Testing Electronic Health Records Applications with a Security Test Pattern Catalog Developed Using Empirical Data

Ben Smith and Laurie Williams
Department of Computer Science
North Carolina Sate University
Raleigh, NC, USA
[ben_smith, laurie_williams]@ncsu.edu

Abstract— **The United States is suffering from a shortage of software security experts. One expert claims that there are approximately 1,000 people in the country with the skills needed for cyber defense, and goes on to say that 20 to 30 times that many are needed. Another report indicates that today's graduates in software engineering are unprepared to enter the workforce because they lack a solid understanding of how to make their applications secure. Due to this shortage of security expertise, we need a vehicle with which we can capture and disseminate knowledge about how to assess whether software systems have adequate defenses against malicious users.**

We adapt the notion of a software design pattern as proposed by Gamma et al. to the domain of black box security testing. A design pattern is a description of a recurring problem and a well-defined description of the core solution to the problem that is described such that the pattern can be used many times but never in exactly the same way. A software *security test* pattern is a template of a test case that exposes vulnerabilities, typically by emulating what an attacker would do to exploit those vulnerabilities.

Capturing attacker behavior in a security test case allows the systematic, repeated assessment of a system's defenses against a particular attack. We codify a process for developing security test patterns by identifying the similarities between test cases that expose known vulnerabilities and abstracting common components to make the test strategy reusable. Additionally, others can use this process of developing patterns to capture and disseminate security testing knowledge and to contribute additional patterns. Just as design patterns disseminate design knowledge, expressing proven security testing techniques as patterns makes them more accessible to people who are not experts in security, and makes it easier to reuse successful testing strategies.

***The goal of this research is to codify a process for developing a software security test pattern catalog that provides a vehicle for capturing and disseminating knowledge about software security testing based upon grounded theory analysis of empirical data*. We analyzed the CWE/SANS Top 25 Most Dangerous Programming Errors[1] using a grounded theory approach to produce six initial test patterns. Future studies will allow us to evolve our pattern catalog and validate our process within the context of other data sources.**

We applied our initial six test patterns to the Certification Commission for Health Information Technology (CCHIT) Ambulatory Criteria to develop test cases from our patterns. Specifically, we employed 284 functional requirements from the CCHIT criteria to create a black box security test plan consisting of 137 security tests for four open source and one proprietary electronic health record (EHR) system: OpenEMR[2], ProprietaryMed[3], WorldVistA[4], Tolven[5], and PatientOS[6]. We then executed the 137 test cases on each of these five released EHR systems that are currently used to manage the records of over 59 million patients. This resulted in a total of 685 test executions. Overall, our test plan launched 253 (37%) successful attacks in the five EHR systems that consisted of both implementation-level defects, such as cross-site scripting, and design-level issues, such as the lack of encryption on the backup copy of system data. These vulnerabilities could be catastrophic with respect to the objective of protecting patients' medical records. We also alerted developers to the vulnerabilities we found by posting respective healthcare IT communities' bug report pages.

We further evaluated the test plan by comparing it to two techniques: automated penetration testing and automated static analysis, to identify the common vulnerabilities discovered by each technique. Additionally, our comparison to other techniques shows that two automated security assessment tools missed 80-90% of our discovered vulnerabilities. We have also developed a tool that uses natural language processing to automate the test case generation procedure using customizable patterns and keywords. Our automation of this technique shows promising results for using a tool to help non-experts in security to create and use black box security testing in a systematic fashion. The tool, pattern catalog, test plan and test results are available from our security test patterns wiki[7].

Keywords- security; testing; black box; patterns; health care

[1] http://cwe.mitre.org/top25

[2] http://oemr.org

[3] ProprietaryMed wishes to keep the identity of their product confidential.

[4] http://worldvista.org

[5] http://tolven.org

[6] http://patientos.org

[7] http://securitytestpatterns.org