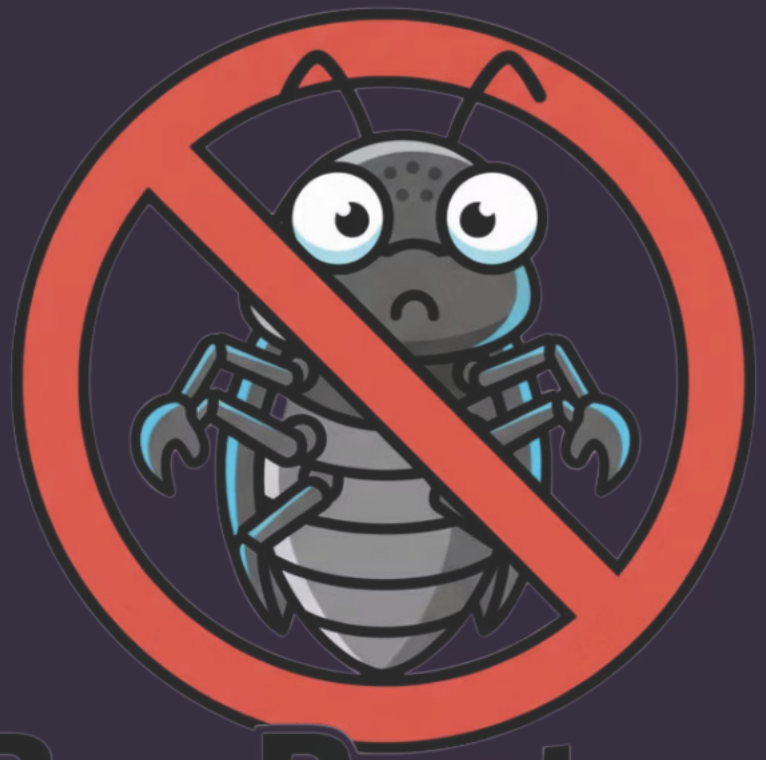


BugBusters

Security Assessment Results:

[redacted]

Executive Briefing - November 2025

Presented by: BugBusters Security Team

Security Assessment Scope

Key Areas Assessed

- Customer login and authentication systems
- Password vault security
- Administrative controls
- Data protection measures

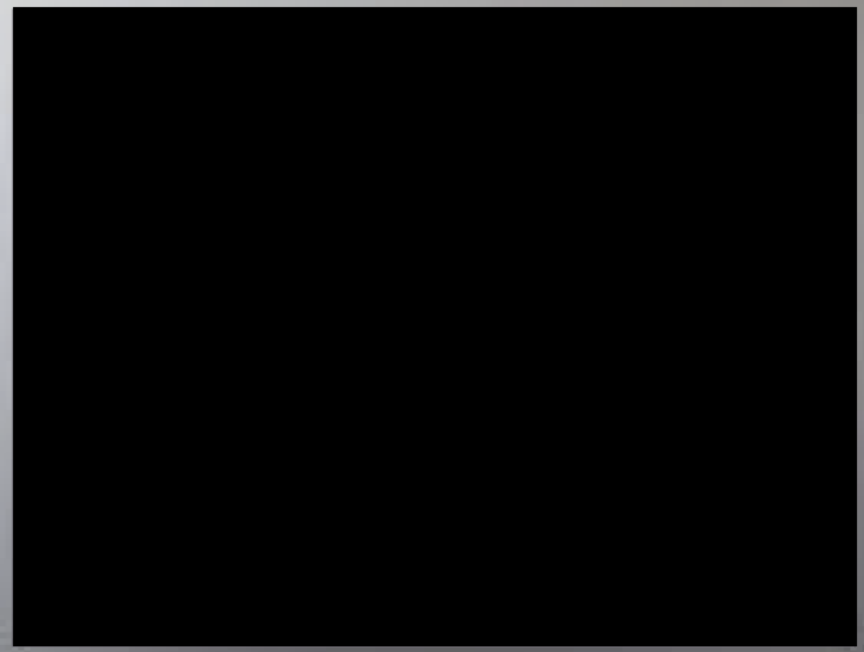
Assessment Methodology

Testing period: October - November 2025

Approach: Info gathering and Simulated real attacker methods

Testing Environment

Staging environment used (no disruption to production systems)



The OWASP Testing Framework

Security Assessment: Approach & Tools

Our Approach

- Manual security testing
- Automated vulnerability scanning
- Real-world attack simulation

Tools & Techniques

- Industry-standard security tools
- Testing scripts
- Ethical hacking methods and frameworks

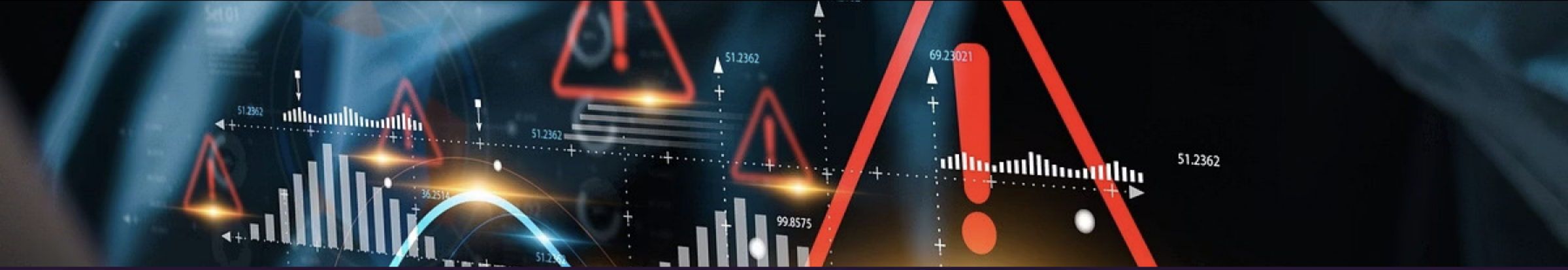
Executive Summary

Current State: Through our work we have found what we believe are weaknesses in the platform's security architecture that could collectively increase the risk of account compromise and unauthorized access.

1	1	2	1
High Severity	**Medium Severity**	**Medium Priority**	**Informational**
Immediate action required	Address within 30 days	Address within 90 days	Further review suggested

Key Findings

- Critical vulnerabilities in file upload and session management
- Missing multi-factor authentication
- Outdated encryption methods
- Certificate management issues



Risk Overview Dashboard

Key security concerns identified during assessment

High Priority Issues

File upload security gaps

Session security vulnerabilities

Medium Priority Issues

Missing multi-factor authentication

Weak authentication methods

Additional Concerns

Outdated encryption technology

Certificate expiration approaching

Finding #1 - Cryptographic Weaknesses

The Issue

MD5 hashing in authentication tokens

Plaintext emails in tokens

Predictable token structure

Translation:

Outdated 1990s encryption that's easily cracked.

Risk

Attackers forge authentication tokens

Account impersonation (including admins)

Token collision attacks

Impact

Complete authentication bypass

Customer data exposure

Compliance violations

Finding #2 - No Two-Factor Authentication

How We Found It:

During authentication testing, we observed no secondary verification prompts after successful password entry. API endpoint analysis confirmed absence of MFA enforcement mechanisms.

The Issue

Only username/password required (single lock on the door)

Business Translation

Like securing a bank with just one key

Risk Scenario

- Password theft = complete access
- No second verification required
- Phishing attacks succeed easily

Finding #3 - Malicious File Upload Vulnerability

HIGH Severity

This critical flaw allows attackers to upload dangerous files, potentially leading to widespread system compromise.

The Security Issue

Our system's profile image upload feature accepts **dangerous file types** (such as SVG files embedded with executable JavaScript code). This is like allowing a malicious program to be disguised as a harmless picture.

The Business Risk

This vulnerability, often referred to as Stored Cross-Site Scripting (XSS), means that uploaded files are not properly checked. An attacker could embed **malicious code** within a file, which then gets stored on our servers and can be executed when other users view it. This is similar to a Trojan horse, where a seemingly benign file can unleash significant damage.

How We Found It

During testing, we were able to change a harmless image file (PNG) into a malicious one (SVG) during the upload process. Crucially, our server accepted this manipulated file without flagging it as a threat.

Potential Impact

If exploited, this vulnerability could lead to severe consequences, including: **account takeover** (attackers gaining full control of user accounts), **credential theft** (stealing login information), and **malware distribution** to our users. This directly threatens user trust, data privacy, and could result in significant reputational and financial damage.

The Required Fix

Implement **robust server-side file validation**. This means the server must thoroughly inspect all uploaded files to ensure they are safe and of the expected type, rejecting any that pose a risk, regardless of initial file extension or user manipulation.

Finding #5 - Missing Session Cookie Protection

Severity: MEDIUM

This finding highlights a security oversight related to how our website manages user sessions, specifically concerning "session cookies." A session cookie acts like a temporary digital ID card, allowing our system to recognize a logged-in user without them needing to re-enter credentials for every action. If these digital ID cards are not properly protected, they can be easily stolen, leading to unauthorized access to user accounts.

Issue: Vulnerable Session Cookies

Our website's session cookies, which identify logged-in users, are missing the **HttpOnly flag**. This flag is a critical security measure that prevents client-side scripts (like JavaScript) from accessing these cookies. Without it, these digital ID cards are exposed.

Technical Detail: Exposed Tokens

Specifically, the `document.cookie` object exposes sensitive AWSALBAPP tokens. This means that if malicious code were to execute on our site (e.g., through an XSS attack), it could easily read and steal these tokens. Think of it like leaving your car keys on the dashboard with the windows down.

Business Risk: Session Hijacking & Impersonation

If an attacker successfully injects malicious code onto our platform (a Cross-Site Scripting or XSS vulnerability), they can steal these unprotected session cookies. This allows them to **impersonate legitimate users**, gaining full access to their accounts without needing passwords. This could lead to data breaches, unauthorized transactions, or reputational damage due to compromised user trust.

Evidence: Verified Vulnerability

During testing, we successfully demonstrated this vulnerability by using a standard web browser console to access and retrieve all session cookies. This confirms that an attacker with even basic technical knowledge could exploit this flaw to steal user sessions.

Required Fix: Implement HttpOnly Flag

The solution is to add the **HttpOnly flag** to all session cookies. This simple, yet powerful, change will make it significantly harder for malicious scripts to access these critical tokens, acting as a robust shield against session hijacking. This is a straightforward technical adjustment that can be implemented quickly, estimated to take only about **2 hours**, providing a high return on security investment.

Security Test - Privilege Escalation (Unsuccessful)

What We Tested

Tried to use admin cookies on regular user account

Result

System properly blocked with 403 Forbidden

Good News

Authorization controls are working

Note

Shows session management is properly validated server-side

No Action Needed

This security control is functioning correctly

Additional Security Observations

Beyond our successful privilege escalation test, our security audit identified several critical vulnerabilities that require immediate attention. These observations highlight significant risks to our operational continuity, data security, and overall business integrity, which could lead to severe financial and reputational consequences if not addressed promptly.

SSL Certificate Expiring

- Expires December 4 (1 day after testing)
- Would cause complete service outage
- Need auto-renewal process

The expiration of our SSL certificate will immediately halt all online operations, making our website and services inaccessible to customers. This translates to direct revenue loss, significant damage to our brand reputation, and a severe erosion of customer trust. Implementing an automated renewal process is crucial to prevent such critical outages in the future.

500+ Directories Exposed

- Found via automated scanning
- Reveals backend structure
- JavaScript bundles show all application logic

Exposing over 500 directories reveals the internal workings and structure of our application to potential attackers. This information can be used to identify weaker points, making our systems more vulnerable to targeted attacks, and potentially exposing proprietary application logic which could be a competitive disadvantage.

Configuration Files Accessible

- /wp/appSettings.js public
- Potential API keys exposed
- No authentication required

The public accessibility of critical configuration files, such as 'appSettings.js', is an extremely severe security flaw. These files can contain sensitive information like API keys – essentially digital master keys that grant access to our internal systems or third-party services. Unsecured, this could lead to unauthorized data access, financial fraud, and a complete compromise of our digital infrastructure.

Customer Impact Summary

What This Means for Your Customers

Data Exposure

Their stored passwords could be exposed

Business Risk

Their business credentials at risk

Trust Compromised

Their trust in [redacted] compromised

Financial Impact Estimate

Breach Costs

Significant financial exposure from incident response, forensics, and remediation

Customer Loss

Substantial churn expected as customers move to more secure alternatives

Legal/Regulatory

Major fines possible from compliance violations and data protection failures

Reputational Impact: [redacted] suffers major breach" headlines, customer exodus to competitors, years to rebuild trust

Summary & Conclusion

This assessment provided a comprehensive review of your system's security posture, identifying several areas for improvement as well as key strengths.

Key Findings:

- Identified cryptographic weaknesses
- Lack of Multi-Factor Authentication (MFA)
- Discovered a file upload vulnerability
- Issues with session cookie handling
- SSL certificate nearing expiration
- Presence of exposed directories

Thank you for your time and attention during this assessment. We appreciate your commitment to enhancing security.