Top 5 OWASP Vulnerabilities

1. Injection (A9): This vulnerability occurs when untrusted data is injected into an application's input fields. This can be done by an attacker through a variety of ways, such as through a web form, a file upload, or an API call. Once the malicious data is injected, it can be executed by the application, which can lead to a variety of impacts, such as data theft, system compromise, and denial of service attacks.

2. Broken Authentication and Session Management (A10): This vulnerability occurs when an application's authentication or session management mechanisms are not implemented correctly. This can allow an attacker to gain unauthorized access to the application, even if they do not have the correct credentials. Once an attacker has gained unauthorized access, they can steal sensitive data, modify data, or disrupt the application's functionality.

3. Insecure Deserialization (A8): This vulnerability occurs when untrusted data is deserialized without proper validation. This can allow an attacker to execute malicious code on the application's server. Serialization is the process of converting an object into a format that can be stored or transmitted. Deserialization is the process of converting a stored or transmitted object back into its original form. If an attacker can inject malicious code into serialized data, and that data is then deserialized by the application, the malicious code will be executed on the server.

4. Cross-Site Scripting (XSS) (A7): This vulnerability occurs when an attacker can inject malicious code into a web page. This malicious code is then executed by the victim's browser when the page is loaded. XSS can be used to steal sensitive data, such as cookies or session tokens, or to redirect the victim to a malicious website.

5. Security Misconfiguration (A4): This vulnerability occurs when a web application is not configured correctly. This can leave the application exposed to a variety of attacks, such as unauthorized access, data theft, and denial of service attacks. Security misconfiguration can occur in a variety of ways, such as using weak passwords, not patching known vulnerabilities, or exposing sensitive data.

Business impact of top 5 vulnerabilities:

1. Injection (A9): This vulnerability allows an attacker to inject malicious code into a web application, which can then be executed by the application. This can lead to a variety of impacts, including data theft, system compromise, and denial of service attacks.

- Financial losses: An attacker could inject malicious code into a web application that steals financial data, such as credit card numbers or bank account information. This could lead to financial losses for the organization and its customers.
- Reputational damage: If an attacker successfully compromises a web application, it could damage the organization's reputation. Customers may lose trust in the organization and its products or services.
- Regulatory fines: In some cases, a data breach or system compromise could result in regulatory fines. For example, the European Union's General Data Protection Regulation (GDPR) imposes fines of up to €20 million or 4% of global annual turnover, whichever is greater, for data breaches.

2. Broken Authentication and Session Management (A10): This vulnerability allows an attacker to gain unauthorized access to a web application by bypassing or exploiting weak authentication mechanisms. This could allow the attacker to steal sensitive data, modify data, or disrupt the application's functionality.

- Data theft: An attacker could gain unauthorized access to a web application and steal sensitive data, such as customerPII or financial information. This could lead to financial losses for the organization and its customers.
- Fraud: An attacker could gain unauthorized access to a web application and use the credentials to commit fraud, such as making unauthorized purchases or accessing restricted accounts.
- Denial of service: An attacker could gain unauthorized access to a web application and use the credentials to disrupt the application's functionality, such as by flooding the application with requests.

3. Insecure Deserialization (A8): This vulnerability occurs when untrusted data is deserialized without proper validation. This can allow an attacker to execute malicious code on the application's server.

- Data corruption: An attacker could inject malicious code into serialized data, which could then be deserialized by the application and executed on the server. This could lead to data corruption or the disclosure of sensitive data.
- System compromise: An attacker could use insecure deserialization to gain unauthorized access to the application's server and take control of the system.
- Denial of service: An attacker could use insecure deserialization to flood the application's server with requests, causing a denial of service attack.

4. Cross-Site Scripting (XSS) (A7): This vulnerability allows an attacker to inject malicious code into a web page, which is then executed by the victim's browser when the page is loaded. This can lead to a variety of impacts, including data theft, system compromise, and denial of service attacks.

- Data theft: An attacker could inject malicious code into a web page that steals sensitive data, such as cookies or session tokens. This could then be used to gain unauthorized access to the victim's account.
- Phishing: An attacker could inject malicious code into a web page that looks like a legitimate website. When the victim loads the page, the malicious code could steal the victim's credentials or redirect the victim to a phishing website.

- Denial of service: An attacker could inject malicious code into a web page that floods the victim's browser with requests, causing a denial of service attack.

5. Security Misconfiguration (A4): This vulnerability occurs when a web application is not properly configured. This can leave the application exposed to a variety of attacks, such as unauthorized access, data theft, and denial of service attacks.

- Data theft: A misconfigured web application could allow an attacker to gain unauthorized access to sensitive data, such as customer PII or financial information.
- System compromise: A misconfigured web application could allow an attacker to gain unauthorized access to the application's server and take control of the system.
- Denial of service: A misconfigured web application could be used to launch a denial of service attack against the application or its infrastructure.

CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection'-A9)

Description –

The product constructs all or part of an SQL command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended SQL command when it is sent to a downstream component.

Extended Description –

Without sufficient removal or quoting of SQL syntax in user-controllable inputs, the generated SQL query can cause those inputs to be interpreted as SQL instead of ordinary user data. This can be used to alter query logic to bypass security

checks, or to insert additional statements that modify the back-end database, possibly including execution of system commands.

SQL injection has become a common issue with database-driven web sites. The flaw is easily detected, and easily exploited, and as such, any site or product package with even a minimal user base is likely to be subject to an attempted attack of this kind. This flaw depends on the fact that SQL makes no real distinction between the control and data planes.

Common Consequences

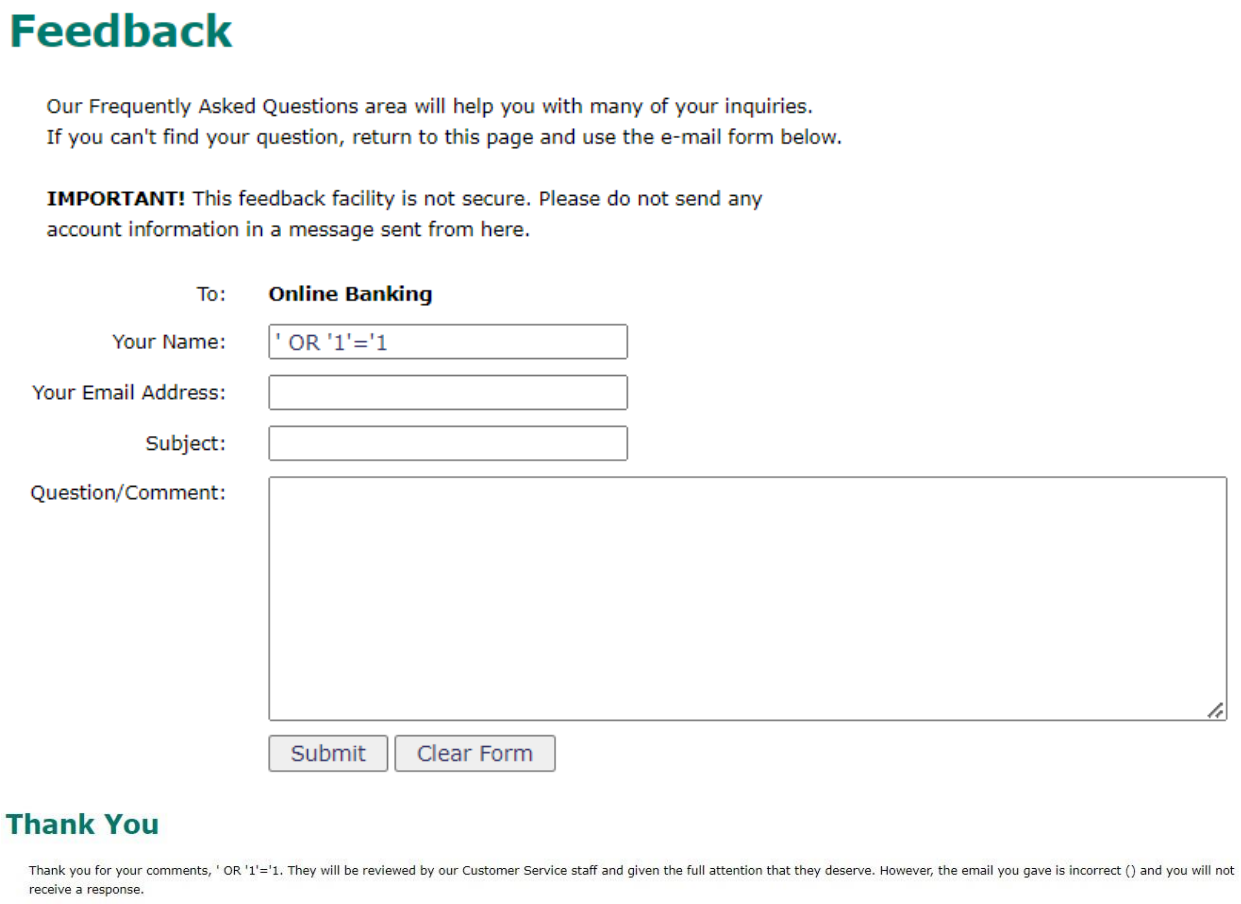
Scope	Impact	Likelihood
Confidentiality	**Technical Impact:** *Read Application Data* Since SQL databases generally hold sensitive data, loss of confidentiality is a frequent problem with SQL injection vulnerabilities.	
Access Control	**Technical Impact:** *Bypass Protection Mechanism* If poor SQL commands are used to check user names and passwords, it may be possible to connect to a system as another user with no previous knowledge of the password.	
Access Control	**Technical Impact:** *Bypass Protection Mechanism* If authorization information is held in a SQL database, it may be possible to change this information through the successful exploitation of a SQL injection vulnerability.	
Integrity	**Technical Impact:** *Modify Application Data* Just as it may be possible to read sensitive information, it is also possible to make changes or even delete this information with a SQL injection attack.	

Likelihood Of Exploit

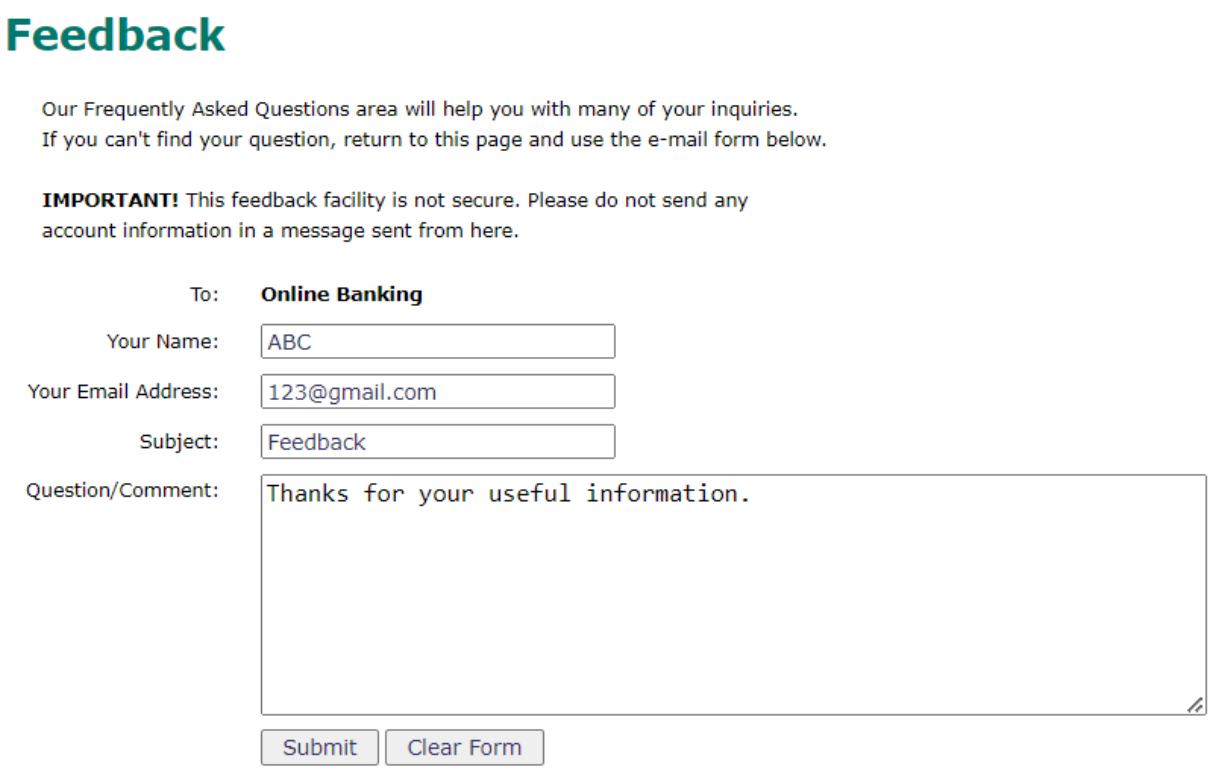
High

Implementation of top 5 OWASP vulnerabilities:

1. Injection (A9):



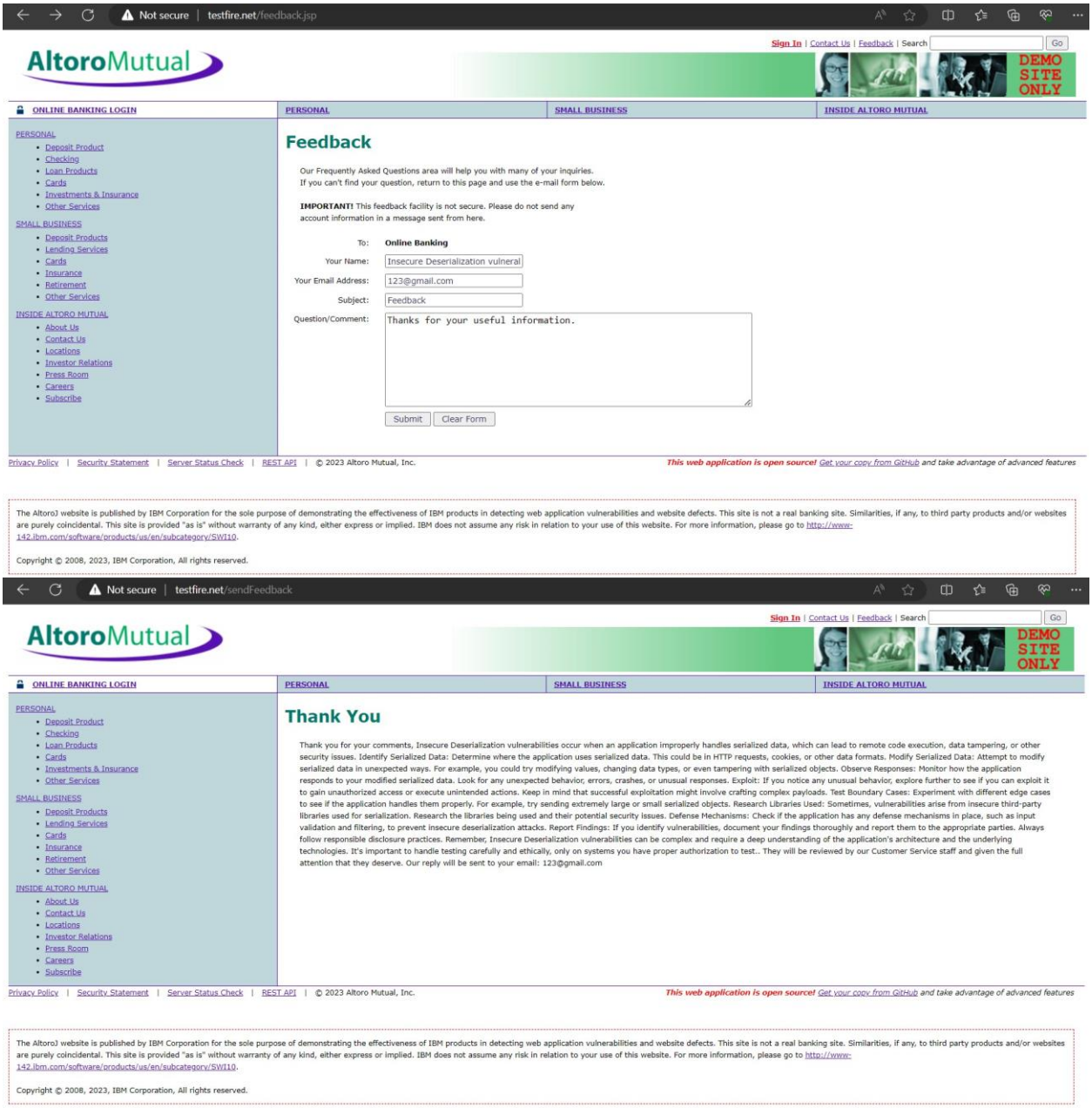
2. Broken Authentication and Session Management (A10):



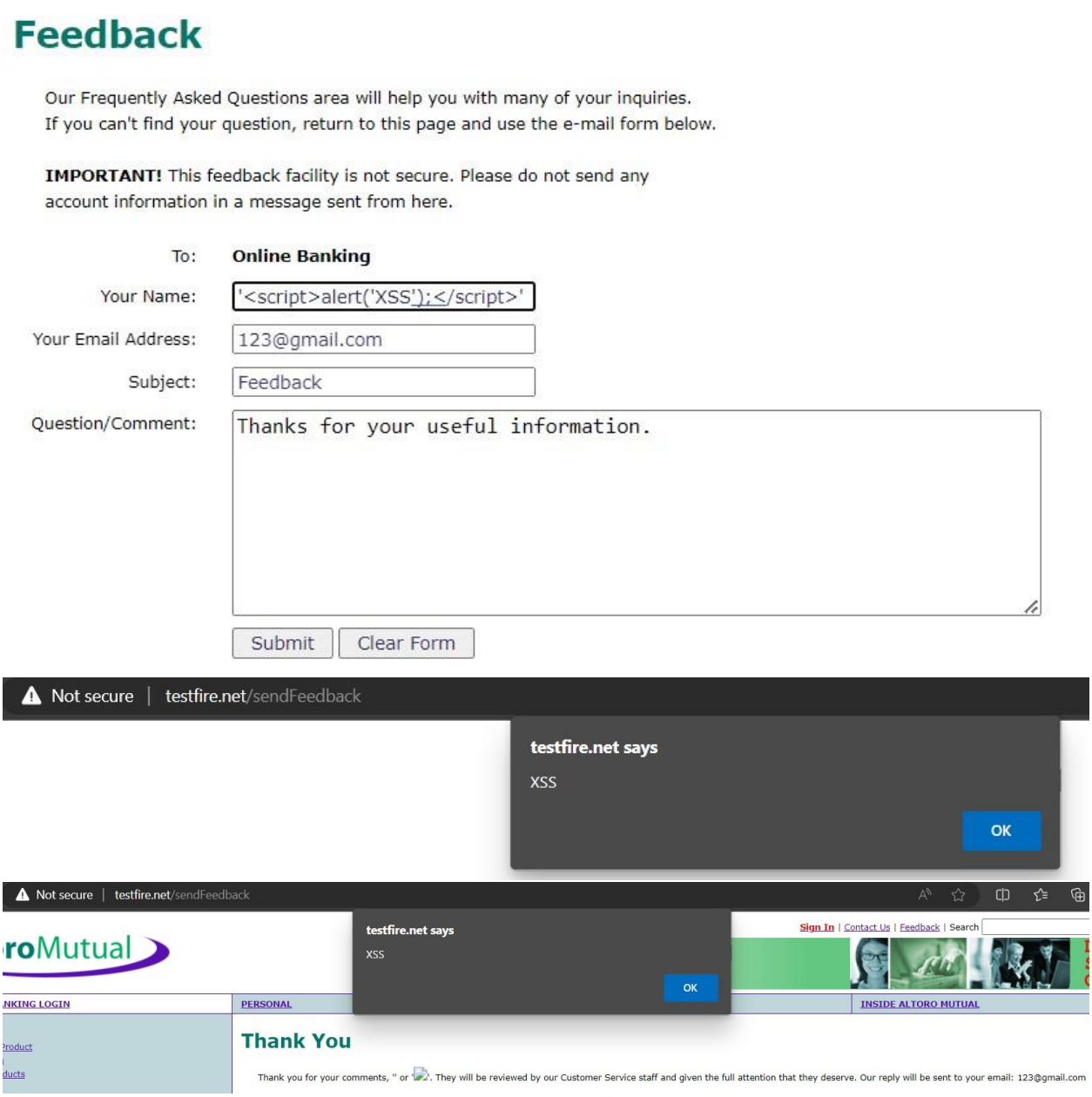
Test for Account Lockout: Determine if the system locks user accounts after a certain number of unsuccessful login attempts. Try to trigger this behavior.

3. Insecure Deserialization (A8):

   Test Boundary Cases: Experiment with different edge cases to see if the application handles them properly. For example, try sending extremely large or small serialized objects.



4. Cross-Site Scripting (XSS) (A7):
   Inject Payloads: Inject various payloads into these input fields. Common payloads include `<script>alert('XSS');</script>` or `<img src="x" onerror="alert('XSS')">`.



5. Security Misconfiguration (A4):
   Nil.