Black Box vs. White Box: A Demonstration of Two Testing Techniques

on an Automated Teller Machine (ATM)

A Report by

Mark Kardash

SDEV360: Secure Software Engineering

Professor David Gabello

4 December 2023

University of Maryland Global Campus

Black box testing and white box testing are two techniques with one common goal: Finding flaws in a program or application. However, while white box testing focuses on the application’s external components and functional requirements, completely disregarding its inner workings, white box testing, on the contrary, focuses on the latter. Also, while one can automate black box testing, white box testing does not provide this ability (Goldfine, n.d.). To make the job of testing easier, developers create test cases. These are sets of steps and input to be applied to a program, which verify that it functions properly, and detect any issues it may have (Goldfine, n.d.). This report will apply one black-box, and one white-box test case, to the hypothetical ATM machine that was the subject of previous reports. The two test cases are found below:

Black Box Test Case:

When operating an ATM, one of the most important steps is for the user to verify their identity. This is done after the customer’s card is inserted into the ATM and properly read. Identity is verified through a 4-digit PIN linked to the user’s account. The following test case uses this scenario as its basis.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Step: | Actions: | Expected Results: | Comments: |
|  | Activate ATM. | ATM Screen turns on.  Customer prompted to select bank. | ATM most likely activated by a button. A “Welcome” message may appear. |
|  | Customer Selects Bank | Selection processed. Customer prompted to insert card into ATM. |  |
|  | Customer inserts card into ATM. | ATM recognizes the card. Customer is prompted for a PIN. | The PIN must always have 4 digits. |
|  | Customer enters incorrect PIN. | ATM does not recognize the PIN. Customer is again prompted for a PIN. | The incorrect PIN is entered specifically to test the ATM’s reaction and functionality. |
|  | Customer repeats Step 4 three (3) times. | After the third time entering the PIN, a simple error message pops up on the screen, informing the user of too many attempts to enter PIN. The machine locks the user out for a certain period (Ex. 24 hours). The card may also be taken in by the ATM as a precaution. If that happens, the user must contact the bank. | Number of times for repeating Step 4 may vary before the expected result is reached.  The error message displayed must be simple enough to not reveal any unnecessary information to the user. |

Conclusion:

The test data involved in this test case is user data, such as their four-digit PIN, and any information that the ATM reads from the card. The issue, in this case being the incorrect PIN, is mitigated by the machine’s design. This is because this particular ATM is programmed to throw an error message and lock the user out after too many attempts. This test case does not require any external tools to be applied, as it is performed “from the outside”, depending solely on the actions of the user.

White Box Test Case:

The next test case, as explained earlier, will delve more into the internal functionality of the ATM machine. This means that the test will involve a correct PIN and examine actions beyond that stage. One of the most important stages of an ATM’s use is when it prompts the customer four options: Withdrawal, Balance Inquiry, Deposit, and Transfer (Snow, 2016). When tested, each of these options must have its own, separate case. For this report, only one of the options will be examined.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Step: | Actions: | Expected Results: | Comments: |
|  | Activate ATM. | ATM Screen turns on.  Customer prompted to select bank. | ATM most likely activated by a button. A “Welcome” message may appear. |
|  | Customer Selects Bank | Selection processed. Customer prompted to insert card into ATM. |  |
|  | Customer inserts card into ATM. | ATM recognizes the card. Customer is prompted for a PIN. | The PIN must always have 4 digits. |
|  | Customer enters valid PIN. | ATM recognizes PIN. User is presented with four options: Withdrawal, Deposit, Balance Inquiry, and Transfer. | Each of these options must get its own test case if tested. |
|  | Customer selects “Withdrawal”. | ATM processes option. Customer prompted for withdrawal amount. |  |
|  | Customer enters larger amount than the one available for withdrawal. | A message appears, informing the user that the amount entered is not available for withdrawal. User receives no cash. | Once again, the message must be as basic as possible, so as not to enable an information leak, while still giving enough clues as to what went wrong. |

Here, the testing goes beyond the authentication of the PIN, delving into the inner process of how the ATM handles the withdrawal. There is a lot of data involved in this test. This is the information that the user has on their card (Card number, bank account number, cash amount, etc.), as well as the commands that the user gives (The request for withdrawal). Once the bank is selected and the customer’s PIN verified, the responsibility for the remaining task passes to the machine itself, as it is now authorized to read the information from the card. This is done with the help of a scanner inside the ATM. By entering the requested amount, the user triggers the ATM to read the card information and compare it to their request. This operation is done at code level, and the code producing it may look something like this (Example, non-functional code):

If Balance = 0:

Print (“No Funds Available”);

Elif WITHDRAWAL\_AMT > Balance:

Print (“Amount Entered Not Available. Please enter smaller amount:”);

WITHDRAWAL\_AMT = input();

Elif WITHDRAWAL\_AMT ≤ Balance:

Print (“Please wait for funds to dispense...”);

Balance. dispense();

Explanation:

As is probably quite clear from the code above, despite it not being very accurate, when the ATM machine has a balance of 0, a message is displayed notifying the user that funds are not available, due to their account being empty. If a user enters an amount that is greater than the balance available in the bank account, the machine displays a message to inform them that it is not available. It then asks the user to enter a smaller amount, and displays the prompt for it. Finally, when the customer (user) enters just the right amount, the ATM activates a function that dispenses the funds requested.

All the above decisions were taken through the ATM’s internal workings, with minimal input from the user, which makes white box testing the ideal method for this case.

**Conclusion:**

Testing of any kind is extremely important, both during software development and after the product is finished. The distinction between black box and white box testing allows for more accurate results, as internal and external components each get their own, separate treatment. By having as many individualized approaches to an application’s components as possible, developers and testers can significantly reduce the occurrence of issues and bugs.

References

Goldfine, A. (n.d.). *Types of Testing Methods*. Leocontent.umgc.edu. <https://leocontent.umgc.edu/content/dam/course-content/tus/sdev/sdev-360/document/Types%20of%20testing%20methods.pdf>

Snow, J. (2016, September 29). *4 ways to hack an ATM — video*. Kaspersky Cyber Security Solutions for Home and Business | Kaspersky. Retrieved December 4, 2023, from <https://www.kaspersky.com/blog/4-ways-to-hack-atm/13126/>