Experiment No. 9: SQL Injection

Aim: To implement SQL Injection attack

Theory:

A SQL injection attack is a type of security exploit in which an attacker injects malicious SQL code into input fields or parameters used by an application, typically a web application, to interact with a database. This malicious code can then be executed by the database server, potentially allowing the attacker to access, modify, or delete data, as well as perform other unauthorized actions.

SQL injection attacks can occur when:

1. Input Validation is Insufficient: If the application does not properly validate user input before using it in SQL queries, an attacker can exploit this vulnerability by injecting SQL commands into the input fields.

2. Improperly Constructed Queries: If the application dynamically constructs SQL queries by concatenating user input with SQL strings without proper sanitization, attackers can manipulate the query to execute unintended commands.

3. Insecure APIs: If an API endpoint accepts user input and constructs SQL queries based on that input without proper validation and sanitization, it can also be vulnerable to SQL injection attacks.

4. Stored Procedures: If stored procedures or other database objects are not properly secured and validate input, they can be exploited for SQL injection.

The consequences of a successful SQL injection attack can be severe, ranging from unauthorized access to sensitive data, data manipulation, data loss, and potentially even complete compromise of the application or database server.

Preventing SQL injection attacks involves a combination of practices such as:

1. Parameterized Queries: Using parameterized queries (prepared statements) with bound parameters instead of dynamically constructing SQL queries with user input. Parameterized queries separate SQL code from data, making it impossible for attackers to inject SQL commands.

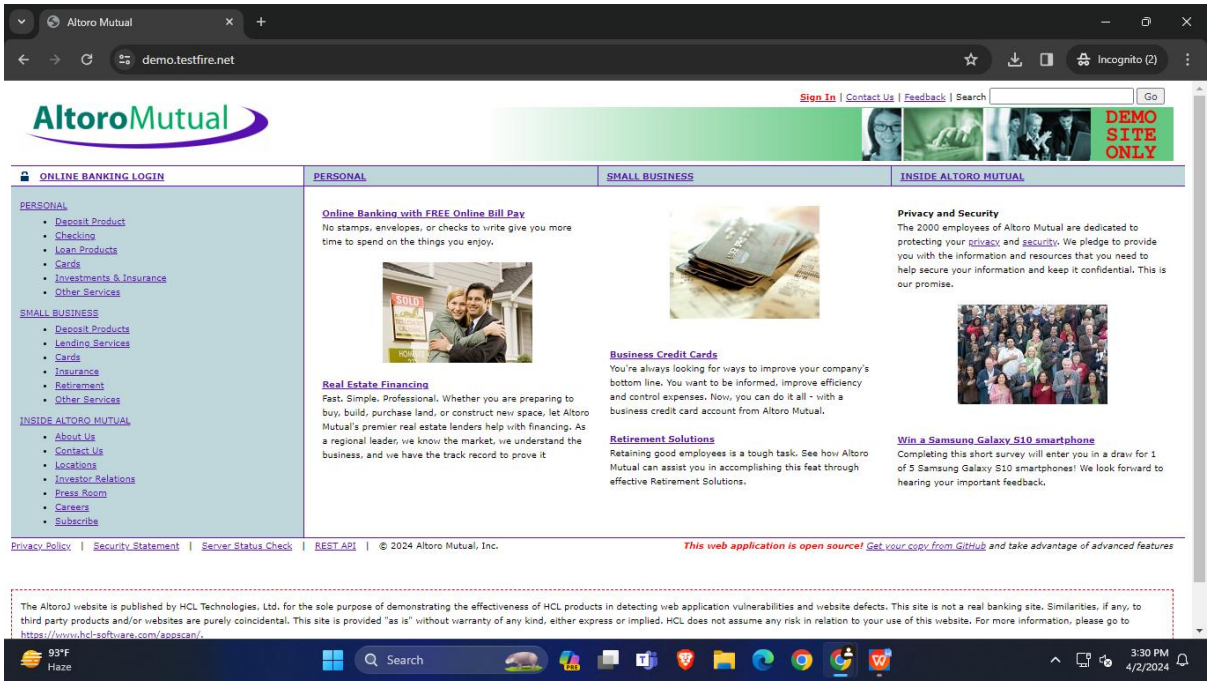
2. Input Validation and Sanitization: Ensuring that user input is properly validated and sanitized to prevent the injection of malicious SQL code. This includes techniques such as input validation, whitelisting acceptable input, and escaping special characters.

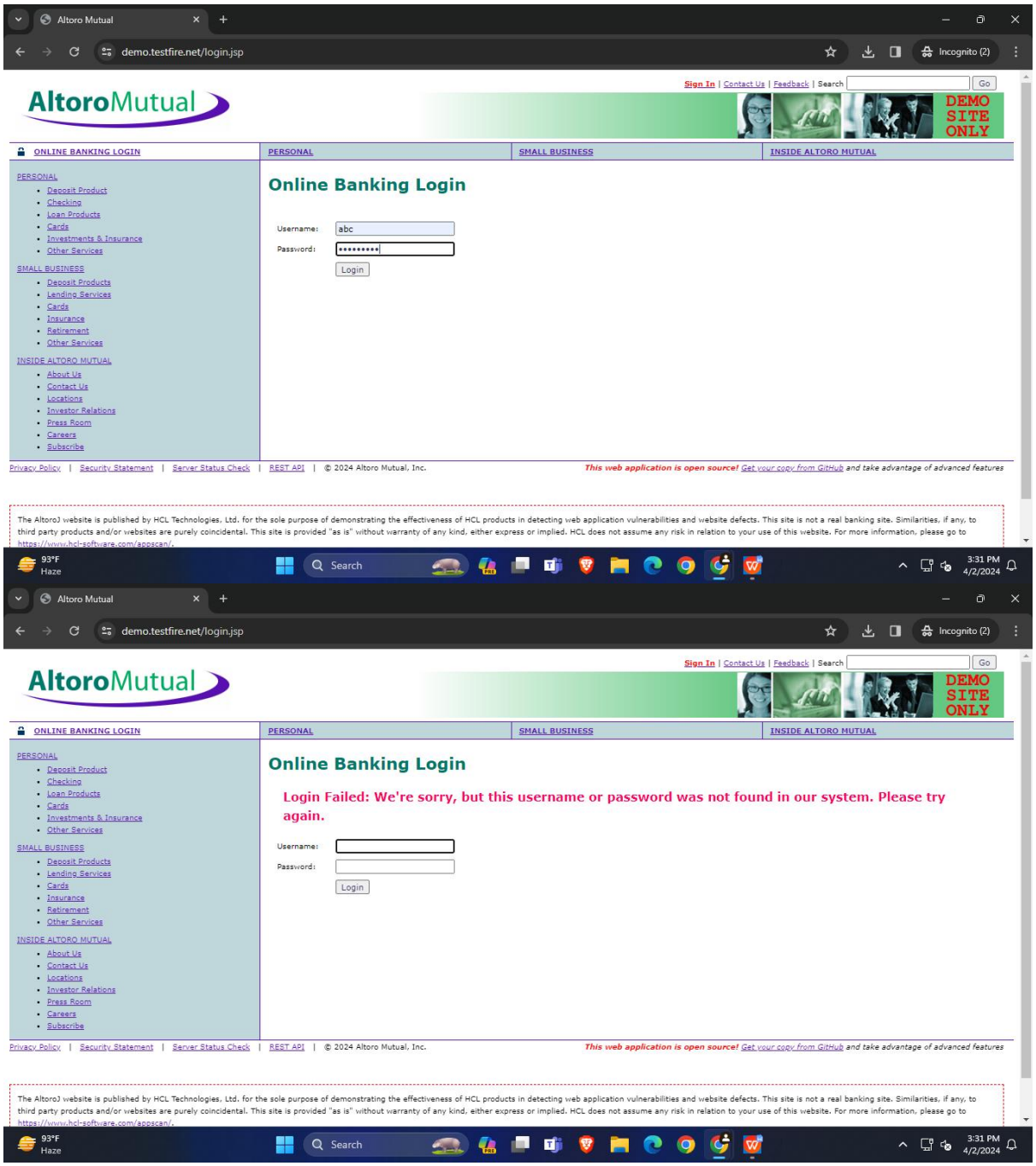
3. Least Privilege: Ensuring that database users and application components have the minimum necessary privileges to perform their functions, reducing the potential impact of a successful attack.

4. Regular Security Audits:Conducting regular security audits and vulnerability assessments to identify and address potential SQL injection vulnerabilities.

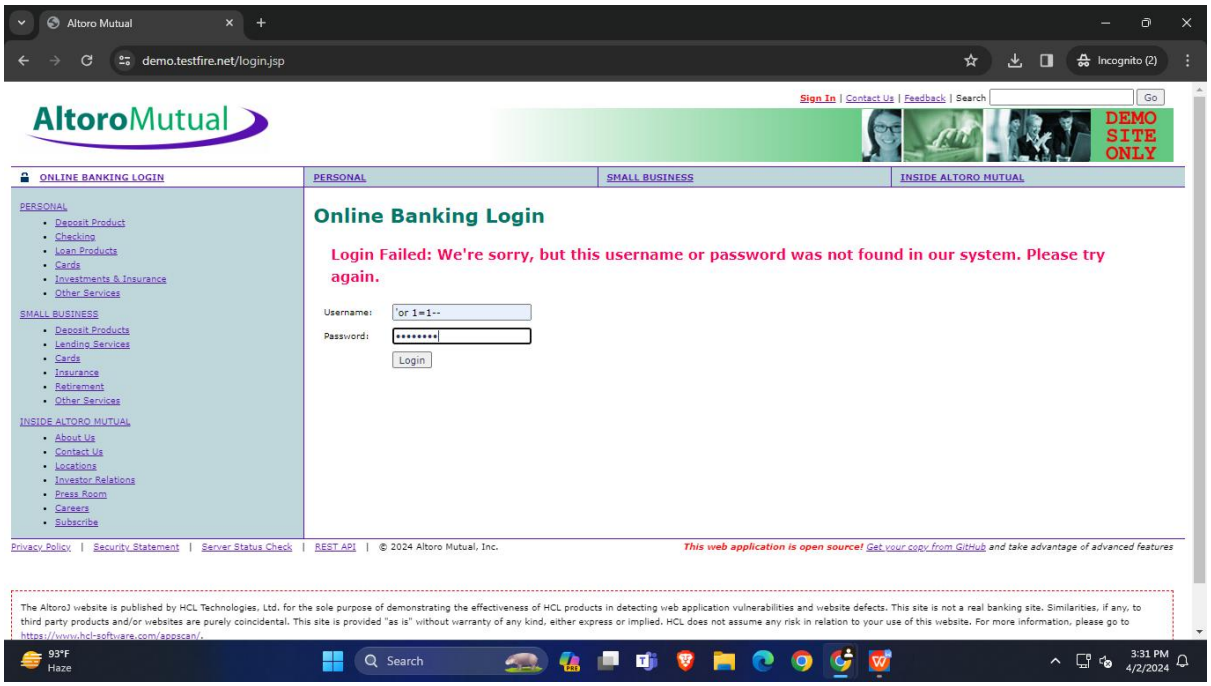
5. Using Web Application Firewalls (WAFs): Implementing WAFs that can detect and block SQL injection attempts in real-time by analyzing web traffic patterns and filtering out malicious requests.

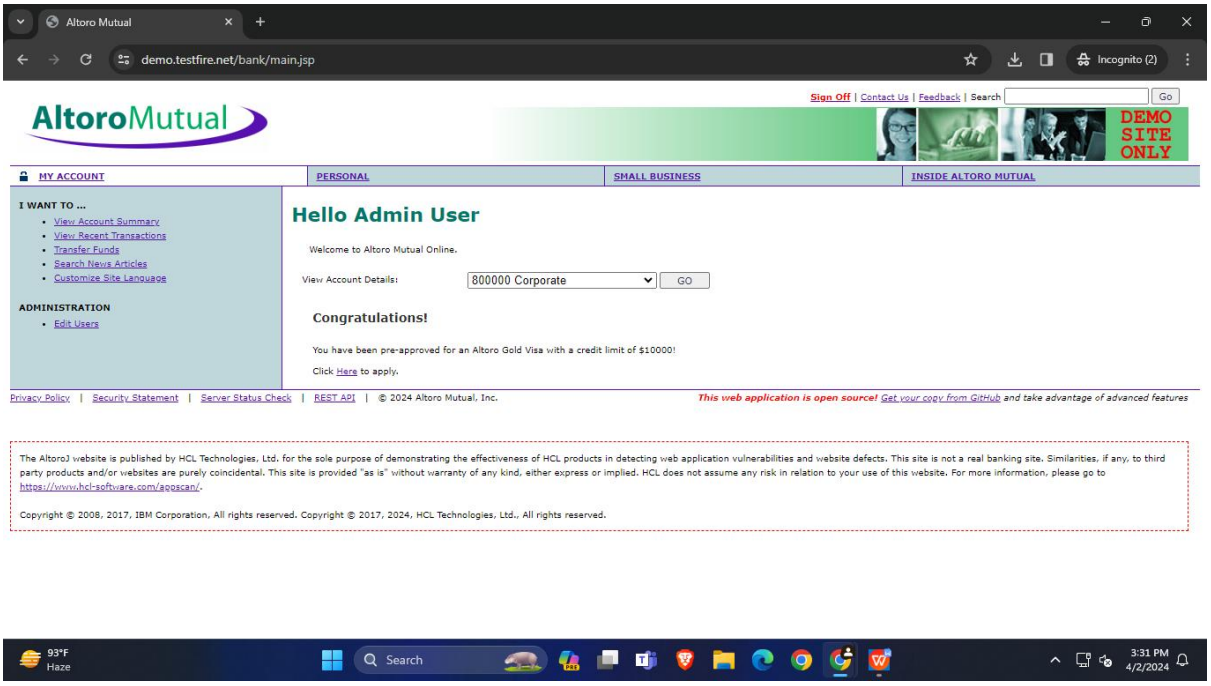
Output:

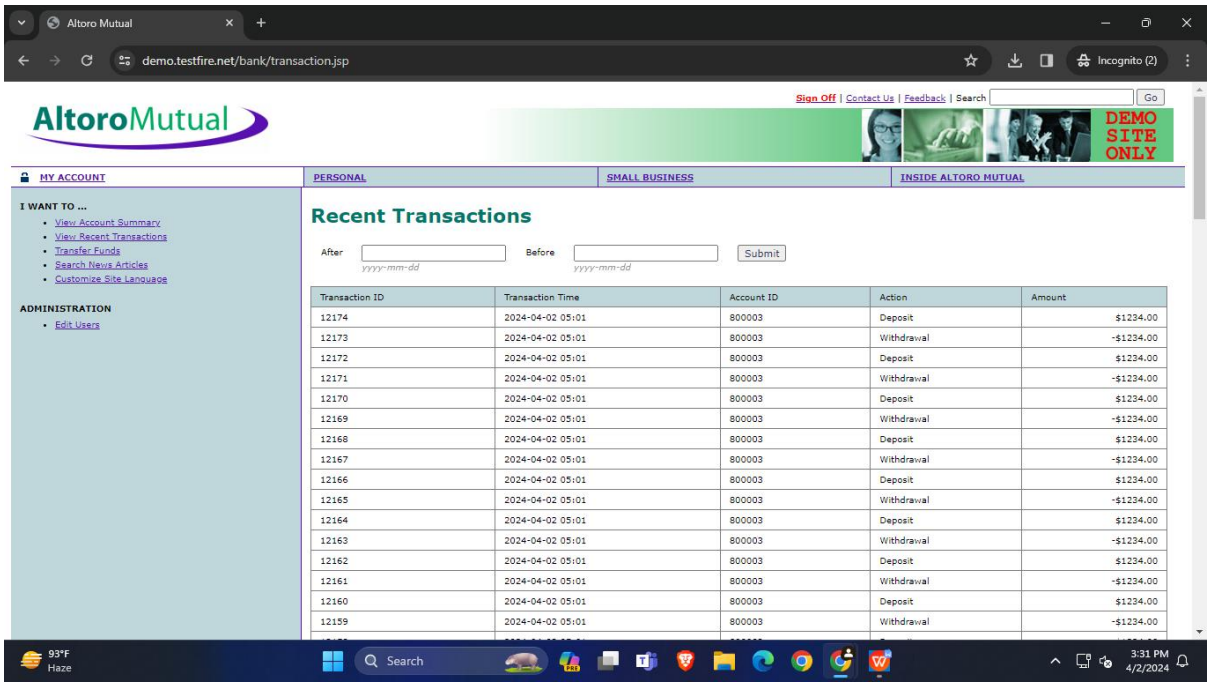



Altoro Mutual
demo.testfire.net/login.jsp
Incognito (2)
AltoroMutual
Sign In | Contact Us | Feedback | Search
Go
DEMO SITE ONLY
ONLINE BANKING LOGIN
PERSONAL
SMALL BUSINESS
INSIDE ALTORO MUTUAL
Online Banking Login
Username:
abc
Password:
Login
Deposit Product
Checking
Loan Products
Cards
Investments & Insurance
Other Services
Deposit Products
Lending Services
Cards
Insurance
Retirement
Other Services
About Us
Contact Us
Locations
Investor Relations
Press Room
Careers
Subscribe
Privacy Policy | Security Statement | Server Status Check | REST API | © 2024 Altoro Mutual, Inc.
This web application is open source! Get your copy from GitHub and take advantage of advanced features
The AltoroJ website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to https://www.hcl-software.com/appscan/.
93°F Haze
Search
3:31 PM 4/2/2024

Altoro Mutual
demo.testfire.net/login.jsp
Incognito (2)
AltoroMutual
Sign In | Contact Us | Feedback | Search
Go
DEMO SITE ONLY
ONLINE BANKING LOGIN
PERSONAL
SMALL BUSINESS
INSIDE ALTORO MUTUAL
Online Banking Login
Login Failed: We're sorry, but this username or password was not found in our system. Please try again.
Username:
Password:
Login
Deposit Product
Checking
Loan Products
Cards
Investments & Insurance
Other Services
Deposit Products
Lending Services
Cards
Insurance
Retirement
Other Services
About Us
Contact Us
Locations
Investor Relations
Press Room
Careers
Subscribe
Privacy Policy | Security Statement | Server Status Check | REST API | © 2024 Altoro Mutual, Inc.
This web application is open source! Get your copy from GitHub and take advantage of advanced features
The AltoroJ website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to https://www.hcl-software.com/appscan/.
93°F Haze
Search
3:31 PM 4/2/2024


Altoro Mutual
demo.testfire.net/login.jsp
AltoroMutual
Sign In | Contact Us | Feedback | Search
DEMO SITE ONLY
ONLINE BANKING LOGIN
PERSONAL
SMALL BUSINESS
INSIDE ALTORO MUTUAL
Online Banking Login
Login Failed: We're sorry, but this username or password was not found in our system. Please try again.
Username:
'or 1=1--
Password:
Login
Privacy Policy | Security Statement | Server Status Check | REST API | © 2024 Altoro Mutual, Inc.
This web application is open source! Get your copy from GitHub and take advantage of advanced features
The AltoroJ website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to https://www.hcl-software.com/appscan/.
93°F Haze
Search
3:31 PM 4/2/2024


Altoro Mutual
demo.testfire.net/bank/main.jsp
AltoroMutual
Sign Off | Contact Us | Feedback | Search
DEMO SITE ONLY
MY ACCOUNT
PERSONAL
SMALL BUSINESS
INSIDE ALTORO MUTUAL
I WANT TO ...
View Account Summary
View Recent Transactions
Transfer Funds
Search News Articles
Customize Site Language
ADMINISTRATION
Edit Users
Hello Admin User
Welcome to Altoro Mutual Online.
View Account Details:
800000 Corporate
GO
Congratulations!
You have been pre-approved for an Altoro Gold Visa with a credit limit of $10000!
Click Here to apply.
Privacy Policy | Security Statement | Server Status Check | REST API | © 2024 Altoro Mutual, Inc.
This web application is open source! Get your copy from GitHub and take advantage of advanced features
The AltoroJ website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to https://www.hcl-software.com/appscan/.
Copyright © 2008, 2017, IBM Corporation, All rights reserved. Copyright © 2017, 2024, HCL Technologies, Ltd., All rights reserved.
93°F Haze
Search
3:31 PM 4/2/2024



Conclusion: In conclusion, SQL injection attacks pose a significant threat to the security of web applications and databases, potentially leading to unauthorized access, data manipulation, and even complete compromise of systems. Preventative measures such as parameterized queries, input validation, and least privilege access are crucial in mitigating these risks. Regular security audits and the use of Web Application Firewalls further enhance the defense against SQL injection vulnerabilities. By implementing these best practices, organizations can bolster their defenses and safeguard their systems against this prevalent and damaging form of cyber attack.