IJRRECS/June 2015/Volume-3/Issue-6/3991-3995 **ISSN 2321-5461**

INTERNATIONAL JOURNAL OF REVIEWS ON RECENT ELECTRONICS AND COMPUTER SCIENCE

IMPLEMENTATION OF EFFECTIVE TRANSACTION BY ACCURACY MANAGEMENT

Kudikala Hemalatha[1], B.Srikanth[2]

[1]M.Tech Student, Dept of CSE, Holy Mary Institute of Technology & Science, Hyderabad, T.S, India

[2]Assistant Professor, Dept of CSE, Holy Mary Institute of Technology & Science, Hyderabad, T.S, India

ABSTRACT:

There are remarkable aspects of cloud computing and among them flexibility is the one which make available an illusion of unlimited, on demand resources making it a striking setting for extremely scalable, multi-tiered applications. We deal with confluence of data, policy, as well as credential inconsistency problems in our work that can become known as transactional database systems which are cloud structured. Concept of trusted transactions were formalised which are the transactions that do not go against credential or else policy inconsistencies over the existence of transaction. We suggest a Two-Phase Validation Commit practice that guarantees that a transaction is safe by means of inspecting policy, credential, as well as data constancy throughout transaction execution. The Two-Phase Validation procedure put into effect trustworthy transactions, but does not put into effect safe transactions since it does not authenticate any integrity controls.

Keywords: Cloud computing, Safe transaction, Two-Phase Validation, Data constancy, Trusted transactions.

1. INTRODUCTION:

A transactional database system that is positioned in an extremely distributed system for instance cloud; policies would usually be replicated extremely much like data among numerous sites. To make available scalability as well as elasticity, cloud services regularly make much usage of replication to make sure reliable performance and accessibility [1]. Consequently, numerous cloud services

depend on eventual consistency when propagating information throughout system. This consistency representation is a variation of weak constancy that allows data to be incompatible between some replicas during update process, but make sure that updates will ultimately be propagated to the entire replicas. Interesting consistency exertions can take place since transactional database systems are positioned in cloud environments and make use of policy-based authorization systems to look after sensitive resources. We must moreover handle two types of security inconsistency circumstances such as: first, system might experience from policy inconsistencies throughout policy updates because of relaxed constancy representation underlying the majority of cloud services. Secondly, it is likely for exterior factors to cause user credential irregularities over existence of a transaction. In our work we address confluence of data, policy, as well as credential inconsistency problems that can become known as transactional database systems which are organized to cloud. We formalize notion of trusted transactions which are the transactions that do not go against credential or else policy inconsistencies over the existence of transaction [2][3]. We put forward a Two-Phase Validation Commit (2PVC) procedure that guarantees that a transaction is safe by means of inspecting policy, credential, as well as data constancy throughout transaction execution. As Two-Phase Commit atomic procedure typically used to apply integrity constraints has comparable structure as Two-Phase Validation, we put forward integrating these procedures into a Two-Phase Validation Commit practice.

2. APPROACHES FOR IMPLEMENTATION OF TRUSTED TRANSACTIONS:

One of the most interesting aspects of cloud computing is its flexibility, which make available an illusion of unlimited, ondemand resources making it an striking setting for extremely scalable, multi-tiered applications. However, this build extra challenges for back-end, transactional databases, which are considered devoid of elasticity in mind. The system model illustrates interaction among components in our system. We imagine a cloud infrastructure which consists of a set of servers, where every server is accountable for hosting a subset of the entire data items that belongs to a specific application domain. Users cooperate with system by

means of submitting queries or else update requests. A transaction is provided to a Transaction Manager that manages its execution. Multiple transaction managers might be invoked since system workload increases in support of load balancing; however each transaction is handled by simply one transaction manager. We suppose that queries that belong to a transaction carry out successively, and that a transaction does not split sub transactions. These suppositions make simpler our presentation, but do not have an effect on the exactness or strength of our consistency definitions. The set of the entire credentials, are issued by Certificate Authorities within system [4]. We imagine that each Certificate Authorities presents an online means that allows any server to make sure present status of credentials that it has provided. These consistency models reinforce trusted transaction definition by defining setting in which policy versions are constant relative to rest of system. Deferred proofs provide an optimistic method with comparatively weak authorization assurances. The proofs of authorizations are assessed concurrently only at commit time to make a decision whether transaction is trusted. Punctual proofs provide a more proactive method in which proofs of authorizations are evaluated instantly as soon as a query is being managed by a server. These make possible early discoveries of unsafe transactions which can accumulate system from going into pricey undo operations. Punctual proofs do not enforce any restrictions on freshness of policies used by servers throughout the transaction execution. Incremental Punctual proofs build up a stronger view of trusted transactions, since a transaction is not authorized to continue unless each server achieves required level of policy constancy with all earlier servers. In Continuous proofs, when a proof is evaluated, the entire earlier proofs have to be re-evaluated when a latest version of policy is found at any of participating servers [5]. At commit time, Continuous proofs perform equally to Incremental Punctual proofs. The decision of which method to approve is expected to be a considered choice made autonomously by every application. While with any trade-off, stronger the security and accurateness provided by an approach, more the system has to pay regarding functioning and messages exchange overheads.

3. AN OVERVIEW OF TWO-PHASE VALIDATION COMMIT PROCEDURE:

Safe transaction is a transaction that is trustworthy specifically satisfies accuracy properties of proofs of authorization as well as database correct. A common feature of most of our projected approaches to attain trusted transactions is requirement for policy consistency justification at end of a transaction. In order for a trustworthy transaction to commit, its transaction manager has to put into effect either views or else global consistency among servers participating in transaction. We recommend a new algorithm known as Two-Phase Validation. As the name means, Two-Phase Validation operates in two phases such as collection as well as validation. During collection, transaction manager initially sends a Prepare-to-Validate message towards each participant server. In reply to this message, every participant assess proofs for each query of transaction by means of most recent policies it has accessible and sends a response back to transaction manager containing truth value of those proofs all along with version number as well as policy identifier for each used policy. The Two-Phase Validation procedure put into effect trustworthy transactions, but does not put into effect safe transactions since it does not authenticate any integrity constraints. While Two-Phase Commit atomic procedure usually used to implement integrity constraints has comparable structure as Two-Phase Validation, we put forward integrating these procedures into a Two-Phase Validation Commit procedure. Two-Phase Validation Commit procedure can be used to guarantee the data as well as policy consistency needs of safe transactions. The flexibility of Two-Phase Validation Commit procedure to system as well as communication failures can be attained in similar manner as Two-Phase Commit by recording progress of procedure in logs of transaction manager and participants [6].

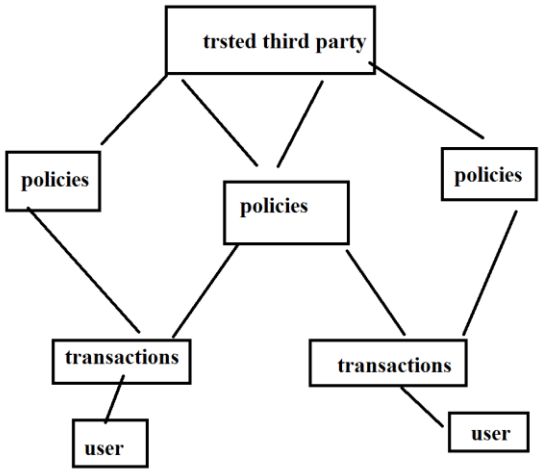

Fig1: Associations between system components.

4. CONCLUSION:

To grant access to flexibility and scalability, cloud services regularly make much usage of replication to make sure reliable performance and accessibility. In our work we tackle confluence of data, policy, as well as credential inconsistency problems that can become known as transactional database systems which are organized to cloud. We formalize perception of trustworthy transactions which are the transactions that do not go against credential or else policy inconsistencies over the existence of transaction. Two-Phase Validation Commit procedure was introduced that guarantees that a transaction is safe by means of inspecting policy, credential, as well as data constancy throughout transaction execution. We suggest a new algorithm known as Two-Phase Validation and as the name means, Two-Phase Validation operates in two phases such as collection as well as validation. The Two-Phase Validation process enforces trustworthy transactions, but does not put into effect safe transactions since it does not authenticate any integrity constraints. Though Two-Phase Commit atomic process typically used to apply integrity constraints has comparable structure as Two-Phase Validation, we put forward integrating these procedures into a Two-Phase Validation Commit process.

REFERENCES

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), 2007.

[2] K.D. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09), 2009.

[3] A. Shraer, C. Cachin, A. Cidon, I. Keidar, Y. Michalevsky, and D. Shaket, "Venus: Verification for Untrusted Cloud Storage," Proc. ACM Workshop Cloud Computing Security (CCSW '10), 2010.

[4] M.K. Iskander, D.W. Wilkinson, A.J. Lee, and P.K. Chrysanthis, "Enforcing Policy and Data Consistency of Cloud Transactions," Proc. IEEE Second Int'l Workshop Security and Privacy in Cloud Computing (ICDCS-SPCCICDCS-SPCC), 2011.

[5] G. DeCandia et al., "Dynamo: Amazons Highly Available Key- Value Store," Proc. 21st ACM SIGOPS Symp. Operating Systems Principles (SOSP '07), 2007.

[6] F. Chang et al., "Bigtable: A Distributed Storage System for Structured Data," Proc. Seventh USENIX Symp. Operating System Design and Implementation (OSDI '06), 2006.