
# Application Security Testing Tools

## Facts and Emerging Vendors

January 7th, 2017

# Contents

# 1. Introduction

The purpose of web application security testing is to find any security weaknesses or vulnerabilities within an application and its environment, to document the vulnerabilities, and to explain how to fix or remediate them. The business drivers behind the testing may be requirements of corporate policy, security requirements mandated by the corporate financial auditors or an internal audit department, compliance requirements for PCI or other industry standards, or compliance with regulatory standards such as Sarbanes-Oxley or HIPAA (Lepofsky). Application security is the key risk area for exploits, and exploits of applications can be devastating. A vulnerability is an error or weakness that a hacker can exploit. There are many types of vulnerabilities can be found in software system or websites, including interface errors, access controls check errors, design-level errors and object-sharing system errors. Vulnerabilities typically fall into two categories: bugs at the implementation level and flaws at the design level. MITRE has catalogued circa 800 different kinds of software weaknesses in their CWE project.

In this paper, we focus different types of application security testing techniques, tools and service offering from vendors, market overview and common application security challenges an organization can face.

# 2. Code review

Application security testing is important topic since it takes time and money to implement and does not require additional business functions or add glitz to an application. Since financial executives receive only risk analysis data as the result of testing, they sometimes put this type of testing in the backseat of production priorities.

Once the application code has been written in a secure fashion, it is of course time to test the code to verify its security health. One might think that after adhering to a framework for writing secure application code, testing it may be overkill. However, this could not be further from the truth; this is separation of duties in real life (Lepofsky).

Application testing includes:

**Reviewing lines of code:** Reviewing ensures that lines of code comply with the security plan and that their logic will produce the intended results.

**Real-time testing:** This type of testing assesses how applications actually respond and function.

**Constant testing and retesting:** It is necessary to test all the time, including during code writing, after end-user testing, just before introducing the code into production, and continuously thereafter. Testing is especially important after changes are made to the application environment including to both software and network technology.

## 3. Manual vs. Automated Code Testing

A source code review is an effective method of detecting security vulnerabilities as well as other logic flaws. Manual reviews, the tried-and-true method of code testing, especially applicable prior to the advent of automated-code testing tools, are time consuming and expensive. The reasons for this include the requirements of:

- **A team effort**: A team effort is necessary since programmers are required to review each other's work. The reasoning for this is that programmers may notice errors in another developer's code with much more clarity than their own errors.

- **Real-time testing**: Reviews need to be repeated at regular intervals to review fresh code or re-review code after recommended changes have been applied.
- **Expertise**: Those reviewing the code need to have extensive application-development experience and security expertise.

However, automated application-source-code analyzer tools can shorten the time and cost required to review and subsequently make the requisite corrections to source code, particularly for large applications. A number of different select tools can analyze source code or a compiled version of the code.

Automated tools are most cost effectively used in the application development environment since correcting security vulnerabilities at an early stage is less expensive than finding and correcting them late in the development cycle. However, automated tools can provide a false sense of security that everything is being addressed, when, in fact, they cannot identify every kind of web application vulnerability and can produce false positives and false negatives (Sebesta, 2016). (It should be noted that this also applies to static-code analysis.)

There are two basic models of automated-code testing tools:

- **Static-code analyzers (SAST)**
- **Dynamic-code analyzers (DAST)**

Static analyzers collect information based on looking directly at the syntactical structure of code and drawing conclusions about the program's behavior. Dynamic analyzers take a different approach, wherein they evaluate how the code actually behaves when it is interacting with the real world, taking state information into account.

## 4. Static-Code Analysis Advantages

Static-code analysis can provide an early security warning system for developers as they write sections of code. A static-code analysis tool:

- *Reduces cost*. This type of analysis greatly reduces the cost of eliminating security defects in software. The earlier an error is detected, the lower the cost of remediation.
- *Finds security vulnerabilities at specific locations*.
- *Is quick and less expensive*. Because this analysis tool is quicker, it is therefore a less expensive means of fixing security vulnerabilities.
- *Provides granularity and scale*. This degree of detail is possible because an automated static-code analysis tool can scan the entire code base rather than just samples of code.
- *Provides immediate feedback*. An analysis tool can be run repetitively, such as after each batch of mitigations is complete.
- *Finds specific classes of problems*. The tool is effective at detecting certain classes of problems that dynamic-code analyzers cannot always find, such as buffer overflows and SQL injection flaws. An alternative solution to using a dynamic-code analyzer is to deploy manual testing by expert testers.
- *Examines how data flows through an application*. In addition to investigating data flows, this tool looks at how specific types of data, such as confidential and personal data, are processed and protected.
- *Examines how sensitive data is encrypted and decrypted*.
- *Uncovers logic flaws*. The tool's discovery of an application's logic

## 5. Static-Code Analysis Limitations

There are, however, limitations to what a static-code analyzer can accomplish, including that it:

- *Requires trained software developers*. The testing involved needs to be conducted by trained software developers who fully understand the code.
- *Possibly does not support all programming languages*. A particular code analyzer might not support all programming languages.
- *Produces a false sense of security*. Static-code analysis can foster the belief that everything is being addressed, when in fact this is not always the case. *Is unable to find configuration problems*.
- *Cannot find runtime problems*. It cannot find vulnerabilities introduced in the runtime environment, such as authentication problems and access control issues.
- *Cannot identify insecure cryptography*.
- *Does not detect noncompliance with a security policy*.
- *Does not identify back doors*.
- *Cannot diagnose memory leaks and concurrency errors*.
- *Can be inconvenient to use*.

Three specific ways are:

1. Automated tools can produce spurious warning/error messages that the developers cannot silence. If developers feel comfortable ignoring compiler warnings, the compile phase will eventually be filled with warnings that are ignored, even though they may include unresolved security vulnerabilities.
2. Since these tools take a long time to run, developers sometimes do not bother running them.

3. Many of these tools have difficulty analyzing code that can't be compiled. Analysts frequently can't compile code because they don't have the right libraries, all the compilation instructions, or all the code.

## 6. Dynamic-Code Analysis Advantages

Dynamic-code analysis has several advantages, particularly in identifying runtime security flaws (Shema). It can:

- Identify vulnerabilities in a runtime environment. Dynamic-code analysis deals with real runtime values, which static-code analysis cannot do.
- Test applications when there is no access to the actual code.
- Find false negatives. This analysis can identify vulnerabilities that might have been false negatives in the static-code analysis.
- Provide validation of static-code analysis findings.
- Detect vulnerabilities that static analysis cannot.

## 7. Dynamic-Code Analysis Limitations

Dynamic-code testing technology is not perfect and does exhibit some limitations. In particular, it has limited scope. A dynamic-code tester will test for all activities it is directed to test, but if certain options or activities are not specified to the tool, it may miss testing those options or activities (Lepofsky).

There are two more type application testing types recently emerged as **IAST (Interactive Application Security Testing)** and **Mobile AST**. The IAST combines elements of Static and Dynamic code analysis simultaneously. It's typically implemented as an agent within the test runtime environment. Whereas Mobile AST uses a combination of behavioral analysis using static and dynamic techniques to discover malicious or potentially risky actions the app may be taking unknown to the user

## 8. Market overview

Research and Market has announced in their "Security Testing Market - Global Forecast to 2021" report that the global security testing market is expected to grow from USD 3.31 billion in 2016 to USD 7.61 billion by 2021 dominated by application security testing segment with largest market size while BFSI (Banking, Financial services and Insurance) vertical is expected to grow at a highest rate. (Security Testing Market , 2016)

The security testing market is divided upon Network, Application, and Device verticals and vendors market tools such as Penetration Testing, Web Testing, Automated Testing, and Code Review. This market growing rapidly as the organizations are more focused on protection of valuable assets such as web & mobile applications and customer data.

Application security testing is estimated to have the largest market size in 2016 in Security testing as the increased deployment of web and mobile applications and the protection of these applications is the prime objective of the organizations. The growth in the application security testing is associated with the rise in security breaches targeting enterprise applications, whereby hackers try to gain access to sensitive data.

All the testing methods can be delivered as either a tool or a service and adaptation is variable. The most popular code analysis method is Dynamic Application Security Testing (DAST), followed by Static Application Security testing (SAST). IAST and Mobile AST are very recent additions in code analysis methods. Organizations are rapidly deploying security testing solutions either on-premises or on cloud. The demand for cloud-based security testing solutions is increasing due to the cost-effective and time-efficient features of cloud; its growth is specifically high in enterprises, where low cost solutions are much required.

Some of the major technology vendors include:

- Hewlett Packard Enterprise (U.S.)
- IBM Corporation (U.S.)
- Qualys, Inc. (U.S.)
- WhiteHat Security (U.S.)
- Acunetix. (U.S.)
- Veracode (U.S.)
- Checkmarx (Israel)
- RAPID7 (US).

## 9. Market Challenges

According to the SANS Institute's "2015 State of Application Security" survey, the majority of security leaders feel the effectiveness of their application security programmes needs to be improved in order to lower the risk of a successful attack. Below are most common application security market challenges.

1) **Inconsistent demand**

Testing demands are not always consistent as most companies no longer follow fixed-release schedules

2) **Only tools are no more sufficient**

Application security changes constantly with new threats, emerging attacks and evolving compliance regulations. Organization need someone with the expertise for in-depth manual testing and result interpretation to keep testing and prevention strategies current.

3) **Scarce Security Workforce**

According to CISCO's annual security report there are few internal security experts looking for new roles, with 1 million unfilled IT security jobs worldwide. The human augmentations are very vital element in application security testing.

4) **Reliance on single tool**

Over the years, automates testing tools are become must sophisticated. Each tool has its own unique strength and implementation scenarios, which may not fir into any customization need such as testing method or regulatory requirements.

## 10. Comparative Analysis of Major Products

| Vendor & Product Name | IBM AppScan | HP Fortify | Qualys WAS | WhiteHat Security | Veracode App. Security | Acunetix WVS | Checkmarx CxSAST | Rapid 7 |
|---|---|---|---|---|---|---|---|---|
| On-premises Product | √ | √ | - | √ | - | √ | √ | √ |
| Cloud or Service | √ | √ | √ | √ | √ | √ | √ | √ |
| Testing Method | SAST, DAST IAST, Mobile AST | SAST, DAST, IAST, Mobile AST, & RASP | DAST | SAST (only for limited programing language), DAST, Mobile AST | SAST, DAST, Mobile AST | DAST, IAST (not available for JAVA) Mobile AST | SAST (limited integration with leading WAFS),IAST, Mobile AST | DAST |
| Automatically scan un-compiled / un-built code | √ | √ | √ | √ | √ | √ | √ | √ |
| Integrated SDLC | √ | √ | √ | √ | √ | √ | √ | √ |
| Supported coding language | Java, JS, php, python, Groovy, Apex, Android, Perl, .net, HTML, Windows Mobile, C++, Scala, Ruby, ASP.net, PL/SQL | | | | | | | |
| Common supported vulnerabilities | SQL Injection, Cross-Site Scripting, Code Injection, Buffer Overflow, Parameter Tampering, Cross-Site Request Forgery, HTTP Splitting, Log Forgery, Denial of Service, Session Fixation, Session Poisoning, Unhandled Exceptions, Unreleased Resources, Un-validated Input, Dangerous Files Upload, Hardcoded Password, etc. | | | | | | | |
| Supported standard | OWASP Top 10 2013, OWASP Mobile Top 10, SANS 25, HIPAA, CWE Mitre CWE, FISMA, PCI DSS, MISRA, BSIMM | | | | | | | |
| False positive ratio | Near <0 | Near <0 | <5 | Near <0 | Near <0 | <5 | <5 | <5 |
| Reporting & dashboards | PDF, RTF, CSV or XML | | | | | | | |

## 11. About CyberDB

CyberDB (www.cyberdb.co) is the leading global research databank for Cyber solutions and vendors.

CyberDB database includes over 1,200 vendors and 5,000 products, categorized into 8 main cyber categories and 146 sub-categories. The company publishes market researches and summaries on bi-weekly basis on cyber categories.

The database sis being used by VC's, multinationals, CISO's and system integrators worldwide to help them navigate through the dynamic cyber landscape.

In addition, CyberDB offers its customers Consulting Services for Cyber Product Strategy, Cyber Technology Scouting and tailored Market researches.

CyberDB is established by the founders of Stratechy, strategy consulting practice that has been working with management teams of Hi-Tech vendors to shape their product strategy turn-around and design and execute their Go-To-Market plan

Among its customers, are NEC Corporation, Amdocs, Nice, Adallom (Microsoft), Brother, Cyberbit (Elbit), Rafael, S21Sec, NextNine, CyberX and Sixgill,

Please contact CyberDB at **info@cyberDB.co** or visit us in **www.cyberdb.co**, on **Twitter** or **LinkedIn**

## 12. References

Lepofsky, R. *The Manager Guide to Web Application Security.* New York: Apress.

Michael Sikorski, A. H.. *Practical Malware Analysis, The Hands-On Guide to Dissecting Malicious Software.* San Francisco: William Pollock.

Sebesta, R. W. (2016). *Concepts of Programming Languages 11ed. Global Ed.* Essex, England: Pearson Education Limited.

*Security Testing Market* . (2016, Dec 16). Retrieved from www.prnewswire.com: http://www.prnewswire.com/news-releases/security-testing-market-by-type-deployment-mode-organization-size-vertical--region---global-forecast-to-2021---research-and-markets-300351500.html

Shema, M. *Anti-Hacker Toolkit 4ed.* New York: McGraw-Hill Education.