AI-Augmented Web Application Security Assessment

Target: https://www.indusrangers.com/

Report Date: 2025-06-25

Scan Date: 2025-06-25

Report Version: 1.0

Generated by: AutoPent.AI Security Assistant

Classification: CONFIDENTIAL

Distribution: Internal Use Only

Executive Summary

This report presents the findings of an automated web application security assessment conducted using AI-augmented analysis. The assessment identified **6** potential security vulnerabilities across the target application.

Risk Distribution:
• High Risk: 1 vulnerabilities
• Medium Risk: 2 vulnerabilities
• Low Risk: 3 vulnerabilities
• Informational: 0 findings

Immediate Actions Required:
1. Address all high-risk vulnerabilities within 24-48 hours
2. Implement input validation and output encoding
3. Review and update security configurations
4. Conduct regular security assessments

Vulnerability Summary

Vulnerability Name	Risk Level	Confidence	URL	CWE ID
Missing Security Header: Content-Securit	Medium	High	https://www.indusrangers.com/	N/A
Missing Security Header: X-Frame-Options	Medium	High	https://www.indusrangers.com/	N/A
Missing Security Header: Referrer-Policy	Low	High	https://www.indusrangers.com/	N/A
Information Disclosure - Server Header	Low	High	https://www.indusrangers.com/	N/A
SSL Certificate Name Mismatch	High	High	https://www.indusrangers.com/	N/A
Inline JavaScript Detected	Low	High	https://www.indusrangers.com/	N/A

Detailed Findings

SSL Certificate Name Mismatch [High Risk]

Property	Value
Risk Level	High
Confidence	High
CWE ID	Not specified
Affected URL	https://www.indusrangers.com/
Parameter	

Description:

This vulnerability, SSL Certificate Name Mismatch, occurs when the common name (CN) or subject alternative name (SAN) listed on an SSL certificate does not match the domain name that is in the address bar of the browser. This could be due to a configuration error, an expired certificate, or a malicious attempt to deceive users.

Impact:

This vulnerability is dangerous because it undermines the trust users have in a secure connection. It could lead to man-in-the-middle attacks where an attacker intercepts communication between the user and the server. This could result in unauthorized access to sensitive data, such as login credentials, credit card information, or personal data, leading to potential data breaches and reputational damage.

Remediation:

To fix this vulnerability, the SSL certificate needs to be correctly configured to match the domain name. This can be done by reissuing the certificate with the correct common name (CN) or subject alternative name (SAN). Ensure that the certificate is obtained from a trusted Certificate Authority (CA) and is properly installed and configured on the server. Regularly check the validity of the SSL certificate and renew it before it expires.

Prevention:

1. Regularly audit and monitor SSL certificates for any discrepancies or mismatches. 2. Implement automated SSL certificate management tools to ensure certificates are always valid and correctly configured. 3. Use a trusted Certificate Authority (CA) for issuing SSL certificates. 4. Regularly educate and train IT staff on the importance of correct SSL certificate configuration and the potential risks of misconfiguration.

Missing Security Header: Content-Security-Policy [Medium Risk]

Property	Value
Risk Level	Medium
Confidence	High
CWE ID	Not specified
Affected URL	https://www.indusrangers.com/
Parameter	

Description:

What is this vulnerability? This vulnerability refers to the absence of the Content-Security-Policy (CSP) header in the web application. The CSP is a security feature that helps prevent cross-site scripting (XSS), clickjacking and other code injection attacks. It works by specifying the domains that the browser should consider as valid sources of executable scripts. **Why is it dangerous?** The absence of a CSP header can expose the web application to various security risks such as XSS an...

Impact:

The absence of a CSP header can expose the web application to various security risks such as XSS and data injection attacks. These attacks can lead to data breaches, unauthorized changes to data, or even complete takeover of the web application. Attackers can exploit this vulnerability to inject malicious scripts into web pages viewed by users, potentially leading to theft of sensitive data or unauthorized actions on behalf of the user.

Remediation:

To fix this vulnerability, the web application should include a properly configured CSP header in its HTTP responses. This can be done by adding the following line to the HTTP response headers: `Content-Security-Policy: default-src 'self'; script-src 'self' example.com;` This example allows scripts to be loaded from the current domain ('self') and example.com. The actual policy should be tailored to fit the specific needs of the web application.

Prevention:

1. Regularly review and update the CSP policy to ensure it only allows scripts from trusted sources.
2. Conduct regular security audits of the web application to identify and fix potential vulnerabilities.
3. Educate developers about the importance of security headers and how to properly implement them. 4. Use automated tools to check for missing security headers and other common vulnerabilities.

Missing Security Header: X-Frame-Options [Medium Risk]

Property	Value
Risk Level	Medium
Confidence	High
CWE ID	Not specified
Affected URL	https://www.indusrangers.com/
Parameter	

Description:

What is this vulnerability? This vulnerability refers to the absence of the X-Frame-Options header in the HTTP response. The X-Frame-Options header is a security measure that prevents the website from being displayed in a frame or iframe. Without this header, the website is susceptible to 'Clickjacking' attacks, where a malicious actor can trick a user into clicking on something different from what the user perceives, leading to unauthorized commands or revealing confidential information. *...

Impact:

The danger of this vulnerability lies in the potential for 'Clickjacking' attacks. These attacks can lead to unauthorized actions being performed on behalf of the user without their knowledge. This could range from making purchases, changing password, or even compromising sensitive data. An attacker exploiting this vulnerability could potentially gain control over a user's interactions with the site, leading to significant business impact including financial loss, reputational damage, and potential legal implications.

Remediation:

To fix this vulnerability, the X-Frame-Options header should be included in the HTTP response. This can be done by adding the following line to the .htaccess file or the server configuration file: `Header always append X-Frame-Options SAMEORIGIN` This will ensure that the page can only be displayed in a frame on the same origin as the page itself. Alternatively, if you never want the page to be framed, you can use: `Header always append X-Frame-Options DENY`

Prevention:

1. Regularly conduct security audits and vulnerability assessments to identify and fix potential security issues. 2. Implement a Content Security Policy (CSP) to add an extra layer of security against 'Clickjacking' and other cross-site scripting attacks. 3. Educate developers about secure coding practices, including the importance of setting HTTP response headers correctly. 4. Use security tools and plugins that automatically check for missing security headers and other common vulnerabilities.

Missing Security Header: Referrer-Policy [Low Risk]

Property	Value
Risk Level	Low
Confidence	High
CWE ID	Not specified
Affected URL	https://www.indusrangers.com/
Parameter	

Description:

The Missing Security Header: Referrer-Policy vulnerability is a security issue where the web application does not include the Referrer-Policy HTTP header in its response. This header controls the amount of referrer information included with requests made from a particular request client. When this header is missing, browsers may send the full URL of the page the user is currently on to other sites when the user follows a link, potentially leaking sensitive information.

Impact:

While the risk level for this vulnerability is low, it can still pose significant threats. An attacker could exploit this vulnerability to gather sensitive information about the user's browsing habits or data from the URL. This could potentially lead to privacy breaches, data leakage, or other forms of unauthorized information access. The business impact could include loss of customer trust, regulatory penalties for privacy violations, and potential legal implications.

Remediation:

To fix this vulnerability, the Referrer-Policy header should be included in the HTTP response. This can be done by adding the following line to the .htaccess file or server configuration file: `Header set Referrer-Policy "no-referrer"` This line sets the Referrer-Policy to "no-referrer", which means that no referrer information will be sent along with requests.

Prevention:

1. Regularly conduct security audits and vulnerability assessments to identify and fix potential vulnerabilities. 2. Implement a robust security policy that includes the use of all necessary security headers. 3. Educate developers about the importance of security headers and how to correctly implement them. 4. Use security tools and plugins that automatically check for missing headers and other common security vulnerabilities.

Information Disclosure - Server Header [Low Risk]

Property	Value
Risk Level	Low
Confidence	High
CWE ID	Not specified
Affected URL	https://www.indusrangers.com/
Parameter	

Description:

This vulnerability, known as Information Disclosure - Server Header, occurs when a web server unintentionally reveals sensitive information about the underlying technology (such as the server type, version, or other software details) in its HTTP response headers. This information can be used by attackers to identify potential weak points and exploit known vulnerabilities specific to the disclosed technology.

Impact:

Although the risk level is low, the danger lies in the potential for targeted attacks. By knowing the specific technology and version used by the server, an attacker can tailor their attack strategies to exploit known vulnerabilities of that technology. This could lead to unauthorized access, data breaches, or even server takeover, which could have significant business impact including financial loss, reputation damage, and legal implications.

Remediation:

To fix this vulnerability, you need to configure your server to stop revealing sensitive information in its headers. For Apache servers, you can use the 'ServerTokens Prod' directive in your httpd.conf file. For Nginx servers, add 'server_tokens off;' in your nginx.conf file. In case of IIS servers, use URLScan tool's 'RemoveServerHeader' option. Always remember to restart your server after making these changes.

Prevention:

1. Regularly update and patch your server software to minimize the risk of known vulnerabilities. 2. Implement a strict Content Security Policy (CSP) to control what and how resources are loaded on your website. 3. Regularly conduct vulnerability assessments and penetration testing to identify and fix potential security issues. 4. Use security headers like HTTP Strict Transport Security (HSTS) and X-Content-Type-Options to further secure your server communications.

Inline JavaScript Detected [Low Risk]

Property	Value
Risk Level	Low
Confidence	High
CWE ID	Not specified
Affected URL	https://www.indusrangers.com/
Parameter	

Description:

This vulnerability, Inline JavaScript Detected, refers to the practice of embedding JavaScript code directly within HTML files. It occurs when developers place JavaScript code blocks within HTML

tags, instead of linking to external JavaScript files.

Impact:

While the risk level is low, inline JavaScript can still pose a threat. It can lead to Cross-Site Scripting (XSS) attacks if user input is not properly sanitized. An attacker could inject malicious scripts into the web application, potentially leading to data theft, session hijacking, or defacement of the website.

Remediation:

To fix this vulnerability, follow these steps: 1. Remove all inline JavaScript code from HTML files. 2. Place the JavaScript code in separate .js files. 3. Link to these .js files from your HTML files using the script src attribute. For example, ``.

Prevention:

1. Avoid using inline JavaScript whenever possible. Always use external .js files. 2. Regularly review and update your code. Remove any unnecessary or outdated JavaScript. 3. Implement a Content Security Policy (CSP) to restrict the execution of scripts.

Recommendations

General Security Recommendations:

1. Implement a comprehensive input validation framework
2. Deploy proper output encoding mechanisms
3. Enable security headers (CSP, HSTS, X-Frame-Options)
4. Conduct regular security code reviews
5. Implement automated security testing in CI/CD pipeline
6. Establish an incident response plan
7. Provide security training for development team

Recommended Remediation Timeline:
- **Critical/High Risk:** Immediate action required (24-48 hours)
- **Medium Risk:** Address within 1-2 weeks
- **Low Risk:** Include in next maintenance cycle
- **Informational:** Monitor and review during next assessment