Secure Multi-tenant Design for Cloud Computing Environment

Pallavi G B, P Jayarekha

Abstract*: Multi-tenancy is one of the most important elements of cloud computing environment due to salability and economic benefit it offers for both cloud service provider (CSP) and end user. As resources are shared, it induces security and privacy issues and tenant dependent on cloud service provider to assign trustworthy cotenants. However, CSP maximize its resource utilization by allowing maximum co-tenancy irrespective of the behaviors of tenants. Subsequently techniques like Reputation management based design aid in identifying good and malicious tenants. However, the state-of-art model is not efficient when behavior of malicious tenant changes rapidly. Addressing the said research issue, a secure multi-tenant (SMT) design using a modified reputation management model is proposed which handles dynamic behavior of tenants efficiently for cloud computing environment. Experiments are conducted using TPCC benchmark shows significant performance in terms of latency and throughput by SMT over state-of-art model.

Index Terms*: Cloud computing, Multi-tenant, Resource scheduling, Reputation management Security.

I. INTRODUCTION

Cloud computing [1] offers numerous benefits to organizations to ensemble their dynamic service requirements for hosting online application services in the Internet. For example, cloud computing promotes rapid elasticity, broad network access, on-demand service, pay for use and high availability. Thus, various areas including healthcare, government and social networks are migrating their operational services to the cloud computing environment.

There are different kinds of cloud models such as private, public cloud, and hybrid cloud, as well as different types of deployment infrastructures such as Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [1]. Multitenancy is one of the key element of the cloud model.

Revised Manuscript Received on May 20, 2019.

Pallavi G B, Assistant Professor, Dept of CSE,BMSCE , Bengaluru, Karnataka

Dr. P Jayarekha, Associate Professor, Dept of ISE,BMSCE, Bengaluru, Karnataka

Multi-Tenancy enables sharing the same service instance among different tenants[2][3]. Multi-tenant data management system is one of the major applications provided by SaaS. Further, multi-tenant jobs on a set of shared computing worker can increase energy efficiency by 20% [4] and can reduce number of active server by 50% [5]. Although cloud offers several advantages, it also poses certain significant challenges. In present generation most of the information services are hosted online and accessing online services has been the daily routine of everyone's life.

For example, users could be performing online banking with complex functionalities. Or using them to read today's news with rich content. On the other hand, online services are constant targets for malicious intruders, finding way to exploit vulnerabilities to gain access to sensitive data for financial benefits or to gain access to systems which can then be utilized to further create more malicious activity on other systems. Clearly security and privacy challenges arise when there are multiple tenants with different policies and requirements using the cloud infrastructure. When sensitive data belonging to enterprises and individuals are stored and used through services in the cloud, it poses security as well as privacy issues. Examples of such problems are side-channel attacks, probe attacks, DDoS attacks, etc. [6], [7]. There are also significant security issues arising out of malware and attacks in the cloud which not only have access to both data and services of many users but also the ability to propagate to many systems over the cloud infrastructure. Moreover, there is the issue of trust on the cloud providers themselves . Consequently, it has become inevitable for the service operators to guarantee the security, quality and availability of their services. For increasing tenant's trust on cloud service provider, the reputation of the Cloud Providers (tenant's workers) can be utilized [8], [9]. As it aids tenant to select an appropriate tenant's worker. A reputation management mechanism (RMM) targets to take note of the selfish and malicious behaviors of tenant's workers and reflect this on their reputation [10]. Recently, number of RMM based approaches has been presented [11], [12], [13], [14], [15], [16], and [17] to address security and privacy issues of cloud computing environment. The state-of-art reputation management mechanism can efficiently eliminate malicious tenants when the tenants behave in an expected manner. However, the model suffers immensely when tenants start showing dynamic behavior (i.e. rapid behavior changes) and due to biased feedback. For overcoming these research challenges, the authors present a secure multi-tenant (SMT) design for cloud computing environment. The SMT is a reputation (feedback) based design which is dynamic in nature. SMT reduce storage overhead as it aids in using exponential mean operation for computing secureness weight of tenant's workers. Also, the SMT workload scheduling model distributes load efficiently. Thus, support in reducing latency, and enhance system throughput performance.

The research contribution as follows A novel secureness weight metrics and scheduling design using reputation based approach. A model to evaluate accuracy of feedback. An efficient workload scheduling. Experiment outcome shows SEMTDBMS reduces latency and attain higher throughput compared with state of-art model. The rest of the paper is organized as follows. Literature survey is presented in section II. In section III the proposed Secure Multi-Tenant Design for Cloud Computing Environment is presented. In penultimate section experimental study is carried out. The conclusion and future work is described in last section.

II. LITRATURE SURVEY

This section conducted extensive survey on addressing security and privacy issue in designing Multi-tenant database design under cloud computing environment. In [11], the authors presented a cost efficient and robust strategy to identify malicious events in a public virtualized cloud computing environment. They also presented robust workload estimation of virtual host events in cloud environment and also presented a detection model to distinguish infected hosts with low malicious events hidden within their appropriate workload and potentially scattered across multiple tenants. In [12] D. Gonzales et al., presented a cloud computing design that includes a wide range of security mechanism and practices, and cloud security evaluation model using cloud trust model. Cloud trust model estimated high level security metrics that define the level of integrity and confidentiality offered by cloud service provider. They further showed that security level of four multi-tenant IaaS cloud architectures probability of cloud computing system penetration (high value data compromise) is high if a minimal set of security controls are implemented. In [13], a SLA-aware fine-grained QoS provisioning and bandwidth allocation model. Their model maximized revenue by predicting the SLA popularity using K-nearest neighbor algorithm. In [14] the authors showed that tenants must be separated from one another based on security policies and cloud infrastructure. Their model presented a multi-level authorization (rules) separation model by learning from trustworthiness in cloud. In [15], two-stage strategy for provisioning security was discussed. Firstly, a trustworthy trust relationship toward guest Virtual Machines is established by hypervisor by considering both subjective and objective trust sources and by employing Bayesian inference is used to aggregate them. Further, they also presented maximin game among hypervisor trying to maximize this minimization under limited resources constraint and DDoS intruder trying to minimize the cloud computing system's detection. In [16] F. Banaie and S. A. H. Seno, described method for identifying wide range of attacks in trusted virtual domain, malicious insider attack, attack among computing worker in different domains and attacks against specific services such as DNS, database and web servers within a domain. They presented a fine granular detection of malicious entities and mechanisms and aided in enhancing communications among computing worker in different domains. In [17] the authors mentioned that as communication channels and other computational resources are shared in multi-tenant architecture, the state-of-art model is prone to various security and privacy issues. As tenants are anonymous in nature a user may not find a trustworthy co-tenant. The tenants depend on cloud service provider to assign trustworthy co-tenants. However, cloud service provider allows maximum co-tenancy irrespective of the behaviors of tenants to maximize resource utilization. For, demonstration they presented a robust reputation management mechanism that aid cloud service provider to identify good and malicious tenants and allocate resources in such a way that they do not share resources.

III. SECURE MULTITENANT DESIGN FOR CLOUD COMPUTING ENVIRONMENT

This section presents a secure multitenant (SMT) design. The SMT model provides dynamic security design even in presence of malicious tenant's worker (system). The working is as follows: Let us assume tenant need to compute the security weight of tenant operating with same tenant worker. First, tenant's worker under consideration is evaluated based on feedback information and accuracy of feedback information. Followed by which, a secureness weight based scheduling design is described.

A. *Secureness evaluation of tenant worker*

This section evaluates the degree of secureness a tenant has on a given tenant's worker in a multi-tenant system. That is, it keeps information of secureness status of all transaction a tenant perform with presence of another tenant operating with same tenants workers. This work aimed to reduce storage overhead and also assign session specific threshold to the transaction. As a result, rather than keeping all information of previous transaction, this work describes an exponential mean update operation to store the outcome of secureness. Let $Sec_o^u(x,y)$ depicts total secureness of tenant x has upon y of service execution considering o number of transaction in the u^{th} period of time instance. The secureness operation can be described as follows

$$Sec_o^u(x,y) = \gamma * Sec_{rec}(x,y) + (1-\gamma) * Sec_{o-1}^u(x,y). \quad (1)$$

(1)

where Sec_{rec} depicts the secureness parameter of most recent transaction and this work considered a feedback based model where an tenant rates secureness of other tenants operating in same worker using following expression such as

Firstly, the outcome of secureness at the start of u^{th} time instance/session is equal to the previously estimated secureness in the $(u-1)^{th}$ time instance and the initial secureness outcome is $Sec_o^0(x,y)=0$ which can expressed as follows

$$Sec_o^u(x,y) = Sec_{prev}^{u-1}(x,y) \quad (2) \qquad (2)$$

Secondly, if a transaction is completely secured then

$$Sec_{rec} = 1, \quad (3) \qquad (3)$$

If a transaction is completely not secured then $Sec_{rec} = 0, \quad (4)$

If it's neither secured nor secured then

$$Sec_{rec} = \in (0,1). \quad (5)$$

The threshold γ changes with respect to cumulated deviation $\beta_o^u(x,y)$.

$$\gamma = W + d * \frac{\mu_o^u(x,y)}{1+\beta_o^u(x,y)}, \qquad (6)$$

$$\beta_o^u(x,y) = d * \mu_o^u(x,y) + (1-d) * \beta_{o-1}^u(x,y). \quad (7)$$

where $\beta_o^u(x,y) = \beta_{prev}^{u-1}(x,y)$ and $\beta_o^0(x,y) = 0$, d is a system distinct constant parameter for addressing how system react to recent error $\mu_o^u(x,y)$ which is expressed as follows

$$\mu_o^u(x,y) = |Sec_{o-1}^u(x,y) - Sec_{rec}|. \qquad (8)$$

So, as we decease parameter size of d, then we give less importance to recent deviation than cumulated deviation and vice versa. Further, we can notice that recent failure $\mu_o^u(x,y)$ increases so does β which depicts more importance is given to recent feedback (i.e., recent importance is given higher threshold than cumulated secureness). The W depicts secureness weights which aid in preventing β from becoming to fixed parameter.

B. Feedback weight accuracy evaluation

In state-of-art model, the feedback information given by good tenant is considered to be true and feedback given by malicious tenant is false. However, in actual environment this assumption may not be always real as good tenant might give false feedback information and malicious tenant might periodically give good feedback information to preserve their malicious nature. As a result, accuracy of feedback information need to be computed. During secureness weight evaluation, feedback given by tenants with high reliability are highly secured and trusted. Therefore, higher weights is given to them when compared with low reliable tenants. Let $\mathrm{F}_o^u(x,p)$ depicts feedback accuracy of tenants y from tenant $x's$ point of view which is computed as follows

$$\mathrm{F}_o^u(x,p) = \begin{cases} 1 - \dfrac{\log\left(Sec_o^u(x,p)\right)}{\log\theta}, & if\ Sec_o^u(x,p) > \theta, \\ 0, & else \end{cases}$$

(9)

C. Final secureness matric evaluation and secure scheduling model

In this section we present the final secureness weight matrix for SMT. Let $\mathcal{F}_o^u(x,y)$ depicts the final secureness weight given by tenant x on tenant y operating with same tenant worker and the final secureness matrix is computed as follows

$$\mathcal{F}_o^u(x,y) = Sec_o^u(x,y) * \mathrm{F}_o^u(x,p). \qquad (10)$$

Using Eq. (10), we obtain good tenants worker. If we start scheduling work to good tenant's worker it incurs overload to worker with high secureness weight metrics. For balancing load among tenants worker we first compute the approximate traffic/load at tenant worker. Based on increasing order of load the tenant's worker are sorted and then work is assigned to tenant worker with least load. The SMT design attain significant performance improvement over state-of-art model which is experimentally proven in next section below.

IV. SIMULATION RESULT AND ANALYSIS

This section present performance evaluation of proposed SMT model over state-of-model [2] in terms of workload execution latency, time, and throughput (transaction per seconds). In [2] presented a multitenant design namely MuTeBench and carried out experiment using YCSB benchmark. The MuTeBench did not considered security parameter as an SLA parameter. For, provision security, this work presented SMT model and experiment is conducted to evaluate performance over exiting model [2]. For experiment analysis this work used H2 database and our model can support other database such as MySQL, Oracle etc. by using Hibernate framework. The SMT is developed using JAVA programming language using eclipse neon. The system environment used for workload execution I-5, 3.2 GHz, Quad core Intel class processor with 16 GB RAMS and Nvidia CUDA enabled 4 GB dedicated graphic card.

A. Workload execution latency performance evaluation considering varied tenant worker size:

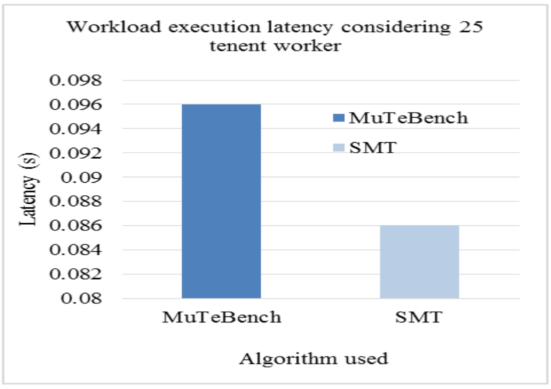

Fig. 1. Workload execution

latency performance evaluation considering 25 tenants worker.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

This work considered workload execution of OLTP and YCSB benchmark on H2 database. The workload execution is carried out for MuTeBench and SMT model. Each tenant is given a set of worker for workload execution. The number of worker is varied as 25 and 50.

The tenant ID is incremented by 2 and 4 tenant per execution is considered. Each tenant executes its workload with unlimited data rate. The OLTP and YCSB workload mix is composed of 25% read record and 15% for each other transaction types. In Fig. 1 the workload execution latency performance considering 25 tenants worker is obtained. The outcome shows SMT reduce workload execution latency by 10.42% considering 25 tenant worker. Similarly, in Fig. 2 the workload execution latency performance considering 50 tenants worker is obtained. The outcome shows SMT reduce workload execution latency by 26.7% considering 50 tenant worker. An average latency reduction of 18.54% is attained by SMT model over MuTeBench

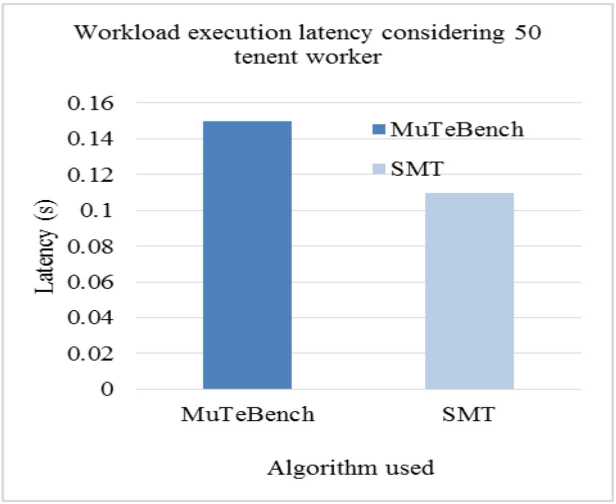

Fig. 2. Workload execution latency performance evaluation considering 25 tenants worker.

B. Workload execution throughput performance evaluation considering varied tenant worker size:

This work considered workload execution of OLTP and YCSB benchmark on H2 database. The workload execution is carried out for MuTeBench and SMT model. Each tenant is given a set of worker for workload execution. The number of worker is varied as 25 and 50. The tenant ID is incremented by 2 and 4 tenant per execution is considered. Each tenant executes its workload with unlimited data rate. The OLTP and YCSB workload mix is composed of 25% read record and 15% for each other transaction types. In Fig. 3 the throughput performance considering 25 tenant worker is obtained. The outcome shows SMT model improves throughput by 2.51% over MuTeBench. Similarly, in Fig. 4 the throughput performance considering 50 tenant worker is obtained. The outcome shows SMT model improves throughput by 3.02% over MuTeBench. An average throughput improvement of 2.77% is attained by SMT over MuTeBench.

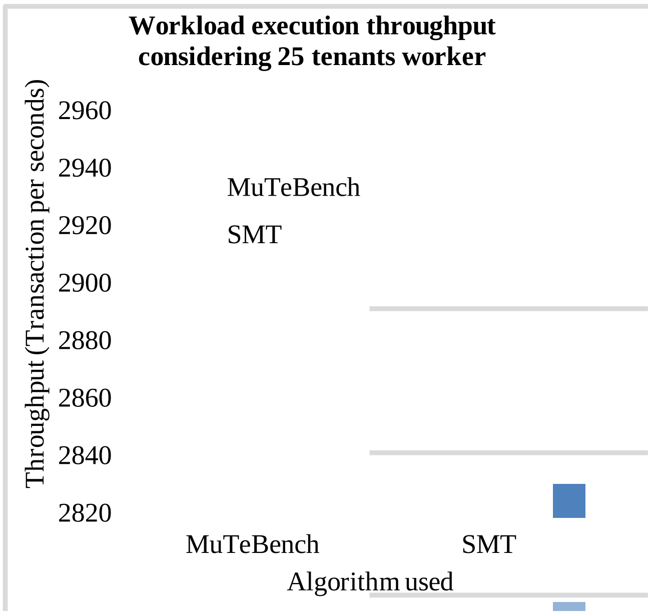

Fig. 3. Workload execution throughput performance evaluation considering 25 tenants worker.

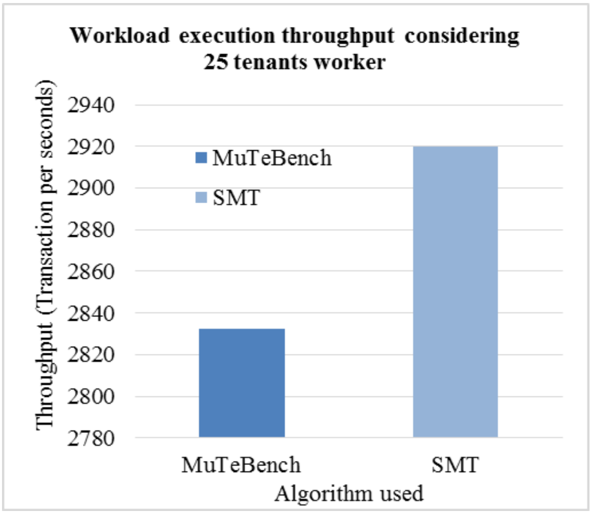

Fig. 4. Workload execution throughput performance evaluation considering 50 tenants worker.

V. CONCLUSION

As resources are shared and tenant are anonymous in nature in multi-tenant cloud environment it induce security and privacy issues. Further, tenant may not find a trustworthy co-tenant as they are dependent on CSP to assign trustworthy co-tenants. Thus, CSP maximize co-tenancy irrespective of the tenant's behavior. For addressing, number of state-of-art approaches used reputation management mechanism to identify good and malicious tenants. However, exiting model are not efficient when rapid changes of behavior of malicious tenant exist.

For overcoming research challenges this work presented a secure multitenant model for cloud computing environment. Then, preened a secureness weight matric based scheduling design using reputation based approach and also presented a model to evaluate accuracy of feedback. Experiment is conducted considering TPCC benchmark shows an average latency reduction of 18.44% and throughput performance improvement of 2.77% over MuTeBench. The overall result obtained shows significant performance in terms of latency, reduction and throughput improvement by SMT over state-of-art model. The future work we would consider evaluating under different workload/benchmarks. Further, this work would consider adding more secureness weight parameter to obtain more dynamic security model.

ACKNOWLEDGMENT

The authors would like to acknowledge and thank Technical Education Quality Improvement Program [TEQIP] Phase 3, BMS College of Engineering.

REFERENCES

1. NIST Special Publication 800-145, September 2011.
2. Andreas Gobel,"MuTeBench: Turning OLTP-Bench into a Multi-Tenancy Database Benchmark Framework", 2014, The fifth International Conference on Cloud Computing, GRIDs and Virtualization, ISBN: 978-1-61208-338-4, 2014.
3. G. B. Pallavi and P. Jayarekha, "An efficient resource sharing technique for multi-tenant databases," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, pp. 90-95, 2017.
4. A. D. Breslow, A. Tiwari, M. Schulz, L. Carrington, L. Tang, and J. Mars. Enabling fair pricing on hpc systems with node sharing. In SC, 2013.
5. D. Gmach, J. Rolia, and L. Cherkasova. Selling t-shirts and time shares in the cloud. In CCGRID, 2012.
6. A. Bates, B. Mood, J. Pletcher, H. Pruse, M. Valafar, and K. Butler, "On detecting co-resident cloud instances using network flow watermarking techniques," Int. J. Inf. Secur., vol. 13, no. 2, pp. 171– 189, Apr. 2014.
7. Y. Azar, S. Kamara, I. Menache, M. Raykova, and B. Shepard, "Colocation- resistant clouds," in Proceedings of the 6th Edition of the ACM Workshop on Cloud Computing Security, ser. CCSW '14. New York, NY, USA: ACM, pp. 9–20, 2014.
8. S. Habib, S. Hauke, S. Ries, and M. Mhlhuser, "Trust as a facilitator in cloud computing: a survey," Journal of Cloud Computing, vol. 1, no. 1, 2012.
9. J. Huang and D. Nicol, "Trust mechanisms for cloud computing," Journal of Cloud Computing, vol. 2, no. 1, 2013.
10. R. Ko, P. Jagadpramana, M. Mowbray, S. Pearson, M. Kirchberg, Q. Liang, and B. S. Lee, "Trustcloud: A framework for accountability and trust in cloud computing," in Services (SERVICES), 2011 IEEE World Congress on, pp. 584–588, 2011.
11. R. Cogranne, G. Doyen, N. Ghadban and B. Hammi, "Detecting Botclouds at Large Scale: A Decentralized and Robust Detection Method for Multi-Tenant Virtualized Environments," in IEEE Transactions on Network and Service Management, vol. 15, no. 1, pp. 68-82, March 2018.
12. D. Gonzales, J. M. Kaplan, E. Saltzman, Z. Winkelman and D. Woods, "Cloud-Trust—a Security Assessment Model for Infrastructure as a Service (IaaS) Clouds," in IEEE Transactions on Cloud Computing, vol. 5, no. 3, pp. 523-536, 1 July-Sept. 2017.
13. G. Li, J. Wu, J. Li, Z. Zhou and L. Guo, "SLA-Aware Fine-Grained QoS Provisioning for Multi-Tenant Software-Defined Networks," in IEEE Access, vol. 6, pp. 159-170, 2018.
14. W. Ma, Z. Han, X. Li and J. Liu, "A multi-level authorization based tenant separation mechanism in cloud computing environment," in China Communications, vol. 13, no. 5, pp. 162-171, May 2016.
15. O. Abdel Wahab, J. Bentahar, H. Otrok and A. Mourad, "Optimal Load Distribution for the Detection of VM-based DDoS Attacks in the Cloud," in IEEE Transactions on Services Computing. doi: 10.1109/TSC.2017.2694426
16. F. Banaie and S. A. H. Seno, "A cloud-based architecture for secure and reliable service provisioning in wireless sensor network," 2014 4th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, 2014, pp. 96-101.
17. S. Thakur and J. G. Breslin, "A Robust Reputation Management Mechanism in Federated Cloud," in IEEE Transactions on Cloud Computing. doi: 10.1109/TCC.2017.2689020.

AUTHORS PROFILE

Pallavi G B, Assistant Professor, Dept of CSE,BMSCE is currently pursuing Ph.D. in the field of Multitenancy in SaaS in VTU,India. E-mail: pallavi. cse@bmsce.ac.in
.

Dr. P Jayarekha, Associate Professor, Dept of ISE,BMSCE holds Ph.D. degree in computer science. She has published more than 15 research papers in referred International Journals and also few in national and international conferences. Research Scholars are working on various fields like cloud computing,WSN and related areas under her guidance. E-mail: jayarekha.ise@bmsce.ac.in