Measuring the Forensic-Ability of Audit Logs for Nonrepudiation

Jason King
Department of Computer Science
North Carolina State University
890 Oval Drive
Raleigh, NC, USA, 27695
http://www4.ncsu.edu/~jtking
jtking@ncsu.edu

***Abstract*— Forensic analysis of software log files is used to extract user behavior profiles, detect fraud, and check compliance with policies and regulations. Software systems maintain several types of log files for different purposes. For example, a system may maintain logs for debugging, monitoring application performance, and/or tracking user access to system resources. *The objective of my research is to develop and validate a minimum set of log file attributes and software security metrics for user nonrepudiation by measuring the degree to which a given audit log file captures the data necessary to allow for meaningful forensic analysis of user behavior within the software system*. For a log to enable user nonrepudiation, the log file must record certain data fields, such as a unique user identifier. The log must also record relevant user activity, such as creating, viewing, updating, and deleting system resources, as well as software security events, such as the addition or revocation of user privileges. Using a grounded theory method, I propose a methodology for observing the current state of activity logging mechanisms in healthcare, education, and finance, then I quantify differences between activity logs and logs not specifically intended to capture user activity. I will then propose software security metrics for quantifying the forensic-ability of log files. I will evaluate my work with empirical analysis by comparing the performance of my metrics on several types of log files, including both activity logs and logs not directly intended to record user activity. My research will help software developers strengthen user activity logs for facilitating forensic analysis for user nonrepudiation.**

***Index Terms*—forensics, metric, security, nonrepudiation, logging, software logs, grounded theory**

I. Introduction

Microsoft defines repudiation threats as threats "associated with users who deny performing an action without other parties having any way to prove otherwise." *Nonrepudiation*, therefore, refers to the ability of a system to prevent repudiation threats – users cannot perform an action and later deny performing that action [1]. Software developers can facilitate nonrepudiation in software systems through the implementation of logging mechanisms that provide accurate, thorough traces of user activity. The act of detecting anomalous user activity is a reactive process performed after the anomalous user activity has already occurred. However, designing and implementing adequate activity logging mechanisms is a proactive means of improving a software system's ability to counter repudiation threats in the future. Individuals can be legally prosecuted for inappropriate use of a software system. The United States Computer Emergency Readiness Team (US-CERT), part of the United States Department of Homeland Security, defines *computer forensics* as "the discipline that combines elements of law and computer science to collect and analyze data from computer systems, networks, wireless communications, and storage devices in a way that is admissible as evidence in a court of law" [2]. Forensic analysis of software log files helps extract common software usage patterns and identify misuse of sensitive, protected data or resources [3] [4].

With no widely adopted standard for activity logging mechanisms [5] [6], software developers may overlook user events that should trigger a log entry [7], misunderstand the purpose of activity logging, or mistakenly assume debug or performance logs fulfill the same purpose as activity logging. *The objective of my research is to develop and validate a minimum set of log file attributes and software security metrics for user nonrepudiation by measuring the degree to which a given audit log file captures the data necessary to allow for meaningful forensic analysis of user behavior within the software system.*

To facilitate nonrepudiation in a software system, an activity logging mechanism should record relevant user activity such as creating, viewing, modifying, and deleting system resources. Activity logs should also track security events, such as addition/revocation of user privileges, session timeouts, and authentication attempts. We previously analyzed logging mechanisms of three open-source electronic health record (EHR) systems and found inconsistent, inadequate log files for promoting user nonrepudiation [7]. Log entry content across the three EHR systems varied. Only one EHR system recorded detailed SQL queries as part of each log entry. Similarly, only one EHR system recorded security events related to assigning or revoking user privileges. Overall, the EHR systems studied only recorded an average of 12.5% of general auditable events that involved transactions with protected health information (PHI). In healthcare, logging interactions with PHI is legally required in the United States by the Health Insurance

978-1-4673-3076-3/13/$31.00 © 2013 IEEE

Portability and Accountability Act of 1996 [8]. The inconsistency and inadequacy among log files of the EHR systems studied led us to question whether the EHR system developers understood the forensic purpose of user activity logs for nonrepudiation. We also question whether the EHR developers understand what user activity should be recorded for maintaining an accurate, complete trace of user activity for analysis. Based on the inadequate log files in my previous study, forensic analysis on the EHR system logs would be difficult and unfruitful since many user events were not logged [7], resulting in gaps within recorded user activity. By identifying a minimal set of log file attributes and by developing software security metrics to quantify forensic usefulness of software logs, I hope to improve compliance with regulations and policies in domains that manage sensitive, protected resources and in the forensic-ability of all systems for which repudiation is important.

In my research, I use a grounded theory method to observe the current state of software activity logs; technical documentation; and standards, policies, and regulations in healthcare, education, and the financial industries. Next, through user studies, I will construct a benchmark log file structure that facilitates nonrepudiation of user behavior in a software system. Data collected through the user studies will help me propose security metrics for quantifying the forensic-ability of a given log file. Finally, I will validate the proposed forensic-ability metrics. The contributions of my research will include:

- A suggested minimum set of log file attributes (including both log entry content such as username and timestamp, and a list of events that should trigger log entries) to facilitate nonrepudiation.
- A set of software security metrics for measuring the degree of nonrepudiation enabled by software log files

Related work on forensic analysis is discussed in Section 2. My research direction is discussed in Section 3. My methodology, plans for evaluation, and current progress are discussed in Section 4. Section 5 concludes this research abstract.

II. Related Work

Forensic analysis is used throughout the financial industry to help detect cardholder fraud and money laundering [3]. In the financial industry, fraud detection involves analyzing transaction logs of consumer activity. Transaction log records may include data fields for account number, merchant code, purchase amount, time of day, date of card issue, type of card, type of purchase, and client name, among others [3] [9]. Ghosh and Reilly [9] discuss fraud detection through the use of neural networks. Likewise, Kou et al. [4] present a literature survey of several fraud detection techniques, including outlier detection, neural networks, state transition analysis, and rule-based approaches. Each of the techniques discussed by Kou et al. seem to assume the transaction logs being analyzed contain a standard set of data fields and transaction event types that give an accurate, complete accounting of consumer activity. In other domains, such as the healthcare domain, technical standards for such transaction logs are not as widely used [5] [6] as in the financial industry [10].

In addition, Abraham and de Vel present research on investigative profiling through forensic analysis of log files. The authors discuss a technique for profiling user behavior based on association rules applied on log file contents. However, the authors did not perform immediate analysis on raw log file data. Instead, an investigator had to first process and convert raw log file data intended for analysis into a functional format using subsets and aggregations of the original log file data [11]. In their work, Abraham and de Vel analyze standard UNIX system logs based on prior knowledge that the UNIX logging mechanism *already* records data fields (such as user, duration, origin of access, date of access, etc.) and user events that are meaningful for forensic analysis of user activity.

Chuvakin and Peterson seem to support my view of activity logging for forensic-ability [12]. Chuvakin and Peterson state that using debugging logs for investigations is "…an exercise in frustration because they might not contain key details needed for incident response and forensics." The researchers further suggest five types of events that should be logged:

- Authentication, authorization, and access events
- System, data, and application changes
- Availability issues, such as startups, errors, and backups
- Resource issues, such as connectivity issues and exhausted resources
- Threats, such as invalid inputs

In addition, Chuvakin and Peterson suggest including the following context as part of each log entry:

- Username
- Affected system component
- Success or failure status
- System, application, or component context
- Source
- Timestamp and time zone
- Reason
- Action performed
- Priority (the event's importance)

While Chuvakin and Peterson address parts of my research objective, their work seems to be based upon personal experience and opinion and does not have empirical justification.

III. Research Direction

Section A discusses motivation for my approach. Section B presents my preliminary research questions.

A. Motivation

The outcome of forensic analysis of any log file depends on the content of the log files, themselves. One may only draw conclusions based on the evidence presented by the log. If a user activity log file does not contain detailed data about user activity, then forensic analysis of user behavior will be unproductive. Activity log files are centered on two primary components: a series of user events that trigger log entries, and

a set of data fields that capture context (such as time, username, etc.) of each user event. In activity log files containing inadequate data, anomalous user activity may go undetected. Anomalous user activity includes, for example, a user gaining access to restricted system resources, or a user who maintains multiple active sessions from two different locations. Forensic-ability metrics would quantify the ability to perform forensics intended to discover anomalous user activity.

A *low* forensic-ability score would indicate an inadequate logging mechanism for promoting user nonrepudiation. A low score would suggest that not enough data is being recorded in the log file. For example, a log file may not contain a data field that captures username or a precise timestamp of the user event that occurred. Or, a log file may only record user events that modify system resource state without logging any resource creations, views, or deletions. A low forensic-ability score suggests that anomalous user activity may not be discovered or detected due to temporal gaps or missing data in a trace of user behavior in the software system.

A *high* forensic-ability score would indicate that a log contains detailed data necessary to perform meaningful forensic analysis for user nonrepudiation. A high score would suggest that detailed data is being recorded in the log file to represent an accurate, thorough trace of user behavior in the software system. For example, a log file may record all data creations, views, modifications, and deletions, along with security events such as addition/revocation of user privileges and session timeouts. A high forensic-ability score suggests a better possibility of detecting anomalous user activity due to the thorough, detailed trace of user activity within the software system.

B. Research Questions

I approach the notion of forensic-ability metrics for nonrepudiation with the following questions:

- **RQ1**: What attributes should be recorded in each log entry to facilitate user nonrepudiation?
- **RQ2**: What user events should be logged to provide a meaningful trail of user activity in the software system for nonrepudiation?
- **RQ3**: Can a log file that facilitates strong user nonrepudiation be distinguished from a weak log file for user nonrepudiation?

IV. Methodology and Evaluation

Section A details my preliminary methodology and evaluation plans. Section B presents my current progress.

A. Methodology and Evaluation

For this research, I will use a grounded theory method [13] [14] for first observing the current state of logging mechanisms for user nonrepudiation, then forming my forensic-ability metric. I will observe open-source software and technical documentation in two domains that manage sensitive, protected information: education and healthcare. I observe only technical documentation from the financial industry in this study, since access to actual financial software systems is restricted. In addition, I will observe current documentation of logging standards, regulations, and policies for user activity logging.

My preliminary methodology is summarized in seven steps:

Collect the set of data fields included in activity log entries and documentation observed. Data fields include event context, such as the identification of the user and a timestamp of when the user performed the event. From the activity log files and documentation, I extract two sets of information. First, I extract a set of all data fields that are included in the log files and technical documentation. From this set of data fields, I annotate each data field with the number of log files in which the data field was found.

Collect the set of user events recorded in activity logs and documentation observed. User events include actions such as viewing data or logging in/out of the software system. I extract a set of all user events that trigger a log entry and annotate each event with the number of log files in which the event was found. The goal of this analysis is to determine which data fields and user events are more commonly included in activity log files for nonrepudiation.

Use the set of data fields from Step 1 and the set of user events from Step 2 to construct an "ideal" log file structure. I will construct an "ideal" activity log file structure based on user studies. For the user studies, sample log files will be presented to participants. These log files will contain anomalous user activity, and each log will contain differing combinations of data fields and user events. By differing the data fields and user events included in the logs, I intend to measure the ability of participants to effectively identify the anomalous user activity. My goal is to produce a benchmark activity log structure that maximizes the participants' effectiveness of identifying the anomalous user activity.

Collect a set of log files *not intended* to track user activity. (For example, I collect system logs, performance logs, debug logs, server logs, etc.). These log files may include server logs, performance logs, debug logs, or other types of log files. The purpose of my forensic-ability metrics is to indicate whether a log contains detailed data necessary to perform meaningful analysis of user activity for nonrepudiation. Therefore, I must also consider log files that are not solely intended to capture user activity. Having non-activity logs helps me observe the differences and similarities with logs that *are intended* to capture user activity. By observing, documenting, and quantifying these differences and similarities, I will have a set of data to help model the forensic-ability of the log files.

Quantify the differences between the non-activity logs in Step 4 and the ideal activity log structure from Step 3. Using data from user studies performed in the previous steps, I will quantify differences between pure activity logs and non-activity logs.

Propose a set of forensic-ability metrics that captures the quantified observations from Step 5. Using data observed and collected in user studies, I will aggregate the information to help model the degree to which a given audit log is effective for forensic analysis for nonrepudiation.

Validate the performance of the forensic-ability metric. I intend to demonstrate the practicality and meaningfulness of the forensic-ability metric. Practicality includes demonstrating causal relationship validity, economic productivity, usability, and predictability of the metrics [15]. Additional evaluation of the forensic-ability metrics involves verifying the performance of the metrics against not only the "ideal" activity log benchmark, but also the activity logs and non-activity logs previously observed. Furthermore, I will conduct an additional user study to observe and compare the effectiveness of the forensic-ability metrics with the performance of study participants. The participants will attempt to manually identify anomalous user behavior in sample log files that facilitate varying degrees of nonrepudiation. To obtain a more robust evaluation of the forensic-ability metric, the user study will include additional log files (both activity logs and non-activity logs) that have not been previously observed in this study. This user study will help determine whether the performance of the metrics accurately reflects the ability of human participants to perform forensic analysis on the log files for detecting anomalous user activity.

B. Current Status

I have previously collected sets of data fields and user events from the healthcare, education, and financial domains. Currently, I have begun documenting observations from healthcare, education, and financial logging mechanisms and technical documentation. I have also collected a set of standards, policies, and regulations for the healthcare, education, and financial industries. The next step involves designing a user study to help identify key data fields and user events necessary for identifying anomalous user activity in log activity files.

V. Conclusion

This research intends to improve the quality and adequacy of user activity log files for promoting user nonrepudiation. I will propose a suggested minimal set of log attributes (data fields and user events) for all user activity logging mechanisms. In addition, I will propose and validate a set of forensic-ability metrics that indicates the degree to which a given log file captures the data necessary to provide a meaningful trace of user activity for forensic analysis. I hope to provide software developers with a means to help measure the adequacy and usefulness of activity log file content for forensic analysis. Ultimately, the forensic-ability metrics would serve as a standard check for software developers to proactively strengthen user activity logging mechanisms to counter repudiation threats.

Acknowledgment

This work is supported by the USA National Security Agency (NSA) Science of Security Lablet. Any opinions expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSA. I thank the Realsearch group and my advisor, Dr. Laurie Williams, for continued guidance and valuable feedback.

References

[1] S. Hernan, S. Lambert, T. Ostwald, and A. Shostack. (2006, October 30). *Uncover Security Design Flaws Using the STRIDE Approach*. Available: http://msdn.microsoft.com/en-us/magazine/cc163519.aspx

[2] Computer Forensics. *United States Computer Emergency Readiness Team, part of the United States Department of Homeland Security*. Available: http://www.us-cert.gov/reading_room/forensics.pdf

[3] R. J. Bolton and D. J. Hand, "Statistical fraud detection: A review," *Statistical Science,* pp. 235-249, 2002.

[4] Y. Kou, C. T. Lu, S. Sirwongwattana, and Y. P. Huang, "Survey of fraud detection techniques," in *Networking, sensing and control, 2004 IEEE international conference on*, 2004, pp. 749-754.

[5] K. Kent and M. Souppaya, "Guide to Computer Security Log Management," National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 2006.

[6] A. Chuvakin and G. Peterson, "Logging in the age of web services," *IEEE Security and Privacy,* vol. 7, pp. 82-85, 2009.

[7] J. King, B. Smith, and L. Williams, "Modifying Without a Trace: General Audit Guidelines are Inadequate for Electronic Health Record Audit Mechanisms," presented at the ACM SIGHIT International Health Informatics Symposium, Miami, Florida, USA, 2012.

[8] "Health Insurance Portability and Accountability Act," ed: United States Department of Health & Human Services, 2007.

[9] S. Ghosh and D. L. Reilly, "Credit card fraud detection with a neural-network," in *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on*, 1994, pp. 621-630.

[10] "Payment Card Industry Data Security Standard," in *Requiremens and Security Assessment Procedure, v2.0*, ed: Payment Card Industry Security Standards Council, 2010.

[11] T. Abraham and O. de Vel, "Investigative profiling with computer forensic log data and association rules," in *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, 2002, pp. 11-18.

[12] A. Chuvakin and G. Peterson, "How to do application logging right," *Security & Privacy, IEEE,* vol. 8, pp. 82-85, 2010.

[13] B. G. Glaser and A. L. Strauss, *The discovery of grounded theory: Strategies for qualitative research*: Aldine de Gruyter, 1967.

[14] P. Y. Martin and B. A. Turner, "Grounded theory and organizational research," *The Journal of Applied Behavioral Science,* vol. 22, pp. 141-157, 1986.

[15] A. Meneely, B. Smith, and L. Williams, "Validating Software Metrics: A Spectrum of Philosophies," *ACM Transactions on Software Engineering and Methodology (TOSEM),* 2011.