Achieving Database Information Accountability in the Cloud

Kyriacos E. Pavlou [#1] and Richard T. Snodgrass [#2]

[#]*Department of Computer Science, The University of Arizona*
P.O. Box 210077, Tucson, AZ 85721–0077, USA
[1]kpavlou@cs.arizona.edu
[2]rts@cs.arizona.edu

***Abstract*— Regulations and societal expectations have recently emphasized the need to mediate access to valuable databases. Fraud occurs when a person (mostly an insider) tampers illegally with a database. Data owners would like to be assured that such tampering has not occurred, or if it does, that it will be quickly discovered. The problem is exacerbated with data stored in cloud databases such as Amazon's Relational Database Service (RDS) or Microsoft's SQL Azure Database. In our previous work we have shown that information accountability across the enterprise is a viable alternative to information restriction for ensuring the correct storage, use, and maintenance of databases on extant DBMSes. We have developed a prototype audit system (DRAGOON) that employs cryptographic hashing techniques to support accountability in high-performance databases.**

Cloud databases present a new set of problems that make extending DRAGOON challenging. In this paper we discuss these problems and show how the DRAGOON architecture can be refined to provide a more practical and feasible information accountability solution for data stored in the cloud.

I. INTRODUCTION

Corporate abuses by Enron and WorldCom have given rise to recent laws and regulations (e.g., HIPAA [1], Sarbanes-Oxley Act [2]) which require corporations to ensure trustworthy long-term retention of their routine business documents.

A challenging threat is the existence of insiders who work actively to defraud both the company and clients. The aforementioned laws as well as widespread coverage of collusion between auditors and the companies they audit resulted in increased interest within the file systems and database communities about built-in mechanisms to detect and prevent tampering, even in the presence of insider threats.

Ensuring record compliance, or information compliance in general, is usually achieved through *information restriction* which entails rendering retained records immutable and controlling access to them. This is particularly difficult for cloud databases. Even though the growing trend of moving databases to the cloud provides the benefits of ease of deployment, minimal management, scalability, and cheap cost, the user gives up fine control over the data (where it is stored and how) and relies on the cloud to provide security and privacy. Despite cloud providers' best efforts, problems abound (e.g., data loss in MS Sidekick [3] and Ma.gnolia [4], security problems with Amazon's cloud service [5]).

Weitzner et al. argue that access control and cryptography are not capable of protecting information privacy and that there is a true dearth of mechanisms for addressing effectively information leaks. They propose as an alternative that information accountability "must become a primary means through which society addresses appropriate use" [6]. *Information accountability*, in this context, states that information should be transparent so as to easily determine whether a particular use is appropriate under a given set of rules. We assert that a shift towards information accountability presents valuable advantages over information restriction in the correct use and maintenance of databases, whether located within enterprises or stored in a cloud.

In our current research we are working to show that information accountability can effectively realize appropriate use (i.e., guarantee no unauthorized modifications—insertions, deletions, updates) in high-performance databases. Our group has developed DRAGOON, (Database foRensic Analysis safeGuard Of arizONa), a prototype information accountability system which periodically audits a monitored database, detects tampering, and performs forensic analysis even in the presence of insider threats [7], [8], [9], [10]. It is scalable and highly customizable in terms of offering a tunable trade-off between level of security and monetary cost. DRAGOON only supports non-cloud-based databases. We discuss here how the existing prototype can be extended and deployed within a cloud thus providing a cloud-based information accountability solution.

II. DESCRIPTION OF DRAGOON

Within the domain of cryptographic hashing techniques used to achieve information accountability in databases, our research group removed the assumption that the system could keep a secret key that would not be seen by insiders. We proposed an innovative approach in which cryptographically-strong one-way hash functions prevent an intruder, including an auditor or an employee or even an unknown bug within the DBMS itself, from silently corrupting the audit log [11]. This is accomplished by cumulatively hashing all data manipulated by transactions as they become available to the system, thus generating a hash chain which at each time instant represents all the data in the database.

We have also designed a series of forensic analysis algorithms of increasing complexity that allow an analyst to put tight bounds on the "where" and "when" of a detected tampering [9], [12], [10].

DRAGOON features a secure master database and enterprise-level interfaces between the components of the architecture and the company's Chief Security Officer (CSO) who states enterprise-wide security policies, the database administrators (DBA) who are responsible for specific database(s), and one or more crime scene investigators (CSI) who investigate tampering and other corruptions.

Figure 1 provides a high-level structure of the tamper detection protocol and layout of the system architecture, showing how the total chain is computed during the normal processing execution phase. A user application performs transactions on the database, each of which insert, delete, and update rows of the current state. Behind the scenes, the DBMS maintains the audit log by rendering a specified relation as a *transaction-time* table. Major DBMS vendors are now supporting transaction-time tables (e.g., Oracle 11*g* [13]) so the DRAGOON architecture does not require the use of any modified DBMSes. Furthermore, the DBMS sends all record changes to a replication service in order to increase the horizontal scalability of the database. (The flow of information described is shown with magenta solid arrows.) The *replication service notarizer* hashes the data and sends the hash value to an *external digital notarization service* (EDNS) to be notarized. The notary ID returned by the EDNS along with the initially computed hash values are stored in a separate, much smaller database called the *secure master database*. This database is assumed to exist in a different physical location from the DBMS under audit. It is also assumed to be located along with the replication service in a secure site. (The flow of information described is shown with red dotted arrows.) Validation of the monitored database is executed to check for potential tampering. It involves recomputing a hash value over the same data at a later point in time and comparing this new hash value with the one previously notarized. A mismatch between the two values indicates tampering. The CSO and DBA GUIs specify which databases are monitored and specify how often notarizations and validations occur. If the monitored database has been compromised the CSI initiates the forensic analysis phase which utilizes the forensic analysis algorithms. The results of the forensic analysis are reported back to the CSI (information flow shown with thick green arrows).

III. RESEARCH PROBLEM

We propose extending the DRAGOON architecture with the capability of being utilized in a cloud computing service and to support the monitoring of multiple databases. This extension will render information accountability-based security viable, protect against a variety of threats, and successfully deal with the aftermath of information restriction failure concerning data stored in the cloud. It will also demonstrate the advantages of information accountability over information restriction in the area of correct storage, use, and maintenance of cloud databases.

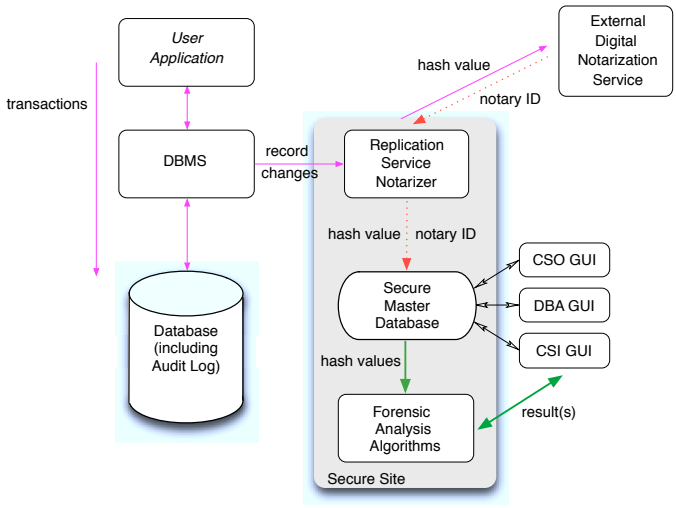

Fig. 1. Total Chain Computation during Normal Processing.

A. *Migrating* DRAGOON *to the Cloud*

There are several advantages to extending the DRAGOON architecture to allow the DBMS and associated database to reside in the cloud. Such a move greatly increases scalability whereby multiple databases and multiple DBMSes can be supported. It also results in more versatility and a lower set-up and maintenance overhead.

However, a naïve approach of moving all components in Figure 1, other than the user application and the EDNS, to the cloud is not desirable. It decreases security by introducing a big insider threat from the cloud itself: the cloud vendor controls the DBMS, the database storage, the forensic analysis, and the secure master database. This also leads to a dubious business relationship between the cloud vendor and the EDNS, in which the former exerts a disproportionate influence on the latter. The result is a reemerged insider threat, from a different perspective, the very problem DRAGOON was developed to address.

We now propose an enhanced architecture that can leverage the cloud to provide increased security. Figure 2 presents a preliminary enhanced architecture for deploying the monitored database in the cloud. This architecture identifies four different domains of control. The first is the domain of user applications and of the GUIs, controlled by the company that has administrative control of the monitored database. The second is the domain of the cloud provider where the monitored database resides (cloud **A**). The third domain is that of the cloud provider where the DRAGOON database resides (cloud **B**). It is best that this be a separate vendor. The final domain is the EDNS, which should not use cloud services from either of these vendors(!)

The flow of information is similar to the normal processing and tamper detection phases described previously (forensic analysis is not shown, but is similar to before). The magenta-colored dashed arrow depicts the stream of data being repli-

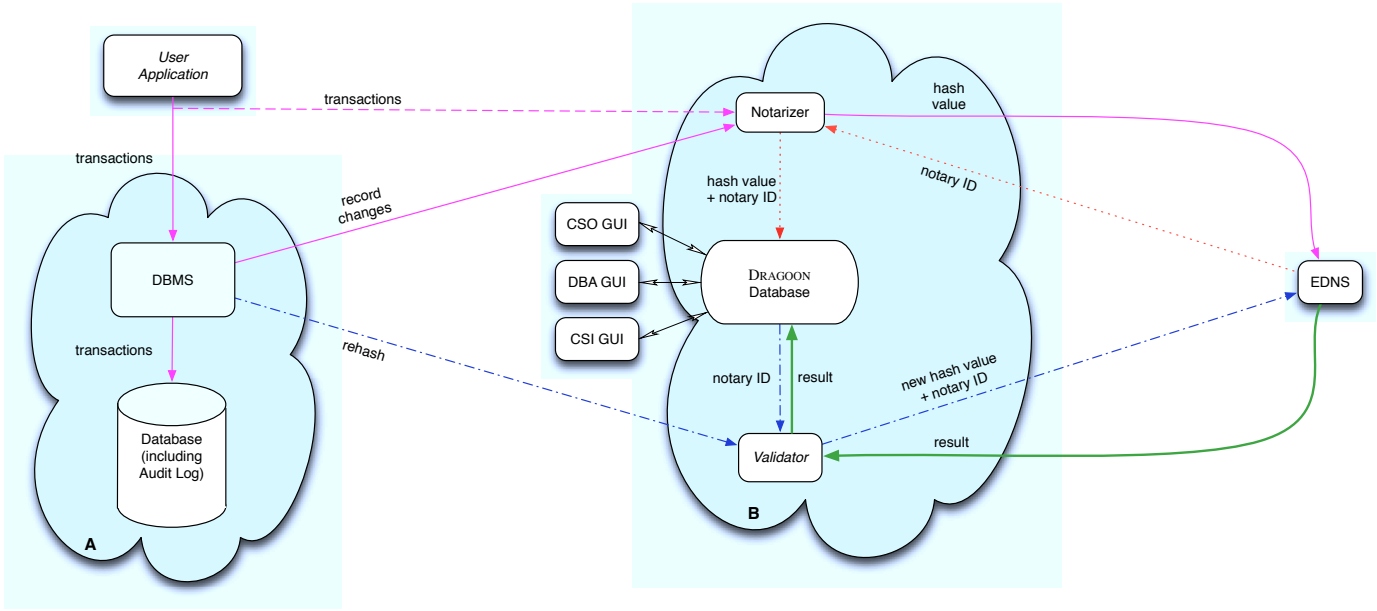

Fig. 2. Normal Processing, Computation of Total Chain, and Tamper Detection of Database in the Cloud.

cated and sent to the notarizer before it reaches the cloud. An alternative architecture has the DBMS sending the changed data to notarizer.

Note that the DRAGOON database is much smaller than the database being monitored as it only stores hash values, timestamps, notary IDs, and settings from the GUIs.

B. Threat Analysis

We identify three separate sources of threat: an external threat, which we assume does not arise from any of the four control domains, and two insider threats for the two cloud vendors. Observe that the proposed architecture eliminates insider threats from the company that owns the monitored database because users have no physical access to the data and must utilize the cloud interface instead.

Tampering by an external threat, e.g., a black hat, is detected via record changes communicated from the DBMS to the notarizer and the notarization/validation process of cloud **B** and the EDNS.

An insider threat from cloud **A** is detected via transactions communicated from the cloud API, also to the notarizer. Data are sent to the notarizer before they are stored on cloud **A**. If the stored data are tampered by an insider they yield a different hash value during validation from the hash value previously computed by the notarizer. This additional channel of information (i.e., the dashed arrow) safeguards the security and integrity of cloud **A**. Hence, migrating DRAGOON to the cloud can provide guarantees about the cloud service itself.

If an insider threat from cloud **B** tampers the monitored database, this scenario reduces to the external threat case. If the insider only tampers the DRAGOON database then this tampering is again detected by a mismatch in the hash values or the inability of the EDNS to locate the provided notary ID. More sophisticated attacks are possible since the DRAGOON database is under the control of a single domain. However, its security, especially the settings and the possibility of spoofing the notarizer and validator, can be enhanced by monitoring it, in the same way as the originally-monitored database.

In general, the fact that there are four separate companies involved greatly increases the difficulty of effecting an undetected tampering. To do so requires colluding insiders at four domains as well as the ability to fake the hashing of the tampered data and subsequent insertion into the security infrastructure (e.g., Merkle tree) of the EDNS.

C. Scalability and Performance

If the transaction stream is replicated at the user application level then we have to deal with concurrency issues. In particular, the notarizer requires information on how to order the transactions (information that is usually provided by the DBMS) so they can be hashed. This is an open problem.

Supporting multiple user clients is not an issue if we trust cloud **A**. However, if the database is replicated in the cloud as well as distributed, then stale data might be hashed by the notarizer and thus additional effort is needed to ensure that transactions are hashed in the correct order.

Another open design issue is whether hashing occurs at the application level or by the notarizer in cloud **B**.

A final open question is how to distribute the notarizer, validator, and the DRAGOON database across geographically distributed processing nodes.

IV. RELATED WORK

The published work featured below describes the state-of-the-art in the fields of database tamper detection and forensics, and secure storage in the cloud.

Basu presents a method of forensic tamper detection and localization of corrupted data in SQL Server [14]. His method

does not provide tamper-prevention measures, but shows how tampering can be detected and the affected data localized. The solution is based on creating an interwoven chain of hash values used by a detection algorithm to determine if a particular audit log table row is modified, inserted, or deleted. Although this method has advantages (e.g., no special deployment strategy required), it suffers from the use of non-cryptographically strong hash functions, and the limited forensic strength of the detection algorithm.

Guo, Jajodia, Li, and Liu formulated a fragile watermarking scheme for databases [15], [16]. Their scheme is based on a watermark that is *invisible* (watermark does not distort data) and can be *blindly verified* (original unmarked relation is not required for verification). The watermark depends on the hash values of the tuples' primary key value, their attribute values, and a secret embedding key. During verification, the extracted watermark indicates the locations of alterations down to the granularity of a range of tuples. Unauthorized tampering of data is detected when forensic analysis yields a mismatch between the originally-embedded watermark and the extracted watermark.

A recent attempt was made to address the issue of secure storage in the cloud with the development of DEPSKY [17]. This system provides storage in a cloud-of-clouds that improves the availability, integrity and confidentiality of information stored in the cloud. It achieves this by employing a combination of Byzantine quorum system protocols, cryptography, secret sharing, and erasure codes which make the whole system fault-tolerant. However, DEPSKY's objectives do not include information accountability.

Wang et al.'s work shares the goal of data integrity and uses cryptographic hashing to enable "a third-party auditor (TPA) on behalf of the could client to verify the integrity of the dynamic data stored in the cloud" [18]. However, this work is theoretical, not providing an architecture nor incorporating forensic analysis should tampering be detected by the TPA.

V. CONTRIBUTIONS

The DRAGOON extension we have proposed here allows organizations using cloud DBMSes to be assured that tampering, including by cloud vendor insiders, has not occurred. Further, the forensic analysis capabilities of the proposed system identifies tampering

Such an architecture will be valuable to virtually all applications storing their valuable data in the cloud. For example, cloud-based DRAGOON can help ensure record compliance for financial and medical institutions. It can also be of use to biosciences labs because it can ensure non-deviation from standard operating protocols thus providing a certain type of provenance for their final results.

The techniques proposed will not just protect data but also through continuous assurance will be able to detect corruption shortly after tampering as well as automate to a great extent the work required in the aftermath of a database corruption. This saves both time and money for those affected. The techniques will also highlight the advantages over approaches relying heavily on information restriction through either hardware which can have prohibitive costs for small institutions, have a limited shelf-life and are relatively complex; or cryptography which does not adequately offer remedies after a leak.

The resulting architecture is scalable and customizable in terms of offering a tunable trade-off between level of security and monetary/forensic cost, as well as strong guarantees, which are of interest to all cloud database users. That said, there is more to be done to realize this approach and to address the identified open problems.

ACKNOWLEDGMENT

The authors would like to thank Nirav Merchant, Somu Perianayagam, Radu Sion, and Marianne Winslett for numerous and very helpful discussions on compliant databases and cloud computing. NSF grants IIS-0415101, IIS-0803229, and a grant from Surety, LLC provided partial support for this work.

REFERENCES

[1] U.S. Department of Health & Human Services. (2006) The Health Insurance Portability and Accountability Act (HIPAA). [Online]. Available: http://www.cms.gov/HIPAAGenInfo/

[2] "U.S. Public Law No. 107–204, 116 Stat. 745. The Public Company Accounting Reform and Investor Protection Act," 2002.

[3] D. Sarno. (2009) Microsoft says lost sidekick data will be restored to users. [Online]. Available: http://latimesblogs.latimes.com/technology/2009/10/microsoft-says-lost-sidekick-data-will-be-restored-to-users.html

[4] E. Naone. (2009) Are we safeguarding social data? [Online]. Available: https://www.technologyreview.com/blog/editors/22924/

[5] A. R. Hickey. (2011) Researchers uncover 'massive security flaws' in Amazon cloud. [Online]. Available: http://www.crn.com/news/cloud/231901911/researchers-uncover-massive-security-flaws-in-amazon-cloud.htm

[6] D. J. Weitzner, H. Abelson, T. Berners-Lee, J. Feigenbaum, J. Hendler, and G. J. Sussman, "Information accountability," *Communications of the ACM*, vol. 51, no. 6, pp. 82–87, June 2008.

[7] K. E. Pavlou. (2011) SIGMOD/IDAR Workshop 'Database forensics in the service of information accountability'. [Online]. Available: https://sites.google.com/a/ualberta.ca/idar2011/

[8] DRAGOON. (2011). [Online]. Available: http://www.cs.arizona.edu/projects/tau/dragoon/

[9] K. E. Pavlou and R. T. Snodgrass, "Forensic analysis of database tampering," in *Proc. ACM SIGMOD'06*, June 2006, pp. 109–120.

[10] ——, "Forensic analysis of database tampering," *ACM Transactions on Database Systems*, vol. 33, no. 4, pp. 30:1–30:47, November 2008.

[11] R. T. Snodgrass, S. S. Yao, and C. Collberg, "Tamper detection in audit logs," in *Proc. VLDB'04*, September 2004, pp. 504–515.

[12] K. E. Pavlou and R. T. Snodgrass, "The tiled bitmap forensic analysis algorithm," *IEEE Trans. Knowledge Data Eng.*, vol. 22, no. 4, pp. 590–601, April 2010.

[13] Oracle Corp., "Workspace Manager Developer's Guide 11*g* Release 1 (11.1)," Aug. 2008.

[14] A. Basu. (2006) Forensic Tamper Detection in SQL Server. [Online]. Available: http://www.sqlsecurity.com/images/tamper/tamperdetection.htm

[15] H. Guo, Y. Li, A. Liu, and S. Jajodia, "A fragile watermarking scheme for detecting malicious modifications of database relations," *Inf. Sci.*, vol. 176, no. 10, pp. 1350–1378, 2006.

[16] Y. Li, H. Guo, and S. Jajodia, "Tamper Detection and Localization for Categorical Data Using Fragile Watermarks," in *Proc. 4th ACM-DRM*, 2004, pp. 73–82.

[17] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "DEPSKY: Dependable and secure storage in a cloud-of-clouds," in *The 6th ACM SIGOPS/EuroSys European Systems Conference*, April 2011.

[18] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security and cloud computing," in *Poc. ESORICS'09*, 2009, pp. 335–370.