e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 13
October 2017

A Novel Architecture For Adaptive Encryption Of Public Clouddatabases

Devarapalli Raja Anand & M.Srinivasarao
[1]PG Scholar, Dept of CSE, PACE Institute of Tech and Sciences, Vallur, Ongole, AP, India.
[2] Assistant Professor, Dept of CSE , PACE Institute of Tech and Sciences, Vallur, Ongole, AP, India.

Abstract*—The cloud database as a service is a novel paradigm that can support several Internet-based applications, but its adoption requires the solution of information confidentiality problems. We propose a novel architecture for adaptive encryption of public cloud databases that offers an interesting alternative to the tradeoff between the required data confidentiality level and the flexibility of the cloud database structures at design time. We demonstrate the feasibility and performance of the proposed solution through a software prototype. Moreover, we propose an original cost model that is oriented to the evaluation of cloud database services in plain and encrypted instances and that takes into account the variability of cloud prices and tenant workloads during a medium-term period.*

Keywords—Cloud database, confidentiality, encryption, adaptivity, cost model

1 INTRODUCTION

The cloud computing paradigm is successfully converg-ing as the fifth utility [1], but this positive trend is par-tially limited by concerns about information confidentiality [2] and unclear costs over a medium-long term [3], [4].We are interested in the database as a service para-digm (DBaaS) [5] that poses several research challenges in terms of security and cost evaluation from a tenant's point of view. Most results concerning encryption for cloud-based services [6], [7] are inapplicable to the data-base paradigm. Other encryption schemes that allow the execution of SQL operations over encrypted data either have performance limits [8] or require the choice of which encryption scheme must be adopted for each database column and SQL operation [9]. These latter proposals are fine when the set of queries can be statically determined at design time, while we are interested in other common scenarios where the workload may change after the data-base design. In this paper, we propose a novel architec-ture for adaptive encryption of public cloud databases that offers a proxy-free alternative to the system described in [10]. The proposed architecture guarantees in an adaptive way the best level of data confidentiality for any database workload, even when the set of SQL queries dynamically changes. The adaptive encryption scheme, which was initially proposed for applications not referring to the cloud, encrypts each plain column to mul-tiple encrypted columns, and each value is encapsulated in different layers of encryption, so that the outer layers guarantee higher confidentiality but support fewer com-putation capabilities with respect to the inner layers. The outer layers are dynamically adapted at runtime when new SQL operations are added to the workload.Although this adaptive encryption architecture is attrac-tive because it does not require to define at design time which database operations are allowed on each column, it poses novel issues in terms of applicability to a cloud con-text, and doubts about storage and network costs. We inves-tigate each of these issues and we reach three original conclusions in terms of prototype implementation, perfor-mance evaluation, and cost evaluation.

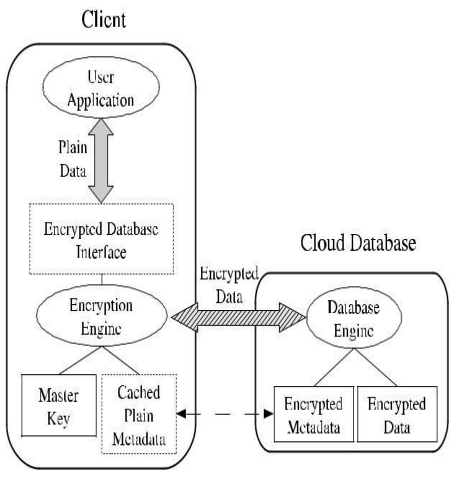
We initially design the first proxy-free architecture for adaptive encryption of cloud databases that does not limit the availability, elasticity and scalability of a plain cloud database because multiple clients can issue concurrent oper-ations without passing through some centralized compo-nent as in alternative architectures [10]. Then, we evaluate the performance of encrypted database services by assum-ing the standard TPC-C benchmark as the workload and by considering different network latencies. Thanks to this testbed, we show that most performance overheads of adaptively encrypted cloud databases are masked by net-work latencies that are typical of a geographically distrib-uted cloud scenario.Finally, we propose the first analytical cost estimation model for evaluating cloud database costs in plaintext and encrypted configurations from a tenant's point of view over a medium-term period. This model also considers the vari-ability of cloud prices and of the database workload during the evaluation period, and allows a tenant to observe how adaptive encryption influences the costs related to storage and network usage of a database service. By applying the model to several cloud provider offers and related prices, the tenant can choose the best compromise between the data confidentiality level and consequent costs in his period of interest.This paper is structured as following. Section 2 exam-ines related solutions for data confidentiality and cost estimation in cloud database services, and compares them against our proposal. Section 3 describes the proposed adaptive encryption architecture for cloud database directions for further research.

2 RELATED WORK

Improving the confidentiality of information stored in cloud databases represents an important contribution to the adop-tion of the cloud as the fifth utility because it addresses most user concerns. Our proposal is characterized by two main contributions to the state of the art: architecture and cost model.Although data encryption seems the most intuitive solu-tion for confidentiality, its application to cloud database services is not trivial, because the cloud database must be able to execute SQL operations directly over encrypted data without accessing any decryption key. Na€ıve solutions encrypt the whole database through some standard encryp-tion algorithms that do not allow to execute any SQL opera-tion directly on the cloud. As a consequence, the tenant has two alternatives: download the entire database, decrypt it, execute the query and, if the operation modifies the data-base, encrypt and upload the new data; decrypt temporarily the cloud database, execute the query, and re-encrypt it. The former solution is affected by huge communication and computation overheads, and consequent costs that would make cloud database services quite inconvenient; the latter solution does not guarantee data confidentiality because the cloud provider obtains decryption keys.The right alternative is to execute SQL operations directly on the cloud database, without giving decryption keys to the provider. An initial solution presented in [5] is based on data aggregation techniques [8], that associate plaintext metadata to sets of encrypted data. However, plaintext metadata may leak sensitive information and data aggrega-tion introduces unnecessary network overheads.The use of fully homomorphic encryption [11] would guar-antee the execution of any operation over encrypted data, but existing implementations are affected by huge compu- tion schemes with a proxy-free architecture was proposed by the same authors in [15]. This paper develops the initial design through a prototype implementation, novel experi-mental results and an original cost model. Indeed, besides data confidentiality, unclear costs are a main concern for cloud tenants. To this purpose, we propose an analytical cost model and a usage estimation methodol-ogy that allow a tenant to estimate the costs deriving from cloud database services characterized by plain, encrypted and adaptively encrypted databases over a medium-term horizon during which it is likely that both the database workload and the cloud prices change. This model is another original contribution of this paper because previous research focuses on the costs of cloud computing from a provider's perspective [16], [17]. For example, the authors in outline the problems related to the cost estimation of a cloud data center, such as servers, power consumption, and infrastructures, but they do not propose an analytical cost estimation model. CloudSim [18] can help a provider to esti-mate performance and resource consumptions of one or multiple cloud data center alternatives.This paper has a focus on database services and takes an opposite direction by evaluating the cloud service costs from a tenant's point of view. This approach is quite origi-nal because related papers evaluate the pros and cons of porting scientific applications to a cloud platform, such as focusing on specific astronomy software and a specific cloud provider (Amazon), and [3] presenting a composable cost estimation model for some classes of scientific applica-tions. Besides the focus on a different context (scientific ver-sus database applications), the proposed model can be applied to any cloud database service provider, and it takes into account that over a medium-term period the database workload and the cloud prices may vary.

3 ARCHITECTURE DESIGN

The costs to the extent that the execution of SQL opera- The proposed system supports adaptive encryption for pub- tions over a cloud database would become impractical.lic cloud database services, where distributed and concur- Other encryption algorithms characterized by acceptable rent clients can issue direct SQL operations. By avoiding ancomputational complexity support a subset of SQL opera- architecture based on intermediate servers [10] between the tors [12], [13], [14]. For example, an encryption algorithm clients and the cloud database, the proposed solution guar may support the order comparison command [12], but not antees the same level of scalability and availability of the a search operator [14]. The drawback related to these feasi- cloud service. Fig. 1 shows a scheme of the proposed archi- ble encryption algorithms is that in a medium-long term tecture where each client executes an encryption engine that horizon, the database administrator cannot know at design manages encryption operations. This software module is time which database operations will be required over each accessed by external user applications through the encrypteddatabase column. This issue is in part addressed in [10] by database interface.



plain metadata represent the additional information that is necessary to execute SQL operations on encrypted data; encrypted metadata are the encrypted version of the plain metadata, and are stored in the cloud database; master key is the encryption key of the encrypted metadata, and is known by legitimate clients.All data and metadata stored in the cloud database are encrypted. Any application running on a legitimate client can transparently issue SQL operations (e.g., SELECT, INSERT, UPDATE and DELETE) to the encrypted cloud database through the encrypted database interface. Data transferred between the user application and the encryption engine are not encrypted, whereas information is always encrypted before sending it to the cloud database. When an application issues a new SQL operation, the encrypted data-base interface contacts the encryption engine that retrieves the encrypted metadata and decrypts them with the master key. To improve performance, the plain metadata are cached locally by the client. After obtaining the metadata, the encryption engine is able to issue encrypted SQL state-ments to the cloud database, and then to decrypt the results. The results are returned to the user application through the encrypted database interface.As in related literature, the proposed architecture guar-antees data confidentiality in a security model in which: the network is untrusted; tenant users are trusted, that is, they do not reveal information about plain data, plain metadata, and the master key; the cloud provider administrators are defined semi-honest or honest-but-curious [19], that is, they do not modify tenant's data and results of SQL operations, but they may access tenant's information stored in the cloud database. The remaining part of this section describes the adaptive encryption schemes (Section 3.1), the encrypted metadata stored in the cloud database (Section 3.2), and the main operations for the management of the encrypted cloud database (Section 3.3).

3.1 Adaptive Encryption Schemes

We consider SQL-aware encryption algorithms that guaran-tee data confidentiality and allow the cloud database engine to execute SQL operations over encrypted data. As each algorithm supports a specific subset of SQL operators, we refer to the following encryption schemes. Random (Rand): it is the most secure encryption because it does not reveal any information about the original plain value (IND-CPA) [20], [21]. It does not support any SQL operator, and it is used only for data retrieval. Deterministic (Det): it deterministically encrypts data, so that equality of plaintext data is preserved. It sup-ports the equality operator Order Preserving Encryption (Ope) [12]: it preserves in the encrypted values the numerical order of the orig-inal unencrypted data. It supports the comparison SQL operators (i.e., ¼; < ; ; > ;). Homomorphic Sum (Sum) [13]: it is homomorphic with respect to the sum operation, so that the multi-plication of encrypted integers is equal to the sum of plaintext integers. It supports the sum operator between integer values. Search: it supports equality check on full strings (i.e., the LIKE operator) Plain: it does not encrypt data, but it is useful to sup- port all SQL operators on non confidential data.If each column of the database was encrypted with only one algorithm, then the database administrator would have to decide at design time which operations must be sup-ported on each database column. However, this solution is impractical for scenarios in which the database workload changes over time. As an example, if we consider a database supporting a web application for which features or security updates are released, data encryption prevents the applica-tion of any update that introduces new SQL operations that were not considered at database design time. Similarly, encryption prevents the execution of data analytics on the encrypted database and of user-defined queries that do not belong to a fixed workload (e.g., because the database is queried directly by tenant employees). This issue can be addressed through adaptive encryption schemes that sup-port at runtime SQL operations while preserving the highest possible data confidentiality level on the encrypted data. To this purpose, the encryption algorithms are organized into structures called onions, where each onion is composed by an ordered set of encryption algorithms, called (encryption) layers [10]. Outer layers guarantee higher level of data confi-dentiality and support fewer operations on encrypted data. Hence, each onion supports a specific set of operations. We design the following onions: its onions. For example, the plaintext values associated with Onion-Eq are encrypted with Det, then the Det value is encrypted with Rand. The most external layer of an onion is called actual layer, which corresponds to its strongest encryption algorithm. The cloud database can only see the actual layer of the onions, and has no access to inner layers nor to plaintext data. The first time that a new SQL opera-tion is requested on a column, the outer layer of the appro-priate onion is dynamically removed at runtime through the adaptive layer removal mechanism that exposes a layer supporting the requested operations. This layer becomes the new actual layer of the onion in the encrypted database. The layer removal mechanism is designed to ensure that the cloud provider can never access plaintext data. A detailed description is in Section 3.3.Fig. 2 shows an example of the onions and layers struc-tures by considering two plaintext columns having data types int and varchar. The integer column is encrypted with Onion-Eq, Onion-Ord, and Onion-Sum, and the string col-umn is encrypted with Onion-Eq and Onion-Search. Each onion represents a column in the encrypted database struc-ture. The actual layers of all the onions are set to Rand, that guarantees the best data confidentiality level but it does not allow computations on encrypted data. When an equality check is requested on the integer column the adaptive layer removal mechanism removes the Rand layer of Onion-Eq, thus leaving Det as its new actual layer.

3.2 Metadata Structure

Metadata include all information that allows a legitimate cli-ent knowing the master key to execute SQL operations over an encrypted database. They are organized and stored at table-level granularity to reduce communication overhead for retrieval, and to improve management of concurrent SQL operations [22]. We define all metadata information associated with a table as table metadata. Let us describe the structure of a table metadata by referring to Fig. 3.Table metadata include the correspondence between the plain table name and the encrypted table name because each encrypted table name is randomly generated. Moreover, for each column of the original plain table they also include a set of column metadata containing the name and the data type of the corresponding plain column (e.g., integer, var-char, datetime). Each set of column metadata is associated with as many sets of onion metadata as the number of onions associated with the column. Onion metadata describe all the encryption information about an onion and its layers, hence they are organized in a data structure that contains the following attributes: the encrypted name is the name of the encrypted col-umn (i.e., the onion) in the encrypted cloud database; the actual encryption layer is the name of the most external layer of the encrypted data (e.g., Rand) stored in the column; the field confidentiality denotes which set of keys must be used to encrypt a column data, because only columns that share the same encryption keys can be joined; we identify three types of field confi-dentiality parameters: self denotes a private set of keys for the column, multi-column identifies the sharing of the same set of keys among two columns, database imposes the same set of keys on all columns of the same data type. the onion parameter identifies the type of onion that is used to encrypt data (e.g., Onion-Eq).Each set of onion metadata is associated with as many sets of layer metadata as the number of layers required by the onion type. Each set of layer metadata includes an encryp-tion key and a label identifying the corresponding encryp-tion algorithm. The set of encryption keys for each onion is generated according to the field confidentiality parameter imposed on each encrypted column.

3.3 Encrypted Database Management

We now describe the three main operations involved in the encrypted database management: database creation, execu-tion of SQL operations, and adaptive layer removal.

3.3.1 Database Creation

In the setup phase, the database administrator generates
a master key, and uses it to initialize the architecture metadata.
The master key is then distributed to legitimate clients. Table
creation requires the insertion of a new row in the metadata table.
For each table creation, the administrator adds a column by
specifying the column name, data type and confidentiality
parameters. These last data are the most impor-tant for
this paper because they include the set of onions asso-ciated
with the column, the starting layer denoting the actual layer
at creation time, and the field confidentiality of each onion.
If the administrator does not specify the confidential-ity
parameters of a column, they are automatically chosen
by the encryption engine with respect to some tenant's pol-icy.
Typically, the default policy assumes that each column is
associated with all the compatible onions, and the starting .

The starting layer of each onion is set to the strongest encryption algo-rithm. For example, integer columns are encrypted by default with Onion-Eq, Onion-Ord and Onion-Sum using Rand as the actual encryption layer (see Fig. 2).

3.3.2 Execution of SQL Operations

When a user/application wants to execute an operation on the cloud database, the client encryption engine analyzes the SQL command structure and identifies which tables, columns and SQL operators are involved. The client issues a request for the table metadata for each involved table, and decrypts the metadata with the master key. Then, the client determines whether the SQL operators are sup-ported by the actual layers of the onions associated with the involved columns. If required, the client issues a request for layer removal in order to support the SQL operators at runtime. By using the information stored in the table metadata, the client is able to encrypt the parame-ters of the SQL operations: tables and columns names, and constant values. The client issues this new statement called encrypted SQL operation to the cloud database which trans-parently executes it over encrypted data. The encrypted results are decrypted using information contained in the metadata.

3.3.3 Adaptive Layer Removal

The adaptive layer removal is the process that dynamically removes the external layer of an onion in order to adap-tively support SQL operations issued by legitimate clients.Let us describe the details of the adaptive layer removal mechanism by referring to the following example. We consider a table T with columns id of type int and name of type string, and a tenant client preparing to issue the following statement to the encrypted cloud database: SELECT FROM T WHERE id < 10. The client encryption engine analyzes the SQL statement, and identifies that the operation id < 10 has to be executed on the encrypted database. Then, the client reads the metadata and checks whether there is the Onion-Ord attribute associated with the column id because this is the only onion supporting the operator < . If the actual layer of Onion-Ord associ-ated with id is set to Rand, then the client dynamically invokes a stored procedure on the cloud database that removes at runtime the Rand layer of Onion-Ord of the column id, thus leaving the Ope layer exposed. The client can now encrypt the SELECT query that contains the oper-ation id < 10 and issue the encrypted query to the encrypted database, that executes it on the Ope layer of Onion-Ord. Any new SQL operation involving an order comparison on the column id does not require to invoke again the layer removal procedure because the actual layer of Onion-Ord is Ope.

The cloud database can execute the adaptive layer removal if and only if a legitimate client invokes the stored procedure and gives to it the decryption key of the most external encryption layer. As each layer has a different encryption key, the data remain encrypted and the cloud provider cannot access plaintext data. For security reasons, we also assume that the adaptive layer removal mechanism does never expose the Plain layer of an onion.

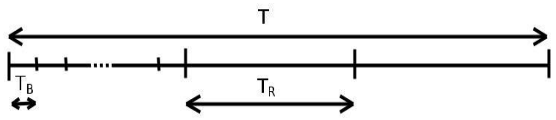


Fig. 4. Example of relationship among estimation (T), reservation (T_R) and billing (T_B) periods.

4. PERFORMANCE EVALUATION

This section aims to verify whether the overheads of adap-tive encryption represent an acceptable compromise between performance and data confidentiality for the ten-ants of cloud database services. To this purpose, we design a suite of performance tests that allow us to evaluate the impact of encryption and adaptive encryption on response times and throughput for different network latencies and for increasing numbers of concurrent clients. The TPC-C standard benchmark is used as the workload model for the database services. The experiments are carried out in Emulab [34], which provides us with a set of machines in a controlled environment. Each client machine runs the Python client prototype of our architecture on a pc3000 machine having a single 3 GHz processor, 2 GB of RAM and two 10,000 RPM 146 GB SCSI disks. The database server is PostgreSQL 9.1 running on a d710 machine hav-ing a quad-core Xeon 2.4 GHz processor, 12 GB of RAM and a 7,200 RPM 500 GB SATA disk.The current version of the prototype supports the main SQL operations (SELECT, DELETE, INSERT and

UPDATE) and the WHERE clause. We consider three TPC-C compli-ant databases having 10 warehouses: Plaintext (PLAIN) is based on plaintext data.Encrypted (ENC) refers to a statically encrypted data-base where each column is encrypted at design time with only one encryption algorithm.Adaptively encrypted (ADAPT) refers to an encrypted database in which each column is encrypted with all the onions supported by its data type (Section 3.3). In the ENC and ADAPT configurations each column is set to the highest encryption layer that supports the SQL operations of the TPC-C workload. During each TPC-C test lasting for 300 seconds, we monitor the number of executed TPC-C transactions, and the response times of all the SQL operations from the standard TPC-C work-load. We repeat the test for each database configuration (PLAIN, ENC and ADAPT) for increasing number of cli-ents (from 5 to 20), and for increasing network latencies (from 0 to 120 ms). To guarantee data consistency the three databases use repeatable read (snapshot) isolation level [35].The experiments aim to evaluate the overhead caused by static and adaptive encryption in terms of system through-put and response time. In Figs. 5 and 6, we report the num-ber of committed TPC-C transactions per minute executed on the three cloud database configurations for 5 and 20 concurrent clients, respectively. We can appreciate that in both cases, and in all other results not reported for space reasons, the throughput of the ENC database is close to that of the PLAIN database. Moreover, as the network latency increases, even the performance of the ADAPT database tends to that of the other two configurations, and it is quite

6 COST EVALUATION

In this section we demonstrate the feasibility of the pro-posed cost model by applying it to PLAIN, ENC and ADAPT configurations (see Section 5) in real cloud database services. We initially validate the usage estimation method-ology presented in Section 4.3. We then analyze how costs vary for different cloud providers and resource usages. We finally evaluate tenant's costs over a medium-term period equal to three years by considering realistic resource usage increments and cloud price reductions.

6.1 Validation of the Usage Estimation

To validate the usage estimation model, we perform several experiments based on the TPC-C benchmark.First of all, we validate the storage usage estimation model. We deploy nine TPC-C compliant databases of three different sizes: 1, 5 and 10 warehouses (the number of warehouses is the TPC-C parameter that influences the initial database size). For each size, we generate three data-base configurations: PLAIN, ENC and ADAPT. Results are summarized in Table 2. Estimated storage of PLAIN, ENC and ADAPT are calculated by using the analytical model presented in Section 4.3. For each estimated value, we report the estimation error with respect to the measured database size. Errors are expressed as a percentage. We observe that the proposed model always overestimates the database size. However, errors show that estimations are close to measured sizes. For PLAIN databases, the error is always below 2%, while for ENC and ADAPT databases the error is always between 5% and 6%.We then validate the network usage estimation model. We deploy PLAIN, ENC and ADAPT TPC-C compliant databases, each having 10 warehouses. We observe that network consumption is invariant with respect to the num-ber of warehouses, because it only depends on encryption and query workload. We measure the network usage of the PLAIN database, and we obtain an average of 7,162 Bytes per transaction. By using Equation (8), we estimate n^p ¼ k 548. Hence, we determine k ¼ 13:07. Then we use this value of k to determine the estimated network usage of ENC and ADAPT configurations. We compare these values with the experimentally measured network usages. Results are summarized in Table 3. Estimations are quite accurate, since we achieve errors of 1:2% and 1:4% for the ENC and ADAPT configurations, respectively.The validation demonstrates the efficacy of the proposed analytical usage estimation methodology in the TPC-C workload. Costs evaluations proposed in the following sec-tions are based on the same usage estimations.

6.2 Analysis of Cloud Database Costs

We analyze cloud database costs with respect to different cloud provider offers and different storage and network usages. We consider a billing period equal to one month, and 24/7 availability (730 uptime hours per month).We initially estimate the monthly costs of a cloud data-base service in the PLAIN, ENC and ADAPT configurations with respect to a plaintext storage usage of 100 GB and a plaintext network usage of 100 GB. In Table 4, we report the results for the following cloud instances: Small, Large, and High Memory: Double Extra Large from Amazon RDS [28]; Premium P1 and Premium P2 from SQL Azure [31].

7 CONCLUSION

There are two main tenant concerns that may prevent the adoption of the cloud as the fifth utility: data confidentiality and costs. This paper addresses both issues in the case of cloud database services. These applications have not yet received adequate attention by the academic literature, but they are of utmost importance if we consider that almost all important services are based on one or multiple databases.We address the data confidentiality concerns by propos-ing a novel cloud database architecture that uses adaptive encryption techniques with no intermediate servers. This scheme provides tenants with the best level of confidential-ity for any database workload that is likely to change in a medium-term period. We investigate the feasibility and per-formance of the proposed architecture through a large set of experiments based on a software prototype subject to the TPC-C standard benchmark. Our results demonstrate that the network latencies that are typical of cloud database environments hide most overheads related to static and adaptive encryption.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility,” Future Genera-tion Comput. Syst., vol. 25, no. 6, pp. 599–616, 2009.

[2] T. Mather, S. Kumaraswamy, and S. Latif, Cloud Security and Pri-vacy: An Enterprise Perspective on Risks and Compliance. Sebastopol, CA, USA: O’Reilly Media, Inc., 2009.

[3] H.-L. Truong and S. Dustdar, “Composable cost estimation and monitoring for computational applications in cloud computing environments,” Procedia Comput. Sci., vol. 1, no. 1, pp. 2175–2184, 2010.

[4] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, “The cost of doing science on the cloud: The montage example,” in Proc. ACM/IEEE Conf. Supercomputing, 2008, pp. 1–12.

[5] H. Hacigum€u€s,¸ B. Iyer, and S. Mehrotra, “Providing database as a service,” in Proc. 18th IEEE Int. Conf. Data Eng., Feb. 2002, pp. 29–38.

[6] G. Wang, Q. Liu, and J. Wu, “Hierarchical attribute-based encryp-tion for fine-grained access control in cloud storage services,” in Proc. 17th ACM Conf. Comput. Commun. Security, 2010, pp. 735–737.

[7] Google. (2014, Mar.). Google Cloud Platform Storage with server-side encryption [Online]. Available: http://googlecloudplatform. blogspot.it/2013/08/google-cloud-storage-now-provides.html.

[8] H. Hacigum€u€s,¸ B. Iyer, C. Li, and S. Mehrotra, “Executing SQL over encrypted data in the database-service-provider model,” in Proc. ACM SIGMOD Int’l Conf. Manage. Data, Jun. 2002, pp. 216–227.

[9] L. Ferretti, M. Colajanni, and M. Marchetti, “Distributed, concur-rent, and independent access to encrypted cloud databases,” IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 437–446, Feb. 2014.

[10] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, “CryptDB: Protecting confidentiality with encrypted query proc-essing,” in Proc. 23rd ACM Symp. Operating Systems Principles, Oct. 2011, pp. 85–100.

[11] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in Proc. 41st ACM Symp. Theory Comput., May. 2009, pp. 169–178.

[12] A. Boldyreva, N. Chenette, and A. O’Neill, “Order-preserving encryption revisited: Improved security analysis and alternative solutions,” in Proc. 31st Annu. Int. Conf. Adv. Cryptology, Aug. 2011, pp. 578–595.

[13] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in Proc. 17th Int. Conf. Theory Appl. Crypto-graphic Tech., May 1999, pp. 223–238.

Author Details:

Devarapalli Raja Anand,
PG Scholar, Department of Computer Science and Engineering, PACE Institute of Tech and Sciences, Vallur, Ongole, AP, India..

M.SrinivasaRao, Department of Computer Science and Engineering, PACE Institute of Tech and Sciences, Vallur, Ongole, AP, India.He Completed B.Tech and M.Tech.He have 8 Years Teaching Experiance.His Interested area is Bigdata ,Data Analytics.