Log Your CRUD: Design Principles for Software Logging Mechanisms

Jason King and Laurie Williams
Department of Computer Science
North Carolina State University
890 Oval Drive, Campus Box 8206
Raleigh, NC, 27695-8206
{jtking, laurie_williams}@ncsu.edu

ABSTRACT
According to a 2011 survey in healthcare, the most commonly reported breaches of protected health information involved employees snooping into medical records of friends and relatives. Logging mechanisms can provide a means for forensic analysis of user activity in software systems by proving that a user performed certain actions in the system. However, logging mechanisms often inconsistently capture user interactions with sensitive data, creating gaps in traces of user activity. Explicit design principles and systematic testing of logging mechanisms within the software development lifecycle may help strengthen the overall security of software. *The objective of this research is to observe the current state of logging mechanisms by performing an exploratory case study in which we systematically evaluate logging mechanisms by supplementing the expected results of existing functional black-box test cases to include log output.* We perform an exploratory case study of four open-source electronic health record (EHR) logging mechanisms: OpenEMR, OSCAR, Tolven eCHR, and WorldVistA. We supplement the expected results of 30 United States government-sanctioned test cases to include log output to track access of sensitive data. We then execute the test cases on each EHR system. Six of the 30 (20%) test cases failed on all four EHR systems because user interactions with sensitive data are not logged. We find that viewing protected data is often not logged by default, allowing unauthorized views of data to go undetected. Based on our results, we propose a set of principles that developers should consider when developing logging mechanisms to ensure the ability to capture adequate traces of user activity.

Categories and Subject Descriptors
K.6.5 **[Management of Computing and Information Systems]**: Security and Protection---*unauthorized access*; K.4.1 **[Computers and Society]:** Public Policy Issues**---***abuse and crime involving computers, privacy, regulation, use/abuse of power*; D.2.5 **[Software Engineering]:** Testing and Debugging---*testing tools*.

General Terms
Security, Measurement, Experimentation.

HotSoS '14, April 08 - 09 2014, Raleigh, NC, USA

http://dx.doi.org/10.1145/2600176.2600183

Keywords
Logging mechanism, audit, accountability, nonrepudiation, black-box testing, science, case study, electronic health record software, healthcare.

1. INTRODUCTION
According to a 2011 Veriphyr Survey of Patient Privacy Breaches [1], the top three most commonly reported breaches of protected health information (PHI) in healthcare involved snooping into medical records of employees (35%), snooping into medical records of friends and relatives (27%), and loss or theft of physical records (25%). In addition, 52% of survey participants indicated that their organization did not have adequate tools for monitoring inappropriate access to PHI. Logging mechanisms should capture all create, read, update, and delete (CRUD) accesses of PHI so that meaningful auditing of log entries can both proactively identify unauthorized PHI access on a continuous basis, as well as reactively recreate traces of user activity after a security or privacy breach occurs.

In software security, logging mechanisms provide a means of forensic analysis to help answer who, what, when, where, and how a security breach occurred. Logging mechanisms also help mitigate repudiation threats, threats associated with users who deny performing some action within the software system without other parties having any way to prove otherwise [11]. Adequate logging mechanisms help strengthen nonrepudiation, the ability of the software system to mitigate repudiation threats, and strengthen user accountability.

Many organizations maintain guidelines, standards, and specifications for logging mechanisms. In 2013, we created a catalog of logging mechanism guidelines for electronic health record (EHR) systems [14]. Based on the catalog, a developer would have to consider at least 13 out of 16 different source documents to identify 100% of the cataloged logging mechanism guidelines. We also found that general guidelines, such as those guidelines in the 16 source documents, are not adequate for ensuring that transactions with sensitive data are consistently captured by the logging mechanism [13]. Developers should guide design and development efforts based on specific guidelines, such as "log viewing of demographics data", "log viewing of allergy data", and "log viewing of prescription data" versus a general guideline of "log viewing of data". With guidelines that are distributed throughout the literature and not specific enough, developers may fail to consider important requirements regarding the implementation of logging mechanisms for identifying unauthorized access and recreating traces of user activity.

The objective of this research is to observe the current state of logging mechanisms by performing an exploratory case study in which we systematically evaluate logging mechanisms by supplementing the expected results of existing functional black-box test cases to include log output. We base our study on our logging guidelines catalog [14] and existing 2014 Edition Approved Test Procedures for Health Information Technology in the United States [2]. In addition, we define a systematic process for taking existing black-box test cases, processing the test case descriptions to identify user actions that are performed, and defining specific expected log output based on the identified user actions. We then perform a case study on four open-source EHR systems: OpenEMR, OSCAR, Tolven eCHR, and WorldVistA. From this exploratory case study, we observe the current state of the four studied logging mechanisms to synthesize a set of principles for logging mechanism design, implementation, and testing.

We define the following research questions:

RQ1: What observations can we make to understand why the four studied EHR logging mechanisms do not capture some specific user actions?

RQ2: What observations can we make about the general security of the four studied EHR logging mechanisms?

RQ3: What principles of logging mechanism design, implementation, and testing may be proposed based on observations of the four studied EHR systems?

In addition, this work contributes the following to the body of knowledge of software logging mechanisms:

- A systematic process for evaluating logging mechanisms based on existing black-box test suites
- An evaluation of four popular open-source EHR system logging mechanisms
- A set of principles to guide the design, implementation, and testing of logging mechanisms

The remainder of this paper is organized as follows. Section 2 presents background of Meaningful Use for healthcare software. Section 3 presents related work. Section 4 discusses the science involved in this study. Section 5 discusses our methodology. Section 6 presents the results of our evaluation. Section 7 provides a discussion of the results. Section 8 presents our proposed principles for logging mechanisms based on our results. Section 9 discusses limitations. Section 10 presents avenues for future work. Finally, Section 11 provides concluding remarks.

2. BACKGROUND

The United States Centers for Medicare and Medicaid Services (CMS) manage an EHR Incentive Program to provide financial incentives to healthcare providers who demonstrate "meaningful use" of EHR systems [9]. CMS provides a set of criteria that must be demonstrated by providers to prove "meaningful use" of EHR systems and receive incentive payment. Meaningful Use (MU) consists of three stages: Stage 1 (for year 2011), Stage 2 (for year 2014), and Stage 3 (for year 2016).

Each MU criteria consists of a description of the core objectives and requirements that must be demonstrated. In addition, the National Institute for Standards and Technology (NIST) provides a set of test procedures for each core objective and requirement. The United States Office of the National Coordinator (ONC) maintains ONC-Authorized Certification and Testing Bodies (ONC-ATCBs) that are responsible for certifying that EHR systems satisfy the requirements for demonstrating MU. The test procedures outline steps that an ONC-ATCB must perform in an EHR system, along with the expected outcomes that must be demonstrated after performing these steps. These NIST test procedures are black-box test cases used to certify that the EHR system satisfies the MU objectives.

In this paper, we use the 2014 MU test procedures as an EHR black-box test suite, since the functionality being tested applies to any United States EHR system that strives to be certified.

3. RELATED WORK

In previous work [13], we performed (1) a general auditable event evaluation; and (2) a specific auditable event evaluation of three open-source EHR systems: OpenEMR 3.2, OpenMRS 1.6.1, and Tolven eCHR RC1. For example, "view data" is a general auditable event that does not specify which data is being viewed. Specific auditable events detail exactly which data is being acted upon, such as "view allergy data", "view medication data", and "view demographics data". In the general auditable event evaluation, test cases passed as long as viewing of any data was captured by the logging mechanism, since "view data" was nondescript. However, the specific auditable event evaluation provided a much finer-grained, more meaningful evaluation of logging mechanisms since "view allergy data", "view medication data", and "view demographics data" were treated as three separate test cases. In the study, the three open-source EHR systems logged an average of 12.6% of general auditable events. The three EHR systems logged only 7.4% of specific auditable events. Overall, with such a lack of logged events, general auditable events provided by guidelines for logging mechanisms were deemed inadequate to ensure logging mechanisms capture useful traces of user activity.

The specific auditable events evaluation [13] was an extension to Smith and Williams [18] systematic approach to developing security test suites from functional requirements specifications. Smith and Williams identified security test patterns for security concerns such as attacks to input validation, uploading malicious files, and attacking authentication mechanisms. Key phrases in each software requirement were used to identify which types of security test patterns should apply to the given requirement. By applying the relevant security test pattern templates, Smith and Williams systematically generated a suite of test cases for evaluating the security of the software system. For our study, we systematically supplement existing functional black-box test cases with expected log output based on the user activity that occurs in the test procedure.

The Common Event Expression (CEE) project from the MITRE Corporation [3] attempts to standardize the representation and framework for logging mechanisms in software systems to facilitate exchange of log files among systems. CEE aims to simplify logging procedures, reduce costs associated with managing logging mechanisms, and improve auditing of log entries. However, funding and development for the CEE project has been discontinued. While a framework such as CEE would be beneficial in standardizing logging mechanism implementation and output, software developers are ultimately still responsible for identifying the transactions and user activity that must trigger log entries.

4. THE SCIENCE

To promote the science of security, we perform an exploratory case study [16] of four open-source EHR systems to understand the current state of logging mechanisms and to propose design principles to help strengthen future development of software

logging mechanisms. Exploratory case studies help researchers discover and observe what is happening in the current environment. Exploratory case studies also allow researchers to seek new insights into the subject matter, and then generate ideas, theories, and hypotheses for new systematic scientific research. By observing the current state of logging mechanisms in our case study, we further expand the body of knowledge of how logging mechanisms currently promote forensic analysis and security of software, including strengths and weaknesses of current implementations. We also discuss how our observations incorporate established security principles, such as security by default, least privilege, and separation of privilege [17]. From the newly gained knowledge, we form principles, theories, and hypotheses of how logging mechanisms may be improved through future scientific research. In addition, our results empirically substantiate guidelines and suggestions for logging mechanisms previously proposed by National Institute of Standards and Technology SP800-92 [12], United States Department of Defense Standard 5200.28 [7], and other documents used to compile our previous catalog [14]. The previously published guidelines and standards do not seem to be empirically derived or substantiated.

We present a systematic process for adding expected log output to existing black-box test cases. By presenting a repeatable, systematic process for our logging mechanism evaluation, we provide a means for software developers to evaluate their own logging mechanisms during the development process. We also enable other researchers to replicate our research methodology and compare results to expand the body of knowledge of logging mechanisms for forensic analysis and software security.

5. METHODOLOGY

In this section, we present our methodology for generating our black-box test plan for logging mechanisms. We also discuss our methodology for evaluating each EHR system.

5.1 Supplementing Existing Black-box Test Cases

We begin by selecting an existing black-box test suite for EHR systems. NIST provides black-box test procedures used for certifying that EHR systems satisfy core objectives and requirements that demonstrate meaningful use (see Section 2). The Approved 2014 Edition Test Procedures from NIST are the most recent version of the test procedures. Since the test procedures generally apply to any EHR system, we use the test procedures as an existing black-box test suite for our study.

The Approved 2014 Edition Test Procedures contain 48 certification criteria for functionality, such as managing demographics, immunizations, and family health history. We select a subset of these criteria for our study.

We begin our selection of criteria by first excluding the following:

- criteria for functionality that requires 3rd party tools/software (other than the EHR system)
- criteria that are optional for ambulatory certification
- criteria for inpatient certification only

From the remaining criteria, we include:

- the two criteria specifically related to logging/audit
- eight criteria randomly selected from the remaining set

We now have the following set of 10 criteria:

- demographics
- medication list
- medication allergy list
- electronic notes
- smoking status
- family health history
- authentication, access, control, and authorization
- auditable events and tamper-resistance
- audit report(s)
- immunization information

Multiple test cases exist for each criterion. After extracting all test cases from each of the 10 criteria, we obtain a total set of 34 individual black-box test cases.

Next, we manually parse each test procedure to identify all actions performed by the tester. For example, "The Tester shall enter the provided demographic test data." In this step of the test procedure, the tester enters demographic data into the EHR system. Each time the test procedure indicates that the tester should interact with the EHR system in some way, we expect a log entry to be recorded for the specific action that occurred. Therefore, in the above example, we create an expected test result that a log entry should have been recorded that indicates the tester entered (or created) demographic data. We repeat this same process for each of the 34 test cases to identify all actions performed by the tester and generate expected log output to supplement the existing black-box test case expected results. A complete example appears in Table 1. After supplementing each of the 34 test cases with expected log output, we observe that 4 of the 34 test cases contain no tester interaction with the EHR system and result in no expected logging outcomes. We omit these 4 test cases from our evaluation. A summary of the selected criteria and resulting black-box test cases appears in Table 2.

With our set of 30 test cases supplemented with expected events that should appear in log output, we must also identify what data should be captured as part of each log entry. Based on our logging guidelines catalog [14] and the ASTM International E2147-01 Standard Specification for Audit and Disclosure Logs for Use in Health Information Systems [8], we require that the following data be provided as part of each log entry for a test case to pass:

- Date and time of event
- Patient identification
- User identification
- Type of action (additions, deletions, changes, queries, print, copy)
- Identification of the patient data that is accessed

5.2 Electronic Health Record Systems Studied

For this study, we evaluate four popular open-source EHR systems:

- **OpenEMR v4.1.2**[1]. A 2011 ONC-ATCB Certified Ambulatory EHR system used by an estimated 15,000 physicians[2]. We include OpenEMR based on its active development community, its Certified EHR status, and previous experience and collaboration with developers of the system.
- **OSCAR v12.1**[3]. An EHR system used by an estimated 2,000 clinical providers, mostly in Canada. We include

[1] http://www.open-emr.org/

[2] http://www.openhealthnews.com/hotnews/openemr-continues-grow-popularity-and-use

[3] http://oscarcanada.org/

Table 1. A complete example showing how an existing black-box test case is supplemented with expected log output (underlined). We manually add the log outcome in the expected results field based on the action performed by the tester in the test description.

Test Identification	Test Description	Expected Results	Actual Results
DTR170.314(a)(3) – 1: Electronically Record Patient Demographics – Required Test Procedure	TE170.314(a)(3) – 1.01: Tester shall select the test data provided in TD170.314(a)(3) – 1 TE170.314(a)(3) – 1.02: Using the Vendor-identified EHR function(s) and three test patients, the Tester shall enter the provided demographic test data selected in TE170.314.a.3 – 1.01	TE170.314(a)(3) – 1.03: Using the Inspection Test Guide, the Tester shall verify that the patient demographic data entered in TE170.314(a)(3) – 1.02 are entered correctly and without omission, and in conformance with the standards for race, ethnicity and preferred language LOG: The tester shall verify that the act of entering demographic data is recorded by the logging mechanism.	

OSCAR based on its active development community and its widespread usage *outside* the United States. Since OSCAR is not widely used in the United States, the EHR system has not pursued Certified EHR status.

- **Tolven eCHR v2.1.3**[4]. A 2011 ONC-ATCB Certified Ambulatory EHR system used in North America, Europe, and Asia. We include Tolven eCHR based on its active development community, its Certified EHR status, and previous experience and collaboration with the organization.
- **WorldVistA v2.0**[5]. A 2011 ONC-ATCB Certified Ambulatory EHR system version of VistA, a mature EHR system developed by the United States Department of Veterans Affairs and deployed in Veterans Affairs Medical Centers across the United States. WorldVistA is a version of VistA intended for use outside of the Veterans Affairs Medical Centers. We include WorldVistA based on its Certified EHR status and the lengthy development history and widespread deployment of VistA-based EHR systems internationally.

5.3 Black-box Test Plan Execution

We first install each EHR system using the default installation instructions provided by each vendor. We do not change any default configuration settings. After installing each EHR system, we generate sample patient data before we begin executing test cases. OSCAR, Tolven eCHR, and WorldVistA provide utilities to generate random sample patient data. We manually created a sample patient using the OpenEMR application.

Next, we execute our 30 test cases that were supplemented with expected log output. For each test case, we perform the steps listed in the test case description. After performing the steps, we open the logging mechanism interface. The logging mechanism interfaces used for our evaluation are accessed as follows:

- In OpenEMR, we view the log entries as an administrative user by opening the "Log" tab under the Administration menu. This logging mechanism interface was used to achieve 2011 MU certification.
- In OSCAR, we view the log entries as an administrative user by opening the "Security Log Report" feature under the Administrative Security menu.
- In Tolven eCHR, we view the log entries as an administrative user by opening the "Performance Log" feature under the Admin tab. This logging mechanism interface was used to achieve 2011 MU certification.
- In WorldVistA, we view the log entries by connecting to the Caché database terminal, executing a function to compile an Audit Trail for Meaningful Use (command: D ^VWMUAUD), then displaying the compiled log file contents. This logging mechanism interface was used to achieve 2011 MU certification.

After executing the steps of the test case description and opening the logging mechanism interface, we check to see if the expected log output was presented in the logging interface.

The following criteria are used when recording the results of the test case execution:

- If any of the expected log output is not logged, or if any of the required data fields (date/time, user identification, patient identification, type of action that occurred, and identification of the data accessed) were not captured as part of each log entry, then we mark the test case as *failed*.
- If we cannot locate the described functionality (the EHR systems may not have the new 2014 MU Stage 2 criteria implemented yet), then we mark the test case as *not applicable* (NA).
- If all expected log output are contained in the log with all the required data fields, we mark the test case as *passed*.

We contacted representatives and users of each EHR system for assistance when we could not determine how to use any functionality of the EHR system.

6. RESULTS

Table 3 summarizes the test execution results. We shared our results with representatives and users of each EHR system for feedback, as well as to double-check the correctness of our results.

In OpenEMR, three test cases were not applicable (NA) because they involved testing functionality that we could not locate (suggesting the functionality has not been implemented yet). In OSCAR, nine test cases were not applicable to the available functionality. In Tolven eCHR, twelve test cases were not applicable. In WorldVistA, seven test cases were not

[4] http://www.tolven.org/echr.html

[5] http://www.worldvista.org/

Table 2. Summary of selected test procedure criteria and the resulting black-box test cases extracted from each criterion			
Criterion #	**Criterion Name**	**Number of Test Cases**	**Number of Test Cases with Log Outcomes**
§170.314(a)(3)	Demographics	3	3
§170.314(a)(6)	Medication List	3	3
§170.314(a)(7)	Medication Allergy List	3	3
§170.314(a)(9)	Electronic Notes	4	4
§170.314(a)(11)	Smoking Status	4	3
§170.314(a)(13)	Family Health History	3	3
§170.314(d)(1)	Authentication , Access, Control, and Authorization	2	2
§170.314(d)(2)	Auditable Events and Tamper Resistance	7	4
§170.314(d)(3)	Audit Report(s)	2	2
§170.314(f)(1)	Immunization Information	3	3
Total		**34**	**30**

applicable. Test cases for disabling the status of audit logs and disabling encryption status were not applicable to any of the four EHR systems.

OpenEMR passes 17 out of 27 (62.96%) of applicable test cases. OSCAR passes 8 out of 21 (38.1%) of applicable test cases. Tolven eCHR passes 4 out of 19 (21.1%) of applicable test cases. WorldVistA passed 0 out of 23 (0%) of the applicable test cases based on our evaluation criteria.

All four EHR systems fail on 6 (20%) of the 30 total test cases. At least three EHR systems fail on 10 (33.3%) of the 30 total test cases. At least two EHR systems fail on 17 (56.7%) of the 30 total test cases. No single test case is passed by all four EHR systems.

7. DISCUSSION

We first discuss important distinctions between MU criteria and the evaluation criteria used in our current study. Next, we outline common observations for failing test cases in each EHR system to discuss RQ1. Finally, we discuss observations related to general software security concerns to answer RQ2.

7.1 Meaningful Use Criteria versus Our Evaluation Criteria

Though we use MU criteria as the basis for our black-box test suite, we do not claim that the EHR systems fail to meet MU certification requirements based on our evaluation results. OpenEMR, Tolven eCHR, and WorldVistA are all 2011 ONC-ATCB certified EHR systems, meaning OpenEMR, Tolven eCHR, and WorldVistA have satisfied the criteria required to be certified EHR systems. We chose to use the newest 2014 MU criteria test procedures as the basis for our black-box test suite since the test procedures are general enough to apply to any EHR system. Test cases that are marked "NA" in Table 3 often means the functionality accessed by the test procedure is new for the 2014 MU criteria and has not yet been implemented in the EHR systems.

Our evaluation criteria do not follow the same test procedures ONC-ATCBs use to certify logging mechanisms in EHR systems. Instead, our evaluation criteria are considerably more specific and systematic compared to the 2011 NIST test procedures with which the EHR systems were certified. Our evaluation criteria is based on our logging guidelines catalog [14] and the ASTM International E2147-01 Standard Specification for Audit and Disclosure Logs for Use in Health Information Systems[8]. Furthermore, the NIST test procedures for evaluating logging mechanisms require testers to randomly select an action and ensure it was logged. A summary of the NIST test procedure for evaluating logging mechanisms appears in Table 4. Our evaluation methodology, however, is much more systematic. *Every* action performed by the tester in *each* test procedure should be logged, and the tester should also verify that each of the actions performed in the test case were actually recorded in the logging mechanism. While the NIST test procedures perform a random "spot-check" of only a few actions to ensure the actions were logged, our evaluation methodology systematically verifies that every action that accesses PHI in each of the test procedures has been correctly logged.

7.2 Test Failure Observations

RQ1: What observations can we make to understand why the four studied EHR logging mechanisms do not capture some specific user actions?

OpenEMR. The OpenEMR logging mechanism records the exact SQL query executed against the database for each log entry. In a default OpenEMR installation, seven test cases that involve viewing of data fail. By accessing administrative configuration settings and enabling logging of all SELECT queries in OpenEMR, the logging mechanism then passes these seven failing test cases. However, our evaluation was based on a default installation of the EHR system. Microsoft describes the *security by default* principle [15] as a means to minimize the harm that occurs when attackers target security vulnerabilities by promoting security in the software's default state. In EHR systems that manage PHI, access and viewing of PHI is a critical activity that should be logged to ensure patient privacy and accountability of users who may not be authorized to access the PHI.

Tolven eCHR. In Tolven eCHR, updating allergies, medications, and immunizations causes a log entry to be recorded. However, these log entries for updating allergy, medication, and immunization data incorrectly indicates that data were *added* (or created) in the system, not updated. Based on our evaluation criteria, we expected the log entry to accurately indicate the action that was performed by the human user. As a result, four test cases fail in Tolven eCHR because of incorrect descriptions of the action that occurred. Behind the scenes, Tolven eCHR does not replace or update existing database records. Instead, the EHR adds new records to the database to maintain both the old data as well as the newer updated data. However, since we perform a black-box evaluation of the EHR logging mechanisms, we assume no knowledge of the code implementation of the system during the execution of our test cases. Since the human user updates data in the system, we expect the logging mechanism to reflect the action that the human user completed. The human user thinks he

Table 3. Summary of test execution results				
Test ID	**OpenEMR**	**OSCAR**	**Tolven eCHR**	**WorldVistA**
Electronically Record Patient Demographics	PASS	PASS	FAIL	NA
Electronically Change Patient Demographics	PASS	PASS	FAIL[b]	NA
Electronically Access Patient Demographics	FAIL[a]	PASS	FAIL	FAIL
Electronically Record Patient Active Medication List	PASS	PASS	PASS	FAIL
Electronically Change Patient Active Medication List	PASS	PASS	FAIL[b]	FAIL
Electronically Access Patient Active Medication List and Medication History	FAIL[a]	FAIL	FAIL	FAIL
Electronically Record Patient Active Medication Allergy List	PASS	PASS	PASS	FAIL
Electronically Change Patient Active Medication Allergy List	PASS	PASS	FAIL	NA
Electronically Access Patient Active Medication Allergy List and Medication Allergy History	FAIL[a]	FAIL	FAIL	FAIL
Electronically Record Electronic Notes	PASS	FAIL	NA	FAIL
Electronically Change Electronic Notes	PASS	FAIL	NA	FAIL
Electronically Access Electronic Notes	FAIL[a]	FAIL	NA	FAIL
Electronically Search Electronic Notes	NA	NA	NA	FAIL
Electronically Record Patient Smoking Status	PASS	NA	PASS	FAIL
Electronically Change Patient Smoking Status	PASS	NA	FAIL[b]	FAIL
Electronically Access Patient Smoking Status	FAIL[a]	NA	FAIL	FAIL
Electronically Record a Patient's Family Health History	PASS	NA	NA	FAIL
Electronically Change a Patient's Family Health History	PASS	NA	NA	FAIL
Electronically Access a Patient's Family Health History	FAIL[a]	FAIL	NA	FAIL
Authenticate Unique User	PASS	PASS	FAIL	FAIL
Establish Permitted User Access	PASS	FAIL	NA	NA
Permit Audit Log, Audit Log Status and Encryption Status Disabling -- DELETE	PASS	NA	NA	NA
Permit Audit Log, Audit Log Status and Encryption Status Disabling -- DISABLE	NA	NA	NA	NA
Permit Audit Log, Audit Log Status and Encryption Status Disabling -- DISABLE ENCRYPTION	NA	NA	NA	NA
Record Actions	FAIL	FAIL	FAIL	FAIL
Generate an Audit Report	FAIL	FAIL	FAIL	FAIL
Sort Audit Log	FAIL	FAIL	FAIL	FAIL
Electronically Record Immunization Information	PASS	FAIL	PASS	FAIL
Electronically Change Immunization Information	PASS	FAIL	FAIL[b]	FAIL
Electronically Access Immunization Information	FAIL[a]	FAIL	FAIL	FAIL
PASS	17	8	4	0
FAIL	10	13	15	23
TOTAL	27	21	19	23
PASS Percentage	62.96%	38.1%	21.1%	0%

[a]OpenEMR does not log SELECT queries (data views) by default. This feature can be enabled, however.

[b]Tolven eCHR provides a log entry when the action occurs. However, the log entry states that the action was "add" instead of "update" or "change"

or she has updated existing data, but the logging mechanism claims data were added. This mismatch of event descriptions could mislead an auditor to believe the user added or created data in Tolven eCHR.

Tolven eCHR also stores additional query parameters in its database that gives more details about what actions the user performed in the system. However, one must have authorized access to the Tolven eCHR database, itself, to manually combine tables and manually construct an audit trail with more detailed information about user activity. Since the performance log interface was used for demonstrating 2011MU objectives, and since we performed a black-box evaluation, we base our results on the information displayed in the performance log interface.

WorldVistA. In WorldVistA, only data accesses seemed to generate log entries. All log entries generated by executing our test cases indicated "ACCESSED" actions. No log entries indicated modifications or creations of data. Furthermore, log entries did not provide a clear, human-readable description of the event that occurred. As a result, no test cases passed for WorldVistA based on our evaluation criteria

Table 4. NIST Test Procedure for evaluating logging mechanisms [10]			
Test Identification	**Test Description**	**Expected Results**	**Actual Results**
DTR170.314(d)(2) – 3: Record Actions	TE170.314(d)(2) – 3.01: Using the EHR function(s) and test data identified by the Vendor, the Tester shall make an **addition** to electronic health information in the patient record provided by the Vendor TE170.314(d)(2) – 3.02: Using the EHR function(s) and test data identified by the Vendor, the Tester shall make a **deletion** to electronic health information in the patient record provided by the Vendor • If the EHR technology does not allow any users to make deletions related to electronic health information, refer to the IN170.314(2) – 3.04 TE170.314(d)(2) – 3.03: Using the EHR function(s) and test data identified by the Vendor, the Tester shall make a **change** to electronic health information in the patient record provided by the Vendor TE170.314(d)(2) – 3.04: Using the EHR function(s) and test data identified by the Vendor, the Tester shall make a **query** about electronic health information in the patient record provided by the Vendor TE170.314(d)(2) – 3.05: Using the EHR function(s) and test data identified by the Vendor, the Tester shall **print** electronic health information in the patient record provided by the Vendor TE170.314(d)(2) – 3.06: Using the EHR function(s) and test data identified by the Vendor, the Tester shall **copy** electronic health information in the patient record provided by the Vendor	IN170.314(d)(2)–3.01: For each action performed in TE170.314(d)(2)–3.01 through TE170.314(d)(2)–3.06, the Tester shall verify that the action was recorded in the audit log IN170.314(d)(2)–3.02: Tester shall verify that the following data elements were recorded by the audit log for each action performed (additions, changes, queries, print, copy) in TE170.314(d)(2)–3.01 and TE170.314(d)(2)–3.03—3.06 • Date and time (utilizing a system clock that has been synchronized following the (RFC 1305) Network Time Protocol, or (RFC 5905) Network Time Protocol Version 4) • Patient identification • User identification • The action(s) taken, specifying inquiry and any changes made (with pointer to original data state)	

7.3 General Security Observations

RQ2: What observations can we make about the general security of the four studied EHR logging mechanisms?

In OpenEMR, OSCAR, and Tolven eCHR, administrative users may (simultaneously) be physicians or other user roles. Saltzer and Schroeder [17] provide a core set of design principles for computer security. These principles include *separation of privilege* and *least privilege*. Separation of privilege suggests that a system should not grant permission based on a single condition. In OpenEMR, for example, administrative users (who may also be doctors) have direct access to a running copy of phpMyAdmin. The phpMyAdmin interface allows administrative users to tamper (modify or delete) existing log entries, or create fake log entries. Suppose Doctor Alice has been granted typical access privileges for physicians, as well as access privileges for administrators. If Doctor Alice decides to access the medical records for her neighbor and change the neighbor's blood type, Doctor Alice could then open phpMyAdmin and delete all log entries that show she inappropriately accessed PHI and changed the blood type. Separation of privilege would ensure that no single user could have direct access to such vital administrative resources as phpMyAdmin. Instead, more than one condition should be met (such as having a secondary or tertiary administrative user approve Doctor Alice's access request to phpMyAdmin). Similarly, least privilege suggests that users should be granted only the minimum access necessary to perform an operation. For example, in a typical daily workflow, Doctor Alice would primarily only need access to clinical functions associated with patient care. Doctor Alice likely would not need access to administrative configuration settings (such as disabling logging) or access to phpMyAdmin. Both OSCAR and Tolven eCHR also allow physicians to simultaneously have administrative access privileges in a default installation.

Common Weakness Enumeration (CWE) provides a dictionary of common software vulnerabilities. Included in this dictionary are three common weaknesses associated with logging mechanisms:

- CWE532: Information Exposure through Log Files – "Information written to log files can be of a sensitive nature and give valuable guidance to an attacker or expose sensitive user information"[4].
- CWE778: Insufficient Logging – "When security-critical events are not logged properly, such as a failed login attempt, this can make malicious behavior more difficult to detect and may hinder forensic analysis after an attack succeeds"[5].
- CWE779: Logging Excessive Data – "The software logs too much information, making log files hard to process and possibly hindering recovery efforts or forensic analysis after an attack"[6].

The OpenEMR logging mechanism exemplifies CWE532, the risk of information exposure through log files. Since OpenEMR captures each individual SQL query executed against the database, queries that update PHI contain the exact data being updated. For example, adding an allergy causes the OpenEMR

logging mechanism to capture and display the following SQL query (abbreviated for space):

```
INSERT INTO lists (date, pid, type, title)
VALUES (NOW(), '1', 'allergy', 'penicillin')
```

From this SQL query, we can derive that Patient 1 has an allergy to penicillin. Anyone who views the OpenEMR log entries can glean sensitive information about patients from the SQL queries captured. An administrative user who is also a physician could search the patient directory to discover the name, address, and other data about Patient 1. With CWE532, the sensitivity of the log content should be considered when granting and revoking read access to the log entries. Access should only be granted to auditors who need to perform job-related responsibilities. Furthermore, auditor access to the logs should also be logged.

In terms of CWE778 (Insufficient Logging), based on our evaluation results, all four EHR systems do not adequately log critical events such as viewing PHI. Malicious behavior, including inappropriately accessing PHI, may be more difficult to detect as a result. Similarly, forensic analysis of log files from the EHR system may be difficult with gaps in traces of user activity since not all PHI accesses are logged.

OpenEMR also exemplifies CWE779 (Logging Excessive Data). OpenEMR does not log SELECT queries (viewing data) by default. Enabling logging of SELECT queries, however, results in a distinct increase in the number of log entries generated for each action performed by the user. For example, clicking a patient name to view the patient's summarized health record generates nearly 80 individual log entries within a 2 second interval. Since OpenEMR captures *each* individual SQL query executed, log entries are generated for selecting global variable values, allergy list values, medication list values, insurance data values, etc. each time a patient record summary is accessed. While recording the exact SQL query gives a detailed understanding of how user activity affects sensitive data stored in the database, an excessive amount of log entries are generated for a succinct act, such as "view summarized patient record." When you consider a deployment of OpenEMR at a large healthcare organization with hundreds of physicians viewing PHI every few minutes, the log file may quickly become overwhelming and hinder meaningful forensic analysis by auditors.

Tolven eCHR and WorldVistA do not provide log entries when users authenticate into the EHR system. Security events such as authentication are vital when reconstructing traces of user activity. Authentication provides a record of when a user gained (or was denied) access to the software. Likewise, since log entries may contain sensitive information (CWE532), logs should be accessible only to authorized auditors whose job involves detecting inappropriate access to PHI and identifying security or privacy breaches. All four EHR systems fail to generate a log entry when log entries, themselves, are viewed or sorted. Overall, current EHR logging mechanisms seem to focus on logging transactions with PHI, while security events such as authentication and changes to access control privileges are not logged as often. Logging security events is important for detection of malicious attacks on authentication mechanisms, or detection of information exposure through other security attacks.

8. PRINCIPLES FOR LOGGING MECHANISMS

RQ3: What principles of logging mechanism design, implementation, and testing may be proposed based on observations of the four studied EHR systems?

Based on our observations and evaluation results, we propose a set of principles for the design, implementation, and testing of logging mechanisms.

Log by Default. *Logging mechanisms should capture all required events out-of-the-box by default, based on the default installation configuration of the software.* Four test cases involving viewing of data failed in all four EHR systems. While OpenEMR does log data views after enabling the logging of SELECT queries, such logging should be enabled by default. Developers should not assume administrators will configure logging mechanism settings after deployment. By default, the software should be configured to log all data transactions and all data fields for each log entry upon installation without requiring any additional configuration or intervention. Additionally, any changes to the default logging mechanism configuration settings should generate a log entry, as well as an alert to immediately signal that logging settings have been altered.

Specify Logging Requirements. *Developers should systematically develop and document a set of actions that should be logged by their specific piece of software. The set of actions to be logged should be customized for the specific piece of software since functionality varies among different software projects and vendors.* In this study, OpenEMR did not contain functionality for the execution of 3 test cases. OSCAR did not contain functionality needed for the execution of 9 test cases. Tolven eCHR did not contain functionality needed to execute 11 test cases. Likewise, WorldVistA did not contain functionality needed for 7 test cases. General guidelines, such as those used in our evaluation, cannot apply to *every* individual piece of software. While the four EHR systems did not contain the functionality to perform certain test cases, the four EHR systems likely also have functionality not covered by general guidelines or evaluation criteria. Logging mechanism requirements should be directly tied to the functionality implemented in each given software system. Additionally, the EHR systems often inconsistently log events. For example, OSCAR logs *accessing* demographics data for patients. However, OSCAR does not log *accessing* allergy or medication lists. Having a specific set of actions that should be logged may help avoid inconsistencies where viewing of demographics data is logged, but viewing of allergy or medication lists is not logged.

Capture Adequate Context. *For each log entry, logging mechanisms should capture adequate context to help understand who, what, when, where, why, and how an event occurred.* At a minimum, logging mechanisms should capture the timestamp, username, exact transaction that occurred, and specify which piece of data was being acted upon. In Tolven eCHR and WorldVistA, for example, many log entries did not contain a description of which data had been accessed. In OpenEMR, the SQL queries captured for each log entry give a complete understanding of which data was being acted upon. To facilitate meaningful forensic analysis, auditors should quickly be able to understand who, what, when, where, why, and how an action occurred. Without appropriate context in log entries, user accountability diminishes because the logging mechanism does

not provide enough details to clearly prove a user performed certain actions.

Support Human-readable Reporting. *Audit reports should display log entries in a clearly understandable, human-readable format to facilitate forensic analysis.* WorldVistA often includes a sequence of characters to indicate which data was being acted upon by the user. For example, "XQALERT" is displayed in certain log entries to provide more details of the user access that occurred. However, it is unclear what "XQALERT" means or represents without internal knowledge of the system. Audit reports should clearly state the contents of each log entry in a human-readable format that minimizes the amount of internal knowledge of the system required to understand the log entry. A third-party auditor with no knowledge of the internal system should still be able to clearly understand each log entry to facilitate forensic analysis.

Succinctly Represent User Behavior. *Audit reports should display a succinct, meaningful description for the action performed by the user in the system.* If a user views allergy information for a patient, the audit report should display a single entry to indicate the user viewed allergy information. The audit report should not display, as in OpenEMR for example, twenty different log entries that represent twenty different SQL queries executed. Similarly, as discussed in Section 7, changes to data in Tolven eCHR are all displayed in the audit report as *additions* of data. If we update a medication dosage, the audit report states that we added medication data. Even though medication data is internally added (not updated) in the database, the logical user action involved changing data. The audit report should accurately reflect the logical behavior of users in the system, not internal system behavior.

Enforce Immutability. *Logging mechanisms should prevent user write access to create, modify, delete, or otherwise tamper with log entries.* OpenEMR allows any administrative user to directly access log entries through an installation of phpMyAdmin. The administrative user can create fake log entries, delete existing log entries, or alter existing log entries. Therefore, the OpenEMR logs are not trustworthy. The OpenEMR log entries cannot be used to prove that users performed certain actions in the system since the log entries are not guaranteed to be truthful or accurate.

Perform Systematic Black-box Testing. *Logging mechanisms should be systematically black-box tested based on specific logging requirements derived for the individual software to ensure all user interactions with data and security events are correctly logged.* In this study, we present our process for systematically black-box testing logging mechanisms in EHR systems. Logging mechanisms provide a black-box representation of what happens in the system. No internal understanding of the underlying software system should be required. Logging mechanisms also span the breadth of functionality in software. Any time data is accessed by any piece of functionality, the logging mechanism should be triggered to generate a log entry. Therefore, logging mechanisms should be thoroughly tested to ensure each possible data access point adequately triggers log entries. As discussed in Section 7, existing test procedures for logging mechanisms provided by NIST are only random "spot-checks" of a small subset of all possible data access points that could trigger log entries. Systematic black-box testing helps ensure that the logging mechanism is adequately incorporated throughout the entire software system to ensure accountability of user actions.

9. LIMITATIONS

First, our principles are based solely on observations from four open-source EHR systems. These four EHR systems may not accurately reflect the state of logging mechanisms in commercial EHR software. However, we chose four open-source EHR systems that are widely used in multiple countries to help generalize our results.

Next, our evaluation criteria are based on a subset of all NIST test procedures for health information technology. While we manually included both test procedures directly related to logging and auditing, the randomly selected subset of test procedures may not adequately capture a representative posture of the security of the logging mechanisms, as a whole. Overall, 30 out of 120 (25%) total individual test cases were deemed "Not Applicable" since we could not locate the functionality necessary to execute the test cases. A more comprehensive, expanded set of evaluation criteria may reveal additional security considerations.

10. FUTURE WORK

We plan to develop and evaluate a systematic process for defining specific logging requirements to help developers gain a comprehensive understanding of the specific requirements for their individual logging mechanism, instead of relying on general guidelines. Similarly, we plan to develop and evaluate additional systematic black-box testing processes for generating a complete black-box test suite for logging mechanisms based on specific logging requirements.

We define the following research questions to guide future research:

> **$RQ1_{future}$:** What criteria should be considered when constructing an evaluation framework for evaluating the ability of logging mechanisms to hold users accountable and promote meaningful forensic analysis?
>
> **$RQ2_{future}$:** How well do logging mechanisms based on *specific* logging requirements promote user accountability and meaningful forensic analysis versus logging mechanisms based on *general* logging requirements?
>
> **$RQ3_{future}$:** How adequate/representative is an evaluation of logging mechanisms using a black-box test suite generated from specific logging requirements?

We define the following hypotheses for future experiments based on our current case study results:

> **H_1**: Specific loggable events can be accurately extracted from natural language requirements documents.
>
> **H_2:** Systematically processing a natural language requirements document to identify specific loggable events generates a larger set of loggable events, compared to systematically processing existing black-box test cases to identify loggable events.
>
> **H_3:** Developers who follow a set of specific loggable events implement logging mechanisms that capture a larger set of user events, compared to developers who follow a set of general loggable events.

Our research plan involves developing and validating a set of security metrics for measuring the degree to which a logging mechanism promotes meaningful forensic analysis and user accountability. We also intend to automate our current black-box evaluation process using behavior-driven development tools and techniques.

11. CONCLUSION

Overall, 61 out of 90 (67.8%) applicable executed test cases failed, indicating several current weaknesses in the four existing EHR logging mechanisms. Four out of six test cases that failed across all four EHR systems involve viewing of sensitive data, indicating that users may view protected information without being caught. Even though OpenEMR passed 17 of the 27 (62.96%) applicable test cases, the percentage of tests passed does not fully capture or represent the posture of the OpenEMR logging mechanism. The OpenEMR logging mechanism exemplifies several security weaknesses, including failing to make the log entries immutable and logging a potentially excessive amount of data.

Logging mechanisms should not be implemented without a solid software design in place. By proposing seven principles for the design, implementation, and testing of logging mechanisms, we intend to guide developers in building logging mechanisms that improve forensic analysis of user activity in software should a security or privacy breach occur.

12. ACKNOWLEDGMENTS

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. We thank the Realsearch group for providing helpful feedback on this research. We also thank Sam Bowen, Jens Weber, Tom Jones, Nancy Anthracite, and David Whitten for helpful feedback on this research and for excellent help with using the EHR systems.

13. REFERENCES

[1] *2011 Survey of Patient Privacy Breaches*. August 2011. Veriphyr Incorporated, http://www.veriphyr.com/landing/HIPAA_violation_survey/.

[2] *2014 Edition Test Method*. December 14, 2012. The Office of the National Coordinator for Health Information Technology, http://www.healthit.gov/policy-researchers-implementers/2014-edition-final-test-method.

[3] *Common Event Expression: A Unified Event Language for Interoperability*. 2013. MITRE Corporation, http://cee.mitre.org.

[4] *CWE-532: Information Exposure Through Log Files*. 2013. The MITRE Corporation, http://cwe.mitre.org/data/definitions/532.html.

[5] *CWE-778: Insufficient Logging*. 2013. The MITRE Corporation, http://cwe.mitre.org/data/definitions/778.html.

[6] *CWE-779: Logging of Excessive Data*. 2013. The MITRE Corporation, http://cwe.mitre.org/data/definitions/779.html.

[7] *Department of Defense Standard: Trusted Computer System Evaluation Criteria*. 1985. United States Department of Defense, http://csrc.nist.gov/publications/history/dod85.pdf.

[8] *E2147-01 Standard Specification for Audit and Disclosure Logs for Use in Health Information Systems*. 2013. American Society for Testing and Materials, http://www.astm.org/Standards/E2147.htm.

[9] *Electronic Health Records and Meaningful Use*. 2011. The Office of the National Coordinator for Health Information Technology, http://healthit.hhs.gov/portal/server.pt/community/healthit_hhs_gov__meaningful_use_announcement/2996.

[10] *Test Procedure for 170.314(d)(2) Auditable events and tamper-resistance*. July 11, 2013. The Office of the National Coordinator for Health Information Technology, http://www.healthit.gov/sites/default/files/170.314d2auditableevents_2014_tp_approvedv1.2.pdf.

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

[12] Kent, K. and Souppaya, M. 2006. *Guide to Computer Security Log Management*. National Institute of Standards and Technology, Gaithersburg, Maryland, USA. Available: http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf.

[13] King, J., Smith, B. and Williams, L. 2012. Modifying Without a Trace: General Audit Guidelines are Inadequate for Electronic Health Record Audit Mechanisms. In *Proceedings of the ACM SIGHIT International Health Informatics Symposium* (Miami, Florida, USA).

[14] King, J. and Williams, L. 2013. Cataloging and Comparing Logging Mechanism Specifications for Electronic Health Record Systems. *Presented as part of the 2013 USENIX Workshop on Health Information Technologies* (Washington, DC, USA).

[15] Lipner, S. and Howard, M. 2005. *The Trustworthy Computing Security Development Lifecycle*. Microsoft Developer Network, http://msdn.microsoft.com/en-us/library/ms995349.aspx.

[16] Runeson, P. and Host, M. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131-164.

[17] Saltzer, J. and Schroeder, M. 1975. The Protection of Information in Computer Systems. *Proceedings of the IEEE*, 63(9):1278-1308.

[18] Smith, B. and Williams, L. #2011-5. *Systematizing Security Test Planning Using Functional Requirements Phrases*. North Carolina State University.