P.Nagamani Journal of Engineering Research and Application www.ijera.com
ISSN : 2248-9622, Vol. 8, Issue5 (Part -V) May 2018, pp 57-76

RESEARCH ARTICLE OPEN ACCESS

Security Design Framework For Data Management And Distribution For Middleware In Grid Extended Cloud Computing

P.Nagamani*, Dr.P. Suresh Varma**Dr.M. Upendra Kumar***
**Research Scholar, CSE, Rayalaseema University Kurnool, A.P. India.*
***Professor, CSE, Adikavi Nannaya University, A.P, India.*
****Professor, CSE, MVSR Engg. College, O.U. HYD, T.S. India.*
Corresponding auther: P.Nagamani

ABSTRACT
This research paper is a novel and innovative idea of providing a security framework for Data Management and Distribution for Middleware technology for Grid extended Clouds, with a validation on a case study. The proposed research model is an Object Oriented Pattern and Framework using UML Security Design Model for Authentication and Authorization for Data Management and Distribution in Grid extended clouds. Appropriate implementations are performed in case study to adequately validate the proposed Object oriented model. This model can be extended for High Performance Computing in Distributed Systems and Big Data.
Keywords–Grid Data Management, Cloud Computing, Security Design, Authorization, Authentication, Object Oriented Framework, Middleware

---

Date of Submission: 20-05-2018 Date of acceptance:05-06-2018

I. INTRODUCTON

Overview of Grid Computing:

The vision of grid computing is to enable computing to be delivered as a utility. This vision is most often presented with an analogy to electrical power grids, from which it derives the name "grid". So, grid computing was meant to be used by individual users who gain access to computing devices without knowing where the resource is located or what hardware it is running and so on. In this sense, it is pretty similar to cloud computing. However, just as electrical power girds can derive power from multiple power generators and deliver the power as needed by the consumer, the key emphasis of grid computing was to enable sharing of computing resources or forming a pool of shared resources that can then be delivered to users. So, most of the initial technological focus of grid commuting was limited to enabling shared use of resources with common protocols for access. Also, since the key takers of this fascinating vision were educational institutions, a particulars emphasis was given the handle heterogeneous infrastructure. Which was typical of a university datacenter? From a technical perspective, a software-only solution was proposed (Globus) and implemented on this heterogeneous infrastructure to enable use of the resources for higher computing needs. Once reasonably successful within universities, grid-computing faced a serious issue when it came to sharing resources across commercial institution. Establishing that and security models between infrastructure resources pooled from two different administrative domains became even more important. [1]

Three Fundamental Characteristics of a Grid:

The version of grid computing is to enable computing to be delivered as a utility. This vision is most often presented with an analogy to electrical power grids, from which it derives the name "grid". So, grid computing was meant to be used by individual users who gain access to computing devices without knowing where by individual the resource is located or what hardware it is running, and so on. In this sense, it is pretty similar to cloud computing. However, just as electrical power grids can derive power from multiple power generators and deliver the power as needed by the consumer, the key emphasis of grid computing was to enable sharing of computing resource or forming a pool of shared resources that can then be delivered to users. So, most of the initial technological focus of grid computing was limited to enabling shared use of resources with common protocols for access, Also, since the key takers of this fascinating vision were educational institutions, a particular emphasis was given to handle heterogeneous infrastructure, which was typical of a university datacenter. From a technical perspective, a software-only solution was proposed (Globus) and implemented on this heterogeneous infrastructure to enable use of these resources for higher computing needs. Once reasonably successful within universities, grid computing faced a serious issue when it came to

sharing resolves across commercial institutions.] Establishing trust and security models between infrastructure resources pooled from two different administrative domains became even more important.

Three Fundamental Characteristics of a GridIn 2002, Ian Foster from Argonne National Laboratories proposed a three-point checklist for determining whether a system is a grid or not. Ian Foster along with Steve Tucker in the popular article "Anatomy of Grid" defined grid computing as "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations."

So, the key concept emphasized was the ability to negotiate resource sharing agreements among a set of participating parties – where sharing did not really mean "exchange" but direct access to computing resources either in a collaborative resourced sharing or negotiated resource brokering strategies. Further, this sharing was highly controlled with resource providers and consumers grouped into virtual organizations primarily based on sharing conditions.

The following is the precise simple checklist that was proposed: A grid is a system that

- Co-ordinate resources that are not subject to centralized control
- Using standard, open, general purpose protocols and interfaces
- To deliver nontrivial quality of service

The first criterion sates that a grid should integrate computing resources from different control domains (say servers from commuter centers of different universities, each center having a different system administrator in each university). Technologically, this requirement addresses the issues of cross-domain security, policy management, and membership. Use of a common standard for authentication, authorization, resource discovery and resource access becomes a necessity in such cases and hence the second criterion. Finally, in an effort towards commercializing the usage of shared resources, it is important to support various quality-of-service parameters such as response time, throughput, availability or even co-allocation of resources to meet use demands.

A Closer Look at Grid Technologies:

First of all, grid computing defines a notion of a virtual organization to enable flexible, co-ordinate, secure resource sharing among participating entities. A Virtual organization (VO) is basically a dynamic collection of individuals' or institutions from multiple administrative domains. A VO forms a basic unit for enabling access to shared resources with specific resource-sharing policies applicable for users from a particular VO The key technical problem addressed by grid technologies is to enable resources sharing among mutually distrustful participants of a VO who may have varying degrees of prior relationship (perhaps none at all) and enable them to solve a common task.

An extensible and open Grid Architecture was defined by Ian Forstrn "The Anatomy of the Grid [36] in which protocols, services APs and SDKs are categorized according to their roles in enabling resource sharing. The Grid Fabric layer provides the resources to which shared access is mediated by grid protocols. These can be computational resources, storage systems. Catalogs, network resources or even a logical entity, such as s distributed file system, computer cluster, or distributed computer pool,. A well-known toolkit for the fabric layer is they Globus Toolkit that provides local resource specific operations on existing computing elements [37]. The connectivity layer includes the core protocols for communication and authentication for inter-node communication. The key aspects of these protocols include single sign on, delegation, use-based trust relationships and integrating with local security solutions. One important protocol whose reference implementation is available in Globus is the public key based GSI protocol (Grid Security Infrastructure), which extends TLS (Transport Layer Security) to address these issues. The resource layer includes APIs and SDKs for secure negotiation, monitoring, control, accounting, and payment for operations on a single shared resource. An example protocol at this layer is the GRAM (Grid Resource Access and Management) protocol used for allocating, monitoring and control of computational resources; and the GRIP (Grid Resource Information Protocol) and GridFTP (File Transfer Protocol), which are extensions of LDAP and FTP protocols. The Collective Layer implements a variety of sharing behaviors with directory services, brokering services, programming systems community accounting and authorization services and even collaborative services. One such services is the GIIS (Grid Information Index Servers) that supports arbitrary views on resource subset, which can be used with LDAP and the DUROC library that supports resource co-allocation. [38].

Current implementation of Open Grid architecture follows a Web Services-based interface enabling interoperability between different implementations of the protocols. Since web services by definition are stateless, the Grid community (Globus alliance) introduced a set of enhanced specifications called Web services Resource Framework (WSRF) that web services could implement to become state full. Open Grid Services Architecture now defines a service-oriented grid computing environment, which not only provides standardized interfaces, but also removes

the need for layering in the architecture and defines a concept of virtual domains, allowing dynamic grouping of resources as well.

A reference implementation of these protocols is available in popular open source software toolkit called Globus toolkit (GT), which was developed by the Globus alliance, a community of organizations and individuals developing fundamental technologies behind the grid [39-41]. The nice thing about this software is that it enables existing resources to easily join a grid pool by enabling the required protocols locally. To get started on setting up a grid, one just needs to download and install GT on any of the supported platforms. To create a resource pool, it is a good idea to install a resource scheduler such as the Condor cluster scheduler and configure that s a grid gateway for resource allocating. After some initial security configurations (obtaining signed certificates and setting up access rights), the grid can be up and running.

Comparing Grid and Cloud:

From the earlier description of grid computing. It can be seen that it has many similarities with cloud computing. However, there are differences as well, notably the fact that grid computing emphasizes the pooling of resources from multiple organizations, and that it mostly targets high-performance computing (HPC) applications. This section compares the two technologies in more detail using different parameters. Readers are referred to studies made in 2008 [42, 43] for a detailed comparison of grid and cloud computing from a practical implementation perspective.

Similarities between Grid and Cloud:

The key similarity between cloud computing and grid computing is the intent of providing resources that can scale and go beyond what a use personally owns. In grid computing, the scalability is provided by increasing the utilization of resources and is achieved by load balancing across shared resources. On the other hand scalability in a cloud service is achieved by using sophisticated auto-re-provisioning techniques or simply by provisioning more than what these asked for (always catering to peak loads).

The need for multitasking and multi-tenancy is also common between the two-Multiple users can simultaneously access the same resources and run multiple instances of applications. However, since cloud computing typically involves a more commercial agreement between the vendor and the user, the system has more rigorous need for multi-tenancy at every aspect of the stack-infrastructures, platforms well as application.

Since both the forms of computing require use of resources form someone else either the cloud vendor or collaborator in the grid case, strict service-level agreements need to be in place to ensure fair play, especially when the resource usage comes with certain commercial agreements. Similarly, many grid systems provide support for application failover (Condor) and this is particularly useful for long running HPC applications to restart from the nearest failure point. Fault tolerance of applications on a cloud system risk in fact, critical and the vendor needs to ensure service availability through appropriate failover mechanisms.

Difference between Grid and Cloud:

Given the detailed discussion of cloud computing in the earlier models and the short introduction to grid commuting, if will be clear that there are differences between the computing models. A grid basically links disparate resources from multiple organizations to form one large infrastructure pad. Grid computing allocates compute and storage resources to a use from a shared point of assets that can even have a contribution from the user's own organization. The key focus is in harnessing unused resources and typically these resources are heterogeneous in nature. On the other hand, cloud infrastructure will usually consist of homogeneous resources and is provided by a single vendor to a consumer or user (different from the vendor).

Advance Reservation:

In fact, there were many advance reservation algorithms (Grid-ARS) and APIs (Grid Engine) [44] proposed around 2005, to enable optimal resource utilization in grid systems. On the contrary, no reservation is needed in a cloud infrastructure. On-demand resource provisioning is one of the key benefits of cloud computing. The resources are supposed to magically expand when the demand increases. Some of the techniques and APIs provided to enable this elasticity in computing have been described earlier, and massive scale up of resource on demand is a key aspect of cloud computing, which removes the need for advance resource reservations.

Another aspect that is different between the two models is the ownership of resources. Since resources from multiple organizations are pooled, the machines in a grid pool will typically come from different administrative domains. So, protocols to manage authenticated access in such a virtual organization become important. Resources on cloud, however, are owned by a single cloud vendor and any joint partnerships are handled at a business level and no technology components for the same are used.

Further, in a cloud environment, consumers use what they need and pay only for what they used (even in a private cloud, different departments in a business may pay for their resource usage)-while

payment is not an aspect studied in the grid context. Users may also pay implicitly by contributing their resources to a shared pool for other's use. So, while fine-grained usage monitoring becomes important on a cloud, it is not of much value in a grid system. There are also differences in the target user segment that the two computing models address. The target segment for cloud computing is established industry, academia and also startups or new ventures. And they are hosted by commercial companies like Amazon and HP, who charge users for what they use. On the other hand, the target populations for a grid are primarily researchers and technology collaborators (groups of institutions) that are interested in sharing their individually owned resources among each other.

Grid computing is a software-only solution, with tools (Globus toolkit deployed to enable grid protocols over existing systems. A cloud-based solution, on the other hand, involves technologies at multiple layers of the stack, leading to different clod models (IaaS, PaaS and SaaS). Also, grid applications are parallel, distributed, message-passing applications that either execute certain modules on specialized computing resources located in a different geography, or execute a data parallel application loosely coupled and distributed on a number of similar compute and storage resources. Grids are therefore suited for HPC applications for large-scale computation where large data sets are crouched by parallelizable compute intensive applications. A cloud application, on the other hand, need not be a distributed application. It needs to be architected in a way to scale based on demand. So apart from using distributed machines, it can also use a scale-out technique on clusters or parallel threads on multiple compute nodes with s shared memory, for example. Cloud computing is also used to host web services that tend to be a long-serving daemon--like services that run for a long time, as opposed to grid applications that tend to be more compute intensive and batch-like, needing a lot of resources for a limited amount of time (and this estimated completion time is used for prior reservation of the resources). Similarity, the unit of storage used by a cloud consumer can vary from I byte to petabytes, where a data grid is particularly useful for large-scale date storage and manipulation.

Since cloud applications execute on a web browser, they are much easier to use without any client software installed: whereas grid applications tend to be distributed and need a specific types of heterogeneous resources requiring appropriate grid schedulers for installation. Though the consumers here too can use a simple browser-like interface, the results of grid applications tend to large amounts of data that require sophisticated visualization tools to consume. The key aspect of cloud is abstraction of complex technologies-he it hardware, software or applications-and delivering it in the most simplistic fashion. The main advantage of clouds over grid is simplicity of usage and that of grids over cloud is efficient use of resources.

Comparing Grid and Cloud Computing:

Can we combine the two technologies? Though in principle, it is possible to deliver cloud computing services over a resource pool of a grid system, the business viability of harnessing such resources from different organizations to collectively deliver as a joint cloud infrastructure vendor seems less likely. Similarly, it is possible to think of a cloud infrastructure participating as one of the nodes in a resource pool enabling shared access to cloud-hosted paid infrastructure. Again, linking up the pay-per-use model with sharing is tricky. While the technologies are underlying both grid computing and cloud computing may converge or become interoperable ion the future, differences in the commercial aspects will remain, specifically around type of usage and access patterns.

II. PROPOSED RESEARCH WORK AND MODEL

The Grid computing discipline involves the actual networking services and connections of a potentially unlimited number of ubiquitous computing devices within a "grid". Grid Computing is widely regarded as a technology of immense potential in both industry and academia. The evolution pattern of grid technologies is very similar to the growth and evolution of internet. A computational grid is defined as hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.

Data movement requires secure data transfers, thus virtualized data storage mechanisms such as Storage Area Networks (SAN), network file systems, storage servers and virtual databases evolved that help developers to design such infrastructure with much more flexibility. Data-intensive grids is on data management of variety of data storage facilities in geographically dispersed locations capable of providing data virtualization services to provide transparency for data access, integration and processing

The open Grid Services Architecture (OGSA) describes the overall structure and services to be provided in the grid environment OGSA allows a system to perform a specific task, or solve a challenging problem, by using distributed resources over the Interconnection network. The Open Grid Service Infrastructure (OGSI) defines mechanism for creating, managing and exchanging information across the grid and standard interfaces and behaviors of grid services builder on platform web service

many Grid computing. Mechanismsinvolve static data, infrastructure for aggregation of resources for large scale problems solving, science, engineering and commerce. The Globus security Infrastructure provides security based upon public key encryption or any encryption technique with virtual organization.

The primary motivation for remote sharing application services deployed on community resources is virtual organization. In traditional mode of operation, from the owner each user has to obtain an account of each resource participating in VO (virtual organization which is not a satisfactory concept. Thus VO credentials allow the use of both hardware and software resources as its increasing trends or updates. Globus toolkit's resource management mechanism combining with rich VO policies to provide an architecture that allows authorization and authentications that using application services and traditional computing resources. This architecture involves GRAM and the Grid security Infrastructure (GSI) mechanisms.

III. LITERATURE SURVEY AND RESEARCH OBJECTIVES

Literature Survey : The Grid consists of loosely coupled, heterogeneous, and geographically dispersed computing elements that are connected by a network acting together to perform large tasks .Grids require general software libraries called the middleware to accomplish coordination among a large number of nodes that comprise them. Resource discovery is the process of finding the location of the required resources such as the database tables in the Grid. Resource allocation process, on the other hand, tries to map these resources to the application requirements for the best performance. Both resource discovery and resource allocation are active research areas for the grids. Recently, a number of systems have arisen that attempt to convert what is essentially a manual large-scale resource provisioning and programming problem into a more abstract notion commonly referred to as elastic, utility, or cloud computing.

Infrastructure and information technology primary idea is to provide or enable the people more effectively and efficiently to perform their tasks.

The computations in modern scientific problems are so huge that current solutions are either incapable of solving those or may take large amount of time to solve those. The Grid computing is a solution that meets such requirements and gives optimal solution by reducing computational time by use of a large number of integrated resources. The grid computing is analogues to electric grid which gives consistent, steady and transparent access to its shared resources irrespective of source. It is often constructed over LAN, WAN OR internet environments. The grid has multiple resource sites that offer computing resources like workstations, large servers, a mesh of processors and Linux clusters to satisfy a chain of computational needs.

According to Globus Proposed model, the grid computing is an infrastructure that enables the integrated, collaborative use of high-end computers, networks, databases and scientific instruments owned and managed by multiple organizations.

The Grid bus defines grid computing as a type of parallel and distributed system that enables the sharing, selection and aggregation of geographically distributed autonomous resources dynamically at runtime depending on their availability, capability. Performance, cost and user's quality of service requirements.

There are many benefits associated with grid are:

- Provides heterogeneous system support
- Provides increased capacity and Productivity
- Supports virtual resources and virtual
- Organizations
- Provide load balancing of resources
- Provides scalability
- Improves execution by reducing time

The following broad categories of requirements:

- Trust—A grid deployment must achieve high levels of trustworthiness and the trust infrastructure must accommodate the unique needs for federated control.
- Management Reporting—the grid environment should provide a variety of reports that support managements' need to understand the deployment and utilization of resources.
- Monitoring-- A wide variety of processing tasks may be using the resources of he grid, and these need to be monitored in real-time and both normal and abnormal events must be reported using various modalities.
- Service Levels—Commercial collaborators who choose to use grid technologies have concrete expectations about service levels tied to specific business relationships; the tools and techniques for monitoring and managing service levels must be present.
- Data Catalogs and Replicas-Sharing data resources is often the compelling motivation for the deployment of computing grids. Meta-data based mechanisms support the needs for data distribution in grids.

In addition to these management concerns, facilities that support end user ease-of-use have an impact on the robustness of management tools.

IV. PROPOSED WORK VALIDATED ON CASE SUTDY

- We propose a new decentralized access control scheme for secure data storage in clouds that

supports anonymous authentication. In the proposed scheme, the cloud verifies the authenticity of the server without knowing the user's identity before storing data.

- The scheme prevents replay attacks and supports creation, modification, and reading data stored in the cloud. We also address user revocation
- The main aim of this research is that the identity of the user is protected from the cloud during authentication.
- Revoked users cannot access data after they have been revoked.
- To avoid replay attacks in the cloud and also improve the security to the cloud data

V. SCOPE OF WORK

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort. Cloud providers typically use a "pay as you go" model. Cloud computing is a kind of grid computing; it has evolved by addressing the QoS (quality of service) and reliability problems. Cloud computing provides the tools and technologies to build data/compute intensive parallel applications with much more affordable prices compared to traditional parallel computing techniques.

Attribute-Based Encryption (ABE) and Proxy Re-Encryption (PRE) scheme are one of the schemes to flexibly control cloud data access in an efficient way by integrating the concept of trust and reputation evaluation into a cryptographic system

It is proposed a middleware technology framework for data management and distribution in grid computing. The goal of this framework is to provide good resource allocation for grid application and support collaboration with consistency or transparency over field test on distributed data which automates. These grid frameworks with Open grid Services Architecture (OGSA) & GT4 has main characteristics providing middleware technologies along with distributed computing. The data distribution or resource distribution have effective grid collaboration framework. Thus in this paper we discuss about the architecture /framework of Grid computing with required security policies in present environment which allows the heterogeneous resources with grid resource broker. We also discuss the elements of Grid and various grids that gives the choices according to the user applications. P2P middleware grid technology provides re-configurability dynamics which helps from failure by load dynamics .Globus toolkit, OGSA, OGSI, provides Virtual organization for data managements and resource managements are handled with great security level job submitters and administrators to access .This has its future enhancement for big data and cloud computing in which still security policies to be increased in their level of criteria authentication or key provisions

We have presented a decentralized access control technique with anonymous authentication, which provides user revocation and prevents replay attacks. The cloud does not know the identity of the user who stores information, but only verifies the user's credentials. Key distribution is done in a decentralized way. One limitation is that the cloud knows the access policy for each record stored in the cloud. The scheme introduced for realizing scalable, flexible, and fine-grained access control in cloud computing. The scheme seamlessly incorporates a hierarchical structure of system users by applying a delegation algorithm to R-HABE. Also achieves efficient user revocation because of multiple value assignments of attributes which allows flexible access revocation capability also. Revocation is achieved using interpolation which does not require any change in other user's shares. . A sequence of walk through steps for both the initial key agreement and revocation provided also show how these work together. We formally proved the security of scheme based on the security of CP-ABE. Finally, we improve the flexibility, scalability and fine grained access control of this system by reducing the number of keys generated for same attributes in the same level of organization. We implemented the proposed scheme, and conducted comprehensive performance analysis and evaluation, which showed its efficiency and advantages over existing schemes. Thus R-HABE can also introduce fuzzy –IBE as its future work which is an application of polynomial for biometric identity and also for error tolerance identity

VI. PROPOSED MODEL

The knowledge required to develop complex software has historically existed in programming folklore, the heads of experienced developers, or buried deep in the code. These locations are not ideal since the effort required to capture and evolve this knowledge is expensive, time-consuming, and error-prone. Many popular software modeling methods and tools address certain aspects of these problems by documenting how a system is designed. However, they only support limited portions of software development and do not articulate why a system is designed in a particular way, which complicates subsequent software reuse and evolution.[77]

Patterns, frameworks, and middleware are increasingly popular techniques for addressing key aspects of the challenges outlined above. Patterns codify reusable design expertise that provides time-

proven solutions to commonly occurring software problems that arise in particular contexts and domains. Frameworks provide both a reusable product line architecture [1] – guided by patterns – for a family of related applications and an integrated set of collaborating components that implement concrete realizations of the architecture. Middleware is reusable software that leverages patterns and frameworks to bridge the gap between the functional requirements of applications and the underlying operating systems, network protocol stacks, and databases. This paper presents an overview of patterns, frameworks, and middleware, describes how these technologies complement each other to enhance reuse and productivity, and then illustrates how they have been applied successfully in practice to improve the reusability and quality of complex software systems.Fig.1 provides Middleware Layers in Context. Figure 2 provides relationship between framework components.

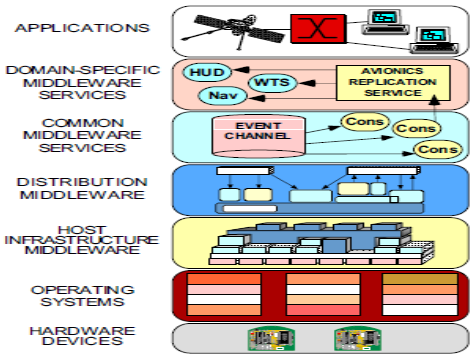

Fig. 1 Middleware Layers in Context

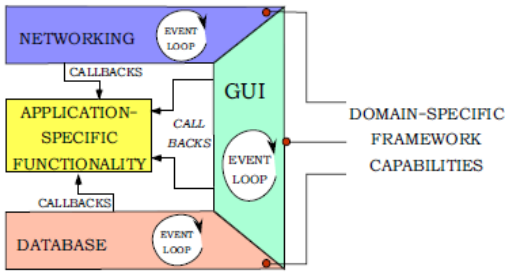

Fig. 2 Relationships between Framework Components

VII. DATA MANAGEMENT

A grid is a collection of distributed computing resources over a local or wide area network that appears to an end user or application as one large virtual computing system.

A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high end computational capabilities.

Web Services and Grid Computing:

The users of such grids viz. citizens will require web services over the Internet. They will not b e interested or required to know of any details of hardware or software resource locations or resource allocations management. All this has to be provided by the grid computing environment. Thus, integrating web service with grid architecture becomes a necessity for these purposes. The Open Grid Services Architecture (OGSA) becomes essential to offer effective, shameful web services based on Service Oriented Architecture (SOA) on the grid.

Key Functional Requirements in Grid Computing:

In any grid the functional elements are: [79]

- Resource Management: The ability to keep track, allot and remove grid resources.
- Security Management: The ability to ensure authenticated and authorized access to grid Resources, from the users in the external world.
- Data Management: The3 ability of transporting, cleaning, parceling and processing the Data, between any two nodes in the grid, without the knowledge of the user.
- Services Management: The ability of the users and applications to query and obtain response from the grid efficiently.

Table 1 Provides Layered Architecture:

Table 1: Layered Architecture

1.	Infrastructure layer-Processors, storage, software, data
2.	Security layer-Authentication and authorization
3.	Job management layer-job scheduling, job management, accounting
4.	Resource management layer-Resource access and management, and scheduling services
5.	Middleware layer-Tools, languages, libraries
6.	Application layer-Scientific/engineering, commercial, governance applications

Table 2 Provides object oriented Grid tool kit:

Table 2: object oriented Grid tool kit

Application		
Legion library (method invocation server)		
Legion	Common	Resource
File	Space	Management
System	Directory	Service
	Service	
Legion object management server (core objects)		
Computational infrastructure		

Data Management:

This model describes the basics of data transfer on grid environments using the GridFTP APIs provided by the Java Commodity Grid (COG) Kit 1.2 and the Globus Toolkit,[71]

This model includes information on the following topics:

- An overview of the GridFTP protocol for remote file transfer
- A sample Java program for connecting to theGridFTP server, performing single, multiple, or parallel transfers.
- Common mistakes and error usages seen when working with GripFTP and troubleshooting ti8ps for those errors
- Transferring files using other grid protocols such Globus Access to Secondary Storage (GASS)
- GridFTP is a high performance file transfer protocol designed specifically for grid environments. The following sections describe GridFTP features along with source code.

Computational grids provide the infrastructure for powerful new tools for investigation, including desktop computing, smart instruments, collaboration, and distributed commuting. The Globus Proposed models are engaged in defining and developing a persistent data grid with the following capabilities [Reliable Data Transfer]:

- High-performance, source, robust datatransfer mechanisms
- A set of tools for creating and manipulating Replies of large datasets
- A mechanism for maintaining a catalog of Dataset replicas

GridFTP is a high-performance-secure, reliable data transfer protocol optimized for high-bandwidth wide-area networks. The GridFTP protocol is based on FTP, the popular Internet file transfer protocol [Reliable Data Transfer]. This protocol and family of tools were born from a a realization that the grid environment needed a fast, secure, efficient, and reliable transport mechanism. According to Allocock and colleagues [Grid Data Management large decentralized computational grids require a robust transport mechanism with the following features:

- Parallel data transfer:

Multiple TCP streams to improve bandwidth over using a single TCP stream Parallel data transfer is supported through FTP command extensions and data channel extensions.

- Grid Security Infrastructure (GSI) and Kerberos authentication support:

Use-controlled settings of various levels of data integrity and confidentiality, user-controlled settings of various levels of data integrity and confidentiality. This feature provides a robust and flexible authentication, integrity, and confidentiality mechanism for transferring files.

- Third-party control of data transfer**:**

Support for managing large data sets for large distributed communities. This provides third-party control of transfers between storage servers.

- Striped data transfer:

Capabilities to partition data across multiple servers to improve aggregate bandwidth. GridFTP supports stripe data transfers through extensions defined in the Grid Forum Draft.

- Partial File Transfer:

New FTP commands to support transfers of regions of a file, unlike standard FTP that requires the allocation to transfer the entire file, file, unlike standard FTP that requires the application to transfer the entire file.

- Reliable data transfer:

Fault recovery methods for handling transient network failures and server outages and for restarting failed transfers.

- Manual Control of TCP buffer size:

Support for achieving maximum bandwidth with TCP/IP.

- Integrated Instrumentation:

Support for returning restart and performance markets.

- Connecting in a GridFTP Server:

The Java program in Listing1 implements a basic GridFTP transfer to the local file system.

Listing 1:

Listing 1 GridFTP.java A client program to transfer files via Grid FTP

```
package grid.ftp;

import java.io.*;

import org.globus.ftp.*;
import org.globus.ftp.exception.*;
import org.globus.gsi.*;
import org.globus.gsi.gssapi.*;
import org.ietf.jgss.GSSCredential;

import java.util.*;
import org.apache.log4j.Logger;
import org.apache.log4j.Level;

import java.security.cert.X509Certificate;

/**
 * GridFTP: Transfer files via the GridFTP protocol
 * run grid-proxy-init before running this program
 *
```

```
 * Both client and server must have each other CA certificates
 * for mutual authentication to work
 */
public class GridFTP
{
   private static Logger logger =

Logger.getLogger(GridFTP.class.getName());

   // Protocol client,  provided by he Cog API
   private GridFTPClient client = null;

   // default port
   private static int GRIDFTP_PORT = 2811;

   /**
    * Constructor
    * @param host remote GridFTP host
    * @param port remote GridFTP port (default is 2811)
    */
   public GridFTP(String host, int port)
      throws ServerException, IOException
   {
         client = new GridFTPClient(host, port);

         /**
          * Authenticate using the default credentials.
          * Requires GSI to be configured properly on the client.
          * Including user cert/key pair an CA certificates
          *
          */
         client.authenticate(null);
   }
```

The Globus Toolkit uses the standard Apache log4j package from http://jakarta.apache.org/log4j/docs/ to displays log messages by defining a static logger;

```
Private static Logger logger =
Logger.getLogger
(MyGrdFTP.class.getName ( ) ):
```

This package is very popular among Java programmers and can be very helpful when debugging your classes. The class constructor takes the hostname and port as arguments and authenticates against the server using GSI credentials.

This class works only with the Globus Toolkit 2.2 or later. GSI changed significantly after version 2.0. The functionality of the org.globus.security.GlobusProxy class is largely replaced by the or.globus.gs1.GlobusCredential class. However, Globus recommends not using the org.globus.gsi.GlobusCredential class because it is a representation of Public Key Infrastructure (PKI) credentials that are specific to one security protocol. Instead, Globus recommends using the Generic Security Service (GSS) abstractions as much as possible.

The code snippet from Listing 2 shows how to do this conversion:

```
GlobusCrdential globusCred = new GlobusCredential (…);
GSSCredential creg = new GlobusGSSCredential Impl ( lobusCred,
GSSCredential.DEFAULT_LIFETIME);
```

Transferring Data:

The code in Listing 2 implements a single file transfer through GridFTP

Listing 2:

Listing 2 Single GridFTP transfer

```
/**
 * Transfer a file from a remote host
 * @param remoteFile Remote file. It must exist in the server
 * @param localFile Where should the remote file be stored?
 * @param transferType FTP transfer type. One of:
 *                     GridFTPSession.ASCII
 *                     GridFTPSession.BINARY
 */
public void download(String remoteFile
, String localFile, int transferType)

   throws ServerException, ClientException, IOException

{
  // remote file size
  long size = client.getSize(remoteFile);

  // check if remote file exists
  // if not an exeception will be thrown...
  if (client.exists(remoteFile)) {
    client.setType(transferType);

    /** required to transfer multiple files **/
    client.setLocalPassive();
    client.setActive();

    final FileOutputStream fos = new FileOutputStream(localFile);
```

```
   // get the file, use the DataSink interface to
write incoming data
   // Implement this interface to provide your
own ways
   // of storing data.
   // It must be thread safe; in parallel transfer
mode several
   // streams may attempt to write.
   client.get(remoteFile, new DataSink() {
public synchronized void write(Buffer buffer)
throws IOException
{
System.err.println(
"received " + buffer.getLength() + " bytes");

fos.write(buffer.getBuffer());
}

public void close() throws IOException {
// close File output streams
fos.flush();
fos.close();
};
   }, null);

   // transfer ok
   logger.info(
         "Successfully transfered: "
               + remoteFile
               + " to "
               + localFile
               + " size: "
               + size);

 } else {
       System.err.println(remoteFile      +      "
doesn't exist");
 }
}
```

Transferring Multiple Files:

At first glance, a parallel transfer may sound as though the client is capable of transferring multiple files from multiple servers in Kazaa or Morpheus style. In reality, however, parallel transfer means simply that multiple streams will be opened to transfer the same file from the same server. For two-party transfers, GridFTP will only add overhead in single processor machines [Reliable Data Transfer]. In any case, the client will transfer multiple copies of the same file without any slicing whatsoever, although it may increase performance if you have a multiprocessor system.

In case of two-party transfer, parallelism should be carefully chosen. The advantage of having multiple streams has mostly to do with low-level TCP procedures and is also related to the TCP window size. Using twice-the number of parallel streams will not necessarily produce twice the performance. Actually from a certain point, you will experience a decrease in performance. Current implementation of the FTP package handles each data pathway in a separate thread, so unless your machine has multiple CPUs, you only add computing overhead by increasing parallelism [ReliableDataTransfer04].

A parallel transfer requires extended mode. Furthermore, the transfer type must be image, and the data sink/source must support random data access and be threading safe. Multiple threads may write to it.

Troubleshooting:

Most of the problems in writing this code relates to dealing with legacy proxies and converting them to GSSCredentials, Take a careful look at the constructor of this class to make sure you understand the conversion process. Also ensure that you are running the correct versions of the Globus Toolkit and the GridFTP servers. Older versions (before 1.5) are bit enabled for GSSAPI.

Also make sure your client-side java libraries, prvide3d by the CoG it, are properly configured in your class path. A set of user certificates is also required to connect through GSI.

Runtime or Defective Credential Errors:

- Defective credentials indicate a problem with your client and server certificates. The following tips should be kept in mind when you run the code ion this model:
- Make sure your java environment is setup properly. This is done by setting the environment variable GLOBUS_LOCATION to the install directory, and running the java environment configuration script

```
$GLOBUS_LOCATION/etc/globus.dev-
env. {csh, sh}.
```

- A GSI proxy should be generated first on the client by running the command grid- Proxy-init.

Authentication Failed Errors:

If you run into an authentication failed error, here are some tips that can help you fix them:

- For mutual authentication to succeed, both client and server require a use certificate, private key, and CA certificate to be properly configured. On the client, the path to these files is described in the file cog.properities found in the user's home directory $HOME/.globus (in UNIX systems).

Transferring Files Using Multiple Protocols:

The transfers are performed using URLs of the form:

PROTOCOL: //HOST / FILE
For example to transfer a file between two GridFTP servers:
Source URL: gsiftp://host 1:2811/c:/temp/file.xml
Destination URL: gsiftp://host2.2811/c:/temp/file.xml
Or, to perform a third-party transfer from a GASS server to a GridFTP server:
Source URL: https://host 1:3154/c:/temp/file.xml
Destination URL: gisftp://host2:2811/tmp/filr.xml

GridFTP is a secure, reliable data transfer protocol optimized for high-performance networks based on FTP, the popular Internet protocol. This model has provided a developer's overview of GridFTP features withy sample programs based on the lava Cog Kit API provided by the Globus Toolkit. GridFTP is the foundation of data management services, which constitute the second pillar of the Grid Services Architecture.

Grid Extended Cloud Computing Case Study Validation:

Cloud computing is frequently compared to grid computing Grid computing also has the same intent of abstracting our computing resources to enable utility models and was proposed at least a decade earlier than cloud computing and there are many aspects of grid computing that have formed the basis of the requirements placed on a cloud . Having said that, there are also very specific differences between a grid computing infrastructure and the features one should expect from a cloud computing infrastructure. This can be seen by first describing some funds mental aspects of grid computing and then comparing them with those of cloud computing.

Motivation of the Thesis;

Cloud Computing has been envisioned as the next-generation architecture of IT Enterprise. It moves the application software and databases to the centralized large data centers, where the management of the data and services may not be fully trustworthy. This unique paradigm brings about many new security challenges, which have not been well understood. This work studies the problem of ensuring the integrity of data storage in Cloud Computing. In particular, we consider the task of allowing a third party auditor (TPA), on behalf of the cloud client, to verify the integrity of the dynamic data stored in the cloud. The introduction of TPA eliminates the involvement of the client through the auditing of whether his data stored in the cloud are indeed intact, which can be important in achieving economies of scale for Cloud Computing. The support for data dynamics via the most general forms of data operation, such as block modification, insertion, and deletion, is also a significant step toward practicality, since services in Cloud Computing are not limited to archive or backup data only. While prior works on ensuring remote data integrity often lacks the support of either public auditability or dynamic data operations, this paper achieves both. We first identify the difficulties and potential security problems of direct extensions with fully dynamic data updates from prior works and then show how to construct an elegant verification scheme for the seamless integration of these two salient features in our protocol design. In particular, to achieve efficient data dynamics, we improve the existing proof of storage models by manipulating the classic Merkle Hash Tree construction for block tag authentication. To support efficient handling of multiple auditing tasks, we further explore the technique of bilinear aggregate signature to extend our main result into a multiuser setting, where TPA can perform multiple auditing tasks simultaneously.

Objectives:

- Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
- It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
- When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

Problem Definition:

In the Existing systems, the notion of public audit ability has been proposed in the context of ensuring remotely stored data integrity under different system and security models. Public auditability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data. However, most of these schemes do not consider the privacy protection of users' data against external auditors. Indeed, they may potentially reveal user's data to auditors. This severe drawback greatly affects the security of these protocols in cloud computing. From the perspective

of protecting data privacy, the users, who own the data and rely on TPA just for the storage security of their data, do not want this auditing process introducing new vulnerabilities of unauthorized information leakage toward their data security.

Disadvantages of existing system:

- Although the infrastructures under the cloud are much more powerful and reliable than personal computing devices, they are still facing the broad range of both internal and external threats for data integrity.
- Second, there do exist various motivations for CSP to behave unfaithfully toward the cloud users regarding their outsourced data status.
- In particular, simply downloading all the data for its integrity verification is not a practical solution due to the expensiveness in I/O and transmission cost across the network. Besides, it is often insufficient to detect the data corruption only when accessing the data, as it does not give users correctness assurance for those unaccessed data and might be too late to recover the data loss or damage.
- Encryption does not completely solve the problem of protecting data privacy against third-party auditing but just reduces it to the complex key management domain. Unauthorized data leakage still remains possible due to the potential exposure of decryption keys.

Problem Solution:

In this paper, we utilize the public key based homomorphism authenticator and uniquely integrate it with random mask technique to achieve a privacy-preserving public auditing system for cloud data storage security while keeping all above requirements in mind. To support efficient handling of multiple auditing tasks, we further explore the technique of bilinear aggregate signature to extend our main result into a multi-user setting, where TPA can perform multiple auditing tasks simultaneously. Extensive security and performance analysis shows the proposed schemes are provably secure and highly efficient. We also show how to extent our main scheme to support batch auditing for TPA upon delegations from multi-users.

Advantages of proposed system:

- Public auditability: to allow TPA to verify the correctness of the cloud data on demand without retrieving a copy of the whole data or introducing additional online burden to the cloud users.
- Storage correctness: to ensure that there exists no cheating cloud server that can pass the TPA's audit without indeed storing users' data intact.

Privacy preserving: to ensure that the TPA cannot derive users' data content from the information collected during the auditing process.

- Batch auditing: to enable TPA with secure and efficient auditing capability to cope with multiple auditing delegations from possibly large number of different users simultaneously
- Lightweight: to allow TPA to perform auditing with minimum communication and computation overhead.

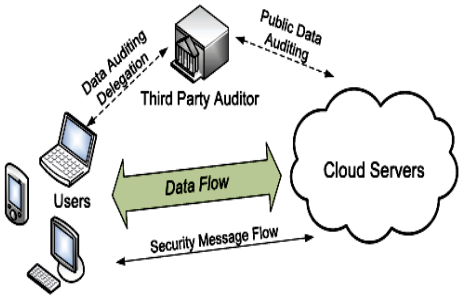

Fig. 1. The architecture of cloud data storage service.

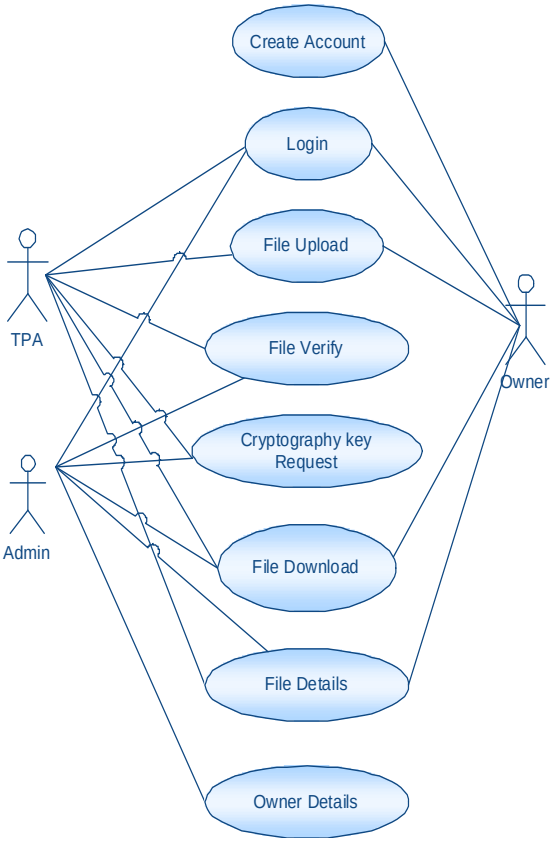

Fig. Use case Diagram

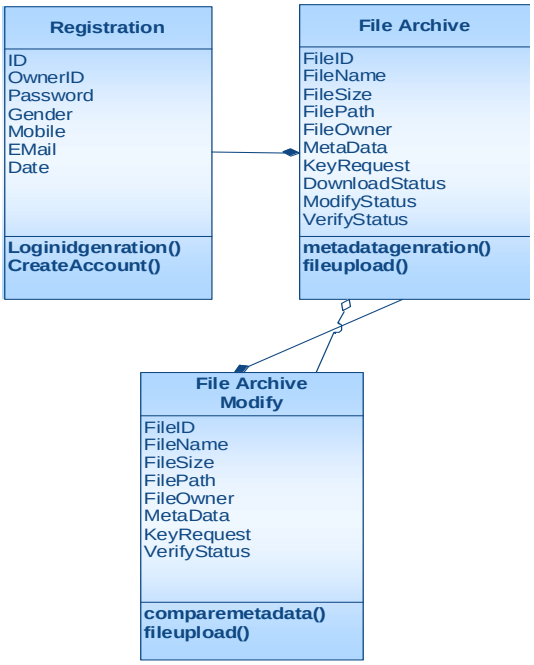

Fig. Class Diagram

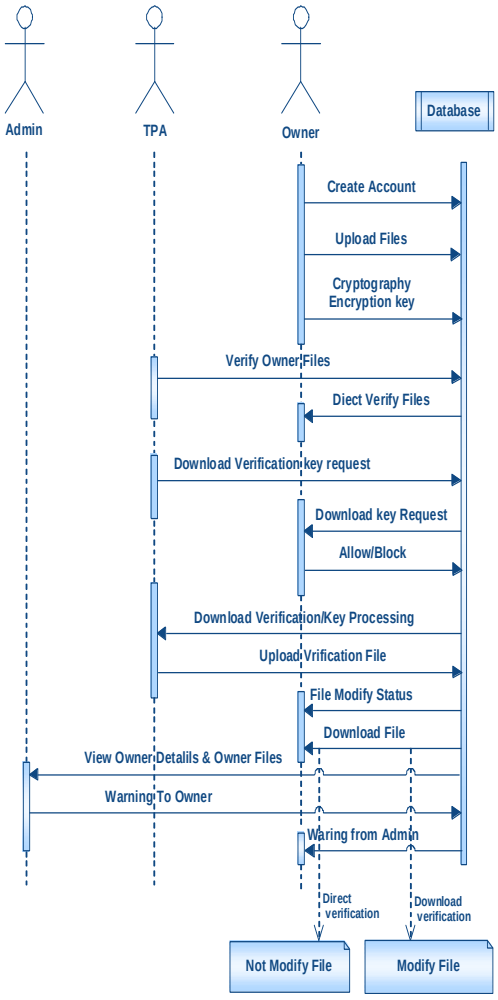

Fig. Sequence Diagram

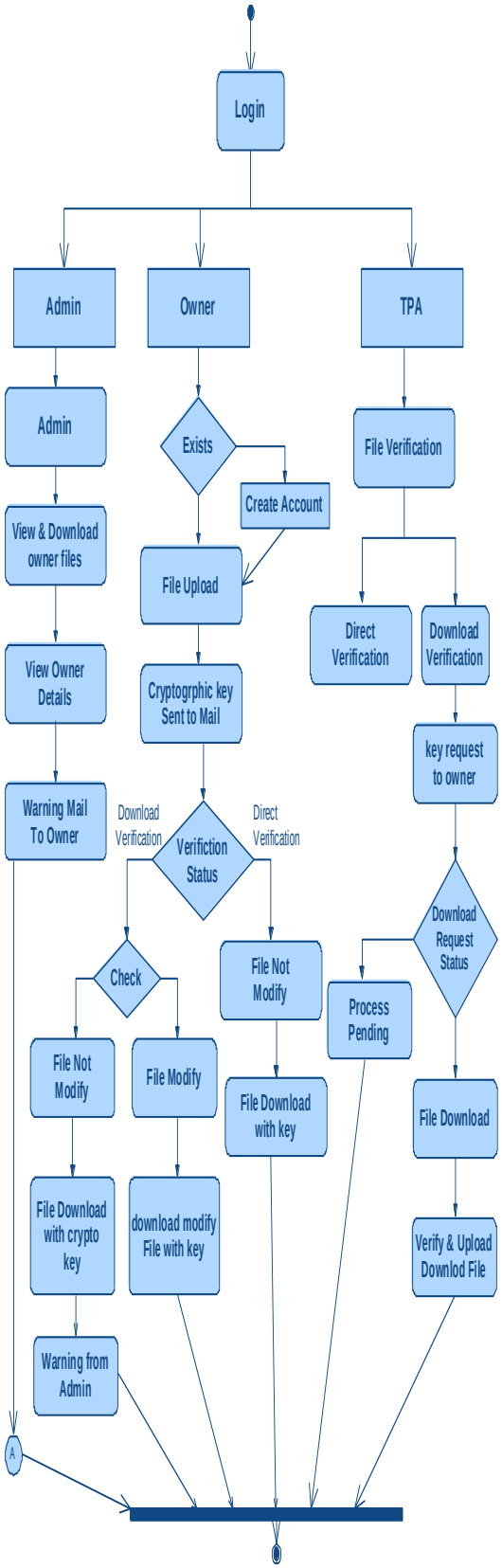

Fig. Activity Diagram

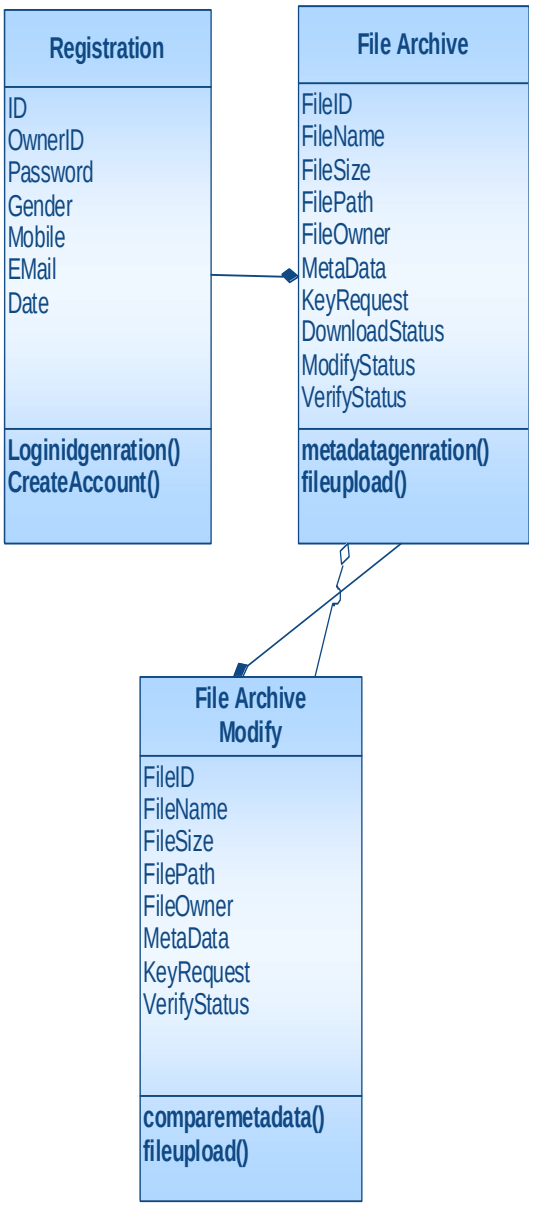

Fig. Implementation of system using class diagram

SCREEN SHOTS;

Fig. Home window

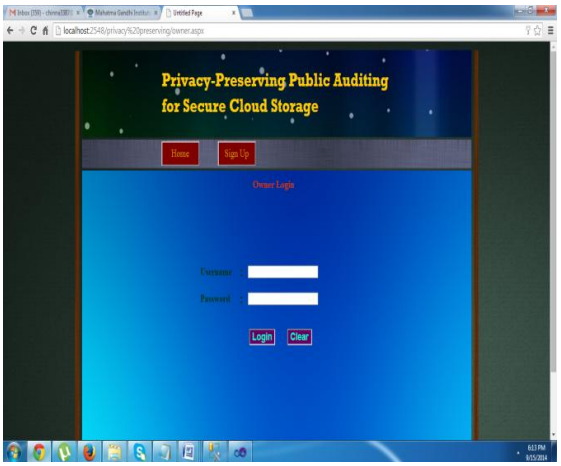

Fig. Owner login form window

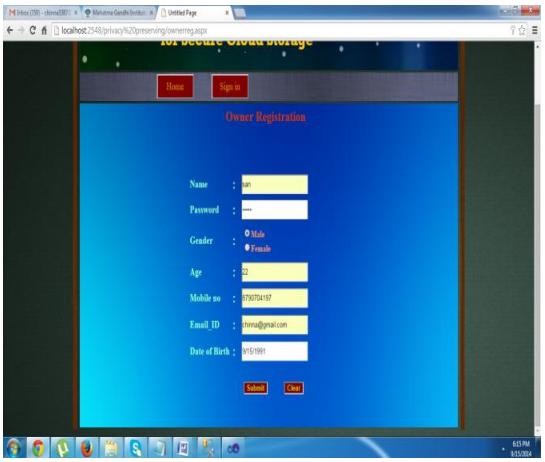

Fig. Owner registration window

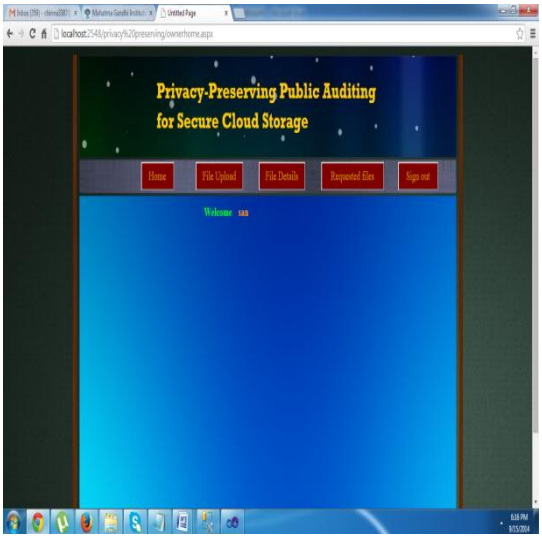

Fig. Owner window after login is success

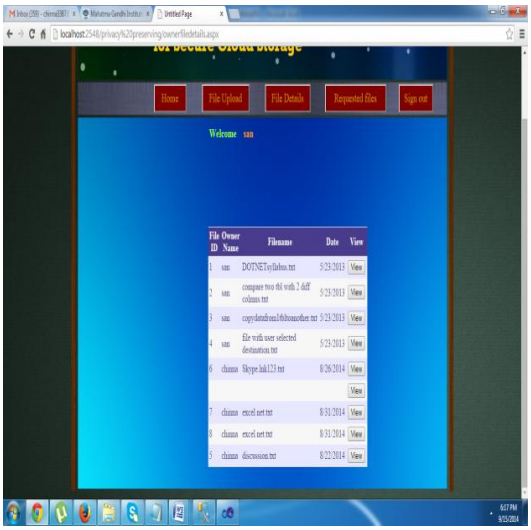

Fig. Owner file detail window

Fig. User login window

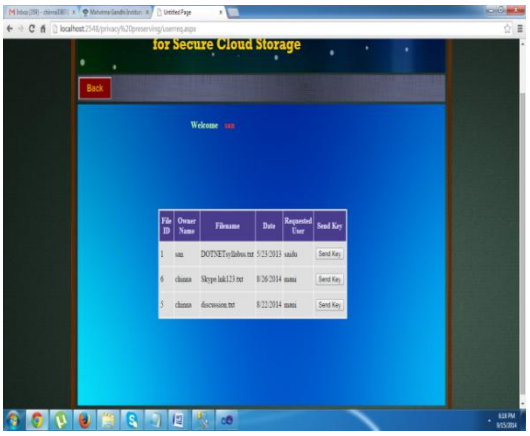

Fig. User requested files on owner window

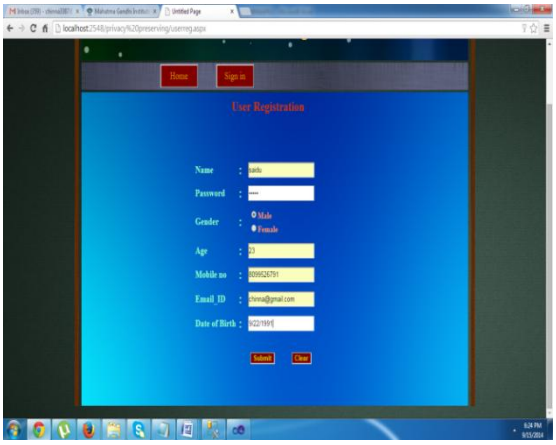

Fig. User registration login window

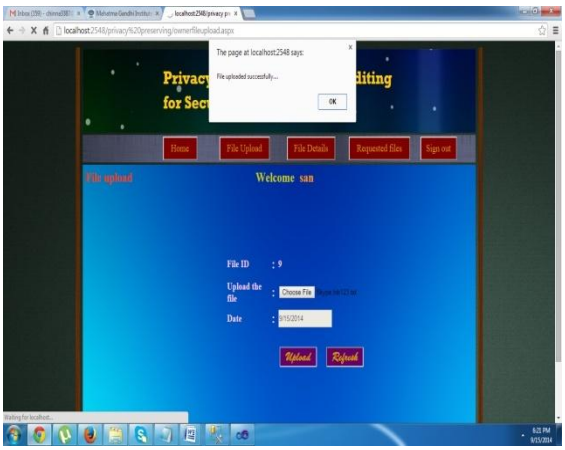

Fig. Owner uploaded files window

Fig. TPA verified files on user window

Fig. User sees all files uploaded by owners

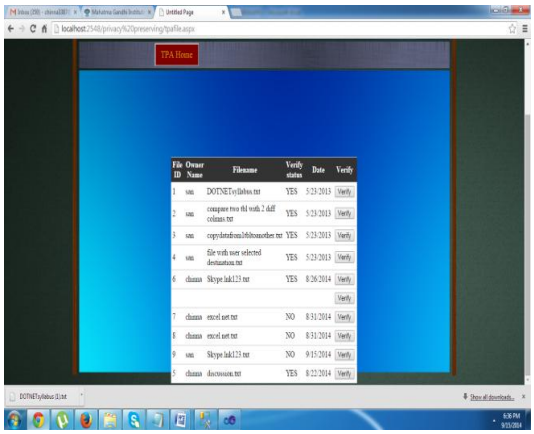

Fig. TPA window after login success

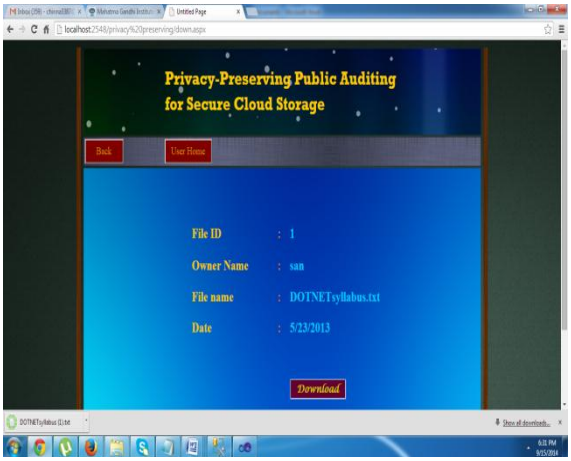

Fig. User download owner files

Fig. TPA verify the owner files

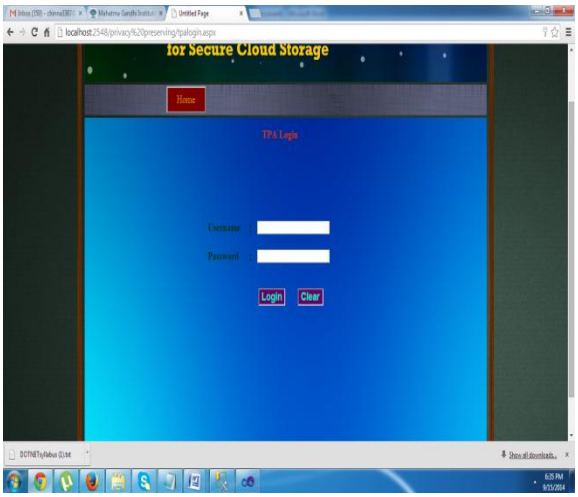

Fig. TPA login form

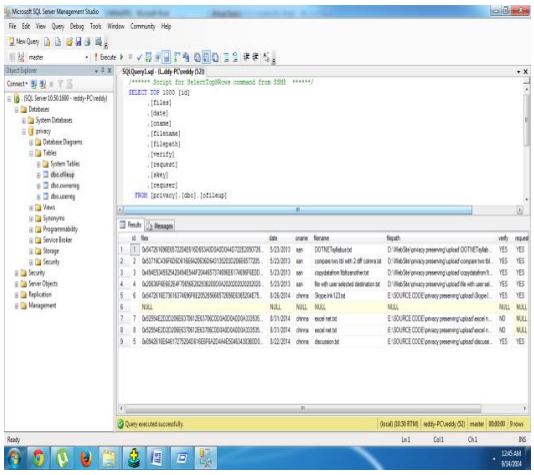

Fig. TabofileupDatabase Table

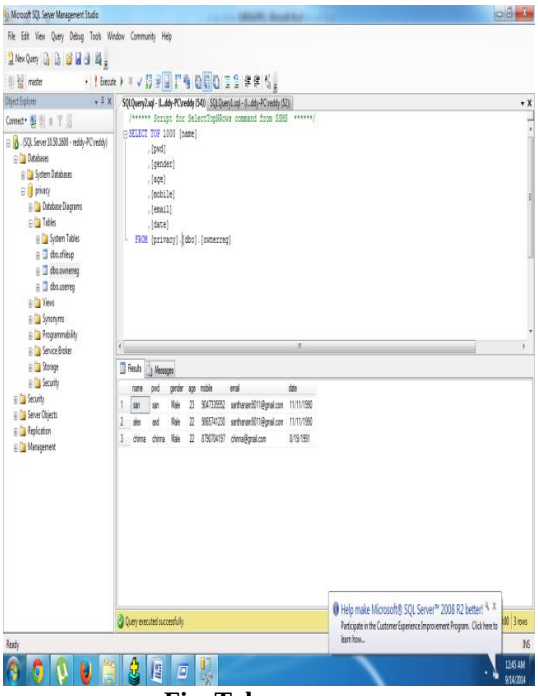

Fig. Tabownerreg

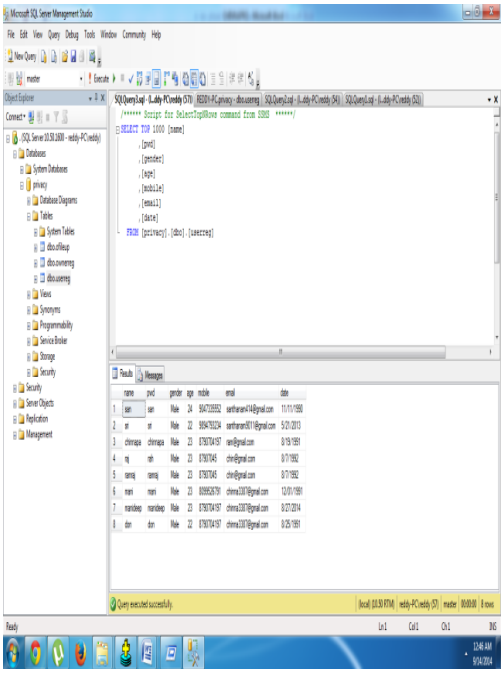

Fig. Userreg

VIII. CONCLUSION

This research paper is a novel and innovative idea of providing a security framework for Data Management and Distribution for Middleware technology for Grid extended Clouds, with a validation on a case study. The proposed research model is an Object Oriented Pattern and Framework using UML Security Design Model for Authnticaton and Authorization for Data Management and Distribution in Grid extended clouds. Appropriate implementations are performed in case study to adequately validate the proposed Object oriented model. This model can be extended for High Performance Computing in Distributed Systems and Big Data.

REFERENCES

[1] M Ozaki, Dinkar Sitaram, Geetha Manunath, ”Moving to the Cloud”, Syngress Elsevier 2012, ISBN: 978-1-59749-725-1.

[2] R.E. Rajkumr Buyya, JamesBroberg, Andrzej Goscinski, “Cloud Computing Principles and Paradigms”, Wiley 2016, ISBN: 978-81-265-4125-6.

[3] AravindDoss,RajeevNanda,“CloudComputin g A Practitioner’s Guide” McGraw Hill Education,2013,ISBN(13): 978-1-25-906571-8, IBN (10): 1-25-906571-2.

[4] Syed A. Ahson, Mohammad Ilyas, “Cloud Computing and Software Services Theory and Techniques”, CRC Press, ISBN: 976-1-4398-0315-8.

[5] Jose C. Cunha,Omer F.Rana (Eds),”Grid ComputingSoftwareEnvironmentsand Tools”, Springer 2006, ISBN: 81-8128-461-5.

[6] Bhushan Jadhave. Sonli Jadhav, “Grid and Cloud Computing”, Technical publications 2017, ISBN: 978-93-332-1162-8.

[7] Fran Berman,Geoffrey C,Fox, Anthony J.G.Hey, “Grid Computing Making the Global Infrastruccture A Reality”, Wiley 2010, ISBN: 978-81-265-2722-9.

[8] Ahmar Abbas, “Grid Computing: A Practical Guide To Technology And Applications”, Charles River Midia Inc. 2006, ISBN: 81-7008-626-4.

[9] Dharanipragada Janakiram, “Grid And Cloud Computing”, Mc Graw Hill 2016, ISBN: 13: 978-93-392-2147-8./*

[10] Grid computing –Joshy joseph and craig fellenstein, Published by Pearson Education, Inc., prentice Hall PTR .

[11] DanielNurmi,RichWolski, Chris Grzegorczyk Graziano Obertelli, Sunil Soman, Lamia Youseff,DmitriiZagorodnov,"The Eucalyptus Open-sourceCloud-computing System",PP.1-8.

[12] Paolo Maggi and Riccando Sisto “A Grid-Powered Framework to Support Courses on DistributedProgramming” IEEE Transactions on Education, VOL.50.No.1. February 2007. PP.21-33.

[13] Ian Foster,CarlKesselman, Jeffrey M.Nick, Steven Tuecke,"Grid Services for Distributed System Integration",0018-9162/02/$17.00 © 2002 IEEE, June 2002, pp.1-46.

[14] IanFoster,CarlKesselman,StevenTuecke,"The Anatomy of the Grid",pp.1-24.

[15] DhineshBabuL.D.aP.VenkataKrishnab,"Appl ied Soft Computing",2013 Elsevier B.V. pp.2292–2303
[16] Bernard Cohen, Rajkumar Buyya. Manzur Murshed,"A Toolkit for the Modeling andSimulationofDistributedResourceManage mentandSchedulingforGrid Computing",pp.1-37.
[17] Geoff Coulson, Paul Grace, Gordon Blair, Laurent Mathy DavidDuce, Chris Coopoer, Wai Kit Yeung, Wei Cai 'Computing Dept. Lancaster University, LA 14YR, UK. "Towards A Component-Based Middleware Framework For Configurable And Reconfigurable Grid Computing" IEEE International Workshops on Enabling Technologies Infrastructure for Collaboraive Enterprises (WETICE'04) 1524-4547/04 & IEEE. PP.
[18] Ken Kennedy , Mark Mazina, John Mellor-Crummey, Keith Coooper, Linda Torczon Rice University, Fran Berman, Andrew Chien, Holly Dail, Otto Sieyert Univeristy of California, San Diego "Toward a Framework for Preparing and Executing Adaptive Grid Program". Proceeding of the International Parallel and Distributed Processing Symposium (IPDPS'02) 2002
[19] David C. Rajkumar Buyya, "Economic-based Distributed Resource Management and Scheduling for Grid Computing",pp.1-166.
[20] Milan Lathia, "Advantages of Grid Computing", IEEE DISTRIBUTED SYSTEMS ONLINE Published by the IEEE Computer Society. Vol.6 No.2; 2005.
[21] H.bennoit-cattin.f.bellet ahmar abbas,"grid computing: a practical guide to technology and applications",charlesrivermedia, inc.pp..1-391.
[22] Coulson,G.,Blair, G.S., Clark,M., Parlavantazas,N.,"The Design of a Highly configurable and Reconfigurable Middleware Platform", ACM Distributed Computing Journal, Vol15, No2,pp109-126,April 2002.
[23] Neal LeavittLizhe Wang, Gregor von Laszewski,"Scientific Cloud Computing: Early Definition and Experience",PP.1-18.
[24] Ajith Abraham, Rajkumar Buyya ,Baikunth Nath,"Nature's Heuristics for Scheduling Jobs on Computational Grids",pp.1-8.
[25] Sergio Nesmachnowa,, Héctor Cancelaa, Enrique Albab, "Applied Soft Computing",2011 Elsevier B.V.doi:10.1016/j.asoc.2011.09.022, pp.626–639.
[26] Sankar Pal, Varun Talwar,"Web Mining inSoft Computing Framework: Relevence, State of the Art and Diorectins", IEEE, publisher item identifies S 1045-9227/05562-5, Vol.13, September,2002, PP.1163-1177.
[27] Pascale Vicat-Blose, Sebesties Soudas, Romanic Guillier, Brice Goglis, ―Compute Network, From Cluster to Cloud Computing‖, Johnwilles Sons, ISN; 978-1-84821-286-2, 2011.
[28] David P. Anderson,"BOINC: A System for Public-Resource Computing and Storage"
[29] [29] Dabu Panda, Arvind Maheshwari,‖ Middle Ware Management with Oracle Enter prize Manager Grid Control‖, 10gR5, Packb Publishing, ISBN: 978-1-847198-34-1, 2009.
[30] Rodrigo N. Calheiros,Rajiv Ranjan,C´esar A. F. De Rose,"CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms",Published online 24 August 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/spe.995, 2010 John Wiley & Sons, Ltd. pp.41:23–50.
[31] Guido Schnutz, Daniel Liebhart, Peter Welkerback, Service Oriented Architecture: An Integration Blue Print‖, ISBN: 978-1-849681-04-9, 2010.
[32] Maozhes li, Mark Baker,‖ The Grid Core Technologies‖, ISBN: 978-04-470-09417-4, 2005.
[33] Ajith Abrahama, Ravi Jainb, Johnson Thomasc,Sang Yong Hana,"D-SCIDS: Distributed soft computing intrusion detection system", Elsevier Ltd.doi:10.1016/j.jnca.2005.06.001,pp,1-19.
[34] H. Liu,Ajith Abraham,"Scheduling Jobs on Computational Grids Using Fuzzy Particle Swarm Optimization Algorithm",publication at: https://www.researchgate.net/publication/237144024,DOI:10.1016/j.future.2009.05.022, PP..1-19.
[35] Edoardo Patti, Angeliti Lydia Ateria Syrri Marco Jals, Pierlvisi Mascerlle,‖ Insrastructure for General Purpose Services is smart Grid‖, IEEE, DOI: 10.1109|TSG.2014-2375197.
[36] Klaus Krauter1, Rajkumar Buyya,Muthucumaru Maheswaran "A taxonomy and survey of gridresource management systems for distributed computing",SOFTWARE—PRACTICE AND EXPERIENCE, 2002; 32:135–164 (DOI: 10.1002/spe.432).
[37] Energy Nikuleteu, Evgesiy Plozhrik, Eles Lukyeschikov, Dmity Biuryukov, ―Features Management & Middleware of Hybrid Cloud Infrastructure‖, IJACSA,International Journal

of Advanced Computer Science & Applications, Vol A, No. PP-31-36, 2016.

[38] SushmitaRuj, Member, Ieee, Milos Stojmenovic, Member, Ieee, And Amiya Nayak,‖ Decentralized Access Control With Anonymous Authentication Of Data Stored In Clouds‖ Ieee Transactions On Parallel And Distributed Systems, Vol. 25, No. 2, February 2015.

[39] S. Ruj, A. Nayak, and I. Stojmenovic, ―DACC:Distributed access control in clouds,‖ in IEEE TrustCom, 2011.

[40] Sung-Soo Kim, Ji-Hwan Byeon, Hongbo Liu,Ajith Abraham,Sea´n McLoone"Optimal job scheduling in grid computing using efficient binary artificial bee colony optimization", Springer-Verlag Berlin Heidelberg 2012, Soft Comput (2013) 17:867–882,DOI 10.1007/s00500-012-0957-7.

[41] Antonio J. Nebro, Enrique Alba,Francisco Luna"Multi-objective optimization using grid computing",Springer-Verlag 2006,Soft Comput (2007) 11: 531–540, DOI 10.1007/s00500-006-0096-0.

[42] Ivan Krsul, Arijit Ganguly, Jian Zhang,José A. B. Fortes, Renato J. Figueiredo,"VMPlants: Providing and Managing Virtual Machine Execution Environments for Grid Computing",0-7695-2153-3/04,2004 IEEE,pp.1-12.

[43] G. Laccetti, G. Schmid, (2007) "A framework model for grid security", Future Generation Computer Systems, Volume 23, Issue 5, June 2007, pp.702-713.

[44] I. Foster and C. Kesselman, eds., The Grid: Blueprint for a Future Computing Infrastructure, Morgan Kaufmann, San Francisco, 1999.

[45] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger, Economic models for resource management and scheduling in Grid computing, Concurrency and Computation: Practice and Experience 14(13–15):1507–1542 (Nov.–Dec. 2002).

[46] Joseph, Craig Fellemtein "Grid Computing", PEARSON,, 2004, ISBN:978-81-317-0885-9.

[47] Anirban Chakrabarti, "Grid Computing", Springer 2007, ISBN: 978-3540-44492-3.

[48] C. Anton-Haro and M. Dohler, Machine-to-machine (M2M) Com-munications: Architecture, Performance and Applications. Elsevier, 2014.

[49] Y.S. Dai, M. Xie and K.L. Poh, "Availability Modeling and Cost Optimization for the Grid Resource Management System", IEEE Transactions on Systems, and Cybernetics — Part A: Systems and Humans, Vol. 38, No. 1, pp.170-179.

[50] Goel, S., Sobolewski, M., "Trust and Security in Enterprise Grid Computing Environment" Proceedings of the IASTED International Conference on Communication, Network and Information Security, New York, USA 2003.

[51] C.S.R Prabha,"Grid Computing", PHI, 2008, ISBN:978-81-203-3428-1.

[52] [52] rajkumar buyya and srikumar venugopal"market-oriented computing and global grids:an introduction". market-oriented grid and utility computing edited by rajkumar buyya and kris bubendorfer copyright 2010 john wiley & sons, inc.

[53] Y. Tanaka, H. Nakada, S. Sekiguchi, T. Suzumura, and S. Matsuoka, Ninf-G: A reference implementation of RPC-based programming middleware for Grid computing Journal of Grid Computing 1(1):41–51 (2003).

[54] Ian Foster, Carl Kesselman,"Grid 2" Morgen Kaufanann publishers, 2006, ISBN-13:978-81-312-0200-5.

[55] I. Foster, C. Kesselman, J. M. Nick, S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration". Draft, June 2002.
http://www.globus.org/research/papers/ogsa.pdf

[56] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich, "The Open Grid Services Architecture, Version 1.0", January 2005, Global Grid Forum (GGF), http://forge.gridforum.org/projects/ogsa-wg.

[57] Barry Wilkinson,"Grid Computing Trehniques and Applications", CRC Press, ISBN: 2010.

[58] David S.Linthiam,"Cloud Computing and SOA Convergence in your Enbterprise",PEARSON, ISBN:978-81-317-3358-5, 2010.

[59] Geore Reesa,"cloud Applicatin Architetures", SPD Publication, 2015, ISBN-13:978-81-8404-714-1.

[60] Michael Miller, "cloud Computing Web-Based Applications That Change The Way You Work and Collborate online",PERSON, 2009, ISBN: 978-81-317-2533-7.

[61] C. C. T. Mark, D. Niyato, and T. Chen-Khong, "Evolutionary optimalvirtual machine placement and demand forecaster for cloud computing,"in I.C. on Adv. Info. Net. and Applications – AINA. IEEE, 2011

[62] L. Wang, J. Xu, M. Zhao, and J. Fortes, "Adaptive virtual resourcemanagement with fuzzy model predictive control," in I.C. on Autonomic Computing – ICAC. ACM, 2011.
[63] Barrie Sosinsky,"Cloud Computing Bible",WILEY published, 2011, ISBN:978-81-265-2980-3.
[64] Cong Wang, Kui Ren, Shucheng Yu and Wenjing Lou "Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing", INFOCOM, 2010 Proceedings IEEE,march 2010.
[65] E. Caron, F. Desprez, D.Loureiro, and A. Muresan, "Cloud computing resource management through a grid middleware: A case study with DIET and eucalyptus," in Cloud Computing, 2009. CLOUD'09. IEEE International Conference on, 2009, pp. 151-154.
[66] John W.Ritting house, James F.Ransome,"Cloud Computing Implementations Management and Security", 2010, ISBN: 978-1-4398-0680-7.
[67] Anthony,TobyJ.VelteRobortElsePeter,"Cloud ComputingAPracticalApproach",TataMcGraw–HillEdition,2010,ISBN:978-0-07-068351-8 ICAC.,ACM, 2011.
[68] A. Gautaom Shroff,"Enterprise Cloud Computing Technology, Architecturte, Applicationss",Cambridge University Press, 2010, ISBN-13:978-1-107-64889-0.
[69] Stephen R.Smoot, Nam K.Tom,"Private CloudComputingConsolidation,VitualkizationandServieOrientedInfrastracture",ELSEVIER, and MK Publication, 2012, ISBN:97893-81269-26-8.
[70] Fran Berman, Geoffrey C.Fox,Anthony J.G.Hey,"Grid Computing Making The Global Infrastructure A Reality",WILEY Publication, Feb 2010.
[71] Vladimir Silva,"Grid Computing For Developers",Dreamtech Publication,2006, ISBN:81-7722-682-7.
[72] Ahmar Abbas,"Grid Compouting:A practical GuidetoTechnologyandApplications",Firewal Media Publicatins,2010, ISBN:81-7008-626-4.
[73] Syed A.Ahson, Mohammad Ilyas,"Cloud Computing and Software Services",CRC Press,May 2017, ISBN:978-1-4398-0315-8.
[74] Bhushan Jadhav, Sonali Jadhav,"Grid and Cloud Compouting",Technical Publications, June 2017, ISBN:978-93-332-1162-8.
[75] Amit Sahai, Brent Waters, "fuzzy identity-based encryption" ,CCS '08 Proceedings of the 15th ACM conference on Computer and communications security 2008.
[76] Carlos Gutierrez, Eduardo Fernandez-Medina, Mario Piattini,"Web Sersdvices Security Development and Architecture Theoretical and Practical Issues", Information Science Reference, 2010, ISBN:978-1-60566-950-2.
[77] Douglas C Schmidt, Frank Buschmann," Patterns, Framework and Middleware: their synergistic relationships", pp .1-11
[78] Tomasz Haupt, Marlon E Pierce, "Distributed Object-based Grid Computing Environments", in Book Grid Computing: making the global infrastructure a reality, Fran Berman, pp 713-728
[79] C.S.R. Prabhu, "Crid and Cluser Computing", PHI, INBN: 978-81-203-3428-1.

P.Nagamani "Security Design Framework For Data Management And Distribution For Middleware In Grid Extended Cloud Computing "International Journal of Engineering Research and Applications (IJERA) , vol. 8, no.5, 2018, pp.57-76