



Inside Application Security

The Truth About White Box Testing vs.
Black Box Testing



Table of Contents

Executive Summary 3
White and Black Box Testing in the Software Development Lifecycle 3
White Box Testing 4
Black Box Testing 5
White Box vs. Black Box Technologies 6
Conclusion 8
About Cenzic 9



Executive Summary

CISOs, information security managers, quality assurance staff and application developers are faced with the enormous responsibility of keeping Cloud, Mobile and Web applications secure from the ever growing menace of hackers and internal threats alike. Newly surfacing threats are overwhelming information security teams. With these applications constantly evolving, finding vulnerabilities is a challenging, costly and time-consuming undertaking.

How can information security personnel protect sensitive data – and ultimately, the corporate reputation – without exhausting internal resources, overspending the budget or being forced to use costly manual penetration testing using external consulting firms?

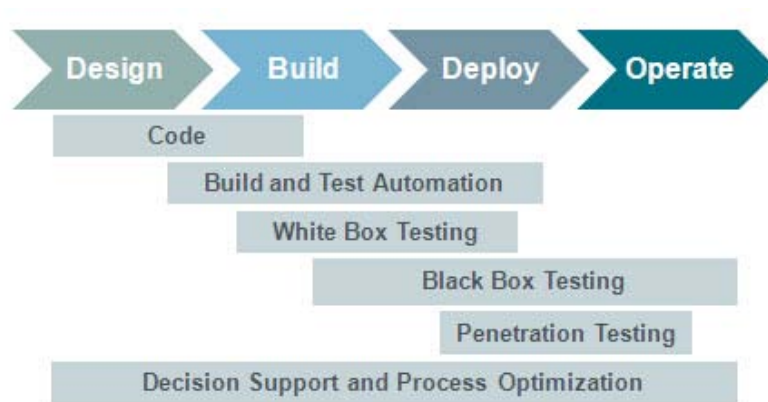
Security teams are dealing with this ominous challenge with myriad of solutions, some highly ineffective. However, as the market matures, companies are applying somewhat effective, but not complete solutions like white box testing tools. Ultimately the challenges of architecture, API usage and integration prevent white box testing tools from having a truly direct impact on the overall security of an application. It is critical to understand that white box analysis tools do not directly find all the risks inherent in applications – period.

This paper explores the role of white box vs. black box testing. White box testing technologies have a definite but limited use and value. From an application security perspective it must be understood that significant blind spots come with white box testing. Ultimately, white box testing is not sufficient to secure your Cloud, Mobile and Web applications – simply put organizations that rely solely on white box technologies will be exposed to vulnerabilities in their applications, thus making it an ineffectual method of testing real-world risks. This paper will demonstrate black box or dynamic testing is ultimately the appropriate solution for truly securing Cloud, Mobile and Web applications.

White and Black Box Testing in the Software Development

The software development lifecycle (SLDC) is composed of design, build, deployment and operation.

When mapping security needs to these process steps it is useful to keep in mind the factors outlined below.



Security Requirements

Security requirements must be built into application design from the point of conceptual development (i.e., the white-board phase) and denote specific functional characteristics of the application.



Security Controls within Design

Security best practices and security controls should be instituted within the functional planning, design and architecture phases then formally specified and considered prior to any actual development on the application. The use of application security checklists provide a baseline for needed security mechanisms while also providing a security awareness tool for developers.

Build

When actual coding and implementation begins, the process should be governed by the specific security requirements specified in the earlier stages of the SDLC process.

Integration Testing

Test cases should be developed that demonstrate characteristics as defined by the security requirements, design requirements and coding practices of the organization. Security testing prior to application release should involve specific application security vulnerability tests, to ensure that the application is resistant to common types of attack and real-world attack scenarios.

Deployment

Security testing in a staging environment that mirrors the production environment must be performed at this stage. The suite of tests must be carried forward from integration testing. A subset of these tests, that are non-invasive, should be carried forward to the operations and maintenance phase.

Operations and Maintenance

Once deployed, applications should be frequently assessed while they are in production.

	Build	Integration	Deploy	Operations
White Box Testing	Run for each major milestone			
Black Box Testing	Run with each build	Run with each build	Run with each build	Run with each change in management procedure

White Box Testing

Source code analysis tools are commonly called white box technologies, because they have visibility into the source code that comprises applications. White box testing tools (source code analysis tools) look for the use of insecure functions and other bugs by inspecting the raw application source code.

Similar to a spelling and grammar checking tool built into a word processor, they operate both on the basis of a signature set for known bad functions (words) and a rule set for identifying sentence-level patterns (code grammar). Source code analysis tools automate the analysis of the code by finding code errors that result in vulnerabilities. They also target bad programming practices. When used during the development cycle, they can help to eliminate common programming mistakes.

These technologies have clear, but limited benefit for a security program; however, it is important to understand how to use them properly.

By using white box testing early in the development life cycle, organizations can eliminate security vulnerabilities before they are deployed in a live system.



White box testing tools have the following benefits:

Reduce Faulty Code Base

Proactive use of white box technologies during the development cycle can minimize the risk of producing faulty code under tight deadlines.

Educate of Developers in Secure Coding Practices

Using white box testing tools can help to educate the developers and avoid the use of insecure functions.

Automate of the Repetitive Aspects of Source Code Analysis

Repetitive tasks when performed by humans often result in errors slipping through or being overlooked. White box testing tools can automate the task of sanity checking code so that fewer mistakes are made. This frees up a developer to focus on the code rather than analyzing code line-by-line for security vulnerabilities.

Conform with Internal Security Standards

Organizations can use white box testing tools to enforce good coding practices and conform to their own internal coding standards and guidelines.

Improve the Security Legacy Code

Some pieces of code used by an organization may have originated by developers that are no longer with the organization. Additionally, this code may not be very well understood. White box testing can be used on legacy code to find and fix some basic vulnerabilities much quicker than a manual review of a legacy code base.

Black Box Testing

Organizations must understand the importance of black box testing, also called dynamic application security testing (DAST) to truly secure their Cloud, Mobile and Web applications. While white box testing looks inside applications at the source code, black box testing looks at how the application functions. It analyzes a working version or final release of Cloud, Mobile and Web applications. This means that the application must be deployed and connected with any components with which it will be interacting. Examples include the Web server, a backend database, outside modules or widgets, the file directory system of the operating system and any other items. However the black box testing tools does not need to know anything about the underlying system – it only has to be able to interact with the application.

Since it is looking at a deployed version of the application, black box testing can analyze the interaction between the different components and detect vulnerabilities not wholly embedded within the application, such as:

- Can an exploit within the application enable a user to access the database, OS or other component?
- Does the Web server provide access to pages the application considers otherwise protected?
- Can a minor issue with the application cause the database to be corrupted or fail?

Black box testing is optimal for looking at an entire functioning Cloud, Mobile or Web application and detecting security concerns. It focuses on the externally visible behavior of a running application. The application can be an application currently in development or a legacy application which has not been updated in the recent past.

Black box testing tools have the following benefits.

Analyze an Entire Deployed Cloud, Mobile or Web Application

Black box testing can analyze an entire application as it is deployed with all its components.

If vulnerability is present in the way two components interact with each, black box testing can detect it.



Provide Development Feedback via Testing on a Staging Server

By integrating black box testing as part of the software development lifecycle, regular and early feedback can be provided to development. This will help catch any security concerns before they get to the deployment stage.

Prevent the Gathering of Additional “Hacking Information”

Black box testing implements many of its detection techniques by looking at results from unexpected input. By looking at the error messages displayed for abnormal data or watching how the application interacts, hackers can gain information about how the application is deployed. This in turn aids their ability to effectively hack the application. Black box testing can effectively prevent this from happening.

In the recent past, source code scanning competed with automated black box assessments. Today, most organizations recognize the need to use both. White box testing is used to identify some security concerns early and to enforce proper coding techniques; black box testing to is used to analyze the application throughout the build, test and staging phases.

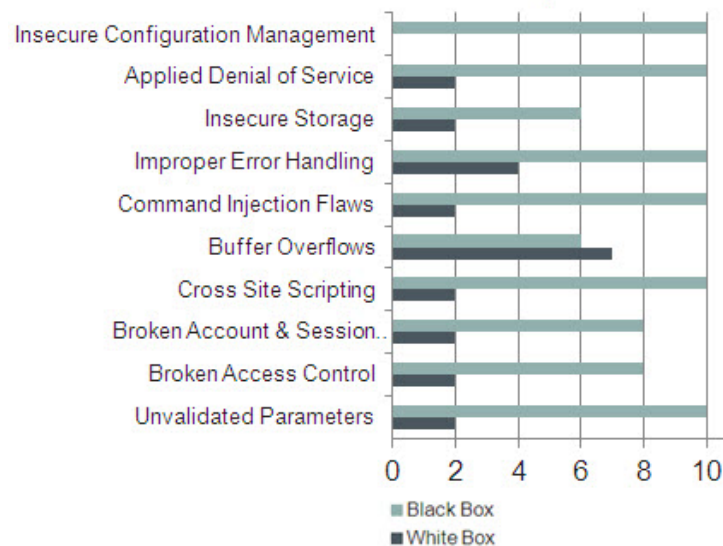
White Box vs. Black Box Technologies

Today, enterprise class deployments of Web application security technologies are gaining momentum, but there is still a tendency to view white box technologies and black box application testing technologies as solving the same problem from different angles. Until the enterprise market matures, the two technologies are likely to continue to be seen as complementary of one another, or trade-offs. In reality, they solve two entirely different problems.

Where Cloud, Mobile and Web applications are concerned, black box technologies offer superior benefits in terms of finding and assisting in the remediation of the most severe vulnerabilities. More importantly, black box technologies are the only viable choice for assessing the security of deployed applications – whether deployed in production or within a test environment.

The graph below shows the real relationship between dynamic application security testing solutions (black box) and automated white box security solutions in terms of the OWASP Top 10. This graph makes a general comparison of black box testing solutions (application vulnerability assessment) to a white box testing tools.

White Box vs. Black Box and the OWASP Top 10





Within the graph above the scores mean the following:

- 0 - 2 means the vulnerability cannot be directly detected or the vulnerability detection capability may be present, but is highly unreliable. This is typically associated with high false positives and high false negatives. The process of detection is similar to guessing based on incomplete evidence.
- 2 - 4 represents a basic ability to detect vulnerability, but only in very limited cases. It is prone to a high percentage of false positives and extremely high false negatives.
- 5 represents the ability to reliably detect vulnerability, but in general cases. It is prone to some percentage of false positives and a high percentage of false negatives.
- 6- 10 represents the ability to detect multiple variations of a vulnerability under a broad range of conditions. It is prone to a low percentage of false positives and a low percentage of false negatives.

Here is a further examination of the pros and cons associated with each.

White Box Testing

Applicable stages: development, integration

Pros:

- Can find certain vulnerabilities yearly in the development of the application
- Can point to a specific line of code

Cons:

- Prone to false positives
- Development resource consuming
- Does not simulate a real world attack
- Not effective for finding “real” vulnerabilities; only showing possible ones
- Complex implementation

Black Box Testing

Applicable stages: development, integration, deployment, operations

Pros:

- Simulates real-world attacks
- Can be performed quickly and automatically
- Assessments may be repurposed from stage to stage of the process

Cons:

- Requires compiled executable
- Tests only visible interfaces

White Box Testing: The Reality

There are limitations in what white box testing tools (source code analysis tools) can really do. Below is a summary of their limitations.

- **They do not protect against the latest threats**
One of the greatest weaknesses of white box solutions is that they are typically not updated with the latest security vulnerabilities. Programmers may catch security flaws in their code, but other vulnerabilities are invisible, because the source code analyzer does not have patterns for those vulnerabilities. An organization that relies on a source code analyzer as its only means of finding Cloud, Mobile and Web application vulnerabilities will face major risk.



- **They are not suited for use against large, complex, or heterogeneous applications**

Many applications are only one piece of a broader application framework. Source code analysis provides a limited picture of the overall security of the application. Source code analysis tools are blind to the logical workflow of the application as a whole and are quite often incompatible with the various platform and source code types involved within the application framework.

- **They do not test application inputs or test input validation**

Source code analysis tools do not test for input validation flaws, because they analyze code and not application inputs. Since the mechanisms of reading, processing and handling input are not interactively tested, limitations or design flaws within input validation schemes will be invisible to the tool. The majority of the most severe application threats come from bypassing input validation filters to perform SQL Injection and Script Injection Attacks (XSS). Source code analysis tools are very limited in their ability to detect these vulnerabilities since they have no ability to attempt to bypass defenses by actively attacking an input filter or parser.

- **Cannot detect design, integration, or architecture flaws**

Cloud, Mobile and Web applications are often designed with complex middleware and implicitly trust these environments to sanitize input and process session data securely. Yet, application code may contain vulnerabilities through lack of key security protections that are assumed to exist elsewhere. Commonly, an application trusts input from a component of a server or application framework. These assumptions can result in vulnerabilities because of mistaken trust in the security of external components that cannot be scanned with white box technologies.

Additionally, development environments like PHP, or other environments, may have their own vulnerabilities that are transferred to the application. Commercial frameworks and platforms, (for example, Macromedia Cold Fusion) contain insecure default configurations and components that are vulnerable to attack when integrated into a custom application. Source code analysis tools cannot detect the improper implementation of such components, or their flawed default configuration.

For example, a financial services company has a Web application to store and export the preferences of a Web user as their workflow crosses multiple Web applications. That preference tracking application has components resident on 20 different Web servers, of different types and version, each Web server running different kinds of middleware. A functional application-layer script ties those servers together: Perl, PHP, Java and Ruby. Aside from the different scripting mechanisms, the application itself is written in five different languages. This presents an intractable problem for source code analysis tools.

Conclusion

It is important for organizations to realize they cannot rely on white box testing tools as a means for securing their Cloud, Mobile and Web applications. Organizations that rely solely on white box technologies will be exposed to vulnerabilities in their application portfolio. White Box scanner technologies have a use, but the real value they hold for really securing Cloud, Mobile and Web applications is limited at best. Again, the challenges of architecture, API usage, and integration prevent white box testing tools from having a direct impact on the overall security of an application. They can neither test in run time nor deal with production applications. White box testing tools have their use in finding faulty code and can show the possibility of vulnerability.

The Ultimate Question

So, what should an organization do? How should you start testing and securing your applications? There are three different ways to approach this:

1. If you have in-house experts on application security and have enough budget, purchase both black box and white box testing solutions. Cenzic offers an integration of its black box testing solution suite with the best source code solutions in the industry.



2. If you have in-house experts but don't have enough budget, start by purchasing the black box testing solution. It will give you broad coverage throughout the Software Development Lifecycle.
3. If you do not have in-house experts, start with using a managed service offering. For example, Cenzic Cloud Managed service tests your Cloud, Mobile and Web applications remotely and delivers detailed results and a walk-through with Cenzic security experts. No software or hardware is required. Once you are ready to move everything in-house and deploy on-premise software, all the data can be migrated instantly when you install Cenzic Enterprise (software).

About Cenzic

Cenzic provides an application security intelligence platform to continuously assess Cloud, Mobile and Web vulnerabilities. This helps brands of all sizes protect their reputation and manage security risk in the face of malicious attacks. Today, Cenzic secures more than half a million online applications and trillions of dollars of commerce for Fortune 1000 companies, all major security companies, government agencies, universities and SMBs.

Cenzic Products

Cenzic Enterprise	Cenzic Enterprise is a software solution that assesses the security of Cloud and Web applications and supports security risk management throughout the software development lifecycle. Cenzic Enterprise provides a company-wide view of security vulnerabilities and status to executives as well as customized views for other users from a Web-based dashboard.
Cenzic Desktop	Cenzic Desktop is a single-user version of Cenzic Enterprise. It is designed for the power user that wants to run their security assessments on Cloud and Web applications from a single system.
Cenzic Managed Cloud	With Cenzic Managed Cloud, Cenzic's security experts remotely perform full vulnerability testing on Cloud, Mobile and Web applications. From a Web-based dashboard, users can request assessments, view results, run reports, analyze trends and re-test applications to verify remediation efforts.
Cenzic Cloud	Cenzic Cloud allows users to test their own Cloud and Web applications for basic attacks and receive actionable results all within their own Web portal – no security experts needed.
Cenzic Hybrid – Software + Cloud	Cenzic Hybrid provides access to both Cenzic Enterprise software and Cenzic Managed Cloud services. Vulnerability testing can be done either by using the software on premise or by leveraging Cenzic's expert security services team.
Cenzic Mobile	Cenzic Mobile service is delivered as a managed service. Cenzic's security experts remotely perform full vulnerability testing on mobile applications. All testing is managed from a Web-based dashboard where users can request assessments, view results, run reports, analyze trends and re-test applications to verify remediation efforts.
Cenzic Services	Cenzic services and training help those responsible for Cloud, Mobile and Web application security to implement best practices and procedures to protect data from hacker attacks.