# A Comprehensive Study on Multi Tenancy in SAAS Applications

Shruti Kanade
Dr B.A. M. University
Aurangabad, India

Ramesh Manza, PhD
Associate Professor
Department of CS and IT
Dr B.A.M. University
Aurangabad, India

## ABSTRACT
In the age of cloud computing or utility computing, multi-tenant applications mainly Software as a Service (SaaS) applications have been popularly acknowledged as the generation next of Internet applications. These applications allow various organizations or tenants to modify (customize) an application in a robust and reliable manner. However, this customization according to the user organization's perspective can be error-prone, so research is being carried out to develop SaaS applications based upon different frameworks, platforms and modelling approaches. Multi-tenancy is an architectural approach in which a single instance of a application serves more than one customer who are referred to as tenants.

In this paper, the researcher has made a broad survey to understand the various features of the issues concerned with multi-tenancy. The results of the study should be used to understand the pros and cons of these aspects as well as to identify the areas of future research.

## Keywords
Multi-tenancy, SAAS application, Cloud computing,

## 1. INTRODUCTION
Cloud computing [1] is a modern technique which is principally based on Grid, Utility computing, adopted by organizations and businesses alike to help increase profit margins by decreasing overall IT costs and provide clients with better implementation of services. Main service models include Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) also have Communication as a Service (CaaS) and Data Storage as a Service (DSaaS).

The various parameters of the cloud such as elasticity, metered usage, on-demand service, broad network access allow the services to be distributed or shared equally among users and organizations. The sharing of resources, infra-structure, bare hardware and mainly data account for multi-tenancy at all layers of the model.

The majority [1] of cloud service providers offer multi-tenancy to reduce incurring IT costs which translate into economic savings for the user.

Multi-tenancy is a well-known concept used to reduce total cost of ownership. In the multi-tenancy model, many tenant's data are stored in the same public cloud and are controlled and bifurcated according to the user through use of tagging of resources owned by the user.

The meaning of multi-tenancy framework has enlarged due to creation of new service models that take advantage of concepts like virtualization and remote access. A SaaS provider can run one instance of the common application on one instance of the database and provide application access to multiple tenants. The concept of multi-tenancy is such that a single instance of the software application can serve multiple clients.

Each customer is a tenant, who is given rights to change user interface or business rules as he requires. The tenant cannot customize application code. This is the basis of multi-tenancy as implemented at SaaS layer. With SaaS, clients are sourcing same application (SalesForce.com) meaning that data of various tenants is stored in same database and they share the same database tables.

However, multi-tenancy introduces uncommon security risks, which are yet to be considered in a serious manner. The risks of multi-tenancy is access of data or trace of operations by other tenant.

When it comes to security issues, risks impacted by multi-tenancy require to be addressed at all layers which include IaaS, PaaS and SaaS

## 2. MULTI-TENANCY IN SAAS APPLICATIONS
Multi-tenancy is one of the main ascepts in SaaS. It is an [3] architectural principle that makes it possible for SaaS application to serve more than one tenant using a single instance of service. Multi-tenancy occurs at the database layer here.

In this phenomenon, customers/ tenants share the same hardware resources, by using the single application and database instance which is shared, while remaining isolated from each other. Since the application code is stored at one place, it is easier not only to maintain and backup but also update application and data. Another essential advantage is low system requirements. It is unnecessary to have a dedicated server for each client [3]. Resource utilization is near optimal.

Like two sides of the same coin, multi-tenancy poses several challenges and difficulties as well. They are:

- Performance: Amount of resources distributed to each tenant may result in inefficient utilization of resources
- Scalability: Tenants from different geographical locations may use same application, which has an impact of scalability.
- Security: A security violation may result in exposing data of other tenants

These are the areas where research is conducted to mitigate the challenges. Multi-tenancy requires that application methods and database table access and store data from different tenant accounts which violates security. But if done in error free manner, benefit is cost savings. For simple web

applications, this concept is a solution, as a single developer develops it and can do it faster as well as scale it.

## 3. SECURITY ISSUES IN MULTI-TENANCY

Security violations in multi-tenancy are fundamental in every layer of cloud like IaaS, PaaS and SaaS. This occurs due to the fact that client's employ the same hardware to share and process information.

This introduces a number of risks in terms of compliance, security and privacy, as well as lack of user network isolation [1]. Data stored in such application is stored in the same database and accessed by tenants using different partitions. A tenant is any application – inside or outside the enterprise- that needs its own secure and mutually exclusive virtual computing environment.

This environment encompasses all or some layers of organization architecture from database to web forms. One of the highlighted issues while using this type of multiple services is ensuring data isolation. Managing data is very critical. All users require privacy. Multi-tenancy and lack of network isolation [1] among clients result in making public clouds susceptible to attacks.

Side channel attacks and interference among various domains create issues in distributed clouds. The researcher may suggest that current approach to access controls in clouds do not scale well to requirements posed by multi-tenancy as they are based on individual tenant's IDs. Multi-tenancy increases security risks naturally.

Cloud Service Providers are responsible for making sure that one tenant cannot break into another tenant's data and application. To provide secure multi-tenancy [1], a well-defined method to enable separation at all layers is required.

- Application layer: A precisely written multi-tenant application or multiple instances which are separately used can provide multi-tenancy here
- Server layer: This means separation of tenants and application instances on servers and controlling utilization and access to resources.
- Network Layer: IP security provides network encryption at IP layer for additional security
- Storage Layer: Entries at partition layer should be encrypted so that data is secured. Sharing data poses risks such as intellectual property infringement, data infringement, technical and industrial business sabotage

Current approaches to access control are infeasible to multi-tenancy requirements, as they are mostly accessed using individual user IDs. Cloud providers take responsibility for ensuring that one tenant cannot break into another tenant's data and application

## 4. WHY MULTI-TENANCY IS NECESSARY

The multi-tenancy architectural approach [4] is beneficial to both cloud service providers and the end users. Enterprise can customize an application. The application dynamically morphs for every single need. A multi-tenant application is similar to a large community hosted by cloud service provider. Development as well as maintenance costs are reduced

Tenants can operate in virtual isolation. The cloud service providers can get feedback on application operations which can be improved in common hardware and software so that entire community benefits at once. Multi-tenancy is appropriate for small startup enterprise or single developer [5] to develop software quickly at a reasonable price, and to create concepts for widgets and web applications. Applications on mobile and small business applications with less security requirements are based on innovation, response time and ease of use

## 5. HOW TO CHOOSE APPROPRIATE TENANCY MODEL

The tenancy model does not have an effect on the function of the application, but has an impact on the other factors of the solution. The assessment of the model is based on the following criteria:

- Scalability of the application which depends on number of tenants, storage per tenant, storage in aggregate and workload
- Tenant isolation includes data isolation and performance
- Per tenant cost which is same as database cost
- Development complexity includes changes to schema and changes to queries
- Operational complexity depends upon monitoring and managing performance, schema management, restoring a tenant and disaster recovery
- Customization depends upon ease of supporting tenant specific or tenant class specific schema customizations. If you divide the software into many small components, your choice of multi-tenancy model may change. You must carefully choose a model that best fits the need of your application.

## 6. ARCHITECTURAL ISSUES OF MULTI-TENANCY

Authors on multi-tenancy argue that this can become a concern cutting across a SaaS application. Generally a tenant-specific mechanism is used to authenticate tenants to access only their data. Further mechanism can be devised to facilitate tenants to customize specifically their own application, so that user gets a feeling that he is working in a dedicated environment.

Data isolation is a major requirement. But most of the current DBMS are incapable of dealing with multi-tenancy as a layer between Business logic and applications. There are two main kinds of architecture of multi-tenancy

- Complete Multi-tenant [9]: This is the purest form of multi-tenancy. This can be called "shared everything model". All resources like infra-structure, applications and database are shared among all tenants. It makes sure that all resources are used optimally. But this model requires a very complex architecture to implement multi-tenancy. It creates business risk. And it is difficult to customize as well as implement load balancing.
- Single Tenant database: Here, application layer is shared among all the tenants. Database is separated

by tenant. Force.com has adopted meta-driven architecture to achieve multi-tenancy.

Everything as seen by developers and application users alike is represented internally as meta-data. It is required that we maintain a balance between various factors effecting the application considered, like affinity and persistence, performance isolation, service differentiation and customization. Affinity [7] defines how requests of different users of the same tenant are bound to processing nodes of a single cluster

The SAP Business BYDesign solution was developed having an affine behavior in consideration due to high amount of caching. Other applications like those based on Google's App Engine are based on low tenant specific context.

This means that one tenant might be accessing several instances of multi-tenant application. Performance isolation exists if for tenants working on their quota, the SLAs are fulfilled. Weak-isolation is achieved if performance isolation is achieved within a number of requests sent by disruptive tenants.

Google's App Engine provides automatic horizontal elasticity for multi-tenancy application. However it is not tenant specific and does not ensure performance isolation even if elasticity is restricted. Qos differentiation is indirectly related to isolation concepts.

The ability to handle different tenant specific configurations regarding the User Interface, the systems functional/nonfunctional behavior and the services referenced is a key enabler for multi-tenant applications. A separate Meta -data manager provides customization information to adapt the application. An application allowing tenant specific code extensions provides most appropriate way to adapt it to customer's needs.

Google Docs4 provides office applications for private usage or small scale companies for customer specific customizing. Other SaaS providers like Salesforce provide a wide range of options including tenant specific code.

# 7. CONCLUSION

Cloud computing has recently emerged as a computing paradigm [10] for managing and delivering services over the Internet.

The concept of cloud computing is rapidly changing the scenario of IT and turning the promise of utility computing into a reality. Despite the benefits offered by cloud computing, its full potential has not been realized as current trends and technologies are not mature enough to do so. Many key challenges are yet to receive attention from research community.

So the researcher believes that this field offers a tremendous opportunity, particularly in multi-tenancy. In this paper, the researcher has surveyed the state-of-art of multi-tenancy in cloud, covering essential concepts, architectural design, prominent characteristics and research directions.

As this concept is still at an early stage, it is hoped that this paper will provide a better insight of design challenges of multi-tenancy in cloud and pave way for future research. Multi-tenancy [11] is often seen as a benefit to cloud service provider, but it comes with a security risk. When security comes first we need to eliminate this risk. But the valuable feature of VM allocation will not be possible in such a scenario. And it will cause performance degradation that is non-optimal level of utilization of resources.

# 8. REFERENCES

[1] K. Venkataramana, Prof M. Padmathavamma, "Multi-tenant Data Storage Security In Cloud Using Data Partition Encryption Technique", July 2013, International Journal of Scientific and Engineering Research

[2] Kaushal Jain, Bimal Kumar, Harshal Shah, "Degree of Multi-tenancy and its Database for cloud computing", ISSN-2321-9939

[3] Pallavi G B , Dr P Jayarekha, "Multi-Tenancy in SaaS : A comprehensive survey" July 2014, International Journal of Scientific and Engineering Research

[4] Shailesh Paliwal, "Cloud application Services SaaS-Multi Tenant Data Architecture", Infosys Technologies Ltd.

[5] https://www.ibm.com/developerworks/cloud/library/cl-multitenantcloud/index.html

[6] https://docs.microsoft.com/en-us/azure/sql-database/saas-tenancy-app-design-patterns

[7] Rouven Krebs, Christof Momm, and Samuel Kounev "Architectural concerns in Multi-tenant SaaS Applications" conference paper April 2012.

[8] W N T de Alwis, C D Gamage, "A Multi-Tenancy aware Architectural Framework for SaaS Application Development" 2013, The Institution of Engineers, Sri Lanka.

[9] https://blog.techcello.com/2013/04/multi-tenancy-architecture-models/

[10] Qi Zhang-Lu Cheng, Raouf Boutaba, "Cloud computing: state of art and research challenges" August 2108.

[11] Husaain aljihadli, Peter Garraghan, Abduliaj Albatli, Lydia M.S." Multi-Tenancy in Cloud Comptuing ", April 2014, http://www.researchgate.net

[12] Steve Bobrowski, "Optimal Multitenant Designs for Cloud Apps", 4th InternationalConference on Cloud Computing, 978-0-7695-4460-1/11, ISBN: 978-0-7695-4460-1 Digital Object Identifier 10.1109/CLOUD.2011.98 IEEE 2011.

[13] Article by Novalys, Access Control in Multi-Tenant Applications with VisualGuard, 2011.

[14] Article by Rajkumar R.S.,Access Control in Multi-tenant Applications , 2012.

[15] Cor-Paul Bezemer, Andy Zaidman, "Multi-Tenant SaaS Applications: MaintenanceDream or Nightmare?", Report TUD-SERG-2010-031, Delft University ofTechnology Software Engineering Research Group Technical Report Series 2010.

[16] PeterMell,TimothyGrance,http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf, The NIST definition of cloud computing, September 2011.

[17] Sanjeev Pippal,Vishnu Sharma, Shakti Mishra, D.S.Kushwaha, "An EfficientSchema Shared Approach for Cloud based Multitenant Database with Authentication and Authorization Framework", International Conference onP2P,Parallel,Grid,Cloud and Internet Computing, Barcelona, 26-28 Oct. 2011.