Security and Vulnerability Testing of ASP.Net Application

Security Testing to Identify and Fix Vulnerabilities in the Existing Enterprise Product

Case Background

The client is in the field of enterprise solutions to multi nodal retail organizations. Client has evolved systems based on the customer requirements instead of a well thought product road map and well thought system design.

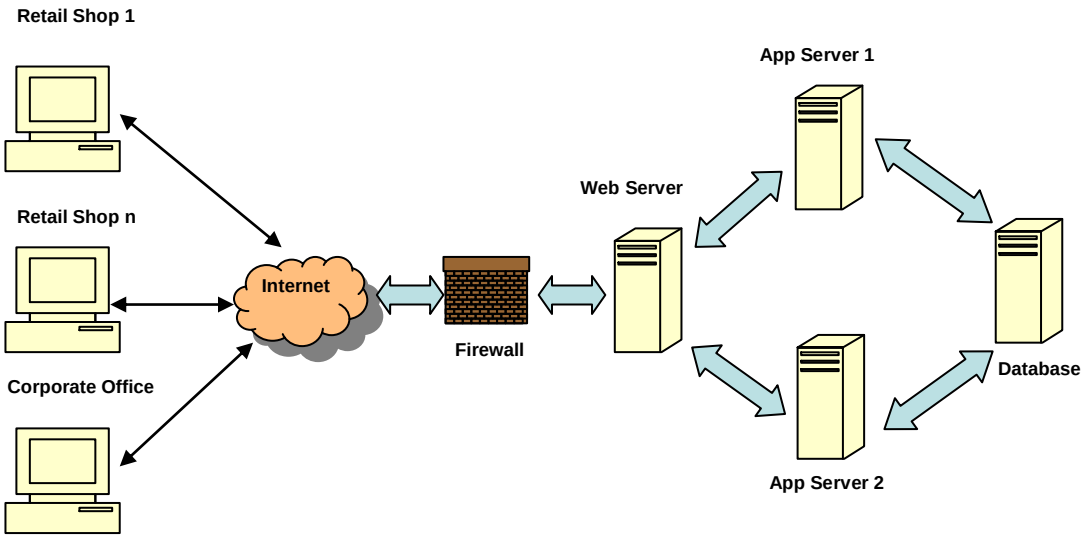
By design system has brought in security vulnerabilities. Client asked AFour Technologies to assist in threat modeling and consequent verification of security fixes.

System Overview

Application Under Test is an enterprise application providing services to Sales Divisions of giant retail organizations across the globe. The system is developed using JSP / Servlet / RDBMS architecture.

System supports application servers such as IBM Websphere, BEA Weblogic, Sun Glassfish, Oracle Application Server. System supports databases such as IBM DB2, Oracle, MySQL. System uses batch process for long, recurring transactions.

Architecture



Goals

1. Threat Modeling of the application
2. Defining Scope of Security Enhancements
3. Verification of Security Enhancement
4. Verification of Functional Integrity
5. Evaluation of Performance Benchmarks

Solution and Approach

1. Did threat modeling to identify following vulnerabilities in the application:
    a. Possibility of Brute Force Attack on Plain Authentication
    b. Session Hijacking Possibilities
    c. XSS Vulnerabilities
    d. HTML Injection Vulnerabilities
    e. SQL Injection Vulnerabilities

2. Verified fixes on Authentication. Used features such as max no. of invalid logins and password encryption to avoid brute force attack.

3. Verified fixes on Session Hijacking. Used defense strategies such as regeneration of session number after successful user authentication, after certain time, certain number of requests. Verified that request for the login page is from the same domain url. Verified that the cookie is associated with same IP.

4. Verified fixes on cross site scripting. Used defense strategies such as - created own functions to validate inputs from user to filter malicious data. For Ajax calls used DWR technology so that the javascript functions are written on the server side. For inline javascript functions created separate javascript files to avoid display of javascript code. Removed all query strings and instead used POST methods to pass data. Encoded special characters using URL encoding methods.

5. Verified fixes on HTML injections. Defence strategy used was to filter html tags in the user input.

6. Verified fixes on SQL injections. Used defense strategies such as - Prevented database keywords in the inputs. Replaced inline string insertions with prepared statements in SQL queries.

7. Verified functional integrity of the system at the end of security fixes

8. Conducted performance test to publish new performance benchmarks at the end of security testing.

Skills Exhibited

- Threat Modeling
- Web Application Fundamentals
- RDBMS / PL-SQL
- HTML
- Java Scripting
- Networking
- Functional Testing

- Performance Testing

Achievements / Value-Adds

- Evaluation of security vulnerabilities
- Released secure version of the enterprise system to the customer
- Known security limitations