Privacy Preserving Approach using Encrypted Cloud Database

Reshma Narawade
M. E.
Department of Computer Engineering
Mumbai University, Mumbai, India

Vilas Jadhav
Professor
Department of Computer Engineering
Mumbai University, Mumbai, 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 for adaptive encryption of public cloud databases that offers an interesting alternative to the trade-off 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. The common characteristics most interpretations share are on-demand scalability of highly available and reliable pooled computing resources, secure access to metered services from nearly anywhere, and displacement of data and services from inside to outside the organization. While aspects of these characteristics have been realized to a certain extent, cloud computing remains a work in progress. Moreover, our proposal guarantees data consistency in scenarios in which independent clients concurrently execute SQL queries, and the structure of the database can be modified.
Keywords – Cloud, Database, Encryption, Concurrent SQL.

I. INTRODUCTION

Cloud-based solutions for database services are now considered as an appealing alternative thanks to their scalability and availability attributes. Nevertheless, outsourcing critical data to untrusted cloud providers still poses many security concerns. One interesting research goal is to allow customers to leverage cloud infrastructures while guaranteeing data confidentiality by avoiding that cloud providers may access customer data. In a cloud context, where critical information is placed in infrastructures of untrusted third parties, ensuring data confidentiality is of paramount importance .This requirement imposes clear data management choices: original plain data must be accessible only by trusted parties that do not include cloud providers, intermediaries, and Internet; in any untrusted context, data must be encrypted. Satisfying these goals has different levels of complexity depending on the type of cloud service. There are several solutions ensuring confidentiality for the storage as a service paradigm, while guaranteeing confidentiality in the database as a service (DBaaS) paradigm is still an open research area. In this context, we propose SecureDBaaS as the first solution that allows cloud tenants to take full advantage of DBaaS qualities, such as availability, reliability, and elastic scalability, without exposing unencrypted data to the cloud provider. Although this adaptive encryption architecture is attractive 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 feasibility in a cloud context, and storage and network costs estimation. In this paper, we investigate each of these issues and we reach original conclusions in terms of prototype implementation, performance evaluation, and cost evaluation.

We implement the first proxy-free architecture for adaptive encryption of cloud databases. It does not limit the availability, elasticity and scalability of a plain cloud database, because concurrent clients can issue parallel operations without passing through some centralized component as in alternative architectures [10]. We evaluate the performance through this prototype implementation by assuming the standard TPC-C benchmark as the workload and different network latencies. Thanks to this testbed, we show that most performance overheads of adaptively encrypted cloud databases are masked by network latency values that are quite typical of a cloud scenario. Other performance evaluations carried out in [10] assumed a LAN scenario and no network latency. Some of the existing solutions for data privacy involve encryption of the entire database when storing it on the server. A client query request requires the entire database to be transferred to the client and decrypted to get the result. However, this has

the disadvantage of heavy network traffic as well as decryption cost for each query. One solution is to encrypt sensitive columns only instead of the entire database so many queries do not require the entire relation to be transmitted to the client. Reference [2] suggests using two (multiple) service providers in order to store the data. The advantage of using two servers is that the columns can be split across the two servers to satisfy privacy constraints without encrypting
the split columns. Thus, in order to satisfy privacy constraints, columns can either be split across servers or stored encrypted. Thus the goal of any decomposition algorithm is to partition the database to satisfy the following. 1. None of the privacy constraints should be violated. For a given workload, minimum number of bytes should be transferred from the servers to the client. We explain both of the above points in detail in the next section. The problem of finding the optimal partition structure for a given set of privacy constraints and query workload can be shown to be intractable. We apply heuristic search techniques based on Greedy Hill Climbing to come up with nearly optimal solutions.
In the past few years important results have been achieved in terms of energy consumption reduction, especially by improving the efficiency of cooling and power supplying facilities in data centers. The Power Usage Effectiveness (PUE) index, defined as the ratio of the overall power entering the data center and the power devoted to computing facilities, had typical values between 2 and 3 only a few years ago, while now big Cloud companies have reached values lower than 1.1. However, much space remains for the optimization of the computing facilities themselves. It has been estimated that most of the time servers operate at 10-50 percent of their full capacity [2], [3]. This low utilization is also caused by the intrinsic variability of VMs' workload: the data center is planned to sustain the peaks of load, while for long periods of time (for example, during nights and weekends), the load is much lower [4], [5]. Since an active but idle server consumes between 50 and 70 percent of the power consumed when it is fully utilized [6], a large amount of energy is used even at low utilization.

II. LITERATURE SURVEY

Guidelines on Security and Privacy in Public Cloud Computing
Cloud computing can and does mean different things to different people. The common characteristics most interpretations share are on-demand scalability of highly available and reliable pooled computing resources, secure access to metered services from nearly anywhere, and displacement of data and services from inside to outside the organization. While aspects of these characteristics have been realized to a certain extent, cloud computing remains a work in progress. This publication provides an overview of the security and privacy challenges pertinent to public cloud computing and points out considerations organizations should take when outsourcing data, applications, and infrastructure to a public cloud environment.

Distributing Data for Secure Database Services
The advent of database services has resulted in privacy concerns on the part of the client storing data with third party database service providers. Previous approaches to enabling such a service have been based on data encryption, causing a large overhead in query processing. A distributed architecture for secure database services is proposed as a solution to this problem where data was stored at multiple sites. The distributed architecture provides both privacy as well as fault tolerance to the client. In this paper we **provide algorithms for (1)distributing data: our results include hardness of approximation results and hence a heuristic greedy hill climbing algorithm for the distribution problem (2) partitioning the query at the client to queries for the various sites is done by a bottom up state based algorithm we provide.** Finally the results at the sites are integrated to obtain the answer at the client. We provide an experimental validation and performance study of our algorithms
.
Performance and cost evaluation of adaptive encryption architecture for cloud databases
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 trade-off 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 workload during a medium-term period.

SPORC: Group Collaboration using Untrusted Cloud Resources

Cloud-based services are an attractive deployment model for user-facing applications like word processing and calendaring. Unlike desktop applications, cloud services allow multiple users to edit shared state concurrently and in real-time, while being scalable, highly available, and globally accessible. Unfortunately, these benefits come at the cost of fully trusting cloud providers with potentially sensitive and important data. To overcome this strict tradeoff, we present SPORC, a generic framework for building a wide variety of collaborative applications with untrusted servers. In SPORC, a server observes only encrypted data and cannot deviate from correct execution without being detected. SPORC allows concurrent, low-latency editing of shared state, permits disconnected operation, and supports dynamic access control even in the presence of concurrency. We demonstrate SPORC's flexibility through two prototype applications: a causally-consistent key-value store and a browser-based collaborative text editor. Conceptually, SPORC illustrates the complementary benefits of operational transformation (OT) and fork* consistency. The former allows SPORC clients to execute concurrent operations without locking and to resolve any resulting conflicts automatically. The latter prevents a misbehaving server from equivocating about the order of operations unless it is willing to fork clients into disjoint sets. Notably, unlike previous systems, SPORC can automatically recover from such malicious forks by leveraging OT's conflict resolution mechanism.

Supporting Security and Consistency for Cloud Database

Typical Cloud database services guarantee high availability and scalability, but they arise many concerns about data confidentiality. Combining encryption with SQL operations is a promising approach although it is characterized by many open issues. Existing proposals, which are based on some trusted intermediate server, limit availability and scalability of original cloud database services. We propose an alternative architecture that avoids any intermediary component, thus achieving availability and scalability comparable to that of unencrypted cloud database services. Moreover, our proposal guarantees data consistency in scenarios in which independent clients concurrently execute SQL queries, and the structure of the database can be modified.

DISADVANTAGES OF EXISTING SYSTEM: ·

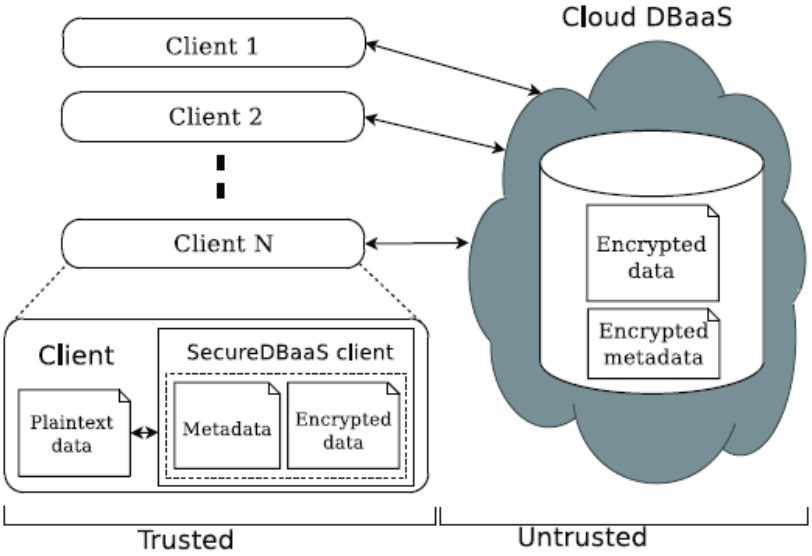
It is power consuming. · Large amount of energy is used even at low utilization.

III. PROBLEM STATEMENT AND PROPOSED SYSTEM

The ever increasing demand for computing resources has led companies and resource providers to build large warehouse-sized data centers, which require a significant amount of power to be operated and hence consume a lot of energy. SCOPE: The optimal assignment of VM's to reduce the power consumption.

This paper proposes a novel architecture that is different from any previous work in the field of security for cloud database services. Cryptographic file systems and secure storage solutions represent the earliest works to guarantee confidentiality and integrity of data outsourced to untrusted cloud storage services. We do not detail the several papers and products in this field (e.g., [6, 10, 11]) because they do not allow any computation on encrypted data. Hence they cannot be applied to the context of cloud DBaaS. Some DBMS engines offer the possibility of encrypting data at the file system level through the so called Transparent Data Encryption (TDE) feature [3, 12]. This feature makes it possible to build a trusted DBMS over untrusted storage.

However, in the DBaaS context the DBMS engine is not trusted because it is controlled by the cloud provider, hence the TDE approach is not applicable to cloud database services. The right alternative is to execute SQL operations directly on the cloud database, but avoiding that the provider obtains the decryption key. An initial solution in this direction was presented in [5]. This proposal is based on data aggregation techniques [8], that associate plaintext metadata to sets of encrypted data to allow data retrieval. However, plaintext metadata may leak sensitive information and data aggregation introduces unnecessary network overheads. Different approaches guarantee some confidentiality (e.g., [12], [13]) by distributing data among different providers and by taking advantage of secret sharing [14]. In such a way, they prevent one cloud provider to read its portion of data, but information can be reconstructed by colluding cloud providers. A step forward is proposed in [15], that makes it possible to execute range queries on data and to be robust against collusive providers. SecureDBaaS differs from these solutions as it does not require the use of multiple cloud providers, and makes use of SQL-aware encryption algorithms to support the execution of most common SQL operations on encrypted data. SecureDBaaS relates more closely to works using encryption to protect data managed by untrusted databases. In such a case, a main issue to address is that cryptographic techniques cannot be naı¨vely applied to standard DBaaS because DBMS can only execute SQL operations over plaintext data.



Query Execution Plans in Distributed Environment: When data is fragmented across multiple servers, there are two plan types used frequently to execute queries on data stored on these servers. Centralized Plans: On execution of a query, data from each server is transmitted to the client and all further processing is done at the client side. In some cases, multiple requests can go the each server but data from one server is never directly sent over to the other servers.

Encryption Details: Encryption of columns can either be deterministic or non-deterministic. A deterministic encryption is one which encrypts a column value k to the same value E(k) every time. Thus, it allows equality conditions on encrypted columns to be executed on the server. Our implementation assumes encryption on columns to be deterministic. The proposal in [4] uses encryption to control accesses to encrypted data stored in a cloud database. This solution is not applicable to usage contexts in which the structure of the database changes, and does not support concurrent accesses from multiple clients possibly distributed on a geographical scale. Our proposal is related to [8] and [13] that preserve data confidentiality in an untrusted DBMS through encryption techniques that allow the execution of SQL queries over encrypted data and are compatible with common DBMS engines. These architectures are based on an intermediate and trusted proxy that mediates all the interactions between clients and the untrusted DBMS server. The reliance on a trusted proxy that characterizes both [8] and [13] facilitates the implementation of a secure DBaaS, but causes several drawbacks. A detailed comparison between the proxy-less architecture proposed in this paper and previous architectures based on a trusted proxy is in Section 3. The architecture we propose moves away from existing solutions because it allows multiple and independent clients to connect directly to the untrusted cloud DBaaS without any intermediate server.

To the best of our knowledge this is the first paper that identifies consistency issues related to concurrent execution of queries over encrypted data and to propose viable solutions for different usage contexts, including data manipulation, modification to the database structure, and data re-encryption. The use of fully homo-morphic encryption [11] would guarantee the execution of any operation over encrypted cloud data, but existing implementations are affected by huge computational costs [11] to the extent that they would make impractical the execution time of SQL operations over a cloud database. Other encryption algorithms characterized by acceptable computational complexity support a subset of SQL operators [12], [13], [14]. For example, an encryption algorithm may support the order comparison command [12], but not a search operator [14]. The drawback related to these feasible encryption algorithms is that in a medium-long term horizon, the database administrator cannot know at design time which database operations will be required over each database column. This issue is in part addressed in [10] by proposing an adaptive encryption architecture that is founded on an intermediate and trusted proxy. This tenant's component, which mediates all the interactions between the clients and a possibly untrusted DBMS server, is fine for a locally distributed architecture, but it cannot be applied to a cloud context. Indeed, any centralized component at the tenant side prevents the scalability and availability that are among the most important features of any cloud utility service. A solution to this problem was presented in [9]: the proposed architecture allows multiple clients to issue concurrent SQL operations to an encrypted database without any intermediary trusted server, but it assumes that the set of SQL operations does not change after the database design. A first idea to integrate adaptive

encryption 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 experimental results and an original cost model. Secure DBaaS moves away from existing architectures that store just tenant data in the cloud database, and save metadata in the client machine [9] or split metadata between the cloud database and a trusted proxy [8]. When considering scenarios where multiple clients can access the same database concurrently, these previous solutions are quite inefficient. For example, saving metadata on the clients would require onerous mechanisms for metadata synchronization, and the practical impossibility of allowing multiple clients to access cloud database services independently. Solutions based on a trusted proxy are more feasible, but they introduce a system bottleneck that reduces availability, elasticity, and scalability of cloud database services.

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 original because previous papers are mainly interested to evaluate the pros and cons of porting scientific applications to a cloud platform, such as [4] focusing on specific astronomy software and a specific cloud provider (Amazon), and [3] presenting a compos able cost estimation model for some classes of scientific applications. Besides the focus on a different context (scientific vs. 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.

IV. SYSTEM DESIGN

The general architecture of a distributed secure database service, as illustrated in Figure 1 is described more in [2]. It consists of a trusted client as well as two or more servers that provide a database service. The servers provide reliable content storage and data management but are not trusted by the client to preserve content privacy. The proposed system supports adaptive encryption methods for public cloud database service, where distributed and concurrent clients can issue direct SQL operations. By avoiding an architecture based on one [10] or multiple intermediate servers between the clients and the cloud database, the proposed solution guarantees the same level of scalability and availability of the cloud service. Figure 1 shows a scheme of the proposed architecture where each client executes an encryption engine that manages encryption operations. This software module is accessed by external user applications through the encrypted database interface. The proposed architecture manages five types of information. • plain data is the tenant information; • encrypted data is stored in the cloud database; • plain metadata represent the additional information that is necessary to execute SQL operations on encrypted data; • encrypted metadata is the encrypted version of the metadata that are stored in the cloud database; • master key is the encryption key of the encrypted metadata that is distributed to legitimate clients.

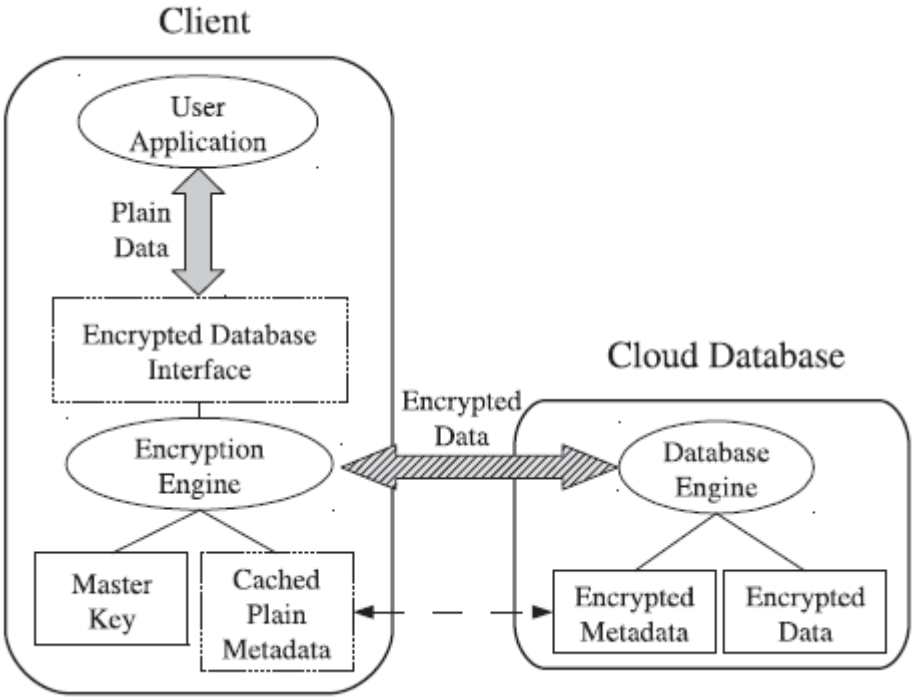


Fig. 1: Encrypted cloud database architecture

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 in plain format, whereas information is always encrypted before sending it to the cloud database. When an application issues a new SQL operation, the encrypted database interface contacts the encryption engine that retrieves the encrypted metadata and decrypts it through the master key. In order to improve performance, the plain metadata are cached locally by the client as a volatile information. After obtaining the metadata, the encryption engine is able to execute the SQL operation on encrypted data, and then application throug h the encrypted database interface. As a default behavior, SecureDBaaS uses a different encryption key for each column; hence, equal values stored in different columns are transformed into different encrypted representations. This design choice guarantees the highest confidentiality level, because it prevents an adversarial cloud provider to identify data that are repeated in different columns. However, to allow remote processing of SQL statements over encrypted data, sometimes it is required to encrypt different columns by means of the same encryption key. Common examples are the joinqueries and the foreign key constraint.

VI. CONCLUSION

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 , Windows Azure, and Xeround . 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) concurrent modifications of the database structure are supported, but at the price of high computational costs. These performance results open the space to future improvements that we are investigating.

REFERENCES

[1] Vignesh Ganapathy Google Inc. Vignesh@google.com Dilys Thomas Stanford niversity Dilys@cs.stanford.edu Tomas Feder Stanford University tomas@theory.stanford.edu Hector Garcia-Molina Stanford University hector@cs.stanford.edu Rajeev Motwani Stanford University rajeev@cs.stanford.edu Distributing Data for Secure Database Services. Http://ilpubs.stanford.edu:8090/809/1/2007-23.

[2] Ariel J. Feldman, William P. Zeller, Michael J. Freedman, and Edward W. Felten Princeton universitysporc Group Collaboration using Untrusted Cloud Resources. Https://www.usenix.org/legacy/event/osdi10/tech/full_papers/Feldman.

[3] Wayne Jansen, Timothy granceguidelines on Security and Privacy in Public Cloud Computing., http://csrc.nist.gov/publications/nistpubs/800-144/SP800-144.

[4] Luca Ferretti, Fabio Pierazzi, Michele Colajanni, and Mirco marchettiperformance and cost evaluation of an adaptive encryption architecture for cloud databases. Https://mailattachment.googleusercontent.com/attachment/u/2/?Ui=2&ik=9dca45840e&view=att&th=14e6c9071a23ad66&attid=0.1&disp=safe&realattid=f_ibug16o00&zw&saddbat=angjdj_t8mqrga2tyri7dpaudcu77cg4typyz9lxz78xsdqou7ivayo48ybvlevaee1fk4yq5gz9mkfbq_- ,IEEE Transactions on Cloud Computing,Volume:2,Issue:2, Issue Date : April-June 2014

[5] H. Hacigu¨mu¨ s,, B. Iyer, C. Li, and S. Mehrotra, “Executing SQL over Encrypted Data in the Database-Service-Provider Model,” Proc. ACM SIGMOD Int’l Conf. Management Data, June 2002.

[6] J. Li and E. Omiecinski, “Efficiency and Security Trade-Off in Supporting Range Queries on Encrypted Databases,” Proc. 19th Ann. IFIP WG 11.3 Working Conf. Data and Applications Security, Aug. 2005.

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

[8] D. Agrawal, A.E. Abbadi, F. Emekci, and A. Metwally, “Database Management as a Service: Challenges and Opportunities,” Proc. 25th IEEE Int’l Conf. Data Eng., Mar.-Apr. 2009.

[9] V. Ganapathy, D. Thomas, T. Feder, H. Garcia-Molina, and R. Motwani, “Distributing Data for Secure Database Services,” Proc. Fourth ACM Int’l Workshop Privacy and Anonymity in the Information Soc., Mar. 2011.

[10] Amazon. Amazon elastic compute cloud. Available from URL: http://www.amazon.com/b/ref=sc fe l 2/?Node=201590011&no=3435361.

[11] N. Garg, V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. In SIAM J. Comp., pages 235–251, 1996.
[12] GLB. Gramm-Leach-Bliley Act. Available from URL: Http://www.ftc.gov/privacy/privacyinitiatives/glbact.html.
[13] Google. Google apps for your domain. Available from URL:http://www.google.com/a/.
[14] A. Gupta. Improved results for directed multicut. In Proc. 14th Ann. ACM-SIAM SODA, pages 454–455, 2003.
[15] 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 Generation Computer Systems, vol. 25, no. 6, pp. 599–616, 2009.
[16] T. Mather, S. Kumaraswamy, and S. Latif, Cloud security and privacy: an enterprise perspective on risks and compliance. O'Reilly Media, Incorporated, 2009.
[17] H.-L. Truong and S. Dustdar, "Composable cost estimation and monitoring for computational applications in cloud computing environments," Procedia Computer Science, vol. 1, no. 1, pp. 2175 – 2184, 2010, iccs 2010.
[18] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The cost of doing science on the cloud: the montage example," in Proc. 2008 ACM/IEEE Conf. Supercomputing, ser. SC '08. Piscataway, NJ, USA: IEEE Press, 2008, pp. 50:1–50:12.
[19] H. Hacig¨um¨us¸, B. Iyer, and S. Mehrotra, "Providing database as a service," in Proc. 18th IEEE Int'l Conf. Data Engineering, Feb. 2002.
[20] C. Ellis and S. Gibbs. Concurrency control in groupware systems. ACM SIGMOD Record, 18(2):399–407, 1989.