CASE STUDY

Strengthening Security: **Application Penetration Testing for a Leading E-commerce Platform**

At a glance:

The operation of an online platform is the backbone of our client's e-commerce business. To protect customer data and maintain platform integrity, they engaged us to perform comprehensive penetration testing. The goal was to uncover and fix any security weaknesses, maintain the integrity of their online platform, and ensure that sensitive customer data are protected. The result is a platform now reinforced with stronger defenses, capable of withstanding cyber threats and safeguarding sensitive customer data with greater resilience.

Industry

E-commerce

Technology

- OWASP ZAP
- Burp Suite
- Nessus

Introduction

The security of web applications is of paramount importance, especially for organizations that handle personal information of their users and customer financial transactions. Our client, a high-profile e-commerce company, depends heavily on its web application to operate and communicate with its customers.

With the continuous changes and evolving threats in cyberspace, protecting these types of web applications from potential breaches and attacks is a critical component of maintaining trust and ensuring business continuity. Given their critical role in handling sensitive user data and financial transactions, web applications are prime targets for cybercriminals..They process and store large volumes of sensitive data and a single vulnerability can lead to severe consequences – such as loss of data, money, brand image, and job positions. Our client was aware of these threats, which is why they took proactive actions to detect and resolve any security flaws in their online application, by performing rigorous penetration testing.

Penetration testing, also known as ethical hacking, is widely recognised as a vital cybersecurity processes. This is how it works: first, we emulate a cyberattack on the application to reveal vulnerabilities before a hacker does. This type of testing is used to reveal weaknesses, but also to assess the risks associated with them and, most critically, helps to suggest improvements in security mechanisms.

The client approached our cybersecurity team with a clear mandate: perform a complete penetration test of their web application, uncover any potential security problems, and recommend effective mitigation techniques. In this case study, we outline how this penetration testing was carried out, together with its findings and results, revealing how an active cybersecurity strategy can significantly enhance the resiliency of web applications against emerging threats.

Background

Industry: E-commerce
Employees: 2,000
Objective: Assess the security of the client's web application to identify vulnerabilities and recommend improvements.

Scope and Objectives

The primary objectives of the application penetration testing were:

1. Identify Vulnerabilities: Detect potential security weaknesses in the web application.
2. Assess Risk Impact: Evaluate the potential impact of identified vulnerabilities.
3. Test Security Controls: Evaluate the effectiveness of existing security measures.
4. Provide Recommendations: Suggest remediation measures to improve application security.

Methodology

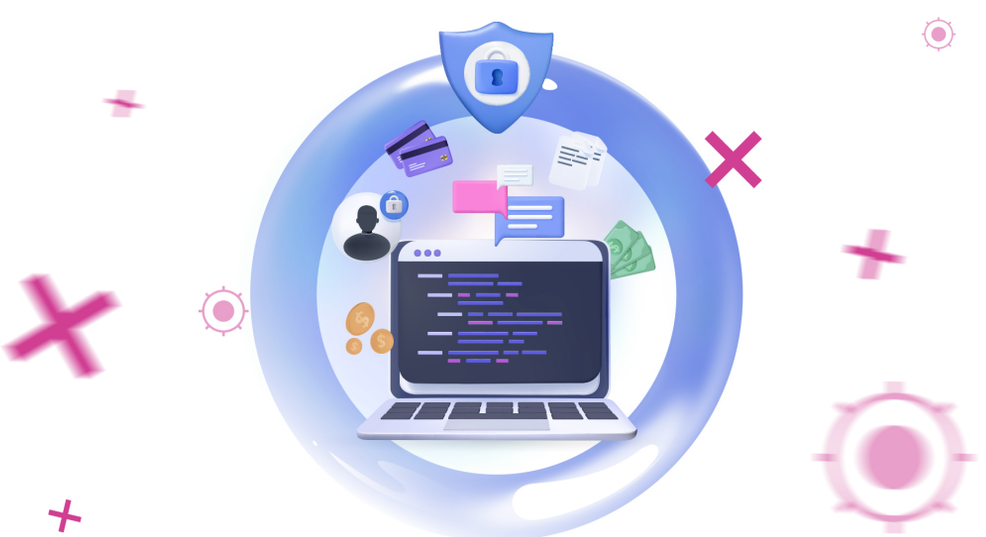
The application penetration testing followed a structured approach consisting of the following phases:

Information Gathering: Collecting information about the application, its architecture, and components.

Vulnerability Scanning: Using automated tools to identify common vulnerabilities.

Vulnerability Classification: Evaluate identified vulnerabilities and their potential impact on application security.

Reporting: Documenting findings and providing actionable recommendations.



Phase 1: Information Gathering

In the information gathering phase, the penetration testing team collected detailed information about the application. This included:

Application Architecture: Understanding the application's architecture, including front-end and back-end components.
Technology Stack: Identifying the technologies, programming tools and languages, and frameworks used in the application.
User Roles: Understanding the different user roles and their permissions within the application.

Phase 2: Vulnerability Scanning

During this step, automated technologies were utilized to do a thorough scan of the web application in order to uncover known vulnerabilities quickly and efficiently. To find vulnerabilities, the team employed industry standard automated scanning technologies such as OWASP ZAP, Burp Suite, and Nessus. These tools look for common vulnerabilities including SQL injection, cross-site scripting (XSS), and unsafe setups.

In addition to automated technologies, the team employed manual testing since it is essential to discover sophisticated and business logic vulnerabilities that can be missed by automatic tools. An analysis of the code of the application was performed, and dynamic testing is done to identify the logical flaws, insecure management of sessions, and authentication bypass. Login procedures and session management were evaluated. The application was scanned for flaws and input fuzzing techniques were used to inject unexpected data into the application to identify input handling vulnerabilities

Phase 3: Vulnerability Classification

In this phase, the penetration testing team evaluates discovered vulnerabilities, their severity, and impact they have on overall application security. The team assessed the real-world impact of the exploits by determining the extent of data compromise, unauthorized access, and potential for financial and reputational damage.

Phase 4: Reporting

The final phase involved documenting findings and providing a comprehensive report to client.

Detailed Vulnerability Findings: The report included detailed description of each identified vulnerability. It contained evidence such as screenshots. Logs and exploited payloads were also included.

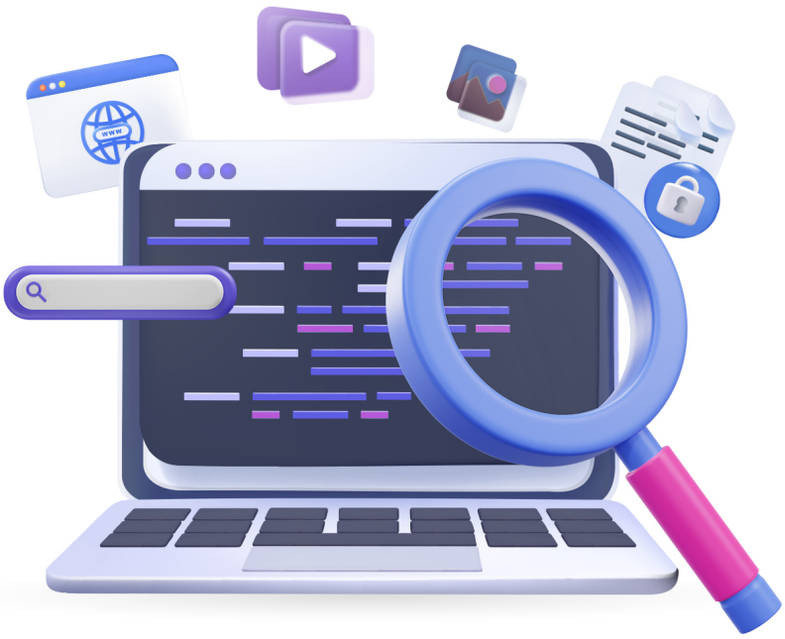
Severity and Impact Ratings: Each vulnerability received a rating based on its severity (low, medium, high, critical) and potential impact on application and organization.

Risk Mitigation Recommendations: Specific recommendations were provided for mitigating each identified vulnerability. These included code changes and configuration adjustments. Best practices for secure coding and development were also suggested.

Executive Summary: A high-level summary of findings and recommendations was included for senior management. It highlighted key risks and overall security posture of the application.

Actionable Roadmap: A prioritized roadmap for remediation was provided. It guided client on steps to take immediately and in longer term to enhance their web application security

Follow-up Support: Offering follow-up support and re-assessment plan to ensure that recommended changes were effectively implemented. It is important that new vulnerabilities were not introduced during the remediation process.



Key Findings

There were multiple vulnerabilities found, 8 of them critical, which required immediate actions to remediate them on customer side:

1. SQL Injection: Multiple input fields were vulnerable to SQL injection, allowing attackers to execute arbitrary SQL queries.

2. Outdated version of Open SSL, older versions contain known vulnerabilities, which have been fixed/patched in newer versions.

3. Outdated version of MySQL, application relies on older version of database, containing known vulnerabilities, which have been fixed/patched in newer versions.

4. Reliance on obfuscation or encryption of security relevant inputs without integrity checking, allowing attackers to modify input.

5. Plain text transmission of sensitive information, passwords, user credentials, or other private information is transmitted unencrypted over a network.

6. SSL/TLS: Report vulnerable cipher suites for HTTPS, this vulnerability allows attackers to exploit outdated or weak encryption algorithms to compromise the security of communications.

7. Directory Traversal, a security flaw that allows attackers to access and browse files outside of the intended path within web applications or systems

8. Insecure transport security protocol supported (TLS 1.0), application uses older version of TLS protocol, which is exposed to serious security flaws and vulnerabilities.

Key Recommendations

1. Input Validation: Implement robust input validation and parameterized queries to prevent SQL injection attacks and directory traversal.

2. Update OpenSSL and MySQL versions

3. Sanitize User Input: Use proper encoding, sanitization, and integrity verification techniques to secure the data.

4. Implement encryption mechanisms and secure protocols for data transmission.

5. Update SSL/TLS configuration to use strong cryptographic algorithms. Update regularly.

6. Access Controls: Enforce strict access controls to prevent unauthorized access to sensitive resources.

Conclusion

Penetration testing of the web application for our client resulted in the identification of several highly critical vulnerabilities that attackers could easily exploit. By addressing these issues and implementing recommended measures, the client was able to significantly enhance the security of their web application. This proactive approach not only helps in mitigating risks, but also ensures the protection of customer data and the integrity of their e-commerce platform.

Take the Next Step in Securing Your Applications

If you're looking to fortify your applications against emerging threats while optimizing performance, explore how Comtrade 360's Application Modernization and Security Services can help. Our experts are ready to guide you through upgrading your digital infrastructure and implementing robust security measures to meet the highest industry standards.