

Defining a Set of Common Benchmarks for Web Application Security

Benjamin Livshits
Computer Science Department
Stanford University
Stanford, USA

`livshits@cs.stanford.edu`

A recent explosion in the number of security vulnerabilities being discovered every day motivated a great deal of interest in tools that attempt to address this problem. While buffer overruns have been plaguing C programs for years, application-level vulnerabilities such as SQL injections, cross-site scripting, and path traversal attacks have become increasingly common in the last year. Looking at a daily snapshot of Security-Focus vulnerabilities reveals that upwards of 60% of all exploits are due to application vulnerabilities such as the ones listed above.

While the number of commercially available and open-source tools that claim to address these issues is on the rise, there is generally no approach that would validate the claims made by the tool makers. We believe that there is a need for a set of universally accepted benchmarks that can be used as a validation test bed for security tools.

Putting together such a set of realistic benchmarks is challenging because many if not most applications suffering from application-level vulnerabilities are closed-source proprietary software deployed at banks, medical centers, etc. While some attempts have been made at constructing artificial benchmarks [1, 4], we believe that real-life programs are much better suited for testing security tools.

In the course of our research in application security [3] at Stanford, our group has developed a suite of benchmarks called STANFORD SECURIBENCH [2]. Thus far it consists of 8 real-life open-source Web-based applications written in Java and developed on top of J2EE. Most programs are medium-sized, with the larger ones consisting of almost 200,000 lines of code. We have identified about 30 vulnerabilities in these programs and there are probably more that can be found with appropriate static and dynamic tools.

We are making these benchmarks publicly available in hopes of fostering collaboration between researchers. These benchmarks can serve as test cases for researchers and industry practitioners working in this area.

References

- [1] Foundstone, Inc. Hackme books, test application for Web security. <http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/proddesc/hacmebooks.htm>.
- [2] Benjamin Livshits. Stanford SecuriBench. <http://suif.stanford.edu/~livshits/securibench/>, 2005.
- [3] V. Benjamin Livshits and Monica S. Lam. Finding security errors in Java programs with static analysis. In *Proceedings of the 2005 Usenix Security Conference*, 2005.
- [4] The Open Web Application Security Project. WebGoat Project. <http://www.owasp.org/software/webgoat.html>.