



API Security Risk Analysis Report

Security risk report

Prepared by:
KAKUMANU JAYADHARANI

Future Interns Project

Cybersecurity Task 3

OBJECTIVE

The objective of this task is to perform a read-only API Security Risk Analysis using public or demo APIs. The analysis focuses on identifying common security risks such as unauthenticated access, excessive data exposure, and missing security controls without exploiting or attacking the system. This task helps understand how API security audits are performed in real-world security consulting environments.

API SELECTED

API Name:JSONPlaceholder

Base URL:

<https://jsonplaceholder.typicode.com>

Why this API?

Public and safe for testing

Designed for learning purposes

No real sensitive data

Suitable for beginner security analysis

TOOLS & TESTING ENVIRONMENT

The following tools were used during analysis:

Postman

- Sending API requests
- Viewing responses
- Header inspection

Kali Linux Terminal

- Header analysis using curl command

Command Used:

- curl -I
<https://jsonplaceholder.typicode.com/posts>

TESTING PROCESS

- Reviewed API documentation in browser.
- Selected safe public API endpoints.
- Sent GET requests using Postman.
- Observed response data structure.
- Checked authentication requirements.
- Inspected headers and status codes.
- Identified potential security risks.
- Classified risk severity and documented observations.

ENDPOINTS TESTED

./posts Endpoint

Method: GET

Returns post objects with userId, title, body.

Accessible without authentication.

./users Endpoint

Method: GET

Returns user details such as name, email, phone, address, and company info.

. /comments Endpoint

Method: GET

Returns comment records linked to posts.

SECURITY FINDINGS

⚠ Finding 1 – Unauthenticated Endpoint

All tested endpoints are publicly accessible without authentication.

Risk Level: Medium

Impact: Unauthorized users can access API data.

⚠ Finding 2 – Excessive Data Exposure

The users endpoint returns full user objects including contact and address details.

Risk Level: Medium

Impact: In real APIs this could lead to information leakage.

⚠ Finding 3 – Missing Rate Limiting Indicators

No clear rate-limiting information observed in responses.

Risk Level: Medium

Impact: API may be vulnerable to abuse through excessive requests.

⚠ Finding 4 – Public Demo Data

Data appears to be test/demo information.

Risk Level: Low

Impact: No real sensitive data exposure.

RISK SEVERITY TABLE

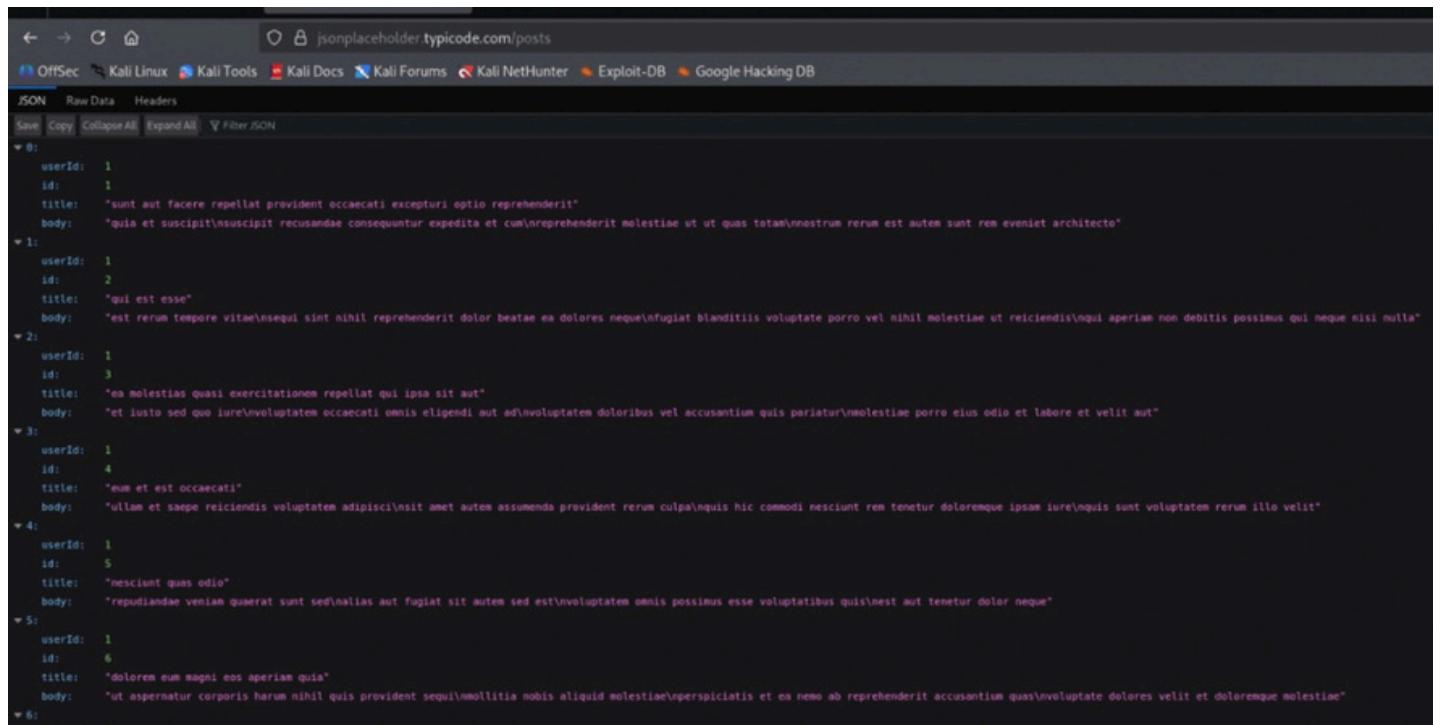
RISK	SEVERITY	IMPACT
Unauthenticated endpoint access	Medium	Unauthorized access
Excessive data exposure	Medium	Information leakage
Missing rate limiting	Medium	API abuse possible
Demo/Public data	Low	Minimal impact

REMEDIATION RECOMMENDATIONS

To improve API security, the following practices are recommended:

- Implement authentication tokens (API keys / OAuth).
- Return only required fields in responses.
- Enable rate limiting to prevent abuse.
- Apply input validation for request data.
- Use security headers consistently.

SCREENSHOTS



A screenshot of a web browser displaying a JSON response from jsonplaceholder.typicode.com/posts. The browser interface includes a header with tabs like OffSec, Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, and Google Hacking DB. Below the header, there are buttons for Save, Copy, Collapse All, Expand All, and Filter JSON. The main content area shows an array of 10 objects, each representing a post with fields: userId, id, title, and body. The titles and bodies contain placeholder Latin text.

```
[{"userId": 1, "id": 1, "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit", "body": "quia et suscipit\\nscipit recusandae consequuntur expedita et cum\\nreprehenderit molestiae ut ut quas totam\\nnostrum rerum est autem sunt rem eveniet architecto"}, {"userId": 1, "id": 2, "title": "qui est esse", "body": "est rerum tempore vitae\\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\\nqui aperiam non debitis possimus qui neque nisi nulla"}, {"userId": 1, "id": 3, "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut", "body": "et justo sed quo iure\\nvolutatem occaecati omnis eligendi aut ad\\nvolutatem doloribus vel accusantium quis pariatur\\nmolestiae porro eius odio et labore et velit aut"}, {"userId": 1, "id": 4, "title": "eum et est occaecati", "body": "ullam et saepe reiciendis voluptatem adipisci\\nsit amet autem assumenda provident rerum culpa\\nquis hic commodi nesciunt rem tenetur doloremque ipsam iure\\nquis sunt voluptatem rerum illo velit"}, {"userId": 1, "id": 5, "title": "nesciunt quas odio", "body": "repudiandae veniam quamrat sunt sed\\nalias aut fugiat sit autem sed est\\nvolutatem omnis possimus esse voluptatis quis\\nest aut tenetur dolor neque"}, {"userId": 1, "id": 6, "title": "dolorem eum magni eos aperiam quia", "body": "ut aspernatur corporis harum nihil quis provident sequi\\nmollitia nobis aliquid molestiae\\nperspiciatis et ea nemo ab reprehenderit accusantium quas\\nvolutate dolores velit et doloremque molestiae"}]
```

The screenshot shows the Postman application interface. At the top, there are three tabs: 'Getting started' (disabled), 'GET https://jsonplaceholder.typicode.com/posts' (selected), and 'GET https://jsonplaceholder.typicode.com/posts'. Below the tabs, the URL 'https://jsonplaceholder.typicode.com/posts' is entered again. On the left side, there are navigation buttons for 'Collections', 'Environments', and 'History'. The main workspace has a 'Params' tab selected under 'GET' requests. A table titled 'Query Params' shows one entry: 'Key' (Key) and 'Value' (Value). Below this, tabs for 'Body', 'Cookies', 'Headers (25)', and 'Test Results' are visible. The 'Body' tab is selected, showing a JSON preview of the response. The response body is a JSON array of 29 objects, each representing a post. The first few objects are:

```
[{"id": 1, "userId": 1, "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit", "body": "quia et suscipit\\nscipit recusandae consequuntur expedita et cum\\nreprehenderit molestiae ut ut quas totam\\nnestrum rerum est autem sunt rem eveniet architecto"}, {"id": 2, "userId": 1, "title": "qui est esse", "body": "est rerum tempore vitae\\nsequi sint nihil reprehenderit dolor beatas ea dolores neque\\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\\nqui aperiam non debitis possimus qui neque nisi nulla"}, {"id": 3, "userId": 1, "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut", "body": "et iusto sed quo iure\\nvolutatem occaecati omnis eligendi aut ad\\nvolutatem doloribus vel accusantium quis pariatur\\nmolestiae porro eius odio et labore et velit aut"}, {"id": 4, "userId": 1, "title": "eum et est occaecati", "body": "ullam et saepe reiciendis voluptatem adipisci\\nsit amet autem assumenda provident rerum culpa\\nquis hic commodi nesciunt rem tenetur dolorerisque ipsam iure\\nquis sunt voluptatem rerum illo velit"}, {"id": 5, "userId": 1, "title": "nesciunt quas odio", "body": "fuga et est occaecati\\niste voluptatem adipisci\\nisi amet autem assumenda provident rerum culpa\\nquis hic commodi nesciunt rem tenetur dolorerisque ipsam iure\\nquis sunt voluptatem rerum illo velit"}]
```

At the bottom of the interface, there are buttons for 'Runner', 'Capture requests', 'Cookies', 'Vault', and 'Trash'.

```
(root㉿kali)-[~/home/kali]
# curl -I https://jsonplaceholder.typicode.com/posts
HTTP/2 200
date: Tue, 17 Feb 2026 13:02:00 GMT
content-type: application/json; charset=utf-8
access-control-allow-credentials: true
cache-control: max-age=43200
etag: W/"6b80-Ybsq/K6GwqrYkAsFxqDXGC7DoM"
expires: -1
nel: {"report_to": "heroku-nel", "response_headers": [{"Via": "1.1 0.0.0.0:443"}, {"Content-Type": "application/json", "Content-Length": "1234", "Date": "Tue, 17 Feb 2026 13:02:00 GMT", "Server": "Cloudflare", "X-Content-Type-Options": "nosniff", "X-Powered-By": "Express", "X-RateLimit-Limit": "1000", "X-RateLimit-Remaining": "999", "X-RateLimit-Reset": "1771161197", "Age": "28132", "CF-Cache-Status": "HIT", "CF-Ray": "9cf57a065a3df1f-SIN", "Alt-Svc": "h3=:443"; ma=86400}], "max_age": 3600, "success_fraction": 0.01, "failure_fraction": 0.1}
pragma: no-cache
report-to: {"group": "heroku-nel", "endpoints": [{"url": "https://nel.herokuapp.com/reports?s=9PWz1wYqSXFl%2B0kpvCXHzdvru%2FCLEF%2Fz1HL197eq80g%3D&sid=e11707d5-02a7-43ef-b45e-2cf4d2036f7d\u0026ts=1771161155"}], "max_age": 3600}
reporting-endpoints: heroku-nel="https://nel.herokuapp.com/reports?s=9PWz1wYqSXFl%2B0kpvCXHzdvru%2FCLEF%2Fz1HL197eq80g%3D&sid=e11707d5-02a7-43ef-b45e-2cf4d2036f7d&ts=1771161155"
server: cloudflare
vary: Origin, Accept-Encoding
via: 2.0 heroku-router
x-content-type-options: nosniff
x-powered-by: Express
x-ratelimit-limit: 1000
x-ratelimit-remaining: 999
x-ratelimit-reset: 1771161197
age: 28132
cf-cache-status: HIT
cf-ray: 9cf57a065a3df1f-SIN
alt-svc: h3=:443"; ma=86400
```

Home Workspaces API Network

Getting started

Contract Testing / Test Response

GET https://jsonplaceholder.typicode.com/users

Send

Body Cookies Headers (26) Test Results (1/5)

{} JSON

```

184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
  },
  {
    "id": 9,
    "name": "Glenna Reichert",
    "username": "Delphine",
    "email": "Chasim_McDermott@dane.io",
    "address": {
      "street": "Bayne Park",
      "suite": "Suite 449",
      "city": "Bartholomewbury",
      "zipcode": "76495-3109",
      "geo": {
        "lat": "24.6463",
        "lng": "-160.8889"
      }
    },
    "phone": "(775)976-8794 x41206",
    "website": "conrad.com",
    "company": {
      "name": "Yost and Sons",
      "catchPhrase": "Switchable contextually-based project",
      "bs": "aggregate real-time technologies"
    }
  },
  {
    "id": 10,
    "name": "Clementine DuBuque",
    "username": "Moriah.Stanton",
    "email": "Rey.Padberg@Kazina.biz",
    "address": {
      "street": "Kattie Tumpike",
      "suite": "Suite 198"
    }
  }
]
```

200 OK · 186 ms · 2.94 KB · Save Response

The screenshot shows the Postman application interface. At the top, there are tabs for Home, Workspaces, and API Network. Below the tabs, a search bar says "Search Postman". A toolbar includes "Ctrl + K", "Save", "Share", and "No environment". The main area shows a "Contract Testing / Test Response" section with a "GET" method selected. The URL is set to "https://jsonplaceholder.typicode.com/comments". Below the URL, there are tabs for Overview, Params, Authorization, Headers (8), Body, Scripts, and Settings. The "Body" tab is active, showing a JSON response with 29 comments. The first few comments are as follows:

```
198 [
199   {
200     "postId": 6,
201     "id": 29,
202     "name": "eum distinctio amet dolor",
203     "email": "Jennings_Pouros@erica.biz",
204     "body": "tempora voluptatem est\\nmagnam distinctio autem est dolorem\\net ipsa molestiae odit rerum itaque corporis nihil nam\\neaque rerum error"
205   },
206   {
207     "postId": 6,
208     "id": 30,
209     "name": "quasi nulla ducimus facilis non voluptas aut",
210     "email": "Lurline@marvin.biz",
211     "body": "consequuntur quia voluptate assumenda et\\nautem voluptatem reiciendis ipsum animi est provident\\nearum aperiam sapiente ad vitae iste\\naccusantium aperiam eius qui"
212   },
213   {
214     "postId": 7,
215     "id": 31,
216     "name": "ex velit ut cum eius odio ad placeat",
217     "email": "Buford@shaylee.biz",
218     "body": "quia incident ut\\naliiquid est ut rerum deleniti iure est\\nipsum quia ea sint et\\nvolutpatem quaerat eaque repudiandae eveniet aut"
219   },
220   {
221     "postId": 7,
222     "id": 32,
223     "name": "dolorem architecto ut pariatur quae qui suscipit",
224     "email": "Maria@lauzel.name",
225     "body": "nihil ea itaque libero ille\\nof officiis quo quo dicta inventore consequatur voluptas voluptatem\\ncorporis sed necessitatibus velit tempore\\nrerum velit et temporibus"
226   },
227   {
228     "postId": 7,
229     "id": 33,
     "name": "voluptatum totam vel voluptate omnis",
  }
```

At the bottom left, there are buttons for "Online", "Find and replace", and "Console". On the right, there are buttons for "200 OK", "470 ms", "40.81 KB", "Save Response", and "Bulk Edit".

CONCLUSION

This assessment demonstrated a basic API security review using safe and ethical testing methods. The API endpoints were openly accessible and highlighted common risks such as unauthenticated access and excessive data exposure. Although the API contains demo data, the analysis reflects real-world API security considerations and provides practical experience in documenting security risks professionally.