What is Gray Box Testing?

Gray box testing (a.k.a grey box testing) is a method you can use to debug software and evaluate vulnerabilities. In this method, the tester has limited knowledge of the workings of the component being tested. This is in contrast to black box testing, where the tester has no internal knowledge, and white box testing, where the tester has full internal knowledge.

You can implement gray box testing as a form of penetration testing that is unbiased and non-obtrusive. In these tests, the tester typically knows what the internal components of an application are but not how those components interact. This ensures that testing reflects the experiences of potential attackers and users.

Gray box testing is most effective for evaluating web applications, integration testing, distributed environments, business domain testing, and performing security assessments. When performing this testing, you should create clear distinctions between testers and developers to ensure test results aren't biased by internal knowledge.

The Gray Box Testing Process

In gray box testing, the tester is not required to design test cases. Instead, test cases are created based on algorithms that evaluate internal states, program behavior, and application architecture knowledge. The tester then carries out and interprets the results of these tests.

When performing gray box testing, you take the following steps:

- Identify and select Inputs from white and black box testing methods.
- Identify probable outputs from these inputs.
- Identify key paths for the testing phase.

- Identify sub-functions for deep-level testing.
- Identify inputs for sub-functions.
- Identify probable outputs from sub-functions.
- Execute sub-function test cases.
- Assess and verify outcomes.
- Repeat steps 4-8.
- Repeat steps 7 and 8.

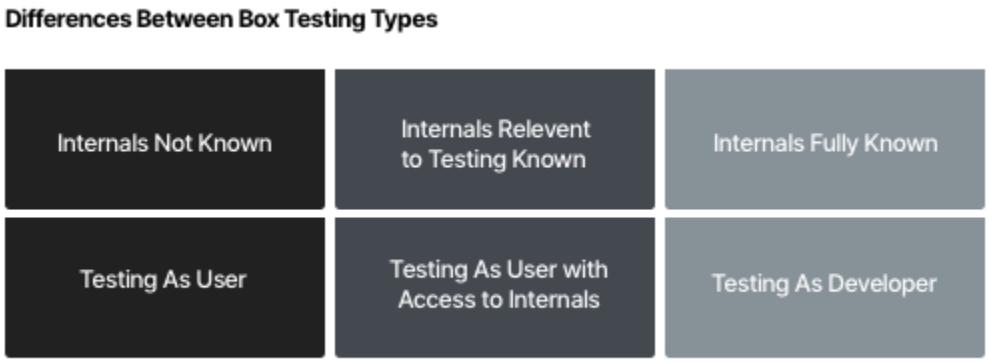
Gray Box, Black Box, and White Box Testing

Gray box testing is a middle ground between white box and black box testing.

White box testing is an approach that involves testing an application based on knowledge of its inner workings, code, and architecture. It can help discover security issues, data flow errors, and bugs in seldomly-used paths.

Black box testing evaluates a product from the user's perspective, with no knowledge of its inner workings. Therefore it is an end-to-end approach that tests all systems that impact the end-user, including UI/UX, web servers, database, and integrated systems.

Grey box testing combines the benefits of the black box and white box testing. On the one hand, tests are performed from the user's perspective. On the other hand, testers do use some inside information to focus on the most important issues and identify the weaknesses of the system.



Differences between white box, black box, and gray box testing

Gray Box Testing Techniques

Gray box testing techniques are designed to enable you to perform penetration testing on your applications. These techniques enable you to test for insider threats, such as employees attempting to manipulate applications, and external users, such as attackers attempting to exploit vulnerabilities.

With gray box testing, you can ensure that applications work as expected for authenticated users. You can also verify that malicious users cannot access data or functionality you don’t want them to.

When performing gray box testing, there are several techniques you can choose from. Depending on which testing phase you are in and how the application operates, you may want to combine multiple techniques to ensure all potential issues are identified.

Matrix Testing

Matrix testing is a technique that examines all variables in an application. In this technique, technical and business risks are defined by the developers and a list of all application variables are provided. Each variable is then assessed according to the risks it presents. You can use this technique to identify unused or un-optimized variables.

Regression Testing

Regression testing is a technique that enables you to verify whether application changes or bug fixes have caused errors to appear in existing components. You can use it to ensure that modifications to your application are only improving the product, not relocating faults. When performing regression testing, you need to recreate your tests since inputs, outputs, and dependencies may have changed.

Pattern Testing

Pattern testing is a technique that evaluates past defects to identify patterns that lead to defects. Ideally, these evaluations can highlight which details contributed to defects, how the defects were found, and how effective fixes were. You can then apply this information to identifying and preventing similar defects in new versions of an application or new applications with similar structures.

Orthogonal Array Testing

Orthogonal array testing is a technique you can use when your application has only a few inputs that are too complex or large for extensive testing. This technique enables you to perform test case optimization, where the quality and number of tests performed balance test coverage with effort. This technique is systematic and uses statistics to test pair-based interactions.

Gray Box Testing Pros and Cons

When determining whether or not to use gray box testing, you should consider the following pros and cons. These can help you determine if gray box testing is appropriate for your testing situation and how much value it may provide.

Pros

Pros of gray box testing include:

- Clear testing goals are established, making it easier for testers and developers
- Testing accounts for a user perspective, improving the overall quality of products
- Testers do not need to have a programming expertise
- Testing methods create more time for developers to fix defects
- It can provide the benefits of both black and white box testing
- It can eliminate conflicts between developers and testers
- It is cheaper than integration testing

Cons

Cons of gray box testing include:

- It can be difficult to associate defects with root causes in distributed systems
- Code path traversals are limited due to restricted access to internal application structure
- It does not allow for full white box testing benefits since not all internals are accessible
- It cannot be used for algorithm testing
- Test cases can be difficult to design