Web application security
White paper
January 2008

Rational. software

IBM

Web application security: automated scanning versus manual penetration testing.

Danny Allan, strategic research analyst,
IBM Software Group

Contents

Introduction

Research has shown that a vast number of Web sites are vulnerable to Web application attacks and that a great percentage of these attacks occur over the HTTP/S protocols, ports that are often exposed to the entire online community. With these facts in mind, it's essential for organizations to take serious measures to help secure their Web applications.

As Web applications become increasingly complex, tremendous amounts of sensitive data—including personal, medical and financial information—are exchanged and stored. Consumers expect and even demand that this information be kept secure. There are two primary methods for discovering Web application vulnerabilities: using manual penetration testing and code review or using automated scanning tools and static analysis. The purpose of this paper is to compare these two methods.

Evolving testing techniques

Manual security penetration testing is one of the oldest methods for discovering application vulnerabilities. Over time, as the frequency of attacks has grown and application complexity has increased, specialists known as penetration, or "pen," testers have emerged. Their sole purpose is to find and exploit Web application

Highlights

security problems. In the late 1990s, companies began developing automated Web application testing techniques. By that point, the Web had become more mature, and Web browsers were beginning to be able to handle the complexities of dynamic applications. The goal of these early automated testing tools was to automate the process of discovering a Web application and inject faults into it to help discover vulnerabilities.

Web application vulnerabilities typically fall into two categories: technical and logical.

Two primary categories of vulnerabilities

Generally, Web application vulnerabilities can be grouped into two categories: technical and logical. Technical vulnerabilities include cross-site scripting (XSS), injection flaws and buffer overflows. Logical vulnerabilities are much harder to explicitly categorize. These vulnerabilities manipulate the logic of the application to get it to do things it was never intended to do. For example, in early 2002, a hacker used a logical vulnerability to bypass the required personal information validation in a popular e-mail application, allowing the hacker to reset users' passwords by guessing the answer to a single security question.

Highlights

Automated testing tools can now traverse, analyze and test for a large percentage of technical vulnerabilities.

Technical vulnerabilities

There are more than 70 techniques that can be used to exploit XSS, one of the most common technical vulnerabilities. A typical registration form on the Web contains approximately 30 unique elements, each of which is potentially vulnerable to XSS, injection flaws, buffer overflows or improper error handling. Therefore, to test the form for XSS vulnerability, you would need more than 2,000 tests to check all 30 elements against the 70 XSS techniques. It's certainly no surprise that a great number of applications are vulnerable to this one exploit.

Given the number of tests needed to check such applications for technical vulnerabilities, automated tools that are able to traverse, analyze and test are perhaps more efficient than manual penetration testing. Automated scanning and testing tools may not currently be able to test 100 percent of technical vulnerabilities, but they can test for a large percentage of them. Early versions of automated tools had trouble dealing with certain issues, including:

- *Client-side-generated URLs.*
- *Required Java™ Script functions.*
- *Application logout.*
- *Transaction-based systems requiring specific user paths.*
- *Automated form submission.*
- *One-time passwords.*
- *"Infinite" Web sites with random URL-based session IDs.*

Highlights

As automated Web application security testing tools have matured, enterprises have experienced fewer incidents of false positives and false negatives.

As automated Web application security tools have matured, the majority of these issues have been addressed, and automated assessments have reduced incidents of uncertain determinations (false positives) and missed issues (false negatives). However, as Web applications continue to grow in size, manual testing is becoming more and more difficult. In many enterprise organizations, it will become impossible to dedicate the time, effort and money to assess the booming number of Web applications. The bottom line is that humans can only look at so many lines of code per day, and as your volume of applications increases, so too must your stable of testers—which can quickly become cost prohibitive.

Logical vulnerabilities are security weaknesses that can be exploited by circumventing the typical flow of an application.

Logical vulnerabilities

Logical vulnerabilities are security gaps that can be exploited by understanding how an application works and circumventing the typical business flow. While automated scanning tools and skilled pen testers can navigate a Web application, only the tester is able to understand the logic behind the application's workflow. This understanding enables the tester to subvert the business logic and expose a security vulnerability. For example, an application might direct the user from point A to point B to point C, where point B is a security validation check. A manual review of the application might show that it is possible to go directly from point A to point C, bypassing the security validation entirely.

Highlights

Although manual penetration testing and automated scanning can both be used to find critical security vulnerabilities in Web applications, each has inherent strengths and weaknesses.

Neither manual penetration testing nor automated scanning is an exhaustive method for identifying Web application vulnerabilities. Each method has its own inherent strengths and weaknesses, and both can be used to discover critical security vulnerabilities in Web applications. Automated tools were never intended to, and should never entirely replace, the manual penetration test. However, if used correctly, automated tools can be used to find a broad range of technical security vulnerabilities in Web applications, saving time and money. Sophisticated organizations will determine the correct mix of automated scanning versus manual penetration testing to provide the best Web application security coverage possible.

Delivering the software and services you need to help secure your Web applications

IBM is a marketplace leader in Web application vulnerability assessment software. Providing a range of security software products, IBM offers individual tools for any size organization.

Using IBM Rational Web application security testing products, developers can build security into every Web application—and work within familiar technology environments.

IBM Rational® Web application security testing products can help your organization build security into applications destined for online deployment. The Rational products allow users to work within familiar technology environments, offering virtually seamless integration with leading quality assurance tools and

Highlights

Rational testing software enables enterprises to continuously audit the security of Web applications and to mitigate business risk prior to deployment.

integrated development environments (IDEs). And the applications enable you to perform continuous security auditing, helping software delivery teams build security into Web applications from the ground up, and helping to mitigate business risk before you deploy your applications.

For more information

To learn more about IBM Rational Web security solutions, contact your IBM representative or IBM Business Partner, or visit:

ibm.com/software/rational/offerings/testing/webapplicationsecurity

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
01-08

RAW14011-USEN-00