ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Access to Cloud Databases of Encrypted Through Distributed, Concurrent, and Independent Method

Rangisetty Pallavi
PG Scholar,
Department of CSE,
Sri Chundi Ranganayakulu Engineering College,
Chilakaluripet, Guntur, AP, India.

G.Mallikharjuna Rao
Assistant Professor,
Department of CSE,
Sri Chundi Ranganayakulu Engineering College,
Chilakaluripet, Guntur, AP, India.

ABSTRACT:

Placing critical data in the hands of a cloud provider should come with the guarantee of security and availability for data at rest, in motion, and in use. Several alternatives exist for storage services, while data confidentiality solutions for the database as a service paradigm are still immature. We propose a novel architecture that integrates cloud database services with data confidentiality and the possibility of executing concurrent operations on encrypted data. This is the first solution supporting geographically distributed clients to connect directly to an encrypted cloud database, and to execute concurrent and independent operations including those modifying the database structure. The proposed architecture has the further advantage of eliminating intermediate proxies that limit the elasticity, availability, and scalability properties that are intrinsic in cloud-based solutions. The efficacy of the proposed architecture is evaluated through theoretical analyses and extensive experimental results based on a prototype implementation subject to the TPC-C standard benchmark for different numbers of clients and network latencies.

INTRODUCTION:

IN a cloud context, where critical information is placed in infrastructures of untrusted third parties, ensuring dataconfidentiality is of paramount importance [1], [2]. Thisrequirement imposes clear data management choices:original plain data must be accessible only by trustedparties that do not include cloud providers, intermediaries,and Internet; in any untrusted context, data must beencrypted. Satisfying these goals has different levels ofcomplexity depending on the type of cloud service. Thereare several solutions ensuring confidentiality for the storageas a service paradigm (e.g., [3], [4], [5]), while guaranteeingconfidentiality in the database as a service (DBaaS) paradigm[6] is still an open research area.

In this context, we proposeSecureDBaaS as the first solution that allows cloud tenantsto take full advantage of DBaaS qualities, such asavailability, reliability, and elastic scalability, withoutexposing unencrypted data to the cloud provider.The architecture design was motivated by a threefoldgoal: to allow multiple, independent, and geographicallydistributed clients to execute concurrent operations onencrypted data, including SQL statements that modify thedatabase structure; to preserve data confidentiality andconsistency at the client and cloud level; to eliminate anyintermediate server between the cloud client and the cloudn provider. The possibility of combining availability, elasticity,and scalability of a typical cloud DBaaS with dataconfidentiality is demonstrated through a prototype of SecureDBaaS that supports the execution of concurrentand independent operations to the remote encrypteddatabase from many geographically distributed clients asin any unencrypted DBaaS setup. To achieve these goals,SecureDBaaS integrates existing cryptographic schemes,isolation mechanisms, and novel strategies for managementof encrypted metadata on the untrusted cloud database.

Thispaper contains a theoretical discussion about solutions fordata consistency issues due to concurrent and independentclient accesses to encrypted data. In this context, we cannotapply fully homomorphic encryption schemes [7] because oftheir excessive computational complexity.The SecureDBaaS architecture is tailored to cloudplatforms and does not introduce any intermediary proxyor broker server between the client and the cloudprovider. Eliminating any trusted intermediate serverallows SecureDBaaS to achieve the same availability,reliability, and elasticity levels of a cloud DBaaS. Otherproposals (e.g., [8], [9], [10], [11]) based on intermediateserver(s) were considered impracticable for a cloud-basedsolution because any proxy represents a single point offailure and a system bottleneck that limits the mainbenefits (e.g., scalability, availability, and elasticity) of adatabase service deployed on a cloud platform.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

UnlikeSecureDBaaS, architectures relying on a trusted intermediateproxy do not support the most typical cloud scenariowhere geographically dispersed clients can concurrentlyissue read/write operations and data structure modificationsto a cloud database. A large set of experiments based on real cloud platformsdemonstrate that SecureDBaaS is immediately applicable toany DBMS because it requires no modification to the clouddatabase services. Other studies where the proposedarchitecture is subject to the TPC-C standard benchmarkfor different numbers of clients and network latenciesshow that the performance of concurrent read and writeoperations not modifying the SecureDBaaS database structure is comparable to that of unencrypted clouddatabase.

Workloads including modifications to the databasestructure are also supported by SecureDBaaS, but atthe price of overheads that seem acceptable to achieve thedesired level of data confidentiality. The motivation ofthese results is that network latencies, which are typicalof cloud scenarios, tend to mask the performance costs ofdata encryption on response time. The overall conclusionsof this paper are important because for the first time theydemonstrate the applicability of encryption to clouddatabase services in terms of feasibility and performance.

RELATED WORK:

SecureDBaaS provides several original features that differentiateit from previous work in the field of security forremote database services.. It guarantees data confidentiality by allowing acloud database server to execute concurrent SQLoperations (not only read/write, but also modifications to the database structure) over encrypted data.. It provides the same availability, elasticity, andscalability of the original cloud DBaaS because itdoes not require any intermediate server. Responsetimes are affected by cryptographic overheads thatfor most SQL operations are masked by networklatencies..

Multiple clients, possibly geographically distributed,can access concurrently and independently a clouddatabase service.. It does not require a trusted broker or a trustedproxy because tenant data and metadata stored bythe cloud database are always encrypted.. It is compatible with the most popular relationaldatabase servers, and it is applicable to differentDBMS implementations because all adopted solutionsare database agnostic.

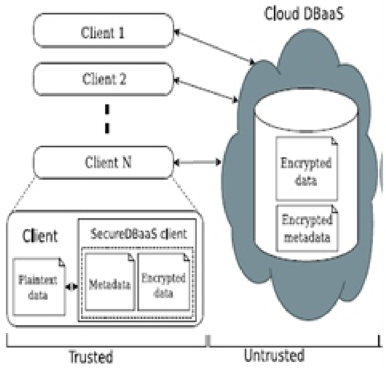

Fig. 1. SecureDBaaS architecture

ARCHITECTURE DESIGN:

SecureDBaaS is designed to allow multiple and independentclients to connect directly to the untrusted cloudDBaaS without any intermediate server. Fig. 1 describes theoverall architecture. We assume that a tenant organizationacquires a cloud database service from an untrusted DBaaSprovider. The tenant then deploys one or more machines(Client 1 through N) and installs a SecureDBaaS client oneach of them. This client allows a user to connect to thecloud DBaaS to administer it, to read and write data, andeven to create and modify the database tables after creation.

We assume the same security model that is commonlyadopted by the literature in this field (e.g., [8], [9]), wheretenant users are trusted, the network is untrusted, and thecloud provider is honest-but-curious, that is, cloud serviceoperations are executed correctly, but tenant informationconfidentiality is at risk. For these reasons, tenant data, datastructures, and metadata must be encrypted before exitingfrom the client. A thorough presentation of the securitymodel adopted in this paper is in Appendix A, available inthe online supplemental material.

Data Management:

We assume that tenant data are saved in a relationaldatabase. We have to preserve the confidentiality of thestored data and even of the database structure because tableand column names may yield information about saved data. We distinguish the strategies for encrypting the databasestructures and the tenant data.Encrypted tenant data are stored through secure tablesinto the cloud database.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Metadata Management:

Metadata generated by SecureDBaaS contain all the information that is necessary to manage SQL statementsover the encrypted database in a way transparent to theuser. Metadata management strategies represent an originalidea because SecureDBaaS is the first architecture storing allmetadata in the untrusted cloud database together with theencrypted tenant data. SecureDBaaS uses two types ofmetadata.

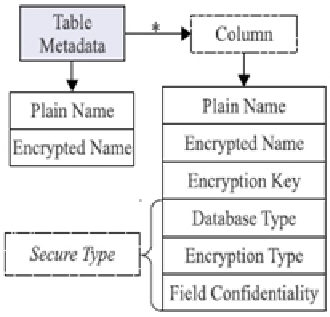

Fig. 2. Structure of table metadata.

and the unencrypted name of the related plaintext table. Moreover, table metadata include column metadata for eachcolumn of the related secure table. Each column metadatacontain the following informationThis mechanism has the further benefit of allowingclients to access each metadata independently, which is animportant feature in concurrent environments. In addition,SecureDBaaS clients can use caching policies to reduce thebandwidth overhead.

Metadata Storage Table

ID	Encrypted Metadata	Control Structure
MAC('.'+Db)	Enc(Db metadata)	MAC(Db metadata)
MAC(T1)	Enc(T1 metadata)	MAC(T1 metadata)
MAC(T2)	Enc(T2 metadata)	MAC(T2 metadata)

Fig. 3. Organization of database metadata and table metadata in themetadata storage table.

OPERATIONS:

In this section, we outline the setup setting operationscarried out by a database administrator (DBA), and wedescribe the execution of SQL operations on encrypted datain two scenarios: a nai¨ve context characterized by a singleclient, and realistic contexts where the database services areaccessed by concurrent clients.

EXPERIMENTAL RESULTS:

We demonstrate the applicability of SecureDBaaS todifferent cloud DBaaS solutions by implementing andhandling encrypted database operations on emulated andreal cloud infrastructures. The present version of theSecureDBaaS prototype supports PostgreSQL, MySql, andSQL Server relational databases. As a first result, we canobserve that porting SecureDBaaS to different DBMSrequired minor changes related to the database connector,and minimal modifications of the codebase. We refer toAppendix C, available in the online supplemental material,for an in-depth description of the prototype implementation.Other tests are oriented to verify the functionality ofSecureDBaaS on different cloud database providers. Experimentsare carried out in Xeround [22], Postgres Plus CloudDatabase [23], Windows SQL Azure [24], and also on an IaaSprovider, such as Amazon EC2 [25], that requires a manualsetup of the database. The first group of cloud providersoffer ready-to-use solutions to tenants, but they do not allowa full access to the database system. For example, Xeroundprovides a standard MySql interface and proprietary APIsthat simplify scalability and availability of the cloud database, but do not allow a direct access to the machine.This prevents the installation of additional software, the useof tools, and any customization. On the positive side,SecureDBaaS using just standard SQL commands canencrypt tenant data on any cloud database service. Someadvanced computation on encrypted data may require theinstallation of custom libraries on the cloud infrastructure.This is the case of Postgres Plus Cloud that provides SSHaccess to enrich the database with additional functions.The next set of experiments evaluate the performanceand the overheads of our prototype. We use the Emulab [26]testbed that provides us a controlled environment withseveral machines, ensuring repeatability of the experimentsfor the variety of scenarios to consider in terms of workloadmodels, number of clients, and network latencies.

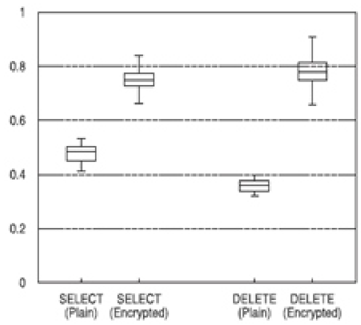

Fig. 6. Plain versus encrypted SELECT and DELETE operations.

To evaluate the performance overhead of encrypted SQL operations, we focus on the most frequently executed SELECT, INSERT, UPDATE, and DELETE commands of the TPC-C benchmark. In Figs. 6 and 7, we compare the response times of SELECT and DELETE, and UPDATE and INSERT operations, respectively. The Y -axis reports the boxplots of the response times expressed in ms (at a different scale), while the X-axis identifies the SQL operations. In SELECT, DELETE, and UPDATE operations, the response times of SecureDBaaS SQL commands are almost doubled, while the INSERT operation is, as expected, more critical from the computational point of view and it achieves a tripled response time with respect to the plain version. This higher overhead is motivated by the fact that an INSERT command has to encrypt all columnsof a tuple, while an UPDATE operation encrypts just one or few values.

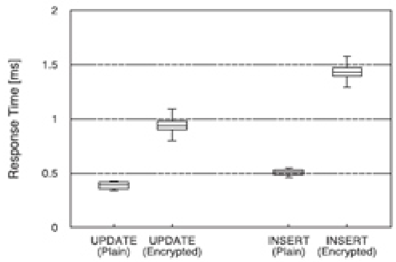

Fig. 7. Plain versus encrypted UPDATE and INSERT operations.

CONCLUSIONS:

We propose an innovative architecture that guarantees confidentiality of data stored in public cloud databases. Unlike state-of-the-art approaches, our solution does not rely on an intermediate proxy that we consider a single point of failure and a bottleneck limiting availability and scalability of typical cloud database services. A large part of the research includes solutions to support concurrent SQL operations (including statements modifying the database structure) on encrypted data issued by heterogenous and possibly geographically dispersed clients.

The proposed architecture does not require modifications to the cloud database, and it is immediately applicable to existing cloud DBaaS, such as the experimented PostgreSQL Plus Cloud Database [23], Windows Azure [24], and Xeround [22]. There are no theoretical and practical limits to extend our solution to other platforms and to include new encryption algorithms.

It is worth observing that experimental results based on the TPC-C standard benchmark show that the performance impact of data encryption on response time becomes negligible because it is masked by network latencies that are typical of cloud scenarios. In particular, concurrent read and write operations that do not modify the structure of the encrypted database cause negligible overhead. Dynamic scenarios characterized by (possibly) concurrency.

REFERENCES:

[1] M. Armbrust et al., “A View of Cloud Computing,” Comm. of the ACM, vol. 53, no. 4, pp. 50-58, 2010.

[2] W. Jansen and T. Grance, “Guidelines on Security and Privacy in Public Cloud Computing,” Technical Report Special Publication 800-144, NIST, 2011.

[3] A.J. Feldman, W.P. Zeller, M.J. Freedman, and E.W. Felten, “SPORC: Group Collaboration Using Untrusted Cloud Resources,” Proc. Ninth USENIX Conf. Operating Systems Design and Implementation, Oct. 2010.

[4] J. Li, M. Krohn, D. Mazie`res, and D. Shasha, “Secure Untrusted Data Repository (SUNDR),” Proc. Sixth USENIX Conf. Opearting Systems Design and Implementation, Oct. 2004.

[5] P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, andM. Walfish, “Depot: Cloud Storage with Minimal Trust,” ACMTrans. Computer Systems, vol. 29, no. 4, article 12, 2011.

[6] H. Hacigu¨mu¨ s¸, B. Iyer, and S. Mehrotra, “Providing Database as aService,” Proc. 18th IEEE Int’l Conf. Data Eng., Feb. 2002.

[7] C. Gentry, “Fully Homomorphic Encryption Using Ideal Lattices,”Proc. 41st Ann. ACM Symp. Theory of Computing, May 2009.

[8] R.A. Popa, C.M.S. Redfield, N. Zeldovich, and H. Balakrishnan,“CryptDB: Protecting Confidentiality with Encrypted QueryProcessing,” Proc. 23rd ACM Symp. Operating Systems Principles,Oct. 2011.

[9] H. Hacigu¨mu¨ s¸, B. Iyer, C. Li, and S. Mehrotra, “Executing SQL over Encrypted Data in the Database-Service-ProviderModel,” Proc. ACM SIGMOD Int’l Conf. Management Data, June2002.

[10] J. Li and E. Omiecinski, "Efficiency and Security Trade-Off inSupporting Range Queries on Encrypted Databases," Proc. 19thAnn. IFIP WG 11.3 Working Conf. Data and Applications Security,Aug. 2005.

[11] E. Mykletun and G. Tsudik, "Aggregation Queries in theDatabase-as-a-Service Model," Proc. 20th Ann. IFIP WG 11.3Working Conf. Data and Applications Security, July/Aug. 2006.

[12] D. Agrawal, A.E. Abbadi, F. Emekci, and A. Metwally, "DatabaseManagement as a Service: Challenges and Opportunities," Proc.25th IEEE Int'l Conf. Data Eng., Mar.-Apr. 2009.