TESTFIRE
Security Assessment Findings Report

Started on -
20/06/2022

Table of Contents

Confidentiality Statement

This document is the exclusive property of Organization Name . This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both organizations.

TestFire Corp 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. Organization prioritized the assessment to identify the weakest security controls an attacker would exploit. Our organization recommends conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

Contact Information

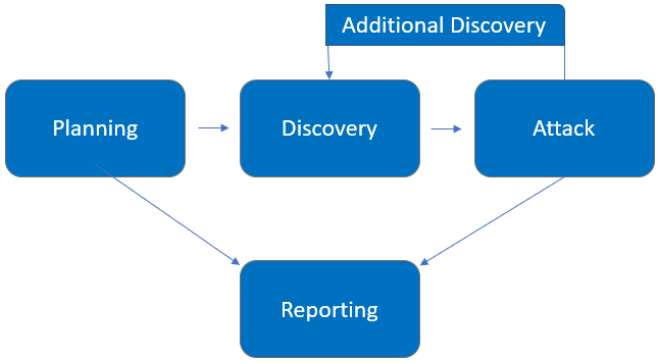
Name	Title	Contact Information
Demo Corp		
Organization		Email:
Kumar Atul	Penetration Tester	Email: kumaratul.dnb@gmail.com

Assessment Overview

From June 20th, 2022 to June 22nd, 2022, Demo Corp engaged Our Organization to evaluate the security posture of its infrastructure compared to current industry best practices that included an internal network penetration test. All testing performed is based on the *OWASP Testing Guide, and customized testing frameworks*.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



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	Definition
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

Assessment	Details
Web Application Penetration test	https://testfire.net | IP-

Scope Exclusions

Per client request, Our Organization did not perform any of the following attacks during testing:

- Denial of Service (DoS)
- Phishing/Social Engineering

All other attacks not specified above were permitted by Testfire.

Client Allowances

Testfire provided our organization the following allowances:

- Full web application pentest on the given website link and no exclusion of certain web pages from the website.

Executive Summary

Our Organization evaluated Testfire web application security posture through penetration testing from June 20th, 2022 to June 22nd, 2022. The following sections provided high-level, medium-level, low-level and Informational-level overview of vulnerabilities discovered, successful and unsuccessful attempts, and strengths and weaknesses.

Scoping and Time Limitations

Scoping during the engagement did not permit denial of service or social engineering across all testing components.

Time limitations were in place for testing. Web Application penetration testing was permitted for two
(2) business days.

Testing Summary

The Web Application assessment evaluated TestFire Web Application security posture. From an internal perspective, the pen tester performed manual vulnerability assessment against the website link provided by testfire to evaluate the overall patching health of the website. The pen tester found many vulnerabilities on the web app and exploited it for poc.

The pen tester used some manual techniques to exploit basic level of vulnerabilities and was successfully able to get admin access to the web app using various exploitation methods. The security of the web app is completely compromised by basic level of exploitation which are mentioned below along with the evidence.

Since it was a time bound assessment so most the vulnerabilities were reported by the pen tester in the given time. However there are more serious exploits still present in the system but due to the time limit only some of them are reported.

Key Strengths and Weaknesses

The following identifies the key strengths identified during the assessment:

The following identifies the key weaknesses identified during the assessment:

1. Default credentials were used in web app
2. Broken Authentication.
3. Improper input validation on various input parameters
4. Internal server error being displayed.
5. Insecure direct object reference allowed.
6. Injection allowed on various input parameters.
7. No mechanism to stop brute force of login information
8. Displaying user information on web page.
9. GitHub source code link of web app displayed on screen

Vulnerability Summary & Report Card

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

Web Application Penetration Test Findings

1	4	1	1	1
Critical	High	Moderate	Low	Informational

Finding	Severity	Recommendation
Web Application Penetration Test		
1: Apache Tomcat insecure default administrative password	High	Change the default password
2: Insecure Direct Object reference	High	Validate and filter any sort of input even if it's not malicious.
3: SQL Injection Vulnerability allowing login bypass.	Critical	Use parametrized query instead of string concatenation within the query.
4: Login Brute on username and password.	High	Restrict multiple request from a same source.
5: Improper input Validation.	High	Proper validation and filter of input to be performed.
6: Reflected XSS.	Moderate	Implement proper filtration of various characters during input.
7. Displaying user info on web app	Low	Properly check for any user info displayed on screen.
8. Displaying internal server error.	Informational	Properly check error log and error messages displayed on screen.

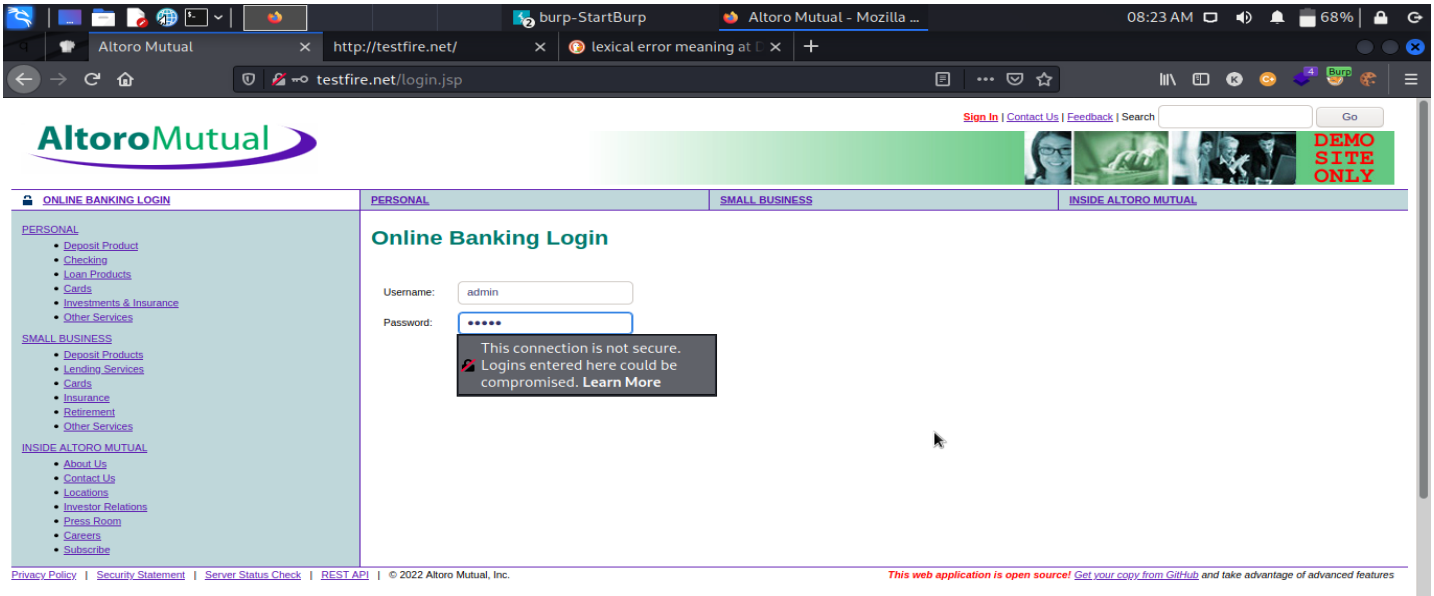
Technical Findings

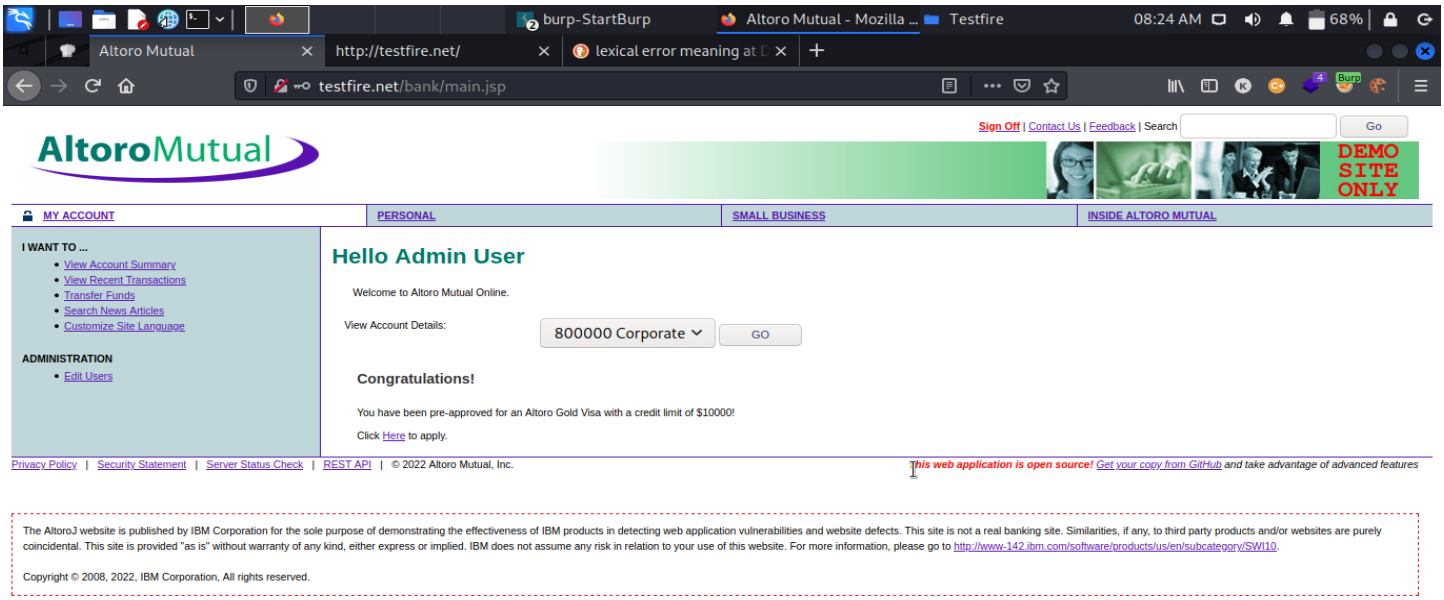
Web Application Penetration Test Findings

Finding 001: Apache Tomcat Insecure Default Administrative Password (High)

Description:	Test fire allows the use of default admin password being used on the login page due to which any unauthenticated user knowing the default password availbale on the internet can gain access to the admin account and have admin privileges.
Risk:	Likelihood: High – This attack is effective on web app and have major consequences to it. Impact: Very High – This attack gives admin privilege to a user who can make any changes on the web application.
System:	All
Tools Used:	Manually
References:	https://www.acunetix.com/vulnerabilities/web/apache-geronimo-default-administrative-credentials/

Evidence





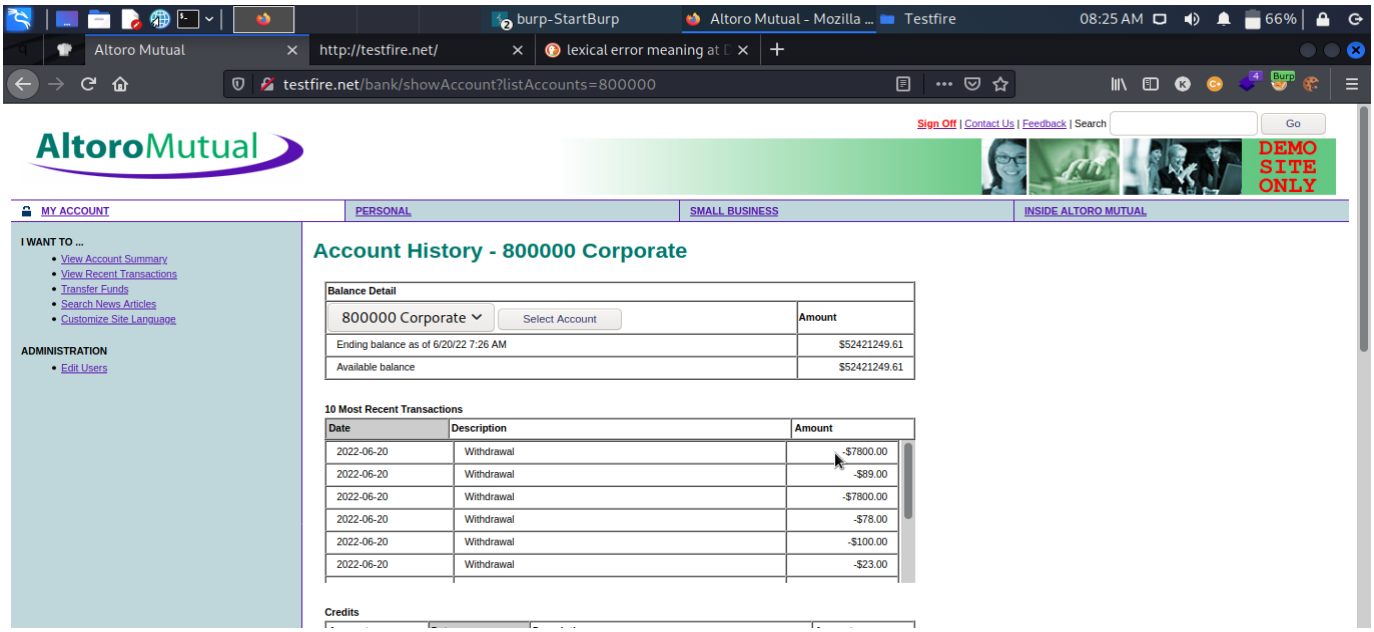
Change the Administrative Default username and password.

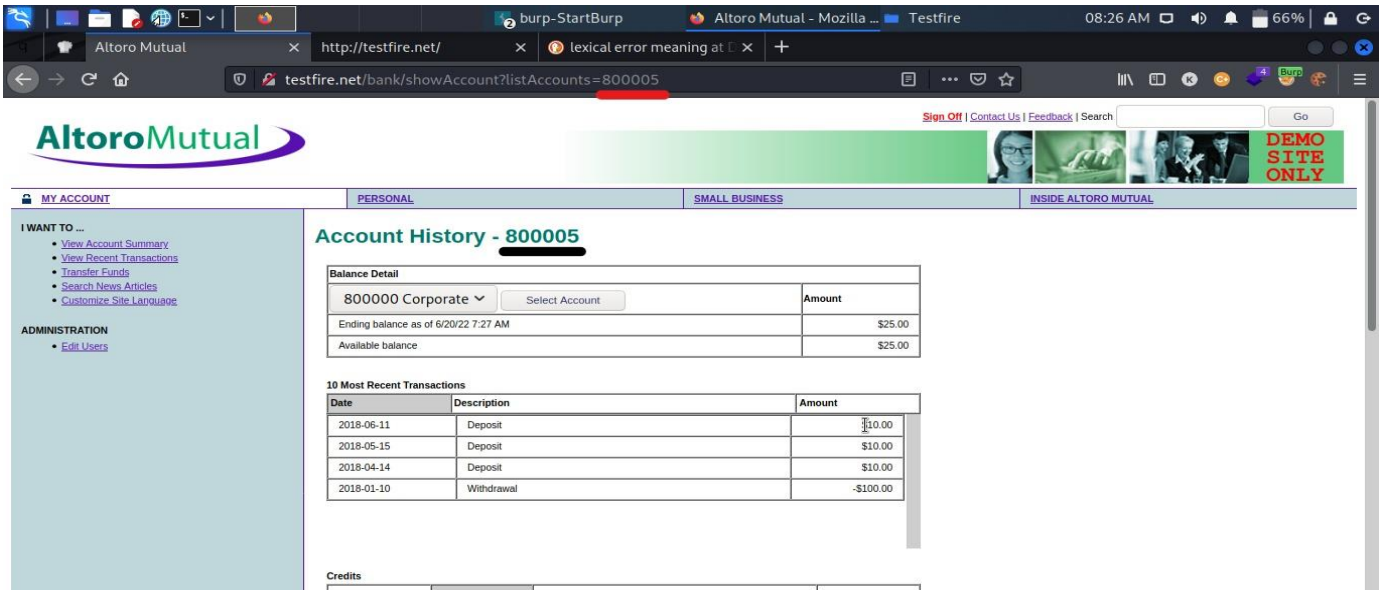
Remediation

Finding 002: Insecure Direct object Reference (High)

Description:	This vulnerability allows any user to view all account info of different user without authentication. The user just has to change the account number of the get request and he/she will be able to view sensitive account information about a diiferent user.
Risk:	Likelihood: High – This attack is effective and can display sensitive account information about a different user Impact: Very High – Changing get request directly from the allows user to view account infromation about different user.
System:	All
Tools Used:	Manually
References:	https://www.cvedetails.com/cve/CVE-2022-29627

Evidence





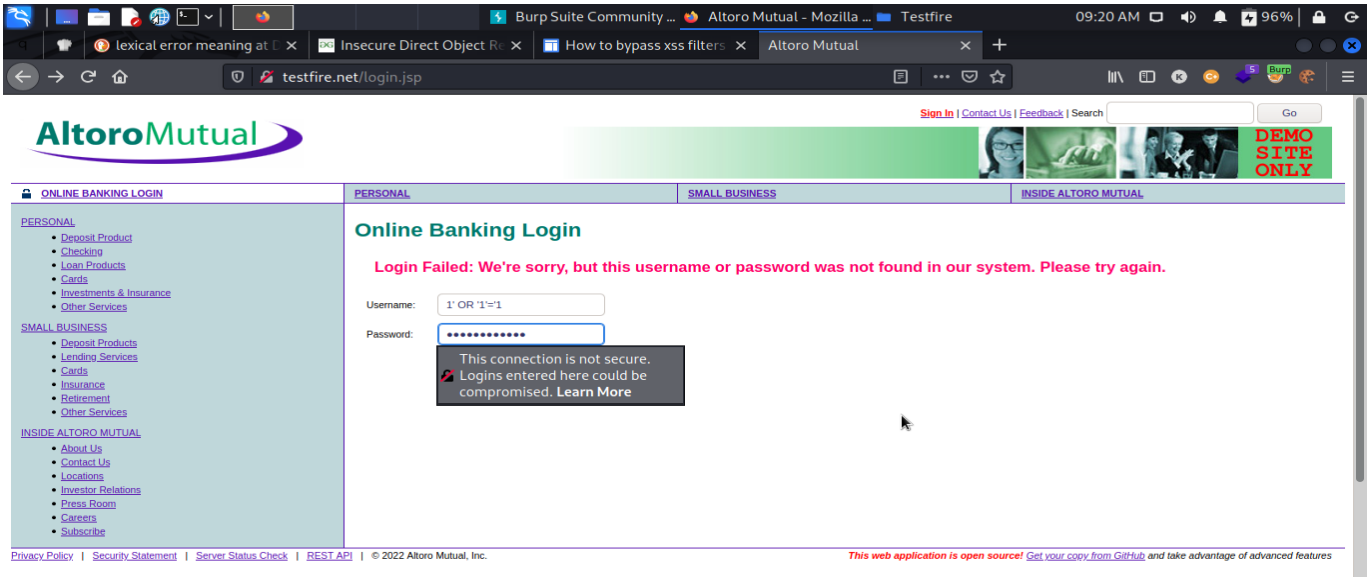
Remediation

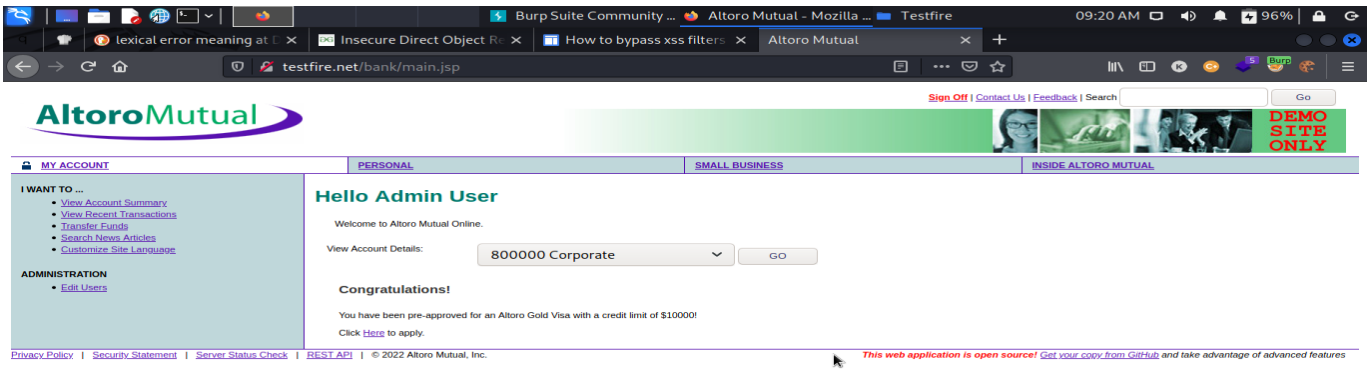
Proper implementation of security standards on the get and past request.

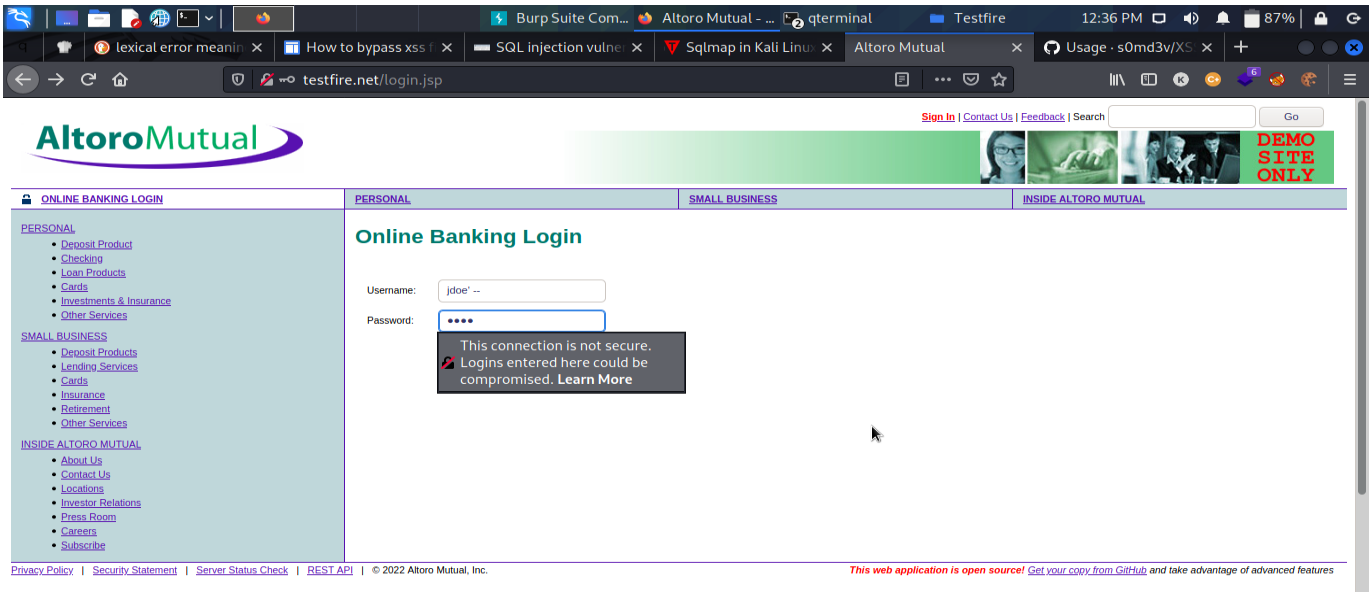
Finding 003: SQL Injection Vulnerability allowing login bypass (Critical).

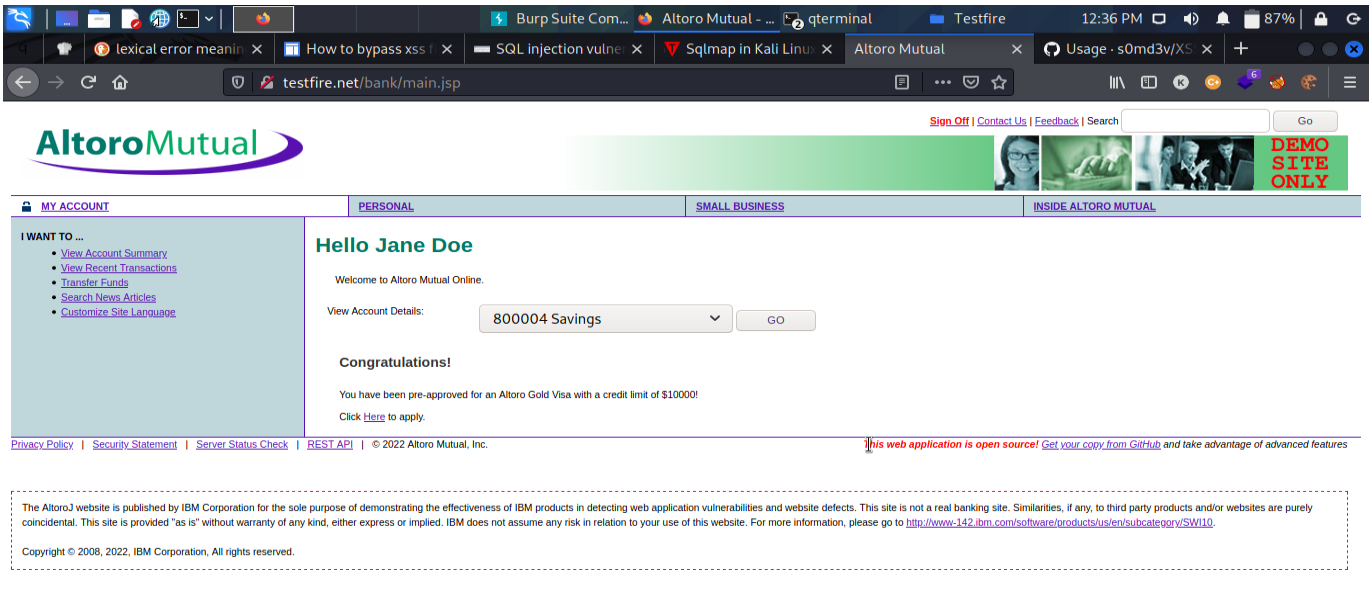
Description:	TestFire allowed a successful SQL injection attack can result in unauthorized access to admin account and user accounts as well using the same method the login security if the web app was fully compromised.
Risk:	Likelihood: Critical – This attack allowed admin as well as user login acces to the web application. Impact: Critical – After gaining admin privilege the user has all acces to the backend of the system
System:	All
Tools Used:	Manually
References:	https://portswigger.net/web-security/sql-injection

Evidence









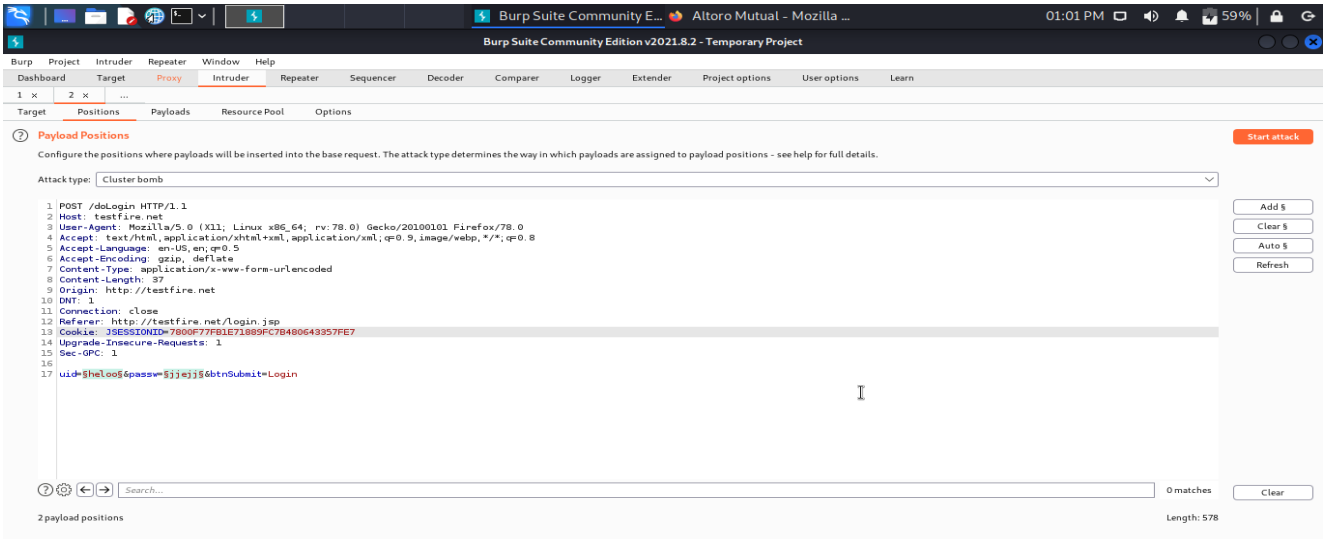
Remediation

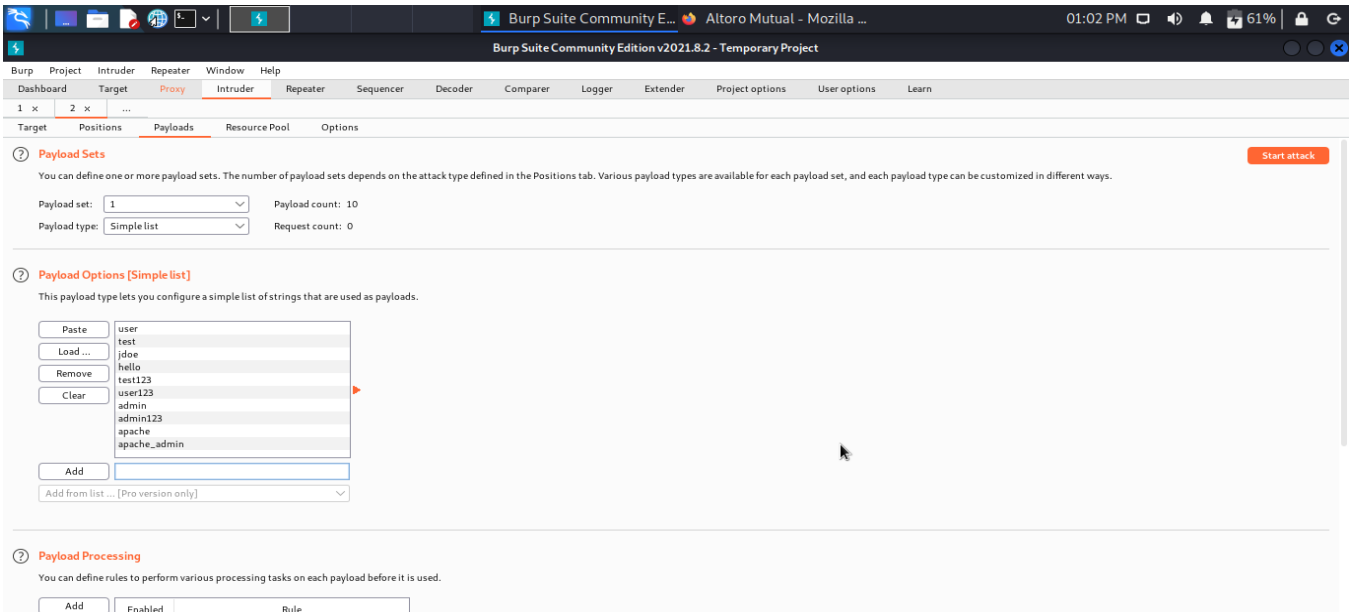
Proper Input Validation And filtering of username and password.

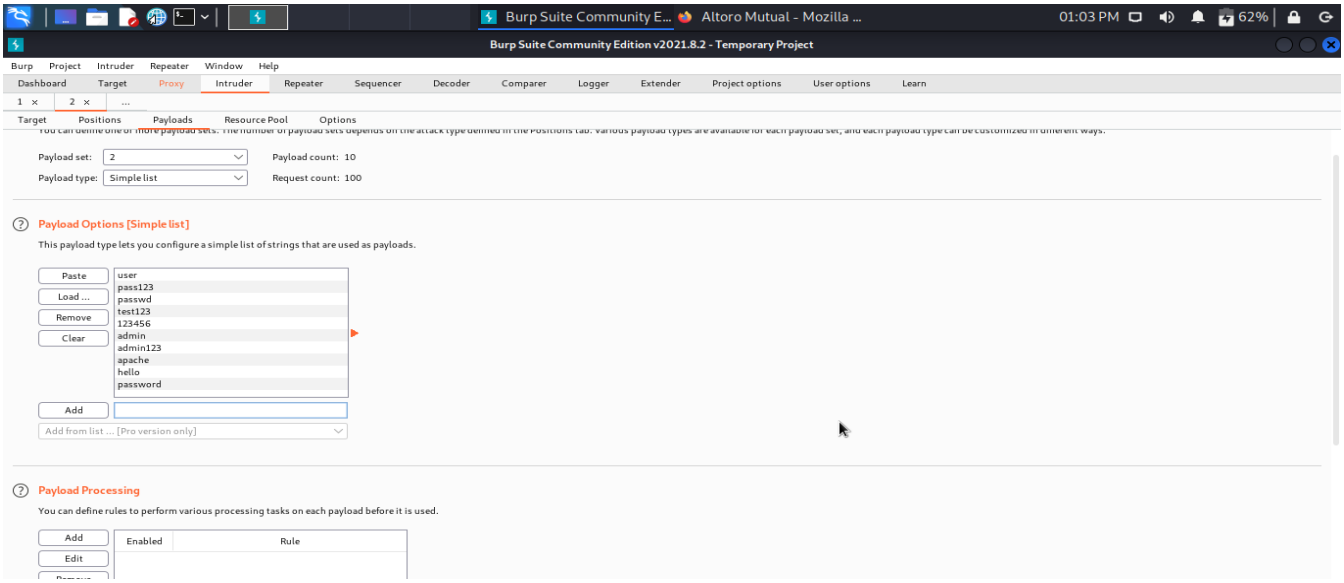
Finding 004: Login Brute force of username and password (High)

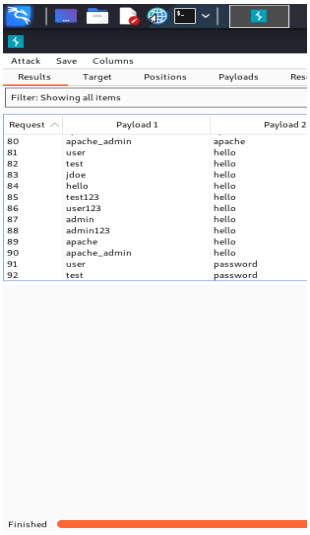
Description:	Testfire allowed multiple spraying of username and password on the web app without any restriction.
Risk:	Likelihood: High – The penetration tester sprayed hundreds of username and password on the web application. Impact: Very High – The attacker can get the username and password of the admin and users.
System:	All
Tools Used:	Burpsuite
References:	https://www.alpinesecurity.com/blog/brute-forcing-login-page-with-burp-suite/

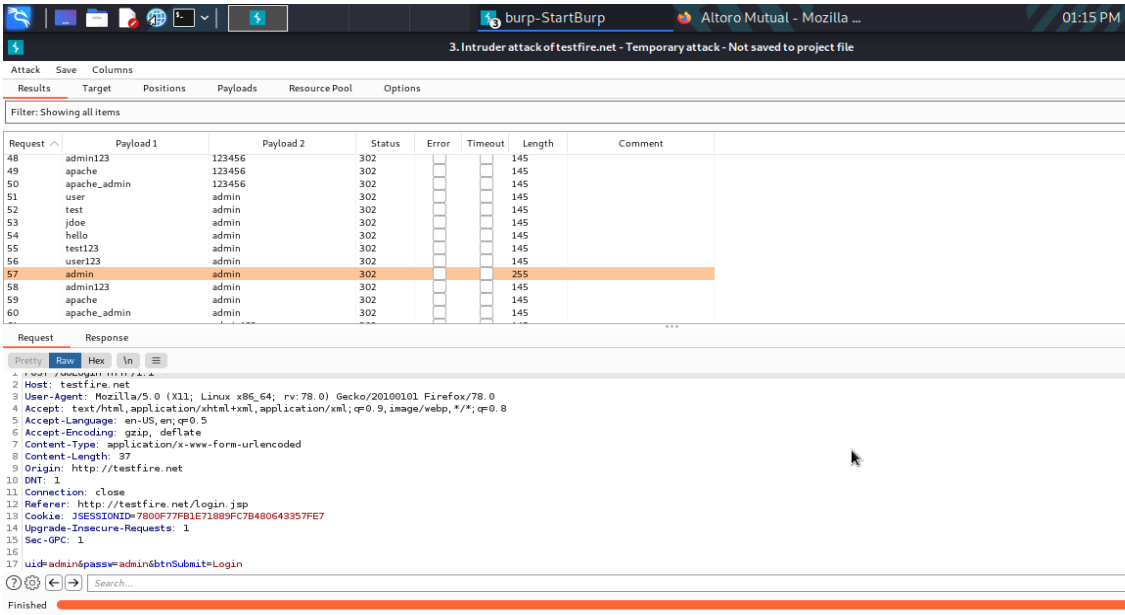
Evidence




Burp Suite Community E...
Altoro Mutual - Mozilla ...
01:02 PM
61%
Burp Suite Community Edition v2021.8.2 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
1 × 2 × ...
Target Positions Payloads Resource Pool Options
Payload Sets
Start attack
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.
Payload set: 1
Payload count: 10
Payload type: Simple list
Request count: 0
Payload Options [Simple list]
This payload type lets you configure a simple list of strings that are used as payloads.
Paste
Load ...
Remove
Clear
user
test
jdoe
hello
test123
user123
admin
admin123
apache
apache_admin
Add
Add from list ... [Pro version only]
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Enabled
Rule


Burp Suite Community E...
Altoro Mutual - Mozilla ...
01:03 PM
62%
Burp Suite Community Edition v2021.8.2 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
1 × 2 × ...
Target Positions Payloads Resource Pool Options
Payload set: 2
Payload count: 10
Payload type: Simple list
Request count: 100
Payload Options [Simple list]
This payload type lets you configure a simple list of strings that are used as payloads.
Paste
Load ...
Remove
Clear
user
pass123
passwd
test123
123456
admin
admin123
apache
hello
password
Add
Add from list ... [Pro version only]
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Edit
Enabled
Rule





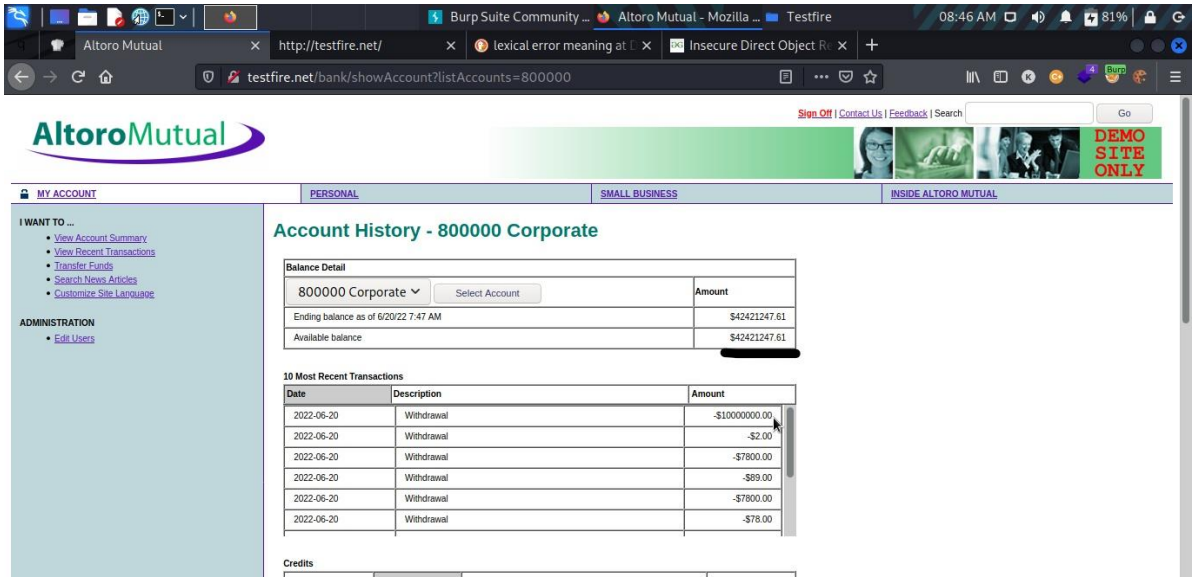
Remediation

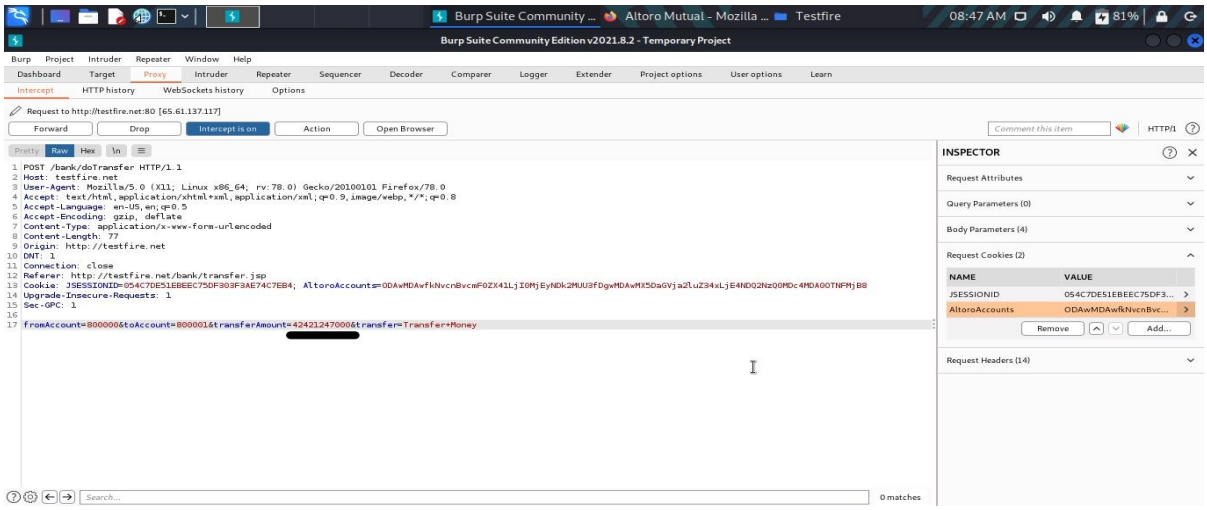
Most instances of SQL injection can be prevented by using parameterized queries (also known as prepared statements) instead of string concatenation within the query.

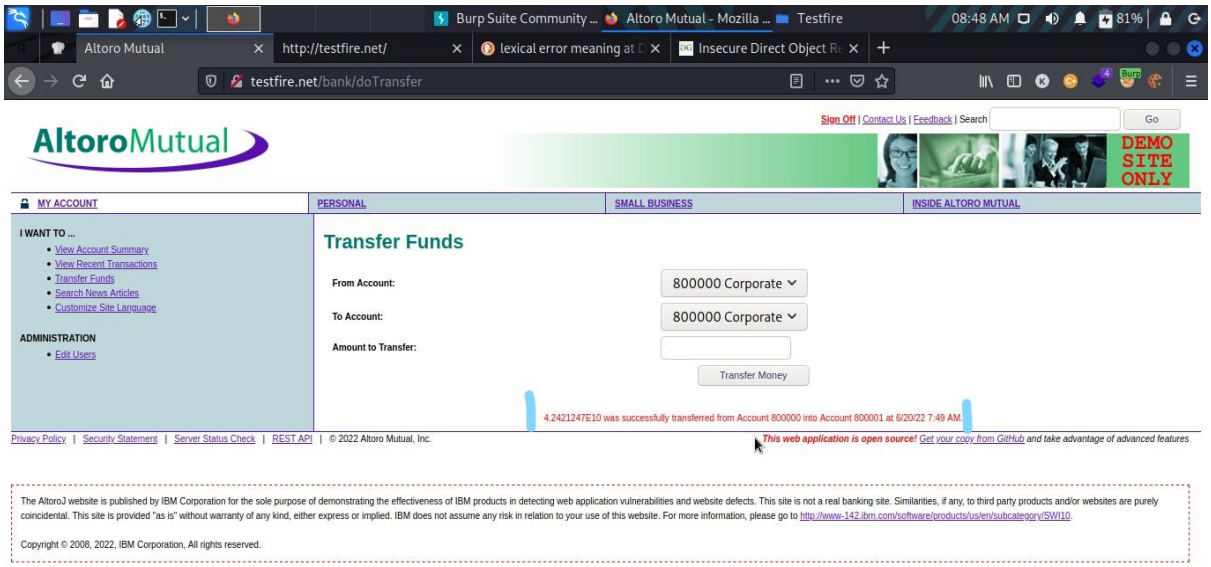
Finding -005: Improper input validation (High).

Description:	The pen tester was able to deposit a huge amount of money from his own bank account inspite of having low account balance because the amount input field was not properly validated.
Risk:	Likelihood: High – Was able to deposit huge amount in his account. Impact: Very High – can cause a huge financial loss to the organization.
System:	All
Tools Used:	Manually
References:	N/A

Evidence





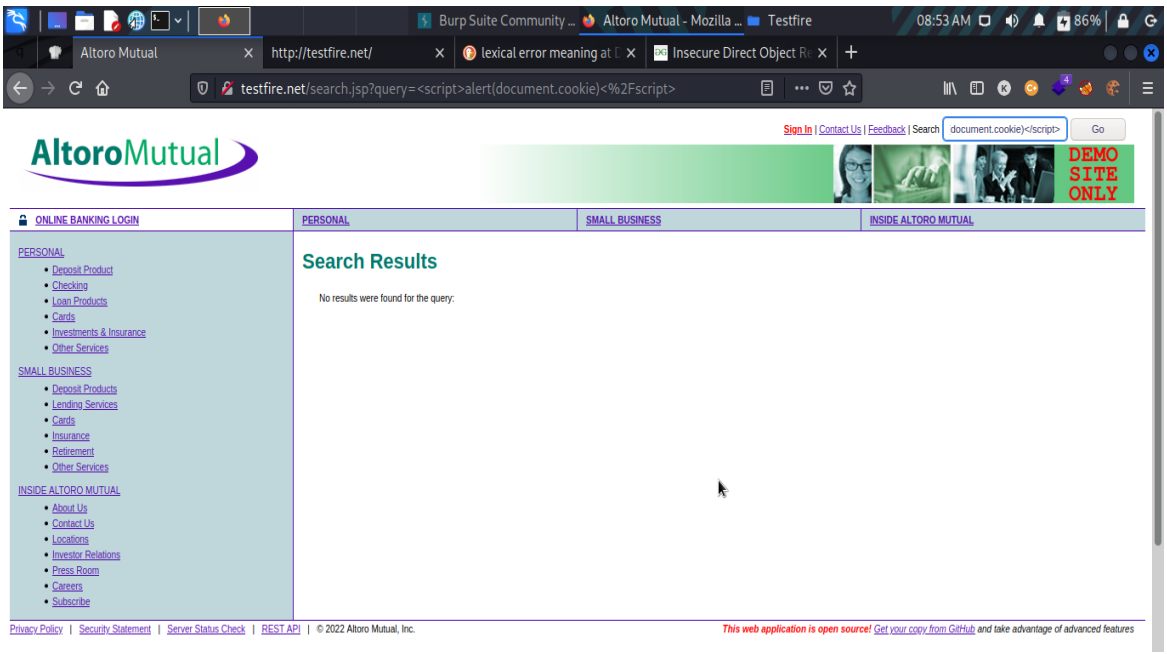


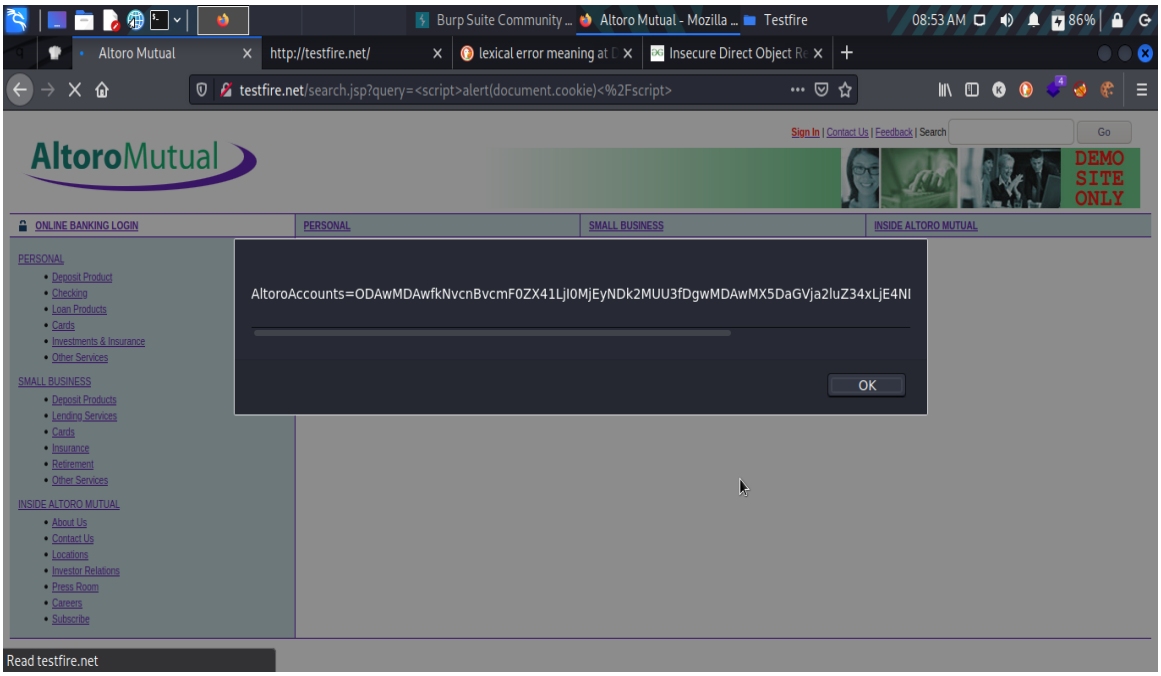
Remediation	Proper Input validation mechanism to be implemented on the amount field with some blocking filters.

Finding -006: Reflected XSS (Moderate).

Description:	The pen tester was able to send a crafted input on the search field which in result lead to a pop up alert displaying the session cookie.
Risk:	Likelihood: High – Attacker can send crafted input to users and can steal the cookie. Impact: Moderate - If exploited, an attacker can send crafted input to other user and can aquire their session cookie.
System:	All
Tools Used:	Manually
References:	https://www.cvedetails.com/cve/CVE-2022-27926

Evidence





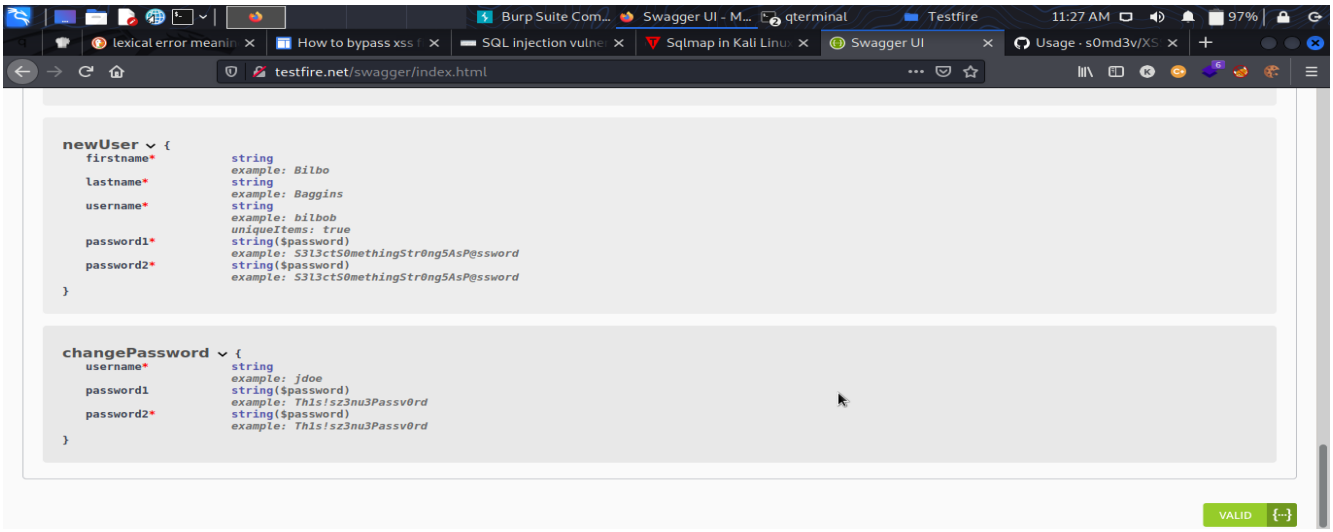
Remediation

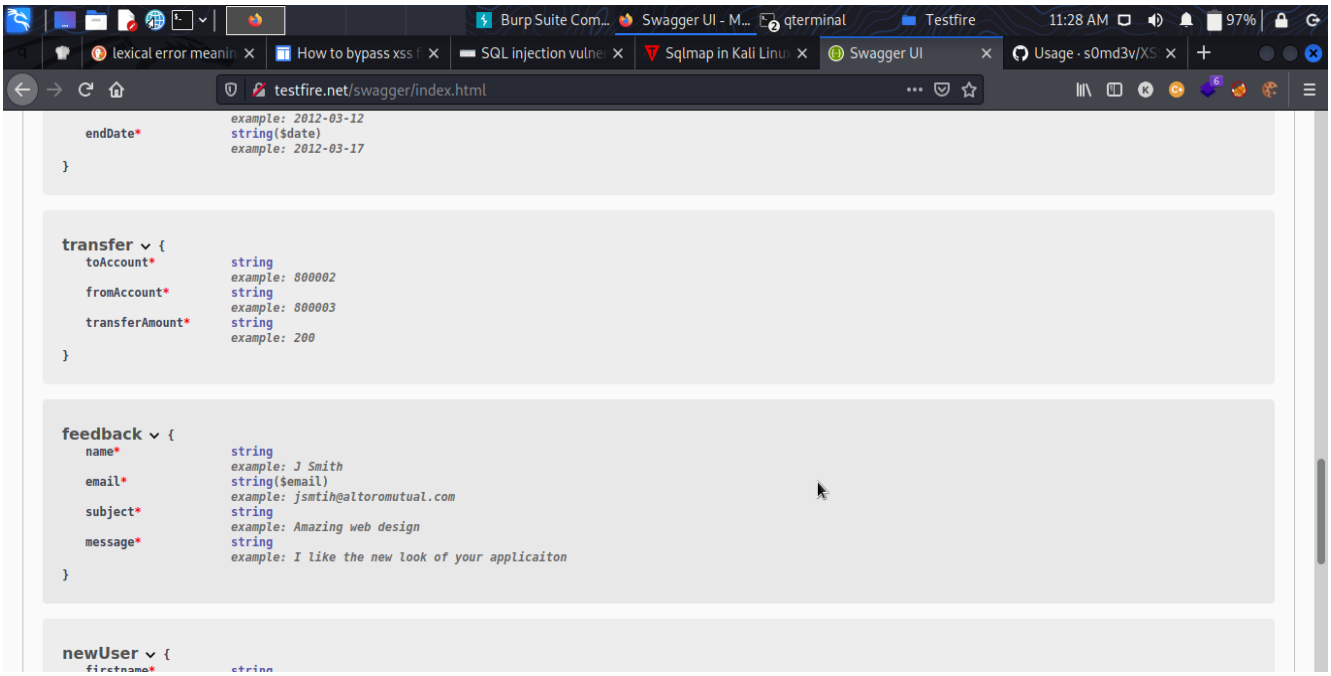
Proper implementation of input filtering on various character on the search field.

Finding -007: Displaying user on web app (Low).

Description:	Testfire displayed username and user account number on their web page which can help the attacker in crafting their attack..
Risk:	Likelihood: Low Impact: High – If Sensitive information like username and account is displayed it help in attacking the web app
System:	All
Tools Used:	Manually
References:	N/A

Evidence





Should do proper review of the web application before hosting it.

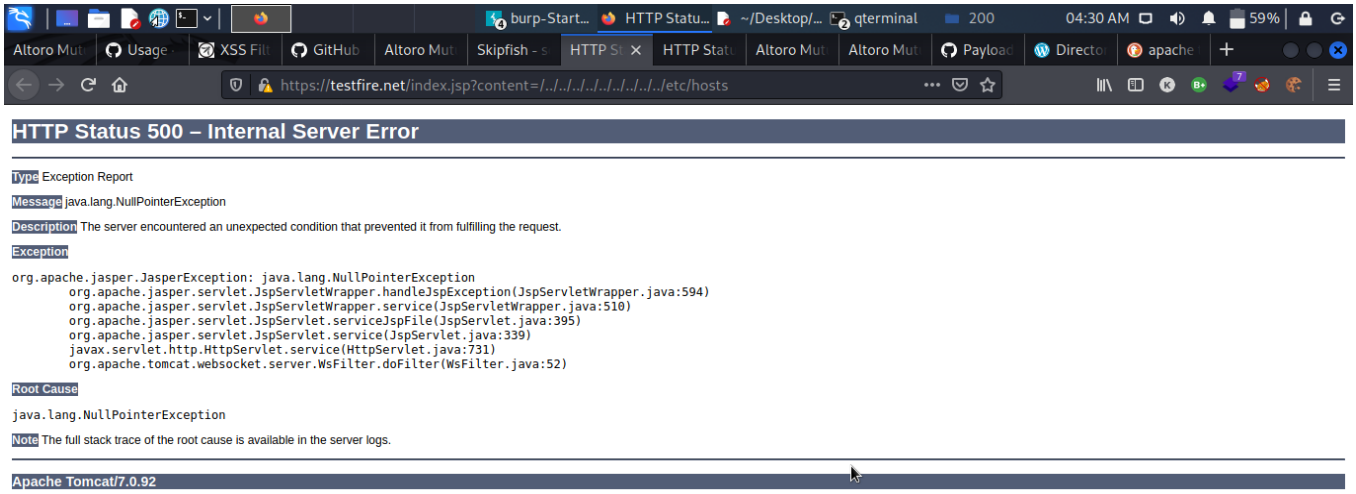
Remediation

Developers

Finding -008: Displaying internal server error (informational).

Description:	Testfire displayed the error message displayin the apache tomcat version being used in the backend which can help attacker to frame the attack accordingly.
Risk:	Likelihood: High – An attacker can discover the version of the server being used Impact: Very High – Attacker can exploit according to the version of apache being used and frame attack accordingly.
Tools Used:	Manually
References:	N/A

Evidence



Remediation

Developer should do proper review of error messages and error log before hosting the web app.