (or worse, the entire population) has the same allele at a given gene then one-point crossover will not change that.

CHAPTER 7
RESULTS AND THEIR POTENTIAL IMPLICATIONS

This research work is based on the optimization of cost and time as mentioned in the base paper. The base paper work is taking more time and higher cost. In the proposed approach using GA, the cost is reduced to atleast 30% which makes it robust cloud environment.

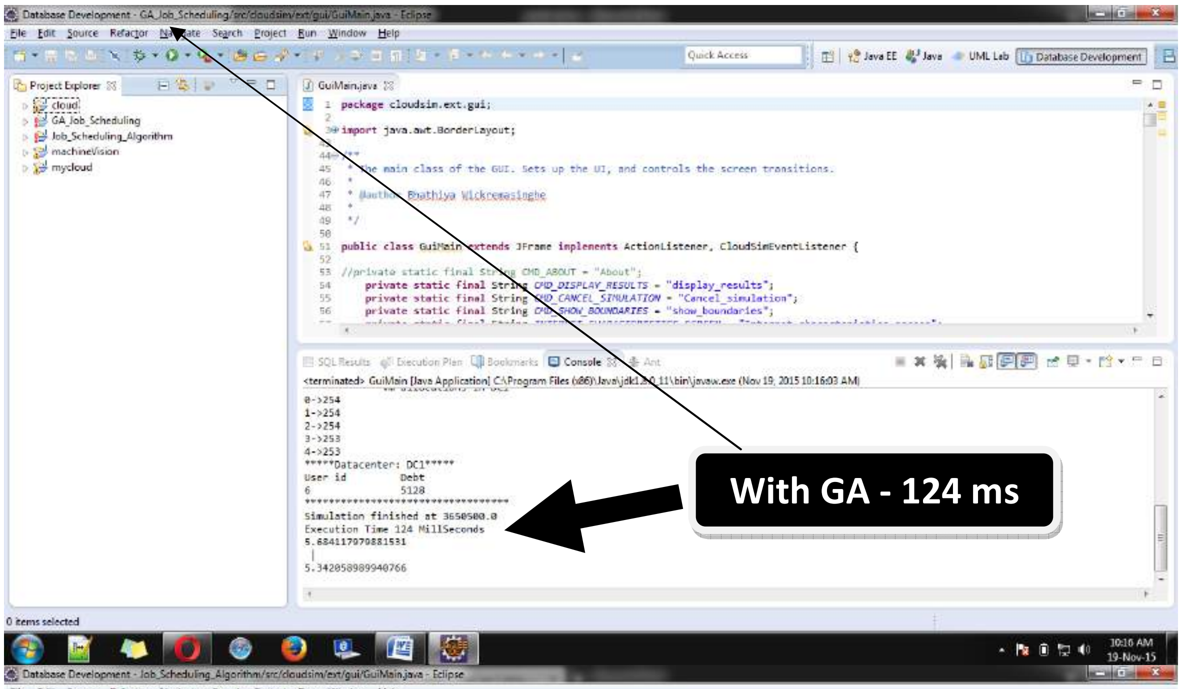
KEY POINTS ACCOMPLISHED

- To perform the detailed literature analysis and comparison between assorted approaches for cloud parameters optimization (it will be written in the final thesis report)
- To implement the existing approach as in the base work by Kumar et. al. so that the simulated environment can be investigated
- To propose a unique and improved algorithm for optimization of parameters in the base work.
- To implement the proposed work so that the comparative analysis between Existing and proposed approach can be accomplished.

7.1 ADVANTAGES OF THE PROPOSED SYSTEM

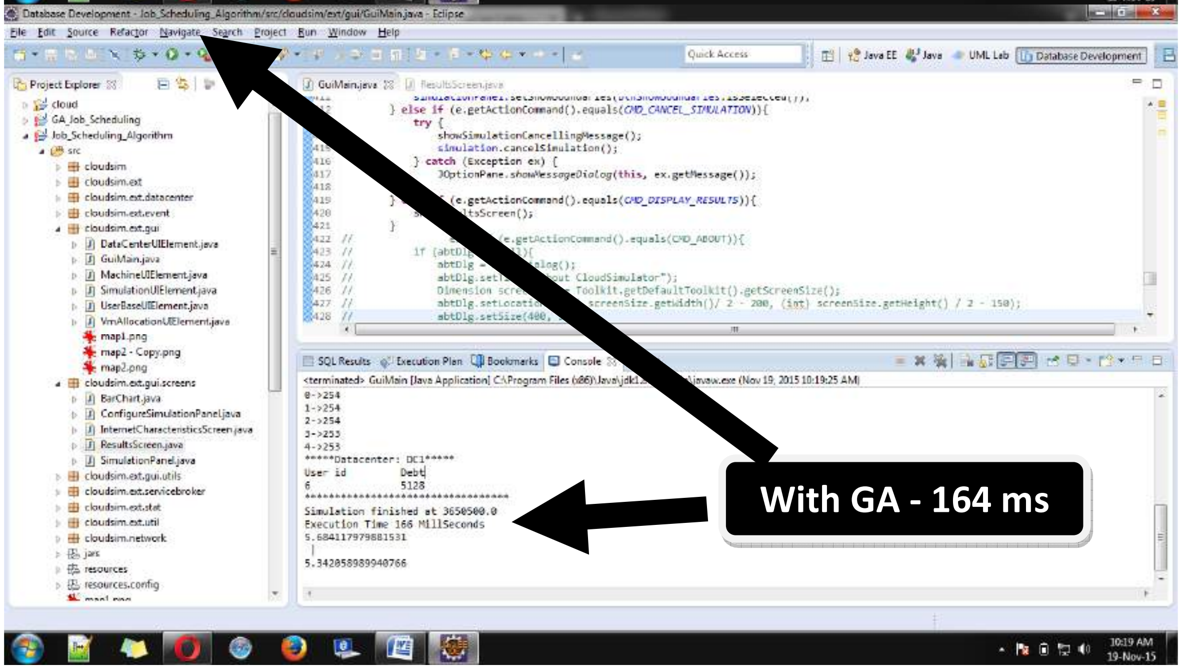
- The proposed system is generating efficient results in terms of the optimal solution when executed using genetic algorithm.
- The proposed technique is efficient also in terms of the execution and turnaround time despite of the number of iterations
- The limitations may be included regarding the proposed work in terms of its further enhancement using assorted metaheuristics.
- The proposed system may give better results in executed using simulated annealing that is one of the prominent metaheuristic technique.

Overall turnaround time in GA is very less as compared to the Existing approach.



```
package cloudsim.ext.gui;

import java.awt.BorderLayout;

/**
 * The main class of the GUI. Sets up the UI, and controls the screen transitions.
 *
 * @author Bhathiya Wickremasinghe
 *
 */

public class GuiMain extends JFrame implements ActionListener, CloudSimEventListener {

//private static final String CMD_ABOUT = "About";
    private static final String CMD_DISPLAY_RESULTS = "display_results";
    private static final String CMD_CANCEL_SIMULATION = "Cancel_simulation";
    private static final String CMD_SHOW_BOUNDARIES = "show_boundaries";
```

```
<terminated> GuiMain [Java Application] C:\Program Files (x86)\Java\jdk1.8.0_11\bin\javaw.exe (Nov 19, 2015 10:16:03 AM)
0->254
1->254
2->254
3->253
4->253
*****Datacenter: DC1*****
User id         Debt
6               5128
**********************************
Simulation finished at 3650500.0
Execution Time 124 MillSeconds
5.684117979881531
 |
5.342058989940766
```



```
        } else if (e.getActionCommand().equals(CMD_CANCEL_SIMULATION)){
            try {
                showSimulationCancellingMessage();
                simulation.cancelSimulation();
            } catch (Exception ex) {
                JOptionPane.showMessageDialog(this, ex.getMessage());

        }   [illegible] (e.getActionCommand().equals(CMD_DISPLAY_RESULTS)){
            [illegible]ltsScreen();
        }
//              [illegible] (e.getActionCommand().equals(CMD_ABOUT)){
//          if (abtDlg [illegible])){
//              abtDlg = [illegible]alog();
//              abtDlg.setTi[illegible]bout CloudSimulator");
//              Dimension scre[illegible] Toolkit.getDefaultToolkit().getScreenSize();
//              abtDlg.setLocation[illegible] screenSize.getWidth()/ 2 - 200, (int) screenSize.getHeight() / 2 - 150);
//              abtDlg.setSize(400, [illegible]
```

```
<terminated> GuiMain [Java Application] C:\Program Files (x86)\Java\jdk1.[illegible]\javaw.exe (Nov 19, 2015 10:19:25 AM)
0->254
1->254
2->254
3->253
4->253
*****Datacenter: DC1*****
User id         Debt
6               5128
**********************************
Simulation finished at 3650500.0
Execution Time 166 MillSeconds
5.684117979881531
 |
5.342058989940766
```

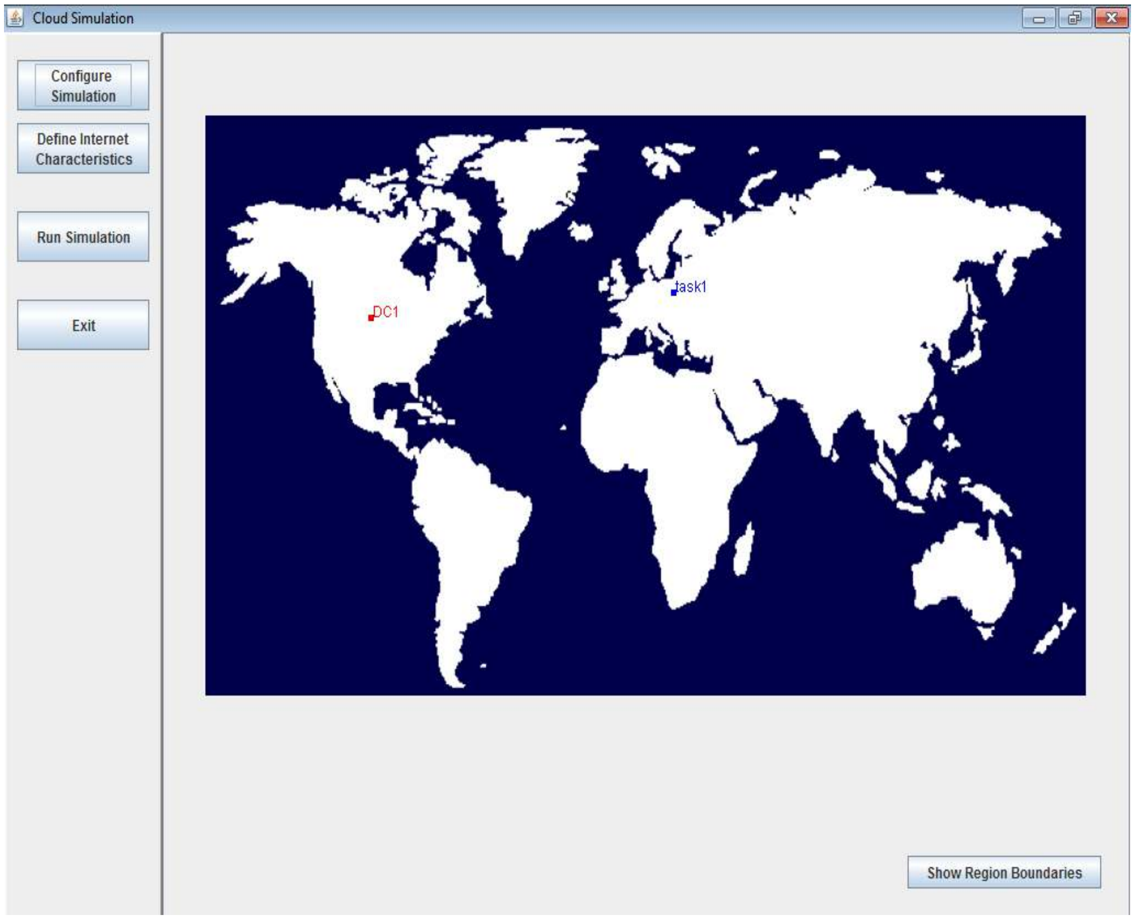
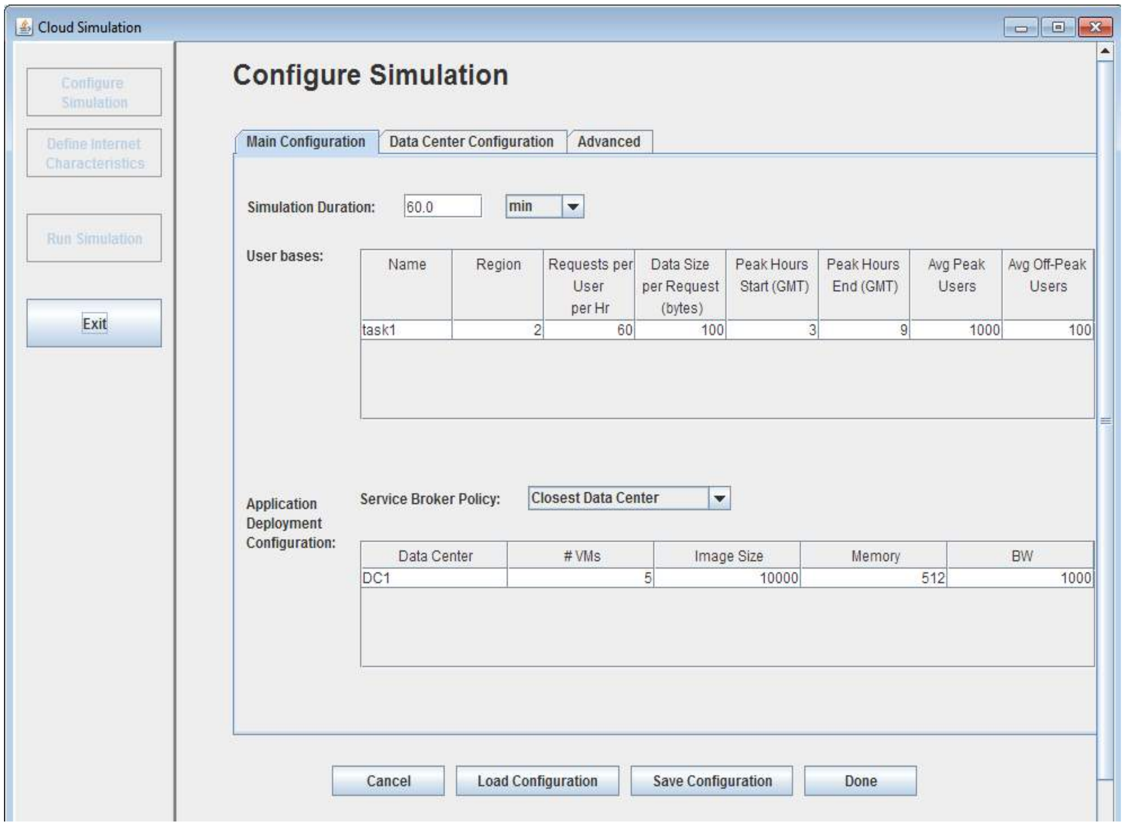


Fig. 7.1 – Simulation Scenario



Configure Simulation

Simulation Duration: 60.0 min

User bases:

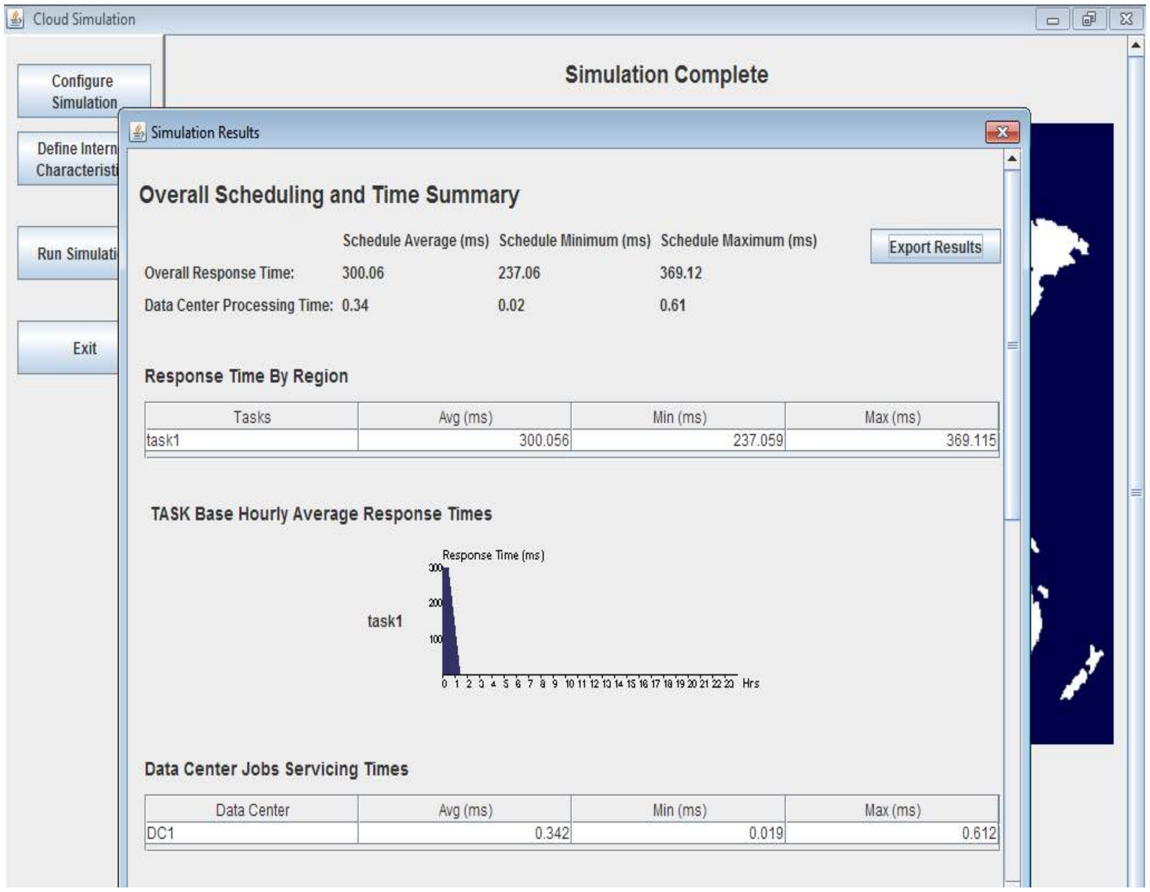
Name	Region	Requests per User per Hr	Data Size per Request (bytes)	Peak Hours Start (GMT)	Peak Hours End (GMT)	Avg Peak Users	Avg Off-Peak Users
task1	2	60	100	3	9	1000	100

Application Deployment Configuration:

Service Broker Policy: Closest Data Center

Data Center	# VMs	Image Size	Memory	BW
DC1	5	10000	512	1000

Fig. 7.2 – Simulation Configuration Panel



Overall Scheduling and Time Summary

	Schedule Average (ms)	Schedule Minimum (ms)	Schedule Maximum (ms)
Overall Response Time:	300.06	237.06	369.12
Data Center Processing Time:	0.34	0.02	0.61

Response Time By Region

Tasks	Avg (ms)	Min (ms)	Max (ms)
task1	300.056	237.059	369.115

TASK Base Hourly Average Response Times

Response Time (ms)
300
200
100
task1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 Hrs

Data Center Jobs Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.342	0.019	0.612

Fig. 7.3 – Simulation Completion Status

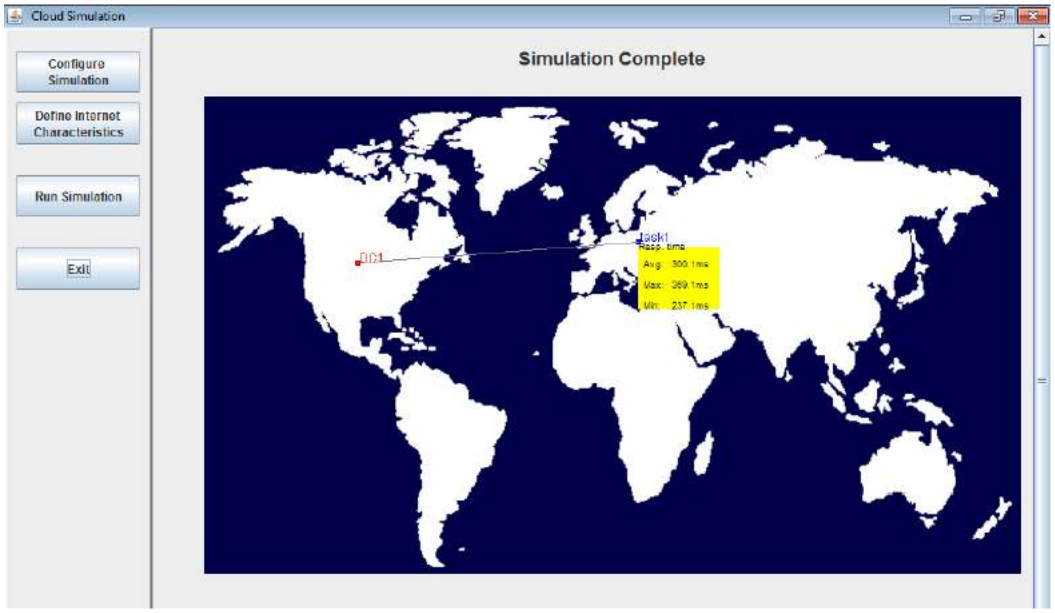
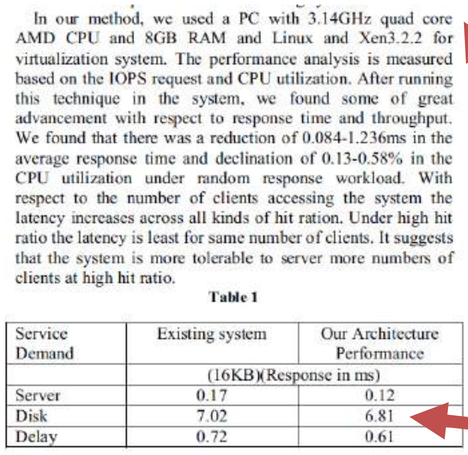


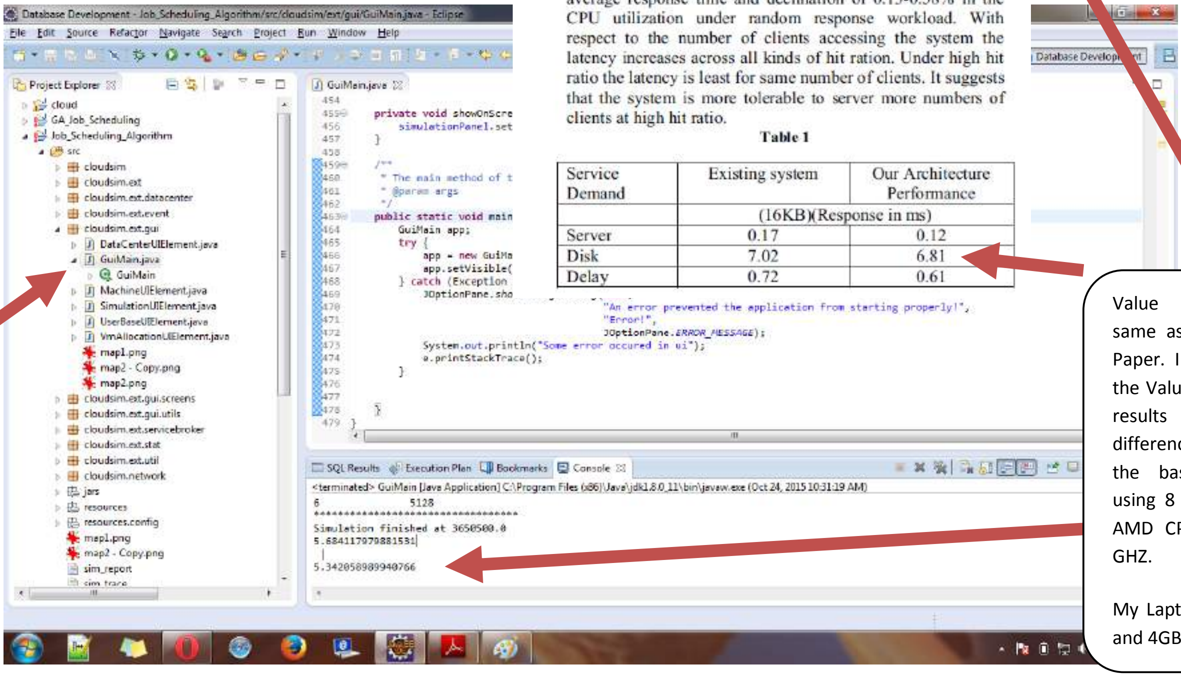
Figure 7.4 – Simulation Completion Status



In our method, we used a PC with 3.14GHz quad core AMD CPU and 8GB RAM and Linux and Xen3.2.2 for virtualization system. The performance analysis is measured based on the IOPS request and CPU utilization. After running this technique in the system, we found some of great advancement with respect to response time and throughput. We found that there was a reduction of 0.084-1.236ms in the average response time and declination of 0.13-0.58% in the CPU utilization under random response workload. With respect to the number of clients accessing the system the latency increases across all kinds of hit ration. Under high hit ratio the latency is least for same number of clients. It suggests that the system is more tolerable to server more numbers of clients at high hit ratio.

Table 1

Service Demand	Existing system	Our Architecture Performance
	(16KB)(Response in ms)	
Server	0.17	0.12
Disk	7.02	6.81
Delay	0.72	0.61



Database Development - Job_Scheduling_Algorithm/src/cloudsim/ext/gui/GuiMain.java - Eclipse

```
    private void showOnScre
        simulationPanel.set
    }

    /**
     * The main method of t
     * @param args
     */
    public static void main
        GuiMain app;
        try {
            app = new GuiMa
            app.setVisible(
        } catch (Exception
            JOptionPane.sho
                    "An error prevented the application from starting properly!",
                    "Error!",
                    JOptionPane.ERROR_MESSAGE);
            System.out.println("Some error occured in ui");
            e.printStackTrace();
        }

    }
}
```

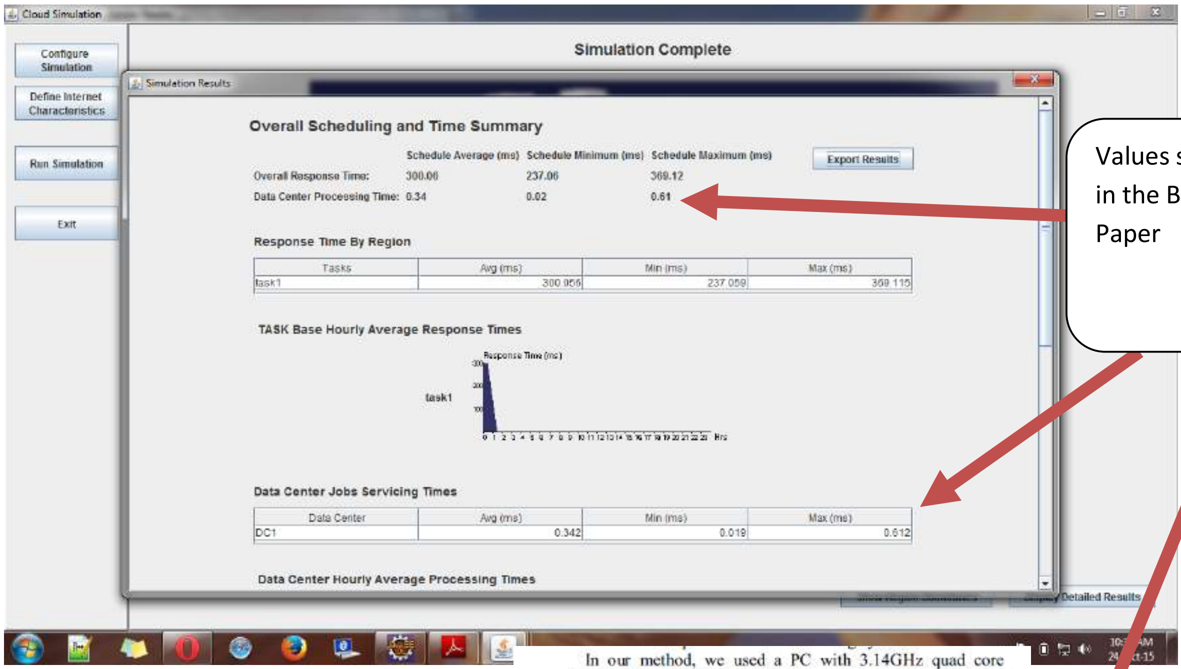
```
<terminated> GuiMain [Java Application] C:\Program Files (x86)\Java\jdk1.8.0_11\bin\javaw.exe (Oct 24, 2015 10:31:19 AM)
6               5128
************************************
Simulation finished at 3650500.0
5.684117079881531

5.342058989940766
```

RUN THE FILE

GuiMain.java

Value 5.34 almost same as in the Base Paper. In Base Paper the Value is 6.81. The results in very low difference because the base paper is using 8 GB RAM and AMD CPU with 3.14 GHZ.

My Laptop is 2.6 GHz and 4GB RAM



Overall Scheduling and Time Summary

	Schedule Average (ms)	Schedule Minimum (ms)	Schedule Maximum (ms)
Overall Response Time:	300.06	237.06	369.12
Data Center Processing Time:	0.34	0.02	0.61

Response Time By Region

Tasks	Avg (ms)	Min (ms)	Max (ms)
task1	300.056	237.059	369.115

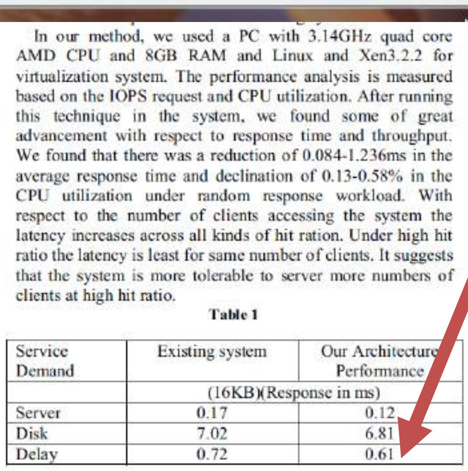
TASK Base Hourly Average Response Times

Data Center Jobs Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.342	0.019	0.612

Data Center Hourly Average Processing Times

Values same as in the Base Paper



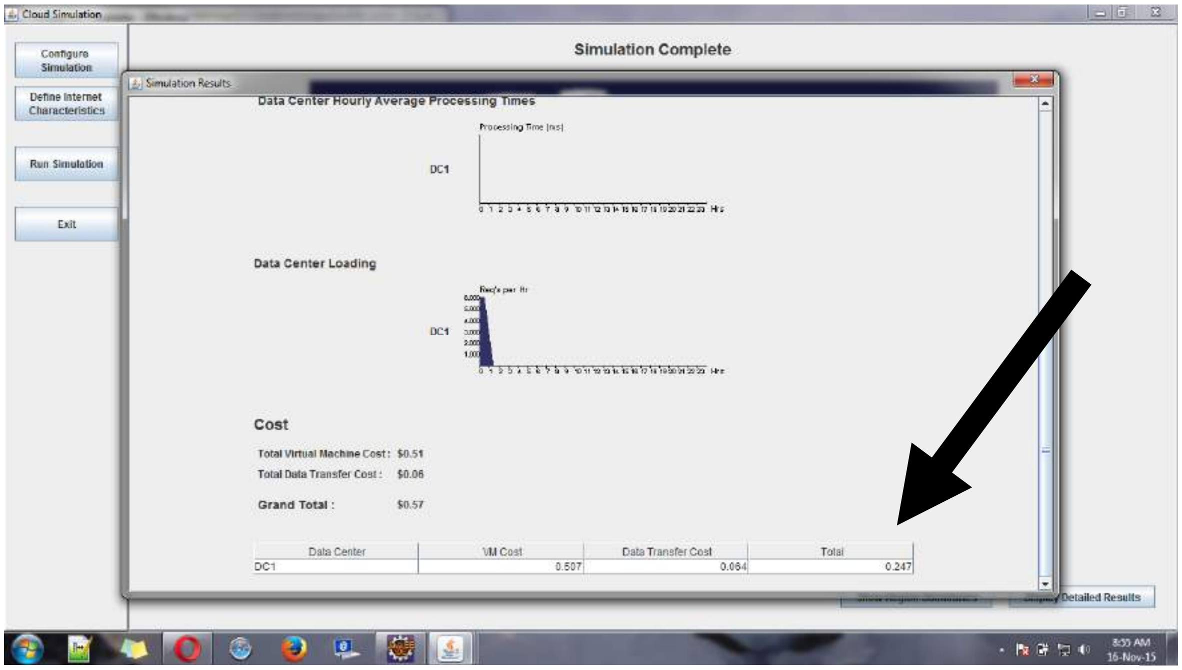
In our method, we used a PC with 3.14GHz quad core AMD CPU and 8GB RAM and Linux and Xen3.2.2 for virtualization system. The performance analysis is measured based on the IOPS request and CPU utilization. After running this technique in the system, we found some of great advancement with respect to response time and throughput. We found that there was a reduction of 0.084-1.236ms in the average response time and declination of 0.13-0.58% in the CPU utilization under random response workload. With respect to the number of clients accessing the system the latency increases across all kinds of hit ration. Under high hit ratio the latency is least for same number of clients. It suggests that the system is more tolerable to server more numbers of clients at high hit ratio.

Table 1

Service Demand	Existing system	Our Architecture Performance
	(16KB)(Response in ms)	
Server	0.17	0.12
Disk	7.02	6.81
Delay	0.72	0.61

Cost Factor **(IN USD)** is the mathematical model or function that is directly proportional to the execution time. If execution time / turnaround time is more than complexity and obviously cost will be more.

In our case, the execution time and turnaround time is very less and that's why is the cost will be less. In addition, the simulation based cost factor is shown as per the data center.



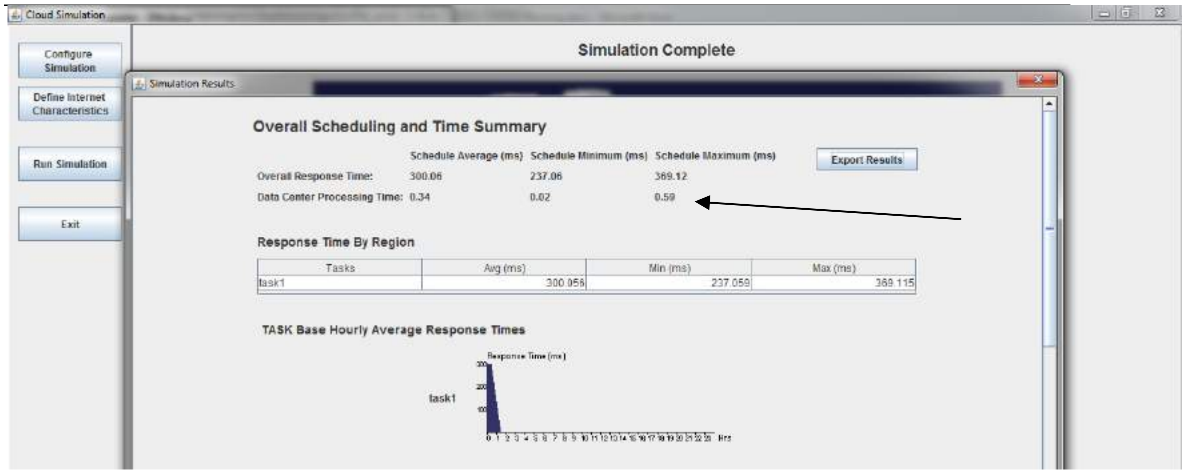
Cost

Total Virtual Machine Cost : $0.51

Total Data Transfer Cost : $0.06

Grand Total : $0.57

Data Center	VM Cost	Data Transfer Cost	Total
DC1	0.507	0.064	0.247

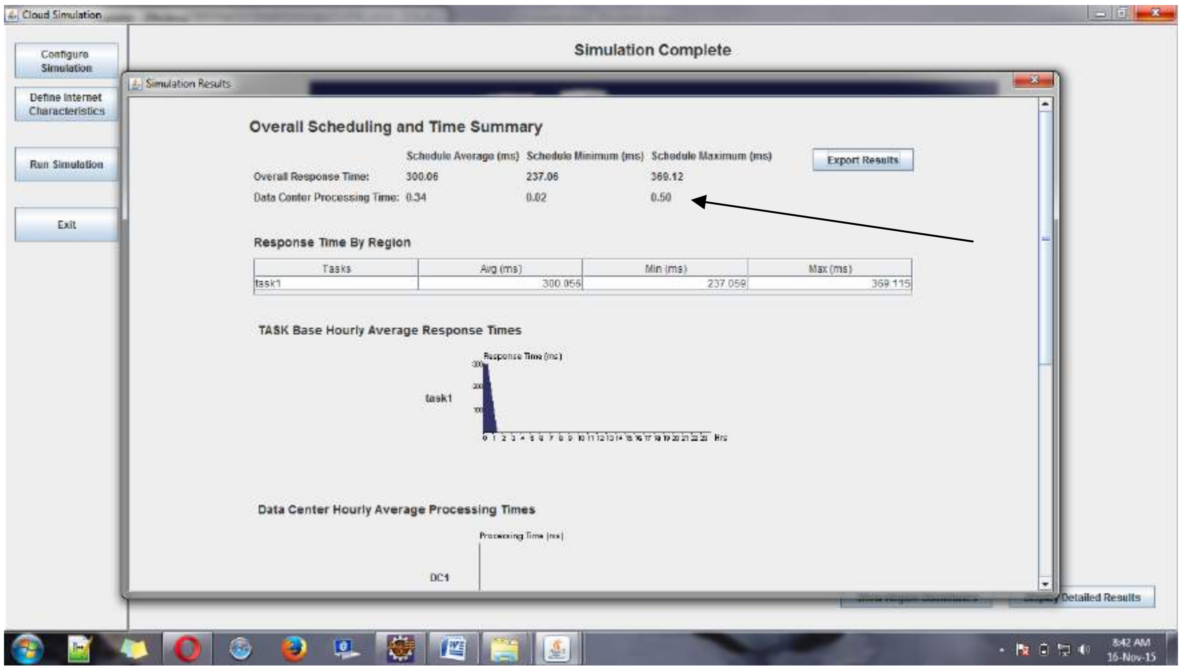


Overall Scheduling and Time Summary

	Schedule Average (ms)	Schedule Minimum (ms)	Schedule Maximum (ms)
Overall Response Time:	300.06	237.06	369.12
Data Center Processing Time:	0.34	0.02	0.59

Response Time By Region

Tasks	Avg (ms)	Min (ms)	Max (ms)
task1	300.056	237.059	369.115



Overall Scheduling and Time Summary

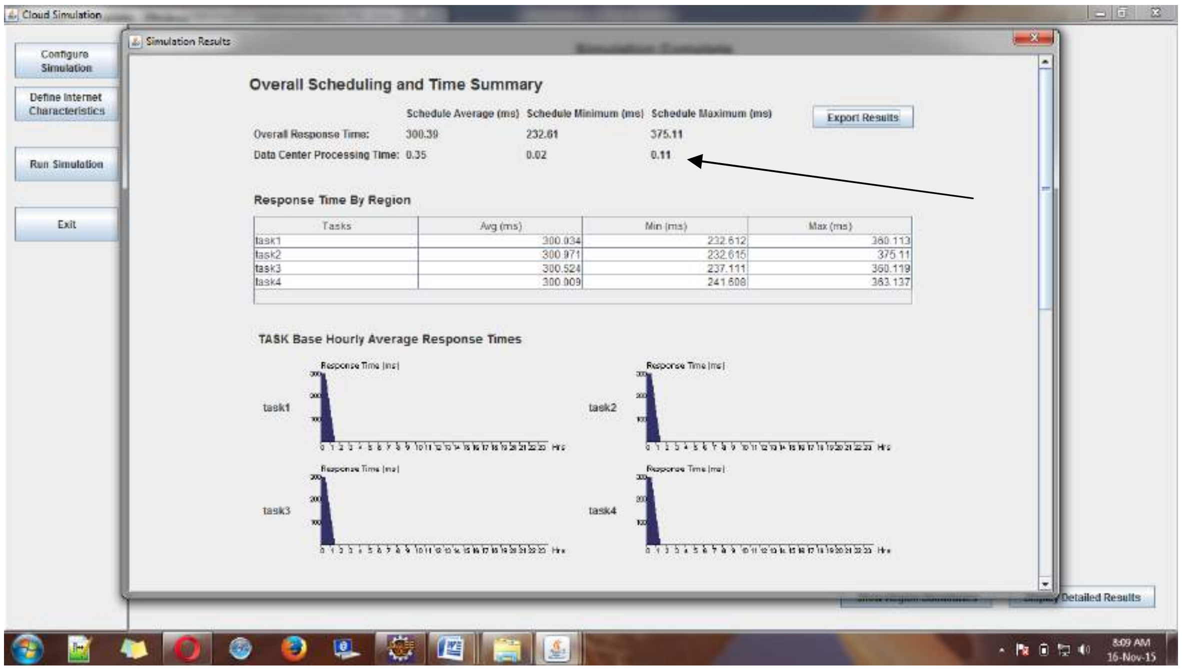
	Schedule Average (ms)	Schedule Minimum (ms)	Schedule Maximum (ms)
Overall Response Time:	300.06	237.06	369.12
Data Center Processing Time:	0.34	0.02	0.50

Response Time By Region

Tasks	Avg (ms)	Min (ms)	Max (ms)
task1	300.056	237.059	369.115

TASK Base Hourly Average Response Times

Data Center Hourly Average Processing Times



Overall Scheduling and Time Summary

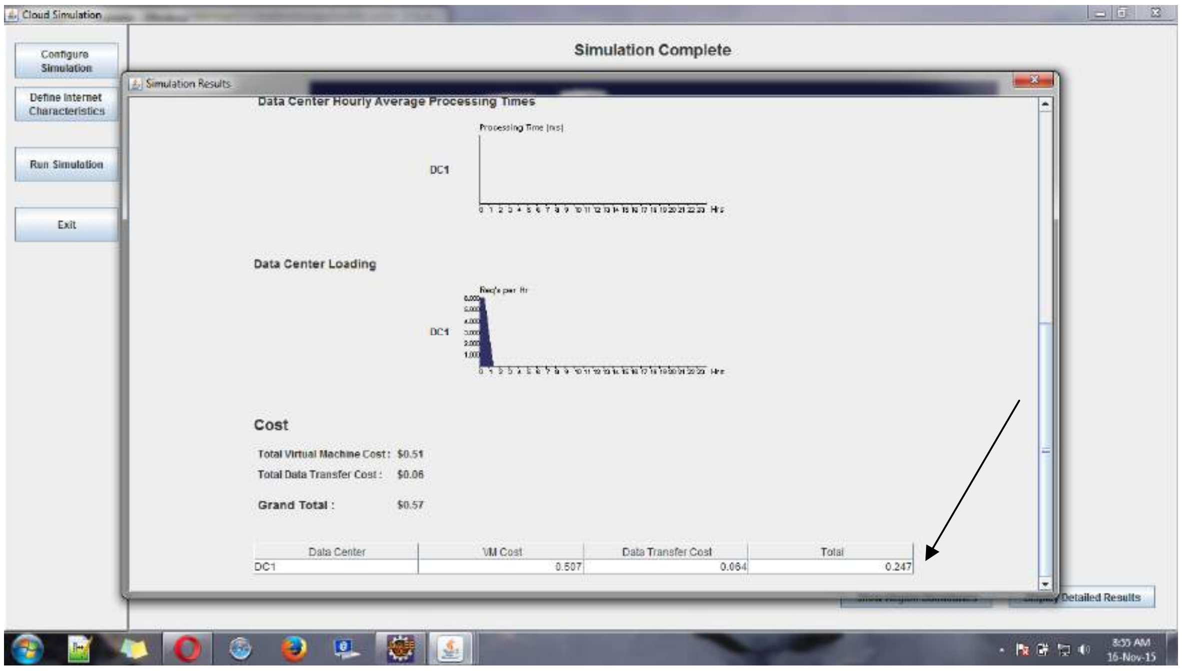
	Schedule Average (ms)	Schedule Minimum (ms)	Schedule Maximum (ms)
Overall Response Time:	300.39	232.61	375.11
Data Center Processing Time:	0.35	0.02	0.11

Response Time By Region

Tasks	Avg (ms)	Min (ms)	Max (ms)
task1	300.034	232.612	360.113
task2	300.971	232.615	375.11
task3	300.524	237.111	360.119
task4	300.009	241.608	363.137

TASK Base Hourly Average Response Times

COST (IN USD)-> In existing, the cost is around 0.57



Simulation Complete

Data Center Hourly Average Processing Times

Data Center Loading

Cost

Total Virtual Machine Cost : $0.51

Total Data Transfer Cost : $0.06

Grand Total : $0.57

Data Center	VM Cost	Data Transfer Cost	Total
DC1	0.507	0.064	0.247

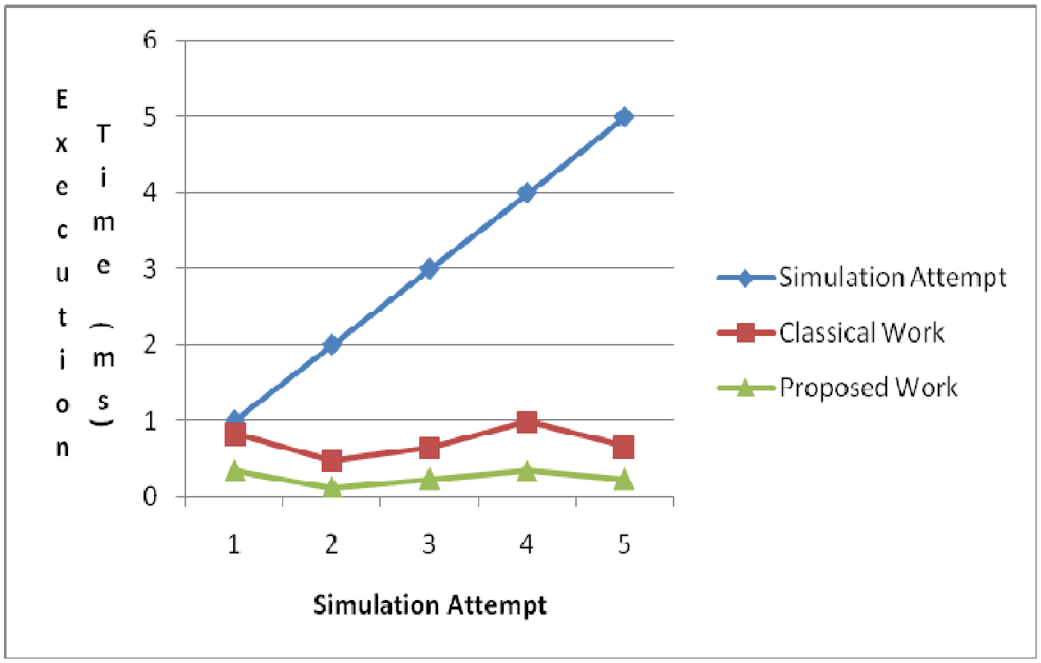


Figure 7.11 - Line Graph Analysis of the Existing and Proposed Approach

It is evident from the graphical results that the execution time and related parameter in the proposed research approach that is very less when compared to the existing *algorithmic approach. The execution time in the Existing work is taking higher units as compared to the proposed work.*

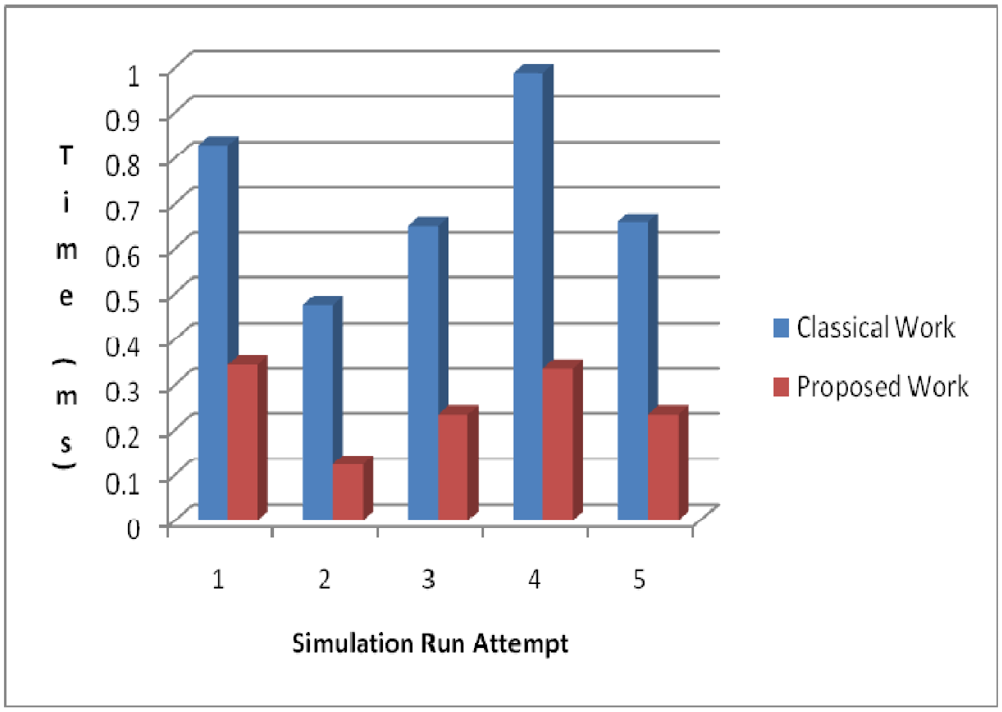


Figure 7.12 – Bar Graph Analysis of the Existing and Proposed Approach

It is clear from the graphical results that the time factor in the proposed research approach that is very less when compared to the existing algorithmic approach. The execution time in the Existing work is taking higher units as compared to the proposed work.

Simulation Attempt	Existing Work (USD)	Proposed Work(USD)
1	89	70
2	78	60
3	67	59
4	76	40
5	49	20

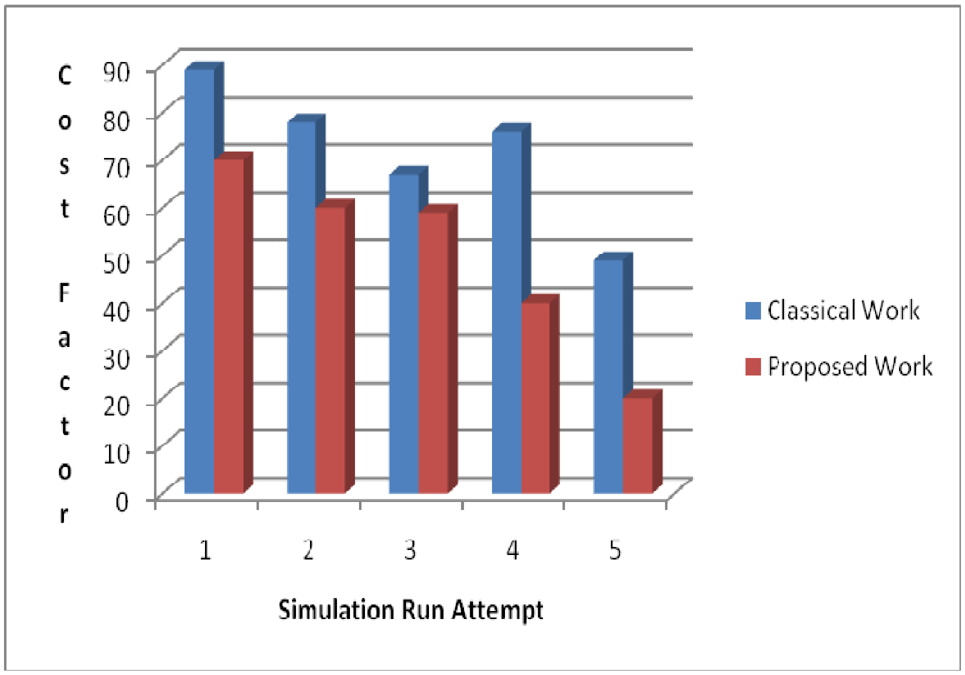


Figure 7.13 – Cost Factor Graph Analysis of the Existing and Proposed Approach

Existing / Existing Approach is not having effectiveness and efficiency as compared to GA based approach. The Existing work is taken as the implementation without integration of metaheuristic based simulation.

Data Centers - 10

Cloudlets – 20 in each module

Tasks executed without GA and then with the integration of GA to evaluate the efficiency and related cost factor.

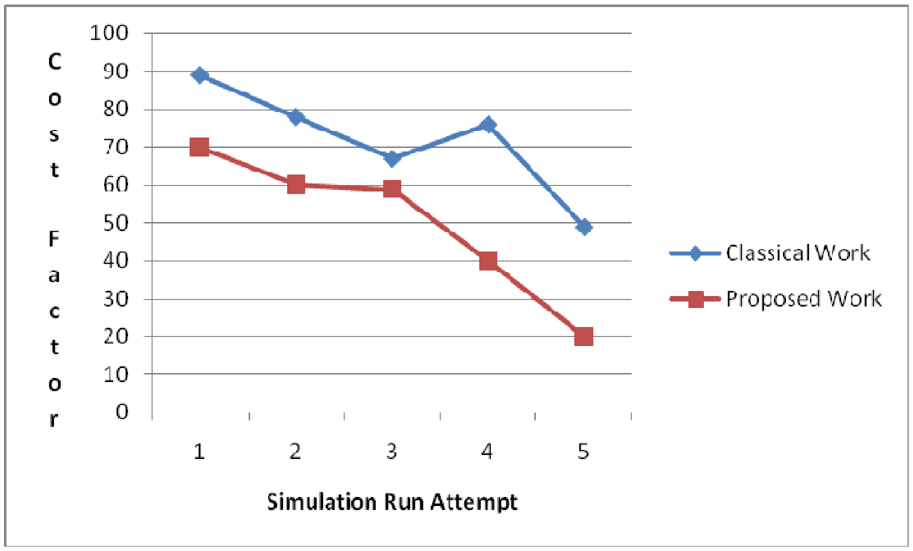


Figure 7.14 – Cost Factor Line Graph Analysis of the Approaches

It is evident from the graphical results that the cost factor in the proposed research approach that is very less when compared to the existing algorithmic approach. The execution time in the Existing work is taking higher units as compared to the proposed work.

Data Centers - 5

Cloudlets - 40 in each module

Tasks executed without GA and then with the integration of GA to evaluate the efficiency and related cost factor.

Existing and Improved Approach

Existing Base Work	Proposed Approach
50	70

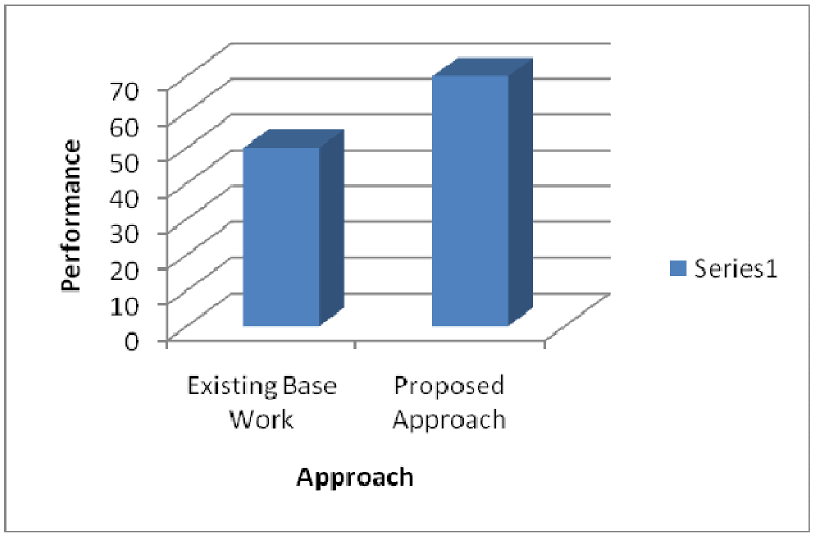


Figure 7.15 – Existing and Proposed approach

The graphical results depict that the performance in the proposed research approach that is very effective and better when compared to the existing algorithmic approach.

Data Centers - 20

Cloudlets – 50 in each module

Tasks executed without GA and then with the integration of GA to evaluate the efficiency and related cost factor.

Existing and Improved Approach

Existing Base Work	Proposed Approach
90	60

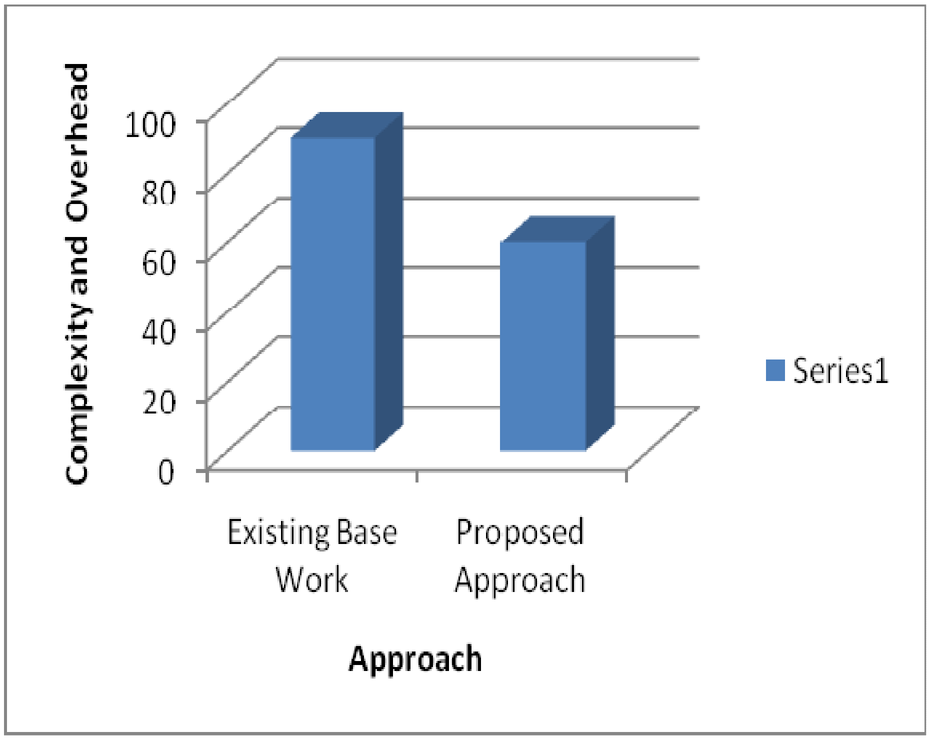


Figure 7.16 – Comparison of Existing and Proposed approach

Data Centers - 2

Cloudlets – 20 in each module

Tasks executed without GA and then with the integration of GA to evaluate the efficiency and related cost factor.

Difference between Existing and Improved Approach

Existing Base Work (Overall Effectiveness)	Proposed Approach (Overall Effectiveness)
49	59
50	87
68	88
49	68

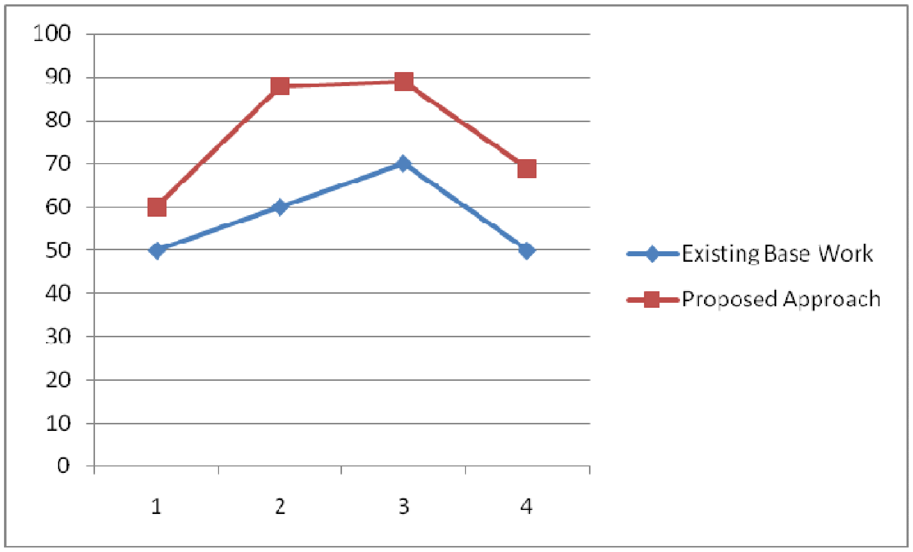


Figure 7.17 – Effective Comparison of Existing and Proposed Algorithm

Tabular Comparison of the Results Obtained (ms)

	W	T	E	P	C	CF
FCFS	1193	51	74	83	689	33
LJF	12	530	18	38	18	290
EDSRTF	7	18	69	86	688	18
GA	6	15	93	92	650	16

It is evident from the simulation results and Table 1 that the cumulative result based on all the parameters are effective and better in the proposed approach name GA.

W – Waiting Time

The amount of time a process has been waiting in the ready queue in the process of execution.

T – Turnaround Time

Amount of the time that is taken to complete a specific process.

E – Efficiency

The numbers of processes which completes its execution or processor / time.

P – Performance

Performance (P) is directly associated with the degree of Efficiency (e)

C – Complexity

Complexity of a structured program is defined with reference to the control flow graph of the program, a directed graph containing the basic blocks of the program, with an edge between two basic blocks if control may pass from the first to the second.

The complexity C is then defined as

$$C = E - N + 2P,$$

where

E = the number of edges of the graph.

N = the number of nodes of the graph.

P = the number of connected components.

$$C = E - N + P.$$

CF – Cost Factor

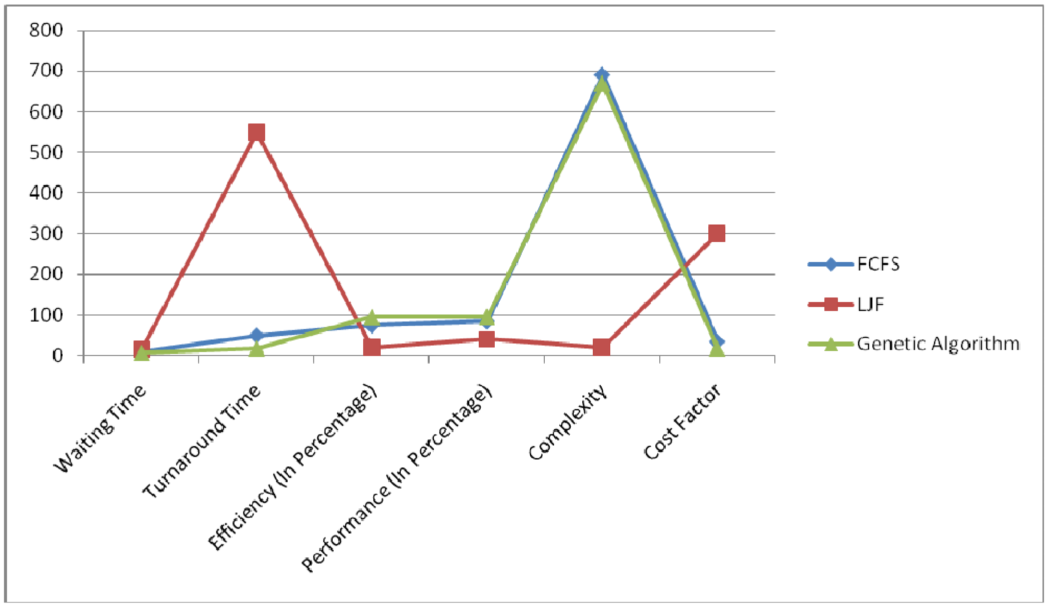
r => n * (1/t) * rnd

n -> Length of the Input

t -> Execution Time

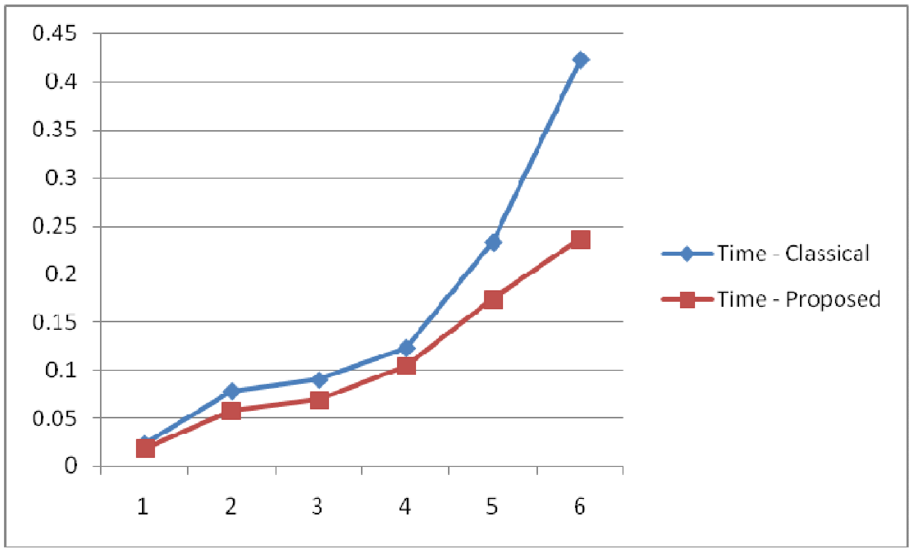
rnd -> Random Fuzzy Random

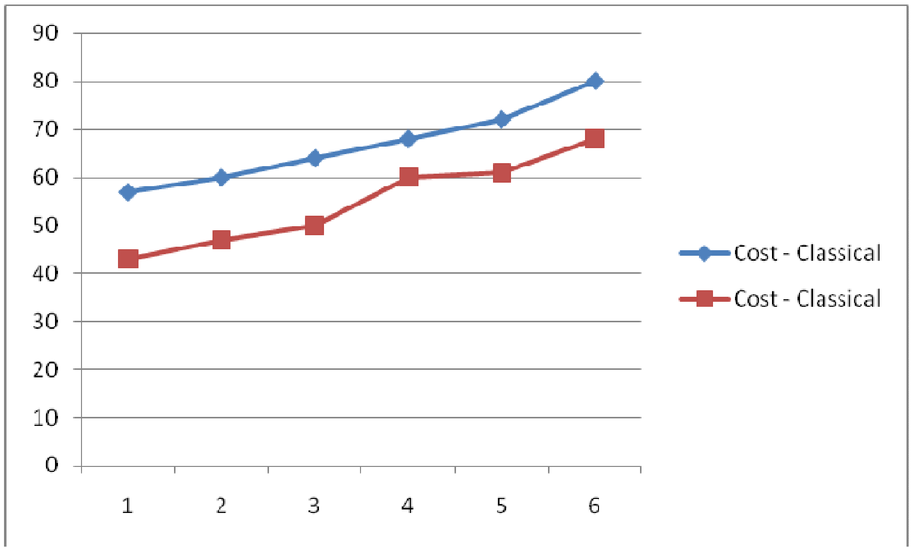
c = 1/(r * (p + e)) * 100

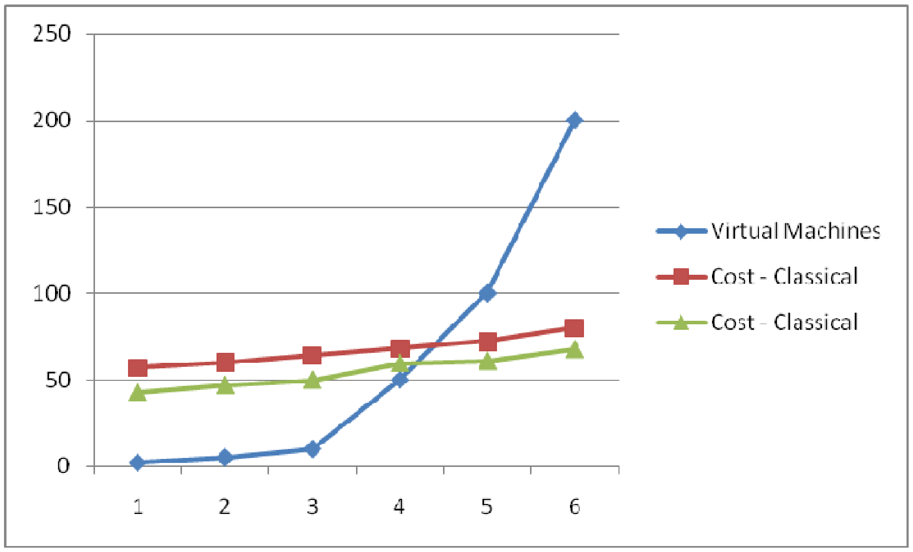


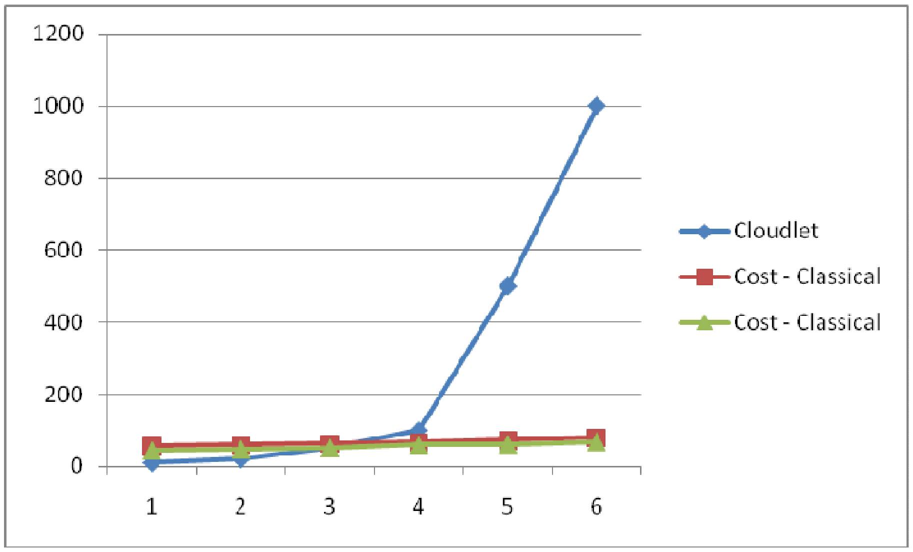
Cloudlet	Virtual Machines	Time – Existing (ms)	Time – Proposed(ms)	Cost – Existing (IN	Cost – Proposed (IN USD)	Percentage Enhanced - Time(ms)	Percentage Enhanced – Cost

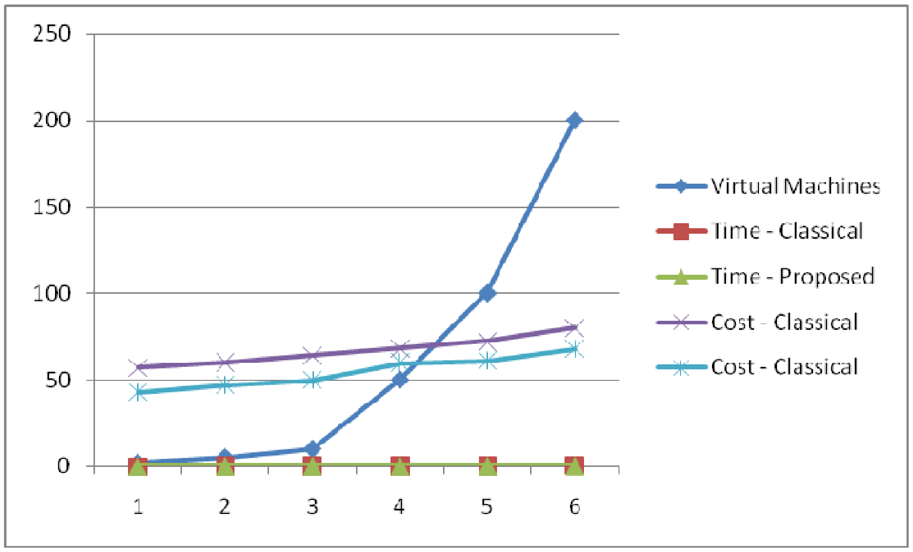
				USD)			**(IN USD)**
10	2	0.02323	0.01746	57	43	14.19513394	14
20	5	0.07833	0.05759	60	47	15.25727991	12.14953
50	10	0.0902	0.0685	64	50	13.67431427	12.2807
100	50	0.12333	0.10384	68	60	8.581497751	6.25
500	100	0.23328	0.17374	72	61	14.62805144	8.270677
1000	200	0.42323	0.2366	80	68	28.28513067	8.108108






250
200
150
100
50
0
1
2
3
4
5
6
Virtual Machines
Cost - Classical
Cost - Classical


1200
1000
800
600
400
200
0
1
2
3
4
5
6
Cloudlet
Cost - Classical
Cost - Classical



It is clear from all of the above graphical results that the existing approach is not effective and in contrast the proposed approach is efficient in all the parameters.

7. 2 SECURITY ENABLED VIRTUALIZATION IN THE PROPOSED APPROACH

Following are the results from the implementation of algorithm in CloudSim -

```
MD5 Hash Digest(in Hex. format):: 1e962b1bb2ec1218bc22a2cdc27ad1a8
Hybrid Approach (SHA+MD5) Hash Hex format :
2626d68c4ee76f4028179065479c497d116eed64a4d7f0e3e81547155a7608bb
Hybrid Approach Based (SHA+MD5) Security Key Transmitted =>
luwstkjvxap1??°°?¯?°?¢°
Hybrid Approach Secured Mode Transmission (SHA+MD5) from Broker with Key =>
luwstkjvxap1??°°?¯?°?¢°
---------------------------------------------------------
My Cloud Simulation with Security -> luwstkjvxap1??°°?¯?°?¢°
---------------------------------------------------------
Starting Simulation...
Cloud sim 3.0...
Initialising...
Creating new broker DC1-Broker
0.0 Creating new user base task1
Starting GridSim version 4.2
Entities started.
Starting broker 6 name=DC1-Broker
Starting user base 7 task1
Starting internet 9
5.0: DC1-Broker: Cloud Resource List received with 1 resource(s)
5.0: DC1-Broker: Trying to Create VM #0
5.0: DC1-Broker: Trying to Create VM #1
5.0: DC1-Broker: Trying to Create VM #2
```

```
5.0: DC1-Broker: Trying to Create VM #3
5.0: DC1-Broker: Trying to Create VM #4
Gathering simulation data.
task1 finalizing. Messages sent:631, Received:631
task1 requests sent=6058 , received=6058
Got response for 700630 but it seems to be completed.
DC1-Broker finalizing, submitted cloudlets=631 processing cloudlets=0
,allRequestsProcessed=6058
Simulation completed.
************ Vm allocations in DC1
0->254
1->254
2->254
3->253
4->253
*****Datacenter: DC1*****
User id            Debt
6            5128
**********************************
Simulation finished at 3650500.0
Execution Time 65 MillSeconds
5.684117979881531
 |
5.342058989940766
```

7.3 EXISTING WORK – SIMULATION SCENARIO WITH STATIC KEY

```
Atomicity Based Approach for Static Security Key Transmitted =>
^(*@^$*^$&*@(#
Atomicity Based Approach for Static Security Key Transmitted from Broker =>
(&)(&@#&97219733
Randomly Selected Number of Processors int in a range  :  19
RAM => 1024
Bandwidth =>1000
Processors =>19
and
Virtual Machine Manager =>XenVM
Initialized
Starting CloudSim version 3.0
CloudDataCenter-1 is starting...
CloudDataCenter-2 is starting...
Broker is starting...
Entities started.
```

```
0.0 :  Broker :  Cloud Resource List received with 2 resource(s)
0.0 :  Broker :  Trying to Create VM #0 in CloudDataCenter-1
0.0 :  Broker :  Trying to Create VM #1 in CloudDataCenter-1
[VmScheduler.vmCreate] Allocation of VM #1 to Host #0 failed by MIPS
0.1 :  Broker :  VM #0 has been created in Datacenter #2, Host #0
0.1 :  Broker :  Creation of VM #1 failed in Datacenter #2
0.1 :  Broker :  Trying to Create VM #1 in CloudDataCenter-2
0.2 :  Broker :  VM #1 has been created in Datacenter #3, Host #0
0.2 :  Broker :  Sending cloudlet 0 to VM #0
0.2 :  Broker :  Sending cloudlet 1 to VM #1
0.2 :  Broker :  Sending cloudlet 2 to VM #0
160.2 :  Broker :  Cloudlet 1 received
320.2 :  Broker :  Cloudlet 0 received
320.2 :  Broker :  Cloudlet 2 received
320.2 :  Broker :  All Cloudlets executed. Finishing...
320.2 :  Broker :  Destroying VM #0
320.2 :  Broker :  Destroying VM #1
Broker is shutting down...
Simulation :  No more future events
CloudInformationService :  Notify all CloudSim entities for shutting down.
CloudDataCenter-1 is shutting down...
CloudDataCenter-2 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

================= OUTPUT =================
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
==========================================
   1      SUCCESS      3          1       160       0.2       160.2
   0      SUCCESS      2          0       320       0.2       320.2
   2      SUCCESS      2          0       320       0.2       320.2
```

```
-----------------------------------------------------------------
Cloud Simulation Finish
Simulation Scenario Finish with MATCHING (AUTHENTICATION SUCCESSFUL)  of the
Keys
-----------------------------------------------------------------
Simulation Scenario Execution Time in MillSeconds => 323
Security Parameter => 10.419725571323061
2014-06-30 23 : 53 : 30.297
```

7.4 PROPOSED WORK – SIMULATION SCENARIO WITH PROPOSED KEY

```
-----------------------------------------------------
Starting Cloud Simulation with Dynamic and Hybrid Secured Key
-----------------------------------------------------
Cloud Simulation Starts for Security in Data Centers
Initialising...
MultiLayeredSecurityDigest(in Hex. format) :  :  f7604f374fd73e4cdc121c2bd2c58629
Hybrid Approach (MULTILAYEREDENCYPTION) Hash Hex format  :
8f82897ef4c8018d6fdab5c5532383e1337eca1d6b5a8e650ce92fd8b7ae6229
Hybrid Approach Based (MULTILAYEREDENCYPTION) Security Key Transmitted =>
Hskgh*&(*@&#(@#
Hybrid Approach Secured Mode Transmission (MULTILAYEREDENCYPTION) from
Broker with Key => (&$@(@B@JK@J#
Randomly Selected Number of Processors int in a range  :  8
RAM =>512
Bandwidth =>1000
Processors =>8
and
Virtual Machine Manager =>Xen
Initialized
Starting CloudSim version 3.0
CloudDataCenter-1 is starting...
CloudDataCenter-2 is starting...
```

```
Broker is starting...
Entities started.
0.0 :  Broker :  Cloud Resource List received with 2 resource(s)
0.0 :  Broker :  Trying to Create VM #0 in CloudDataCenter-1
0.0 :  Broker :  Trying to Create VM #1 in CloudDataCenter-1
[VmScheduler.vmCreate] Allocation of VM #1 to Host #0 failed by MIPS
0.1 :  Broker :  VM #0 has been created in Datacenter #2, Host #0
0.1 :  Broker :  Creation of VM #1 failed in Datacenter #2
0.1 :  Broker :  Trying to Create VM #1 in CloudDataCenter-2
0.2 :  Broker :  VM #1 has been created in Datacenter #3, Host #0
0.2 :  Broker :  Sending cloudlet 0 to VM #0
0.2 :  Broker :  Sending cloudlet 1 to VM #1
0.2 :  Broker :  Sending cloudlet 2 to VM #0
160.2 :  Broker :  Cloudlet 1 received
320.2 :  Broker :  Cloudlet 0 received
320.2 :  Broker :  Cloudlet 2 received
320.2 :  Broker :  All Cloudlets executed. Finishing...
320.2 :  Broker :  Destroying VM #0
320.2 :  Broker :  Destroying VM #1
Broker is shutting down...
Simulation :  No more future events
CloudInformationService :  Notify all CloudSim entities for shutting down.
CloudDataCenter-1 is shutting down...
CloudDataCenter-2 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

================= OUTPUT
============================================================
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
```

```
============================================================ 1
SUCCESS       3          1       160      0.2      160.2
0       SUCCESS       2          0       320      0.2      320.2
2       SUCCESS       2          0       320      0.2      320.2
------------------------------------------------------------------
Cloud Simulation Finish
Simulation Scenario Finish with MATCHING (AUTHENTICATION SUCCESSFUL) of the
Keys
------------------------------------------------------------------
Simulation Scenario Execution Time in MillSeconds => 141
Security Parameter => 30.73876581994184
```

7.5 PROPOSED WORK – SIMULATION SCENARIO WITH PROPOSED KEY

```
--------------------------------------------------------------------------------
Starting Cloud Simulation with Dynamic and Hybrid Secured Key
--------------------------------------------------------------------------------
Cloud Simulation Starts for Security in Data Centers
Initialising...
MultiLayeredSecurityDigest(in Hex. format) :  :  %#*Z*#*%#FFFFZJEJFJJ
Hybrid Approach (MULTILAYEREDENCYPTION) Hash Hex format  :  i7987907!&473
Hybrid Approach Based (MULTILAYEREDENCYPTION) Security Key Transmitted =>
okvwnpdkgjc4©³²¬¯¯¦¦²«±
Hybrid Approach Secured Mode Transmission (MULTILAYEREDENCYPTION) from
Broker with Key => okvwnpdkgjc4©³²¬¯¯¦¦²«±
Randomly Selected Number of Processors int in a range  :  18
RAM =>512
Bandwidth =>1000
Processors =>18
and
Virtual Machine Manager =>Xen
Initialized
Security Key Not Matched... Operation Terminated
```

```
Simulation Scenario Finish with NON-MATCHING (AUTHENTICATION FAILED) of the
Keys
Operation Terminated
Simulation Scenario Execution Time in MillSeconds => 118
Security Parameter => 30.47201430637181
```

EXISTING WORK – SIMULATION SCENARIO WITH STATIC KEY

```
Atomicity Based Approach for Static Security Key Transmitted =>
*!(!&^%$#@$^)(*
Atomicity Based Approach for Static Security Key Transmitted from Broker =>
pwsthvgkayc4???¡¥ª¦²?¡®
Randomly Selected Number of Processors int in a range  :  9
RAM =>512
Bandwidth =>1000
Processors =>9
and
Virtual Machine Manager =>Xen
Initialized
Security Key Not Matched... Operation Terminated
Simulation Scenario Finish with NON-MATCHING (AUTHENTICATION FAILED) of the
Keys
Operation Terminated
Simulation Scenario Execution Time in MillSeconds => 239
Security Parameter => 10.859743354629565
```

EXISTING APPROACH – SIMULATION SCENARIO WITH STATIC KEY

```
Atomicity Based Approach for Static Security Key Transmitted =>
*!(!&^%$#@$^)(*
Atomicity Based Approach for Static Security Key Transmitted from Broker =>
jdtpptwpqqu4¤?«?¢«¬¥§?¦
```

```
Randomly Selected Number of Processors in a range  :  17
RAM =>512
Bandwidth =>1000
Processors =>17
and

Virtual Machine Manager =>Xen
Initialized
Starting CloudSim version 3.0
CloudDataCenter-1 is starting...
CloudDataCenter-2 is starting...
Broker is starting...

Entities started.
0.0 :  Broker :  Cloud Resource List received with 2 resource(s)
0.0 :  Broker :  Trying to Create VM #0 in CloudDataCenter-1
0.0 :  Broker :  Trying to Create VM #1 in CloudDataCenter-1
[VmScheduler.vmCreate] Allocation of VM #1 to Host #0 failed by MIPS
0.1 :  Broker :  VM #0 has been created in Datacenter #2, Host #0
0.1 :  Broker :  Creation of VM #1 failed in Datacenter #2
0.1 :  Broker :  Trying to Create VM #1 in CloudDataCenter-2
0.2 :  Broker :  VM #1 has been created in Datacenter #3, Host #0
0.2 :  Broker :  Sending cloudlet 0 to VM #0
0.2 :  Broker :  Sending cloudlet 1 to VM #1
0.2 :  Broker :  Sending cloudlet 2 to VM #0
160.2 :  Broker :  Cloudlet 1 received
320.2 :  Broker :  Cloudlet 0 received
320.2 :  Broker :  Cloudlet 2 received
320.2 :  Broker :  All Cloudlets executed. Finishing...
320.2 :  Broker :  Destroying VM #0
320.2 :  Broker :  Destroying VM #1
```

```
Broker is shutting down...
Simulation :  No more future events
CloudInformationService :  Notify all CloudSim entities for shutting down.
CloudDataCenter-1 is shutting down...
CloudDataCenter-2 is shutting down...

Broker is shutting down...
Simulation completed.
Simulation completed.

================= OUTPUT ==============================
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
====================================================    1
SUCCESS       3          1       160       0.2       160.2
   0       SUCCESS       2          0       320       0.2       320.2
   2       SUCCESS       2          0       320       0.2       320.2
----------------------------------------------------------------------

Cloud Simulation Finish

Simulation Scenario Finish with MATCHING (AUTHENTICATION SUCCESSFUL)  of the
Keys

----------------------------------------------------------------------
Simulation Scenario Execution Time in MillSeconds => 280
Security Parameter => 10.858184353150472
```

7.6 COMPARISON OF EXECUTION TIME IN EXISTING AND PROPOSED APPROACH

Comparison Table of Existing and Proposed Approach in terms of Execution Time

Attempt	Existing	Proposed

1	1316	2724
2	1242	2914
3	1113	2367
4	1192	2300
5	1281	2971
6	1804	2359
7	1483	2613
8	1286	2096
9	1855	2700
10	1090	2049
11	1415	2939
12	1376	2588
13	1005	2151
14	1039	2485
15	1340	2797
16	1828	2755
17	1680	2132

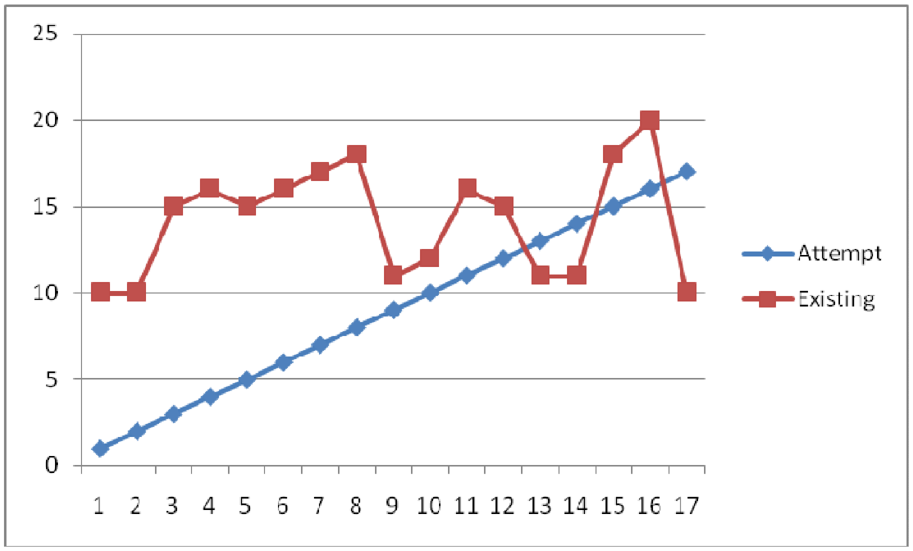


Figure 7.18 – Comparison between Existing and Proposed Approach

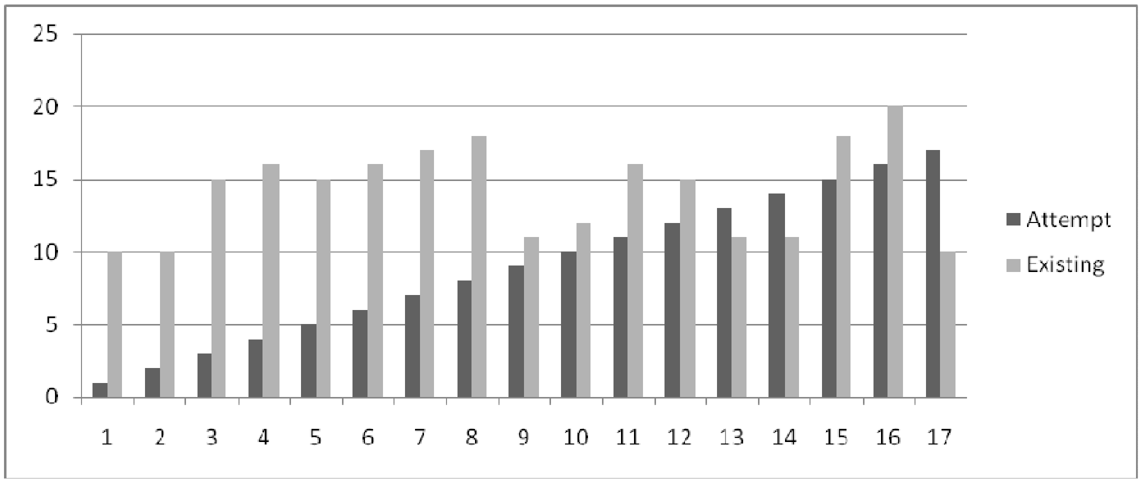


Figure 7.19 – Bar Comparison between Existing and Proposed Approach

COMPARISON TABLE FOR SECURITY PARAMETER

Comparison Table of Existing and Proposed Approach in terms of Security

Attempt	Existing	Proposed
1	10	37
2	10	32
3	15	32
4	16	38
5	15	33
6	16	39
7	17	38
8	18	33
9	11	33
10	12	40
11	16	39
12	15	31
13	11	40
14	11	34
15	18	30
16	20	38
17	10	33

18	19	34
19	16	37
20	10	36

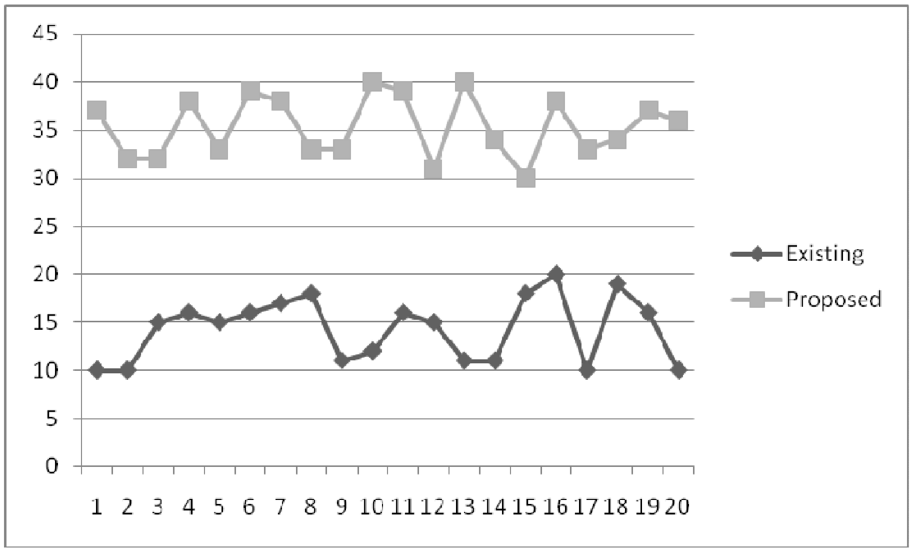


Figure 7.20 – Line Graph Comparison between Existing and Proposed Approach

KEYS GENERATED BY CLOUDSIM IN TERMS FOR CLOUD SECURITY IN EXISTING APPROACH

Execution Time=> 13207 ExistingApproachKey : *!(!&^%$#@$^)(* TimeStamp :2014-06-27 16:17:36.526 Attempt Flag : 1

```
Execution Time=> 4678     ExistingApproachKey : $(*&^@)(&@(@&(*
Execution Time=> 3915     ExistingApproachKey : $(*&^@)(&@(@&)(*
Execution Time=> 6031     ExistingApproachKey : $(*&^@)(&@(@& (*
Execution Time=> 12631    ExistingApproachKey $(*&^@)(&@(@& (*
Execution Time=> 6658     ExistingApproachKey *!(!&^%$#@$^)(*
Execution Time=> 8561     ExistingApproachKey *!(!&^%$#@$^)(*
Execution Time=> 5744     ExistingApproachKey : *!(!&^%$#@$^)
Execution Time=> 4348     ExistingApproachKey : *!(!&^%$#@$^)
```

PROPOSED APPROACH WITH DYNAMIC HASH ALGORITHMS

```
Execution Time=> 8038     Proposed (Dynamic-Multilayer-Key) Key qucjlkpmfnr4¨? §-?££-ª¨
Execution Time=> 3723     Proposed (Dynamic-Multilayer-Key) Key : wwrsgpavsrn4?ª¡±«¤?©®¬¢
Execution Time=> 4035     Proposed (Dynamic-Multilayer-Key) Key : dlrvakssuni4?¢?©?¨²?±¦²
Execution Time=> 5649     Proposed (Dynamic-Multilayer-Key) Key : dneumimwqga4¨°¡?¡¢?¦¤±ª
Execution Time=> 6802     Proposed (Dynamic-Multilayer-Key) Key fbebxupsjhs4²®?®£¯ ?£¯¥?
Execution Time=> 3768     Proposed (Dynamic-Multilayer-Key) Key : kwinksoausi4?????¨?¨²?®
Execution Time=> 4312     Proposed (Dynamic-Multilayer-Key) Key : ninkqqvxmdd4?²¯ ?¯ ³-¡?³?
Execution Time=> 31888    Proposed (Dynamic-Multilayer-Key)    Key dtoodwfoltm4ª?±£-¥¥§©¯ ¢
Execution Time=> 10408    Proposed (Dynamic-Multilayer-Key) Key : rhtpjoardll4©¥ ?¡-®?©?©
Execution Time=> 6947     Proposed (Dynamic-Multilayer-Key) Key puavxilpgdw4?²²¦«ª®£?¬±
Execution Time=> 7267     Proposed (Dynamic-Multilayer-Key) Key : dmqnegjlppm4?®?¯ ?°?®§¢®
Execution Time=> 7964     Proposed (Dynamic-Multilayer-Key) Key : mimdoxtjwwp4°?£?£²¦-?¦¬
Execution Time=> 6925     Proposed (Dynamic-Multilayer-Key) Key : vqnsymhapoo4¡£°¨?¯ ¡¥±ªª
Execution Time=> 3860     Proposed (Dynamic-Multilayer-Key) Key : ajdwcxfubba4«® ©§ ¥± ?-
Execution Time=> 5896     Proposed (Dynamic-Multilayer-Key) Key : nrsieyvocfm4±?-£³ ??°?®
Execution Time=> 4681     Proposed (Dynamic-Multilayer-Key) Key cqjyyptkwpq4±ª®¨¢?²£?®¬
Execution Time=> 5904     Proposed (Dynamic-Multilayer-Key) Key iyoqhefjwcn4??¯¨¡®??²?¬
Execution Time=> 5767     Proposed (Dynamic-Multilayer-Key) Key : ygcxsbyybpr4¢- ª¢£?¡®-£
Execution Time=> 15417    Proposed (Dynamic-Multilayer-Key) Key : xcyvqjijbyp4??¡® ª?¬-?¡
Execution Time=> 24093    Proposed (Dynamic-Multilayer-Key) Key : rojoeonhivv4©¥¨¥??¥?-²®
Execution Time=> 28061    Proposed (Dynamic-Multilayer-Key) Key : anfcjunqhct4§¯¨?¡²¡®°²²
```

CHAPTER 8
CONCLUSION AND FUTURE WORK

The existing problems in the storage virtualization and related dimensions have been removed using genetic algorithm and optimal results has been obtained. The further work in this area can be improved by using the other metaheuristics including ant colony optimization, simulated annealing, and honeybee algorithm. These algorithms are very prominent in terms of solving the combinatorial optimization problems. The Existing storage virtualization, scheduling and security algorithms can be passed with assorted swarm intelligence techniques to get the multiple results and from which the optimal result be obtained.

For future scope of the work, following techniques can be used in hybrid approach to better and efficient results -

- Particle Swarm Optimization
- HoneyBee Algorithm
- Simulated Annealing
- Ant Colony Optimization
- Firefly Algorithm
- Bat Algorithm

REFERENCES

[1] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., ... & Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50-58.

[2] Buyya, R., Yeo, C. S., & Venugopal, S. (2008, September). Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on (pp. 5-13). Ieee.

[3] Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications, 1(1), 7-18.

[4] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing.

[5] Hamdaqa, Mohammad, Tassos Livogiannis, and Ladan Tahvildari. "A Reference Model for Developing Cloud Applications." In CLOSER, pp. 98-103. 2011.

[6] Formu, J. Cloud Cube Model: Selecting Cloud Formations for Secure Collaboration. April, 2009.

[7] Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. Future Generation computer systems, 28(3), 583-592.

[8] Singh, A., Korupolu, M., & Mohapatra, D. (2008, November). Server-storage virtualization: integration and load balancing in data centers. In Proceedings of the 2008 ACM/IEEE conference on Supercomputing (p. 53). IEEE Press.

[9] Huang, L., Peng, G., & Chiueh, T. C. (2004, June). Multi-dimensional storage virtualization. In ACM SIGMETRICS Performance Evaluation Review (Vol. 32, No. 1, pp. 14-24). ACM.

[10] Brinkmann, A., Heidebuer, M., Meyer auf der Heide, F., Rückert, U., Salzwedel, K., & Vodisek, M. (2004). V: Drive–Costs and Benefits of an Out-of-Band Storage Virtualization System. In Proceedings of the 12th NASA Goddard, 21st IEEE Conference on Mass Storage Systems and Technologies (MSST).

[11] Virtualize Your IT Infrastructure. (2012). http://www.vmware.com/ virtualization/

[12] Connor, D. (2004). Server virtualization is on the rise. Network World, 21(49), 25-25, 28.

[13] Spiegel, E. (2006). Real world virtualization: Realizing the business benefits of application and server virtualization. Computer Technology Review, 26(2), 8-8, 26.

[14] Hassell, J. (2007). Server virtualization: GETTING STARTED. Computerworld, 41(22), 31-31.

[15] McAllister, N. (2007). Server virtualization. InfoWorld, 29(7), 20-22.

[16] Bele, R., & Desai, C. (2012). Review on virtualization: In the light of storage and server virtualization technology. Journal of Information and Operations Management, 3(1), 245-249.

[17] Schultz, B. (2009). The changing face of virtualization. Network World, 26(26), 28-30.

[18] Norall, S. (2007). Storage virtualization. InfoWorld, 29(7), 24-27.

[19] Peggy, B. K. (2007). Virtualization reality -- by optimizing computing resources, virtualization can lower infrastructure operating costs while improving business agility and speed to market. Insurance & Technology, 32(8), 42-42.

[20] Kontzer, T. (2010). Virtualization as a tool for agile it. CIO Insight, (113), 28-31.

[21] Kovar, J. F. (2008). Virtualization's dream team -- our exclusive research reveals just how hot virtualization is. no need to tell the folks at VMware. CRN, (1257), 22-24.

[22] Yoshida, H. (2008). Reinventing storage virtualization. Network World, 25(39), 25-25.

[23] Dubie, D. (2009). Weighing desktop virtualization. Network World, 26(22), 16-16, 32.

[24] Kennedy, R. C. (2007). Application and desktop virtualization. InfoWorld, 29(7), 28-28,30,32.

[25] Hsieh, C. (2008). Strategies for successfully implementing a virtualization project: A case with VMware. Communications of the IIMA, 8(3), 1-I.

[26] Violino, B. (2009). Virtualization's new frontier. CIO Insight, (101), 28-30, 32-33.