

OSCAR SIX RADAR

Security Assessment Report

Automated Vulnerability Scanning & Analysis

krazyuncle.com

MEDIUM RISK

Remediation recommended

Assessment Date: February 12, 2026 at 14:40 UTC

CONFIDENTIAL - For Authorized Recipients Only

Report ID: crazyuncle-20260212

Table of Contents

1. Summary Dashboard

2. Executive Summary

3. Technical Findings

4. Remediation Recommendations

5. Risk Assessment

6. Reconnaissance Summary

7. Attack Analysis

8. Appendix

1. Summary Dashboard



Detected Technologies

- cloudflare
- google cloud
- google cloud load balancing
- onsen ui
- osano:3.1.0
- cdnjs
- jquery:2.2.4
- google frontend

Metric	Value
Target	krazyuncle.com
Total Findings	7
Tests Executed	8
Assessment Date	February 12, 2026
Overall Risk Level	MEDIUM

2. Executive Summary

Executive Summary

An automated vulnerability scan was conducted on the web application hosted at krazyuncle.com to assess its security posture and identify potential vulnerabilities. The assessment involved eight comprehensive security tests designed to evaluate the application's resilience against common attack vectors and security weaknesses.

The testing revealed a moderate security posture with no critical or high-severity vulnerabilities identified. However, four medium-risk findings and three low-risk findings were discovered, along with one informational item. While the absence of critical vulnerabilities indicates that the application has basic security controls in place, the medium-risk findings represent areas where security improvements are needed to reduce the organization's exposure to potential threats. These vulnerabilities, if exploited, could potentially lead to unauthorized access to sensitive information, data manipulation, or service disruption.

The most significant concerns stem from the four medium-severity findings, which typically involve issues such as inadequate input validation, improper access controls, or insufficient security configurations. Although these vulnerabilities may not pose immediate catastrophic risk, they could serve as stepping stones for attackers to escalate their access or combine multiple weaknesses to achieve more significant compromise of the system.

We recommend prioritizing the remediation of all medium-risk findings within the next 30-60 days, as these represent the most pressing security concerns identified during testing. Additionally, addressing the low-risk findings should be incorporated into routine security maintenance activities. Regular security assessments should be conducted quarterly to ensure continued protection against evolving threats, and a comprehensive security review of the application's architecture and coding practices would strengthen the overall security framework.

3. Technical Findings

MEDIUM SEVERITY FINDINGS

****Security Headers (2 findings)****

The application is missing critical security headers that provide defense-in-depth

protection against common web attacks. Two key headers were found to be absent from HTTP responses:

HTTP Strict Transport Security (HSTS) header is not implemented. Evidence shows that the 'strict-transport-security' header was not present in HTTP responses from the target application. Without HSTS, the application is vulnerable to protocol downgrade attacks where attackers can force users to connect over HTTP instead of HTTPS, enabling man-in-the-middle attacks and session hijacking. This is particularly dangerous on public networks where attackers can intercept unencrypted traffic.

Content Security Policy (CSP) header is also missing from all responses. The absence of the 'content-security-policy' header leaves the application vulnerable to Cross-Site Scripting (XSS) attacks. CSP acts as a last line of defense by controlling which resources the browser is allowed to load, preventing execution of malicious scripts even if they are injected into the application. Without CSP, any successful XSS payload can execute with full privileges in the user's browser context.

****Cross-Origin Resource Sharing (1 finding)****

The application implements an overly permissive CORS policy that poses significant security risks. Analysis of HTTP responses revealed that the server returns "Access-Control-Allow-Origin: *" header, allowing any website to make cross-origin requests to the application.

This wildcard CORS configuration enables malicious websites to make authenticated requests to the application on behalf of logged-in users. An attacker could host a malicious website that automatically sends requests to extract sensitive user data, perform unauthorized actions, or exfiltrate authentication tokens. While modern browsers prevent reading responses from cross-origin requests without proper CORS headers, this configuration still allows the requests to be sent, potentially triggering state-changing operations.

LOW SEVERITY FINDINGS

****Security Headers (2 findings)****

Two additional security headers are missing that provide supplementary protection

mechanisms:

The Referrer-Policy header is absent from HTTP responses. Without this header, browsers use default referrer behavior which may leak sensitive information in URLs when users navigate to external sites. While not immediately exploitable, this can result in information disclosure through web server logs and analytics platforms of third-party sites.

The Permissions-Policy header (formerly Feature-Policy) is not implemented. This header controls access to browser features and APIs such as geolocation, camera, and microphone. Without explicit permissions policies, the application cannot prevent potentially malicious embedded content from accessing these sensitive browser capabilities.

****SSL/TLS Configuration (1 finding)****

The server accepts connections using deprecated TLS 1.1 protocol. Testing confirmed that the server at krazyuncle.com:443 accepts TLS 1.1 connections alongside more secure protocol versions.

While TLS 1.1 is not as severely compromised as TLS 1.0, it is considered deprecated by current security standards and lacks modern cryptographic improvements found in TLS 1.2 and 1.3. Older TLS versions may be subject to downgrade attacks where adversaries force connections to use weaker protocols, and they lack forward secrecy and other modern security features. Major browsers and security frameworks now recommend disabling all TLS versions below 1.2.

4. Remediation Recommendations

Immediate Actions (Quick Wins)

****1. Missing HTTP Strict Transport Security (HSTS)****

EVIDENCE: Header 'strict-transport-security' was not present in the HTTP response from <https://krazyuncle.com>

REMEDATION:

- Add the HSTS header to all HTTPS responses: ``Strict-Transport-Security: max-age=31536000; includeSubDomains; preload``
- Configure this at the Google Cloud Load Balancer level or in your application server
- Consider submitting your domain to the HSTS preload list for maximum protection

LEARN MORE: Search "HSTS header Google Cloud Load Balancer implementation"

2. Missing Content Security Policy (CSP)

EVIDENCE: Header 'content-security-policy' was not present in the HTTP response from <https://krazyuncle.com>

REMEDIATION:

- Start with a basic CSP: ``Content-Security-Policy: default-src 'self'; script-src 'self' cdnjs.cloudflare.com; style-src 'self' 'unsafe-inline``
- Given your use of jQuery 2.2.4 from cdnjs, ensure script sources include trusted CDNs
- Test in report-only mode first: ``Content-Security-Policy-Report-Only``
- Gradually tighten the policy based on your application's needs

LEARN MORE: Search "Content Security Policy implementation jQuery cdnjs"

3. CORS Wildcard Origin Configuration

EVIDENCE: Access-Control-Allow-Origin: * detected at <https://krazyuncle.com>

REMEDIATION:

- Replace wildcard (*) with specific trusted domains
- Configure dynamic origin validation if multiple domains are needed
- Update your Google Cloud Run service or load balancer CORS configuration
- Example: ``Access-Control-Allow-Origin: https://yourdomain.com``

LEARN MORE: Search "CORS configuration Google Cloud Run security best practices"

****4. Outdated TLS Protocol Support****

EVIDENCE:

- Server accepts TLS 1.0 connections at krazyuncle.com:443
- Server accepts TLS 1.1 connections at krazyuncle.com:443

REMEDIATION:

- Configure Google Cloud Load Balancer to only accept TLS 1.2 and TLS 1.3
- Update the SSL policy in Google Cloud Console
- Set minimum TLS version: `gcloud compute ssl-policies create secure-ssl-policy --profile MODERN --min-tls-version 1.2``
- Apply the policy to your load balancer

LEARN MORE: Search "Google Cloud Load Balancer SSL policy TLS 1.2 minimum"

****5. Additional Security Headers****

EVIDENCE: Multiple security headers missing from <https://krazyuncle.com>:

- Header 'x-xss-protection' was not present
- Header 'referrer-policy' was not present
- Header 'permissions-policy' was not present

REMEDIATION:

- Add these headers to your responses:
 - ``X-XSS-Protection: 1; mode=block``
 - ``Referrer-Policy: strict-origin-when-cross-origin``
 - ``Permissions-Policy: geolocation=(), microphone=(), camera=()``
- Implement via Google Cloud Load Balancer custom headers or application-level middleware

LEARN MORE: Search "security headers implementation Google Cloud"

Long-term Security Improvements

****1. Modernize JavaScript Dependencies****

Your application uses jQuery 2.2.4, which is significantly outdated (released in

2016). Upgrade to the latest jQuery version or consider migrating to modern frameworks. Outdated libraries often contain known vulnerabilities and lack security patches.

****2. Implement Comprehensive Security Monitoring****

Leverage Google Cloud Security Command Center to monitor your Cloud Run applications. Set up alerts for configuration changes, unusual traffic patterns, and potential security incidents. Consider implementing Cloud Armor for additional DDoS and application-layer protection.

****3. Establish Content Security Policy Reporting****

Once CSP is implemented, set up a reporting endpoint to collect violation reports. This will help you identify potential XSS attempts and refine your CSP over time. Google Cloud Logging can be used to store and analyze these reports.

****4. Container Security Hardening****

Since you're using Google Cloud Run, ensure your container images are regularly scanned for vulnerabilities using Google Cloud's Container Analysis API. Implement least-privilege principles in your container configuration and consider using distroless base images.

****5. API Security Framework****

Given the CORS configuration findings, implement a comprehensive API security strategy including rate limiting, input validation, and proper authentication/authorization. Consider using Google Cloud Endpoints or API Gateway for centralized API management and security controls.

5. Risk Assessment

Overall Risk Assessment Report

1. Overall Risk Rating: ****Medium****

Based on the current vulnerability distribution, the organization's security posture presents a Medium risk level. While no Critical or High-severity vulnerabilities are present, the concentration of Medium-severity issues requires attention to prevent potential escalation and maintain acceptable security standards.

2. Key Risk Factors Identified

The primary risk factors stem from four Medium-severity vulnerabilities that collectively represent the most significant security concerns. These vulnerabilities typically involve issues such as outdated software components, misconfigurations, or moderate authentication weaknesses that could be exploited by determined attackers. The three Low-severity findings, while less concerning individually, contribute to the overall attack surface and may provide stepping stones for more sophisticated attacks.

3. Business Impact Assessment

The current risk profile suggests **moderate business impact potential**. Medium-severity vulnerabilities could lead to limited data exposure, service disruptions, or unauthorized access to non-critical systems. While immediate catastrophic business disruption is unlikely given the absence of Critical and High-severity issues, the cumulative effect of unaddressed Medium-severity vulnerabilities could result in:

- Moderate operational disruptions
- Potential compliance violations
- Limited data confidentiality breaches
- Reputational impact from security incidents
- Increased remediation costs if vulnerabilities are exploited or escalate

4. Recommended Risk Treatment Priorities

Priority 1: Address all four Medium-severity vulnerabilities within the next 30-60 days. These represent the primary risk drivers and should receive immediate resource allocation.

Priority 2: Resolve Low-severity findings within 90 days as part of routine security maintenance to reduce overall attack surface.

Priority 3: Implement continuous monitoring and regular vulnerability assessments to maintain this improved risk posture and prevent the introduction of higher-severity vulnerabilities.

The organization should maintain current security practices while focusing remediation efforts on the Medium-severity findings to achieve a Low overall risk rating.

6. Reconnaissance Summary

DISCOVERED SUBDOMAINS

```
-----  
  • {'subdomain': 'iap-gcip-hosted-ui-viewer-backend-service-7qznarhz7q-  
nw.a.run.app', 'source': 'subfinder', 'resolved': '34.143.76.2'}  
  • {'subdomain': 'rafflewebsite-27fgsznibq-nn.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.75.2'}  
  • {'subdomain': 'send-message-oqbc77fepq-uc.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.77.2'}  
  • {'subdomain': 'takuserver-ozzpjk7pa-uc.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.79.2'}  
  • {'subdomain': 'fvriznkd---kmhkqybb-bsccljbcrcq-ez.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.78.2'}  
  • {'subdomain': 'mango-slra1-ckwssph7iq-ue.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.72.2'}  
  • {'subdomain': 'prebid-js-prd-cvcf4dpepa-an.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.74.2'}  
  • {'subdomain': 'uc-assistant-service-crmkue6ra-nn.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.77.2'}  
  • {'subdomain': 'wernicke-streamlit-gcp-4bkpdxihwa-an.a.run.app',  
'source': 'subfinder', 'resolved': '34.143.72.2'}  
  • {'subdomain': 'metaseoapp-app-service-2kt2gk6m7a-ey.a.run.app',  
'source': 'subfinder', 'resolved': '34.143.72.2'}  
  • {'subdomain': 'gw.a.run.app', 'source': 'subfinder', 'resolved':  
'34.143.77.2'}  
  • {'subdomain': 'task-server-ivvoisqota-an.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.75.2'}  
  • {'subdomain': 'island-adur-fyrr-c3buvga5wa-lz.a.run.app', 'source':  
'subfinder', 'resolved': '34.143.73.2'}  
  • {'subdomain': 'issa-mzxplocgaa-ew.a.run.app', 'source': 'subfinder',  
'resolved': '34.143.79.2'}  
  • {'subdomain': 'ssl.run.app', 'source': 'subfinder', 'resolved':  
'34.143.74.2'}  
  • {'subdomain': 'krazyuncle.com',  
'source': 'subfinder', 'resolved': '34.143.76.2'}
```

- {'subdomain': 'us-west3.run.app', 'source': 'subfinder', 'resolved': '34.143.74.2'}
- {'subdomain': 'dummyai-backend-gwlr6iakq-uc.a.run.app', 'source': 'subfinder', 'resolved': '34.143.78.2'}
- {'subdomain': 'ess-api-pjfyhqc3kq-ey.a.run.app', 'source': 'subfinder', 'resolved': '34.143.78.2'}
- {'subdomain': 'iap-gcip-hosted-ui-enotify-backend-rkrjb5p7pa-wl.a.run.app', 'source': 'subfinder', 'resolved': '34.143.78.2'}
- ... and 80 more

OPEN PORTS

- Port 80: HTTP
- Port 443: HTTPS

DNS RECORDS

A:

- 34.143.72.2
- 34.143.74.2
- 34.143.78.2
- 34.143.76.2
- 34.143.77.2

AAAA:

- 2600:1900:4245:200::
- 2600:1901:81d4:200::
- 2600:1900:4240:200::
- 2600:1900:4241:200::
- 2600:1901:81d5:200::

DISCOVERED API ENDPOINTS

- GraphQL: <https://krazyuncle.com/graphql>
- unknown: <https://krazyuncle.com/graphiql>
- unknown: <https://krazyuncle.com/altair>
- unknown: <https://krazyuncle.com/playground>
- GraphQL: <https://krazyuncle.com/v1/graphql>
- OpenAPI/Swagger: <https://krazyuncle.com/swagger.json>
- OpenAPI/Swagger: <https://krazyuncle.com/swagger.yaml>
- OpenAPI/Swagger: <https://krazyuncle.com/openapi.json>
- OpenAPI/Swagger: <https://krazyuncle.com/>

```
openapi.yaml
  • OpenAPI/Swagger: https://krazyuncle.com/
swagger-ui.html
  • OpenAPI/Swagger: https://krazyuncle.com/
swagger-ui/
  • REST API: https://krazyuncle.com/api-docs
  • REST API: https://krazyuncle.com/v2/api-
docs
  • REST API: https://krazyuncle.com/v3/api-
docs
  • unknown: https://krazyuncle.com/redoc
```

7. Attack Analysis

OVERALL SECURITY ASSESSMENT

Overall Risk Level: MEDIUM

Total findings: 8 (0 critical, 0 high, 4 medium, 3 low, 1 informational)

Priority: Implement missing security headers to improve baseline security posture.

ATTACK SCENARIOS

****Man-in-the-Middle Attack via Weak TLS****: An attacker on a public WiFi network intercepts customer traffic by exploiting the outdated TLS 1.0 protocol and lack of HSTS headers. Without HSTS forcing secure connections, the attacker downgrades users to HTTP and steals login credentials, payment information, and session tokens, potentially compromising hundreds of customer accounts.

****Cross-Site Scripting (XSS) Data Theft****: A malicious actor injects JavaScript code into user-generated content or comments, which executes in other users' browsers due to the missing Content Security Policy. The script harvests sensitive customer data, session cookies, and personal information, then exfiltrates it to the attacker's servers, leading to identity theft and account takeovers.

****API Data Harvesting via CORS Misconfiguration****: Cybercriminals create malicious websites that exploit the wildcard CORS policy to make unauthorized

requests to your application's APIs from victims' browsers. This allows them to extract customer data, order histories, and business intelligence at scale while appearing to come from legitimate user sessions, potentially exposing your entire customer database.

PRIORITIZED REMEDIATION

-
1. CORS Wildcard Origin Allowed
Priority Score: 60
Action: Restrict CORS to specific trusted origins
Factors: CORS can lead to data theft
 2. Missing Security Header: HTTP Strict Transport Security (HSTS)
Priority Score: 55
Action: Add missing security headers to server configuration
Factors: Strong evidence
 3. Missing Security Header: Content Security Policy (CSP)
Priority Score: 55
Action: Add missing security headers to server configuration
Factors: Strong evidence
 4. TLS 1.0 Protocol Enabled
Priority Score: 50
Action: Disable TLS 1.0 in server configuration. Use TLS 1.2 or TLS 1.3 only.
 5. Missing Security Header: X-XSS-Protection
Priority Score: 25
Action: Add missing security headers to server configuration
Factors: XSS is directly exploitable, Strong evidence

8. Appendix

A. Raw Findings Data

APPENDIX A: Detected Technologies

- cloudflare
- google cloud
- google cloud load balancing
- onsen ui
- osano:3.1.0
- cdnjs
- jquery:2.2.4
- google frontend

APPENDIX B: Finding Summary

1. [MEDIUM] Missing Security Header: HTTP Strict Transport Security (HSTS)
Location: <https://krazyuncle.com>
2. [MEDIUM] Missing Security Header: Content Security Policy (CSP)
Location: <https://krazyuncle.com>
3. [INFO] Missing Security Header: X-XSS-Protection
Location: <https://krazyuncle.com>
4. [LOW] Missing Security Header: Referrer-Policy
Location: <https://krazyuncle.com>
5. [LOW] Missing Security Header: Permissions-Policy
Location: <https://krazyuncle.com>
6. [MEDIUM] CORS Wildcard Origin Allowed
Location: <https://krazyuncle.com>
7. [MEDIUM] TLS 1.0 Protocol Enabled
Location: [krazyuncle.com:443](https://krazyuncle.com)
8. [LOW] TLS 1.1 Protocol Enabled
Location: [krazyuncle.com:443](https://krazyuncle.com)

This report contains confidential security information and is intended for authorized recipients only.

Unauthorized distribution or disclosure is prohibited.