[in]security

by Gary McGraw

Application Security Testing Tools: Worth the Money?

Application security testing tools are being sold as a solution to the problem of insecure software. However, these solutions aren't all they're cracked up to be. They may help us diagnose, describe, and demonstrate security problems, but they do little to help us fix them.

Today's application security testing tools treat software applications as "black boxes," prone to misbehavior and in need of probing and prodding to prevent security disaster. Unfortunately, this approach is too simple.

Software testing requires planning. It should be based on software requirements and the architecture of the code under test. You can't "test quality in" by painstakingly finding and removing bugs once the code is finished.

The same goes for security: Running a handful of canned tests that simulate malicious hackers by sending malformed input streams to a program won't work. Real attackers don't simply "fuzz" a program with input to find problems. They take software apart, determine how it works, then make it misbehave by doing what users aren't supposed to do. Black box tests only scratch the surface of software, instead of digging into its guts to secure things from the inside.

> There's no set of prefabricated tests that will probe every possible application in a meaningful way.

BADNESS-OMETERS

While most architecture and coding vulnerabilities are beyond the reach of simple canned tests, these tools can tell you *something* about security—namely, that you're in very deep trouble. That is, if your software fails any of the canned tests, you have some serious security work to do. Even if the software passes all the tests with flying colors, you know nothing more than that it passed a handful of tests with flying colors.

Put in more basic terms, application security testing tools are "badness-ometers." They provide a reading in a range from "deep trouble" to "who knows?" but they don't provide a reading up into the "security" range at all.

BEYOND PORT 80

The other major weakness with application security testing tools is that they focus on input to an application over port 80. Understanding and testing a complex program by relying only on the protocol it uses to communicate provides a shallow analysis. Though many attacks do arrive via HTTP, this is only one route of entry. Input arrives to modern applications in many forms: Consider SSL, environmental variables, outside libraries, distributed components that communicate using other protocols, and so on. Beyond program input, software security must also consider architectural soundness, data security, access control, software environment, and any number of other aspects, all of which depend on the application itself.

In short, there's no set of prefabricated tests that will probe every possible application in a meaningful way. In fact, the only good use for application security tools is testing off-the-shelf commercial software, where simple dynamic checks set a reasonable low bar to hold vendors to. If the software fails to pass those tests, you can either reject it or attempt to manage the risks associated with the security flaws.

In the final analysis, application security testing tools provide only a modicum of value. Organizations that are just beginning to think through software security issues can use them as badness-ometers to help determine how much trouble they're in. Results can alert all the interested parties to the presence of a problem and spur mitigation efforts, but you won't get anything more than a rudimentary analysis with these tools. Your money will be better spent on building better software to begin with.

Gary McGraw is CTO of Cigital, a software quality management consulting provider. He is co-author of *Exploiting Software* (Addison-Wesley, 2004), *Building Secure Software* (Addison-Wesley, 2001), and *Java Security* (Wiley, 1996). Reach him at gem@cigital.com.