ISSN 2278-3091
Volume 9, No.1, January – February 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse73912020.pdf
https://doi.org/10.30534/ijatcse/2020/73912020

# Enhancing Data Security under Multi-Tenancy within Open Stack

**M Trinath Basu[1], JKR Sastry[2]**
[1]Koneru Lakshmaiah Education Foundation, Vaddeswaram, India, miriiyala68@kluniversity.in
[2]Koneru Lakshmaiah Education Foundation, Vaddeswaram, India, drsastry@kluniversity.in

## ABSTRACT

Many businesses all over the world are turning their IT solutions through the use of Cloud computing systems hosted by third-party service providers using pay by use concept. However, businesses have to implement their IT systems fitting into the way the infrastructure used dictated by the cloud computing system, and there is either little or none provision for architecting the infrastructure as per the requirements of the business solutions. Open Source Software like OpenStack effectively used for building customized Cloud computing systems. But this kind of platform suffers from acute security threats due to the existence of several vulnerabilities. Data security, as such, is left to the users. Open Stack at best implemented a certain level of security enforcement in terms of user authentication and access control.

Data security was the primary concern when Multi-tenancy implemented through the provision of access to multiple users; the Applications installed on VM's or access provision is made for the same applications installed on VMs to Multiple users. In this paper, an enhancement to OpenStack architecture presented through the addition of a new component for securing the data emanating from the user and stored within the cloud. Three data isolation techniques presented, which proved that even brute force methods could not reveal the secrecy when the data isolations techniques implemented within the open stack.

**Key words :** Cloud computing Systems, Authentication, Access Control, Data Security, Open Source Cloud Computing System

## 1. INTRODUCTION

### 1.1 Need for maintaining data security due to the requirement of Implementing Multi-Tenancy

Cloud computing provides several services to multiple tenants using the same physical machine through the implementation of a concept called virtualization. Providing security and privacy to data owned by several tenants is a challenge when the resources connected to the same machine shared among several tenants. In cloud computing, the software marketed by the cloud computing service provider is made available to the user through the implementation of a service model called SaaS. Many users are allowed to use the same application and therefore give scope for encroachment into a data segment that is not related to some users. Customers cannot monitor or control the way the data is going to be dealt with by the service provider as the user has no idea of the infrastructure being used to store the data. Multi-tenancy implies that a set of users allowed to utilize the same application hosted by the service provider. A virtual machine (VM) is a kind of implementation of the software. The VM can be configured with an Operating system or any other program which runs on the Physical machine on which the VM provisioned. The software configured on the VM uses all the computing facilities that existed on the physical machines that include memory, storage, network, etc. Ever physical machine is loaded with OS, and then a software called Hypervisor installed for managing various resources connected to the machines that include memory, storage, software, etc. Hypervisor helps in sharing the resources among several applications that are loaded into virtual machines configured into the physical machine. The sharability of the resources provided by the Hypervisor such that no two applications or users conflict from each other. Users share various kinds of resources that include Memory, storage bandwidth, Virtual machines etc.

Several techniques are used within the hypervisor to implement effective sharing of the resources through the use of technologies such as Access Control, Virtual LANs, Virtual controllers for storage, and other resources. Cloud computing systems are being attacked through the use of different methods that include Brute-forcing, Probing networking, side-channel attacking etc. These kinds of attacks exploit the vulnerabilities and disturb the functioning of the applications or catch hold of the secrecy of the data or spoil the data itself. When several users are allowed to use the same application resident on a Virtual machine, there is a chance that one user can encroach into the storage area of others leading to attacking each other data. This kind of attack counter-attacked through the adaption of techniques that lead to data isolation. The issue of Multi-tenancy that allows several users to the same VM and the application running on the VM. Many software packages are being made available as a service to the customers by the cloud service providers on a sharable basis software packages installed on virtual machines and access to the software packages provided to several users.

As the users are resident on the same virtual machines sharing the software packages installed in the same VM, there is a possibility that a user invades into the resources of other users. Multi-tenancy helps to reduce the cost of processing as the same machine, and the users share other resources, but at the same time, there is an increase in risk. Isolation of critical resources that include memory, data, customizations, configuration files, self-administration, process management, tenant aware software version control, error tracking achieved such fill enforcement of security achieved.

Different kinds of models are used for implementing Multi-tenancy based on the sharing used. Some of the issues related to sharing the resources include either shared everything or shared-nothing or sharing some resources. The sharing that can be employed decided considering OS, Hardware, database, software, and files. Generally, access to the same database or sharing the tables designed into the database is required. Users are also given access to the configuration files so that the users can customize their way of using the software. In all the cases, there is a requirement of protecting the resources when access to the same given to multiple users.

Data is the main element that gets attacked when access to the same given to multiple users. The users will have the opportunity of attacking each other data. Isolation of data of each user is the essential thing when multi-tenancy implemented. The data isolation done in such a way that the owner of the data only will have access to the data and not to the others. Complete confidentiality of the data will be maintained. Managing the information effectively and efficiently will be the key issue to be taken into consideration for maintaining the confidentiality of the data. Many systems, rules, processes and regulations are to be built into the cloud computing system so that a security requirement of different users met. Many challenges met when one implements multi-tenancy within the cloud computing system. The users of a cloud computing system need several kinds of resources, provision for proper performance, scalability Availability of value-added applications, and, most importantly the need for privacy and security. The users also require the rules that allow them to make customizations that the users would like to make for implementing their own IT requirements.

The solution providers are developing many third party components for providing the tools to the users, which can help to implement access control, implementing business logic, workflows and user interfaces. Services providers have to consider many aspects of providing cloud computing services. Many issues that include data sharing, data backup, data restoration in case of data crashes, reducing the cost of operations, the time required for marketing the product, etc. have to be considered especially when the provider decides to implement the Multi-tenancy based services. The issue of implementing Multi-tenancy is much more complicated when the same performed under virtual machines.

A hypervisor like VMware required for implementing the virtualization of a physical machine. The hypervisor provides services related to resource sharing, scalability, VM Management and other coordinating required to effect virtualization. The hypervisor is like another program that runs on the Operating System to create an atmosphere of simulating a Physical machine as if it represents many machines. Database Management software installed on a Virtual machine and the same shared among several users. Virtual Machine must control the access to the database, confirm to SLA conditions while providing service to a database, assist customers so that the customers can create their triggers and stored procedures. The hypervisor must also communicate with other services for being able to back up the database, retrieve the database in case of disasters, undertake the upgrade of the software, implement security at the database level and also provide the support required to affect the legal issues.

The database installed on a virtual machine used in multiple ways. The database can be installed either in single or multiple instances modes. In the case of a single-instance method, the database created using a separate partition of data or through the use of various barriers. The database created in terms of multiple user spaces, each database space assigned to a specific user. Multiple partitions of the database can either be created using the storage area supported on a single machine or each partition created using storage space supported on several machines. Thus data isolation can be achieved using several instances, user spaces, data partitions, and multiple databases. The use of databases using different alternatives shown in Figure 1,

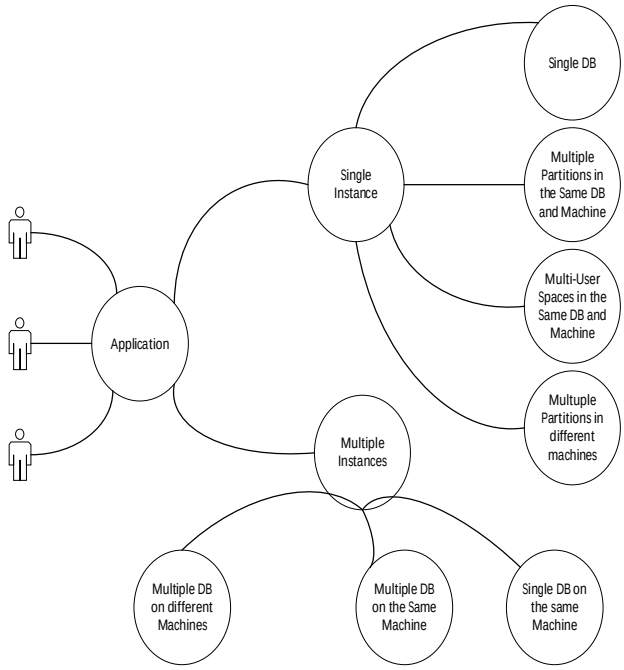


**Figure 1:** Data organization mechanisms under cloud Different strengths of enforcement of data security and data isolation achieved by using one of the methods shown in Figure-1.

### 1.2 Mechanisms for accessing the data through Native Database systems

The accessibility of a database provided through many database management mechanisms that include the following:

1. Allocation of separate databases resident on the same machine.
2. Several databases installed on different machines. Allocation of the segmented database.
3. Horizontally partitioned database made to be resident on the same machine.
4. Horizontally partitioned databases made to be resident on different machines.
5. Installing the shared database using several schemas
6. Establishing a database and schema in shared mode

A database with its associated schema can be allowed for access by several users. The sharing of the database usually implemented within an enterprise setup. However, this kind of scalability of a single database leads to unauthorized access to the data. The grant permission adopted by the users limits the access by the users. If access permissions exist, there could be data encroachments by the users. The administrators of the database will have access to the entire data losing the secrecy of the same. More mechanisms have to be adapted to ensure the confidentiality of the data. Multi-tenancy within SaaS achieved through the use of a database with data isolation achieved at the application layer.

Different level data isolation achieved using the data arrangement methods described above. Data isolation built into an application through access to the data is provided to the users. The security controls directed by the users at different levels incorporated into the database directly have to be bypassed by the service providers. The security, as such, must be provided by the database service providers through a separate application developed and implemented on the Virtual machines. It was a considerable risk to the users when complete data security management left to the cloud computing service providers. Multi-tenancy leads to high risk as many users are allowed to uses several users to use the same application installed on a single VM, which implies the same hardware on the physical machine. Both the Victim and the attackers run on the same computer. Traditional methods implemented within the application or database management software cannot mitigate the risk caused by the attackers running on the same VM and using the same application.

The attacker and the Victim can be situated in three different ways. In method one, the attacker and the Victim are simply internet users. Security in such a case implemented using traditional methods. In method two, both the Victim and the attacker situated on different VMs, which are located either on the same Physical machine or multiple virtual machines. Multitenancy arises when a user uses the same VM and access the same application running on the VM. Separate security mechanisms implemented to secure user applications or databases. The network security provisions cannot protect the data processed through the application on the same physical machine or virtual machine. There is a need to find the methods which can be implemented and enforced within the native database for affecting the data security under a multi-tenancy environment implemented through OpenStack Cloud computing System

### 1.3 Implementing data security within OpenStack-issues for traditional data access services through TROVE component

OpenStack data is managed in 4 different ways. VM Images are stored and implemented through the “GLANCE” component. In OpenStack different objects, such as documents, stored and managed through the “SWIFT” component. Traditional processing in terms of volumes, directories, files, etc. supported through the “CINDER” components. Database oriented management system supported through a segment called “TROVE.” User applications installed within VM, and given with multi-user access, use standard database management systems for storing and manipulating the data.

In OpenStack, traditional Database implementations supported through the “TROVE” component. Any user who wants to store the data as a relational database can use this service; Trove provides Database as a Service for OpenStack. TROVE designed to run entirely on OpenStack allowing users to quickly and efficiently utilize the features of a relational database without the burden of handling complex administrative tasks. Cloud users and database administrators can provision and manage multiple database instances as needed. Initially, the service will focus on providing resource isolation at high performance while automating complex administrative tasks, including deployment, configuration, patching, backups, restores, and monitoring. The architecture of the TROVE module shown in Figure 2.

The module TROVE built using four components includes “Trove-API” that provides the interface to the VM based applications through Restful API that supports JSON and XML to provision and manage Trove instances. The component “Trove-Task Manager” helps in providing database instances, managing the database instances, and also carries the required database operation. The SQL server used as a database engine in the backend. The component “Trove-guest agent” as service runs within the guest instance, responsible for managing and performing operations within the Database itself. The Guest agent listens for RPC messages through the message us and completes the requested

transaction. ‘Trove-conductor” is a service that runs on the host, responsible for receiving messages from guest instances to update information on the host.

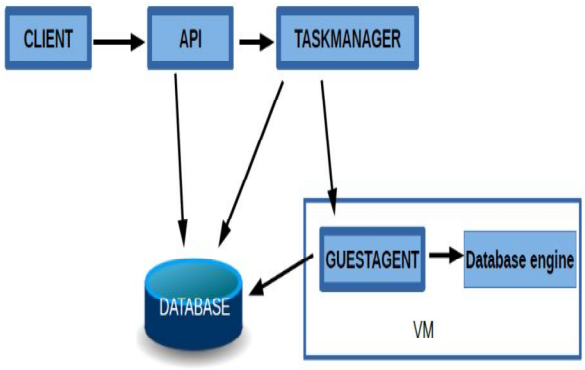


**Figure 2 :**TROVE Architecture

**Security issue**

TROVE is not designed with built-in security features to protect the data. KEYSTONE module provides access to the trove through the process of identity services. Most cloud providers do not encrypt data before saving it to storage. OpenStack does not provide any data encryption at all; thus, users would need to encrypt their data before uploading it and manage their encryption keys themselves.

### 1.4 Problem Definition

OpenStack has provisions to deal with three different types of storage, which include object storage, Image storage, and Block Storage. Access to the data implemented using user authentication and access control and providing the ability to the user for making operations through URL based entry points or through making calls to the services through the use of API supported by the concerned services. Block storage used for creating and storing databases through OpenStack based component "TROVE," which uses SQL Server as the Native Database. TROVE provides API to make different kinds of Data based related calls using which data manipulated. TROVE, as such, has no provision to maintain the confidentiality of the user data. To the maximum, access control restrictions applied. Trove has no support in dealing with data protection required under the Multi-Tenancy situation. The trove component does not provide any support for data security.

In this paper, architecture is proposed that includes the use of TROVE (OpenStack) for data management and enforcement of methods that ensure data security under Multi-tenancy of the applications installed by the user within Virtual machines. Three Algorithms have been proposed that ensure data security when data is used under Multi-Tenancy and operated through traditional database engines such as SQL server and using the TROVE component of OpenStack that provides block storage services to the end-user.

Many issues arise when data related to different users stored in the same database — the same set of cryptographic algorithms used for securing the data of multiple users. Physical separation of data or cryptographic algorithms, when not used leads to substantial risk. There is also a significant risk when encrypted data stored in the database is retrieved and decrypted for carrying with processing. The tasks that process the data can be attacked by an attacker task running on the same machine. There should be complete and composited security systems that ensure security at every stage of data storage and processing.

In this paper, three methods presented, which are implemented within OpenStack, considering several approaches that include the use of multiple instances, multiple user spaces, and multiple partitions. The plans provide a high level of data security and data isolation within an OpenStack system

## 2. RELATED WORK

SaaS is a kind of delivery model implemented by cloud computing service providers for making available software as a service. Customers access the application through the Internet. The service provider centrally hosts the software and related data. The user is not concerned with the kind of underlying infrastructure used by the application [1]. DPET (data Partition encryption techniques is one such method)[2]. In this method, each record is encrypted twice before storing the same in a portion allocated to the tenant. The entire database portioned (user space), and one portion allocated to one tenant only. A scheme is used for partitioning and allocating the partition to a specific tenant. The record is encrypted using the public, and a private key is known to both the tenant and CSP (Cloud service provider). The kind of encryption algorithm to be used is randomly selected. First, the record is encrypted by the tenant using the public key and then encrypted by the CSP using their public key. The private key of the tenant used at the time of decryption. The key pair to be used for each of the tenants is different and the same stored in the data segment related to the tenant concerned. The DEPT algorithm, while provides a particular level of security, the data processed within the server can be still be attacked by the co-resident users due to lack of traffic and bandwidth isolation.

An attack model presented based on a threat model that takes advantage of the Multi-Tenancy situation presented by [3]. Mitigating the attacking is the best course of action. One can know the information related to resource allocation, resource utilization, and access from the logs maintained by the clouds. The scanning of the logs and applying brute force methods the details of locations where data is stored could be known and, therefore, can be attacked. [4] Have shown the kind of issues addressed when multi-tenancy implemented in the IaaS layer. When new hardware added to increase performance, it sometimes leads to many of the security issues as well. The authors have presented a model using which the performance of a cloud computed. Security of the data stored on the cloud computing system achieved through implementing access control systems considering both authentication and authorizations; they have presented a mechanism to encrypt the data based on the location of the user and geo location of

the data where it is stored. [5] Have presented a comparison of the attribute-based encryption (AES) of the data to be stored in the cloud. However, the methods aim towards achieving the access controlling of the data than dealing with issues related to multi-tenancy. Data privacy and security of the data stored in the cloud achieved through implementing access control mechanisms.

A comparison of currently existing AES-Based schemes of data access control has been presented [6]. A list of unsolved problems that arise when AES used is enumerated and presented. Even though the AES based current existing control schemes could satisfy the requirements of data access control for cloud storage, there are still problems such as revocation of the user, reduction of the computational effort, implementation of the hierarchical structure of the user, etc. There are many problems such as revocation of the users, computational efficiency, the hierarchical structure of the users, etc., which are related to securing the cloud storage while attribute-based encryption could solve many access control related issues concerning cloud storage.

An analysis of data storage in cloud computing and the kind of security enforcement built into a cloud computing system is presented. They have emphasized that the central concept is to provide integrity to the cloud storage area with distinct data models and security algorithms. They have presented cloud data storage architecture along with the cloud data models. Manjinder Singh et al. [7] have emphasized that one has to ensure data integrity to the cloud storage through the use of different data models and security algorithms. They have presented an architecture that includes the security models within the cloud computing system. They have introduced a modified RSA based algorithm provided with a different key generation and decryption system for ensuring the cloud storage security.

The challenges that one has to face in providing security within a cloud computing system presented in detail by Katie Wood et al. [8]. They have explicitly focused on cloud deployment and data storage, in particular, relation to privacy concerns due to multi-tenancy. Katie Wood et al.[8] have presented a series of challenges that one must face when security to the cloud storage is needed. They have concentrated mainly on securing the cloud storage considering the multi-tenancy implemented through SaaS. The main issue addressed when multi-tenancy used is to protect the application and the hardware on which the application deployed as the clients are allowed to share both the elements. Multi-tenancy thus possesses many challenges to secure and preserve the privacy of the data owned by different users. Many methods presented in the literature aims at isolating data storage, allocating separate data storage for each tenant, etc. [9]. Each process releases a different security issue altogether that involves the use of different types of encryption techniques. Data Isolation is a critical issue addressed when it comes to multi-tenancy. Effective data management systems implemented to control access to the data by multiple users through access to the same application. Appropriate and extensive privacy and security to the data achieved. The public clouds, as such, can be attacked through several means as no network isolation is implemented [10]. Users who are attackers can launch attacks on co-resident users as no traffic or bandwidth isolation achieved as the multi-tenancy is an issue that is implemented within a single server [11]

Encryption algorithms are quite frequently employed to secure user data. The encryption algorithms used to transform the users' critical data so that the data made to be inaccessible to unauthorized users even in the situations of availability of such data to unauthorized users. Access control is one of the approaches that enforced to prevent unauthorized access to data. Access control implemented for controlling the access while the users are in multi-tenancy mode is not an effective method as the access control as it merely achieved through using IDs[12]. The service provider can provide an interface within the application using which the users can configure the application for imposing some security constraints. The security enforcement externalized without imposing any load on the application Mohamed Almorsy et al. [13].

Cloud computing architecture must include various issues related to the enforcement of security issues. The very first attempt to incorporate security into cloud computing infrastructure attempted by Kamara et al. [14]. They have covered both consumer and enterprise scenarios, and they have used nonstandard encryption algorithms such as searchable encryption and attribute encryption. The algorithm presented by Zarandioon et al. [15] is designed using user attributes and their signature for securing data presented by Zarandioon et al. [15]. The algorithm included a protocol call K2C (Key to Cloud - user-centric privacy-preserving cryptographic access control protocol. The end users can securely share, manage, and store their data in the cloud computing infrastructure, which is unstructured. Several data security models discussed in the literature but quite limited securing the data while the data moves over the network [16] [17][18][19][20][21][22].

## 3. INVESTIGATIONS AND FINDINGS

### 3.1 Architectural modification to Trove Implementation architecture within OpenStack

To the standard OpenStack Architecture, an additional component added for providing the data security to protect the data when several applications placed in a VM, and several users are given access to the applications. Addition of an additional component "Data Security" needed for achieving data Isolation to be effected among the users accessing several applications running on the same Virtual Machine. Figure 3 shows the Extended Architecture. The data security components implement three different sub-components, each responsible for performing data security using one of the approaches that include Multi-Instance, Multi-User Spaces, or Multiple database segments. The sub-components interact with TROVE for carrying database operations through calling Restful-API.

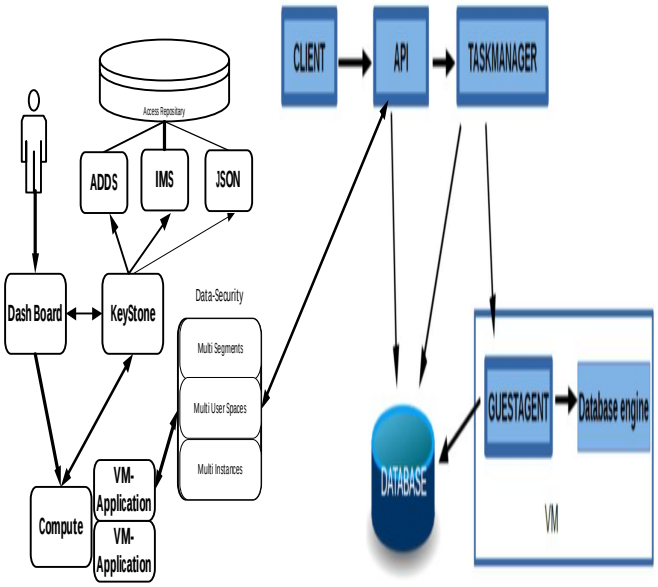


**Figure 3 Extending OpenStack for implementing Data Security under the implementation of Multi-Tenancy**

Keystone provides the Authentication of the users. Users' requests for a VM by making service requests to the "Compute" component using the authentication tokens received from Keystone. The Compute Node verifies the credential of the user with the keystone module. On receiving the valid credentials, the compute node will provide the logical port address of the application to the user, and the user from then on words start interacting with the application. In a Multi-Tenancy situation, many users given access to several applications running on the same VM, and sometimes several users are given access to the same application through several threads leading to serious security issues. One user may corrupt the data of the other. When a user wants to carry database related operations, a program contained in the data security module initiated and executed, which in turn undertakes required security operations before the actual Database operation carried through making a call to TROVE supported API.

### 3.2 Data Isolation through Data Partitioning and placing data segments in different Machines.

Many approaches presented in the literature for securing the data in a single database in which the data related to different customers stored and processed by a single application. All the approaches suffer from one kind of risk are the other. The data stored in the database can be double encrypted by the user and then the service provider to guarantee the confidentiality from both perspectives. Later, in that case, the way the encryption algorithms are selected or the way the keys generated is the most crucial aspect of securing the data stored in the database.

Data isolation is the most significant aspect of securing the data. The data as long as it is in a single database will be insecure at least at the physical storage level — physical data isolation achieved by distributing the data into different storage areas connected to different physical machines. The database created considering physical data storage situated in different machines. Many techniques used to segregate the data such that a very high level of data isolation achieved.

There are many ways to achieve data isolation within a given repository. The merits and demerits of every mechanism, are weighed and the appropriate decision taken to choose a machine that is most suitable for a given situation. The database portioned horizontally, and each partition situated on the storage located in different machines. Each partition allotted to one single user. Since the data of different users situated in various machines, data isolation achieved thereby, there is no chance of one user encroaching into the storage area of some other user. The arrangement of such an implementation shown in Figure 4.

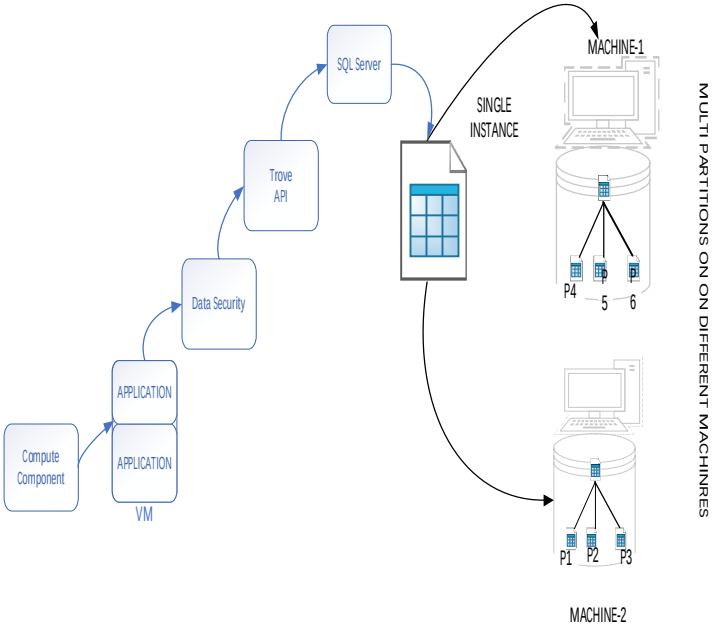


**Figure 4:**Multi-portioning the database

Each of the partitions recognized by the IP address of the Machine where it is situated and the location where the partition located.

Every user has an authentication token in which the user ID stored besides containing the user access rights. The user is given access to the application through a Logical port Number. When related data service is required, the application requests the data security component, which converts the data as per a chosen algorithm and calls the Trove API for carrying the level of database needed operation. Trove API requests are made such that a separate partition maintained for each user request where the partitions related to the concurrent users placed in different Machines.

The Trove API returns the IP address of the machine and the location where the partition created to the data Security module. The user can encrypt his data using its public key. The public key is user-generated on the user side. The

algorithm to be used by the user for undertaking the encryption can be fetched dynamically at the user side based on his ID. The key and the encryption algorithm used to encrypt the data on the user side before transmitting to the user application for storing the same on the user-related Partition. The User request is forwarded to the Data Security module by the Application so that data security module lookups for an algorithm based on the IP address of the physical machine and the location of the partition within the physical machine, using which an encryption algorithm and a key generated and the same used for encrypting data on the cloud side. Thus the data protection is implemented and privacy maintained on both the ends of the client and CSP. The decryption of the data, however, undertaken on the client-side using the private key of the user[23].

**Algorithm**

**Initialization process**

1. The client will request for partitioning the database. The request sent to the Hypervisor.
2. The hypervisor gets the details of the partition in terms of the IP address and the location of the separation from the guiding application software. A different IP address chosen every time a request initiated.
3. The key and the encryption algorithm used on the cloud side are generated based on the TCP/IP address and the location of the partition.
4. The user ID, Key, and the encryption algorithm are stored along with user ID within the cloud.

**Client-side process**

5. Tenant 'Ci' generates a large Prime Cp from his credentials, generally using his ID and sent to Cloud Service Provider.
6. Tenant Ci computes N=2*Cp
7. Tenant Ci generates Cyclic group ZN* of order Ø(N)(Euler Quotient function) .
8. A subgroup ZØ(N)* subset of ZN* of order Ø(Ø(N)) generated by Ci with generator g ∈ Zn*
9. Tenant Ci randomly picks up two private keys Tq and Cr ∈ ZN* Cq≡ gk1 mod N and Cr≡ gk2 mod N where k1,k2 ∈ZØ(N)* where g is a generator for ZN*
10. Tenant Ci computes N= Cq* Cr
11. Ci chooses 'e' such that gcd(e, Ø(N))=1.
12. Ci determined' such that ed≡ 1 mod Ø(N)
13. Tenant Ci computes

I. CPr = e.rst such that e.rst ≡ 1 mod Ø(N) and CPb =d.rsd such that d.rsd ≡ 1 mod

II. where CPr: Tenant Private Key,

III. CPb: Tenant public key

IV. Public key <N,CPb>

V. Private key <CPr,d,e>

VI. Tenant Ci encrypts the data of each record R (ER) ER= Re mod n

VII. Tenant Ci sends ERj to CSP to store in its Partition Pi.

**Processing on the Cloud side within the data security Module**

14. DSM fetches the related Encryption algorithm and the key with the help of user ID.
15. The data record received from the client is encrypted again using the key and encryption algorithm EER = ERTPb mod n
16. Fetch the partition details PT of the client using the lookup table with the help of user ID
17. DSM stores ER in partition Pi of Ci

Data Retrieval process

18. Tenant encrypts the Primary data using his key and algorithm and sends the same to the cloud along with his ID.
19. DSM fetches the record from the partition related to the client using the encrypted key data, which is further encrypted using the client's encryption algorithm and the key generated and stored in the lookup table on the CSP side.

The details of the partition related to the client stored within the Application software or the hypervisor.

20. The queried data is sent to the client. It should be noted that no decryption done on the cloud side. Querying done using the encrypted key values only.
21. After receiving Tenant, Ci computes R = EERrst mod N to obtain the original Record.
22. If Tenant Ci does not get Record R from the above data, then Ci assumes R is modified by CSP or intruder, so R discarded and requests for the current record.

**3.3 Data Isolation through multiple user spaces**

Within a database, several user spaces created, each allocated to a specific user. User spaces within a database lead to data isolation within the same database. It is like segmenting the database and attaching each segment to a different user.

Database engines have a concept of creating a user-space within a database and creating user-defined data tables within the allocated segment. The complete control of data accesses implemented within the DBMS software that serves to prevent attacking the database by multiple users registered within the same database. Figure 5 shows the architecture. But the database is still prone to attacking through stored procedures and triggers and by the administrators.

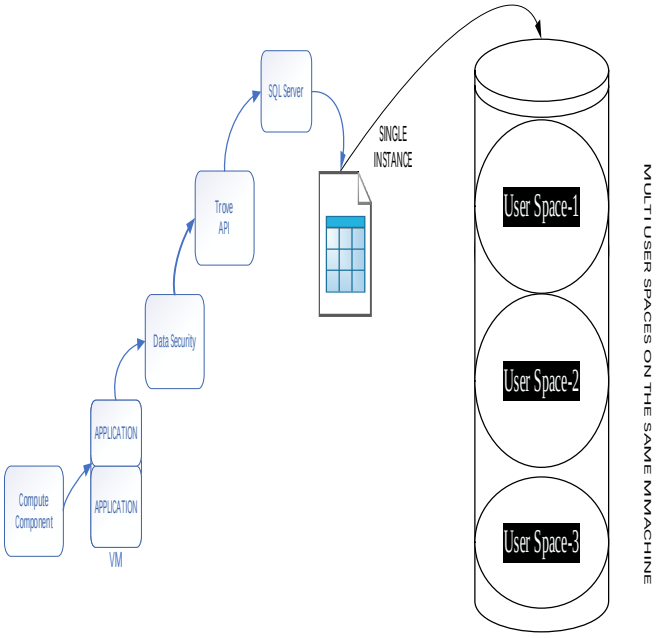


**Figure 5:** Multi user spaces segmented database

Each of the user space recognized by the name of the user-space within the database. Every user identified by the cloud computing system using some ID that is assigned by the cloud computing system. When a user requests the application for the allocation of space within the database, the application will require the Data Security module to allocate user space the name dynamically generated. The Data security software and the user exchange their public keys, and both have paired private keys with their respective public keys. The Data Security module dynamically selects an encryption algorithm based on the name of the user space. A hash developed using the name, and the hash is indexed into an encryption algorithm using a lookup table. Similarly, on the User side, an algorithm for encryption is chosen based on either a lookup table or dynamically selected based on the combination of the public key pair of the user and the Data security module. The method is based on double encryption both on the user side and the Data Security Module side, and the data is decrypted only on the user side.

The key and the encryption algorithm used to encrypt the data on the user side before transmitting to the Application based on the logical port number provided by the Data security module exchanged when the user makes the first request. The data security system uses its public key, and the encryption algorithm selected based on the name of the user space and encrypts the data before writing to the concerned user space. These methods ensure the isolation at the application level and data level. Within the Application level, a thread created for each of the users for transporting the data, either way, thereby ensuring the application layer isolation. Thus the data protection is implemented and privacy maintained on both the ends of the client and the data security module. The description of the data, however, can be undertaken on the client-side using the private key of the user[24].

**Algorithm**

**Initialization process**

1. Client requests for user space in the database. The request sent to the Hypervisor
2. Database returns the name of the user space to the application running on the VM
3. A value is generated based on the name of the user space, and the same indexed into the kind of algorithm used for data encryption on the user side. A different encryption algorithm is selected every time hash indexing done
4. The generation of Public key pair done on the CSP side and the public key of the User exchanged
5. The user ID, Key, and the encryption algorithm stored along with user ID within the cloud

**Client-side process**

1. Tenant 'Ci' generates a large Prime Cp from his credentials, generally using his ID and sent to Cloud Service Provider.
2. Tenant Ci computes N=2*Cp
3. Tenant Ci generates Cyclic group ZN* of order Ø(N)(Euler Quotient function)
4. A subgroup ZØ(N)* is a subset of ZN* of the order Ø(Ø(N)) generated by Ci with generator g ∈ Zn*
5. Tenant Ci randomly picks up two private keys Tq and Cr ∈ ZN* Cq≡ gk1 mod N and Cr≡ gk2 mod N where k1,k2 ∈ZØ(N)* where g is a generator for ZN*
10. Tenant Ci computes N= Cq* Cr
11. Ci chooses 'e' such that gcd(e, Ø(N))=1
12. Ci determined' such that ed≡ 1 mod Ø(N)
13. Tenant Ci computes

CPr = e.rst such that e.rst ≡ 1 mod Ø(N) and

CPb =d.rsd such that d.rsd ≡ 1 mod

whereCPr: Tenant Private Key,
CPb: Tenant public key
Public key <N,CPb>
Private key <CPr,d,e>

10. Tenant Ci encrypts the data of each record R (ER)

ER= Re mod n

11. Tenant Ci sends ERj to CSP to store in its Partition Pi.

**Processing @Data Security Module**

12. DSM fetches the related Encryption algorithm and the key with the help of user ID
13. DSM also fetches the user space related to the user using the User ID
14. The data record received from the client is encrypted again using the key and encryption algorithm

EER = ERTPb mod n

15. DSM stores ER in userspace Pi of Ci

**Data storage Retrieval process**

16. Tenant encrypts the Primary data using his key and algorithm and sends the same to the cloud along with his ID
17. DSM receives the primary data and encrypts the same using the key and dynamically selected algorithm, which connected with the name of the user space.
18. DSM fetches the requested record from the related user-space using encrypted primary key data.
19. The queried data is sent to the client, and no decryption is done on the cloud side. Querying is done using the encrypted key values only.
20. After receiving Tenant, Ci computes R = EERrst mod N to obtain the original Record.
21. If Tenant Ci does not get Record R from the above data, then Ci assumes that R is modified by DSM or intruder, so R discarded and requests for the current record.

### 3.4 Data isolation through Multi DB Instances

Many approaches presented in the literature for securing the data in a single database in which the data related to different customers have been stored and processed by a single application. All the approaches used for dealing with the data suffer from one kind of risk are others. The data stored in the database can be double encrypted by the user and then the service provider to guarantee the confidentiality from both perspectives. Then, in that case, the way the encryption algorithms are selected or the way the keys generated is the most crucial aspect of securing the data stored in the database. Data isolation is the most significant aspect of securing the data. The data, as long as it is in a single database, will be insecure at least at the physical storage level. Physical data isolation carried by distributing the data into different storage areas connected to different physical machines. The database is assigned to physical data storage situated indifferent machines. Many techniques are in use to segregate the data such that a very high level of data isolation achieved.

Both the application level and data level isolation, when achieved, will make the entire multi-tenancy system fully secured from the perspective of all the users. Application-level security is achieved by using several instances of the DBMS software and allocation of each instance to each user. Making available multiple instances of the DBMS software is quite expensive but will provide a high level of security. Creating several databases, each operated through its corresponding DBMS instance, gives a very high level of data isolation. The isolation level will further increase when the databases are situated in several physical machines. The databases, in this case, are physically distributed. The arrangement of multiple instances of DBMS software shown in Figure 6.

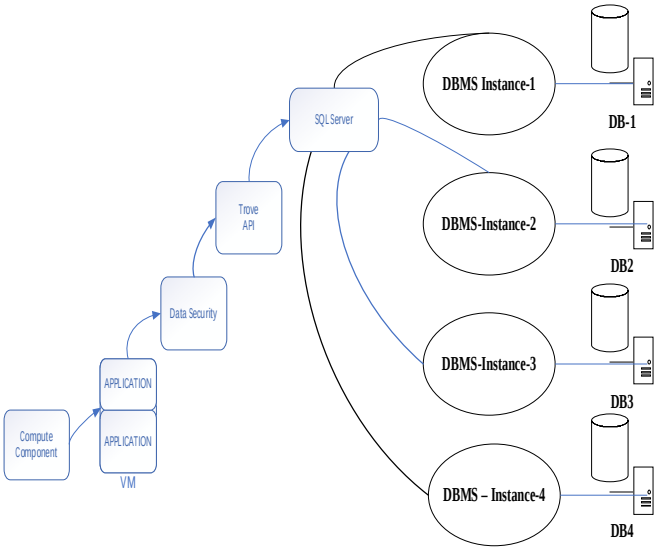


**Figure 6:** Data Security through Multi-Instance Database

Each database recognized by the IP Number of Physical machines on which DB created. Every user identified by the cloud computing system using some ID that is assigned by the cloud computing system. When a user requests a VM to run a SaaS service, the hypervisor shall require the application to create a database on different servers on which no other database created. The application server is also guided to create a spate instance of DBMS software and then attach a newly created database on a different machine. Thus the Instance and the database are bonded tightly. Thus the user, instance, and DB are completely inter-linked tightly.

An encryption algorithm dynamically selected on the Data Security Module side based on the Name of the instance and the IP address of the physical machine on which the DB created. A hash is developed using the name of the instance, and IP address and the hash is indexed into encryption algorithm using a lookup table. Similarly, on the User side, an algorithm for encryption is chosen based on either a lookup table or dynamically selected based on the combination of the public key pair of the user and the data security module. The method is based on double encryption both on the user side and the Data security Module side, and the data is decrypted only on the user side.

The key and the encryption algorithm used to encrypt the data on the user side before it is transmitted to the application running on VM based on the logical port number provided by the Cloud computing system at the time of registration of the user. The cloud computing system uses its public key, and the encryption algorithm selected based on the name of the user space and encrypts the data before writing to the concerned user space. These methods ensure the isolation at the application level and data level. Within the Application level, a thread is created for each of the users for transporting the data, either way, thereby ensuring the application layer isolation. Thus the data protection is implemented and privacy maintained on both the ends of the client and the Data security module. The description of the data, however, can be

undertaken on the client-side using the private key of the user[25].

Algorithm

Initialization process

1. Client request for an instance and databases sent to the Hypervisor
2. The application running on the VM will create an instance of the DBMS, locates a physical machine, and then establishes the Database on the chosen Physical computer. A bonding established between the Application, DBMS instance and the Database
3. A hash value is generated based on the name of the instance and the IP number of the physical machine on which the DB created. The hash value indexed into the kind of Encryption algorithm used for data encryption on the CSP side. A different encryption algorithm is selected every time hash indexing done.
4. Public key pair generation CSP sided one, and the public key of the User exchanged
5. The user ID, Key, and the encryption algorithm stored along with user ID within the cloud

**Client-side process**

6. Tenant 'Ci' generates a large Prime Cp from his credentials, generally using his ID and sent to Cloud Service Provider.
7. Tenant Ci computes N=2*Cp
8. Tenant Ci generates Cyclic group ZN* of order Ø(N)(Euler Quotient function)
9. A subgroup ZØ(N)* the subset of ZN* of the order Ø(Ø(N)) generated by Ci with generator g ∈ Zn*
10. Tenant Ci randomly picks up two private keys Tq and Cr ∈ ZN* Cq≡ gk1 mod N and Cr≡ gk2 mod N where k1,k2 ∈ZØ(N)* where g is the generator for ZN*
10. Tenant Ci computes N= Cq* Cr
11. Ci chooses 'e' such that gcd(e, Ø(N))=1
12. Ci determined' such that ed≡ 1 mod Ø(N)
13. Tenant Ci computes

CPr = e.rst such that e.rst ≡ 1 mod Ø(N) and
CPb =d.rsd such that d.rsd ≡ 1 mod
whereCPr: Tenant Private Key,
CPb: Tenant public key
Public key <N,CPb>
Private key <CPr,d,e>

14. Tenant Ci encrypts the data of each record R (ER)

ER= Re mod n

15. Tenant Ci sends ERj to DSM to store in the databasePi.

Processing on the Data Security Module

16. DSM fetches the related Encryption algorithm and the key with the help of user ID
17. DSM also brings the database into which the data is written
18. The data record received from the client is encrypted again using the key and encryption algorithm

EER = ERTPb mod n

19. DSM stores ER in user space Pi of Ci

Data storage Retrieval process

20. Tenant encrypts the Primary data using his key and algorithm and sends the same to the cloud along with his ID
21. CSP receives the primary data and encrypts the same using the key and dynamically selected algorithm, which is connected with the instance name and location of the database.
22. CSP fetches the requested record from the related database and through the connected DBMS instance using encrypted primary key data.
23. The queried data is sent to the client, and no decryption done on the cloud side. Querying done using the encrypted key values only.
24. After receiving Tenant, Ci computes R = EERrst mod N to obtain the original Record.

If Tenant Ci does not get Record R from the above data, then Ci assumes CSP or intruder modifies r, so R discarded and requests for a fresh record.

## 4. Experimentation and results

The algorithmsrelating to Multi segments, multi instances, and Multi-user spacesimplemented within Open Stack and even the brute force method applied to access the data sitting on the database software side did not reveal the secrecy of the data stored in the database. It has not been possible even to locate the partition or gain a handle on the algorithm and the key used for undertaking the encryption on the OpenStack side.

## 5. CONCLUSION

When application software that uses a database provided as data service to multiple users through virtual machines, the issue of Multi-tenancy arises. The users can attack the data due to the reasons of multi-tenancy. One user can attack others as both the VM related to the clients are situated on the same server. Therefore it becomes necessary to implement methods Mechanisms that help in protecting the data when the same services provided to several users. In Open Stack, access to the services is limited to Authentication and Access control. The data, as such, is not secured and the responsibility of securing the data left to the users. Open stack deals with four kinds of storage that include Virtual Images, Objects, Block storage, and conventional database storage systems. The access control is limited to entry points and not to the actual data. Data isolation is the key to protect the data achieved through locations and the partitions of data related to a specific user on a different physical machine or in a completely different location on the same physical computer. More complex but simple to implement system achieved through double encryption carried on the customer and cloud side. The key and the encryption algorithm are selected dynamically using the reference data. Data isolation is the key

to protect the data achieved by maintaining several user spaces within the database. The method is simple as the mechanism to maintaining multiple databases is a built-in feature with the database management systems. Double encryption by both the user and CSP ensures using different keys and algorithms to ensure complete secrecy. Decryption is done on the user side, only reviving computational overhead on the CSP side. Data isolation is the key to protect the data achieved by creating a separate database on a different physical machine, thus making booth logical and physical isolation. Application-level separation is achieved through the creation of multiple database instances. The method is expensive that the number of DBMS instances created and several physical servers used for locating the databases. However, the technique is full, proves that it secured the data, and the application 100%. Double encryption by both the user and CSP ensures using different keys and algorithms to ensure complete secrecy. Decryption done the user side only reviving computational overhead on the CSP side.

**REFERENCES**


1. M.Saraswathi, T.Bhuvaneswari, Multitenancy in Cloud-based Software, as a Service Application, International Journal of Advanced Research in Computer Science and Software Engineering, Vol. l3,issue11, page 1-4,2013
2. K.Venkataramana, M Padmavathamma, Multi-Tenant Data Storage Security In Cloud Using Data Partition Encryption Technique International Journal of Scientific & Engineering Research, vol4,issue7,page1-5,2013.
3. Hussain Auahdali, AbdulazizAlbatli, Peter Garraghan, Paul Townend, Lydia Lau, Jie Xu, Multi-Tenancy in Cloud Computing, 8th International Symposium on Service-Oriented System Engineering(SOSE), Page1-9,2014 https://doi.org/10.1109/SOSE.2014.50
4. Bhawna Sehgal Er. JasbeerNarwal, An Analysis of Performance for Multi-Tenant Application through Cloud SIM, International Journal of Emerging Research in Management &Technology,vol4 issue6,page1-5,2015.
5. GoikarVandana T., JagdaleSupriya K., Parade Priya B., PawarSumedha D., IMPROVE SECURITY OF DATA ACCESS IN CLOUD COMPUTING USING LOCATION, IJCSMC, vol4 issue2, page1-10,2015.
6. Tengfei Li, Liang Hu, Yan Li, Jianfeng Chu, Hongtu Li, and Hongying Han, The Research and Prospect of Secure Data Access Control in Cloud Storage Environment, Journal of Communications, VOL 10,ISSUE 10,PAGE 1-7,2015.
7. Manjinder Singh, Charanjit Singh, Multi-tenancy security in cloud computing, international journal of engineering sciences & research technology,VOL. 4,ISSUE116,PAGE1-7,2017
8. Katie Wood and Mark Anderson, Understanding the complexity surrounding Multitenancy in cloud computing, Eighth IEEE International Conference on e-Business Engineering, 10.1109/ICEBE.2011.68,2011.
https://doi.org/10.1109/ICEBE.2011.68
9. http://www.gartner.com/id=2058722.
10. K. Wood, M. Anderson, Understanding the complexity surrounding multitenancy in cloud computing, Eighth IEEE International Conference on e-Business Engineering,VOL1,PAGE NO 119-124,2011
https://doi.org/10.1109/ICEBE.2011.68
11. Paul Feresten, Storage Multi-Tenancy for Cloud Computing, SNIA,2010.
12. W. Tsai, Q. Shao, Role-Based Access-Control Using Reference Ontology in Clouds, Tenth International Symposium on Autonomous Decentralized Systems, VOL 11,PAGE 121-128,2011.
https://doi.org/10.1109/ISADS.2011.21
13. Mohamed Almorsy, John Grundy, and Amani S. Ibrahim, TOSSMA: A Tenant-Oriented SaaS Security Management Architecture, IEEE Fifth International Conference on Cloud Computing, PAGE 1-9,2012.
https://doi.org/10.1109/CLOUD.2012.146
14. S.Kamara, Kristin Lauter, Cryptographic cloud storage, FC'10 Proceedings of the 14th international conference on Financial cryptography and data security, PAGE 136-149,2010
15. Jose M,AlcarazCalero, Nigel Edwards, Johannes Kirschnick, Lawrence Wil cock, and Mike Wray, Toward a multi-tenancy authorization system for cloud services, IEEE Security and Privacy, Page 48-55,2010
https://doi.org/10.1109/MSP.2010.194
16. M.Trinath Basu, Dr.JKRSastry, A full security included Cloud Computing Architecture, International Journal of Engineering & Technology, Volume 7, Issue 2.7, Page 807-812, 2018
17. M. TrinathBasu, JKRSastry, Improving the Open Stack Authentication system through federation with JASON Tokens, International Journal of Advanced Trends in Computer Science and Engineering, Volume 8, Issue 6, Pages 3596-3614,2019
https://doi.org/10.30534/ijatcse/2019/143862019
18. TrinathBasu, JKRSastry, Strengthening Authentication within Open Stack Cloud Computing System through Federation with ADDS System, International Journal of Emerging Trends in Engineering Research, Volume 8, No. 1, Page, 213-238, 2020
https://doi.org/10.30534/ijeter/2020/29812020
19. JKRSastry, M TrinathBasu, Multi-Factor Authentication through Integration with IMS System, International Journal of Emerging Trends in Engineering Research, Volume 8, No. 1, Page, 88-113, 2020
20. J. K. R. Sastry, K. Sai Abhigna, R. Samuel and D. B. K. Kamesh, Architectural models for fault tolerance within clouds at the infrastructure level, ARPN Journal of Engineering and Applied Sciences, VOL. 12, NO. 11, 2017, Pages 3463-3469
21. DBK Kamesh, JKRSastry, Ch. Devi Anusha, P. Padmini, G. Siva Anjaneyulu, Building Fault Tolerance within Clouds at Network Level, International Journal of Electrical and Computer Engineering (IJECE), Vol. 6, No. 4, pp. 1560~1569, 2016
https://doi.org/10.11591/ijece.v6i4.10676
22. S. L. SUSHMITHA, Dr. D. B. K. J.K. R. SASTRY, V. V. N. SRI RAVALI, Y.SAI KRISHNA REDDY, building fault tolerance within clouds for providing uninterrupted software as service, Journal of Theoretical and Applied Information Technology, Vol.88. No.1, Pages 65-76, 2016

23. JKRSastry, M TrinathBasu, Securing Multi-tenancy systems through user spaces defined within the database level, Jour of Adv Research in Dynamical & Control Systems, Volume 10, issue 7, Page 405-412, 2018
24. JKRSastry, M TrinathBasu, Securing Multi-tenancy systems through multi DB instances and multiple databases on different physical servers, International Journal of Electrical and Computer Engineering (IJECE), Volume 9, Issue 2, Pages 1385-1392, 2019.
https://doi.org/10.11591/ijece.v9i2.pp1385-1392
25. JKRSastry, M TrinathBasu, Securing SAAS service under cloud computing-based multi-tenancy systems, Indonesian Journal of Electrical Engineering and Computer Science, Volume 13, Issue 1, Page 65-71, 2019
https://doi.org/10.11591/ijeecs.v13.i1.pp65-71

.