DRAGOON: An Information Accountability System for High-Performance Databases

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, even access by insiders. Fraud occurs when a person, often an insider, tries to hide illegal activity. Companies would like to be assured that such tampering has not occurred, or if it does, that it will be quickly discovered and used to identify the perpetrator. At one end of the compliance spectrum lies the approach of *restricting access to information* and on the other that of *information accountability*. We focus on effecting information accountability of data stored in high-performance databases.**

The demonstrated work ensures appropriate use and thus end-to-end accountability of database information via a *continuous assurance technology* based on cryptographic hashing techniques. A prototype tamper detection and forensic analysis system named DRAGOON was designed and implemented to determine when tampering(s) occurred and what data were tampered with. DRAGOON is scalable, customizable, and intuitive.

This work will show that information accountability is a viable alternative to information restriction for ensuring the correct storage, use, and maintenance of databases on extant DBMSes.

I. INTRODUCTION

Corporate abuses by Enron and WorldCom have given rise to recent regulations which require many corporations to ensure trustworthy long-term retention of their routine business documents. The US alone has over 10,000 regulations [1] that mandate how business data should be managed [2], [3], including the Health Insurance Portability and Accountability Act: HIPAA [4], the Sarbanes-Oxley Act [5], the 1997 U.S. Food and Drug Administration (FDA) regulation “21 CFR Part 11” [6], and other laws requiring audit logs [1].

A challenging threat is the existence of insiders who work actively to defraud both the company and the 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.

Compliant records are those required by myriad laws and regulations to follow certain “processes by which they are created, stored, accessed, maintained, and retained” [1]. It is common to use Write-Once-Read-Many (WORM) storage devices to preserve such records [7]. The original record is stored on a write-once optical disk. As the record is modified, all subsequent versions are also captured and stored, with metadata recording the timestamp, optical disk, filename, and other information on the record and its versions. But these solutions do not work with high-performance databases.

This way of ensuring record compliance can be described as *information restriction* which entails rendering retained records immutable and controlling access to them. The means of addressing security and compliance can be viewed as a spectrum. If information restriction lies at one end of the spectrum, the question which inevitably arises is, what lies at the other end? 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 effectively addressing information leaks. They propose as an alternative that information accountability “must become a primary means through which society addresses appropriate use” [8]. *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 particular area of correct storage, use, and maintenance of databases. An information accountability approach to database security is cheaper, can deal with restriction failure, can render complex security problems tractable, and can protect against a variety of threats including insider threats.

Information accountability has been tried and tested successfully since ancient times [9]. It has been applied in modern times and in many varied areas including the Fair Credit Reporting Act of 1970 [10]. Under this act strict rules are imposed not on the collection of data or their analysis but on the way the data or the result of the analysis (e.g., credit reports) can be used.

Lest a conclusion be drawn that accountability is only appropriate for information with a low associated risk, we offer as an example the widespread use of simple wire-loops as tamper-indicating seals for nuclear safeguarding [9].

As part of our current research, in which we are working to show that information accountability can effectively guarantee no unauthorized modifications—insertions, deletions,

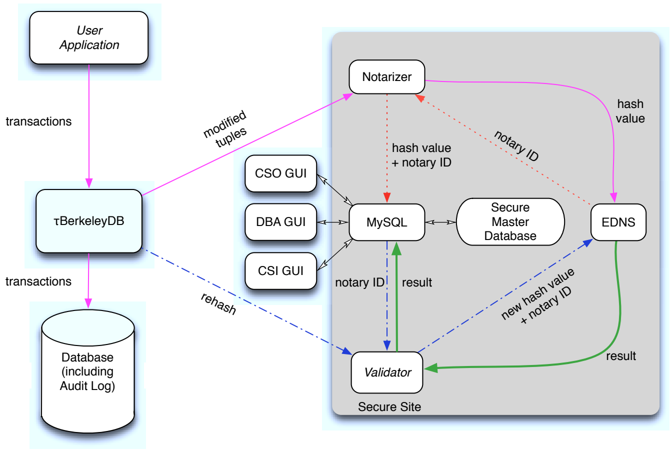


Fig. 1. Dragoon Architecture, Normal Processing, and Tamper Detection.

updates—in high-performance databases, we have designed and implemented a prototype system called Dragoon which provides tamper detection and forensic analysis capabilities.

Dragoon employs *continuous assurance technology*, defined as "technology-enabled auditing which produces audit results simultaneously with, or a short period of time after, the occurrence of relevant events" [11], to achieve a meaningful operationalization of information accountability.

II. The Dragoon System

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 using cryptographically-strong one-way hash functions allows us to prevent an intruder, including an auditor or an employee or even an unknown bug within the DBMS itself, from silently corrupting the audit log [12].

Dragoon (Database foRensic Analysis safeGuard Of arizONa), a prototype auditing system, was implemented using the above approach as its basis. It has been expanded to include forensic analysis and to be highly customizable in terms of offering a tunable trade-off between level of security and monetary cost. It is lightweight and scalable and is able to adequately address aspects of information accountability even in the presence of insider threats.

A. System Architecture and Tamper Detection

Dragoon's architecture along with the flow of information during normal processing and tamper detection are illustrated in Figure 1.

A user application performs transactions on the monitored database, each of which insert, delete, and update rows of the current state. Behind the scenes, τBerkeleyDB (an extension of BerkeleyDB with transaction-time support) maintains the audit log by rendering a specified relation as a *transaction-time* table. On each modification of a tuple, the DBMS is responsible for *hashing* the tuples. (The flow of information described is shown with magenta solid arrows.) When a transaction commits, the DBMS obtains a timestamp and computes a *cryptographically strong one-way hash function* of the tuple data and the timestamp. The hash values obtained from the different transactions are cumulatively hashed and thus linked with each other in order to create a hash chain which at each time instant represents all the data in the database. This chain is termed the *total hash chain*.

A module called a *notarizer* periodically sends that hash value, as a digital document, to an *external digital notarization service* (EDNS) such as Surety (www.surety.com), which notarizes the hash and returns a notary ID. The notary ID along with the initially computed hash values are stored in a separate smaller MySQL-managed database. (The flow of information described is shown with red dotted arrows.) This database, termed the *secure master database*, is assumed to exist in a secure site which is in a different physical location from the monitored database [13].

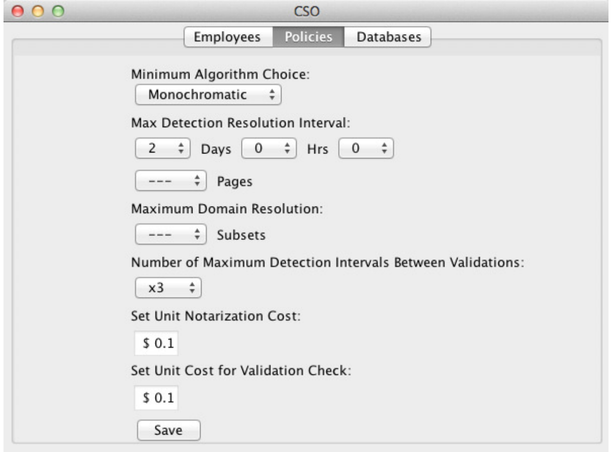
Figure 1 also shows how tamper detection is achieved. At a later point in time an application called the *validator* initiates a scan of the entire database and hashes the scanned data along with the timestamp of each tuple. The validator retrieves the previously stored (during notarization) notary ID from the secure master database and sends the information to the EDNS (information flow shown with blue dotted-dashed arrows). The EDNS then locates the notarized document/hash using the provided notary ID and checks if the old and the new hash values are consistent. If not, then the monitored database has been compromised. The validator stores the validation result in the secure master database (information flow shown with thicker green solid arrows). The computation of the total chain, together with the periodic notarizations and validations comprise the *normal processing* execution phase of the system.

The system in its current form is scalable since it can support multiple databases managed by the same DBMS, using the same tamper detection and forensic analysis infrastructure. Moreover, it can be scaled to support databases managed by different DBMSes. The performance of the prototype has been evaluated elsewhere [12]. The effect of number of transactions, the transaction and tuple sizes was studied. The maximum overhead observed was 15% but only for small tuples (10 bytes)—this overhead diminishes as the tuple size increases.

B. Dragoon *Interface*

Dragoon provides graphical 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 shows how the three GUIs interact with the MySQL secure master database which, in addition to the hash values and notary IDs, stores the system configuration. Specifically, the CSO GUI is used to assign DBA and CSI roles to employees, designate which databases are monitored, and along with the DBA GUI, set how often notarizations and validations occur.

For example, Figure 2 shows the "Policies" tab of the CSO GUI. The CSO can use this to set the minimum forensic analysis algorithm that all monitored databases must support—in this case the Monochromatic Forensic Analysis Algorithm. She can also configure the system so that notarizations happen every two days (Max Detection Resolution Interval) while validations occur every three notarizations (i.e., every six days). Finally, the unit cost for each notarization and validation can be customized (here both are set to 10 cents) since each interaction with the EDNS costs real money. (A cost model for this has been developed and incorporated into DRAGOON [14].)



Minimum Algorithm Choice:
Monochromatic

Max Detection Resolution Interval:
2 Days 0 Hrs 0

--- Pages

Maximum Domain Resolution:
--- Subsets

Number of Maximum Detection Intervals Between Validations:
x3

Set Unit Notarization Cost:
$ 0.1

Set Unit Cost for Validation Check:
$ 0.1

Fig. 2. The Policies Tab in the CSO GUI.

If we determine during validation that the monitored database has been compromised the CSI GUI initiates the forensic analysis phase which utilizes the forensic analysis algorithms. The results of the forensic analysis are reported back to the CSI and displayed in a graphical manner.

C. Forensic Analysis

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 [15], [14], [16].

Figure 3 shows a *corruption diagram* which is a graphical representation of the tampering(s) (denoted by CE for "corruption event") in terms of the temporal-spatial dimensions of a database. The diagonal line represents the cumulative total hash chain with notarizations (*NE*) and validations (*VE*) occurring every two and six days respectively (and according to the CSO GUI settings). The y-axis represents clock time while the x-axis measures transaction commit time (which can provide an indirect means of characterizing the "where"). In basic forensic analysis (Monochromatic Algorithm), once tampering is detected, the algorithm revisits previously notarized parts of the hash chain and validates them. This gradually restricts the set of data where the tampering occurred ("where"). The time interval between a successful and a failed validation puts bounds on the time of tampering ("when"). The shaded area represents the spatial and temporal bounds of the corruption. This shaded area is the result of forensic analysis and is displayed in a similar way by the CSI GUI. More sophisticated algorithms (e.g., the a3D Algorithm) employ more hash chains in order to make the bounds even tighter.

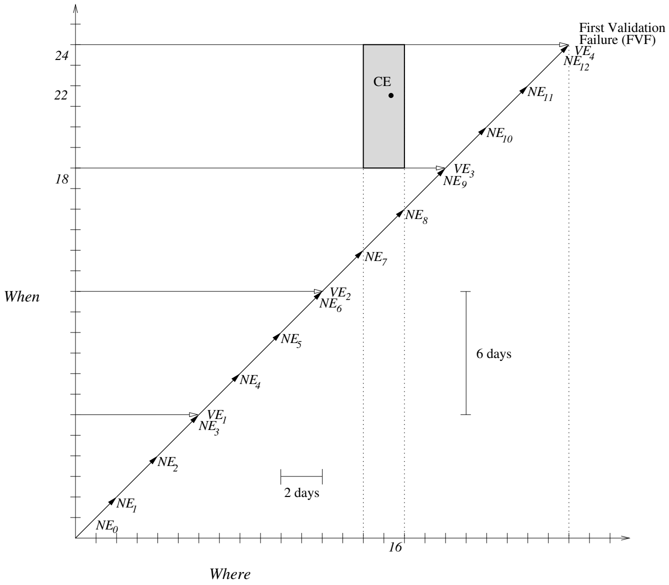


Fig. 3. Corruption Diagram for Monochromatic Algorithm.

III. DEMONSTRATION

The DRAGOON demonstration will be framed in terms of a real-world scenario formulated in a corporate setting. In general, interactive parts will be interleaved with expository interludes which will explain the system details and theory behind what is being demonstrated.

The demo will begin with a brief discussion of the concept of information accountability along with motivating examples (e.g., Egyptian *bullae*, HIPAA, Creative Commons licensing).

This will be followed by the introduction of the real-world scenario featuring a company which requires an enterprise-wide solution for guaranteeing the appropriate use of its relational data stored across various databases. The objectives of the solution sought will be clearly delineated.

Volunteers from the audience will be asked to play the roles of the CSO, DBA, CSI, and adversary. The DRAGOON system interface will then be introduced and using the GUIs, the volunteers will be added as employees to the company. Using the CSO GUI we will formally assign DBA and CSI duties to the employees.

The next step will be to discuss the basic architecture and tamper detection protocol of DRAGOON as shown in Figure 1. We will describe how the modified tuples are hashed and how the total hash chain is created and maintained. Moreover, we will cover the protocol governing notarization of the current hash value of the total chain, as well as how validation of the database is initiated and executed.

The CSO will then decide which of the existing databases in the company will be monitored and a global minimum security policies for all monitored databases will be set. Using the DBA GUI, adjustments will be made to the global policies to better suit the particular database administered by the DBA. We will demo monitored databases differing in their notarization and validation frequencies, notarization/validation unit costs, as well as forensic analysis algorithms used. In particular, two such algorithms will be demonstrated: the Monochromatic Algorithm and the more complex a3D Algorithm.

The notarization server (EDNS) will be started along with the considered database. The database will be processing transactions, performing the hashing of modified tuples, with the resulting hash values periodically being notarized and validated. Visual cues attesting to these will be provided.

The adversary will then tamper the database using a script. The contents of the database will be examined to verify the unauthorized change. The first validation after tampering will fail and alert the CSO, DBA, and CSI that the database has been compromised.

The forensic analysis phase will then be described with special emphasis on how the algorithms work. Both the Monochromatic and a3D algorithms will be analyzed. We will use animated corruption diagrams similar to the one in Figure 3 to facilitate the discussion. These corruption diagrams will be compared to the ones generated by the CSI GUI in order to see how the theoretical spatial and temporal bounds on the tampering compare to the ones resulting from forensic analysis. Furthermore, the bounds on the tampering in CSI-generated corruption diagrams will be compared between monitored databases using different notarization/validation and algorithm policies in order to see the effect of the policies on the tightness of the bounds.

The overall cost of normal processing and forensic analysis will also be compared between monitored databases in order to demonstrate the balance between higher level of security (better bounds on the tampering) and monetary cost.

If time permits, we will discuss how multiple tamperings in a single database can be detected and how other types of tampering, like timestamp and schema corruption, can give rise to more complex issues.

IV. Contributions

DRAGOON provides guarantees against insider threats, scalability, forensic analysis tools, and can be extended to an enterprise-wide solution. Hence, it is valuable and applicable to a variety of sectors. For example, these features can help ensure record compliance for financial and medical institutions. They can serve as an unbiased witness to any type of database storing sensitive information. These may include court-submitted data from police databases or biological research results. The unbiased witness can be of particular use to bioscience labs because it can ensure non-deviation from protocols thus providing a certain type of provenance for their final results.

The developed system does not just protect data but also through continuous assurance is 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 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.

V. Conclusion

A prototype audit system named DRAGOON was designed and implemented with tamper detection and forensic analysis capabilities. DRAGOON is highly customizable in terms of offering a tunable trade-off between level of security and monetary cost. It is scalable and is able to adequately address aspects of information accountability even in view of insider threats.

Acknowledgment

The authors would like to thank Peter Downey, Yifeng Li, Nirav Merchant, Soumyadeb Mitra, Radu Sion, Joseph Watkins, and Marianne Winslett for providing valuable feedback. NSF grants IIS-0415101, IIS-0803229, and a grant from Surety, LLC provided partial support for this work.

References

[1] P. A. Gerr, B. Babineau, and P. C. Gordon, "Compliance: The effect on information management and the storage industry," Enterprise Storage Group, Research Report, May 2003.

[2] C. C. Chan, H. Lam, Y. C. Lee, and X. Zhang, *Analytical Method Validation and Instrument Performance Verification*. Wiley-IEEE, 2004.

[3] G. Wingate, Ed., *Computer systems validation: Quality Assurance, Risk Management, and Regulatory Compliance for Pharmaceutical and Healthcare Companies*. Informa Health Care, 2003.

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

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

[6] F.D.A. (2003) Title 21 Code of Federal Regulations (21 CFR Part 11) Electronic Records; Electronic Signatures. [Online]. Available: http://www.fda.gov/ICECI/EnforcementActions/default.htm

[7] Q. Zhu and W. W. Hsu, "Fossilized index: The linchpin of trustworthy non-alterable electronic records," in *Proc. ACM SIGMOD'05*, 2005, pp. 395–406.

[8] 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.

[9] R. G. Johnston, "Tamper-indicating seals," *American Scientist*, vol. 94, no. 6, pp. 515–524, Nov–Dec 2006.

[10] 15 U.S.C.1681. (1970) Fair Credit Report Act. [Online]. Available: http://www.law.cornell.edu/uscode/15/usc_sup_01_15_10_41_20_III.html

[11] M. Alles, A. Kogan, and M. Vasarhelyi, "Black box logging and tertiary monitoring of continuous assurance systems," *Information Systems Control Journal*, vol. 1, 2003.

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

[13] M. Malmgren. (2007) An infrastructure for database tamper detection and forensic analysis. [Online]. Available: http://www.cs.arizona.edu/projects/tau/tbdb/MelindaMalmgrenThesis.pdf

[14] K. E. Pavlou and R. T. Snodgrass, "Forensic analysis of database tampering," *ACM Transactions on Database Systems*, vol. 33, no. 4, pp. 30:1–30:47, November 2008.

[15] ——, "Forensic analysis of database tampering," in *Proc. ACM SIGMOD'06*, June 2006, pp. 109–120.

[16] ——, "The tiled bitmap forensic analysis algorithm," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 4, pp. 590–601, April 2010.