Assignment - 4

OWASP Top 10 Web Application Security Risks

The Open Web Application Security Project (OWASP) publishes a critical list of the ten most prevalent web application security risks. Understanding these vulnerabilities and their potential consequences is essential for securing web applications.

1. Injection Attacks

Injection attacks occur when untrusted data is introduced into a query or command. Attackers can exploit these flaws to manipulate application behavior, steal sensitive data, or gain unauthorized access to systems. Common examples include SQL injection and NoSQL injection.

2. Broken Authentication

Weak authentication mechanisms can leave applications vulnerable to various attacks. Attackers can leverage these weaknesses to steal credentials, hijack sessions, or gain unauthorized access through brute-force attacks.

3. Sensitive Data Exposure

Failure to adequately protect sensitive data (passwords, credit card numbers, personal information) can result in data breaches and privacy violations. Attackers can exploit these vulnerabilities to steal data, commit identity theft, or conduct financial fraud.

4. XML External Entities (XXE)

XXE attacks exploit vulnerabilities in applications that process XML data containing external entity references. Attackers can leverage these vulnerabilities to read sensitive files, launch denial-of-service attacks, or perform server-side request forgery (SSRF).

5. Broken Access Control

Inadequate access control mechanisms can allow unauthorized users to access restricted functionality or sensitive data. Attackers can exploit these weaknesses to escalate privileges, bypass authorization checks, or

manipulate access controls for unauthorized access.

6. Security Misconfigurations

Improper security settings (default passwords, unnecessary services, excessive permissions) create vulnerabilities that attackers can exploit to compromise the application or underlying infrastructure.

7. Cross-Site Scripting (XSS)

XSS vulnerabilities arise when untrusted data is included in web pages without proper validation or escaping. Attackers can exploit these vulnerabilities to inject malicious scripts that steal session cookies, deface websites, or redirect users to malicious sites.

8. Insecure Deserialization

Insecure deserialization vulnerabilities occur when applications deserialize untrusted data without proper validation or integrity checks. Attackers can exploit these vulnerabilities to execute arbitrary code, perform object injection attacks, or tamper with application logic.

9. Using Components with Known Vulnerabilities

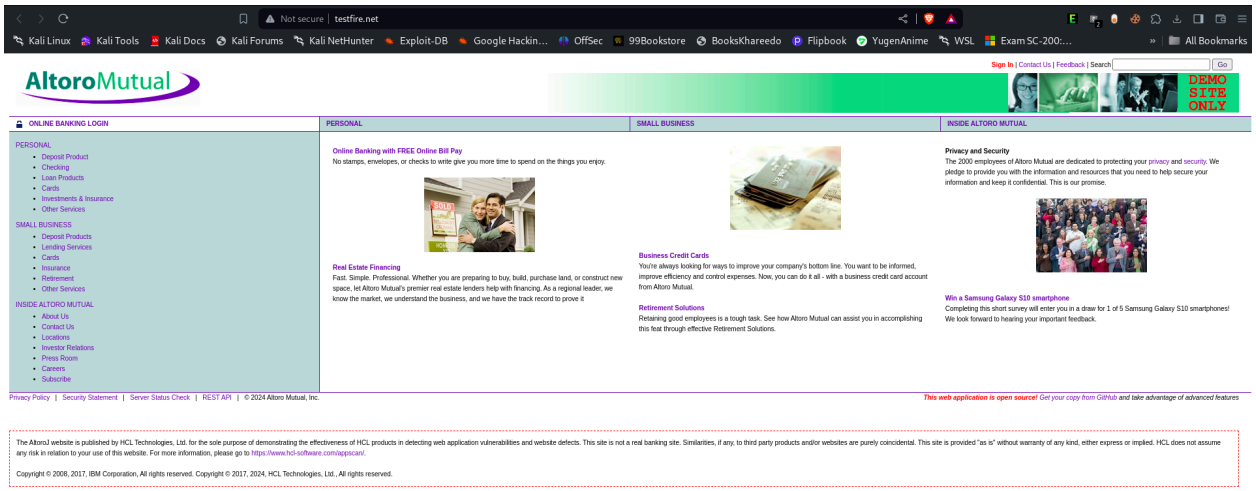
Integrating third-party components with known security vulnerabilities can expose the application to exploitation. Attackers can leverage these vulnerabilities to compromise the application, escalate privileges, or gain unauthorized access to the underlying system.

10. Insufficient Logging and Monitoring

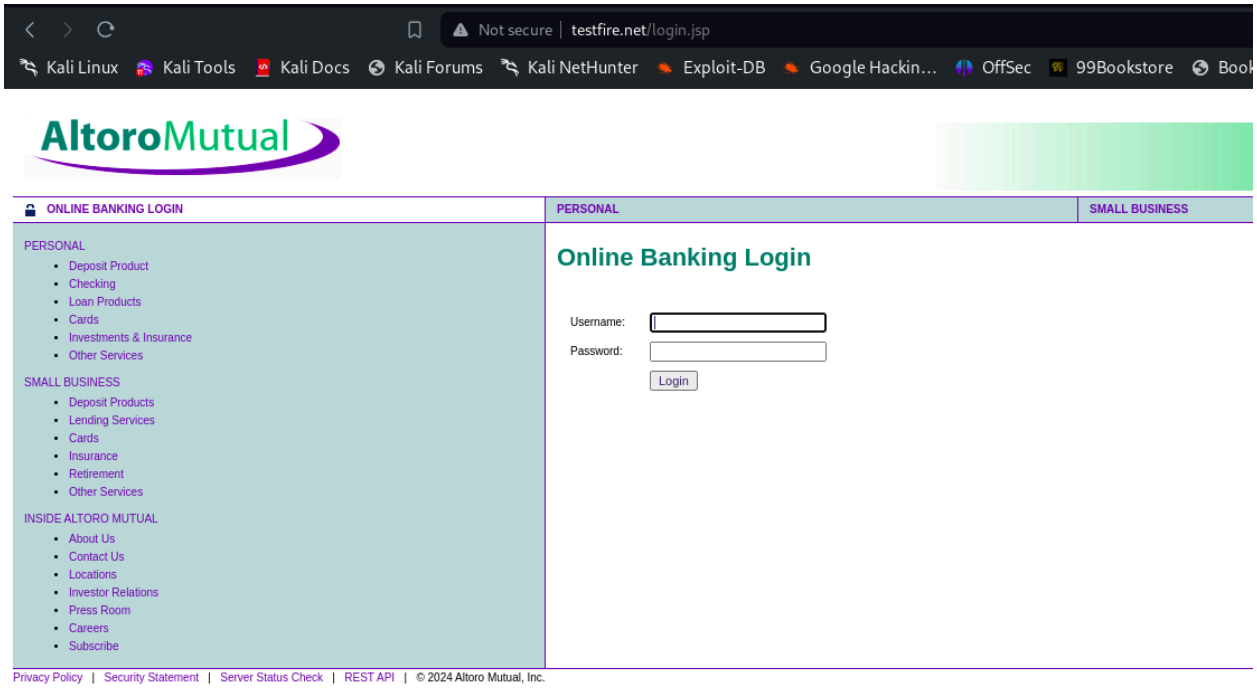
Inadequate logging and monitoring practices can hinder the detection and response to security incidents. Attackers can exploit these deficiencies to conduct stealthy attacks, evade detection, or maintain persistence within the environment.

Altoro Mutual Website Security Analysis

The AltoroMutual website offers various services such as banking, investment, and insurance. It provides features like user registration, login, payment portal, contact forms, and other interactive functionalities. Users can manage accounts, make transactions, and interact with customer support.



By click on the online banking login we can find the login page

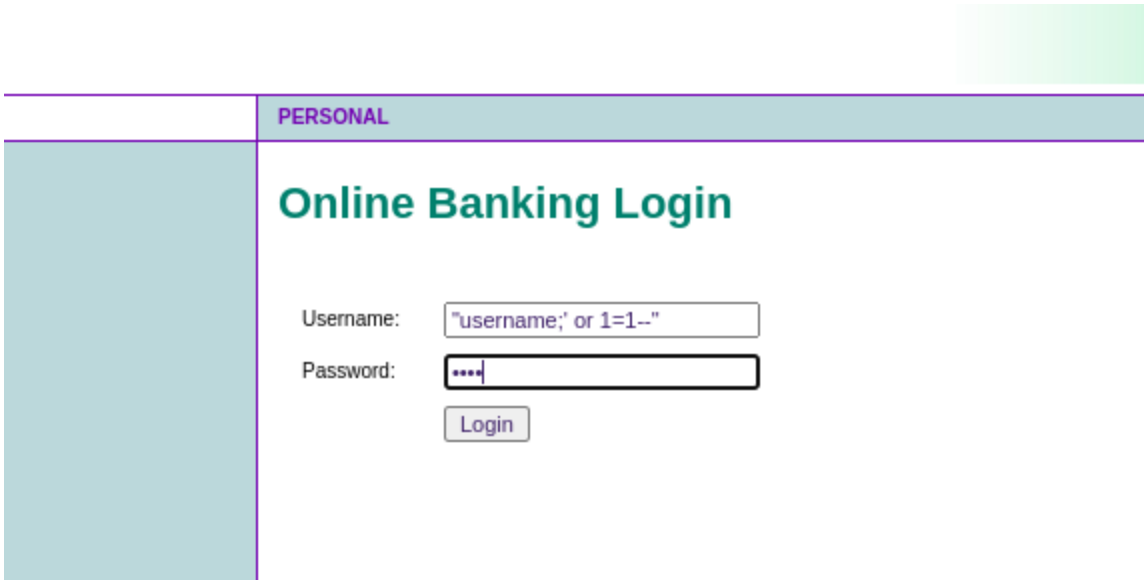


Vulnerability Identification Report

we will analyze vulnerabilities discovered in the system, focusing on sensitive data exposure, insecure direct object references (IDOR), broken access control, and cross-site scripting (XSS). These vulnerabilities pose significant risks to the confidentiality, integrity, and availability of the system and its data.

Sensitive Data Exposure

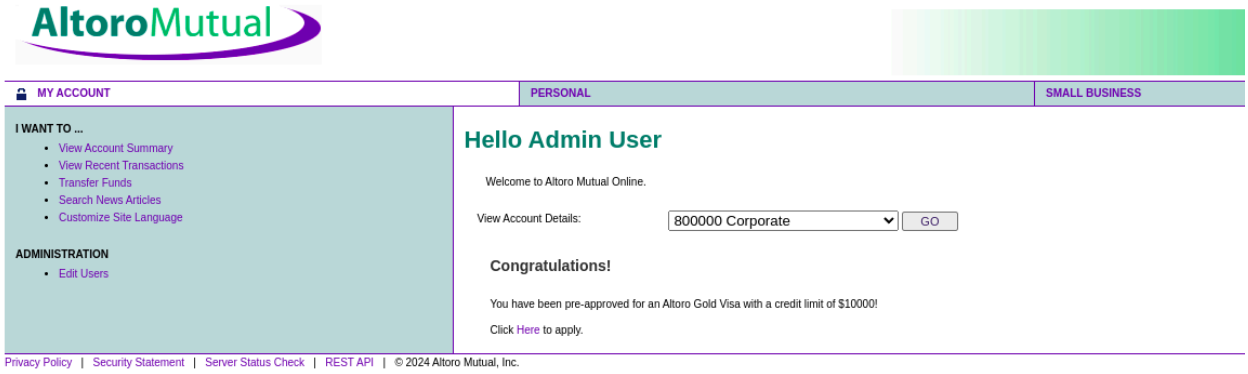
- **Description:** Upon attempting to log in with the credentials "username;' or 1=1--" and password "1234", we successfully accessed an unknown user's account, revealing sensitive data.



- **Impact:** This vulnerability exposes sensitive information such as passwords, credit card numbers, or personal identifiable information (PII).

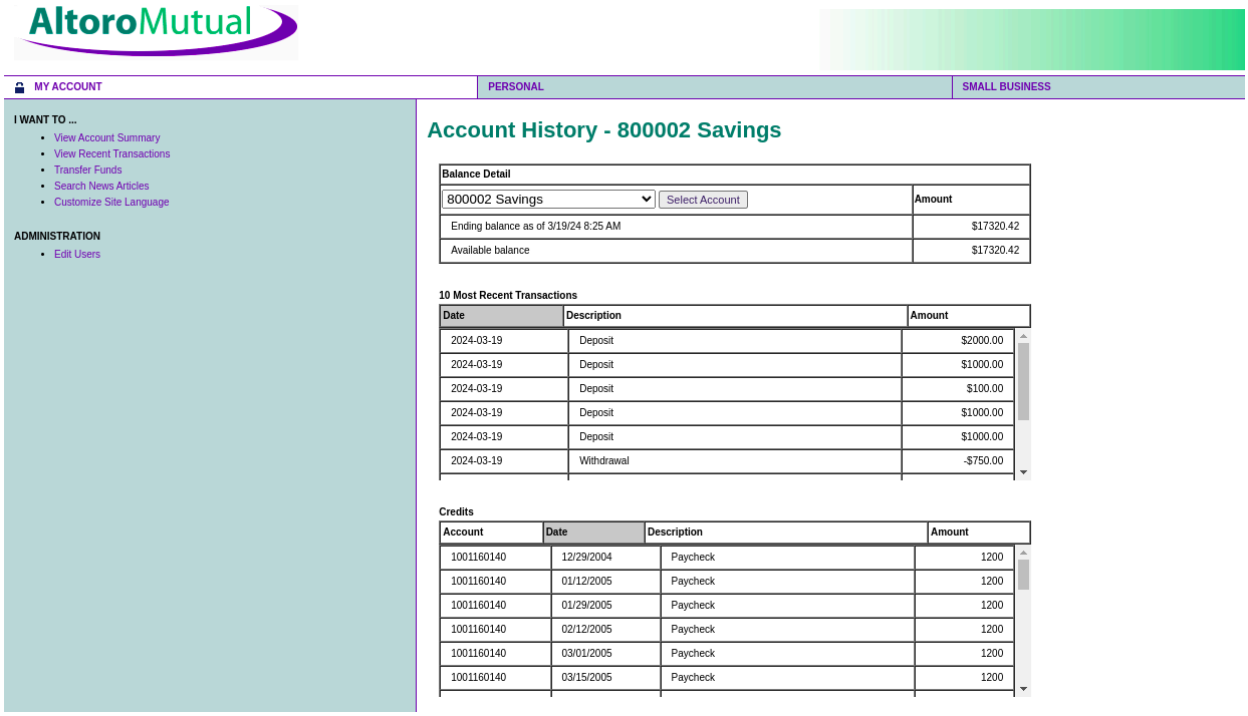
- **Risk:** Attackers could exploit this vulnerability to obtain and misuse sensitive data, leading to potential financial or reputational damage.

- **Recommendation:** Implement robust encryption, secure storage mechanisms, and strict access controls to protect sensitive information effectively.



Insecure Direct Object References (IDOR)

- **Description:** Accessing the payment portal of another user indicates an insecure direct object references (IDOR) vulnerability.



- **Impact:** This vulnerability allows unauthorized access to internal objects, compromising user data and credentials.

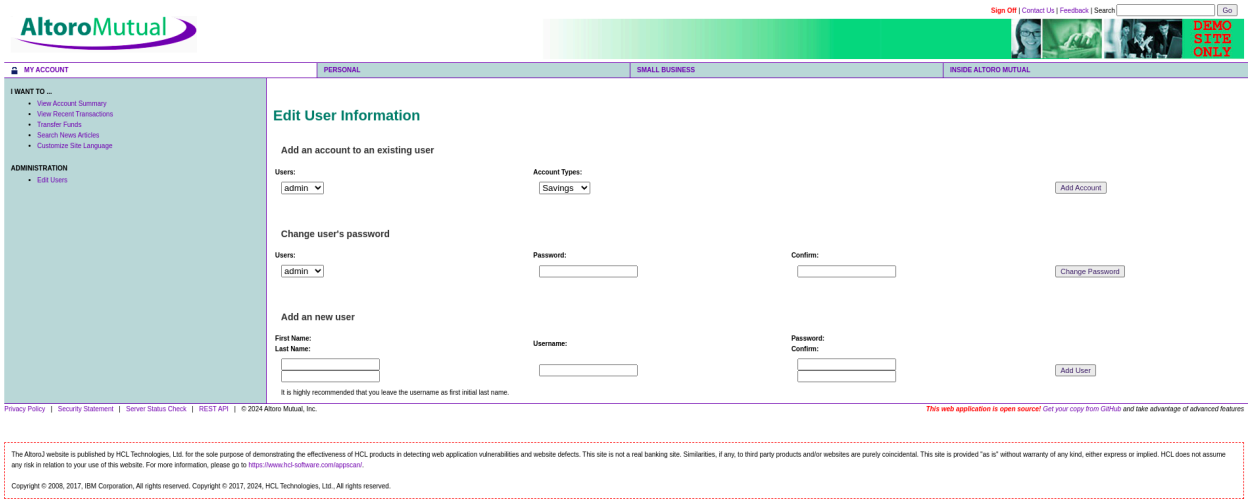
- **Risk:** Attackers can manipulate parameters or URLs to access unauthorized data,

bypassing intended access controls.

- **Recommendation:** Implement proper authorization checks and access controls to prevent unauthorized access to sensitive data.

Broken Access Control

- **Description:** Editing user details without proper authentication or authorization indicates a broken access control vulnerability.



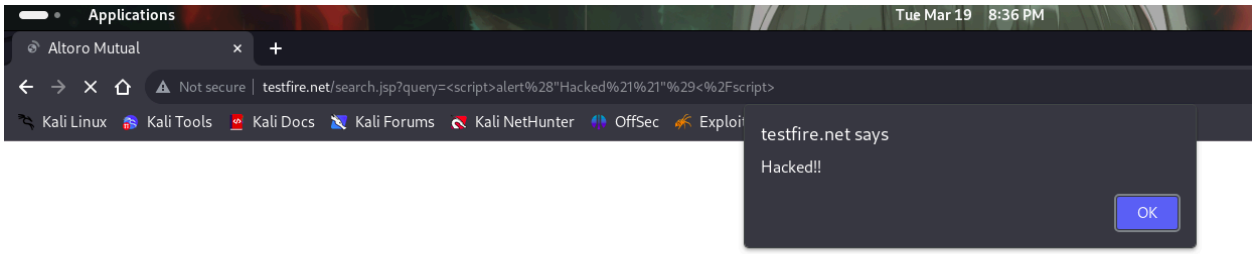
- **Impact:** This vulnerability enables attackers to compromise user accounts, session tokens, or passwords.

- **Risk:** Unauthorized access to accounts or manipulation of user data can lead to various security breaches.

- **Recommendation:** Strengthen authentication mechanisms and enforce strict access controls to mitigate the risk of unauthorized access.

Cross-Site Scripting (XSS)

- **Description:** Injecting malicious scripts into the site via the search bar using the payload “<script>alert(“Hacked!!”)</script>” demonstrates a cross-site scripting (XSS) vulnerability.



- **Impact:** This vulnerability allows attackers to execute malicious scripts in the context of other users' browsers.

- **Risk:** Attackers can steal session cookies, perform actions on behalf of users, or deface the website, compromising its integrity.

- **Recommendation:** Implement input validation and output encoding to prevent XSS attacks and ensure the security of web applications.