Penetration Test Report for 165.232.190.225

Tester’s Name :

Mohammed althaf

Amjad Ameen

Vaishnav p

OCTOBER 3, 2024

Table of Contents

Introduction

This report documents the results of a penetration test conducted on the web application hosted at IP address 165.232.190.225. The purpose of this assessment is to identify and evaluate potential security vulnerabilities that could be exploited by malicious actors. The penetration test was authorized and performed with the intent to strengthen the security posture of the application by identifying weaknesses that could lead to data breaches, service disruptions, or unauthorized access.

The testing process followed a structured methodology, including reconnaissance, vulnerability identification, exploitation, and reporting, ensuring comprehensive coverage of the application's attack surface. The test aimed to emulate real-world attacks, using both automated tools and manual techniques, while maintaining the confidentiality and integrity of the system.

1. Methodology

1.1Scope : This scop of this phase is exploit has many vulnerability as possible .this testing manly focus on exploiting web application vulnerabilities mean while the other vulnerabilities with in the system is meant to be out of scope for this phase

1.2Testing Approach : we approached the target with basic scanning and exploitations .both manual and automated tools have being used for the testing this target as far of as concerning about the primary information regarding the target is vitally low

2. Testing Phases

2.1 Information gathering : we used network maper and zenmap as tools to find the open ports and services running on the target, and found a HTTP port running on port 56686 which make a potential website.

2.2 Vulnerability Identification : We used burpsuite scanner to scan for any vulnerabilities present in the target website

3. Findings

Vulnerability Title :

3.1 cross site scripting (XSS)

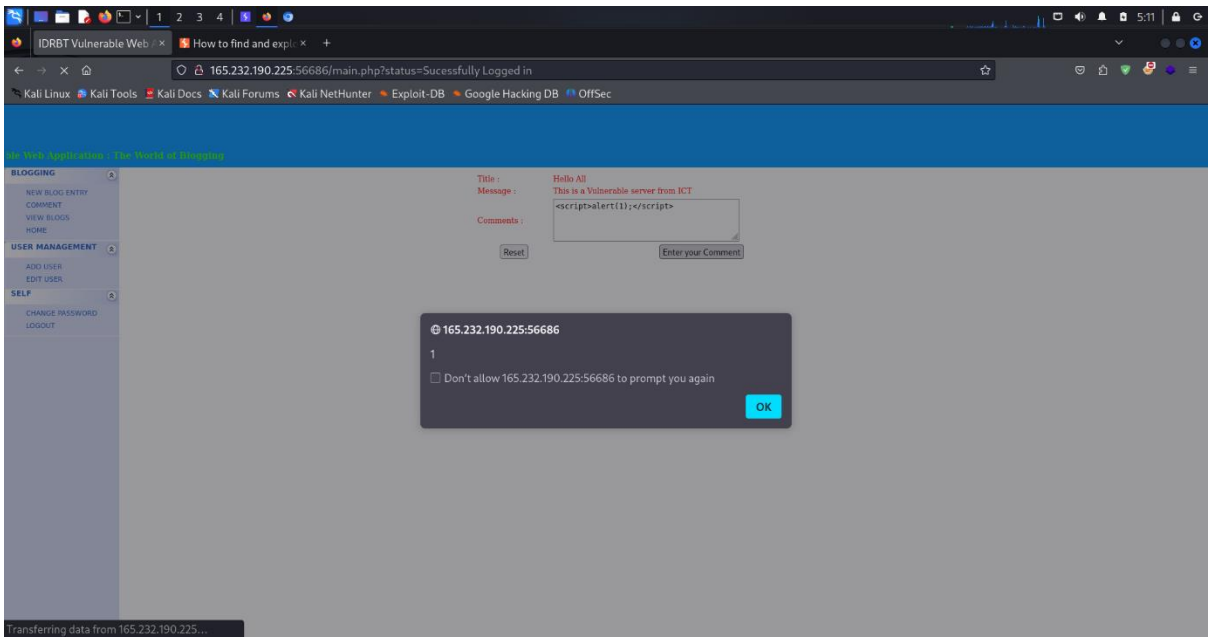
cross site scripting through the comment box

Severity : High (malicious payloads can be run on the server)

Description : xss is a vulnerability where attackers inject malicious script in too the trusted websites . These scripts run in the victim's browser, allowing attackers to steal data, hijack sessions, or manipulate the website's behavior. It can affect user privacy, security, and trust in the site.

Technical impact : Exploiting an XSS vulnerability can lead to severe consequences such as session hijacking, where attackers take over user accounts, data theft (including sensitive information like passwords), defacement of websites, and spreading malware. It can also damage the reputation of the website and lead to legal and financial penalties.

Proof of Concept (PoC) :



Screenshot 1

Remediation: To mitigate XSS:

1. Validate and sanitize input.

2. Encode outputs to prevent script execution.

3. Implement CSP headers.

4.Use security frameworks with built-in protections.

5. Use HTTPOnly and Secure cookies to protect sessions.

Vulnerability title :

3.2 file upload vulnerability :

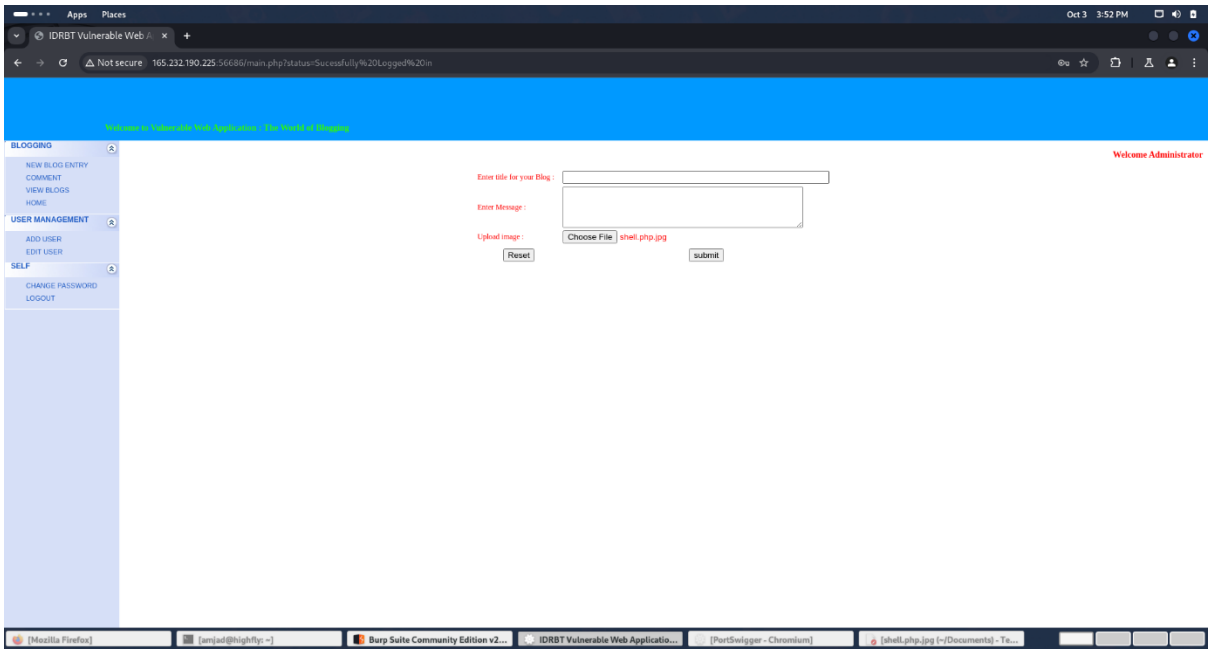
file uploading via ‘new blog’ tab

Severity : High (can allow attackers to gain control of the server

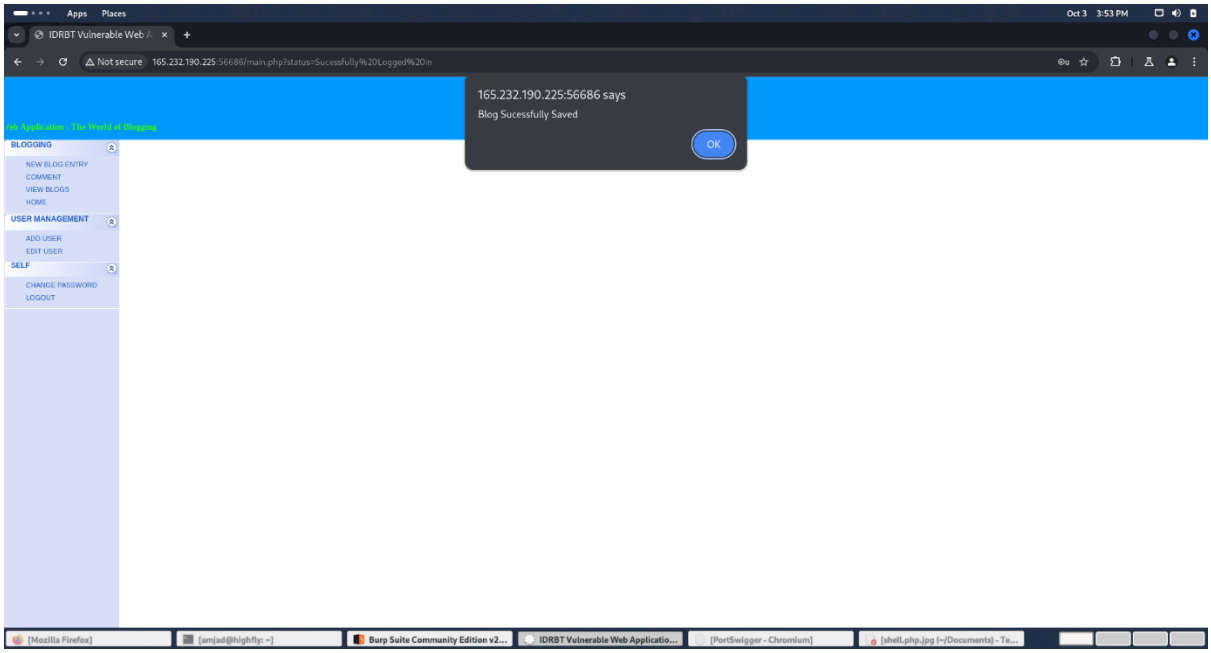
Description : A file upload vulnerability occurs when a web application allows users to upload files without proper validation, enabling attackers to upload malicious files like scripts or executables. This can lead to remote code execution, data breaches, or complete server compromise.

Technical impact : The technical impact of file upload vulnerabilities includes **remote code execution,server compromise, resource exhaustion, malware distribution, and bypassing authentication or gaining elevated privileges.**

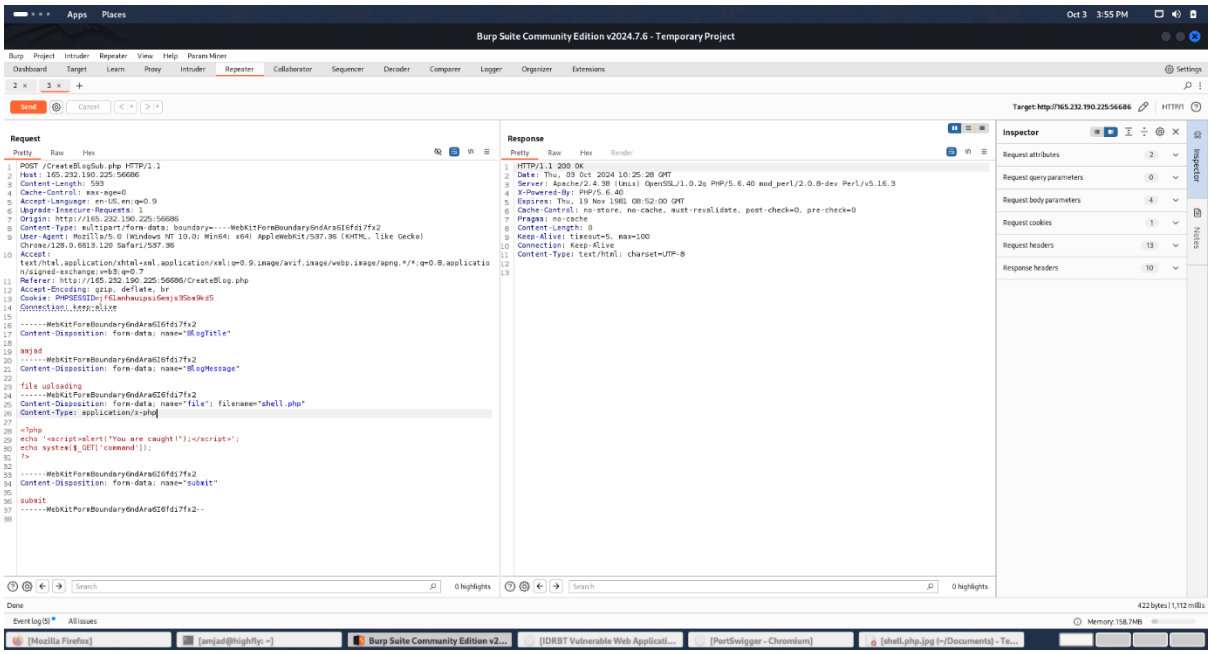
Proof of Concept (PoC) :



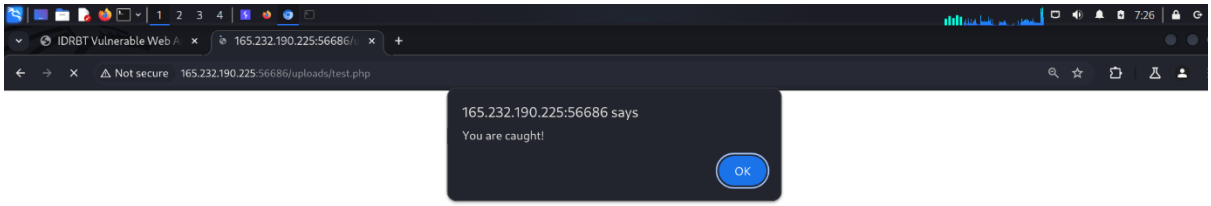
Screenshot 2



Screenshot 3



Screenshot 4



Screenshot 5

Remadiation : To mitigate file upload vulnerability

To mitigate file upload vulnerabilities:

1. Restrict file types.
2. Validate file content.
3. Scan for malware.
4. Rename files.
5. Store outside web directories.
6. Limit file permissions.

Vulnerability title :

3.3 common password vulnerability :

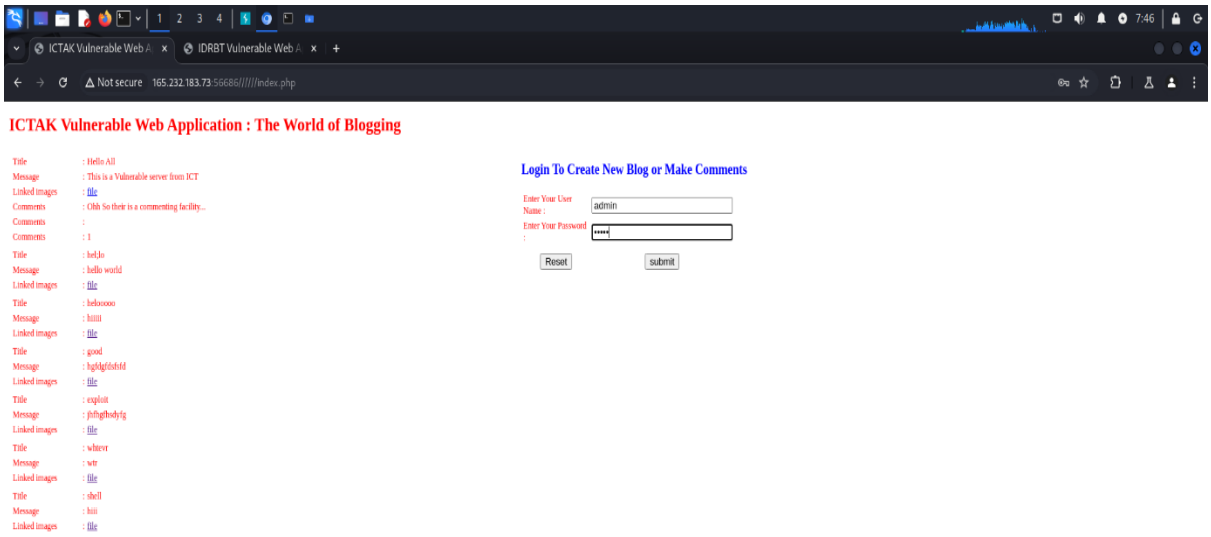
weak or common password on the login page

Severity : High (weak or reused passwords make it easy for the attackers to perform brute force and credential stealings)

Description : A common password vulnerability occurs when users choose weak, easily guessable, or reused passwords across multiple sites. This exposes systems to brute force attacks, credential stuffing, or unauthorized access, compromising account security and sensitive data.

Technical impact : The technical impact of common password vulnerabilities includes **account takeover, data theft, privilege escalation, system compromise, and successful brute force attacks.**

Proof of Concept (PoC) :



Screenshot 6

Remadiation : To mitigate common password vulnerabilities:

1. Enforce strong password policies.
2. Enable multi-factor authentication (MFA).
3. Use password hashing and salting.
4. Implement account lockout mechanisms* after failed attempts.
5. Encourage password managers for unique, strong passwords

Risk Analysis

1.XSS (cross site scripting) : Likelihood: 8.0 (High)(Many web applications are prone to XSS attacks due to inadequate input validation.)

Impact: 6.5 (Medium)

(Can lead to session hijacking and data theft but varies based on context.)

Risk Rating: 7.3 (High)

(CVSS Base Score calculated as [0.6 * Impact + 0.4 * Likelihood] = 0.6 * 6.5 + 0.4 * 8.0)

2. File Upload Vulnerability

Likelihood: 7.0 (Medium to High)

(Common in applications that allow user uploads, though it can be mitigated with proper controls.)

Impact: 7.5 (High)

(Can result in remote code execution, server compromise, and significant data loss.)

Risk Rating: 7.2 (High)

(CVSS Base Score = [0.6 * Impact + 0.4 * Likelihood] = 0.6 * 7.5 + 0.4 * 7.0)

3. Common Password Vulnerability

Likelihood: 9.0 (High)

(Weak and reused passwords are prevalent among users.)

Impact: 7.0 (High)

(Leads to account takeover, data breaches, and potential full system compromise.)

Risk Rating: 8.0 (High)

(CVSS Base Score = [0.6 * Impact + 0.4 * Likelihood] = 0.6 * 7.0 + 0.4 * 9.0)

Recommendations

1.XSS (Cross-Site Scripting)

Input Validation: Enforce strict validation of user inputs.

Output Encoding: Encode user-generated content before display.

Content Security Policy (CSP): Deploy CSP to restrict script sources.

Security Libraries: Use frameworks with built-in XSS protection.

2. File Upload Vulnerability

File Type Restrictions: Allow only specific file types and validate MIME types.

File Scanning: Integrate antivirus tools to scan uploads.

Limit File Size: Set maximum file size limits.

Secure File Storage: Store files outside web-accessible directories.

Rename Files: Change filenames to prevent script execution.

3. Common Password Vulnerability

Strong Password Policies: Enforce complex password requirements.

Multi-Factor Authentication (MFA): Implement MFA for additional security.

Password Hashing: Use strong hashing algorithms for storing passwords.

Account Lockout: Temporarily lock accounts after multiple failed logins

Conclusion

The penetration testing of the web application hosted at IP address 165.232.190.225 revealed several critical vulnerabilities, including Cross-Site Scripting (XSS), file upload vulnerabilities, and weak password practices. Each of these vulnerabilities poses a significant risk to the application's integrity, user data, and overall system security.

The XSS vulnerability, which allows attackers to inject malicious scripts, can lead to severe consequences like session hijacking and data theft. The file upload vulnerability discovered in the "new blog" feature could enable an attacker to upload malicious files and potentially take control of the server. Additionally, the use of weak or common passwords makes the system susceptible to brute force attacks and unauthorized access.

Addressing these vulnerabilities is crucial to ensuring the security of the web application and protecting it from potential exploitation. We recommend implementing the remediation measures outlined in this report, such as improving input validation, enforcing strong password policies, and properly configuring file upload restrictions. By doing so, the application's security posture will be significantly improved, reducing the likelihood of successful attacks.

Continuous monitoring, security audits, and applying best practices in web application security are essential to maintaining the security of the system. It is also advisable to perform regular penetration testing to identify new or overlooked vulnerabilities, ensuring the application remains resilient against evolving threats.