WEB APPLICATION TEST PLAN

COMP307 Lab Assignment 8

Written by: Kevin Ma (#300867968) and Ostap Hamarnyk (#300836326)

Date: Wednesday, November 21, 2018

EXECUTIVE SUMMARY

In this document, test cases have been provided which would address critical security concerns identified by the security team. There are no findings attached to this document as it is not a testing report, but rather a test plan for the Web Application deployed by **Online Appliance Parts**. The security team has identified three areas of the application which may be prone to vulnerabilities and attack. The three areas identified to be at risk include the Identity Management, the Input Validation, and the Business Logic areas of the web application. There have been multiple test cases identified for each of these areas in order to provide guidance to the testing team in how to investigate and remediate the issues. For the test cases mentioned below, it is recommended to use an automated test strategy in order to investigate each of the test cases. Furthermore, it is recommended to deploy testers from various backgrounds to execute the test cases. There should be testers with business background but limited technical expertise, as well as technical testers that are familiar with the codebase of the web application. This ensures that the findings from the execution of the test plan will yield well rounded results which may ensure the top level of security for the web application and the continued success of the business.

TABLE OF CONTENTS

Test Cases Overview

In this section, we have identified the tests we would perform and our reasons for including each test in the test plan. A test is an action to demonstrate that an application meets the security requirements of its stakeholders. Because of this, we have identified three security attributes which we believe should be validated to ensure the success of the online business: Identity Management, Input Validation, and Business Logic.

Identity Management Testing

We plan to do testing for Identity Management because it plays an important role in the business's BAU functions. For the Identity Management Testing, we plan to do the following tests on the *Online Appliance Parts* online application:

OTG-IDENT-001 Test Role Definitions

The objective for this test is to validate that the system roles defined within the application sufficiently define and separate each system and business role to manage appropriate access to system functionality and information.

We will do this test because the different roles assigned to users in this application will determine whether they have elevated access or privileges to tamper with sensitive information which would affect sensitive transactions.

OTG-IDENT-002 Test User Registration Process

The objective for this test is to validate that the system roles defined within the application sufficiently define and separate each system and business role to manage appropriate access to system functionality and information.

We will do this test because it must be verified that users register properly and do not gain access to elevated permissions through unintended user registration processes.

OTG-IDENT-003 Test Account Provisioning Process

The objective for this test is to verify which accounts may provision other accounts and of what type.

We will do this test because there may be mistakes in the automated account provisioning process which may grant accounts elevated permissions which were unintended.

OTG-IDENT-004 Testing for Account Enumeration and Guessable User Account

The objective of this test is to verify if it is possible to collect a set of valid usernames by interacting with the authentication mechanism of the application.

We will do this test because if a hacker can gain access to a user account through these means, this is a critical flaw in the system and it must be remedied immediately.

OTG-IDENT-005 Testing for Weak or unenforced username policy

The objective of this test is to determine whether a consistent account name structure renders the application vulnerable to account enumeration. Determine whether the application's error messages permit account enumeration.

We will do this test because if a hacker can gain access to a user account through brute forcing, this is a critical flaw in the system and it must be remedied immediately.

OTG-IDENT-006 Test Permissions of Guest/Training Accounts

The objective of this test is to determine if there are any flaws in the permissions and privileges granted to the guest/training accounts for this application.

We will do this test because developers often grant elevated permissions to default accounts while they are developing in order to easily test the application. The elevated permissions may have been left granted to the accounts by mistake and would cause great harm to the system if left unattended.

OTG-IDENT-007 Test Account Suspension/Resumption Process

The objective of this test is to determine if there are any flaws in the suspension or resumption process of this application.

We will do this test because there may be some unintended procedures which a hacker may abuse to circumvent the account resumption process. For example, a hacker may abuse the system into granting his/her account discounts based on the amount of their total annual purchases. If the system discovers the hacker, his/her account would be suspended. However, if there are account resumption processes left unchecked in the system, hackers can abuse this weakness and regain access to their account which may have massive discounts enabled.

Input Validation Testing

We plan to do testing for Input Validation because the input provided by the end user can easily “make or break” the online system. For the Input Validation Testing, we plan to do the following tests on the *Online Appliance Parts* online application:

OTG-INPVAL-001 Testing for Reflected Cross Site Scripting

Reflected Cross-site Scripting (XSS) occur when an attacker injects browser executable code within a single HTTP response. The injected attack is not stored within the application itself; it is non-persistent and only impacts users who open a maliciously crafted link or third-party web page. The attack string is included as part of the crafted URI or HTTP parameters, improperly processed by the application, and returned to the victim.

We will do this test because most of this business’ income comes through its online portal/transactions. Thus, it is critical to the business’ success that users do not get infected by XSS attacks, otherwise this would reflect poorly in the customer’s loyalty, the business’ reputation, and the business income.

OTG-INPVAL-002 Testing for Stored Cross Site Scripting

Stored Cross-site Scripting (XSS) is the most dangerous type of Cross Site Scripting. Web applications that allow users to store data are potentially exposed to this type of attack. Stored XSS occurs when a web application gathers input from a user which might be malicious, and then stores that input in a data store for later use. The input that is stored is not correctly filtered. Therefore, the malicious data will appear to be part of the web site and run within the user’s browser under the privileges of the web application. Since this vulnerability typically involves at least two requests to the application, this may also called second-order XSS.

We will do this test because most of this business’ income comes through its online portal/transactions. Thus, it is critical to the business’ success that users do not get infected by XSS attacks, otherwise this would reflect poorly in the customer’s loyalty, the business’ reputation, and the business income.

OTG-INPVAL-003 Testing for HTTP Verb Tampering

HTTP Verb Tampering tests the web application's response to different HTTP methods accessing system objects. For every system object discovered during spidering, the tester should attempt accessing all those objects with every HTTP method.

We will do this test because sometimes developers create many end points for testing and may leave some of them unintentionally in the finished product. For example, developers may create GET endpoints which allow them to quickly retrieve sensitive information from the database while developing new features/testing. Or they may create POST endpoints which allow them to modify sensitive information within the databases. This would give users access to sensitive information which may be abused by users with malicious intent.

OTG-INPVAL-004 Testing for HTTP Parameter pollution

Supplying multiple HTTP parameters with the same name may cause an application to interpret values in unanticipated ways. By exploiting these effects, an attacker may be able to bypass input validation, trigger application errors or modify internal variables values.

We will do this test because this is a test case which is not normally tested with conventional testing. Developers may not be aware of the problem; the presence of duplicated parameters may produce an anomalous behavior in the application that can be potentially exploited by an attacker.

OTG-INPVAL-005 Testing for SQL Injection

SQL injection testing checks if it is possible to inject data into the application so that it executes a user-controlled SQL query in the back-end database. Testers find an SQL injection vulnerability if the application uses user input to create SQL queries without proper input validation. A successful exploitation of this class of vulnerability allows an unauthorized user to access or manipulate data in the database.

We will be doing SQL Injection testing because there may be multiple occurrences in which the user input may create SQL queries into the database in this application. For example, when the user is dealing with account management, as well as during financial transactions and inventory management.

OTG-INPVAL-006 Testing for XML Injection

XML injection testing checks if it possible to inject an XML document into the application. Testers find an XML injection vulnerability if the XML parser fails to make appropriate data validation.

We will be doing this test because XML injection attacks have existed for a long time, so this would be one of the first types of attacks a hacker would attempt.

OTG-INPVAL-007 Testing for Buffer overflow

Buffer overflow errors are characterized by the overwriting of memory fragments of the process, which should have never been modified intentionally or unintentionally. Overwriting values of the IP (Instruction Pointer), BP (Base Pointer) and other registers causes exceptions, segmentation faults, and other errors to occur. Usually these errors end execution of the application in an unexpected way.

We will be doing this test because there are several ways in which the input provided by the user may cause overflow issues. If these issues are not handled properly by the application, the hacker may be able to steal money from the business, crash the system or do other sorts of damage to the business.

Business Logic Testing

We plan to do testing for the Business Logic because this business earns most of its revenue through its online transactions. This means that its business is predominantly focused on its online market so it's business rules and logic should be maintained in strict conditions here to ensure that its business performs well. For the Business Logic Testing, we plan to do the following tests on the *Online Appliance Parts* online application:

OTG-BUSLOGIC-001 Test Business Logic Data Validation

In business logic data validation testing, we verify that the application does not allow users to insert "unvalidated" data into the system/application.

We will do this test because without this safeguard attackers may be able to insert "unvalidated" data/information into the application/system at "handoff points" where the application/system believes that the data/information is "good" and has been valid since the "entry points" performed data validation as part of the business logic workflow.

OTG-BUSLOGIC-002 Test Ability to Forge Requests

In forged and predictive parameter request testing, we verify that the application does not allow users to submit or alter data to any component of the system that they should not have access to, are accessing at that time or in that particular manner.

We will do this test because without this safeguard attackers may be able to "fool/trick" the application into letting them into sections of the application of system that they should not be allowed in at that time, thus circumventing the applications business logic workflow.

OTG-BUSLOGIC-003 Test Integrity Checks

In integrity check and tamper evidence testing, we verify that the application does not allow users to destroy the integrity of any part of the system or its data.

We will do this test because without these safe guards, attackers may break the business logic workflow and change of compromise the application/system data or cover up actions by altering information including log files.

OTG-BUSLOGIC-004 Test for Process Timing

In process timing testing, we verify that the application does not allow users to manipulate a system or guess its behavior based on input or output timing.

We will do this test because without this safeguard in place attackers may be able to monitor processing time and determine outputs based on timing or circumvent the application's business logic by not completing transactions or actions in a timely manner.

OTG-BUSLOGIC-005 Test Number of Times a Function Can be Used Limits

In function limit testing, we verify that the application does not allow users to exercise portions of the application or its functions more times than required by the business logic workflow.

We will do this test because without this safeguard in place attackers may be able to use a function or portion of the application more times than permissible per the business logic to gain additional benefits.

OTG-BUSLOGIC-006 Testing for the Circumvention of Work Flows

In circumventing workflow and bypassing correct sequence testing, we verify that the application does not allow users to perform actions outside of the "approved/required" business process flow.

We will do this test because without this safeguard in place attackers may be able to bypass or circumvent workflows and "checks" allowing them to prematurely enter or skip "required" sections of the application potentially allowing the action/transaction to be completed without successfully completing the entire business process, leaving the system with incomplete backend tracking information.

OTG-BUSLOGIC-007 Test Defenses Against Application Mis-use

In application mis-use testing, we verify that the application does not allow users to manipulate the application in an unintended manner.

We will do this test because this is a test case which is not normally tested with conventional testing. Developers may not be aware of the use cases which may produce an anomalous behavior in the application that can be potentially exploited by an attacker.