API SECURITY RISK ANALYSIS REPORT

Completed By: Dhanush Kumar S
Program: Cyber Security Internship – Future Interns
Task: API Security Risk Analysis (Task 3 – 2026)
Assessment Type: Read-Only Security Review
Date: January 2026

Executive Summary

Modern SaaS applications depend heavily on APIs to deliver functionality across web applications, mobile apps, dashboards, and third-party integrations. While APIs improve efficiency and scalability, they also introduce significant security risks if not properly protected.

This report documents a read-only API Security Risk Analysis performed on a public demo API. The assessment focuses on identifying common API security weaknesses, explaining their business impact, and recommending remediation steps using industry best practices aligned with the OWASP API Security Top 10.

No exploitation or unauthorized actions were performed during this assessment.

API Overview

API Name: JSONPlaceholder
API Type: Public Demo REST API
Base URL:
https://jsonplaceholder.typicode.com
Purpose:
JSONPlaceholder is a fake online REST API designed for testing and learning. It provides sample data such as users, posts, and comments.
Although this API is intentionally open, the findings in this report represent real security risks if discovered in a production SaaS environment.

Scope & Ethics

In Scope

- Public API endpoints
- Read-only GET requests
- Request and response header inspection
- Response data analysis

Out of Scope

- Exploitation attempts
- Authentication bypass
- Denial-of-Service or flooding
- Modification of data
- Private or production APIs

Ethics Statement

All activities were conducted legally and ethically on a public demo API intended for educational use.

Tools Used

- Web Browser (Brave / Chrome)
- Browser Developer Tools (Network Tab)
- GitHub (documentation and evidence storage)

Methodology

The assessment followed a structured consultant-style workflow:

1. Reviewed API documentation
2. Identified public endpoints
3. Sent read-only GET requests via browser
4. Inspected request and response headers
5. Analyzed response data for overexposure
6. Mapped findings to OWASP API Security Top 10
7. Assessed risk severity and business impact
8. Proposed remediation recommendations

Endpoints Tested

Endpoint	Method	Description
/posts	GET	Retrieves posts
/users	GET	Retrieves user data
/comments	GET	Retrieves comments

Security Findings

Unauthenticated API Access

Observation:

All tested endpoints are accessible without authentication (no API key, token, or OAuth).

Severity: Medium

OWASP Category: API2 – Broken Authentication

Business Impact:

- Unauthorized users can access API data
- Enables mass data scraping
- Provides attackers with reconnaissance data

Remediation:

- Enforce authentication (JWT, OAuth 2.0, API keys)
- Reject unauthenticated requests by default

Excessive Data Exposure

Observation:
API responses return full objects including email addresses and detailed fields without filtering.
Severity: Medium
OWASP Category: API3 – Excessive Data Exposure
Business Impact:

- Increased privacy risks
- Potential compliance violations
- Easier mapping of internal data structures

Remediation:

- Return only required fields
- Implement response filtering
- Apply least-privilege principles

Missing Rate Limiting

Observation:
No rate-limiting headers such as X-RateLimit-Limit were observed.

Severity: High

OWASP Category: API4 – Lack of Resources & Rate Limiting

Business Impact:

- Enables automated scraping
- Risk of denial-of-service attacks
- Increased infrastructure costs

Remediation:

- Implement rate limiting per IP or token
- Apply throttling and burst controls
- Monitor abnormal traffic

Missing Authorization Controls

Observation:
User-related data is accessible without validating identity or ownership.

Severity: High

OWASP Category: API1 – Broken Object Level Authorization (BOLA)

Business Impact:

- Users could access other users’ data
- Serious privacy and legal risks
- Loss of customer trust

Remediation:

- Enforce object-level authorization
- Validate user ownership
- Implement role-based access control (RBAC)

Missing Security Headers

Observation:
API responses lack common security headers.

Severity: Low

Business Impact:

- Reduced defense-in-depth
- Increased exposure to misuse

Remediation:

- Add headers such as:
  - X-Content-Type-Options
  - Content-Security-Policy

Risk Summary

Risk	Severity
Unauthenticated access	Medium
Excessive data exposure	Medium
Missing rate limiting	High
Authorization issues	High
Missing security headers	low

Key Takeaways

- APIs must never trust the client
- Authentication alone is not sufficient
- Authorization is the most critical control
- Rate limiting protects availability
- Data minimization reduces breach impact

Conclusion

This API Security Risk Analysis demonstrates how common API vulnerabilities can be identified using ethical, read-only techniques. The assessment reflects real-world SaaS security consulting practices and highlights the importance of secure API design.