Grant Final Report

Grant ID: R21HS018218

Use of Affordable Open Source Systems by Rural/Small-Practice Health Professionals

Inclusive Project Dates: 09/30/09 – 09/29/12

Principal Investigator:
Laurie Williams[1]

Team Members:
Mladen Vouk[1], Jacqueline Halladay[2]

Performing Organization:
[1]North Carolina State University, Department of Computer Science
[2]University of North Carolina at Chapel Hill, Department of Family Medicine

Federal Project Officer:
Erin Grace

Submitted to:
The Agency for Healthcare Research and Quality (AHRQ)
U.S. Department of Health and Human Services
540 Gaither Road
Rockville, MD 20850
www.ahrq.gov

Abstract

Purpose: The purpose of the research was to investigate whether the electronic health record (EHR) application needs of rural and small-practice ambulatory health care providers could be met with open source EHR applications. Such applications need to be trusted – among other things trust means functional, affordable, reliable, available, secure, privacy-preserving, standards/regulations-based, and able to interoperate and be integrated with other health care systems.

Scope: We investigated five open source EHR systems: OpenEMR, OpenMRS, Tolven, WorldVista, and PatientOS, and one proprietary system.

Methods: We developed software engineering and evaluation practices and an interview protocol for physician needs assessment. We developed an EHR testbed and analyzed six applications.

Results: While the applications are for the most part functionally complete, our analyses very quickly discovered that one of the biggest trust issues is security. Our observations indicate that existing open source EHR applications may contain significant and serious security vulnerabilities. A vulnerability is a hole or a weakness in the application (i.e., a fault) that allows an attacker to cause harm to the stakeholders of an application. We also concluded that CCHIT and NIST EHR test procedures used for certification are not specified in ways that would detect the code-level vulnerabilities we detected.. Publications resulting from this study represent some of the first, in some cases the first, reports regarding vulnerabilities in some widely used open source EHR software, and should be the cause for considerable concern in the EHR community.

Key Words: electronic health records; EHR; open source software; security; privacy

Final Report

Purpose

The purpose of the research was to investigate whether the electronic health record (EHR) application needs of rural and small-practice ambulatory health care providers could be met with open source EHR applications. Such applications need to be trusted; the applications need to meet a range of trust-related criteria. These criteria include, but are not limited to, functionality; interoperability; maintainability (especially remote maintainability in the case of rural/small doctor practices); reliability and availability; fault-tolerance and recovery; security; privacy-preservation; how much they are standards/regulations-based; policy and regulation compliance; and the ability to be integrated with other health care systems.

Our proposal had five specific objectives:

1. To advance the understanding of engineering practices for *developing new or enhancing existing* trustworthy open source or proprietary EHR applications based upon evaluation experiences;

2. To advance understanding of a process for *evaluating* the trustworthiness, functionality, interoperability (use of standards such that information can be shared with other providers), performance, compliance, affordability, etc., of *existing* open source EHR applications;

3. To gather and analyze the needs of rural/small practice ambulatory health care providers in the realm of electronic health records;

4. To develop and evaluate a prototype system on which promising open source EHR applications can be assessed, i.e., deployed, run, and administered/maintained remotely and for which hardware usage is securely shared/optimized to improve affordability; and

5. To provide an assessment of the capabilities, strengths, and limitations of existing open source EHR applications towards meeting the needs of rural/small practice doctors.

Scope

The majority of our work was on five open source electronic health record (EHR) applications: OpenEMR[1], OpenMRS[2], Tolven[3], WorldVistA[4], and PatientOS[5] and one

[1] http://www.oemr.org/
[2] http://openmrs.org/
[3] http://www.tolven.org/

proprietary system (the identity of the application cannot be released). The five open source EHR applications were chosen because, at the time of the study, they were among the most popular open source EHR applications[6]. Additionally, the applications were diverse to increase the generalizability of the results. The applications were written in four different programming languages (PHP, Java, C#, and Mumps). They were both web applications and client/server applications. They varied in size from 120,000 to 1.6 million lines of code. For one project we used iTrust[7]. iTrust is an educational EHR testbed application developed at North Carolina State University (NCSU).

These applications were the subject of all of our work. We installed these applications in a testbed that allowed a lot of flexibility – specifically a virtualized computing environment based at NCSU [3]. We very quickly found that one of the biggest trust "holes" appears to be security. We evaluated these applications for security vulnerabilities, and other factors related to their trustworthiness. In the process, we also developed new software engineering techniques relevant to EHR type software, and evaluated the techniques on these applications.

Methods

Our approach included:

1. Development of software engineering techniques for developing trustworthy EHR applications. We evaluated these practices on open source EHR applications.

2. Development of techniques for evaluating the trustworthiness of EHR applications. We used these techniques on open source and one proprietary EHR applications.

3. Development of a needs interview protocol. We used the protocol to interview rural/small practice doctors and their information technology support staff.

4. Creation of a reusable testbed. We installed open source EHR applications in a virtual computing environment such that the applications could be used by other researchers and tried by practitioners.

5. Analyses. We analyzed our data and reported on and published (disseminated) the results. In our initial proposal, we had intended to assess multiple aspects of EHR systems: reliability, security, privacy-preserving, standards/regulations-based, and interoperability. However, because we found such fundamental flaws that would compromise the security of the application, the privacy of the data, and regulations (i.e. HIPAA), we focused on the latter three security impact areas.

---

[4] http://www.worldvista.org/
[5] http://www.patientos.org/
[6] See demographics of the application, including frequency of use at: http://www.realsearchgroup.org/healthcare/doku.php
[7] http://agile.csc.ncsu.edu/iTrust/wiki/doku.php

Results

In this section we summarize the results of each of our five objectives discussed in Section 2 (Purpose). In this section, we refer to products and publications produced from this grant as listed in Section 6 (Publications and Products). We refer to products as PRx and to publications as PUx.

Objective 1. To advance understanding of engineering practices for *developing new or enhancing existing* trustworthy open source or proprietary EHR applications based upon evaluation experiences.

As part of this task, we created a portal, which can aid software developers in the healthcare IT, domain to understand the complexities of software development [PR2]. The portal provides links to Meaningful Use criteria, HIPAA, and secure software development practices. The portal also provides links to relevant research papers in software engineering, healthcare, healthcare IT systems, security, and privacy.

We also developed processes for developing secure open source EHR applications. As we discuss these processes, we use the term *vulnerability*. A vulnerability is "a hole or a weakness in the application (i.e., a fault), which can be a design flaw or an implementation bug, that allows an attacker to cause harm to the stakeholders of an application"[8], i.e., exploit the vulnerability at run-time to his/her benefit. Vulnerabilities are generally classified as either security-related *design flaws* or *implementation bugs*. A design flaw is a larger, more widespread problem in the architecture or system design. Repairing a security-related design flaw in an implemented system generally involves a significant amount of rework. A fix to a security-related implementation bug generally involves adding, deleting, or editing only a small number of lines of code. About half of discovered vulnerabilities are design flaws and half are implementation bugs [2].

1. Software developers must regularly use validation and verification (V&V) techniques to efficiently find and remove software vulnerabilities as early in the software development lifecycle as possible. Security vulnerabilities discovered later in the development cycle are more expensive to fix, and often more difficult to fully eliminate, than those discovered early. Therefore, software developers should strive to discover vulnerabilities as early as possible. Unfortunately, the large size of code bases and lack of developer expertise can make discovering software vulnerabilities difficult. A number of vulnerability discovery techniques are available, each with their own strengths.

   As part of this research we compared the vulnerabilities detected by different techniques and compared their efficiencies. We conducted three case studies using EHR systems to compare four vulnerability discovery techniques: exploratory manual penetration testing, systematic manual penetration testing, automated penetration testing, and

[8] https://www.owasp.org/index.php/Category:Vulnerability

automated static analysis. In our case studies, we found empirical evidence that no single technique discovered every type of vulnerability. We also discovered that the specific set of vulnerabilities identified by one tool may be largely orthogonal to that of other tools. Systematic manual penetration testing found the most design flaws related to architecting a secure system, while automated static analysis found the most implementation bugs, programming code-level problems that leave open holes for attackers. The most efficient discovery technique in terms of vulnerabilities discovered per hour was automated penetration testing. The results indicate that employing a single technique for vulnerability discovery is insufficient for finding all types of vulnerabilities. Each technique identified only a subset of the vulnerabilities, which, for the most part were different from each other. Our results suggest that in order to discover the greatest variety of vulnerability types, at least systematic manual penetration testing and automated static analysis should be performed. [PU2, PU3]

2. In regulated domains such as health care, failure to comply with regulation (e.g. United Stated Code or HIPAA) can lead to financial, civil and criminal penalties. Currently, failure to meet certification criteria can lead to compromised financial gains of an EHR vendor because customers are likely to choose to purchase certified applications. While EHR systems vary from organization to organization, HIPAA regulations and HITECH certification criteria apply across organizations. We use Behavior-Driven-Development (BDD) technology [4] to develop scenarios that can be automated and run to assess system behavior in comparison with the regulations and certification criteria. BDD is a software development practice that organizes development effort around the creation of scenarios that illustrate desired system behavior in terms of the vocabulary used by system stakeholders [5]. In our case, the desired system behavior is the compliance with Meaningful Use certification criteria. One of the most studied regulations in this area concerns HIPAA technical safeguards. No test procedures for these have been published, but there are an analogous set developed by NIST for testing the HITECH act meaningful use (MU) provisions. MU covers a wide range of EHR functionality requirements, and the NIST has developed test procedures for each of them. The language of HITECH sections 170.302(o)-(u) closely matches the language of HIPAA 164.302(a)-(g), to the point of verbatim language in some sections. Rather than attempt to claim that these regulations are directly comparable here, we choose to mention the correspondence, and to base our test suite on the test procedures associated directly with the HITECH regulations in CFR 170.302. We wrote system-specific test driver code to execute the seven scenarios on three EHR systems (iTrust, OpenEMR and Tolven). Our results demonstrated that a reusable set of test scenarios could be developed for regulations, specifically Electronic Code of Federal Regulations § 170.302 [PU7] [PY10] and used to automate compliance checking for multiple (three) applications

Objective 2. To advance understanding of a process for evaluating the trustworthiness, functionality, interoperability (use of standards such that information can be shared with other providers), performance, compliance, and affordability, etc., of existing open source EHR applications.

As part of this task, we have developed processes for evaluating the security of open source EHR applications.

1. For the benefit of protecting patient privacy, regulations and certification criteria related to EHR systems stipulate that organizations have a policy that addresses access control of protected health information. The goal of this aspect of the research is to guide development teams, regulators, and certification bodies by assessing the state of the practice in EHR access control for open source systems. In this work, we compiled 25 criteria relative to access control in EHR systems found in the HIPAA Security Rule[9], certification criteria provided by the Certification Commission for Healthcare Information Technology[10], National Institute for Standards and Technology (NIST) test procedures, best practices embodied in the NIST role-based access control standard[11], and other best practices found in the literature. We then examined the state of the practice in access control in open source by evaluating four open source EHR systems using these 25 evaluation criteria. Our research indicates that the systems that meet the MU criteria would also be in compliance with HIPAA access control standards. However, in 2011 when the analysis was completed, the CCHIT certification criteria did not address the NIST best practices related to access control. Additionally, our results indicate that open source EHR system designers are not implementing robust access control mechanisms for the adequate protection of patient data. [PU5]

2. Inadequate audit mechanisms may result in undetected misuse of data in software-intensive systems. EHR systems should log the creating, reading, updating, or deleting of privacy-critical protected health information. The objective of one research project was to assess EHR audit mechanisms to determine the current degree of auditing for non-repudiation and to assess whether general audit guidelines adequately address non-repudiation. We analyzed the audit mechanisms of two open source EHR systems, OpenEMR and Tolven, and one proprietary EHR system. We base our qualitative assessment on a set of 16 general auditable events and 58 black-box test cases for specific auditable events. We find that OpenEMR satisfies 62.5% of our general auditable events and passes 63.8% of our black-box test cases. Tolven eCHR and the proprietary EHR system each satisfy less than 19% of our general auditable events and pass less than 11% of our black-box test cases. The analysis of these open source systems indicates a possible shortfall in adequately logging data access of sensitive health data. [PU6]

3. We adapt the notion of a software design pattern as proposed by Gamma et al. to the domain of black box security testing. A software security test pattern is a description of a generalized test case that could be used to reveal a recurring vulnerability type, that is described such that the test case can be instantiated a million times over, without ever doing it the same way twice. In this research, we produce 11 software security test patterns. To demonstrate that we can use patterns produced using our methodology to develop a black box security test plan, we applied the 11 patterns using a public requirements specification (the 2011 CCHIT Ambulatory certification criteria[12]) for EHR systems to produce a test plan consisting of 117 tests. We then executed these 117 tests

[9] http://www.hhs.gov/ocr/privacy/hipaa/administrative/securityrule/securityrulepdf.pdf
[10] https://www.cchit.org/cchit-certified
[11] http://healthcare.nist.gov/docs/170.302.o_AccessControl_v1.1.pdf
[12] https://www.cchit.org/documents/18/158304/CCHIT+Certified+2011+Ambulatory+EHR+Criteria.pdf

on three open source EHR systems. We find that 65 out of 351 (18.5%) of our test executions were successfully able to launch a security attack in one of the three EHR systems. [PU9]

Objective 3. To gather and analyze the needs of rural/small practice ambulatory health care providers in the realm of electronic health records.

In 2010, we interviewed physicians and their information technology (IT) support staff from four practices. We have published our interview protocol [PU4]. Below are some themes of the physicians and their support staff.

- They were not happy about the need to transition to the use of an EHR system. They would prefer to keep their office running as it currently was.
- They personally had concerns about the security of EHR systems.
- Their patients had even more concerns about the privacy implications of EHR systems.
- The IT staff was cognizant of basic steps for making their office secure, such as requiring passwords for the office wireless systems. However, the IT support staff had little to no knowledge about application-level security.

Dr. Halladay and a supported PhD student attended the North Carolina Academy of Family Physicians Winter Meeting in December 2010. They disseminated our research results via a poster and spoke with the physicians who came to hear about the research results [1].

Objective 4. To develop and evaluate a prototype system on which promising open source EMR applications can be deployed, run, and administered/maintained remotely and for which hardware usage is securely shared/optimized to improve affordability.

As part of this task, we installed the following applications in our virtual computing environment test-bed [3]

- Astronaut WorldVistA Client Test
- Astronaut WorldVistA Server
- EHR System Suite
- iTrust v11.0
- iTrust v12.0
- OpenEMR v3.2.0
- OpenMRS Version: 1.6.1 Build 12909
- PatientOS Client Test

- Patient OS Server 0.99

- Tolven

The intention was to make the applications available so other researchers and medical practices could try out the applications. Unfortunately, while we can run in a protected test-bed, we were unable to keep these applications running for an open (public) access. Attackers were consistently able to find the applications and launch successful attacks against them. The same vulnerabilities we discovered and reported would enable the attackers to launch these types of attacks. The worrying part is that these same applications are currently being used with real patient data, configured with Internet access.

Objective 5. To provide an assessment of the capabilities, strengths, and limitations of existing open source EHR applications towards meeting the needs of rural/small practice doctors.

We have conducted a thorough security analysis of five open source electronic health record (EHR) applications (OpenEMR, OpenMRS, Tolven, WorldVista, and PatientOS) and one proprietary system (the identity of the application cannot be released). The detailed results of our analysis are publically available [PR1]. We reported the found vulnerabilities to the development organizations.

We can summarize our assessment of open source EHR applications by saying the overall security and privacy-preserving attributes are inadequate. In practice, this finding should basically make those versions of the applications untrustworthy. We also note that CCHIT and NIST EHR test procedures[13] used for certification are not specified in ways that would detect the code-level vulnerabilities we detected.

We cannot recommend the use of the open-source EHR applications we examined for use by small and rural medical practices. [PU1] [PU5] Due to the inability to analyze more than one proprietary application, we cannot make any statement about the security of proprietary EHR applications.

A December 25, 2012 Washington Post special "Zero Day" report[14] entitled, "Health-care sector vulnerable to hackers, researchers say" provides to the popular press some of our research results. Our work is cited as excerpted from the article:

> *But Laurie Williams, a computer scientist at North Carolina State University, said health care remains widely vulnerable.*
>
> *"There are basic, basic, Security 101 vulnerabilities we identified," said Williams, who was among a team of researchers that identified numerous security flaws in several electronic heath records systems two years ago. "I'm concerned that at some point the hackers are really going to begin exploiting them. And that's going to be a scary day."*

[13] NIST certification testing criteria http://healthcare.nist.gov/use_testing/finalized_requirements.html
[14] http://www.washingtonpost.com/investigations/health-care-sector-vulnerable-to-hackers-researchers-say/2012/12/25/72933598-3e50-11e2-ae43-cf491b837f7b_story.html

…

Two years ago, Williams, the North Carolina State researcher, and her colleagues found common flaws in four systems that would expose users' login information and enable outsiders to access patients' records.

The group's report urged rigorous security testing before electronic health record vendors could be certified for stimulus funding.

Among the systems that HHS has certified is OpenEMR, an open-source software developed by a nonprofit charitable group called OEMR. The software can be downloaded for free. Williams's group — along with several white-hat hackers — has found hundreds of vulnerabilities in the system.

References

1. A. Austin, B. Smith, L. Williams, and J. Halladay, "How Secure is Your Electronic Health Record System?," in *North Carolina Academy of Family Physicians Weekend (NCAFPW)*, Asheville, NC, December 2010.

2. G. McGraw, *Building Secure Software*. Boston, MA: Addison Wesley, 2002.

3. M. Vouk, A. Rindos, S. Averitt, J. Bass, M. Bugaev, A. Peeler, H. Schaffer, E. Sills, S. Stein, J. Thompson, and M. P. Valenzisi, "Using VCL Technology to Implement Distributed Reconfigurable Data Centers and Computational Services for Educational Institutions," *IBM Journal of Research and Development*,, vol. 53, no. 4, pp. 2:1-18, 2009.

4. M. Wynne and A. Hellesoy, *The Cucumber Book: Behavior-Driven Development for Testers and Developers*. Raleigh, NC: Pragmatic Press, 2012.

List of Publications and Products

Products

[PR1] We have conducted a thorough security analysis of five open source electronic health record (EHR) applications (OpenEMR, OpenMRS, Tolven, WorldVista, and PatientOS) and one proprietary system (the identity of the application cannot be released). The results of our analysis are publically available at this website: http://www.realsearchgroup.com/healthcare/doku.php.

[PR2] We have produced an online resource for software engineers that produce healthcare applications: http://realsearchgroup.org/healthit/.

Publications

[PU1] Austin A, Smith B, Williams L. Towards Improved Security Criteria for Certification of Electronic Health Record Systems Proceedings of *2nd Workshop on Software Engineering in Healthcare* (SEHC) at the International Conference on Software Engineering (ICSE) 2010, 3-4 May 2010; Cape Town, South Africa, electronic proceedings.

[PU2] Austin, A, Holmgreen C, Willliams L. A Comparison of the Efficiency and Effectiveness of

Vulnerability Discovery Techniques, Information and Software Technology, posted online 14 December 2012, to appear.

[PU3] Austin A, Williams L, One Technique is Not Enough: An Empirical Comparison of Vulnerability Discovery Techniques, *Proceedings of International Symposium on Empirical Software Engineering and Metrics (ESEM) 2011*, 22-23 September 2011: Calgary, Canada. p. 97-106.

[PU4] Halladay J, Williams L, Vouk M, On the Affordable Use, Administration, and Maintenance of Open Source Health Care IT Applications by Rural/Small-Practice Health Professionals: Interview Protocol, NCSU Computer Science Technical Report TR-2012-16.

[PU5] Helms E. Williams L. Evaluating Access Control of Open Source Health Record Systems. *Proceedings of 3rd Workshop on Software Engineering in Healthcare* (SEHC) at the International Conference on Software Engineering (ICSE) 2011, 22-23 May 2011; Honolulu, USA, electronic proceedings.

[PU6] King J, Smith B, and Williams L, Modifying Without a Trace: General Audit Guidelines are Inadequate for Electronic Health Record Audit Mechanisms. *Proceedings of the International Health Informatics Symposium (IHI 2012)*, 28-30 January 2012: Miami, FL, p. 305-314.

[PU7] Morrison P, Holmgreen C, Massey A, Williams L. Proposing Regulatory-Driven Automated Test Suites for Electronic Health Record Systems, Software Engineering in Healthcare (SEHC) workshop at the International Conference on Software Engineering (ICSE) 2013, San Franciscio, CA, to appear.

[PU8] Smith B, Austin, A, Brown M, King J, Lankford J, Meneely A, Williams L. Challenges for Protecting the Privacy of Health Information: Required Certification Can Leave Common Vulnerabilities Undetected. *Proceedings of Security and Privacy in Medical and Home-care Systems (SPIMACS 2010)* Workshop of ACM Computers and Communication Security 2010; 8 October 2010; Chicago, IL. p. 1-12.

[PU9] Smith B., Empirically Developing a Software Security Test Pattern Catalog Using a Grounded Theory Approach [dissertation]. Raleigh, NC: North Carolina State University; 2012.

[PU10] Morrison, P., Holmgreen, C., Massey, A., and Williams, L., Proposing Regulatory-Driven Automated Test Suites, 2013, Agile Software Development, Nashville, TN, to appear. (Best Research Paper Award)