API Security Risk Analysis Report

Saghana K S

6/02/2026

1. Introduction

Application Programming Interfaces (APIs) enable communication between different software systems. In cybersecurity, APIs must be tested to ensure they handle requests securely and do not expose sensitive data. This report documents basic API security testing performed using Postman. The objective is to analyze HTTP methods, headers, response data, and identify potential security risks without exploiting the API.

2. Tools Used

- Postman (API testing tool)
- httpbin.org (Public API testing service)

3. Testing Methodology

The testing was performed by sending HTTP requests using Postman. Public endpoints provided by httpbin.org were selected to safely analyze API behavior. GET and POST requests were tested, headers were inspected, and responses were analyzed using HTTP status codes and JSON output.

4. Test Case 1 – GET Request

Objective:

To retrieve data from the server and analyze the response.

API Details:

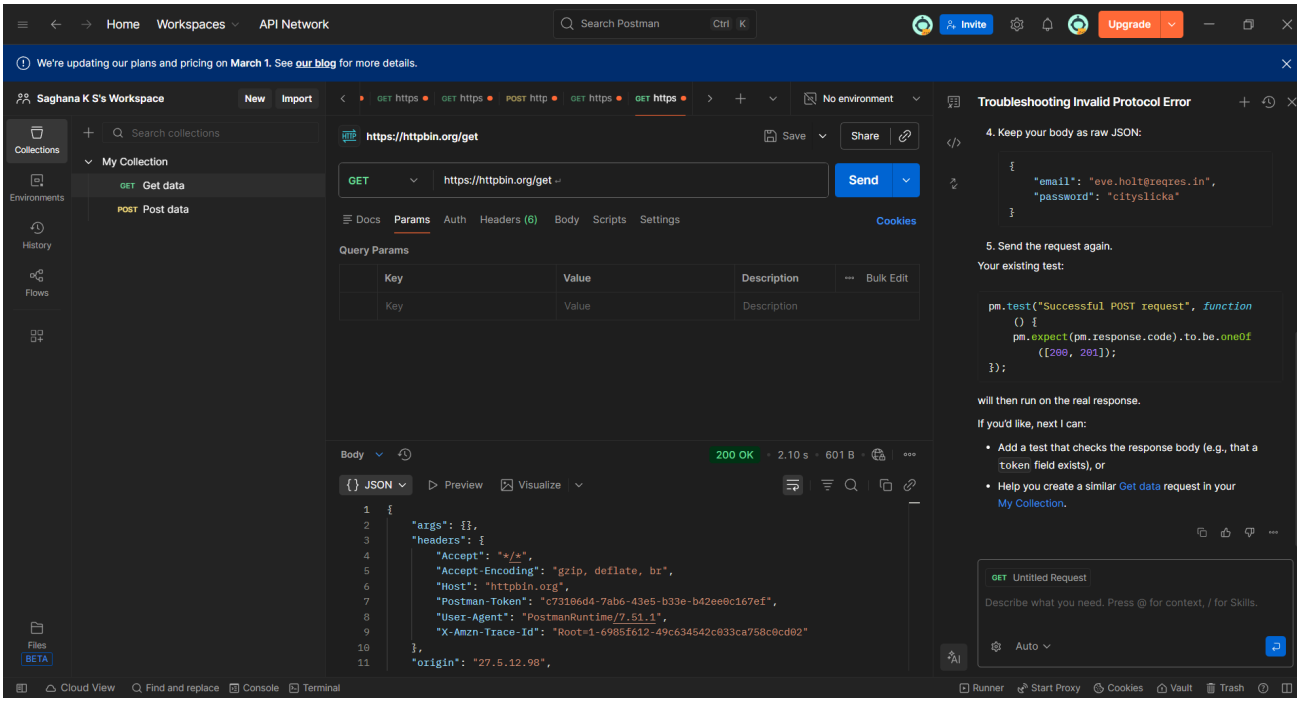
- **Method:** GET
- **Endpoint:** https://httpbin.org/get

Result:

- **Status Code:** 200 OK
- **Response Type:** JSON

Observation:

The server successfully returned data without requiring authentication, indicating an open endpoint.



5. Test Case 2 – POST Request

Objective:

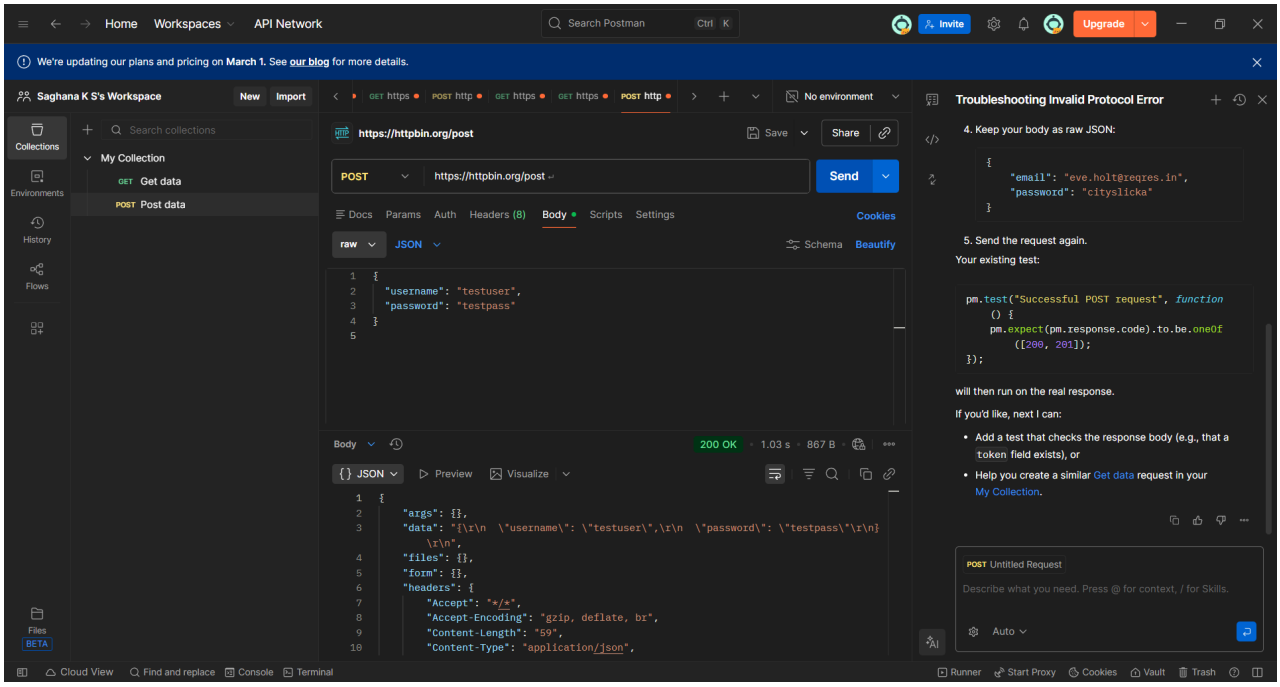
To retrieve data from the server and analyze the response.

API Details:

- **Method:** POST
- **Endpoint:** https://httpbin.org/post
- **Payload:** Username and password in JSON format

Result:

- **Status Code:** 200 OK
- **Response Type:** JSON (data echoed by server)



Observation:

The API accepted and processed the data successfully. No authentication or validation was enforced on the submitted data.

6. Test Case 3 – Header Analysis

Objective:

To retrieve data from the server and analyze the response.

API Details:

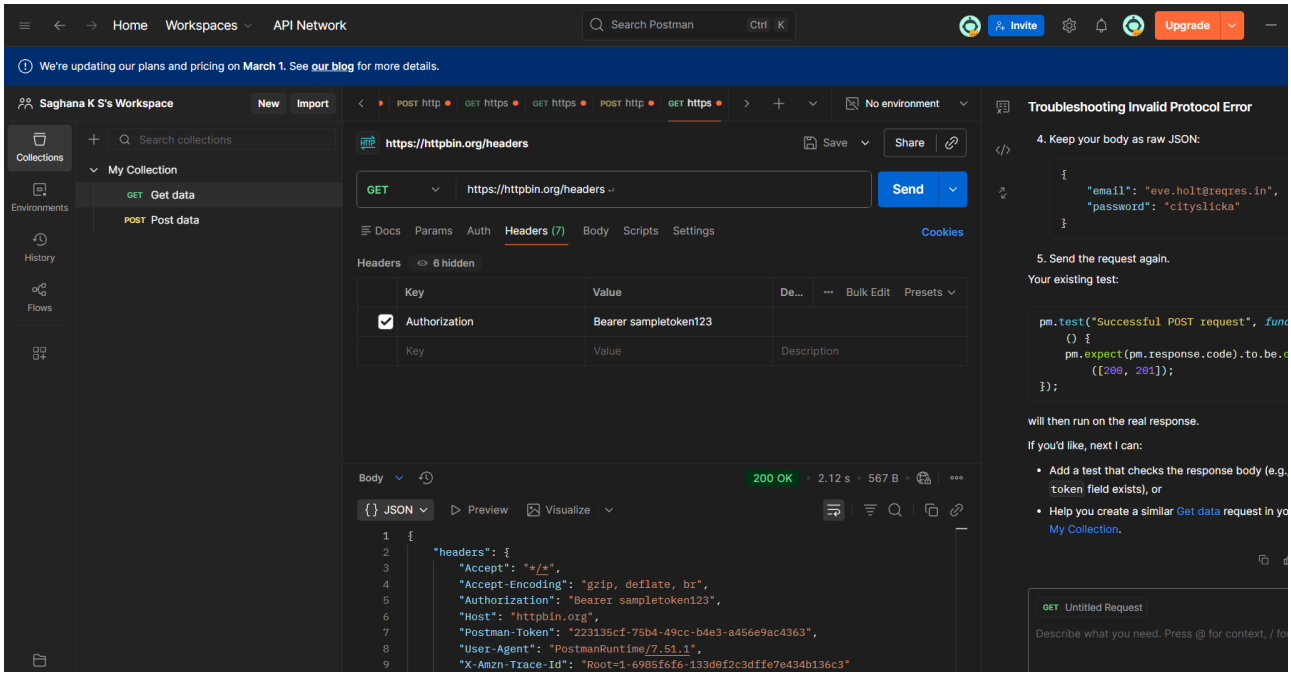
- **Method:** GET
- **Endpoint:** https://httpbin.org/headers
- **Custom Header Added:** Authorization: Bearer sampletoken123

Result:

- **Status Code:** 200 OK
- **Response:** Server reflected received headers

Observation:

The Authorization header was accepted without validation, demonstrating how APIs receive authentication-related headers.



7. Identified Security Risks

Risk Identified	Description	Severity
Open Endpoints	API endpoints accessible without authentication	Medium
Missing Authentication	No enforcement of authentication tokens	High
Header Misuse	Authorization headers accepted without validation	Medium
Input Validation	User input accepted without checks	Medium
Rate Limiting	No visible rate limiting mechanism	Low

8. Risk Severity Classification

- **High Risk:** Missing authentication mechanisms
- **Medium Risk:** Open endpoints, header misuse, input validation issues
- **Low Risk:** Rate limiting not enforced

9. Suggested Remediation Steps

- Implement proper authentication and authorization mechanisms
- Use secure token-based authentication (JWT, OAuth)
- Validate user input to prevent injection attacks
- Apply rate limiting to prevent abuse
- Restrict access to sensitive endpoints

10. Conclusion

This task provided practical exposure to API security testing using Postman. By testing GET and POST requests and analyzing headers, several potential security risks were identified. The exercise improved understanding of API behavior, HTTP status codes, and common security weaknesses. This analysis was conducted safely without exploiting the API.

11. References

- https://httpbin.org
- Postman Documentation