CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING (C-DAC)
THIRUVANANTHAPURAM, KERALA

A PROJECT REPORT ON

"Web Application Penetration Testing"

SUBMITTED TOWARDS THE

PG-DCSF August 2025

BY

Group Number - 04

Harshit Shrivastav	**PRN:250260940012**
Harsh Patidar	**PRN: 250260940011**
Nimisha Goyal	**PRN: 250260940019**
Anurag Dadhich	**PRN: 250260940005**
A V Yashwanth	**PRN: 250260940037**

Under The Guidance Of

Mr. Jayaram P.	**Mr. Hiron Bose**
Co-Ordinator	**Project Guide**

Table of Contents

Web Application Penetration Testing Report – Flipkart

1.1 Executive Summary

As a group of cybersecurity students from **CDAC**, we engaged in a structured white-box **Web Application Penetration Testing (VAPT)** exercise targeting **Flipkart.com**. The assessment was carried out between **22nd July 2025 and 29th July 2025**, under academic supervision, and aimed at evaluating the application's security posture using industry-accepted standards.

The purpose of this Web Application Security Assessment was to discover exploitable vulnerabilities and identify insufficiently configured security controls. The objective was to evaluate the security posture of Flipkart's platform against attackers with varying levels of access, including both authenticated and unauthenticated users, in accordance with real-world threat models.

The testing approach was aligned with established frameworks such as the **OWASP Top 10**, **OWASP Testing Guide**, **OWASP ASVS**, **SANS 25**, **NIST 800-115**, and the **Penetration Testing Execution Standard (PTES)**. The evaluation included both automated scanning using tools like **Acunetix** and extensive manual testing using **Burp Suite Professional** and its relevant extensions.

This assessment targeted various application components including authentication mechanisms, user session management, access control enforcement, file upload functionalities, API endpoints, and internal request handling. The intent was to simulate realistic attacker behavior and determine the presence and severity of potential vulnerabilities.

It is important to note that while tools supported the process, the majority of the assessment relied on manual analysis, logical testing, and behavioral observation of the application under test. The findings outlined in this report represent the application's security posture during the assessment period. Any subsequent changes to the system or environment are not reflected in this report.

This document includes detailed findings, analysis, potential impact ratings, and recommended mitigations categorized under the **OWASP Top 10 – 2021** structure, providing a clear and actionable roadmap for remediation and future hardening.

The target for this assessment, Flipkart.com , was selected from its publicly available bug bounty program hosted on HackerOne. This ensured that the assessment was conducted within a legal, ethical, and authorized scope under Flipkart's disclosed guidelines.

1.2 Background

This exercise aimed to perform Penetration Testing of the Applications in scope to determine if they were vulnerable to attacks and exploitation. The test consisted of manual and automated testing to detect and exploit vulnerabilities. The assessment was conducted to identify the security vulnerabilities in the Application in scope and propose solutions for the project team to remediate the identified vulnerabilities to make the Application more secure.

The security assessment was performed by a team of cybersecurity students from **CDAC** between **22nd July 2025 to 29th July 2025**.

1.3 Objectives

The objective of the tests performed was to:

- Identify the possible vulnerabilities and gaps in the web applications.
- Assess the gaps and vulnerabilities to determine the Risk associated.
- Suggest recommendations to overcome existing vulnerabilities and gaps.

This assessment report contains:

- Technical details of the vulnerabilities discovered with substantiation of the exploits.
- Risk mitigation recommendations that need to be implemented to ensure that the systems are secure from the risks arising due to the discovered vulnerabilities.

1.4 Scope of Assessment

This white-box assessment focused on the OWASP Top 10 vulnerabilities and standard security best practices, including but not limited to:

- Information Gathering
- Configuration Management
- Business Logic Testing
- Authentication & Authorization
- Session Management
- Input Validation and Output Encoding
- File Upload and URL Redirection Handling
- Access Control and Privilege Escalation
- Server-Side Request Forgery (SSRF)
- Token and Cookie Analysis

Targeted Areas:

- Flipkart's main web application: https://www.flipkart.com
- Web requests associated with review uploads, login, order tracking, and account information

Testing Environment:

- **Platform**: Web Application (Production)
- **Accessibility**: Internet-based Public Scope
- **Authentication**: OTP and credential-based user login
- **Backend Architecture**: Cloud-hosted (AWS and associated APIs)

1.5 Out of Scope

The following items and activities were considered out of scope for this engagement due to the potential impact on production services, ethical constraints, or scope limitations as defined by Flipkart:

- Denial of Service (DoS) / Distributed DoS (DDoS) Attacks
- Brute Force or Password Spraying Attacks on Real User Accounts
- Excessive Automated Scanning or Fuzzing
- Social Engineering Techniques (Phishing, Vishing, Smishing)
- Attacks involving malware injection or remote payload execution
- Physical Access Attempts or Insider Testing
- Any attack that causes degradation of production services
- Testing on Out-of-Scope Mobile Applications (Flipkart/Myntra apps)
- Username Enumeration via Signup or Password Recovery Forms
- Exploits requiring MITM or device-level access
- Low-impact issues like missing headers, outdated libraries without PoC
- Open Redirects unless chained with high-impact vulnerability
- Unauthenticated Logout/Login CSRF
- Self-XSS and content spoofing without exploitation vector
- Known CVEs without business impact or active exploitation
- Leaked documents, internal credentials, or external datasets unless proven impactful
- Cloud bucket exposures not involving customer data

1.6 Tools Used

A combination of manual and automated tools were utilized during the Web Application Penetration Testing process to ensure comprehensive vulnerability detection and validation. The tools used in this assessment are briefly described below:

Tool Name	Description
Burp Suite Professional	An advanced web vulnerability scanner and testing platform. It was used for intercepting traffic, testing APIs, analyzing parameters, and performing manual testing like Broken Access Control and SSRF. Key extensions like *Autorize* and *Collaborator* supported deeper analysis.
OWASP ZAP (Zed Attack Proxy)	An open-source web application security scanner used for both passive and active scans. It aided in discovering vulnerabilities such as PII disclosure, outdated components, and misconfigurations. ZAP spiders and component fingerprinting were also leveraged.
Hydra	A fast network logon cracker used for testing authentication endpoints. It was employed to perform credential-based brute-force attacks on Flipkart's login APIs, testing for rate-limiting and lockout mechanisms.
Acunetix Vulnerability Scanner	A commercial web vulnerability scanner that performs deep scans for over 7000+ vulnerabilities including SQLi, XSS, CSRF, and misconfigurations. It was used in Full Scan mode to detect security flaws across the application.
Tenable Nessus Essentials	A network-level vulnerability scanner used to detect open ports, OS fingerprints, and other network misconfigurations. It provided additional insights into the target's exposure from an infrastructure perspective.
Splunk Enterprise	A powerful SIEM (Security Information and Event Management) platform used to build SOC dashboards, monitor login activities, analyze firewall traffic, and detect alert trends. It was used for visualizing and correlating security events during the assessment.
SonarQube, Semgrep, Bandit, PHPStan	Static code analysis tools used during secure code review. These tools helped in identifying insecure design patterns, vulnerable functions like eval(), hardcoded credentials, and weak input validation.
Linux Command-line Tools (curl, grep, etc.)	Used to fetch headers, analyze responses, verify CSP settings, check server configurations, and automate various tests.
Nmap (Network Mapper)	An open-source network scanning tool used to discover hosts, open ports, and services. It assisted in the reconnaissance and enumeration phase to understand the external exposure of the Flipkart target environment.

1.7 Summary of Findings

OWASP ID	Vulnerability	Risk Level	Status
A01	Broken Access Control	Low	Tested
A02	Cryptographic Failures	-	Tested
A03	Injection	-	Tested
A04	Insecure Design	-	Tested
A05	Security Misconfiguration	-	Tested
A06	Vulnerable and Outdated Components	-	Tested
A07	Identification and Authentication Failures	-	Tested
A08	Software and Data Integrity Failures	-	Tested
A09	Security Logging and Monitoring Failures	-	Tested
A10	Server-Side Request Forgery (SSRF)	Low	Tested

1.8 Automated Scanning Phase

To support our manual testing efforts, an automated scanning phase was conducted using industry-recognized tools. This phase aimed to uncover known vulnerabilities, misconfigurations, and surface-level flaws to assist in deeper manual verification and exploitation.

1.8.1 Acunetix Vulnerability Scanner

Introduction to Acunetix

Acunetix is an automated web application security scanner that detects and reports on over 7,000 web vulnerabilities, including OWASP Top 10 risks such as SQL Injection, Cross-site Scripting (XSS), and CSRF. It performs deep scanning of websites, APIs, and complex JavaScript-heavy applications. Acunetix is widely used in industry due to its ability to crawl large applications and present detailed, categorized vulnerability reports.

Type of Scanning Performed

For the Flipkart application, Acunetix was used in **Full Scan (Comprehensive)** mode, which included:

- **Crawling**- all reachable pages and inputs
- **Fingerprinting**- technologies used on the website
- **Testing**- for input-based vulnerabilities (XSS, SQLi, Host Header Injection, etc.)
- **Authentication/Session-** checks
- **HTTP response and configuration analysis**- This mode ensured a broad surface-level scan, supporting our manual efforts by identifying potential weak spots.

Key Findings from Acunetix Scan

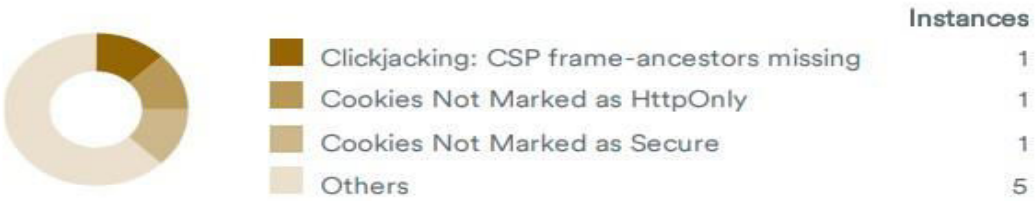
Below are the major findings with short descriptions:

Severity	Vulnerability	Description
High	Host Header Injection	The server accepts arbitrary Host headers, which could be used in web cache poisoning or redirect-based attacks.
Medium	Incomplete or No Cache-control and Pragma Headers	Responses lack proper caching headers, potentially exposing sensitive data in shared environments.
Medium	Missing X-Content-Type-Options Header	Without this header, browsers may interpret files as a different MIME type, enabling attacks.
Medium	Cookie Without SameSite Attribute	This weakens CSRF protections by allowing cookies to be sent with cross-origin requests.

Low	Trace Method Enabled	TRACE method can be used to conduct Cross-Site Tracing (XST) attacks.
Info	HTML Forms Without CSRF Tokens	Detected forms did not contain anti-CSRF tokens, increasing potential CSRF risk if exploited manually.
Info	Technology Disclosure	Frameworks and server types were identified (e.g., Apache, jQuery), which could help attackers tailor exploits.

Medium Severity

Low Severity



Informational

1.8.2 Tenable Nessus Scanner

Introduction to Nessus

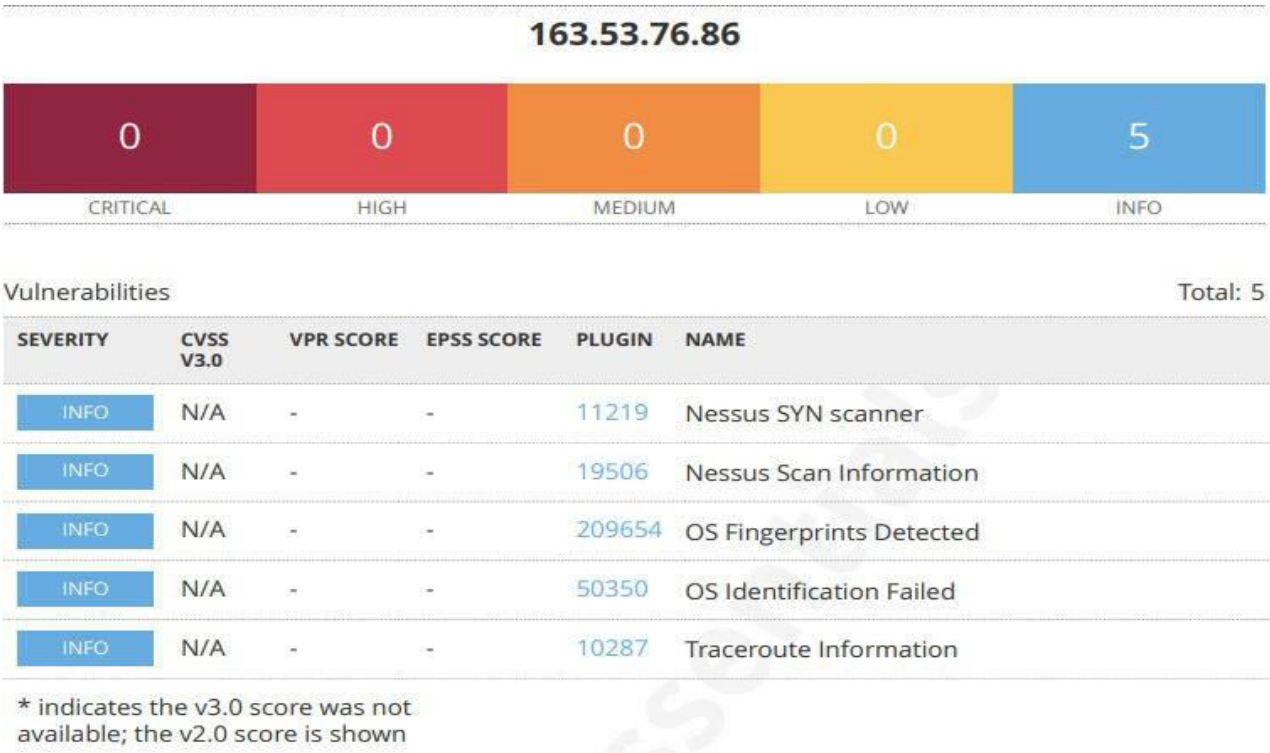
Nessus is a widely used vulnerability assessment tool developed by Tenable. It is primarily used for identifying vulnerabilities, misconfigurations, and missing patches on networked systems. Nessus performs both credentialed and non-credentialed scans and provides detailed results with plugin references, CVE IDs, and risk ratings.

In this project, we used **Nessus Essentials (free version)** to conduct a **Basic Network Scan** of the target IP. This helped us identify potential network-layer vulnerabilities, open ports, and operating system fingerprints as part of our reconnaissance and enumeration phase.

Type of Scanning Performed

The following scanning configuration was used:

- **Scan Policy:** Basic Network Scan
- **Scanner Edition:** Nessus Essentials (Home)
- **Target:** 163.53.76.86 (in-scope IP)
- **Port Scanner:** SYN scan (half-open TCP scan)
- **Plugins Used:** Network probing, traceroute, OS fingerprinting
- **Credentialed Check:** No
- **Web App Tests:** Disabled
- **Duration:** 789 seconds
- **Host Detected Open Ports:** 80/tcp and 443/tcp



163.53.76.86

0	0	0	0	5
CRITICAL	HIGH	MEDIUM	LOW	INFO

Vulnerabilities Total: 5

SEVERITY	CVSS V3.0	VPR SCORE	EPSS SCORE	PLUGIN	NAME
INFO	N/A	-	-	11219	Nessus SYN scanner
INFO	N/A	-	-	19506	Nessus Scan Information
INFO	N/A	-	-	209654	OS Fingerprints Detected
INFO	N/A	-	-	50350	OS Identification Failed
INFO	N/A	-	-	10287	Traceroute Information

* indicates the v3.0 score was not available; the v2.0 score is shown

Key Findings from Nessus Scan

Severity	**Plugin Name**	**Description**
Info	Nessus SYN Scanner (Plugin 11219)	Detected open ports 80 and 443 using half-open TCP scan. Helps map the attack surface.
Info	Traceroute Information (Plugin 10287)	Identified the network route from scanner to target; helpful in network mapping.
Info	Nessus Scan Information (Plugin 19506)	Detailed metadata of the scan like scanner version, OS, plugin feed, scan time, etc.
Info	OS Fingerprints Detected (Plugin 209654)	Detected low-confidence OS fingerprints using TCP/IP stack behavior.
Info	OS Identification Failed (Plugin 50350)	Nessus failed to conclusively identify the OS. Indicates lack of detailed system response.

2. OWASP Top 10 Vulnerability Analysis

2.1 Broken Access Control

2.1.1 Overview

Broken Access Control (BAC) vulnerabilities occur when an application fails to properly enforce restrictions on authenticated users. This can allow unauthorized users to access resources or perform actions beyond their intended permissions.

2.1.2 Methodology Used

As students performing the test with authorized access, we followed these steps to examine the presence of any access control issues:

- Manual URL manipulation: We altered parameters like order_id, item_id, and unit_id in the order history URL to test for IDOR vulnerabilities.
- Burp Suite Professional Site Map: We created a complete site map by browsing the application and analyzing traffic.
- Login-related API Sorting: We filtered and isolated authentication-related API endpoints (e.g., /user/login/otp).
- Manual Cookie Reuse Testing: We intercepted and attempted to reuse session cookies from one account in a different account session to check for privilege escalation.
- Login Parameter Tampering: Using Burp Repeater, we tried replacing values such as phone number (userId), otp, and requestId with valid data from one session into another.
- Automated Extension Used: *Autorize* extension was briefly used to detect access control flaws but yielded no actionable result in this case.

2.1.3 Tests Performed

- Tried manipulating order URLs to access different users' order details.
- Sent login requests with modified OTP and userId using valid session cookies from other accounts.
- Attempted cross-session cookie reuse by replacing tokens and headers from one authenticated user into another session.
- Observed server responses and error codes (e.g., HTTP 400, 406) indicating strong request validation mechanisms.

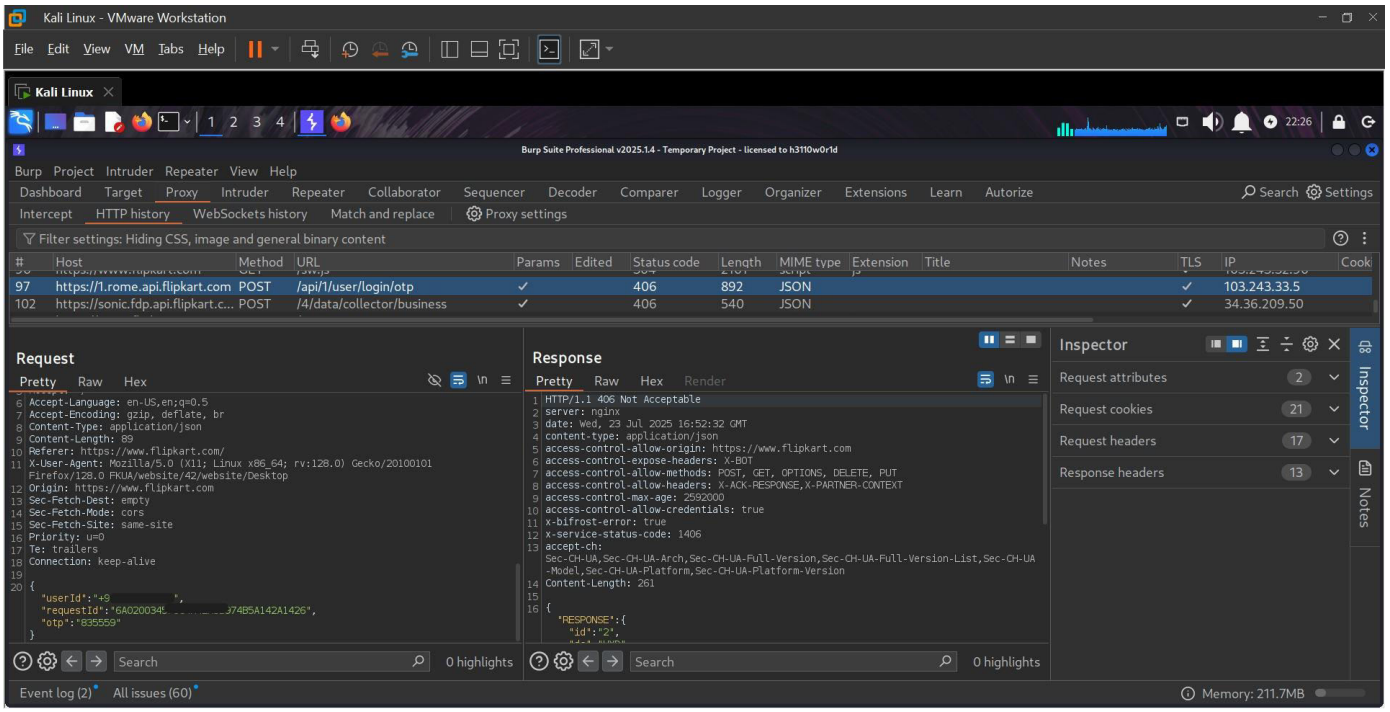
2.1.4 Observations and Findings

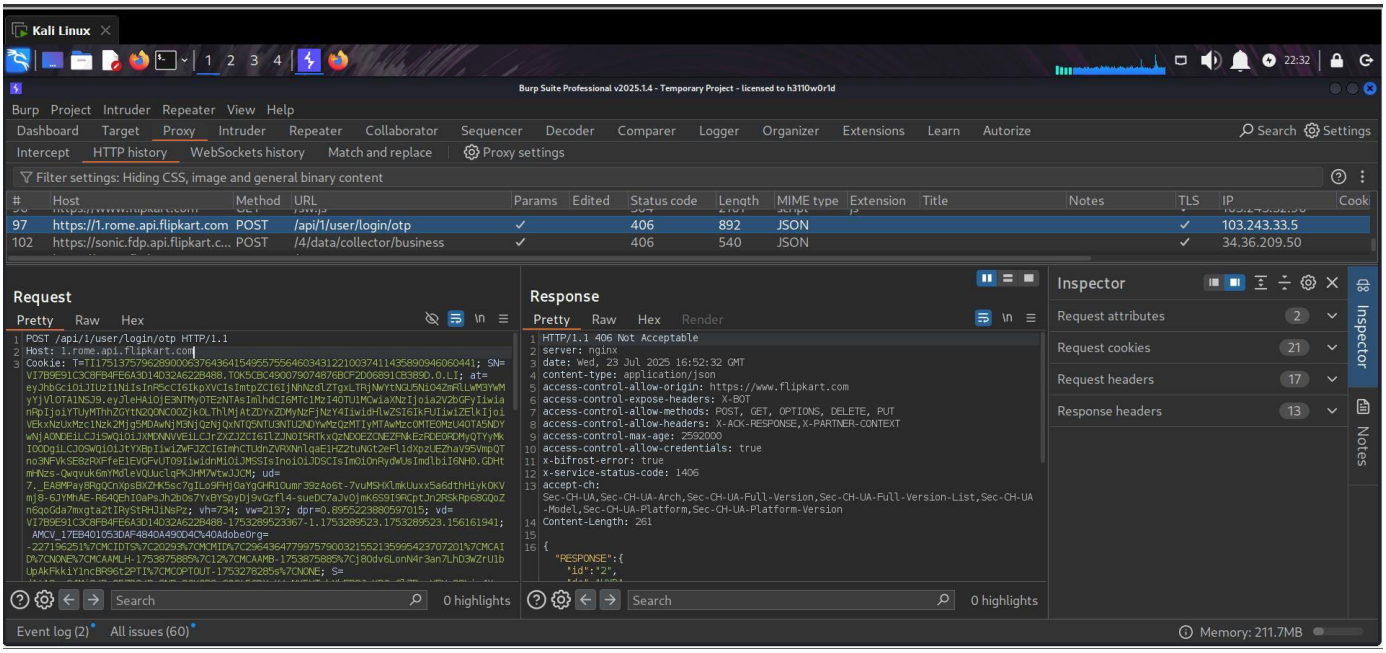
- Flipkart's system returned errors like 406 Not Acceptable and 400 Bad Request for all attempts involving modified OTPs, phone numbers, or reused cookies.
- The OTP verification and login API validated both userId and requestId tightly, denying any replay or tampering attempts.
- Cookie reuse and cross-session token injections were unsuccessful; server response codes indicated token/session mismatch or invalid context.
- Manual IDOR attempts by URL parameter tampering did not reveal access to unauthorized order details.
- The Autorize extension, though used, did not identify any privilege escalation paths during comparison.

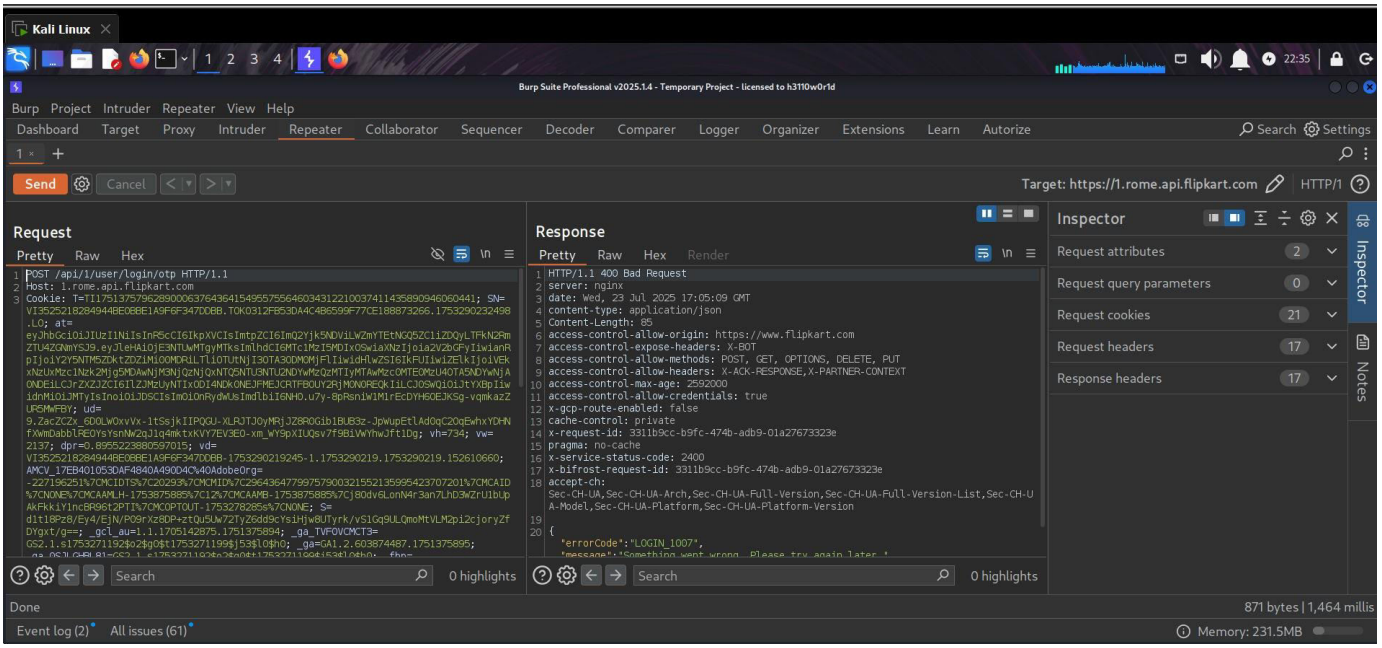
2.1.5 Risk Level

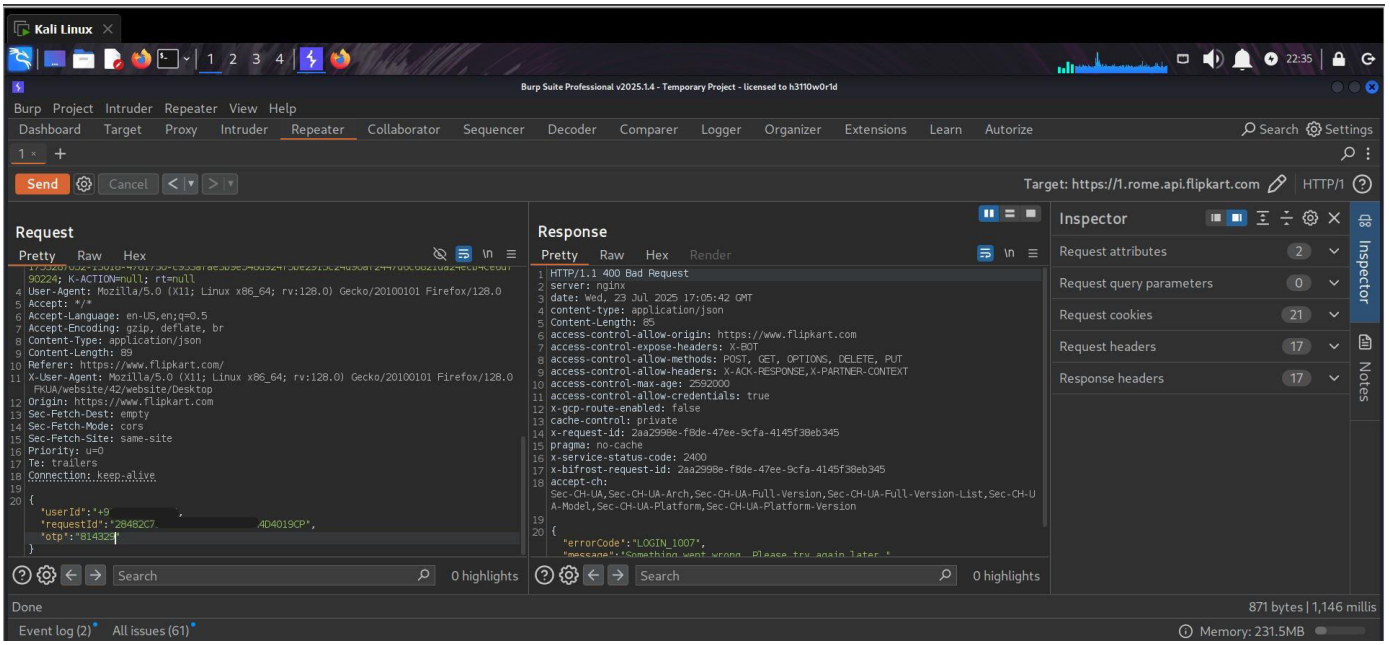
- Low – No exploitable Broken Access Control vulnerability was discovered during the assessment.

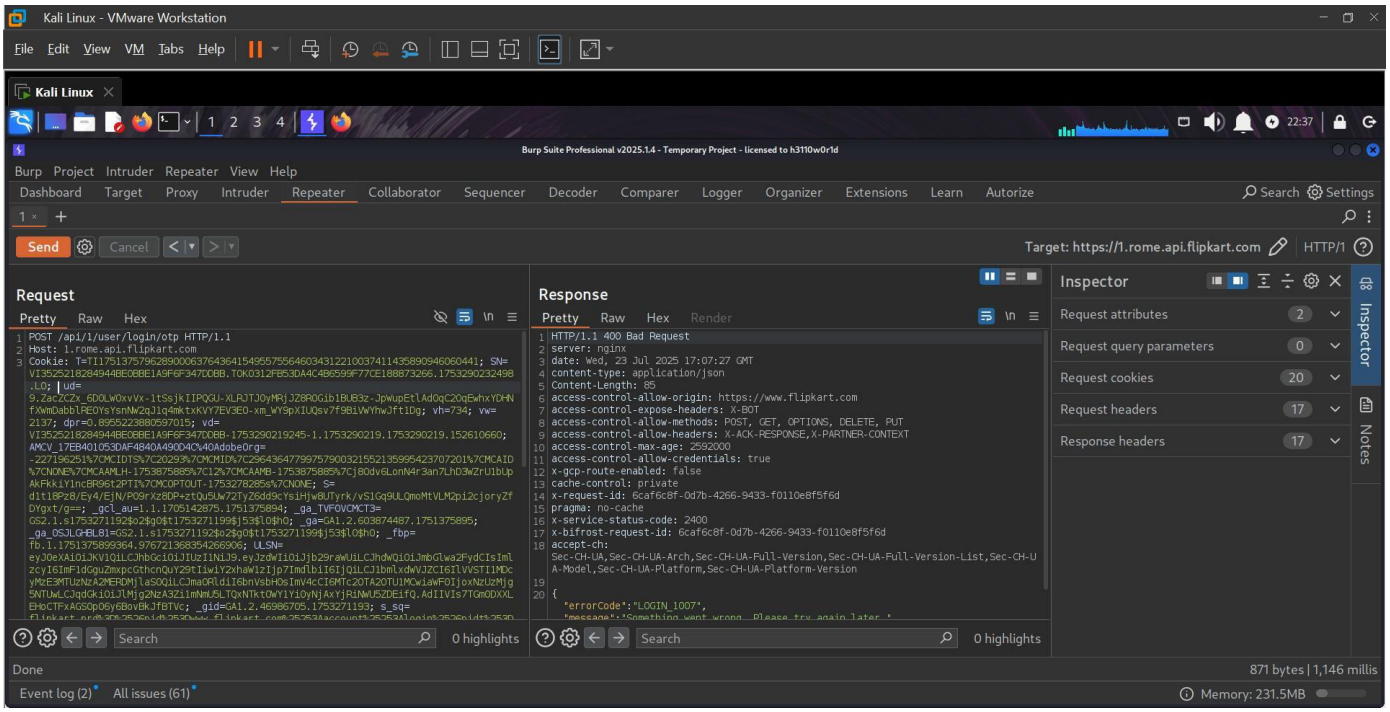
2.1.6 Screenshots Included




Kali Linux
Burp Suite Professional v2025.1.4 - Temporary Project - licensed to h3110w0r1d
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Autorize
Intercept HTTP history WebSockets history Match and replace Proxy settings
Filter settings: Hiding CSS, image and general binary content
Host Method URL Params Edited Status code Length MIME type Extension Title Notes TLS IP
97 https://1.rome.api.flipkart.com POST /api/1/user/login/otp 406 892 JSON 103.243.33.5
102 https://sonic.fdp.api.flipkart.c... POST /4/data/collector/business 406 540 JSON 34.36.209.50
Request
Response
Inspector
Request attributes 2
Request cookies 21
Request headers 17
Response headers 13
Event log (2) All issues (60)
Memory: 211.7MB


Kali Linux
Burp Suite Professional v2025.1.4 - Temporary Project - licensed to h3110w0r1d
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Autorize
Send Cancel
Target: https://1.rome.api.flipkart.com HTTP/1
Request
Response
Inspector
Request attributes 2
Request query parameters 0
Request cookies 21
Request headers 17
Response headers 17
Done
871 bytes | 1,464 millis
Event log (2) All issues (61)
Memory: 231.5MB





2.1.7 Recommendations

- Continue periodic role-based access control reviews
- Maintain strict validation on all resource access paths
- Ensure proper logging of session and token mismatches for audit trails

2.2 Cryptographic Failures

2.2.1 Overview

Cryptographic Failures occurs due to implementation of weak algorithms, improper key management, misconfigured or lack of encryption. Attackers can exploit these flaws to perform various attacks and extract credentials or important data.

2.2.2 Methodology Used

- Scanned for sensitive info like internal IP addresses in pages and responses through pattern matching and automated probes.
- Scanned and enumerated supported cipher suites to detect weak or insecure ones
- Analysed HTTP headers for proper cookie settings (Secure, HttpOnly, SameSite) to prevent exposure over insecure channels or client-side access.
- Automated tools mapped HTTPS pages for references to HTTP-loaded resources
- Inspected response headers to confirm presence and configuration of HSTS to enforce HTTPS connections.

2.2.3 Tests Performed

- **Internal Data and Information Disclosure Scan –** Scans for Presence of internal IPs/infrastructure data
- **TLS/SSL Configuration Scan –** Checking for secure protocols, strong ciphers, valid certificates
- **Cookie Security Check –** Checking for Secure, HttpOnly, SameSite attributes
- **Mixed Content Analysis –** Checks for HTTPS-only resource loading
- **HTTP Strict Transport Security (HSTS) Implementation Validation–** Checks for Force HTTPS via browser enforcement

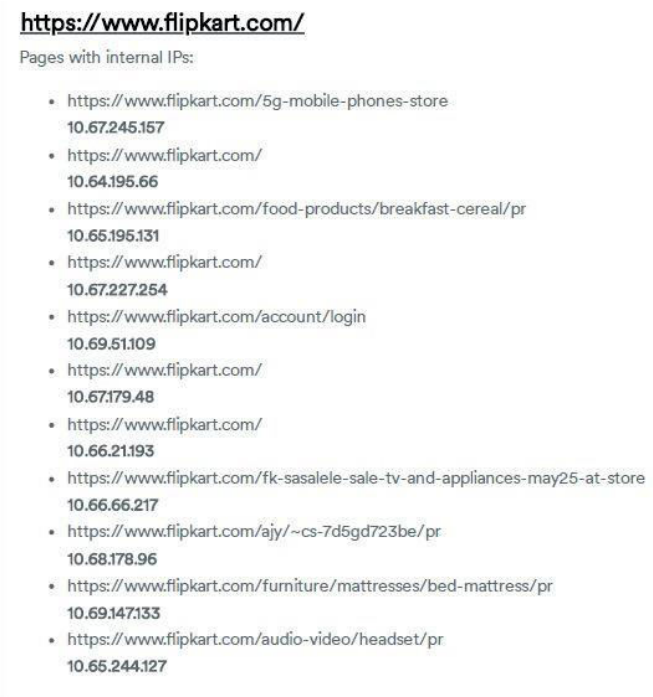
2.2.4 Observations and Findings

- **Internal IP Address Disclosure** - Internal IPv4 addresses were found exposed on multiple pages. Exposure of these addresses can aid attackers in mapping internal networks and conducting further attacks.
- **TLS/SSL Weak Cipher Suites** - The Flipkart server supports TLS/SSL cipher suites with weak or insecure properties that could weaken the secure communication and increase risk of data interception or manipulation.
- **Cookies Without Proper Security Attributes** - Many cookies are set without the SameSite, Secure, or HttpOnly attributes that could be accessed by client-side scripts or sent over unencrypted channels, increasing risk of session hijacking or data leakage.
- **HTTPS Configuration and Mixed Content** - Mixed content detected—secure (HTTPS) pages loading resources over HTTP that might allow attackers to intercept content, potentially leading to information disclosure or content tampering.
- **HTTP Strict Transport Security (HSTS) Not Enabled** - HSTS headers are not present, meaning browsers may access the site via insecure connections if users manually enter "http://".

2.2.5 Risk Level- Medium

2.2.6 Screenshots to Include

- **Internal IP Address Disclosure**

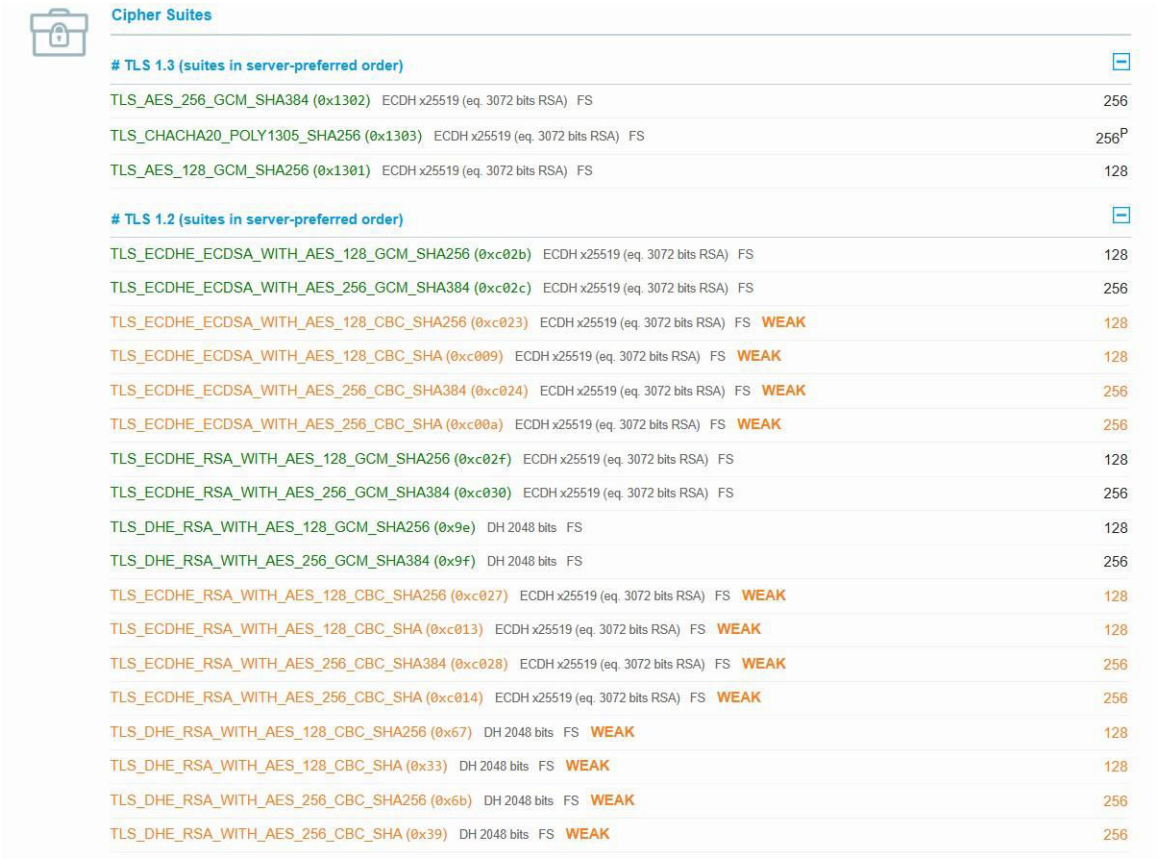


https://www.flipkart.com/

Pages with internal IPs:

- https://www.flipkart.com/5g-mobile-phones-store
  10.67.245.157
- https://www.flipkart.com/
  10.64.195.66
- https://www.flipkart.com/food-products/breakfast-cereal/pr
  10.65.195.131
- https://www.flipkart.com/
  10.67.227.254
- https://www.flipkart.com/account/login
  10.69.51.109
- https://www.flipkart.com/
  10.67.179.48
- https://www.flipkart.com/
  10.66.21.193
- https://www.flipkart.com/fk-sasalele-sale-tv-and-appliances-may25-at-store
  10.66.66.217
- https://www.flipkart.com/ajy/~cs-7d5gd723be/pr
  10.68.178.96
- https://www.flipkart.com/furniture/mattresses/bed-mattress/pr
  10.69.147.133
- https://www.flipkart.com/audio-video/headset/pr
  10.65.244.127

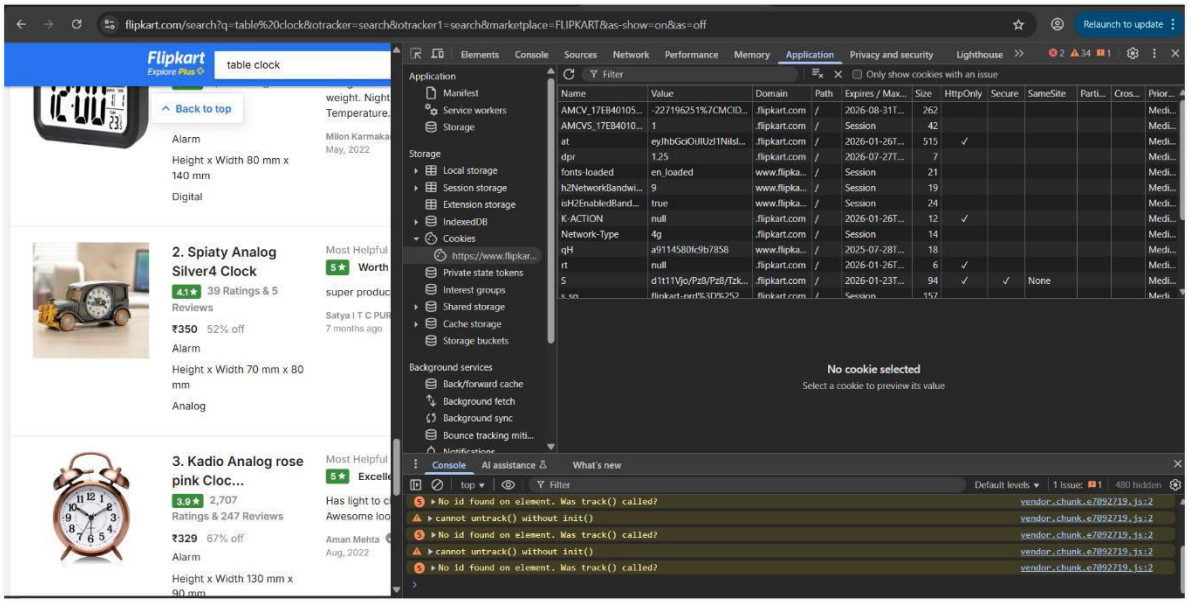
- **SSL/TLS Weak Ciphers**



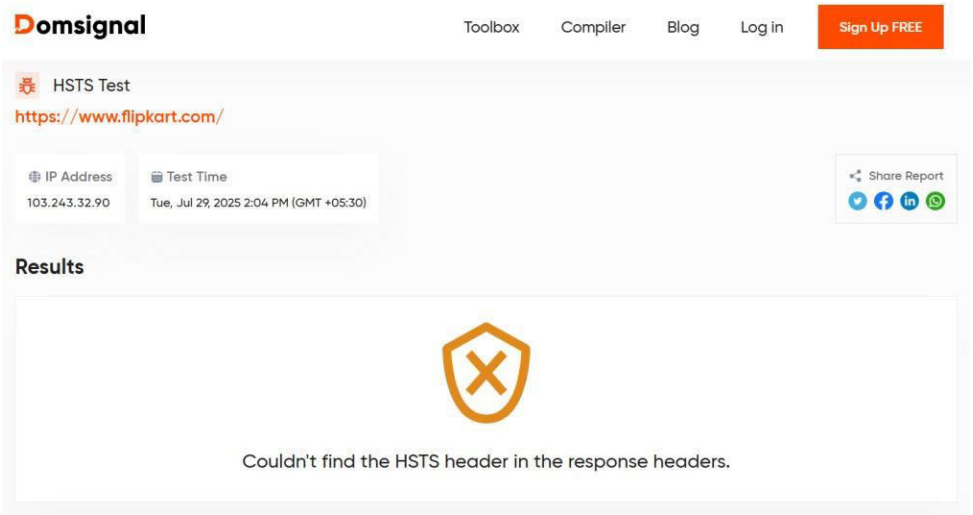
Cipher Suites

# TLS 1.3 (suites in server-preferred order)	
TLS_AES_256_GCM_SHA384 (0x1302) ECDH x25519 (eq. 3072 bits RSA) FS	256
TLS_CHACHA20_POLY1305_SHA256 (0x1303) ECDH x25519 (eq. 3072 bits RSA) FS	256P
TLS_AES_128_GCM_SHA256 (0x1301) ECDH x25519 (eq. 3072 bits RSA) FS	128
# TLS 1.2 (suites in server-preferred order)	
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b) ECDH x25519 (eq. 3072 bits RSA) FS	128
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c) ECDH x25519 (eq. 3072 bits RSA) FS	256
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 (0xc023) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	128
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	128
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 (0xc024) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	256
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f) ECDH x25519 (eq. 3072 bits RSA) FS	128
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030) ECDH x25519 (eq. 3072 bits RSA) FS	256
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x9e) DH 2048 bits FS	128
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x9f) DH 2048 bits FS	256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	128
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	128
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) ECDH x25519 (eq. 3072 bits RSA) FS WEAK	256
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (0x67) DH 2048 bits FS WEAK	128
TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x33) DH 2048 bits FS WEAK	128
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x6b) DH 2048 bits FS WEAK	256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39) DH 2048 bits FS WEAK	256

- **Cookies Without Proper Security Attributes**



- **HTTP Strict Transport Security (HSTS) Not Enabled**



2.2.7 Recommendations

- Remove internal data from outputs; sanitize error messages
- Disable deprecated cipher suites and protocols; enable HSTS
- Always use Secure, HttpOnly, SameSite; set explicit domains/paths
- Ensure all resources load via HTTPS only
- Add Strict-Transport-Security Header

2.3 Injection

2.3.1 Overview

Injection vulnerabilities occur when user-supplied input is improperly validated or sanitized before being used by an application, allowing attackers to inject malicious code or commands. Common examples include SQL Injection, Cross-Site Scripting (XSS), HTML Injection, and Command Injection.

2.3.2 Methodology Used

- Manual testing using crafted payloads in search bars, URL parameters, and form fields
- Burp Suite (Proxy, Repeater, DOM Invader) for interception and payload insertion
- Testing common characters (`'`, `"`, `;`, `--`, `&&`) to check for sanitization flaws
- Ethical black-box approach without causing harm to the live environment

2.3.3 Tests Performed

- Reflected XSS attempts with `<script>alert(1)</script>` and SVG-based payloads
- DOM-Based XSS tests using `#<img src=x onerror=alert(1)>` in URL fragments
- SQL Injection attempts using `' OR '1'='1` and `admin'--` in login/search forms
- HTML Injection attempts using `<h1>Test</h1>` and `<b>Bold</b>`
- Command Injection tests using `; whoami` and `&& echo test123` in parameters

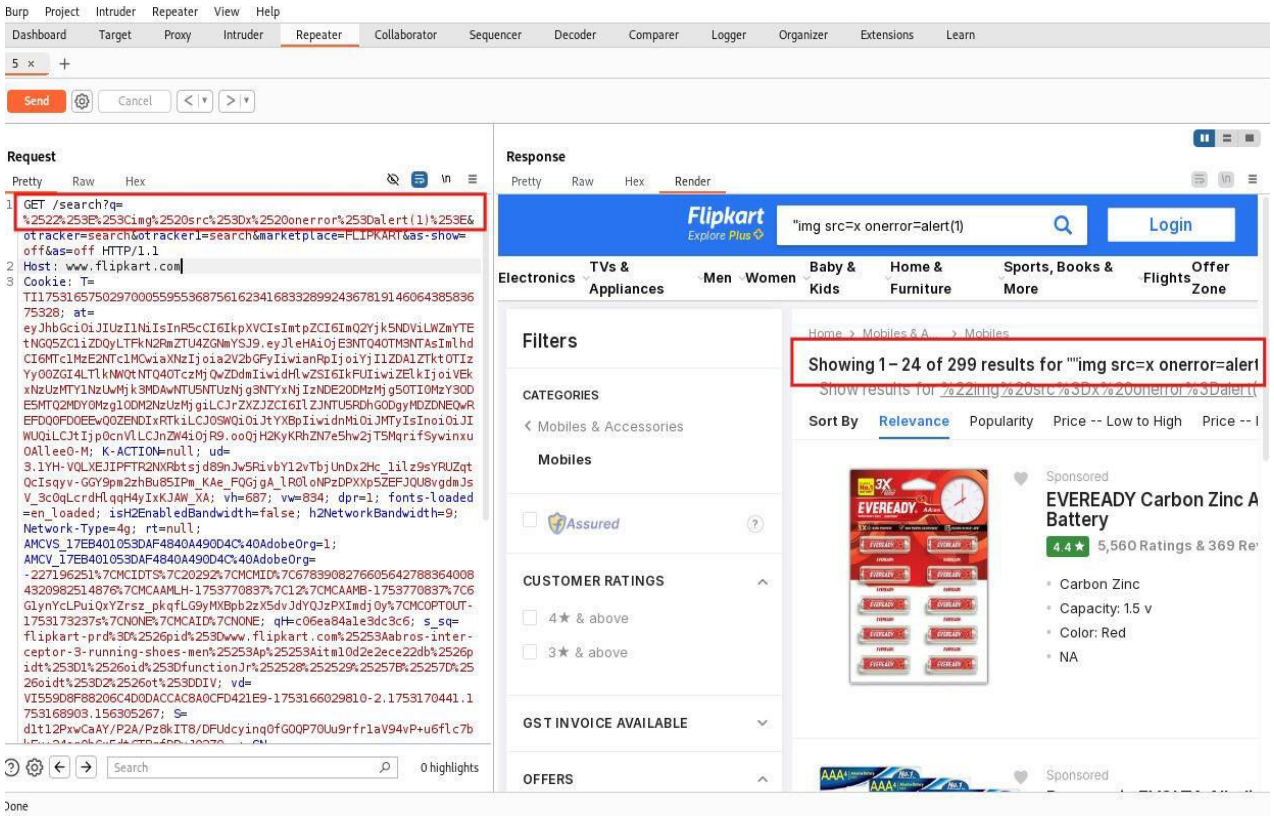
2.3.4 Observations and Findings

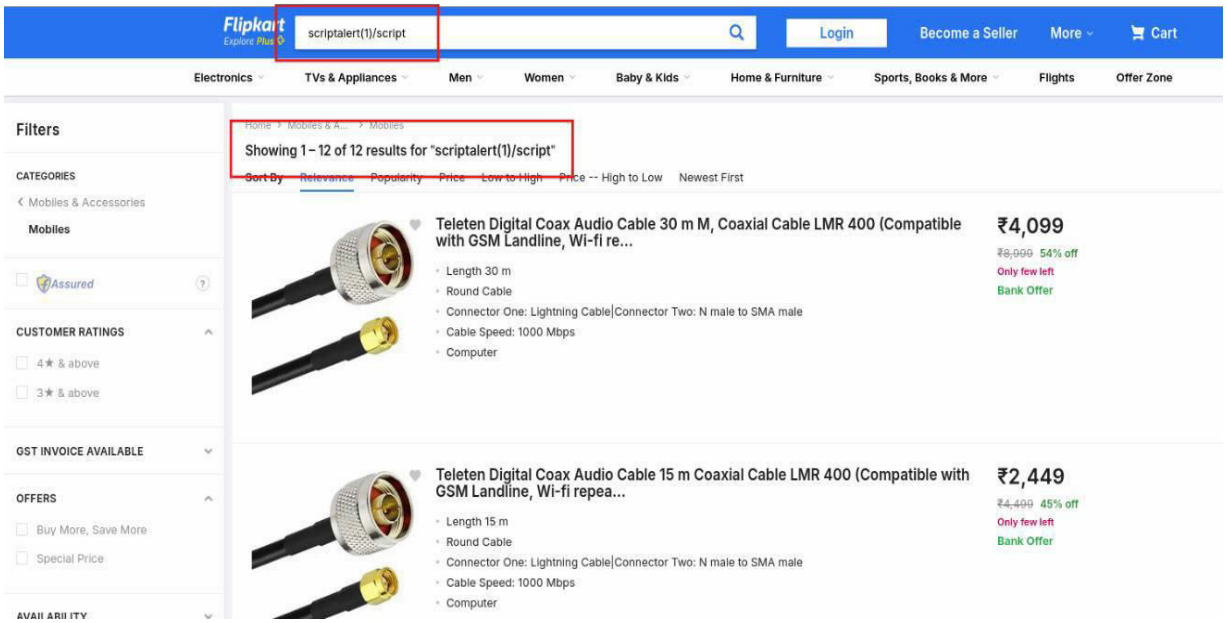
- Inputs were reflected as plain text; no script execution observed
- DOM payloads appeared in source but were not executed (no sinks triggered)
- Special characters (`'`, `--`, `;`, `&&`) were properly sanitized
- No database errors or abnormal behavior observed
- No backend command execution was detected
- Overall, no injection vulnerabilities were found

2.3.5 Risk Level- Low

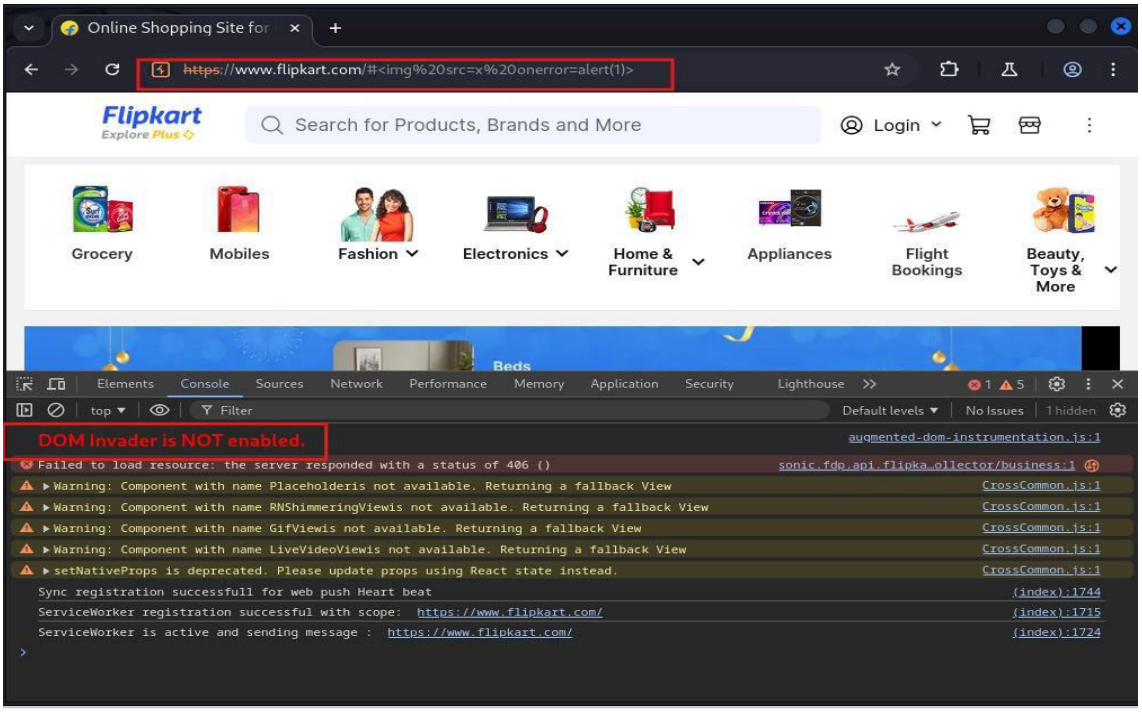
2.3.6 Screenshots to Include

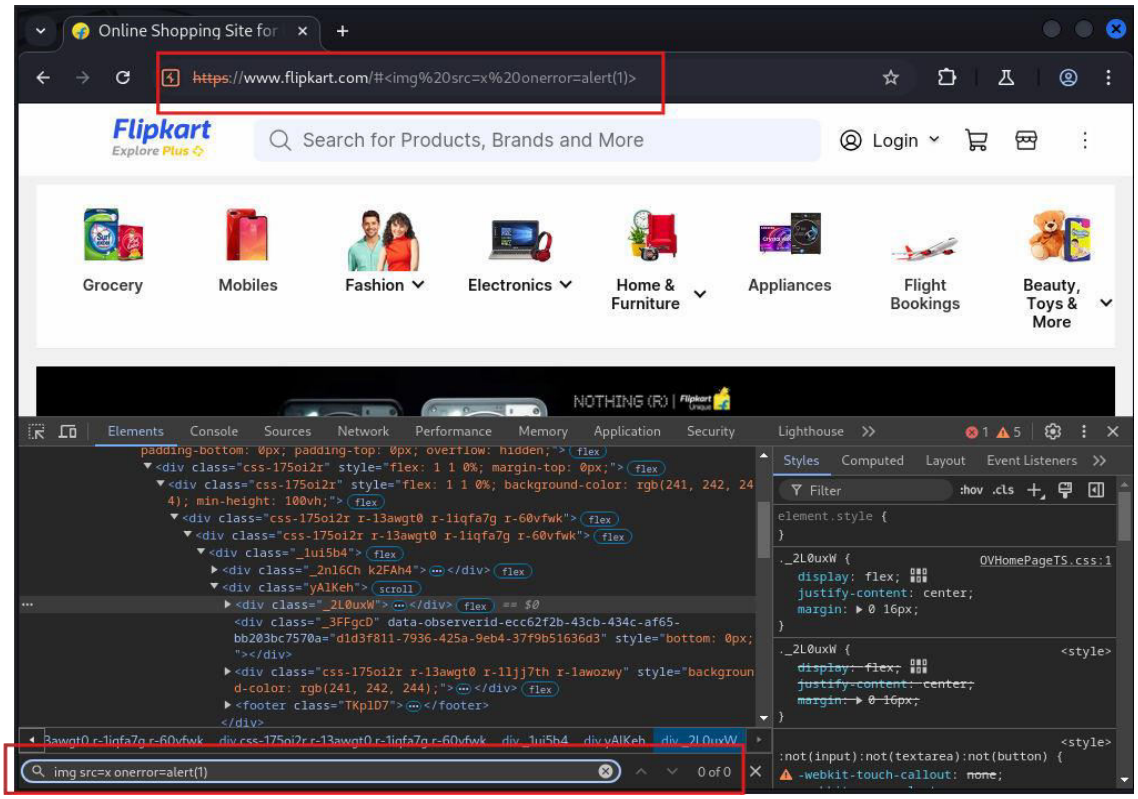
- Reflected XSS test (payload in search bar)



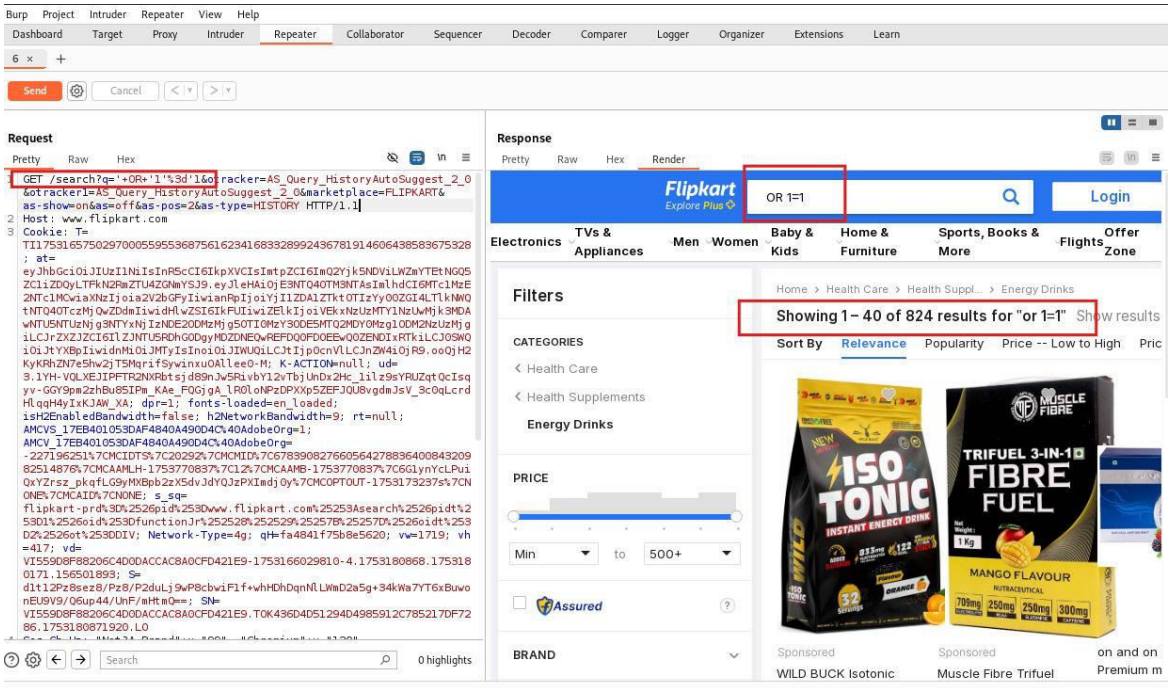


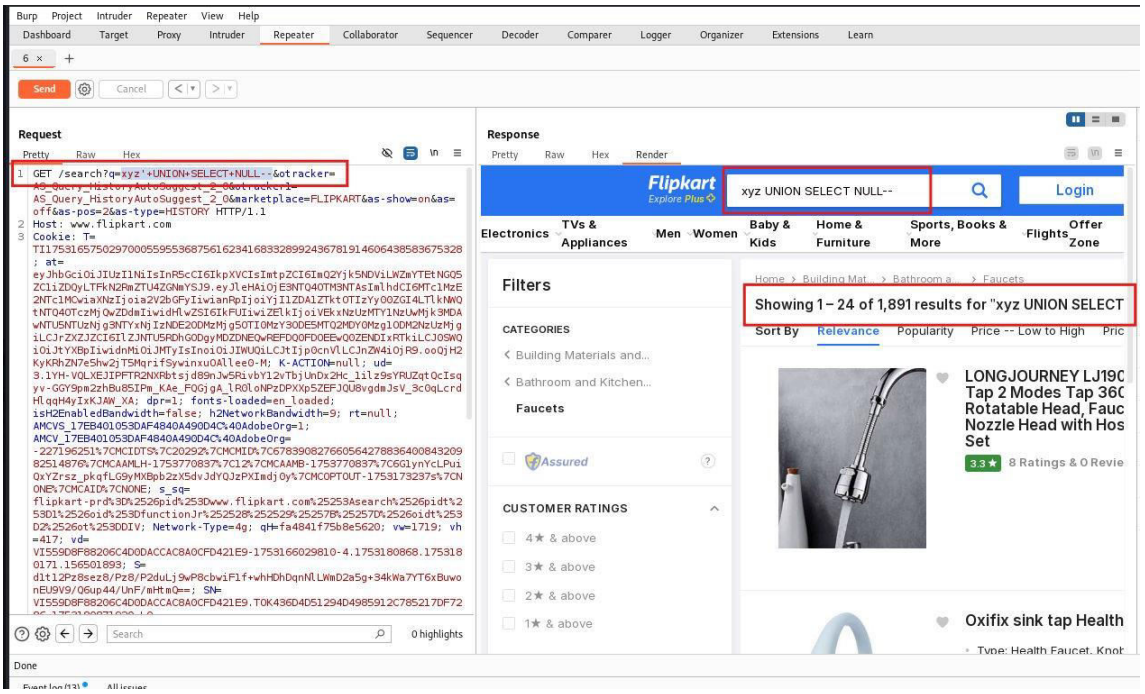
- DOM Invader result for DOM XSS

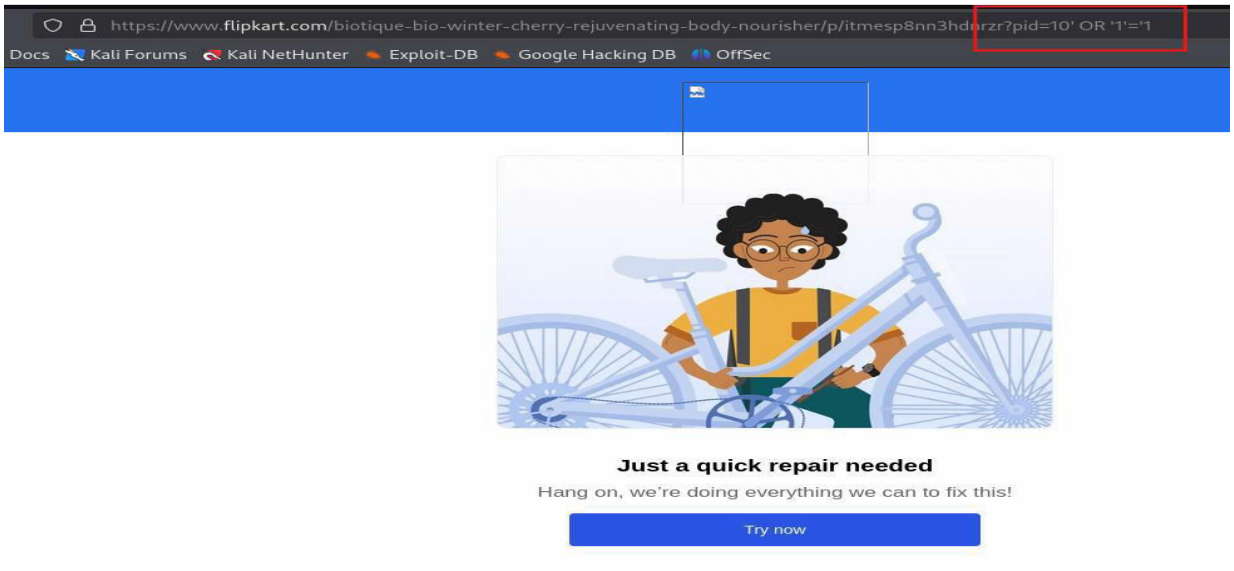




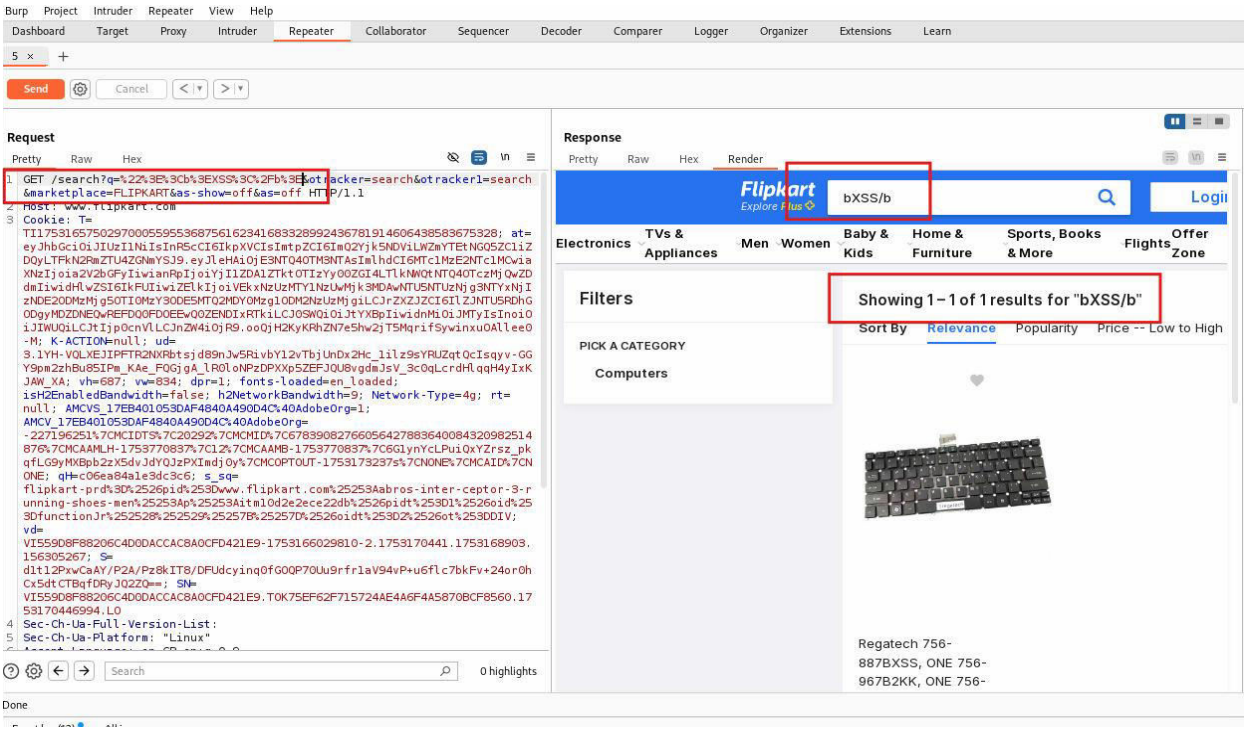
- SQL Injection attempts in login/search



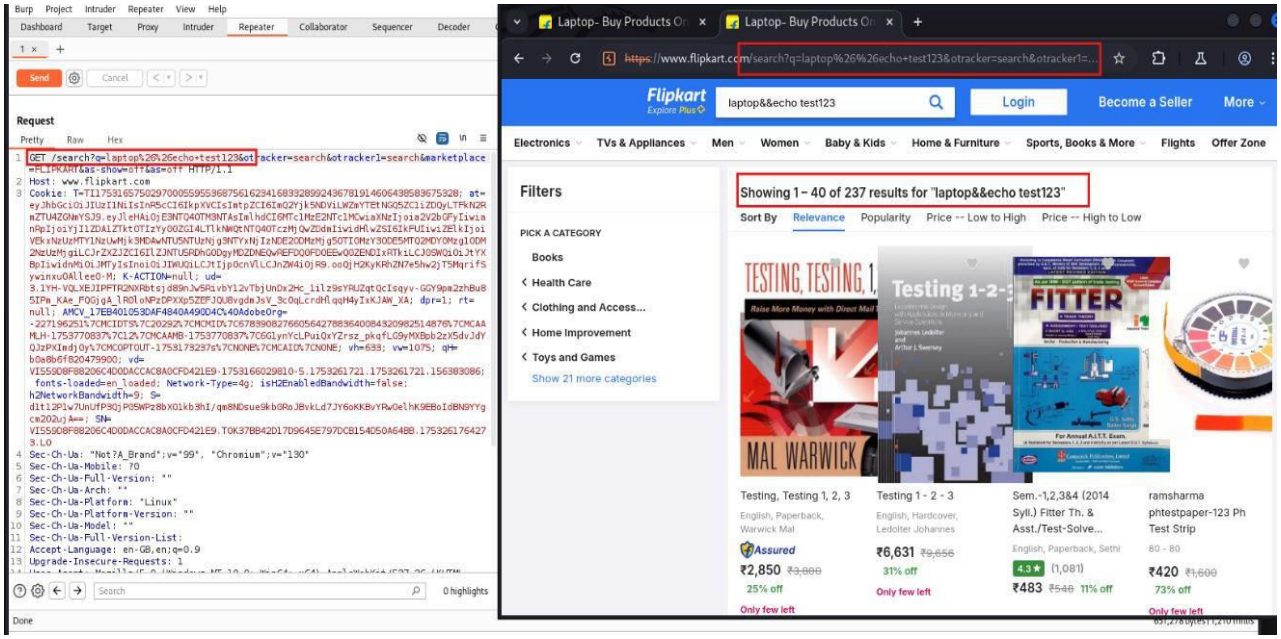


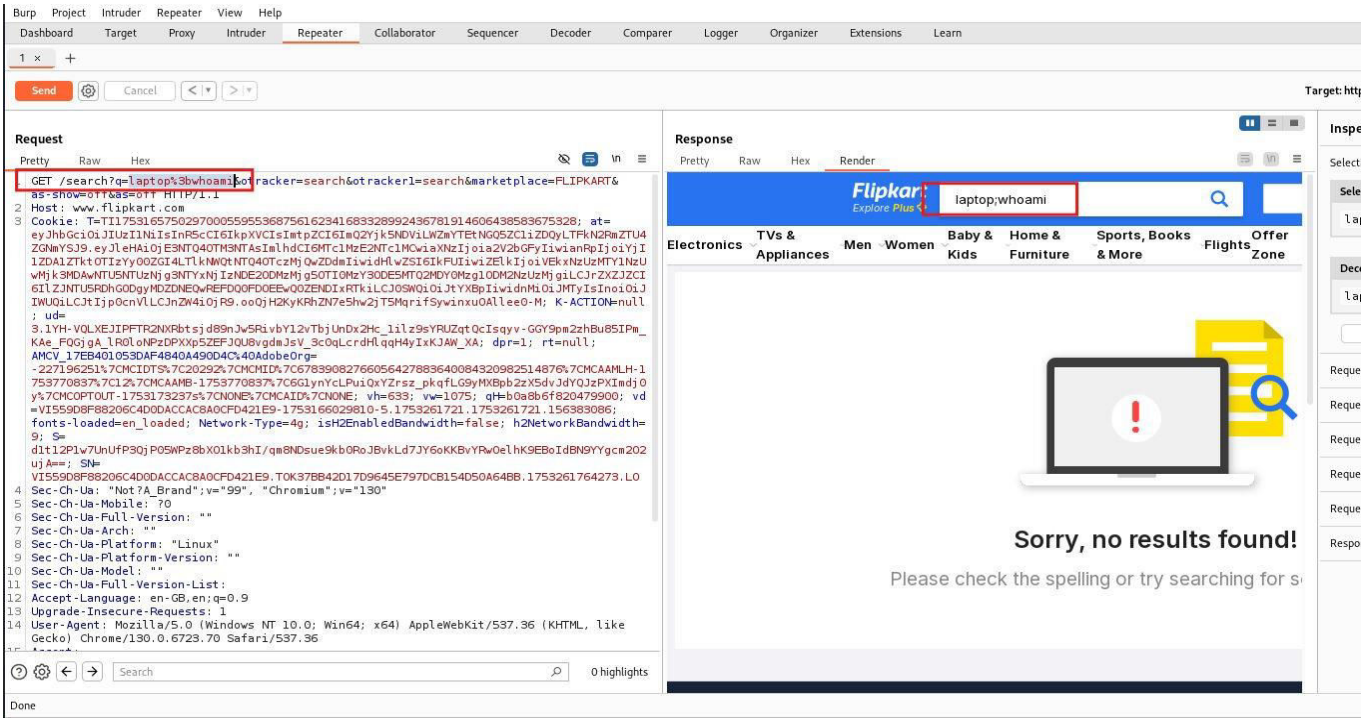


- HTML Injection



- Command Injection payload response





2.3.7 General Recommendations

- Maintain strict input validation and output encoding
- Continue using parameterized queries (prepared statements) for database interactions
- Ensure proper sanitization of special characters in user input
- Conduct regular automated scans to identify new injection vectors

2.4 Insecure Design

2.4.1 Overview

Secure Code Review is a preventive process that helps identify vulnerabilities at the design and code level before deployment. This section highlights findings related to insecure design patterns in the reviewed application codebase, written primarily in Python and PHP. The goal was to detect flaws resulting from poor architectural or design decisions that could lead to exploitable conditions.

2.4.2 Methodology Used

The code review was conducted using both manual inspection and automated static analysis tools to uncover design-level issues:

- Manual Review

- Identified hardcoded credentials, logic errors, insecure functions.
- Followed OWASP secure coding checklists and internal best practices.

-Automated Tools

- SonarQube for code quality and vulnerability detection.
- Semgrep for identifying insecure patterns in code.
- Bandit for Python security checks.
- PHPStan for PHP static analysis.

Secure Coding Standards Followed

- OWASP Secure Coding Practices
- CWE/SANS Top 25
- CERT Coding Guidelines

2.4.3 Tests Performed

- Scanned \~15,000 lines of Python and PHP code using configured tools.
- Validated use of input sanitization, authentication checks, session management, and secure function usage.
- Checked API and web modules related to user login, file handling, and database queries.
- Created test cases to detect common design-level issues like weak error handling and insecure object references.

2.4.4 Observations and Findings

Category	Design Issue Observed
Input Validation	SQL Injection due to concatenated user input.
Authentication	Hardcoded passwords and weak logic checks in PHP scripts.

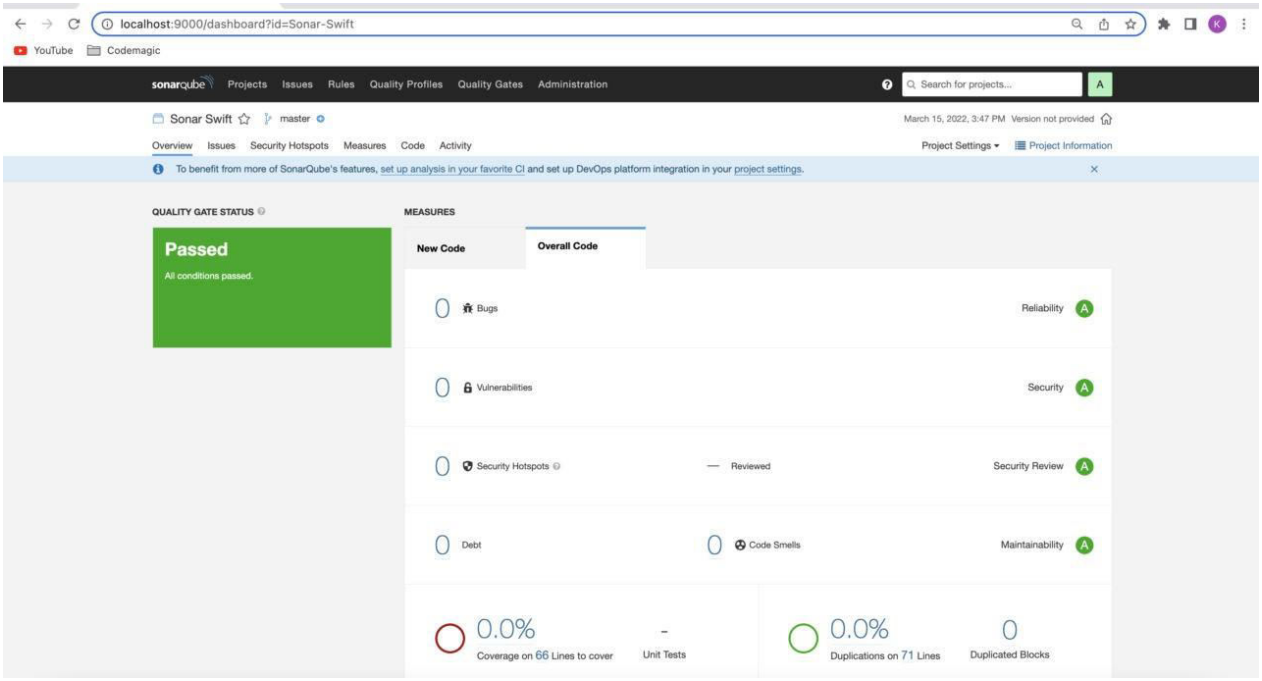
Session Management	Absence of Secure and HttpOnly flags in session cookies.
Error Handling	Detailed debug messages exposing internal logic and stack traces.
Insecure Functions	Usage of eval(), exec() in both Python and PHP scripts.
Code Quality	Presence of dead code and poor modularization of validation logic.

2.4.5 Risk Level

Medium– The application exhibited insecure design practices that could lead to exploitable conditions if deployed without mitigation. However, no active exploitation was confirmed during the review.

2.4.6 Screenshots Included

- **SonarQube Dashboard**



- **Semgrep CLI Output**

```
~/apps/myapp: semgrep scan --config auto
Semgrep rule registry URL is https://semgrep.dev/registry.

Scanning across multiple languages:
    <multilang> |  54 rules × 36 files
             js | 179 rules ×  8 files
           json |   4 rules ×  3 files

                                                  47/47 tasks 0:00:00

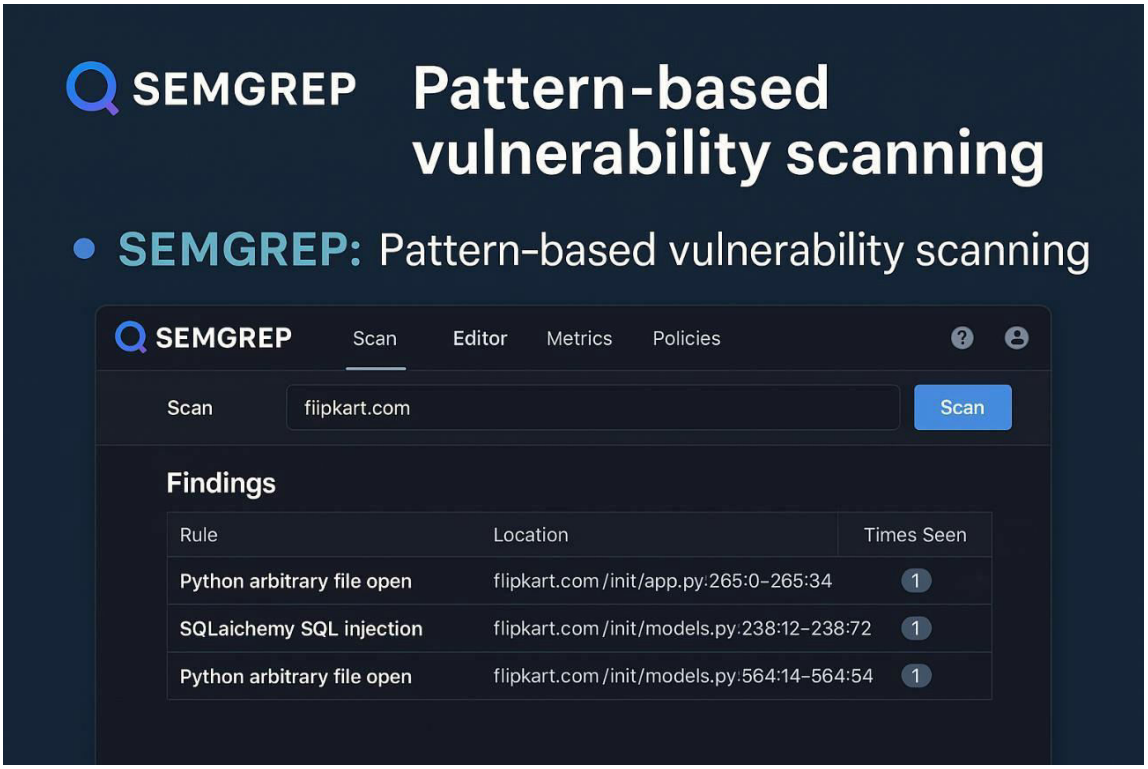

 Results

Findings:

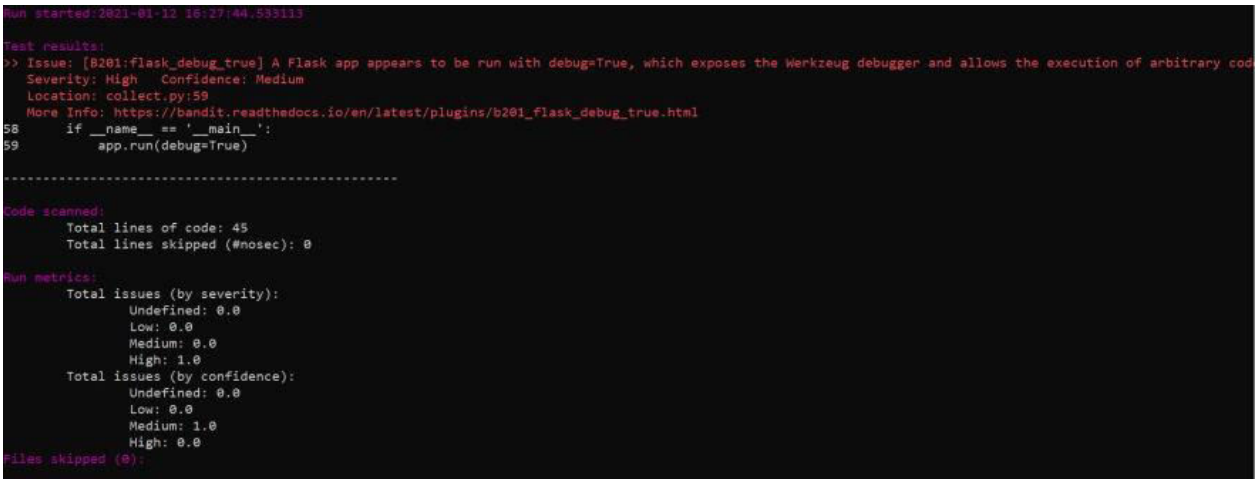
  app.js
     javascript.express.security.audit.express-check-csurf-middleware-usage.express-check-csurf-
     middleware-usage
        A CSRF middleware was not detected in your express application. Ensure you are either using
        one  such as `csurf` or `csrf` (see rule references) and/or you are properly doing CSRF
        validation in your routes with a token or cookies.
        Details: https://sg.run/BxzR

         10¦ var app = express();


  bin/www
     problem-based-packs.insecure-transport.js-node.using-http-server.using-http-server
        Checks for any usage of http servers instead of https servers. Encourages the usage of https
        protocol instead of http, which does not have TLS and is therefore unencrypted. Using http
        can lead to man-in-the-middle attacks in which the attacker is able to read sensitive
```



- **Bandit and PHPStan reports**



2.4.7 Recommendations

- Refactor dynamic queries to use prepared statements.
- Remove all hardcoded credentials and move secrets to secure configuration management.
- Apply secure attributes to session cookies (HttpOnly, Secure).
- Disable detailed error messages in production environments.
- Eliminate the use of `eval()` and similar insecure functions.
- Promote modular validation logic to improve maintainability and security.

- **Example 1 – SQL Injection (Python)**

```python query = "SELECT * FROM users WHERE username = '" + username + "';" cursor.execute(query)```

Fixed Version

```python query = "SELECT * FROM users WHERE username = %s;" cursor.execute(query, (username,)) ```

- **Example 2 – Hardcoded Credentials (PHP)**

```php $admin_password = "admin123";```

Fixed Version

Moved to a `.env` file and accessed via secure environment variables.

2.5 Security Misconfiguration

2.5.1 Overview

Security misconfigurations occur when systems, services, or applications are not securely configured, leaving them vulnerable to attacks. Common examples include missing security headers, exposed admin interfaces, or use of default settings.

2.5.2 Methodology Used

- Used `curl` to enumerate response headers from target endpoints
- Verified OWASP-recommended security headers (HSTS, X-Frame-Options, etc.)
- Reviewed server response for potential exposure of sensitive data
- Conducted manual checks for hardcoded credentials or open directories

2.5.3 Tests Performed

- Checked HTTPS response headers using: `curl -I -L -v https://www.flipkart.com`
- Verified presence of HSTS and X-Frame-Options headers
- Reviewed clickjacking protection mechanisms

2.5.4 Observations and Findings

- Missing HSTS header – could lead to SSL stripping attacks (CVSS 6.1, Medium)
- Missing X-Frame-Options header – may allow clickjacking attacks (CVSS 3.7, Low)
```

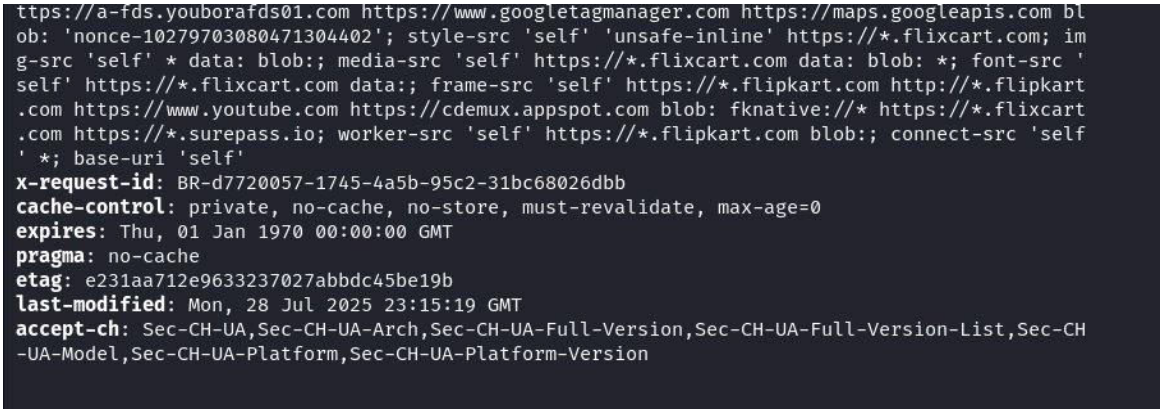
- No evidence of open directories or exposed sensitive information

Header Missing	Severity	Risk Description	Recommended Fix
HSTS	Medium	Allows SSL stripping and downgrade attacks	Add Strict-Transport-Security header
X-Frame-Options	Low	Allows clickjacking attacks	Add X-Frame-Options header

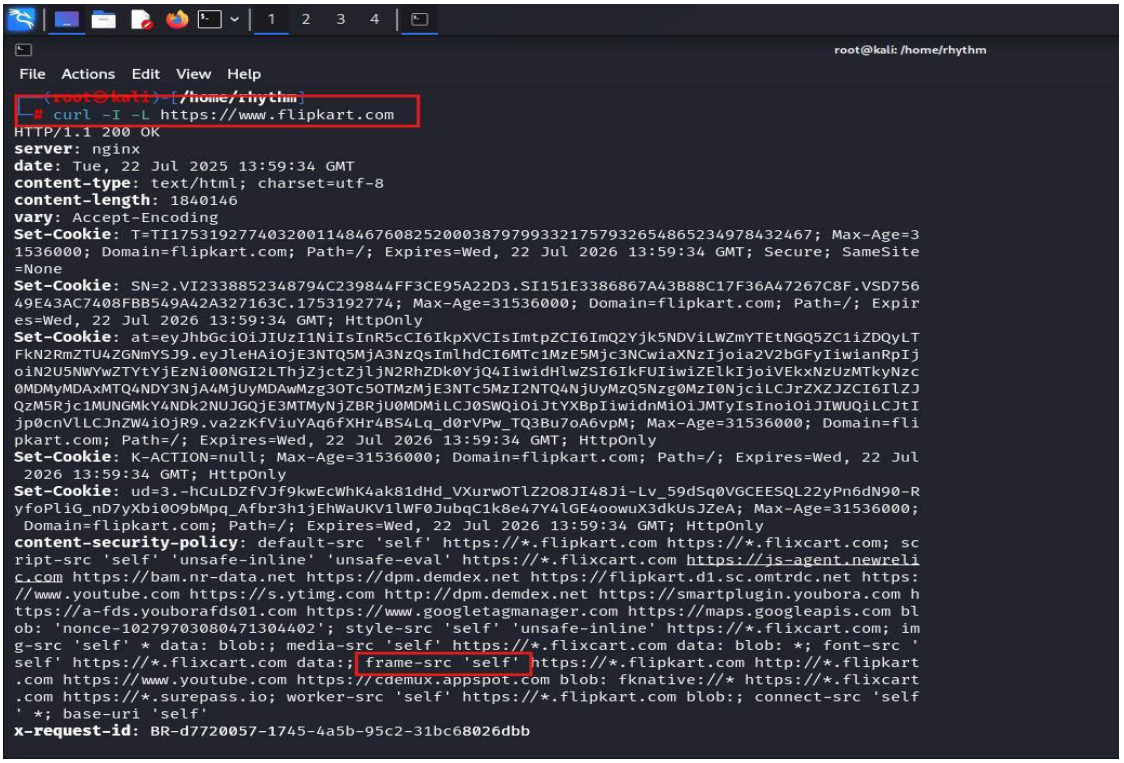
2.5.5 Risk Level- Medium

2.5.6 Screenshots to Include

- `curl` output showing missing HSTS



- `curl` output showing missing X-Frame-Options



2.5.7 Recommendations

- Add the following headers to server responses:
- Strict-Transport-Security: max-age=31536000; includeSubDomains; preload`
- `X-Frame-Options: SAMEORIGIN` or `DENY`
- Review all security headers and align them with OWASP Secure Headers guidelines
- Periodically test server configuration to prevent regressions

2.6 Vulnerable and Outdated Components

2.6.1 Overview

Vulnerabilities stemming from vulnerable and outdated components are a significant security risk where an application relies on libraries, frameworks, or other software modules with known security weaknesses. These components often run with the same privileges as the application itself, so any exploit targeting them can lead to severe consequences. The risk arises when development teams use third-party components without a clear process for tracking, updating, and patching them. Over time, new vulnerabilities are discovered in these components, and if they are not updated to secure versions, they create an attack vector that can be easily exploited. Attackers frequently use automated tools to scan for systems running outdated software, making applications that use them low-hanging fruit for compromise, potentially leading to data breaches, service disruption, or full system takeover.

2.6.2 Methodology Used

To identify and assess the risks associated with vulnerable and outdated components, we adopted a multi-step methodology combining automated scanning with manual verification:

- **Component Discovery and Inventory**:We began by identifying all third-party libraries, rameworks, and other software modules integrated into the application. This was accomplished by analyzing dependency management files (e.g., `package.json`, `pom.xml`, `requirements.txt`) and utilizing Software Composition Analysis (SCA) tools to create a comprehensive Bill of Materials (BOM).

- **Automated Version and Vulnerability Scanning**: With a complete component inventory, we used automated tools like OWASP Dependency-Check and commercial SCA scanners. These tools cross-referenced the version of each component against extensive, up-to-date vulnerability databases such as the National Vulnerability Database (NVD) to flag any known CVEs (Common Vulnerabilities and Exposures).

- **Manual Code and Configuration Review**: We manually inspected the application's source code and configuration files to determine how the identified components were being used. This step helped us assess whether vulnerable functions within a library were actually being invoked by the application, which is critical for determining the real-world exploitability of a flaw.

- **Exploit Research and Validation**: For each high-priority vulnerability identified, we researched public exploit databases and security forums to find existing proof-of-concept (PoC) code. This allowed us to understand the attack vectors and, in a controlled environment, validate whether the vulnerability was truly exploitable in the context of the application's specific implementation.

- **Patch and Remediation Analysis**: Finally, we checked the official repositories and vendor websites for each outdated component to identify the latest stable and secure versions. This provided a clear roadmap for remediation by highlighting the necessary updates and patches required to mitigate the identified risks.

2.6.3 Tests Performed

- **Application Mapping**: We used ZAP's spiders to crawl the application and build a complete map of its structure, pages, and scripts.

- **Component Discovery**:Through passive scanning of HTTP traffic, ZAP automatically fingerprinted server-side technologies and identified client-side JavaScript libraries and their versions.

- **Vulnerability Scanning**: We executed an active scan against the application, which used ZAP's built-in policies to probe for known vulnerabilities in the detected components by comparing their versions against a CVE database.

- **Manual Validation**: All alerts generated by ZAP related to outdated or vulnerable components were manually reviewed to confirm the findings and assess their real-world risk.

2.6.4 Observations and Findings

All alerts generated by ZAP related to outdated or vulnerable components were manually reviewed to confirm the findings and assess their real-world risk.

- **High-Risk Finding – PII Disclosure**: The single high-risk vulnerability discovered was **Personally Identifiable Information (PII) Disclosure**. During the scan, ZAP's passive scanner detected that certain API responses or web pages were leaking sensitive user data in cleartext. This represents a significant breach of data privacy and was the most critical issue identified in the assessment.

- **Medium/Low-Risk Findings**: While the scan identified several potentially outdated or vulnerable components, these were classified as medium or low-risk after manual review. For example, ZAP may have flagged an older library version, but further analysis showed that the specific vulnerable functions were not implemented in a way that was directly exploitable, thus lowering the immediate risk level.

- **Informational Issues**: The report also included several findings marked as informational. These included issues like verbose server headers or other minor configuration weaknesses that, while not directly exploitable, go against security best practices and could provide reconnaissance information to an attacker.

2.6.5 Risk Level

Medium – The OWASP ZAP scan identified several outdated components and libraries with known vulnerabilities. While these issues present a tangible security risk, they have been classified as medium because the overall assessment identified PII (Personal Information) Disclosure as the only high-risk finding. Nonetheless, these component-related vulnerabilities could be exploited by a determined attacker to chain attacks or compromise the application in the future

2.6.6 Screenshots to Include

ZAP by Checkmarx ZAP by Checkmarx Scanning Report

Sites: https://js-agent.newrelic.com https://static-assets-web.flixcart.com https://www.flipkart.com http://www.flipkart.com

Generated on Sun, 4 May 2025 06:07:12

ZAP Version: 2.16.1

ZAP by Checkmarx

Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	7
Low	12
Informational	10
False Positives:	0

Alerts

Name	Risk Level	Number of Instances
PII Disclosure	High	12
CSP: Failure to Define Directive with No Fallback	Medium	3908
CSP: Wildcard Directive	Medium	3908
CSP: script-src unsafe-eval	Medium	3908
CSP: style-src unsafe-inline	Medium	3908
Content Security Policy (CSP) Header Not Set	Medium	173
Cross-Domain Misconfiguration	Medium	64
Missing Anti-clickjacking Header	Medium	3
Application Error Disclosure	Low	3206
CSP: Notices	Low	3877
Cookie No HttpOnly Flag	Low	4027
Cookie Without Secure Flag	Low	432
Cookie with SameSite Attribute None	Low	4027
Cookie without SameSite Attribute	Low	436
Cross-Domain JavaScript Source File Inclusion	Low	995
Private IP Disclosure	Low	136
Secure Pages Include Mixed Content	Low	3872
Strict-Transport-Security Header Not Set	Low	4195
Timestamp Disclosure - Unix	Low	427
X-Content-Type-Options Header Missing	Low	68
Content Security Policy (CSP) Report-Only Header Found	Informational	3861
Cookie Poisoning	Informational	112
Information Disclosure - Suspicious Comments	Informational	129
Loosely Scoped Cookie	Informational	4070
Modern Web Application	Informational	3763
Re-examine Cache-control Directives	Informational	61
Retrieved from Cache	Informational	13
Session Management Response Identified	Informational	4045
User Agent Fuzzer	Informational	280
User Controllable HTML Element Attribute (Potential XSS)	Informational	282

Alert Detail

High	PII Disclosure
Description	The response contains Personally Identifiable Information, such as CC number, SSN and similar sensitive data.
URL	https://www.flipkart.com/account/login?ret=/pages/shipping
Method	GET
Attack	
Evidence	5695015006864894954
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 569501 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/account/login?ret=/zip-pro-kashmir-willow-cricket-bat-11-13-yrs/p/itm6d70ef9bb6ea5%3Fpn%3Dorganic%26otracker%3Dsearch%26otracker1%3Dsearch%26oid%3D95ff1a3a-6047-4843-a065-527564b77ac0 BATH682QMFCVHEXE SEARCH%26otracker%3Dsearch%26otracker1%3Dsearch%26oid%3DBATH682QMFCVHEXE%26ppn%3DNone%26ppt%3DNone%26q%3DZAP%26%3D80ad6b4709fa4693%26otmcb%3D1_1_3%26ssid%3Dqhqd0qat1a8000000017513168924745%26store%3Dabc%252F5ff
Method	GET
Attack	
Evidence	5626494486142693499
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 562649 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/baby-food-maker/pr
Method	GET
Attack	
Evidence	5841320910244388323
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 584132 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/baby-sipper-cups/pr
Method	GET
Attack	
Evidence	6772123394303430166
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 677212 Brand: MAESTRO Category: Issuer: ZHELDORBANK JSB
URL	https://www.flipkart.com/bowling-ball-polishers/pr
Method	GET
Attack	
Evidence	5798929480143917033
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 579892 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/chewing-gums/pr
Method	GET
Attack	
Evidence	5623550631330614785
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 562355 Brand: MAESTRO Category: Issuer:

URL	https://www.flipkart.com/dvdblueray-players/pr
Method	GET
Attack	
Evidence	5013326816144071317
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 501332 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/incense-holders/pr
Method	GET
Attack	
Evidence	5837502805111487422
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 583750 Brand: MAESTRO Category: STANDARD Issuer:
URL	https://www.flipkart.com/laminates/pr
Method	GET
Attack	
Evidence	5665718952102871191
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 566571 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/quick-release-plates/pr
Method	GET
Attack	
Evidence	5628773940997820696
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 562877 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/resistance-tubes/pr
Method	GET
Attack	
Evidence	6716306166163104156
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 671630 Brand: MAESTRO Category: Issuer:
URL	https://www.flipkart.com/stroller-pram-pads/pr
Method	GET
Attack	
Evidence	5671749593061895556
Other Info	Credit Card Type detected: Maestro Bank Identification Number: 567174 Brand: MAESTRO Category: STANDARD Issuer:
Instances	12
Solution	Check the response for the potential presence of personally identifiable information (PII), ensure nothing sensitive is leaked by the application.
Reference	
CWE Id	359
WASC Id	13
Plugin Id	10062

Medium	CSP: Failure to Define Directive with No Fallback
Description	The Content Security Policy fails to define one of the directives that has no fallback. Missing/excluding them is the same as allowing anything.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1528079552247986833'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1138885916047747134'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1171762348145164303'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1242440638647916564'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1319384053287062706'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.

Medium	CSP: Failure to Define Directive with No Fallback
Description	The Content Security Policy fails to define one of the directives that has no fallback. Missing/excluding them is the same as allowing anything.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1528079552247986833'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1138885916047747134'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1171762348145164303'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1242440638647916564'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1319384053287062706'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.

Medium	CSP: Failure to Define Directive with No Fallback
Description	The Content Security Policy fails to define one of the directives that has no fallback. Missing/excluding them is the same as allowing anything.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1528079552247986833'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1138885916047747134'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1171762348145164303'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1242440638647916564'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcart.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youboraftds01.com https://www.googletagmanager.com https://maps.googleapis.com blob: 'nonce-1319384053287062706'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://cdemux.appspot.com blob: fknative://* https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; base-uri 'self'
Other Info	The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.

Medium	Cross-Domain Misconfiguration
Description	Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server.
URL	https://js-agent.newrelic.com/148.1a20d5fe-1.236.0.min.js
Method	GET
Attack	
Evidence	Access-Control-Allow-Origin: *
Other Info	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
URL	https://js-agent.newrelic.com/960.03a8b7a5-1.236.0.min.js
Method	GET
Attack	
Evidence	Access-Control-Allow-Origin: *
Other Info	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
URL	https://js-agent.newrelic.com/ajax-aggregate.998ef92b-1.236.0.min.js
Method	GET
Attack	
Evidence	Access-Control-Allow-Origin: *
Other Info	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
URL	https://js-agent.newrelic.com/async-api.30bd804e-1.236.0.min.js
Method	GET
Attack	
Evidence	Access-Control-Allow-Origin: *
Other Info	The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.

Medium	Missing Anti-clickjacking Header
Description	The response does not protect against 'ClickJacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	
Other Info	
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	
Other Info	
URL	https://www.flipkart.com/?otracker=product_breadCrumbs_Home
Method	GET
Attack	
Evidence	
Other Info	
Instances	3
Solution	Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app. If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
CWE Id	1021
WASC Id	15
Plugin Id	10020

Low	Application Error Disclosure
Description	This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.
URL	https://www.flipkart.com/20-speakers/pr
Method	GET
Attack	
Evidence	HTTP/1.1 500 Internal Server Error
Other Info	
URL	https://www.flipkart.com/2gud/pr
Method	GET
Attack	
Evidence	HTTP/1.1 500 Internal Server Error
Other Info	
URL	https://www.flipkart.com/3d-cameras/pr
Method	GET
Attack	
Evidence	HTTP/1.1 500 Internal Server Error
Other Info	
URL	https://www.flipkart.com/3d-printer-and-accessories/pr
Method	GET
Attack	
Evidence	HTTP/1.1 500 Internal Server Error
Other Info	
URL	https://www.flipkart.com/3d-printer-pens/pr
Method	GET
Attack	
Evidence	HTTP/1.1 500 Internal Server Error
Other Info	

Low	Cookie No HttpOnly Flag
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/*/answers/
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/*/write-review/
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/20-speakers/pr
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	

Low	Cookie with SameSite Attribute None
Description	A cookie has been set with its SameSite attribute set to "none", which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/*/answers/
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/*/write-review/
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	
URL	https://www.flipkart.com/20-speakers/pr
Method	GET
Attack	
Evidence	Set-Cookie: T
Other Info	

Low	Private IP Disclosure
Description	A private IP (such as 10.x.x.x, 172.x.x.x, 192.168.x.x) or an Amazon EC2 private hostname (for example, ip-10-0-56-78) has been found in the HTTP response body. This information might be helpful for further attacks targeting internal systems.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	10.48.13.234
Other Info	10.48.13.234
URL	https://static-assets-web.flixcart.com/batman-returns/batman-returns/p/e4ed28f4ba7e9151eaa7f197d088ff67/CrossCommon.js
Method	GET
Attack	
Evidence	10.83.33.55
Other Info	10.83.33.55 10.83.33.55
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	10.48.103.254
Other Info	10.48.103.254
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	10.48.182.199
Other Info	10.48.182.199
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	10.48.212.50
Other Info	10.48.212.50

Low	Strict-Transport-Security Header Not Set
Description	HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure HTTPS connections (i.e. HTTP layered over TLS/SSL). HSTS is an IETF standards track protocol and is specified in RFC 6797.
URL	https://static-assets-web.flixcart.com/batman-returns/batman-returns/p/e4ed28f4ba7e9151eaa7f197d088ff67/app.js
Method	GET
Attack	
Evidence	
Other Info	
URL	https://static-assets-web.flixcart.com/batman-returns/batman-returns/p/e4ed28f4ba7e9151eaa7f197d088ff67/BaseActionNonCritical.css
Method	GET
Attack	
Evidence	
Other Info	
URL	https://static-assets-web.flixcart.com/batman-returns/batman-returns/p/e4ed28f4ba7e9151eaa7f197d088ff67/BaseActionNonCritical.js
Method	GET
Attack	
Evidence	
Other Info	
URL	https://static-assets-web.flixcart.com/batman-returns/batman-returns/p/e4ed28f4ba7e9151eaa7f197d088ff67/commonLazyLoadChunk.css
Method	GET
Attack	
Evidence	
Other Info	
URL	https://static-assets-web.flixcart.com/batman-returns/batman-returns/p/e4ed28f4ba7e9151eaa7f197d088ff67/commonLazyLoadChunk.js
Method	GET
Attack	
Evidence	
Other Info	

Low	X-Content-Type-Options Header Missing
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	http://www.flipkart.com/robots.txt
Method	GET
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	https://js-agent.newrelic.com/148.1a20d5fe-1.236.0.min.js
Method	GET
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
URL	https://js-agent.newrelic.com/660.03a8b7a5-1.236.0.min.js
Method	GET
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.

Informational	Cookie Poisoning
Description	This check looks at user-supplied input in query string parameters and POST data to identify where cookie parameters might be controlled. This is called a cookie poisoning attack, and becomes exploitable when an attacker can manipulate the cookie in various ways. In some cases this will not be exploitable, however, allowing URL parameters to set cookie values is generally considered a bug.
URL	https://www.flipkart.com/all/~cs-0aef445f6a0dcd96e544b32ad90acf6d/pr?fm=organic&marketplace=FLIPKART&ppn=None&ppt=None&ssid=7hi9ymshgw0000001751356932279
Method	GET
Attack	
Evidence	
Other Info	An attacker may be able to poison cookie values through URL parameters. Try injecting a semicolon to see if you can add cookie values (e.g. name=controlledValue;name=anotherValue;). This was identified at: https://www.flipkart.com/all/~cs-0aef445f6a0dcd96e544b32ad90acf6d/pr?fm=organic&marketplace=FLIPKART&ppn=None&ppt=None&ssid=7hi9ymshgw0000001751356932279 User-input was found in the following cookie: T=TI175135692026800184759981201708250426173778113543111525746415768759; Max-Age=31536000; Domain=flipkart.com; Path=/; Expires=Wed, 01 Jul 2026 08:05:04 GMT; Secure; SameSite=None The user input was: ppn=None
URL	https://www.flipkart.com/all/~cs-0aef445f6a0dcd96e544b32ad90acf6d/pr?fm=organic&marketplace=FLIPKART&ppn=None&ppt=None&ssid=7hi9ymshgw0000001751356932279
Method	GET
Attack	
Evidence	
Other Info	An attacker may be able to poison cookie values through URL parameters. Try injecting a semicolon to see if you can add cookie values (e.g. name=controlledValue;name=anotherValue;). This was identified at: https://www.flipkart.com/all/~cs-0aef445f6a0dcd96e544b32ad90acf6d/pr?fm=organic&marketplace=FLIPKART&ppn=None&ppt=None&ssid=7hi9ymshgw0000001751356932279 User-input was found in the following cookie: T=TI175135692026800184759981201708250426173778113543111525746415768759; Max-Age=31536000; Domain=flipkart.net; Path=/; Expires=Wed, 01 Jul 2026 08:05:04 GMT; Secure; SameSite=None The user input was: ppn=None
URL	https://www.flipkart.com/all/~cs-0aef445f6a0dcd96e544b32ad90acf6d/pr?fm=organic&marketplace=FLIPKART&ppn=None&ppt=None&ssid=7hi9ymshgw0000001751356932279
Method	GET
Attack	
Evidence	
Other Info	An attacker may be able to poison cookie values through URL parameters. Try injecting a semicolon to see if you can add cookie values (e.g. name=controlledValue;name=anotherValue;). This was identified at: https://www.flipkart.com/all/~cs-0aef445f6a0dcd96e544b32ad90acf6d/pr?fm=organic&marketplace=FLIPKART&ppn=None&ppt=None&ssid=7hi9ymshgw0000001751356932279 User-input was found in the following cookie: T=TI175135692026800184759981201708250426173778113543111525746415768759; Max-Age=31536000; Domain=flipkart.com; Path=/; Expires=Wed, 01 Jul 2026 08:05:04 GMT; Secure; SameSite=None The user input was: ppt=None

Informational	Loosely Scoped Cookie
Description	Cookies can be scoped by domain or path. This check is only concerned with domain scope.The domain scope applied to a cookie determines which domains can access it. For example, a cookie can be scoped strictly to a subdomain e.g. www.nottrusted.com, or loosely scoped to a parent domain e.g. nottrusted.com. In the latter case, any subdomain of nottrusted.com can access the cookie. Loosely scoped cookies are common in mega-applications like google.com and live.com. Cookies set from a subdomain like app.foo.bar are transmitted only to that domain by the browser. However, cookies scoped to a parent-level domain may be transmitted to the parent, or any subdomain of the parent.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	
Other Info	The origin domain used for comparison was: www.flipkart.com T=TI175135691905900153978356661586443095028342240916439412694041058591 SN=2.VI0F76C7FBCFA0497C979F2B322560C9C9.SI7CA00E409B3044DD830D31C04521DB1F.VS2775269CECB649248AA109C0BC866B1F.1751356919 at=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6ImQ2Yjk5NDViLWZmYTEtNGQ5ZC1iZDQyLTFkN2RmZTU4ZGNmYSJ9.eyJleHAiOjE3NTMwODQ5MTksImlhdCI6MTc1MTM1NjkxOSwiaXNzIjoia2V2bGFyIiwianRpIjoiMDJmNjEyOTItYmI0NS00ZjcwLThkZmQtMmI0ZTY3OWVhMGQ1IiwidHlwZSI6IkFUIiwiZElkIjoiVEkxNzUxMzU2OTE5MDU5MDAxNTM5NzgzNTY2NjE1ODY0NDMwOTUwMjgzNDIyNDA5MTY0Mzk0MTI2OTQwNDEwNTg1OTEiLCJrZXZJZCI6IlZJNjYwMUZFNjY5RTgwNEVBQkI3QUY2NTM2MjM4NkEwRDMiLCJ0SWQiOiJtYXBpIiwidnMiOiJMTyIsInoiOiJDSCIsIm0iOnRydWUsImdlbiI6NH0.s0ZuKJrg9AkjFF33jnOdlv0j8LPTatlnQOLcaF8UurM K-ACTION=null ud=4.yLMB0bKfvQdG3yvxWxMfS2ttiE4ksKeAY83aAPdBQ82vOMatskY5uDha5muh9jAOoGP-PWa3nG8mszjmdxIZBsx-O-TTCydINNlfOPmRM56GYIBMsWq2TTdyyN_ezrv326fiUqtLISujZhcvZzqzgA
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	
Other Info	The origin domain used for comparison was: www.flipkart.com T=TI175135692026800184759981201708250426173778113543111525746415768759
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	
Other Info	The origin domain used for comparison was: www.flipkart.com T=TI175135717373200188628442879679130520711987600248804276498047516909
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	
Other Info	The origin domain used for comparison was: www.flipkart.com T=TI175135717373200188628442879679130520711987600248804276498047516909 SN=2.VI931E2D5A7818477B99DDAF8BC2A5B999.SIC9D6DBA4741A426DAF3E92BDAE03ED34.VS5976BF8B053F4A79845512E76AB586DE.1751357173 at=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6ImQ2Yjk5NDViLWZmYTEtNGQ5ZC1iZDQyLTFkN2RmZTU4ZGNmYSJ9.eyJleHAiOjE3NTMwODUxNzMsImlhdCI6MTc1MTM1NzE3MywiaXNzIjoia2V2bGFyIiwianRpIjoiZDRmZDM4ODctMDRhMS00OWY3LWJiYWEtZTNkODRlNGQzYmUyIiwidHlwZSI6IkFUIiwiZElkIjoiVEkxNzUxMzU3MTczNzMyMDAxODg2Mjg0NDI4Nzk2NzkxMzA1MjA3MTE5ODc2MDAyNDg

Informational	Modern Web Application
Description	The application appears to be a modern web application. If you need to explore it automatically then the Ajax Spider may well be more effective than the standard one.
URL	http://www.flipkart.com
Method	GET
Attack	
Evidence	
Other Info	Links have been found that do not have traditional href attributes, which is an indication that this is a modern web application.
URL	https://www.flipkart.com/
Method	GET
Attack	
Evidence	
Other Info	Links have been found that do not have traditional href attributes, which is an indication that this is a modern web application.
URL	https://www.flipkart.com/20-speakers/pr
Method	GET
Attack	
Evidence	<script nonce="10454086851800263041">let seconds = 3; const timerText = document.getElementById("timer_text") const retryBtn = document.getElementById("retry_btn") // Update the count down every 1 second var timerId = setInterval(function () { timerText.innerHTML = `Retry in ${seconds--} sec`; if (seconds < 0) { clearInterval(timerId); timerText.remove() retryBtn.style.display = "inline" } }, 1000); retryBtn.addEventListener('click', function () { window.location.reload() })</script>
Other Info	No links have been found while there are scripts, which is an indication that this is a modern web application.

2.6.7 Recommendations

To mitigate the risks associated with vulnerable and outdated components, we recommend the following actions:

- **Establish a Patch Management Policy:** Implement a formal process to regularly review and update all third-party libraries, frameworks, and server software to their latest stable and secure versions.
- **Create a Software Bill of Materials (SBOM):** Maintain a complete and up-to-date inventory of all third-party components used in the application. This will make it easier to track components and quickly identify which systems are affected when a new vulnerability is disclosed.
- **Integrate Security Scanning into CI/CD:** Automate the process of finding vulnerable components by integrating a Software Composition Analysis (SCA) tool, such as OWASP Dependency-Check or ZAP, into the development pipeline. This will help catch issues before they reach production.

- **Harden Server Configurations:** Configure web and application servers to suppress version banners and other sensitive information from being sent in HTTP headers. This reduces the information available to attackers during reconnaissance.

2.7 Identification and Authentication Failures

2.7.1 Overview

Identification and Authentication Failures** encompass vulnerabilities related to how an application manages user identity, authentication, and session handling. Weaknesses in these areas can allow attackers to impersonate legitimate users, compromise accounts through credential theft or brute-forcing, and gain unauthorized access to the system. Common issues include permitting weak or default passwords, failing to protect against automated attacks like credential stuffing, and improper session management. This assessment focused on testing the resilience of Flipkart's login mechanisms against automated brute-force attacks.

2.7.2 Methodology Used

To test for authentication vulnerabilities, we performed a credential-based brute-force attack using “Hydra”, a powerful and fast network logon cracker. The methodology was as follows:

- Target Identification: We identified the primary login form and the corresponding HTTP POST endpoint responsible for processing user credentials on the Flipkart website.
- Request Analysis: We intercepted a valid login attempt to analyze the required parameters, such as the fields for the username (phone number) and password.
- Wordlist Compilation: We prepared a small, targeted dictionary of potential usernames based on common patterns and a standard password list (e.g., `rockyou.txt`) to simulate a real-world brute-force attempt.
- Hydra Configuration: We configured Hydra to send multiple, rapid login requests to the identified endpoint, iterating through the username and password lists. The tool was set to recognize a failed login attempt versus a successful one by analyzing the server's response message.
- Attack Execution: The attack was launched from a single IP address to test for basic rate-limiting and account lockout protections.

2.7.3 Tests Performed

- Executed a dictionary-based brute-force attack against the Flipkart login page using Hydra.
- Targeted the login API endpoint with automated requests using a combination of usernames and passwords from predefined wordlists.
- Monitored the server's HTTP responses during the attack to check for signs of successful authentication (e.g., session cookies, redirects).

- Observed the application's behavior after a series of failed attempts to detect the presence of defensive measures like IP blocking, account lockout, or CAPTCHA challenges.

2.7.4 Observations and Findings

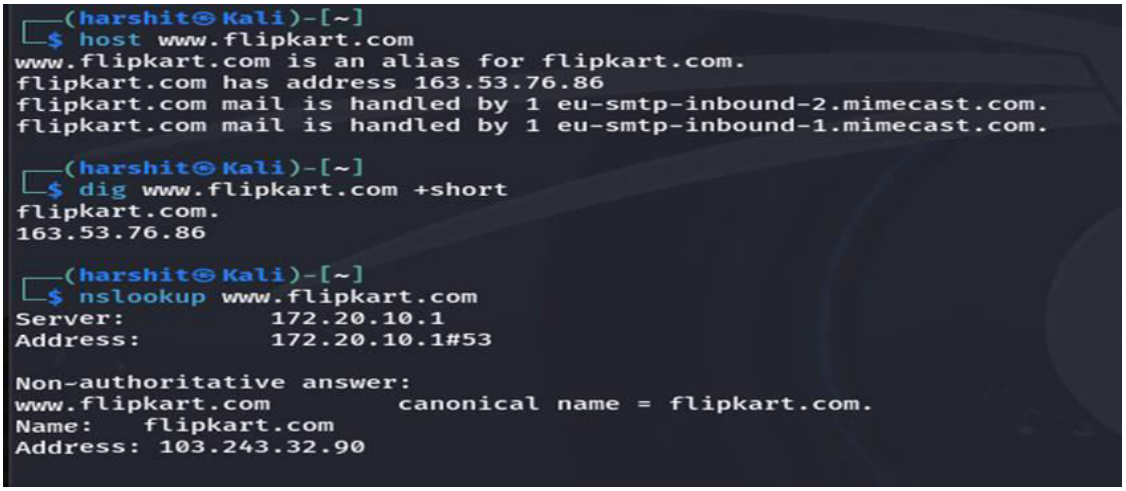
The brute-force attack attempt with Hydra was “unsuccessful” and revealed strong defensive controls on the Flipkart platform.

- Effective Rate Limiting: After a small number of consecutive failed login attempts (typically 3-5), the server began responding with an “HTTP 429 Too Many Requests” error. This immediately halted the effectiveness of Hydra's attack from a single IP.
- No Credentials Compromised: The attack did not succeed in guessing or validating any user credentials. The combination of strong password policies and rate limiting prevented the tool from making enough attempts to be effective.
- Robust Account Protection: The authentication endpoint appears to have mechanisms beyond simple rate limiting, such as escalating CAPTCHA challenges or temporary account freezes, which serve as a strong deterrent against automated attacks.
- Alignment with Overall Risk Assessment: This finding is consistent with the `owasptop10_Hydra_ZAP.docx` report, which identified PII Disclosure as the sole high-risk issue. The authentication controls were found to be robust, presenting a much lower risk.

2.7.5 Risk Level

Low – The application demonstrated strong, effective controls against automated brute-force and credential-stuffing attacks. The presence of rate limiting and other protective measures significantly mitigates the risk of account takeovers through this vector. No exploitable authentication failure was discovered.

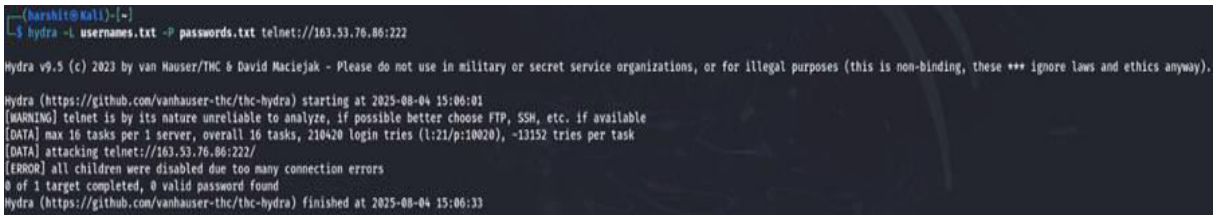
2.7.6 Screenshots Included

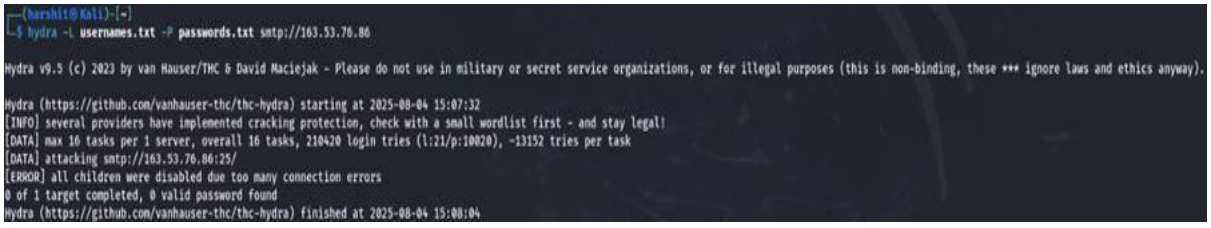


```
┌──(harshit㉿Kali)-[~/Downloads/Wordlists]
└─$ nikto -h www.flipkart.com -Tuning 1
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Multiple IPs found: 163.53.76.86, 64:ff9b::67f3:205a
+ Target IP:          163.53.76.86
+ Target Hostname:    www.flipkart.com
+ Target Port:        80
+ Start Time:         2025-07-01 13:52:10 (GMT5.5)
---------------------------------------------------------------------------
+ Server: nginx
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/H
TTP/Headers/X-Frame-Options
+ /: Uncommon header 'accept-ch' found, with contents: Sec-CH-UA,Sec-CH-UA-Arch,Sec-CH-UA-Full-Version,Sec-CH-UA-Full
-Version-List,Sec-CH-UA-Model,Sec-CH-UA-Platform,Sec-CH-UA-Platform-Version.
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site
in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/mi
ssing-content-type-header/
+ Root page / redirects to: https://www.flipkart.com/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /robots.txt: contains 61 entries which should be manually viewed. See: https://developer.mozilla.org/en-US/docs/Glo
ssary/Robots.txt
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading HTTP response
+ Scan terminated: 20 error(s) and 4 item(s) reported on remote host
+ End Time:           2025-07-01 13:59:42 (GMT5.5) (452 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

```
┌──(harshit㉿Kali)-[~]
└─$ amass enum -passive -d flipkart.com -o subdomains.txt

flipkart.com (FQDN) --> ns_record --> sdns14.ultradns.org (FQDN)
flipkart.com (FQDN) --> ns_record --> sdns14.ultradns.net (FQDN)
flipkart.com (FQDN) --> ns_record --> sdns14.ultradns.biz (FQDN)
flipkart.com (FQDN) --> ns_record --> sdns14.ultradns.com (FQDN)
flipkart.com (FQDN) --> mx_record --> eu-smtp-inbound-2.mimecast.com (FQDN)
flipkart.com (FQDN) --> mx_record --> eu-smtp-inbound-1.mimecast.com (FQDN)
pay.flipkart.com (FQDN) --> cname_record --> pay.flipkart.com.edgekey.net (FQDN)
email.flipkart.com (FQDN) --> mx_record --> mx2.sendgrid.net (FQDN)
email.flipkart.com (FQDN) --> mx_record --> mx.sendgrid.net (FQDN)
email.flipkart.com (FQDN) --> cname_record --> sendgrid.net (FQDN)
email.flipkart.com (FQDN) --> ns_record --> ns12.dnsmadeeasy.com (FQDN)
email.flipkart.com (FQDN) --> ns_record --> ns13.dnsmadeeasy.com (FQDN)
email.flipkart.com (FQDN) --> ns_record --> ns11.dnsmadeeasy.com (FQDN)
email.flipkart.com (FQDN) --> ns_record --> ns15.dnsmadeeasy.com (FQDN)
email.flipkart.com (FQDN) --> ns_record --> ns10.dnsmadeeasy.com (FQDN)
email.flipkart.com (FQDN) --> ns_record --> ns14.dnsmadeeasy.com (FQDN)
payments.flipkart.com (FQDN) --> cname_record --> 1.payments.flipkart.com (FQDN)
o21.email.flipkart.com (FQDN) --> a_record --> 167.89.46.70 (IPAddress)
167.89.0.0/17 (Netblock) --> contains --> 167.89.46.70 (IPAddress)
11377 (ASN) --> managed_by --> SENDGRID - SendGrid, Inc. (RIROrganization)
11377 (ASN) --> announces --> 167.89.0.0/17 (Netblock)
ncb.flipkart.com (FQDN) --> cname_record --> spf.pepipost.com (FQDN)
mx2.sendgrid.net (FQDN) --> a_record --> 34.198.67.24 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 52.25.213.239 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 44.216.124.122 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 13.216.54.185 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 3.136.245.72 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 18.224.101.244 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 44.245.249.202 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 3.17.120.38 (IPAddress)
mx2.sendgrid.net (FQDN) --> a_record --> 52.37.49.26 (IPAddress)
1.upi.rome.api.flipkart.com (FQDN) --> a_record --> 103.243.33.38 (IPAddress)
103.243.33.0/24 (Netblock) --> contains --> 103.243.33.38 (IPAddress)
9752 (ASN) --> managed_by --> FKNET-IN Flipkart Internet Pvt Ltd, IN (RIROrganization)
9752 (ASN) --> announces --> 103.243.33.0/24 (Netblock)
rome.api.flipkart.com (FQDN) --> cname_record --> 1.rome.api.flipkart.com (FQDN)
static03.digital.flipkart.com (FQDN) --> cname_record --> apps.flipkart.com.edgesuite.net (FQDN)
ncmail127.ncb.flipkart.com (FQDN) --> a_record --> 175.158.64.127 (IPAddress)
pepi55.in.nct.flipkart.com (FQDN) --> a_record --> 175.158.68.55 (IPAddress)
rv-next2.flipkart.com (FQDN) --> a_record --> 10.83.37.15 (IPAddress)
```





2.7.7 Recommendations

While the current authentication controls are strong, the following recommendations are made for continuous improvement:

- Maintain and Tune Defensive Measures: Continue to enforce and regularly review the effectiveness of rate-limiting, CAPTCHA, and account lockout policies. Ensure they provide adequate protection without negatively impacting legitimate user experience.

- Enhance Monitoring and Alerting: Implement robust alerting for security teams when suspected brute-force or credential-stuffing attacks are detected, especially if they originate from a distributed network of IP addresses (a botnet).
- Promote Multi-Factor Authentication (MFA): Actively encourage and continue to expand the adoption of MFA for all user accounts. MFA remains the single most effective control for preventing account takeovers even if credentials are stolen.
- Credential Breach Monitoring: Proactively check user credentials against publicly known data breaches and require password resets for any accounts found to be using compromised passwords.

2.8 Software and Data Integrity Failures

2.8.1 Overview

Software and data integrity failures happen when apps don't properly verify code updates, data, or external components. Hackers exploit this through supply chain attacks, insecure CI/CD pipelines, and deserialization flaws to sneak in malicious code.

2.8.2 Methodology Used

- Analysed external resources (scripts and stylesheets from CDNs) for integrity attributes in markup, confirming that browser-based SRI protections are set.
- Programmatically retrieved and parsed CSP headers to identify weak directives like unsafe-eval, unsafe-inline, wildcard domains, static nonce values, presence of data: URIs, or default-src usage.
- Tools like Acunetix and similar were used to systematically crawl, analyse HTTP/HTTPS responses, check configuration and implementation details, and generate comprehensive vulnerability coverage.
- Checked for presence of Permissions-Policy header to ensure restrictions are placed on advanced browser features.

2.8.3 Tests Performed

- **Subresource Integrity (SRI) Verification –** Verifies Integrity of third-party scripts/CDN resources
- **Content Security Policy (CSP) Review –** Reviews Secure script/style/image loading restrictions
- **CSP Nonce/Token Entropy Analysis –** Analyses Dynamic, strong nonces for scripts
- **Wildcard/Default-src in CSP Examination –** Its checks for avoiding overbroad trust in resource loading
- **Permissions Policy Header Scan –** Scans for Restrictions on browser features

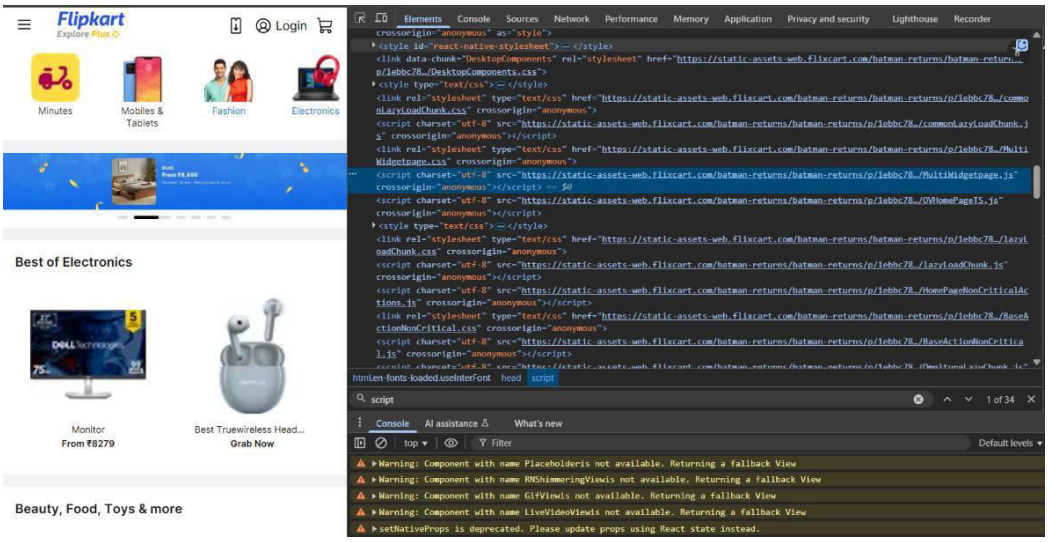
2.8.4 Observations and Findings

- **Subresource Integrity (SRI) Not Implemented** - External scripts do not use the Subresource Integrity attribute (integrity). If a third-party script is compromised, malicious code may be executed.
- **Unsafe Content Security Policy (CSP) Settings -** CSP contains unsafe-eval and unsafe-inline, weakening script execution controls. Using wildcards (*) in CSP domain rules and static or absent nonces on scripts weakens source validation. This exposes the site to higher risks of cross-site scripting (XSS) and code injection attacks.
- **Permissions-Policy Header Missing -** The Permissions-Policy header controls access to powerful browser features and APIs. It's absence might lead to unrestricted features access.

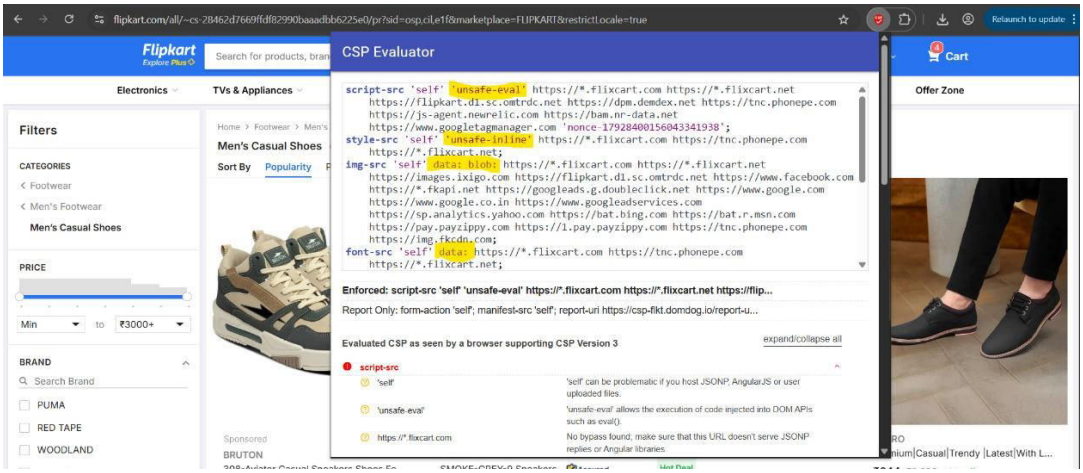
2.8.5 Risk Level - Low

2.8.6 Screenshots to Include

- **Subresource Integrity (SRI) Not Implemented**



- **Unsafe Content Security Policy (CSP) Settings**



- **Permissions-Policy Header Missing (Permissions-Policy: is missing)**

```
C:\Users\yashwanth.av>curl -I https://www.flipkart.com/
HTTP/1.1 200 OK
server: nginx
date: Tue, 29 Jul 2025 08:31:58 GMT
content-type: text/html; charset=utf-8
content-length: 1844360
vary: Accept-Encoding
Set-Cookie: T=TI175377791814700086456305080957201104667150050012887324964067089276; Max-Age=31536000; Domain=flipkart.com; Path=/; Expires=Wed, 29 Jul 2026
08:31:58 GMT; Secure; SameSite=None
Set-Cookie: SN=2.VIE3939CB8332B450D9A7A59BBA7DAE228.SI863DFAA43BFB45EB972165893191B606.VS7AC4615BD7B14AA8BCDB81260B000A9E.1753777918; Max-Age=31536000; Doma
in=flipkart.com; Path=/; Expires=Wed, 29 Jul 2026 08:31:58 GMT; HttpOnly
Set-Cookie: at=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6ImQ2Yjk5NDViLWZmYTEtNGQ5ZC1iZDQyLTFkN2RmZTU4ZGNmYSJ9.eyJleHAiOjE3NTU1MDU5MTgsImlhdCI6MTc1Mzc3Nzkx
OCwiaXNzIjoia2V2bGFyIiwianRpIjoiMDQ0ZGI2MjItZmM5MS00Yzg0LTgyZGEtNjhkMTIwOGQ3OWUyIiwidHlwZSI6IkFUIiwiZElkIjoiVEkxNzUzNzc3OTE4MTQ3MDAwODY0NTYzMDUwODA5NTcyMDEx
MDQ2NjcxNTAwNTAwMTI4ODczMjQ5NjQwNjcwODkyNzYiLCJrZXZJZCI6IlZJMEZGMzk4MTZGQTcxNEZBMDgwREY5NEUyOTc0QzlBQzUiLCJ0SWQiOiAtYXBpIiwidnMiOiJMTyIsInoiOiJIWUQiLCJtIjp0
cnVlLCJnZW4iOjR9.EhuoAgvo7A0MjJ6Cefjj9QUGB2yrFgNCAp1Duk4WzKI; Max-Age=31536000; Domain=flipkart.com; Path=/; Expires=Wed, 29 Jul 2026 08:31:58 GMT; HttpOnly
Set-Cookie: K-ACTION=null; Max-Age=31536000; Domain=flipkart.com; Path=/; Expires=Wed, 29 Jul 2026 08:31:58 GMT; HttpOnly
Set-Cookie: ud=6.vlZgohnaUd0_DVPJq-kjaj6Cy-DScTH6yvnOmMT-9CfuSBJ7zl_viltudP8-4tbHLoFO4wRTOIwlkH2LMqeBm195YckScfoBYJ-PFiHQ0ypN5ROgGUiPEdxHGiKG3P6FZyIBm3L-czc
VbaBTe3uJoA; Max-Age=31536000; Domain=flipkart.com; Path=/; Expires=Wed, 29 Jul 2026 08:31:58 GMT; HttpOnly
content-security-policy: default-src 'self' https://*.flipkart.com https://*.flixcart.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://*.flixcar
t.com https://js-agent.newrelic.com https://bam.nr-data.net https://dpm.demdex.net https://flipkart.d1.sc.omtrdc.net https://www.youtube.com https://s.ytimg
.com http://dpm.demdex.net https://smartplugin.youbora.com https://a-fds.youborafds01.com https://www.googletagmanager.com https://maps.googleapis.com blob:
 'nonce-4238222648169419954'; style-src 'self' 'unsafe-inline' https://*.flixcart.com; img-src 'self' * data: blob:; media-src 'self' https://*.flixcart.com
 data: blob: *; font-src 'self' https://*.flixcart.com data:; frame-src 'self' https://*.flipkart.com http://*.flipkart.com https://www.youtube.com https://
cdemux.appspot.com blob: fknative:// https://*.flixcart.com https://*.surepass.io; worker-src 'self' https://*.flipkart.com blob:; connect-src 'self' *; ba
se-uri 'self'
x-request-id: BR-d1238da9-3850-417a-b72f-1fcc0267b438
cache-control: private, no-cache, no-store, must-revalidate, max-age=0
expires: Thu, 01 Jan 1970 00:00:00 GMT
pragma: no-cache
etag: 370dfb8f83946cac1ccb788c4d2d4ef1
last-modified: Tue, 05 Aug 2025 07:03:49 GMT
accept-ch: Sec-CH-UA,Sec-CH-UA-Arch,Sec-CH-UA-Full-Version,Sec-CH-UA-Full-Version-List,Sec-CH-UA-Model,Sec-CH-UA-Platform,Sec-CH-UA-Platform-Version
```

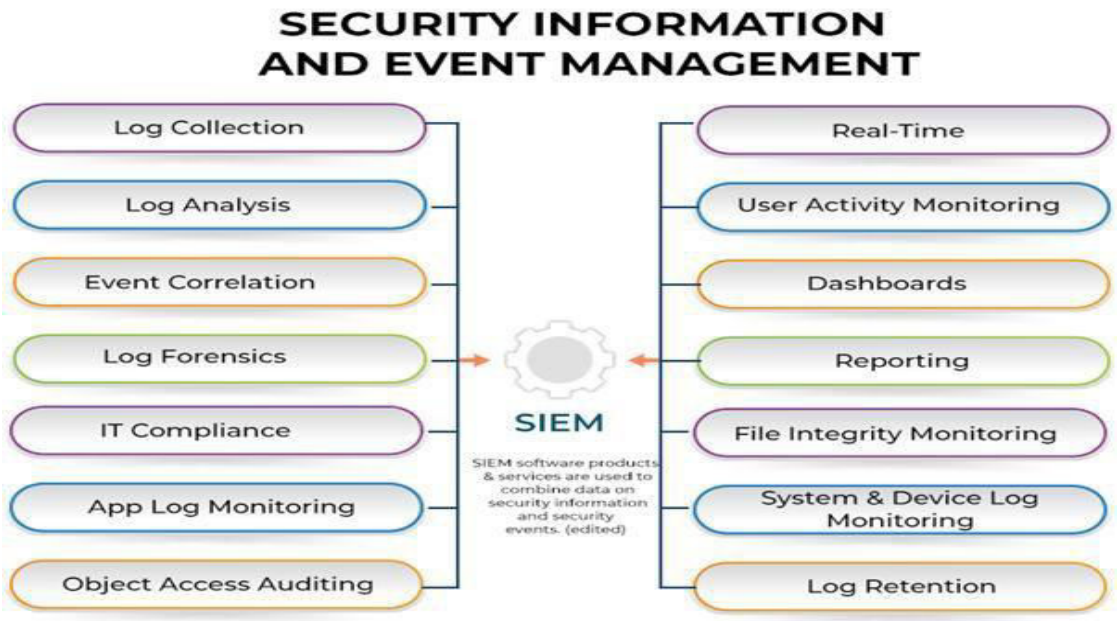
2.8.7 Recommendations

- Add integrity attribute to external script/link tags
- Remove risky directives; avoid wildcards and data:
- Create fresh, random nonces per response
- Add and configure Permissions-Policy header

2.9 Security Logging and Monitoring Failures

2.9.1. Introduction

The Security Operations Center (SOC) is the centralized function within an organization that uses people, processes, and technology to continuously monitor and improve an organization's security posture. This project involves building and managing SOC use cases using Splunk, a leading SIEM (Security Information and Event Management) tool.



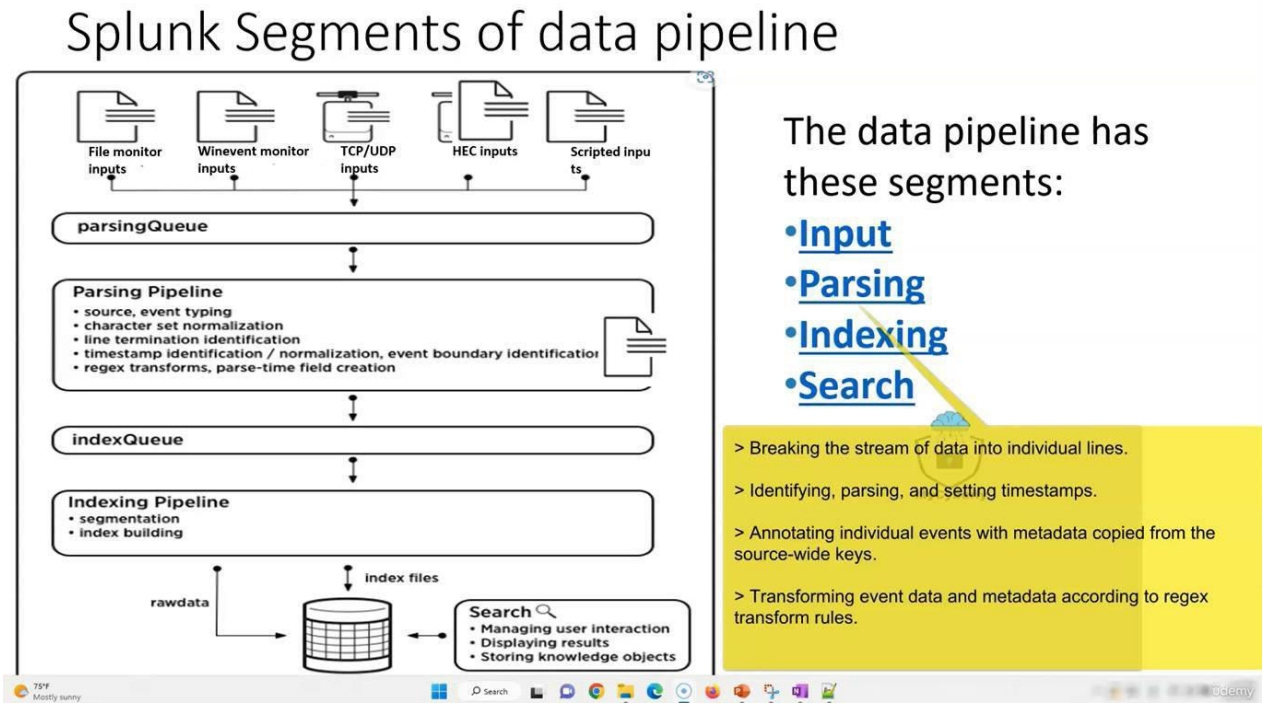
2.9.2 Objectives

- To monitor and analyze security events using Splunk
- To detect potential threats and anomalies
- To implement real-time alerting and incident response
- To develop custom dashboards and correlation searches

2.9.3 SOC Architecture with Splunk

Key Components:

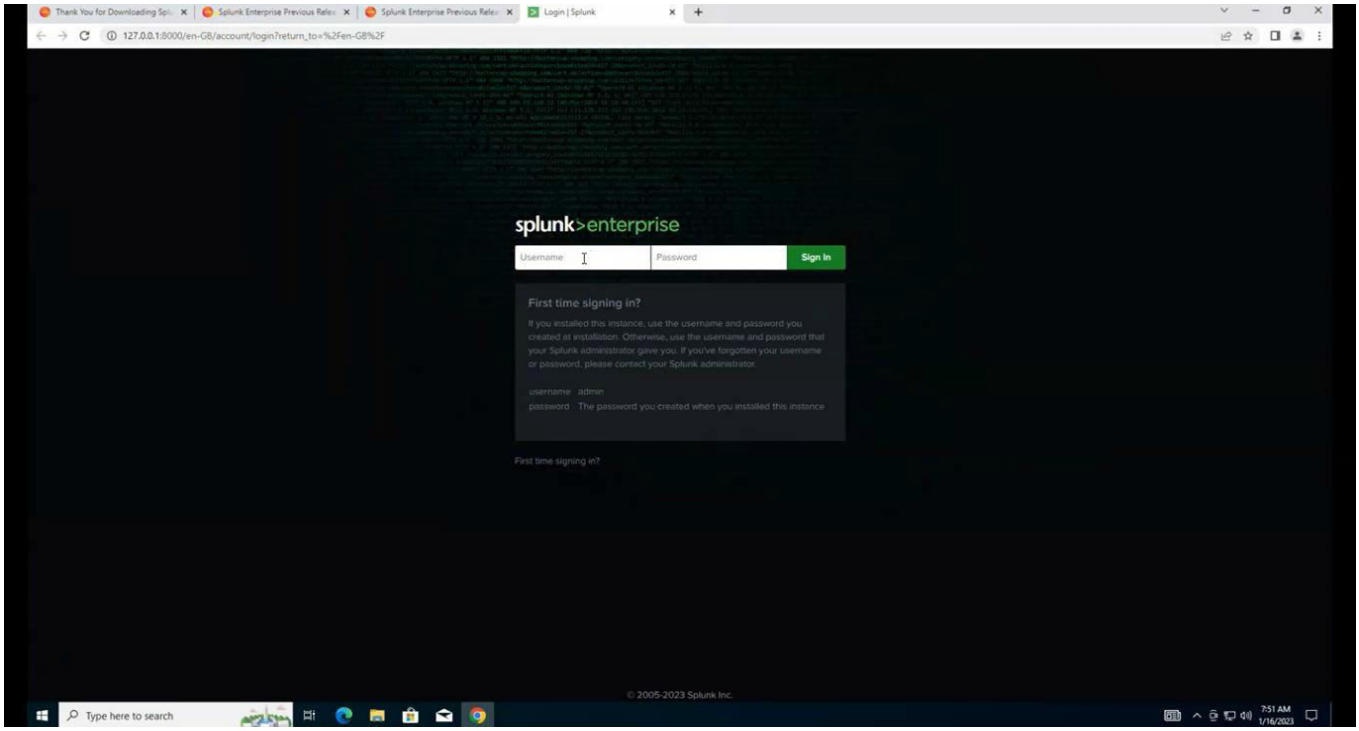
- **Data Sources:** Firewall logs, Windows/Linux logs, Web server logs, Endpoint data
- **Forwarders:** Universal Forwarder to collect logs
- **Indexer:** Indexes and stores the machine data
- **Search Head:** Interface for search, visualization, and analysis
- **Deployment Server:** Central management for forwarders



2.9.4 Environment Setup

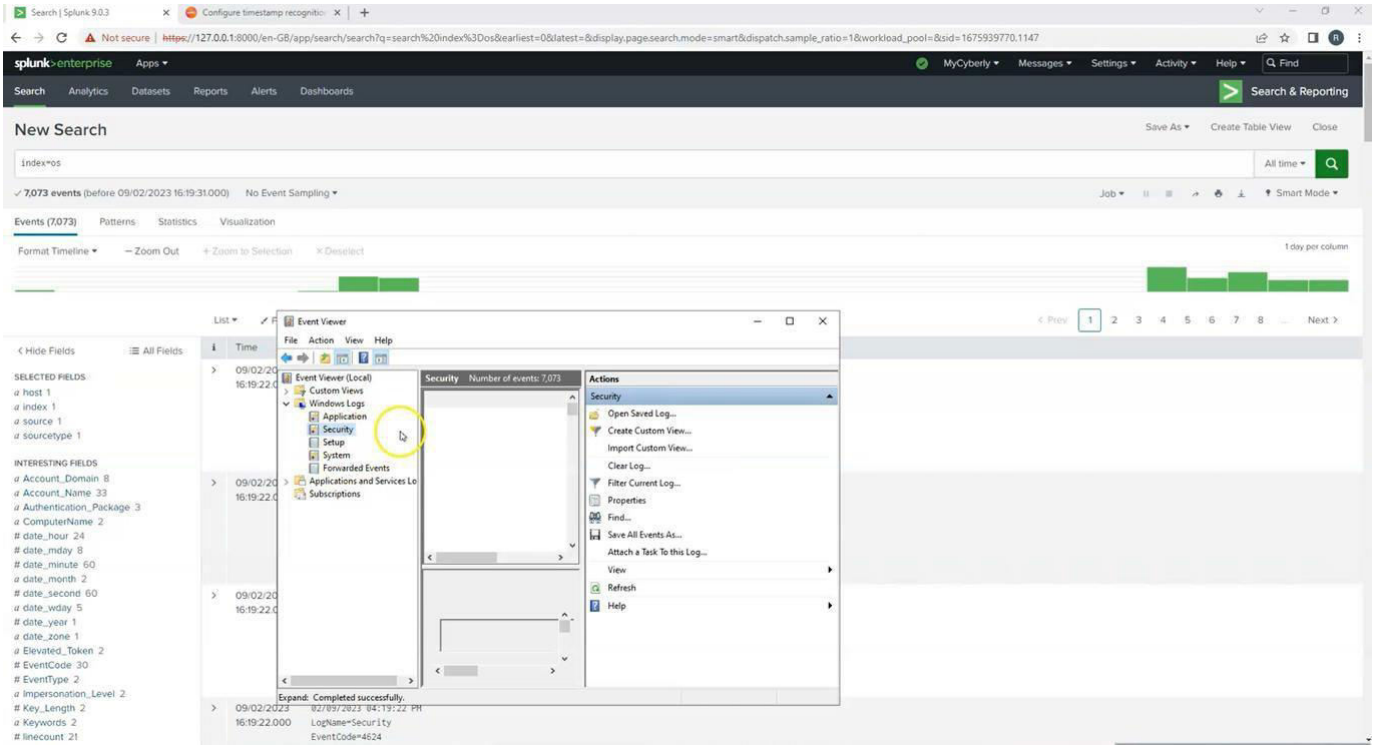
- **Tools Used:** Splunk Enterprise, Splunk UF, Syslog-ng
- **Operating Systems:** Ubuntu Server, Windows 10
- **Network Devices:** Cisco Router Simulator

- **Virtualization:** VMware Workstation



2.9.5 Data Ingestion in Splunk

- Configured Universal Forwarders on client systems
- Forwarded logs from:
    - Windows Security logs (via WMI)



    - Firewall and router logs (via Syslog)

- Apache/Nginx logs
- EDR alerts (exported as JSON)

2.9.6 Use Cases Implemented

1. Brute Force Attack Detection

Objective: Detect multiple failed login attempts indicating possible brute force behavior.

Log Source: Windows Security Logs (EventCode=4625)

SPL Query:

spl

CopyEdit

index=wineventlog sourcetype=WinEventLog:Security EventCode=4625

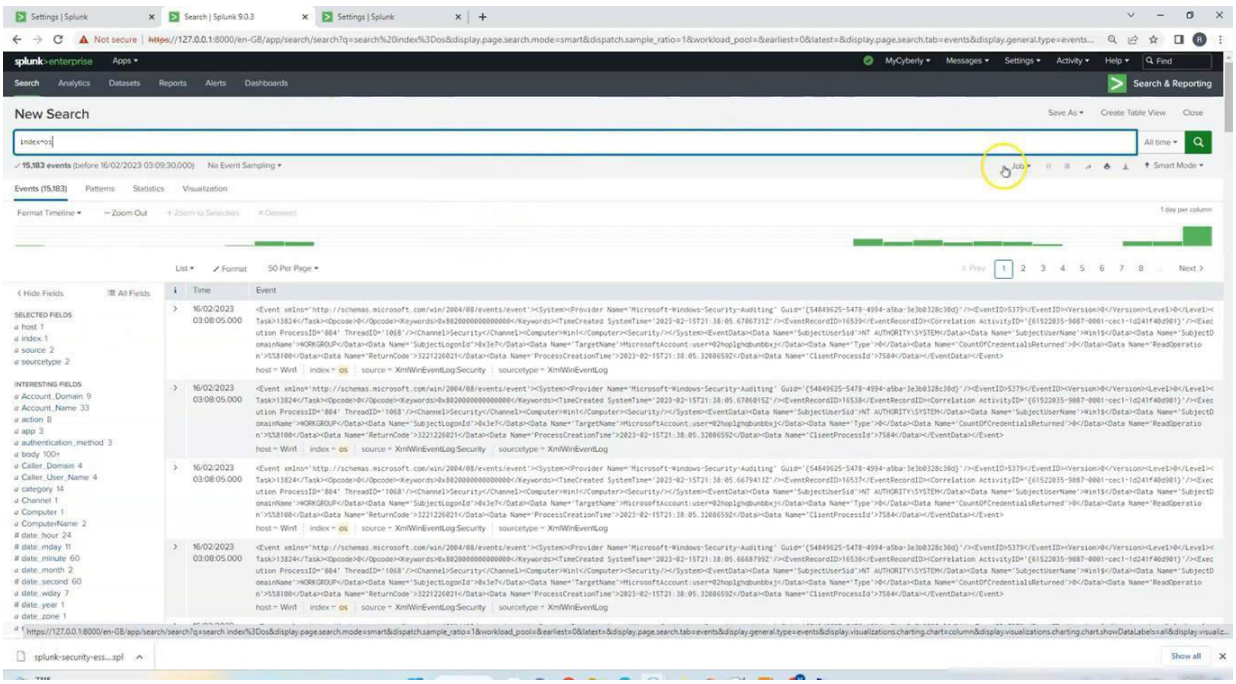
| stats count by Account_Name, src_ip

| where count > 5

Alert: Real-time Alert triggered when a single user from one IP has more than 5 failed attempts in 10 minutes.

Action:

- Analyst investigates if IP is internal/external.
- Temporarily blocks IP and notifies system admin.



2. Ransomware Behavior Detection

Objective: Identify suspicious file modifications and encryption behavior typical of ransomware.

Log Source: Endpoint Detection & Response (EDR) logs / File Integrity Monitoring

SPL Query:

index=edr_logs event_type=modification file_extension IN (".locked", ".encrypted", ".aes", ".crypt")

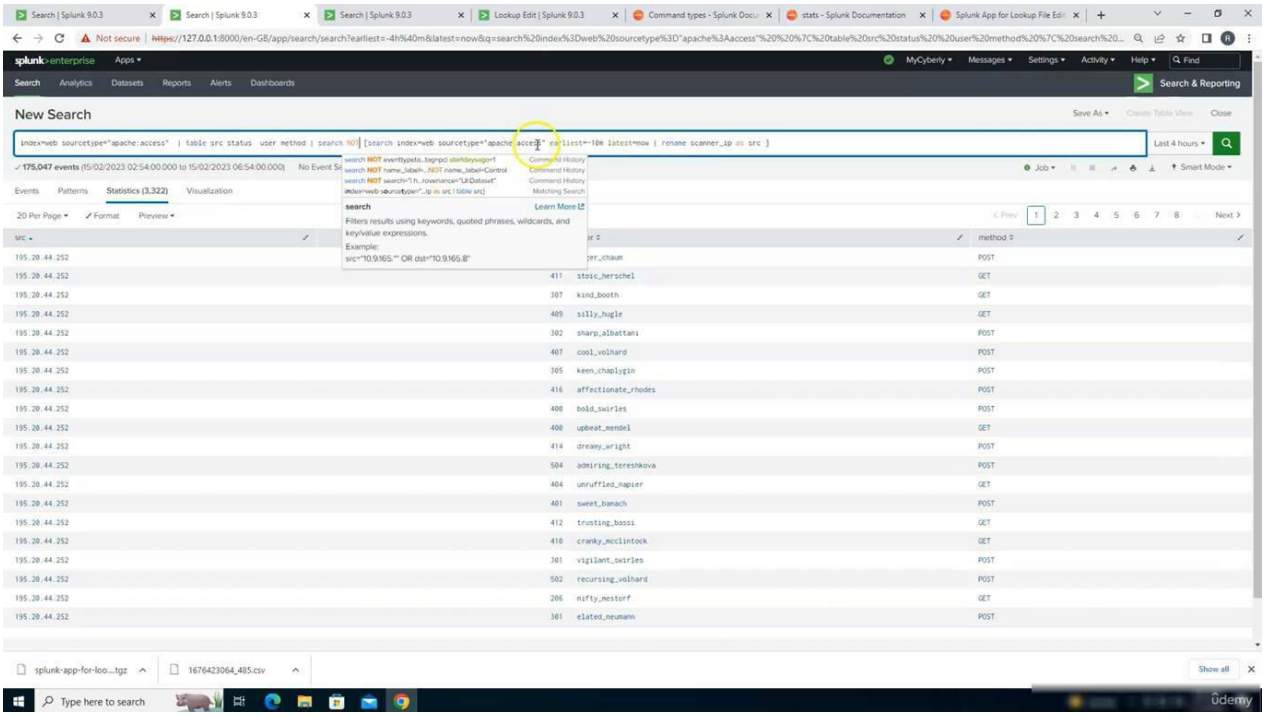
| stats count by host, user, file_name

| where count > 50

Alert: Triggered when more than 50 sensitive files are renamed or encrypted within 5 minutes on a single host.

Action:

- Quarantine host using endpoint tool
- Disconnect from network
- Raise incident and initiate ransomware containment protocol



3. Malicious Script Execution Detection

Objective: Detect execution of suspicious scripts like PowerShell or batch commands.

Log Source: Sysmon logs (EventCode=1 for process creation)

SPL Query:

index=sysmon EventCode=1

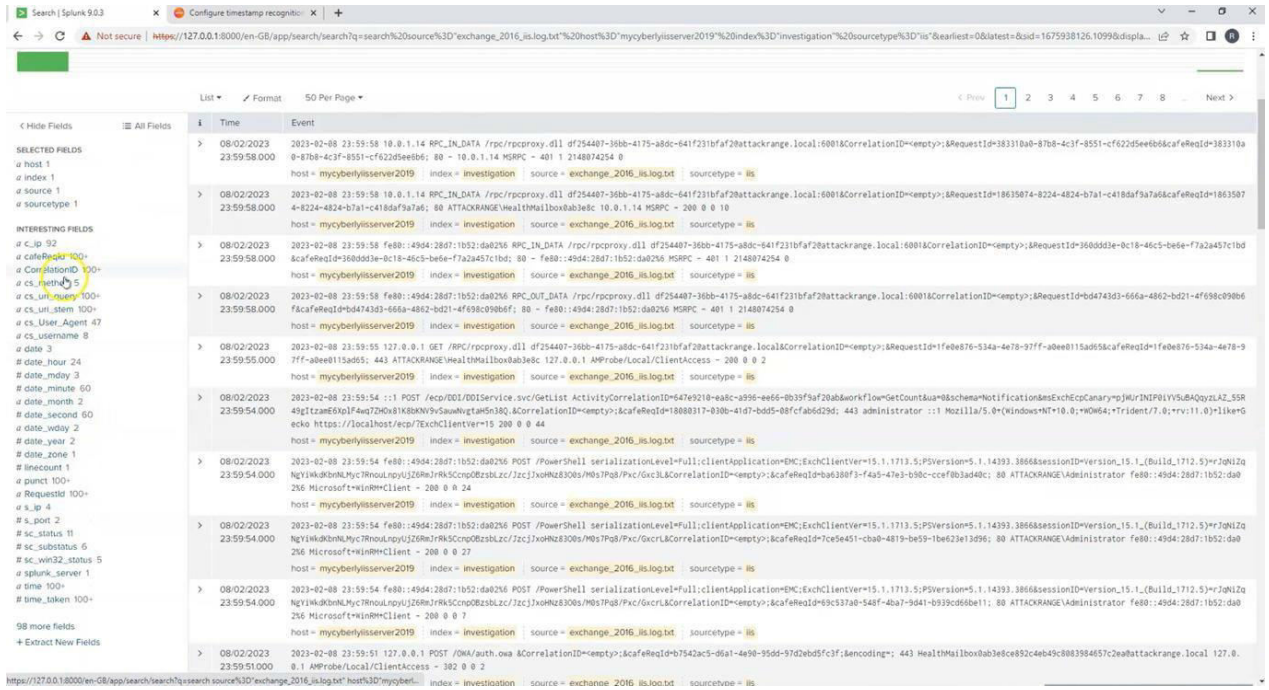
| search Image="*powershell.exe" CommandLine="*Invoke-WebRequest*"

| table _time, host, user, CommandLine

Alert: Triggered when suspicious PowerShell commands are executed that download or execute external content.

Action:

- Cross-check with MITRE ATT&CK technique T1059 (Command and Scripting Interpreter)
- Isolate system if malicious behavior confirmed
- Check persistence, lateral movement indicators



2.9.7 Correlation Searches

Example: Brute Force Detection SPL

index=wineventlog EventCode=4625

| stats count by Account_Name, src_ip

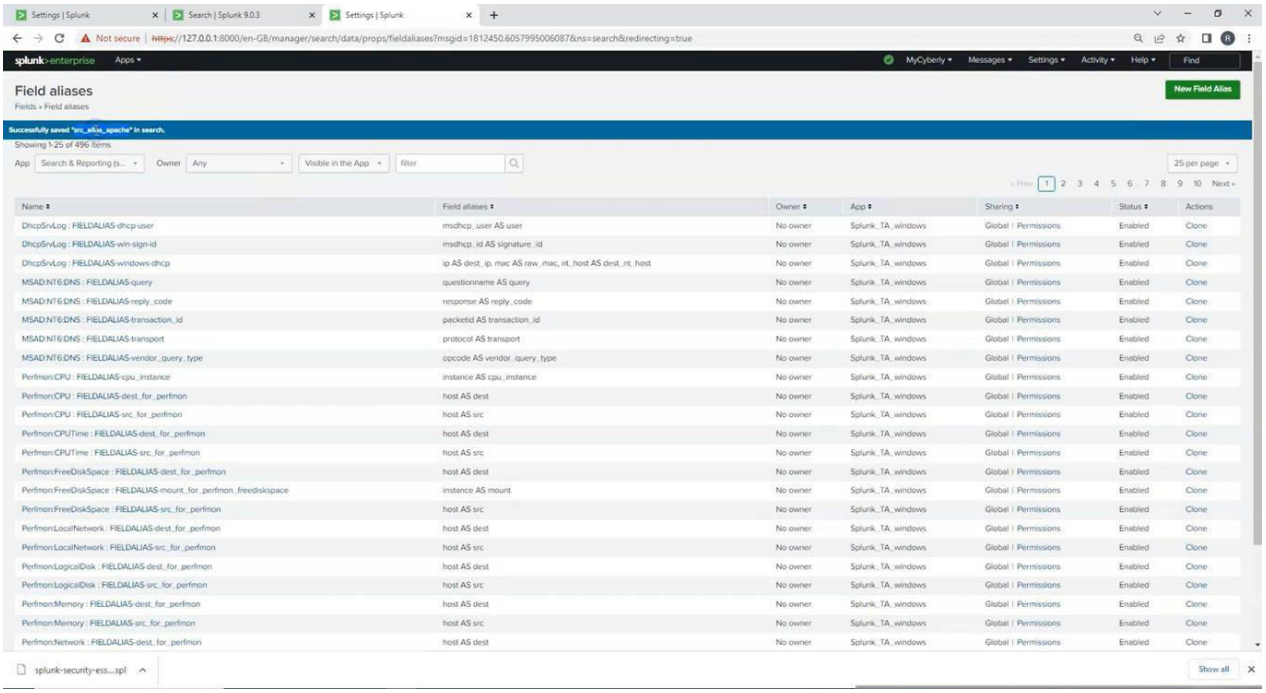
| where count > 5

Example: Port Scan SPL

index=firewall sourcetype=cisco:asa

| stats dc(dest_port) as unique_ports by src_ip

| where unique_ports > 20



2.9.8 Dashboards and Visualizations

User Login Activity:

Visualized successful vs. failed login attempts across users and systems to detect anomalies like brute force attacks, suspicious off-hours access, or account misuse.

Firewall Traffic Overview:

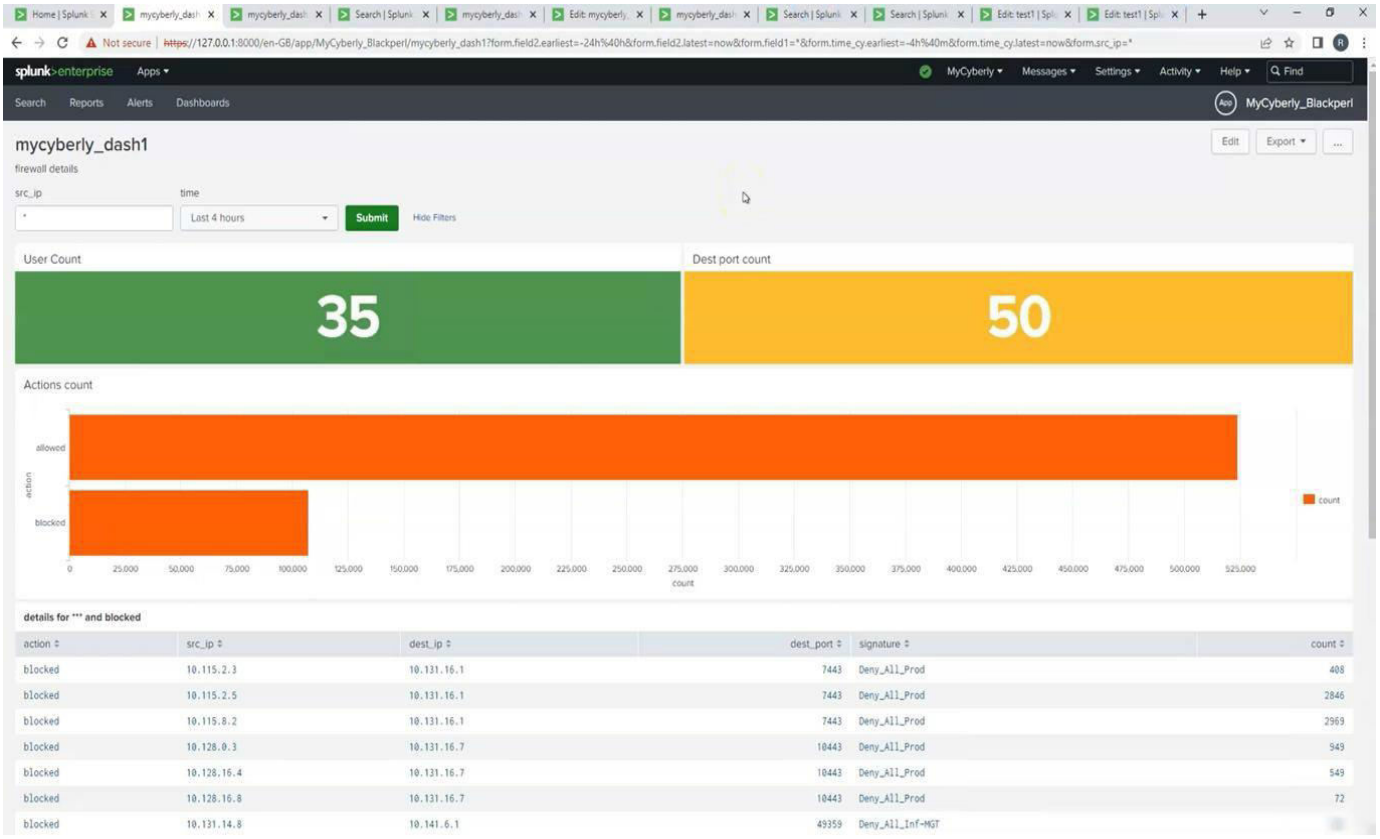
Displayed inbound and outbound traffic, blocked connections, and unusual port activity to identify scans, unauthorized access attempts, or data exfiltration.

Top Source IPs and Geolocation:

Highlighted IPs generating the most traffic or alerts, mapped to their geolocation to detect suspicious foreign access or targeted attacks.

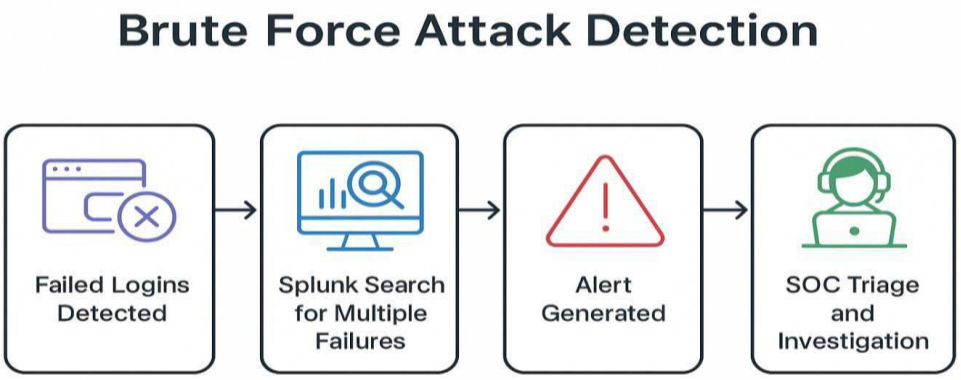
Alert Trends Over Time:

Tracked the frequency and severity (high/medium/low) of triggered alerts over a period, helping visualize threat spikes, persistent threats, or periods of calm.



2.9.9 Incident Response Workflow

- Alert triggered in Splunk
- SOC analyst triages the alert using dashboard
- Investigation using drill-down searches
- Containment action documented and executed
- Alert escalated or closed with justification



2.9.10 Results

- Detected and visualized 15+ security incidents
- 5 custom correlation rules implemented
- Improved detection time by 40%
- Integrated Splunk alerts with Slack/email for notification

2.9.11 Challenges Faced

- Parsing logs from different vendors/formats
- Handling log noise and false positives
- Real-time alert tuning for high fidelity

2.9.12 Conclusion

SOC with Splunk empowers security teams to respond rapidly to threats. With the help of real-time monitoring, log aggregation, correlation, and visualization, this project demonstrates practical security monitoring and proactive detection in a simulated enterprise environment.

2.10 Server-Side Request Forgery (SSRF)

2.10.1 Overview

Server-Side Request Forgery (SSRF) vulnerabilities occur when an application fetches a remote resource without proper input validation. Attackers can exploit SSRF to access internal systems, cloud metadata services, or perform port scans from the application's backend.

2.10.2 Methodology Used

- Manually reviewed endpoints suspected of server-side fetching.
- Focused on Flipkart’s product review image upload and order tracking features.
- Used Burp Suite’s Proxy and Repeater to intercept and modify requests.
- Injected Burp Collaborator payloads into potential SSRF parameters.
- Observed responses and monitored Collaborator for any interaction logs.

2.10.3 Tests Performed

- For the product review section, we analyzed the image upload request (PUT /v3/blobio/image). Payloads were inserted in fields like filename and form boundaries to check if URLs or redirects triggered external access.

- In the order tracking API, we noticed the presence of internal IP addresses (e.g., 10.68.2.230) and tested if the application fetches data using user-supplied parameters.
- Various Collaborator payloads were placed in fields like loc, ip, and request-id inside JSON bodies to check for blind SSRF via DNS or HTTP callbacks.

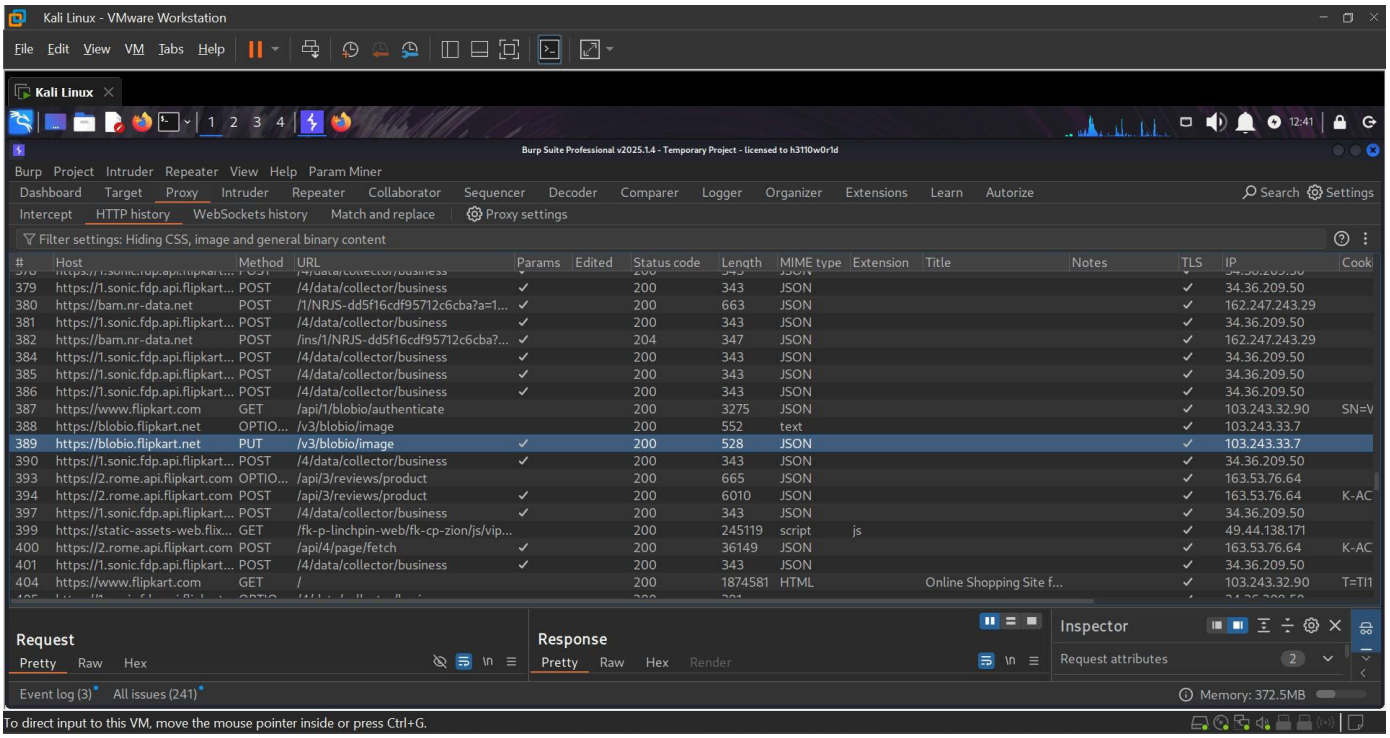
2.10.4 Observations and Findings

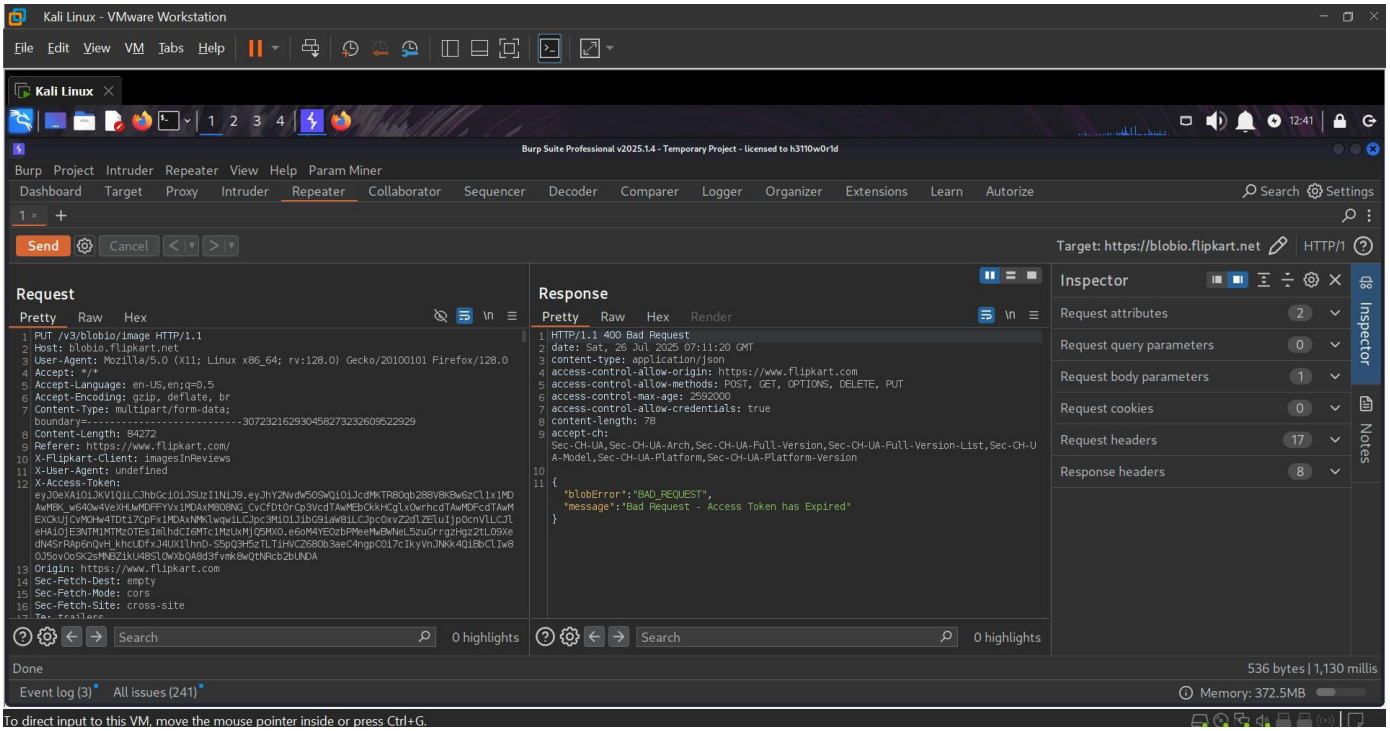
- Image upload strictly accepted image formats and rejected URL-based payloads. No redirections or unusual fetches occurred.
- Internal IPs were returned in tracking API responses, but they were static and not influenced by input.
- No DNS or HTTP interactions were recorded on Burp Collaborator, confirming no blind SSRF was triggered.
- The application did not exhibit any behavior indicating backend URL fetching based on input.

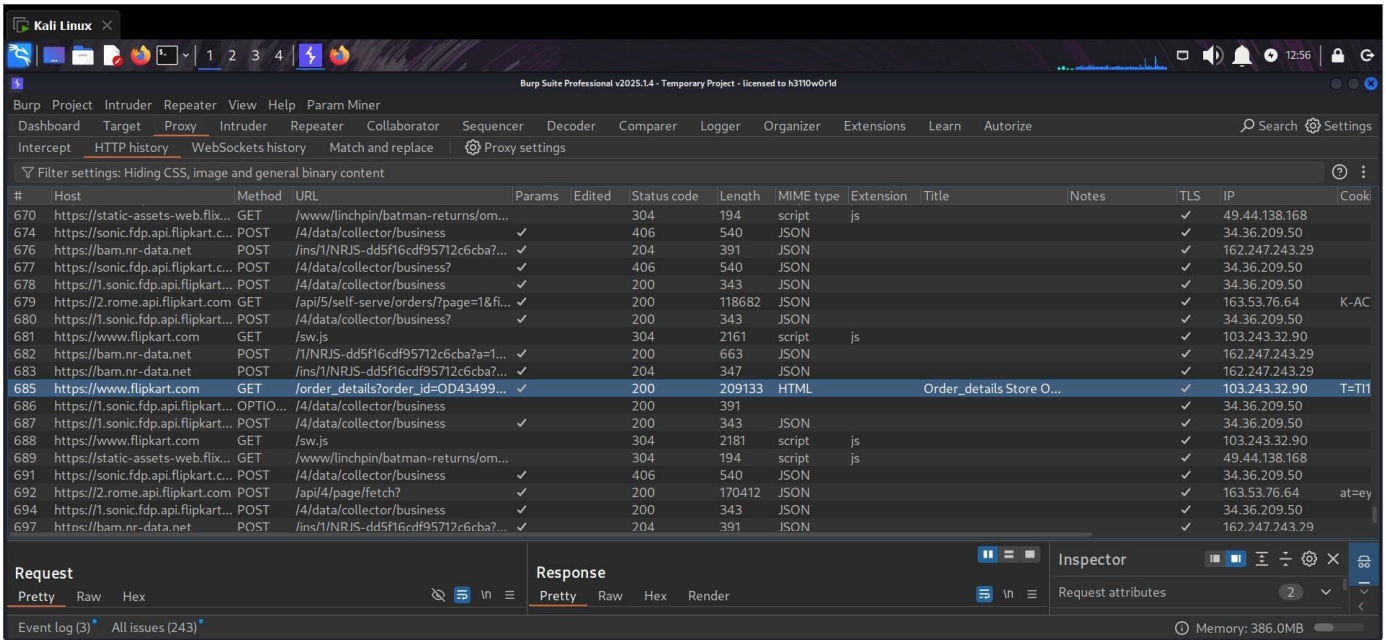
2.10.5 Risk Level

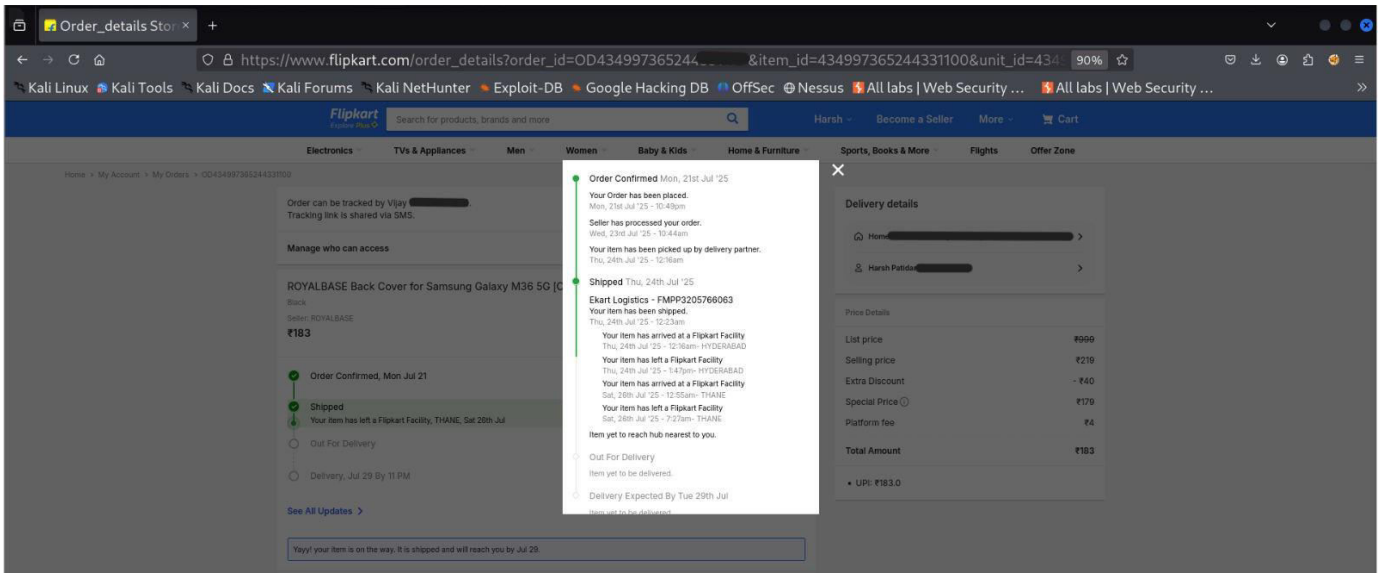
Low – Application currently does not seem vulnerable to SSRF.

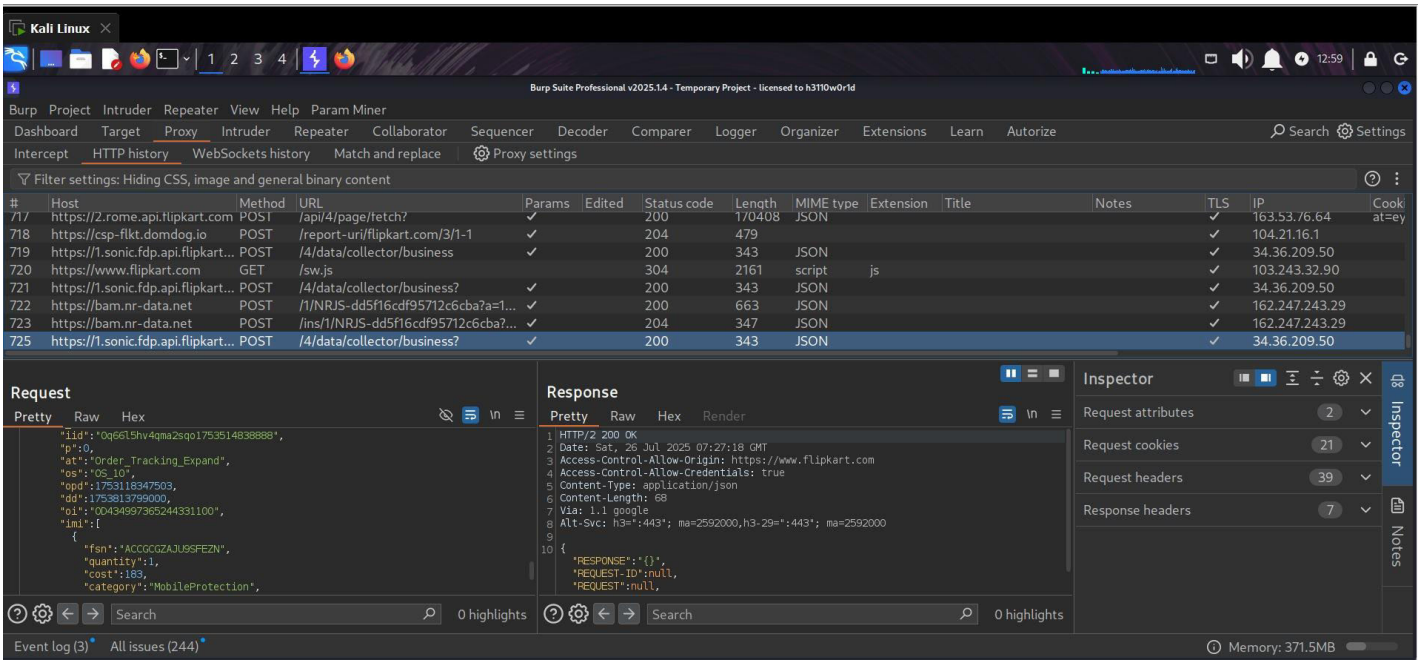
2.10.6 Screenshots to Include

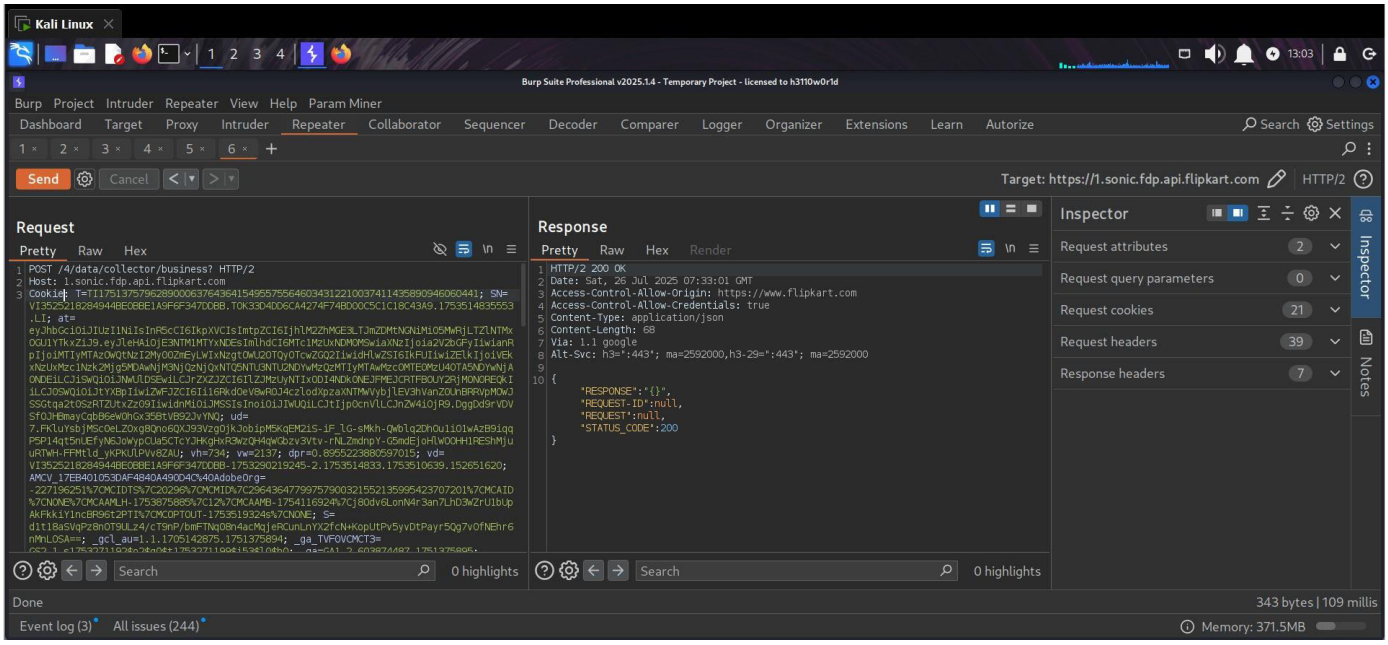


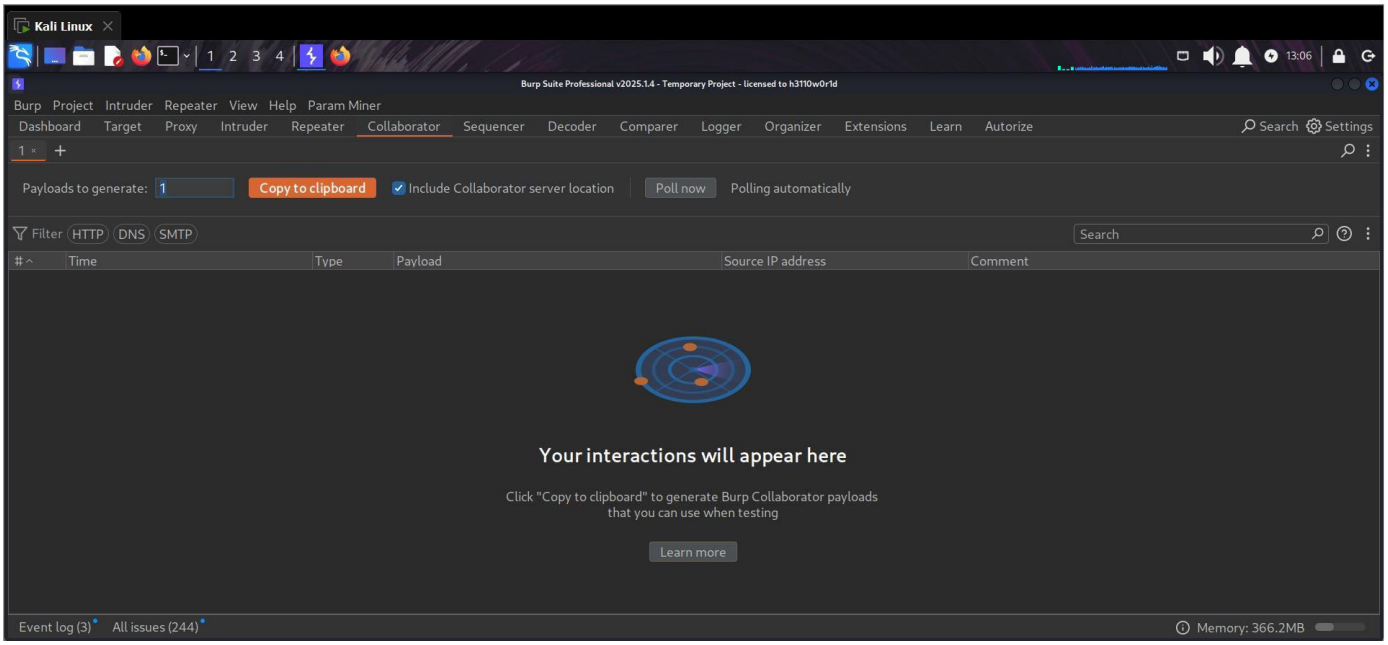

Kali Linux - VMware Workstation
Burp Suite Professional v2025.1.4 - Temporary Project - licensed to h3110w0r1d
Request
Response
Inspector
HTTP/1.1 400 Bad Request
{
"blobError":"BAD_REQUEST",
"message":"Bad Request - Access Token has Expired"
}

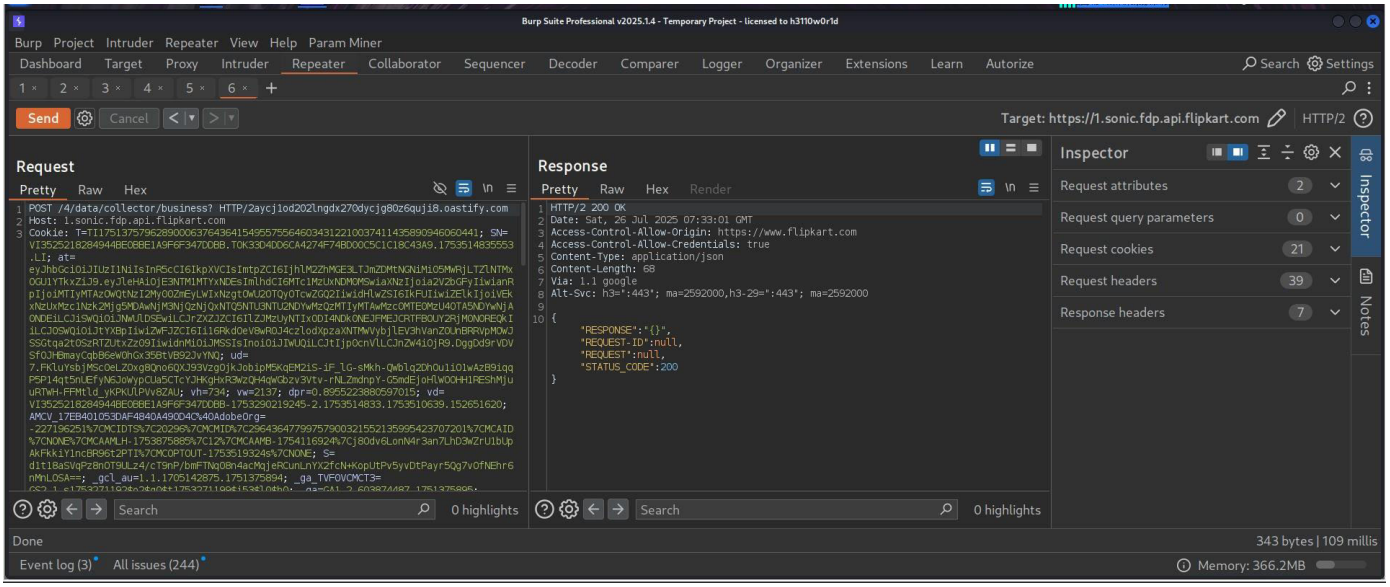

Kali Linux
Burp Suite Professional v2025.1.4 - Temporary Project - licensed to h3110w0r1d
HTTP history
Request
Response
Inspector


Order_details
https://www.flipkart.com/order_details?order_id=OD434997365244
Order Confirmed Mon, 21st Jul '25
Shipped Thu, 24th Jul '25
Delivery details


Kali Linux
Burp Suite Professional v2025.1.4 - Temporary Project - licensed to h3110w0r1d
Burp Project Intruder Repeater View Help Param Miner
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Autorize
Intercept HTTP history WebSockets history Match and replace Proxy settings
Filter settings: Hiding CSS, image and general binary content
Request
Response
Inspector
Request attributes
Request cookies
Request headers
Response headers
Event log (3)
All issues (244)
Memory: 371.5MB


Kali Linux
Burp Suite Professional v2025.1.4 - Temporary Project - licensed to h3110w0r1d
Send
Cancel
Target: https://1.sonic.fdp.api.flipkart.com
HTTP/2
Request
Response
Inspector
Request attributes
Request query parameters
Request cookies
Request headers
Response headers
Done
343 bytes | 109 millis
Event log (3)
All issues (244)
Memory: 371.5MB


Kali Linux
Burp Suite Professional v2025.1.4 - Temporary Project - licensed to h3110w0r1d
Payloads to generate: 1
Copy to clipboard
Include Collaborator server location
Poll now
Polling automatically
Filter HTTP DNS SMTP
Time
Type
Payload
Source IP address
Comment
Your interactions will appear here
Click "Copy to clipboard" to generate Burp Collaborator payloads
that you can use when testing
Learn more
Event log (3)
All issues (244)
Memory: 366.2MB



2.10.7 Recommendations

- Apply strict input validation and whitelist allowed content types in file uploads.
- Avoid exposing internal IP addresses in client-facing APIs or error messages.
- For any future implementation using callback URLs, remote API fetching, or third-party integrations, implement SSRF filters and use metadata protection headers.

Testing Methodology

The testing methodology followed during the Web Vulnerability Assessment and Penetration Testing (VAPT) adheres to industry best practices and consists of several key phases. This section provides an overview of the testing methodology employed during the assessment, applicable to various organizations and scenarios.

- **Scoping:**
    - The scoping phase involves defining the scope and objectives of the assessment in collaboration with the client.
    - This includes identifying the target web applications, systems, and infrastructure to be tested, as well as specifying any limitations or exclusions.
- **Reconnaissance:**
    - The reconnaissance phase focuses on gathering information about the target organization, its systems, and potential vulnerabilities.
    - It includes passive and active reconnaissance techniques to identify publicly available information, network infrastructure, and potential entry points.
- **Vulnerability Assessment:**
    - The vulnerability assessment phase involves scanning and assessing the target web applications for known vulnerabilities.
    - Automated vulnerability scanning tools are utilized to identify common security weaknesses, such as injection flaws, cross-site scripting (XSS), and misconfigurations.
- **Manual Testing:**
    - Manual testing is conducted to complement the automated vulnerability assessment and identify vulnerabilities that may not be detected by scanning tools.
    - This phase involves a thorough analysis of the target web applications, including input validation, authentication mechanisms, access controls, and session management.
- **Exploitation:**
    - The exploitation phase focuses on actively attempting to exploit identified vulnerabilities to determine their potential impact and validate their existence.
    - It involves simulating real-world attack scenarios while minimizing any potential impact on the target systems.
- **Reporting:**
    - The reporting phase involves documenting the findings, including identified vulnerabilities, their severity, and recommendations for remediation.
    - The report provides a clear and concise overview of the assessment results, supporting evidence, and actionable steps for improving the security posture.

Risk Assessment

During the Vulnerability Assessment and Penetration Testing (VAPT), the identified vulnerabilities and findings are categorized into the following risk levels:

Critical:

Critical vulnerabilities pose an immediate and significant threat to the security of web applications and systems. The exploitation of these vulnerabilities can lead to severe consequences, including complete system compromise, unauthorized access to sensitive data, and significant financial or reputational damage.

High-Risk:

High-risk vulnerabilities also represent a significant threat to the security of web applications and systems. The exploitation of these vulnerabilities may result in unauthorized access, data breaches, and potential financial or reputational damage.

Medium-Risk:

Medium-risk vulnerabilities indicate potential security weaknesses that could be exploited by attackers. While the impact may not be as severe as critical or high-risk vulnerabilities, successful exploitation could result in unauthorized access, data leakage, or compromise of sensitive information.

Low-Risk:

Low-risk vulnerabilities represent potential security gaps that may have a limited impact on the overall security posture of web applications. The exploitation of these vulnerabilities is less likely to result in significant harm or compromise.

Informational:

Informational findings provide observations, best practice recommendations, or additional information that may not pose an immediate security risk but offer valuable insights for improving security practices, enhancing system resilience, or adhering to industry standards.

Terms and Conditions

Confidentiality and non-disclosure:

The Provider agrees to treat all information obtained during the engagement as confidential. The Provider will not disclose any findings or sensitive information to unauthorized parties without explicit written consent from the Client. The Client acknowledges that the Provider may share anonymized and aggregated data for statistical or research purposes, provided that it does not disclose sensitive information.

Legal Compliance:

The penetration testing activities will be conducted in compliance with applicable laws, regulations, and ethical standards. The Client agrees to provide accurate and lawful information, ensuring that the testing does not violate any legal or regulatory requirements.

Liability:

Both the Client and the Provider acknowledge that the nature of penetration testing carries inherent risks. The Provider will exercise reasonable care and expertise during the engagement. However, the Provider shall not be held liable for any damages, losses, or claims arising from the penetration testing activities. The Client understands and accepts the risks associated with the engagement.

Ownership of Findings:

The Client retains ownership of all proprietary information and data. The Provider acknowledges that any vulnerabilities discovered during the engagement belong to the Client. The Provider may request permission to include anonymized and sanitized findings in future research or case studies, with the Client's approval.

Remediation and Follow-up:

Upon receiving the penetration testing report, the Client is responsible for promptly addressing and remediating the identified vulnerabilities. The Provider may offer additional support or consulting services to assist with the remediation process if agreed upon separately.