

Website name Web Application VAPT Level 1 Report



AltoroMutual - demo.testfire.net

Web Application VAPT Level 1 Report

For

ASD Academy

Prepared By

Vaibhav khare

TestingDate(07/11/2025)

Disclaimer this document or any of its content cannot account for, or be included in any form of legal advice. The outcome of a vulnerability scan (or security evaluation) should be utilized to ensure that diligent measures are taken to lower the risk of potential exploits carried out to compromise data. Legal advice must be supplied according to its legal context. All laws and the environments, in which they are applied, are constantly changed and revised. Therefore, no information provided in this document may ever be used as an alternative to a qualified legal body or representative. A portion of the information in this report is taken from OWASP's "The Ten Most Critical Web Application Security Vulnerabilities - 2023 Update" document, that can be found at <https://www.owasp.org>.

The results indicated in this report are reflective of the state of the site and its underlying code at the time of testing. **Testing Company** not be liable for any changes that happen after the submission of this report which might add to vulnerabilities in future.

1. Management Summary

1.1 Document Title

DOCUMENT VERSION CONTROL	
Document Title	Web Application Vulnerability Assessment and Penetration Testing Level 1 Report
Organisation Name	Altoro Mutual
Application Name	demo.testfire.net
Document Version	1.0
Last Edit Date	10/11/25

DOCUMENT DISTRIBUTION LIST	
Date	
Classification	Client Confidential
Submitted To	Mr.Harshit
Designation	CTO
Address	
Contact Number	
E-Mail	

1.2 Contact Details

Name	Mr. Vaibhav khare
Designation	Security Analyst
Company	ASD Academy
Address	
Tel. No.	NA
Mobile No.	
E – Mail	<hr/>

Table of Contents

1. ManagementSummary	1	2-3
1.1 DocumentTitle	2	
1.2 ContactDetails	2	3
2. Scope, Scan(Test)andReport(Creation/Review)Details	3.	6-7
AssessmentMethodology	5.	8-10
Tools Used	6.	
Standard Followed		
7. OWASP Top 10andSANS25ApplicationSecurityRisks	8.	11
Zero-day (0-day)ApplicationSecurityRisks	9.	11
EngagementScope	10.	
Details of theAuditingteam	11.	11-13
Server		
Details		14
		15
		15
		16
11.1 OS / WebServer/TechnologyDetails		16
12. Information Gathering		16-19
12.1 Identification ofSpiders,RobotsandCrawler		16
12.2 Search EngineDiscovery/Reconnaissance		17-18
12.3 Port Scanning		19
13. Summary of Key Findings		19-21
14. Graphical Representation		21
15. Detailed Technical ReportwithRecommendation		22-69
1. Authentication bypass-sqlinjection		22-24
1.2. Client side requestforquery		24-27
1.3. Insecure Direct objectreference(IDOR)		27-29

1.4. Default Credentials	29-31
1.5. Login Brute-force	31-34
1.6. TLS 1.0/1.1protocoldetection	34-36
1.7. Cross-sitescripting(xss)	37-38
1.8. No ratelimiting	39-41
1.9. Improperinputvalidation	42-44
1.10. Plaintextcredentials	44-46
1.11. Clickjacking	46-48
1.12. AccessiblebyIPaddress	49-50
1.13. HTMLInjection	50-52
1.14. UnvalidatedURLRedirection	53-55
1.15. 2FA Notimplemented	55-57
1.16. CAPTCHANotFound	58-60
1.17. MissingX-Frame-OptionHeader	61-62
1.18. HardcodedVersionDisclosure	63-65
1.19. MissingX-Content-Type-OptionsHeader	65-67
1.20. MissingPermission-PolicyHeader	67-69
16. Checklist (TestCasesPerformed)	69-76
17. General References 18.	77
Appendices	77

2. Scope, Scan (Test) and Report (Creation/Review) Details

Following Website is considered as scope of work. During the security assessment we have evaluated security of the modules from the application.

Scope of the work	
Application Name	empower web Portal
Scope	Website Audit
Audit URL	https://demo.testfire.net

Audit Activities and Timelines	
Start Date	07/11/2025
End Date	10/11/2025
Scan / Test Time	

Report Created / Reviewed / Approved / Released By	
Report Created By	Ms. Vaibhav khare (Information Security Analyst)
Report Reviewed By	Mr. Harshit (Sr. Cybersecurity Analyst)
Report Approved By	
Report Released By	
Report Released Date	

3. Assessment Methodology

A hybrid approach is followed to perform the assessment that is a combination of tools used to discover the wide range of vulnerabilities. Additionally, the assessment being adaptive in nature allows us to control the assessment methodology as per the application functionality to focus on the critical areas of the application. The attack vectors are controlled as per the assessment needs and the attack selection ensures maximum coverage of the application.

Following diagram represents the assessment approach:



Figure 1

The table below describes various levels and types of assessment. The type of assessment done for current assessment is available in the “Assessment Scope” section of the document.

Scan/Audit Type		
Level	Type	Information
1	Safe	Safe scan discovers minimum types and instances of vulnerabilities. The safe scan mode avoid fault injection such as Java Scripts, HTML tags, crafted SQL queries etc. to ensure that the application retains its state at the end of the assessment. Any fault injections that may trigger Denial of Service situation are avoided in safe scans. Safe scan suits most when the assessment is to be done on a live application instance, and has already undergone either <i>Standard</i> or <i>Destructive</i> scan/s.
2	Standard	Standard scan discovers and exploits most standard checks such as OWASP Top 10 checks. The standard scan performs fault injection such as Java Scripts injection, HTML tag injection, crafted SQL queries etc. Any fault injections that may trigger Denial of Service situation are avoided in standard scans. Standard scan suits most when the assessment is to be done on a staging/pre-prod/testing application instance.
3	Destructive	Destructive scan discovers and exploits most comprehensive checks including checks that may trigger Denial of Service Attacks situations for the application. Destructive scan is usually done on staging/pre-prod/testing application instance. A destructive scan on a live environment is avoided on live/production systems unless it is really required.

The vulnerabilities discovered are associated with a risk level that indicates how critical the vulnerability is and helps application owners/developers to prioritize the vulnerabilities and choose an appropriate mitigation approach.

Risk Level Information and Necessary Actions

Risk Level	Risk Description and Necessary Action
High	The high risk level indicates maximum risk associated with a specific vulnerability instance. Such vulnerability may enable an attacker to successfully exploit the underlying application and its data and partially or completely compromise the application and its data to modify application behaviour to become other than its original intended purpose. The vulnerability marked as "High Risk" is recommended to be handled with utmost priority.
Medium	The medium risk level indicates considerable risk associated with a specific vulnerability instance. Such vulnerability may enable an attacker to exploit the underlying application and its data to a particular level so that the attacker can gain low level information about the application. Such information can be used by an attacker to craft more specific attacks based on the information collected. The vulnerability marked with "Medium Risk" should be mitigated at the earliest or soon after "High Risk" vulnerabilities are mitigated.
Low	The low risk level indicates lowest risk associated with a specific vulnerability instance. Such vulnerability may allow an attacker to gain some information about the application which was not intended to be known otherwise. The attacker may not have exploiting techniques available at that instance based on the information revealed by the system. The vulnerability marked with "Low Risk" can be mitigated soon after high and medium risk vulnerabilities are mitigated.

Figure 2

Severity Level Information and Description

SEVERITY	CVSS SCORE	DESCRIPTION
CRITICAL	9.00 – 10.00	Critical Business Impact - e.g., Remote Code Execution, Database Access, System Take Over. Requires fix immediately.
HIGH	7.00 – 8.99	High Business Impact - e.g., Bypassing security controls, arbitrary script execution, takeover any user's account, bypass of user accounts on critical parts of the site. Requires fix as soon as possible

MEDIUM	4.00 – 6.99	Typically, vulnerabilities that requires the attacker to combine it with another vulnerability to cause serious damage
LOW	0.01 – 3.99	Can cause annoyance to the user and be used in a combination with similar vulnerabilities.
INFORMATIONAL	0.0	Bugs that do not create a threat directly or indirectly fall under this category.

Table 1

Assessment Constraints

We have covered the below modules during security audit:

- Data Management
- Admin Setting
- SupplierAssessment

- Survey

4. Tools Used

The below tools/scans/scripts were used for security audit:

- Burp Suite Professional (Commercial)
- Nmap (Open Source)
- Kali Linux Tools (Open Source)
- SSL Labs (Open Source)

5. Standard Followed

Below are the standards followed for VAPT –

1. Open Web Application Security Project (OWASP)
2. SysAdmin, Audit, Network, and Security (SANS)
3. Penetration Testing Execution Standard (PTES)
4. Payment Card Industry Data Security Standard (PCI DSS)
5. Information Systems Security Assessment Framework (ISSAF)
6. Open-Source Security Testing Methodology Manual (OSSTMM)

6. OWASP Top 10 and SANS 25 Application Security Risks

The Common Weakness Enumeration (CWE) is a list of software security vulnerabilities found all throughout the software development industry. It's a community-driven project maintained by MITRE, a non-profit research and development group. For each entry, the CWE provides a description of the vulnerability and steps for mitigating it.

MITRE partnered with the SANS Institute to develop the CWE/25, a list of the 25 most critical software vulnerabilities. A similar list is provided in the Open Web Application Security Project (OWASP) Top 10 Project, which is also a community-driven compilation of software vulnerabilities. Although the CWE/25 and OWASP Top 10 are different, they share many of the same vulnerabilities. Here is a list of the OWASP Top 10 entries for 2021 and their corresponding CWEs.

OWASP Top 10	SANS CWE Top 25
A01:2021-Broken Access Control	1. CWE-787 Out-of-bounds Write
A02:2021-Cryptographic Failures	1. CWE-79 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
A03:2021-Injection	1. CWE-89 Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
A04:2021-Insecure Design	1. CWE-416 Use After Free
A05:2021-Security Misconfiguration	1. CWE-78 Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
A06:2021-Vulnerable and Outdated Components	1. CWE-20 Improper Input Validation
A07:2021-Identification and Authentication Failures	1. CWE-125 Out-of-bounds Read
A08:2021-Software and Data Integrity Failures	1. CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') 1. CWE-352 Cross-Site Request Forgery (CSRF)
A09:2021-Security Logging and Monitoring Failures	1. CWE-434 Unrestricted Upload of File with Dangerous Type
A10:2021-Server-Side Request Forgery	1. CWE-862 Missing Authorization 1. CWE-476 NULL Pointer Dereference 1. CWE-287 Improper Authentication

	<ul style="list-style-type: none"> 1. CWE-190 Integer Overflow or Wraparound 1. CWE-502 Deserialization of Untrusted Data 1. CWE-77 Improper Neutralization of Special Elements used in a Command ('Command Injection') 1. CWE-119 Improper Restriction of Operations within the Bounds of a Memory Buffer 1. CWE-798 Use of Hard-coded Credentials 1. CWE-918 Server-Side Request Forgery (SSRF) 1. CWE-306 Missing Authentication for Critical Function 1. CWE-362 Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition') 1. CWE-269 Improper Privilege Management 1. CWE-94 Improper Control of Generation of Code ('Code Injection') 1. CWE-863 Incorrect Authorization 1. CWE-276 Incorrect Default Permissions
--	---

Table 2

7. Zero-day (0-day) Application Security Risks

During the security testing, checks were also performed for any zero-day vulnerabilities / attacks.

Below is the brief summary of Zero-day (0-day) security risks –

"Zero-day" is a broad term that describes recently discovered security vulnerabilities that hackers can use to attack systems. The term "zero-day" refers to the fact that the vendor or developer has only just learned of the flaw – which means they have "zero days" to fix it. A zero-day attack takes place when hackers exploit the flaw before developers have a chance to address it.

Zero-day is sometimes written as 0-day. The words vulnerability, exploit, and attack are typically used alongside zero-day, and it's helpful to understand the difference:

A zero-day vulnerability is a software vulnerability discovered by attackers before the vendor has become aware of it. Because the vendors are unaware, no patch exists for zero-day vulnerabilities, making attacks likely to succeed.

A zero-day exploit is the method hackers use to attack systems with a previously unidentified vulnerability.

A zero-day attack is the use of a zero-day exploit to cause damage to or steal data from a system affected by a vulnerability.

Protection against zero-day attacks –

For zero-day protection and to keep your computer and data safe, it's essential for both individuals and organizations to follow cyber security best practices. This includes:

Keep all software and operating systems up to date. This is because the vendors include security patches to cover newly identified vulnerabilities in new releases. Keeping up to date ensures you are more secure.

Use only essential applications. The more software you have, the more potential vulnerabilities you have. You can reduce the risk to your network by using only the applications you need.

Use a firewall. A firewall plays an essential role in protecting your system against zero-day threats. You can ensure maximum protection by configuring it to allow only necessary transactions.

Within organizations, educate users. Many zero-day attacks capitalize on human error. Teaching employees and users' good safety and security habits will help keep them safe online and protect organizations from zero-day exploits and other digital threats.

Use a comprehensive antivirus software solution.

8. Engagement Scope

S. No	URL	Hash Value (in case of applications)	Version (in case of applications)
1	https://demo.testfire.net	352c3725ddd88b56b5cc351dbbec353211abf5d8	Coyote JSP engine 1.1

9. Details of the Auditing team

S. No	Name	Designation	Email Id	Professional Qualifications/ Certifications	Whether the resource has been listed in the Snapshot information published on CERT-In's website (Yes/No)
1	Vaibhav khare	Security Analyst	<u> </u>	NO	NO

10. Server Details

a. OS/WebServer / Technology Details

OS Server	Windows
Web Server	Apache Tomcat/Coyote JSP engine 1.1
IP Address	65.61.137.117
Port	21,80,443,554,1723,8080
Technology Used	HTTP/2, Apache Tomcat, Java

11. Information Gathering

a. Identification of Spiders, Robots and Crawler

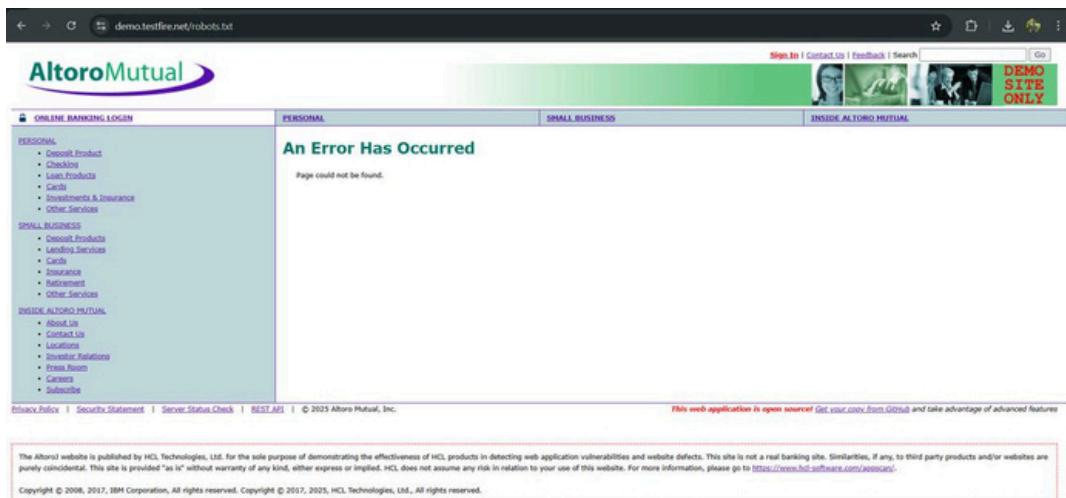
The first phase in security assessment was focused on collecting as much information as possible about a target application.

Tests Conducted:

The site contains robots.txt or not.

While searching: <https://demo.testfire.net/robots.txt>

Results: (Reference Image)



Severity: NIL

Search Engine Discovery/Reconnaissance

Tests Conducted:

Using the advanced "site:" search operator, it is possible to restrict Search Results to a specific domain.

Results: (Reference Image)

To find the web content of site indexed by Google Search, the following Google Search Query is issued:

site: <https://demo.testfire.net/login>

The screenshot shows the Altoro Mutual website at demo.testfire.net. The header features the Altoro Mutual logo and navigation links for Sign In, Contact Us, Feedback, and Search. A green banner at the top right says "DEMO SITE ONLY". The main content area is divided into several sections: "ONLINE BANKING LOGIN", "PERSONAL", "SMALL BUSINESS", and "INSIDE ALTORO MUTUAL". Each section contains text, images, and links. For example, the "PERSONAL" section has a sub-section for "Online Banking with FREE Online BILL Pay" featuring a couple holding a check. The "INSIDE ALTORO MUTUAL" section includes a "Privacy and Security" link with a note about protecting employee information. A footer at the bottom provides links to Privacy Policy, Security Statement, Server Status Check, REST API, and copyright information.

<https://demo.testfire.net/index.jsp>

The screenshot shows a Google search results page for the query "site:demo.testfire.net/login". The search bar at the top contains the query. Below it, there's a search interface with a microphone icon and a magnifying glass. The results list starts with a message: "We could not find any results for: site:demo.testfire.net/login. Try the suggestions below or type a new query above." Underneath this, there's a "Suggestions:" section with a bulleted list: "Check your spelling.", "Try more general words.", "Try different words that mean the same thing.", and "For more helpful tips on searching, visit the Yahoo Search Help Centre". The Yahoo logo is visible in the top right corner of the search interface.

To display the index of Web Application cached by Google, the following Google Search Query is issued

cache: <https://xyz.com/>

(Reference Image)

Severity: NIL

a. Port Scanning

Tests Conducted:

Using the Nmap we checked for Open Ports for the following website.

Results:

site: <https://demo.testfire.net/>

(Reference Image)

```

Zenmap
Scan Tools Profile Help
Target: demo.testfire.net Profile: Quick scan Scan Cancel
Command: nmap -T4 -F demo.testfire.net

Hosts Services
OS Host
demo.testfire.net

Nmap Output Ports / Hosts Topology Host Details Scans
nmap -T4 -F demo.testfire.net Details
Starting Nmap 7.98 ( https://nmap.org ) at 2025-11-04 11:27 +0530
Nmap scan report for demo.testfire.net (65.61.137.117)
Host is up (0.30s latency).
Not shown: 94 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
80/tcp    open  http
443/tcp   open  https
554/tcp   open  rtsp
1723/tcp  open  pptp
8080/tcp  open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 15.23 seconds

```

Severity: NIL

1. Summary of Key Findings

Sr.no	Finding	Severity	Level 1 Status
1.	Authentication bypass – Sql injection	Critical	Open

2.	Client Side Request Forgery (CSRF)	High	Open
3	Insecure Direct Object Reference (IDOR)	High	Open
4	Default Credentials	High	Open
5	Login Brute-Force	High	Open
6	TLS 1.0/1.1 Protocol Detection	High	Open
7	Cross-Site Scripting(XSS)	High	Open
8	No Rate Limiting	High	Open
9	Improper Input Validation	High	Open
10	Plaintext Credentials	High	Open
11	Click-Jacking	Medium	Open
12	Accessible By IP Address	Medium	Open
13	HTML Injection	Medium	Open
14	Unvalidated URL Redirection	Medium	Open
15	2FA Not Implemented	Medium	Open
16	CAPTCH Not Found	Medium	Open
17	Missing X-Frame-Option Header	Medium	Open
18	Hardcoded Version Disclosure	Medium	Open
19	Missing X-Content-Type-Option Header	Low	Open

20	Missing Permissions-Policy Header	Low	Open
----	-----------------------------------	-----	------

1. Graphical Representation

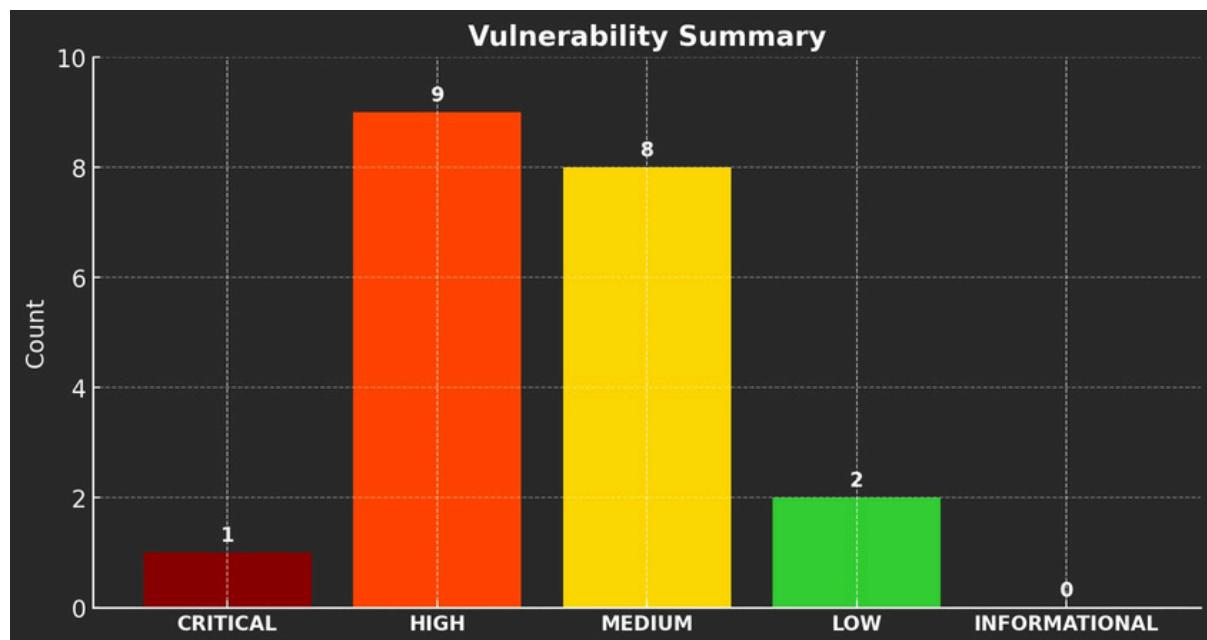


Figure 3

1. Detailed Technical Report with Recommendation

1.1. Authentication Bypass- SQL injection(Login)

Authentication Bypass - SQL Injection (Login)	
SEVERITY	Critical
STATUS	OPEN
CVSS SCORE	9.1
CWE-ID	CWE-89
OWASP CATEGORY	Injection
DESCRIPTION	vulnerability that occurs when an attacker is able to manipulate a web application's SQL queries by injecting malicious input. This can allow the attacker to bypass authentication mechanisms, such as login pages, and . such as login pages, and potentially gain access to sensitive data, including admin credentials.
IMPACT	SQLi allows the attacker to bypass login page and take admin credentials
AFFECTED URL	https://demo.testfire.net/login.jsp
REFERENCE URL FOR MITIGATION	

CWE-89: Improper Neutralization of Special Elements used in an SQL Command.
cwe.mitre.org OWASP Web Security Testing Guide – Testing for Bypassing Authentication. OWASP Foundation PortsWigger / Web Security Academy (conceptual guidance for testing and prevention). [portswigger.net+1](http://portswigger.net)

RECOMMENDATIONS

Use parameterized queries/prepared statements everywhere — never concatenate user input into SQL.

Store passwords with strong hashing (bcrypt/argon2) and verify server-side, not in raw SQL. Limit database privileges for the app account and validate all input with strict server-side rules.

STEPS TO REPRODUCE / PROOF OF CONCEPT

First Step I will Go to <http://testfire.net/login.jsp>

Enter username as ' or 1=1 -- and password random some input and then click on login and we login as administrator

The screenshot shows a web-based banking interface for Altoro Mutual. At the top, there's a navigation bar with links for 'Sign Off', 'Contact Us', 'Feedback', 'Search', and a 'Go' button. Below the navigation is a banner with three small profile pictures and the text 'DEMO SITE ONLY'. The main content area has tabs for 'PERSONAL', 'SMALL BUSINESS', and 'INSIDE ALTORO MUTUAL'. On the left, a sidebar titled 'MY ACCOUNT' lists options like 'View Account Summary', 'View Recent Transactions', 'Transfer Funds', 'Search News Articles', and 'Customize Site Language'. Another sidebar titled 'ADMINISTRATION' includes 'Edit Users'. The central content area displays a message: 'Hello Admin User' followed by 'Welcome to Altoro Mutual Online.' A dropdown menu shows '800000 Corporate' with a 'GO' button. Below this, a 'Congratulations!' message states: 'You have been pre-approved for an Altoro Gold Visa with a credit limit of \$100000! Click [Here](#) to apply.' At the bottom, there's a note about the application being open source, copyright information (© 2008, 2017, IBM Corporation), and a link to HCL's AppScan product.

1.2 Client Side Request Forgery (CSRF)

Client Side Request Forgery (CSRF)	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	8.8
CWE-ID	CWE-352
OWASP CATEGORY	Broken Access Control
DESCRIPTION	<p>A CSRF vulnerability exists when state-changing requests (POST/PUT/DELETE or sensitive GETs) are accepted and executed by the application based only on the victim's browser credentials (cookies, basic auth), without a reliable proof of user intent or origin. An attacker can induce an authenticated user's browser to send such a forged request, causing the application to perform unintended actions.</p>

IMPACT
Attackers trick users into submitting unintended requests using their active session. Can lead to account takeover, data modification, or financial fraud. Exploits browser cookie handling mechanisms.
AFFECTED URL
http://altoro.testfire.net/bank/transfer.jsp
REFERENCE URL FOR MITIGATION
<p>OWASP CSRF Prevention Cheat Sheet (detailed mitigations). cheatsheetseries.owasp.org</p> <p>OWASP Web Security Testing Guide — Testing for CSRF. OWASP Foundation</p> <p>CWE-352 (definition & context). cwe.mitre.org</p> <p>PortSwigger / Web Security (concepts, examples, SameSite discussion). portswigger.net+1</p> <p>OWASP SameSite cookie guidance. OWASP Foundation</p>
RECOMMENDATIONS
To prevent CSRF we can use CSRF tokens and use strict samesite cookie restrictions
STEPS TO REPRODUCE / PROOF OF CONCEPT
First I Select a request of transfer money in Burp Suite Professional that i want to test or exploit.

The screenshot shows the Burp Suite Professional interface with the following details:

Request

```
POST /bank/doTransfer HTTP/1.1
Host: testfire.net
Cookie: JSESSIONID=94AC19E0A3188E414F6051549B1fDCD; AltoroAccounts="ODAMDAHxXvnbwvcmFO2x4KCC440dg4Cbg4Cdg4OgjCDChRzHxDAMDAhXhZMNeW5Nf";JxMTH5MDuHND9
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US, en;q=0.9
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.6992.90 Safari/537.36
Content-Length: 14
Content-Type: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,*/*;q=0.8, application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-store
Sec-Fetch-Dest: document
Referrer: https://testfire.net/bank/transfer.jsp
Accept-Encoding: gzip, deflate, br
Priority: u0, i
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 14
fromAccount=800000toAccount=800001&transferAmount=2000&transferType=Money
```

Response

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/html;charset=UTF-8
Content-Length: 7445
Date: Sat, 01 Nov 2025 14:04:20 GMT
...
<!-- BEGIN HEADER -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml/DTD/xhtml-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" >
    <head>
        <title> Altoro Mutuals
        </title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <link href="style.css" rel="stylesheet" type="text/css" />
    </head>
    <body style="margin-top:5px">
        <div id="header" style="margin-bottom:5px; width: 99%;>
            <form id="findSearch" method="get" action="search.jsp">
                <table width="100%" border="0" cellpadding="0" cellspacing="0">
                    <tr>
                        <td rowspan="2" style="width: 5%;>
                            <a href="#">index.jsp</a>
                            
                        </td>
```

Bottom Status Bar

Event log (4) All issues (307) Memory: 271.3MB

From the right-click context menu, select Engagement tools / Generate CSRF PoC. Burp Suite will generate some HTML that will trigger the selected request , I will choose the option test in browser and copy the link and paste in browser and enter the transfer 2000\$ will done automatically

Transaction ID	Transaction Time	Account ID	Action	Amount
9034	2025-11-01 09:04	800001	Deposit	\$2000.00
9033	2025-11-01 09:04	800000	Withdrawal	-\$2000.00
9032	2025-11-01 08:59	800001	Deposit	\$2000.00
9031	2025-11-01 08:59	800000	Withdrawal	-\$2000.00
9030	2025-11-01 08:58	800001	Deposit	\$2000.00
9029	2025-11-01 08:58	800000	Withdrawal	-\$2000.00
9028	2025-11-01 08:58	800000	Deposit	\$2000.00
9027	2025-11-01 08:58	800000	Withdrawal	-\$2000.00
9026	2025-11-01 08:56	800001	Deposit	\$2000.00
9025	2025-11-01 08:56	800000	Withdrawal	-\$2000.00
9024	2025-11-01 08:53	800001	Deposit	\$2000.00
9023	2025-11-01 08:53	800000	Withdrawal	-\$2000.00
8492	2025-11-01 06:08	800001	Deposit	\$1000000.00
8491	2025-11-01 06:08	800000	Withdrawal	-\$1000000.00
8490	2025-11-01 06:04	800001	Deposit	\$500.00
8489	2025-11-01 06:04	800000	Withdrawal	-\$500.00
8488	2025-11-01 06:03	800000	Deposit	\$500.00
8487	2025-11-01 06:03	800000	Withdrawal	-\$500.00
8485	2025-11-01 06:02	800000	Withdrawal	-\$20500.00
8484	2025-11-01 06:01	800001	Deposit	\$500.00
8483	2025-11-01 06:01	800000	Withdrawal	-\$500.00

1.2 Insecure Direct Object Reference (IDOR)

Insecure Direct object Reference (IDOR)	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	8.7
CWE-ID	CWE-639
OWASP CATEGORY	Broken Access Control
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 different user.
IMPACT	

Manipulating IDs (e.g., /user?id=123→124) accesses unauthorized data. Common in APIs lacking proper authorization checks. Leads to mass data leakage.

AFFECTED URL

<https://demo.testfire.net/bank/showAccount?listAccounts=800000>

REFERENCE URL FOR MITIGATION

CWE-639 – definition and context (MITRE/CWE). cwe.mitre.org

OWASP Web Security Testing Guide – Testing for IDOR. OWASP

RECOMMENDATIONS

Implement access control checks for all object references. Use indirect reference maps. Log and monitor ID access patterns. Implement access control checks on the server side; avoid using predictable IDs; use indirect references or UUIDs

STEPS TO REPRODUCE / PROOF OF CONCEPT

Go to the Affected URL..

Date	Description	Amount
2025-11-01	Deposit	\$1000.00
2025-11-01	Deposit	\$1234.00
2025-11-01	Withdrawal	-\$1234.00
2025-11-01	Deposit	\$1234.00
2025-11-01	Withdrawal	-\$1234.00
2025-11-01	Deposit	\$1234.00

Account	Date	Description	Amount
1001160140	12/29/2004	Paycheck	1200
1001160140	01/12/2005	Paycheck	1200
1001160140	01/29/2005	Paycheck	1200
1001160140	02/12/2005	Paycheck	1200
1001160140	03/01/2005	Paycheck	1200
1001160140	03/15/2005	Paycheck	1200
1001160140	03/29/2005	Paycheck	1200
1001160140	04/12/2005	Paycheck	1200
1001160140	04/29/2005	Paycheck	1200
1001160140	05/12/2005	Paycheck	1200
1001160140	05/29/2005	Paycheck	1200

Then change the listAccounts id 800003 to 800004

The screenshot shows the AltoroMutual bank account history page for account 800004. The page has a navigation bar with links for Sign Off, Contact Us, Feedback, and Search. It also features a 'DEMO SITE ONLY' banner with three small images.

Account History - 800004

Balance Detail

800002 Savings	Select Account	Amount
Ending balance as of 11/1/25 11:56 PM		\$200969022165.00
Available balance		\$200969022165.00

10 Most Recent Transactions

Date	Description	Amount
2025-11-01	Deposit	\$70000000.00
2025-11-01	Deposit	\$100000000.00
2025-11-01	Deposit	\$20500.00
2025-11-01	Deposit	\$200000000000.00
2025-11-01	Deposit	\$1000.00
2025-11-01	Deposit	\$1000.00
....

Credits

Account	Date	Description	Amount
1001160140	12/29/2004	Paycheck	1200
1001160140	01/12/2005	Paycheck	1200
1001160140	01/29/2005	Paycheck	1200
1001160140	02/12/2005	Paycheck	1200
1001160140	03/01/2005	Paycheck	1200
1001160140	03/15/2005	Paycheck	1200
....

1.4 Default Credentials

Default Credentials	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	9
CWE-ID	CWE-1391
OWASP CATEGORY	Authentication Failures
DESCRIPTION	

IMPACT
Factory-set credentials allow immediate system access. Often found in IoT devices and admin interfaces. Brute-force attacks succeed quickly against known defaults.
AFFECTED URL
https://demo.testfire.net/login.jsp
REFERENCE URL FOR MITIGATION
https://owasp.org/www-project-secure-coding-practices/ https://cheatsheetseries.owasp.org/cheatsheets/Default_Password_Cheat_Sheet.html
RECOMMENDATIONS
Change all default credentials during setup. Implement password policies. Disable default admin accounts if possible. Regular credential audits. Enforce password changes after installation, disable default accounts, monitor for common credentials
STEPS TO REPRODUCE / PROOF OF CONCEPT
Go to the Affected URL and enter username or password (admin/admin)

1.5 Login Brute-Force

Login Brute-Force	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	8.2
CWE-ID	CWE-307
OWASP CATEGORY	Broken Authentication
DESCRIPTION	Attackers can perform automated guessing (brute-force), password-spraying, or credential-stuffing attacks to gain account access. This includes abuse via APIs and

automated tools that may try large lists of username/password pairs or bypass naive rate limits.

IMPACT

Account takeover (including administrative accounts) leading to data theft, fraud, or unauthorized actions.

AFFECTED URL

<https://demo.testfire.net/login.jsp>

REFERENCE URL FOR MITIGATION

CWE-307 – Improper Restriction of Excessive Authentication Attempts.
cwe.mitre.org

OWASP Authentication Cheat Sheet (rate-limiting, lockout, MFA, secure recovery).
cheatsheetseries.owasp.org

RECOMMENDATIONS

Rate limit + per-account and per-IP throttling, Require MFA for privileged and high-risk accounts.,

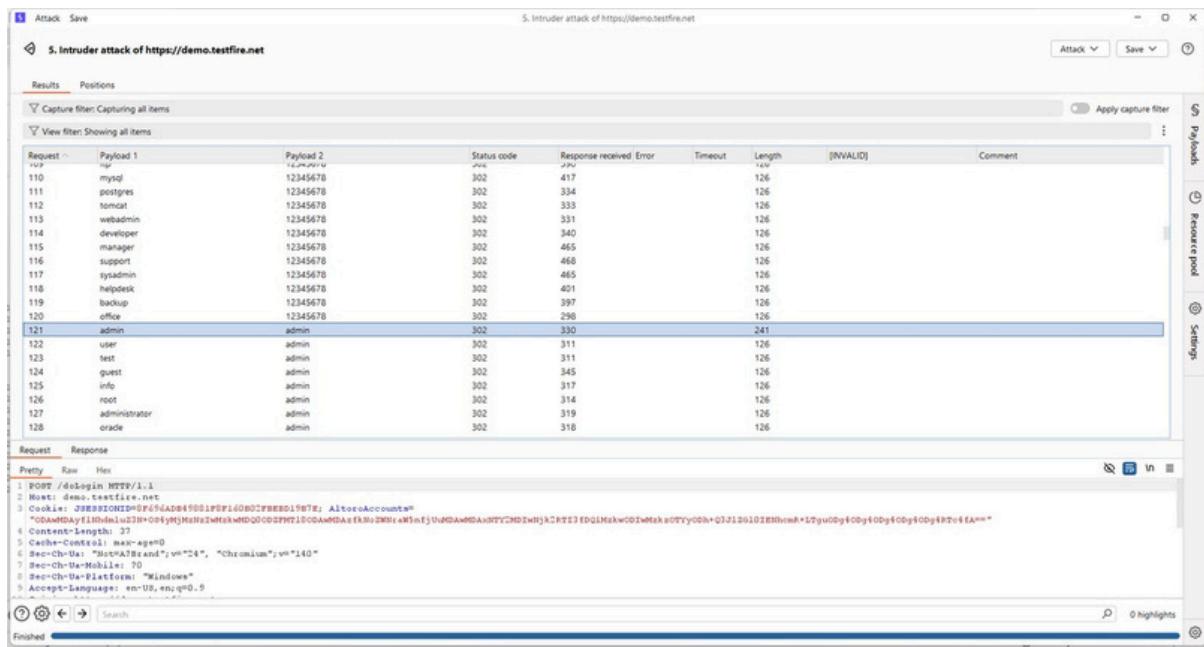
STEPS TO REPRODUCE / PROOF OF CONCEPT

Go to the affected url..

Using Burp Intruder with *Pitchfork* and *Cluster Bomb* modes, username and password lists were loaded to automate login attempts on the target (<https://demo.testfire.net/login.jsp>).

Multiple combinations of usernames and passwords were sent in parallel HTTP POST requests to identify valid login credentials by comparing response length, status code, and redirect behavior.

The response for the pair admin:admin had a different status/length, indicating successful login — confirming that the application is vulnerable to brute-force attacks due to missing rate-limiting or account lockout mechanisms.



1.6 TLS 1.0/1.1 Protocol Detection

TLS 1.0/1.1 Protocol Detection	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	7.5
CWE-ID	CWE-327
OWASP CATEGORY	Cryptographic Failures
DESCRIPTION	<p>TLS 1.0 and/or TLS 1.1. These versions are known to have cryptographic weaknesses and lack modern protections found in TLS 1.2 and TLS 1.3. Supporting older versions allows attackers to attempt downgrade attacks or exploit weak cipher suites, putting the confidentiality and integrity of data at risk.</p>

IMPACT
Vulnerable to chosen-plaintext attacks (BEAST). No longer meets PCI compliance standards. Provides inadequate encryption for modern security needs.
Lacks modern cipher suites and vulnerable to downgrade attacks. Considered obsolete by most security standards.
AFFECTED URL
https://demo.testfire.net
REFERENCE URL FOR MITIGATION
SSL Labs (server testing) – use to validate TLS versions & cipher suites. https://www.ssllabs.com/ssltest/
RECOMMENDATIONS
Disable TLS 1.0 in server configurations. Enforce TLS 1.2+ with AEAD ciphers. Update all client libraries. Disable TLS 1.0; enforce TLS 1.2 or higher; update libraries and configurations Disable TLS 1.1 in all server configurations. Enforce TLS 1.2+ with strong ciphers. Monitor for protocol negotiation attempts.
STEPS TO REPRODUCE / PROOF OF CONCEPT
Open kali terminal and go to ./testssl.sh demo.testfire.net

It give TLS 1.0/1.1 offered (deprecated)

```

└─(kali㉿kali)-[~/tools/testssl.sh]
└$ ./testssl.sh demo.testfire.net

#####
# testssl.sh version 3.3dev from https://testssl.sh/dev/
# (f219fd6 2025-11-03 23:39:04)

This program is free software. Distribution and modification under
GPLv2 permitted. USAGE w/o ANY WARRANTY. USE IT AT YOUR OWN RISK!

Please file bugs @ https://testssl.sh/bugs/
#####

Using OpenSSL 1.0.2-bad (Mar 28 2025) [~183 ciphers]
on kali:./bin/openssl.Linux.x86_64

Testing all IPv4 addresses (port 443): 65.61.137.117
-----
Start 2025-11-04 05:59:46      —> 65.61.137.117:443 (demo.testfire.net) <—
rDNS (65.61.137.117):  --
Service detected:          HTTP

Testing protocols via sockets except NPN+ALPN

SSLv2      not offered (OK)
SSLv3      not offered (OK)
TLS 1       offered (deprecated)
TLS 1.1     offered (deprecated)
TLS 1.2     offered (OK)
TLS 1.3     not offered and downgraded to a weaker protocol
QUIC       not offered or timed out
NPN/SPDY   not offered
ALPN/HTTP2 not offered

```

1.7 Cross-Site Scripting (XSS)

Cross-Site Scripting (xss)	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	8
CWE-ID	CWE-79
OWASP CATEGORY	Cross-Site Scripting (XSS)
DESCRIPTION	Reflected XSS occurs when untrusted input from an HTTP request (query string, headers, form fields) is immediately included in an HTTP response without proper validation or context-aware encoding, allowing an attacker to craft a link or URL that, when visited by a victim, executes attacker-controlled script in the victim's browser.
IMPACT	Attackers execute arbitrary JavaScript in victim's browser session. Can steal cookies, modify page content, or redirect to phishing sites. Persistent XSS stored in databases affects multiple users.
AFFECTED URL	https://demo.testfire.net/search.jsp?query=
REFERENCE URL FOR MITIGATION	OWASP Web Security Testing Guide — XSS testing — testing approaches. https://owasp.org/www-project-web-security-testing-guide/ (search "XSS")

CWE-79 (MITRE) — definition and examples.

<https://cwe.mitre.org/data/definitions/79.html>

RECOMMENDATIONS

Validate and properly encode all user-supplied input before rendering it in web pages. Implement context-aware output encoding, use frameworks with built-in XSS protection, and enforce a strong Content Security Policy (CSP) to prevent script execution.

STEPS TO REPRODUCE / PROOF OF CONCEPT

Step 1 : Go to the affected url

Step 2 : Type the payload on seach bar <script>alert(1)</script>

The screenshot shows a web browser window for 'Altoro Mutual' at the URL <https://testfire.net>. The page displays various banking and business services. A search bar at the top contains the malicious payload. The page includes sections for 'PERSONAL' and 'SMALL BUSINESS' banking, as well as 'INSIDE ALTORO MUTUAL' features like 'Real Estate Financing', 'Business Credit Cards', and 'Retirement Solutions'. A banner at the bottom right reads 'DEMO SITE ONLY'.

1.8 No Rate Limiting

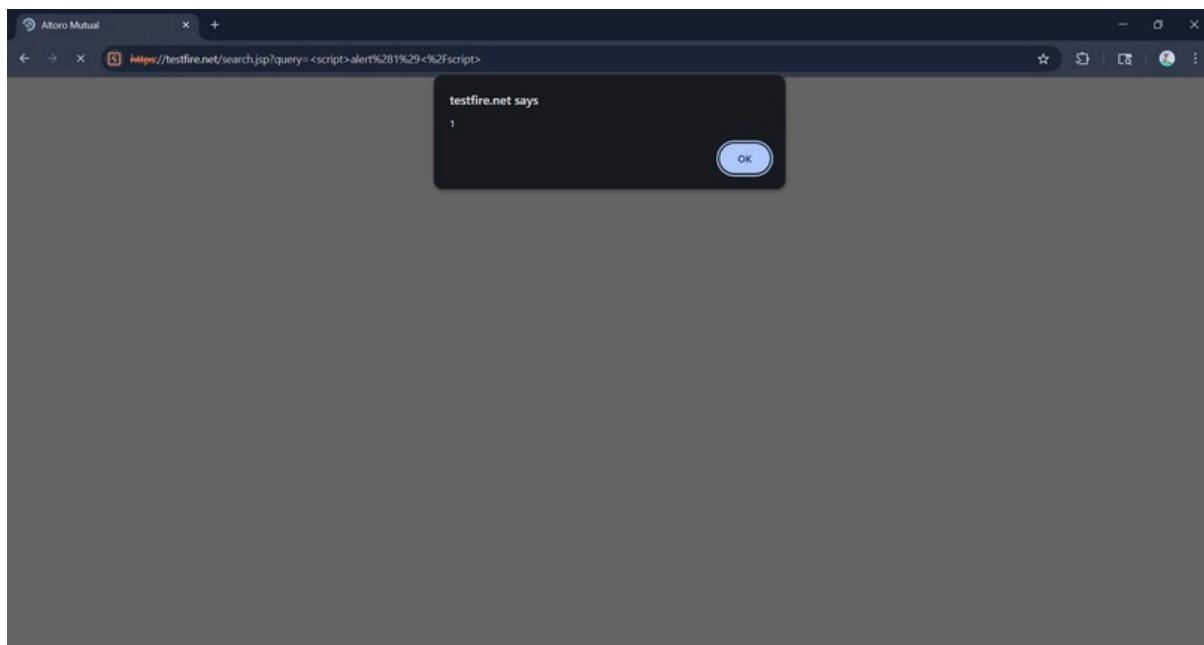
No Rate Limiting

SEVERITY	High
STATUS	OPEN
CVSS SCORE	8.8
CWE-ID	CWE-770
OWASP CATEGORY	Identification and Authentication Failures
DESCRIPTION	<p>The application does not throttle or limit repeated requests (login attempts, password resets, API calls, etc.) from the same user/IP or client. This enables attackers to run automated attacks (brute force, credential stuffing, enumeration, amplification) without meaningful slowdown or blocking.</p>
IMPACT	Unlimited requests overwhelm systems or bypass security controls. API endpoints particularly vulnerable.
AFFECTED URL	https://demo.testfire.net/login.jsp
REFERENCE URL FOR MITIGATION	<p>OWASP Rate Limiting Recommendations (OWASP Top 10: A6 Security Misconfiguration) https://owasp.org/Top10/A05_2021-Security_Misconfiguration/</p> <p>OWASP Cheat Sheet: Blocking Brute Force Attacks Cheat Sheet https://cheatsheetseries.owasp.org/cheatsheets/Blocking_Brute_Force_Attacks_Cheat_Sheet.html</p>
RECOMMENDATIONS	

Implement sliding window rate limiting (e.g., 100 requests/minute). Use Redis for distributed systems. Implement rate limiting, IP-based throttling, CAPTCHA, and account lockout after threshold of failed attempts

STEPS TO REPRODUCE / PROOF OF CONCEPT

Step 1 : Enter random username & password and capture their request /dologin in burpsuite then send this request to intruder and set target or select cluster bomb attack then set payload runtime or add wordlist or username and password then start the attack they will try 72request count..



Attack Save

3. Intruder attack of https://demo.testfire.net

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
361	admin	passw0rd	302	330		209		
362	user	passw0rd	302	347		209		
363	test	passw0rd	302	347		209		
364	guest	passw0rd	302	347		209		
365	info	passw0rd	302	361		209		
366	root	passw0rd	302	361		209		
367	administrator	passw0rd	302	373		209		
368	oracle	passw0rd	302	422		209		
369	ftp	passw0rd	302	440		209		
370	mysql	passw0rd	302	430		209		
371	postgres	passw0rd	302	430		209		
372	tomcat	passw0rd	302	430		209		
373	webadmin	passw0rd	302	430		209		
374	developer	passw0rd	302	430		209		
375	manager	passw0rd	302	370		209		
376	support	passw0rd	302	362		209		
377	sysadmin	passw0rd	302	355		209		
378	helpdesk	passw0rd	302	322		209		
379	backup	passw0rd	302	379		209		
380	office	passw0rd	302	378		209		

Event log: All issues (78)

Memory: 227.7MB

Step 2 : Their is no rate limiting that's way we request 400 times.

Attack Save

3. Intruder attack of https://demo.testfire.net

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
361	admin	passw0rd	302	330		209		
362	user	passw0rd	302	347		209		
363	test	passw0rd	302	347		209		
364	guest	passw0rd	302	347		209		
365	info	passw0rd	302	361		209		
366	root	passw0rd	302	361		209		
367	administrator	passw0rd	302	373		209		
368	oracle	passw0rd	302	422		209		
369	ftp	passw0rd	302	440		209		
370	mysql	passw0rd	302	430		209		
371	postgres	passw0rd	302	430		209		
372	tomcat	passw0rd	302	430		209		
373	webadmin	passw0rd	302	430		209		
374	developer	passw0rd	302	430		209		
375	manager	passw0rd	302	370		209		
376	support	passw0rd	302	362		209		
377	sysadmin	passw0rd	302	355		209		
378	helpdesk	passw0rd	302	322		209		
379	backup	passw0rd	302	379		209		
380	office	passw0rd	302	378		209		

Finished

1.9 Improper Input Validation

Improper Input Validation	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	8
CWE-ID	CWE-20
OWASP CATEGORY	injection
DESCRIPTION	<p>The application does not properly validate user-supplied input before processing it. This can allow attackers to inject unexpected data or exploit business logic flaws, leading to potential SQL Injection, XSS, or command execution vulnerabilities.</p>
IMPACT	Unvalidated inputs enable injection attacks (SQLi, XSS, RCE). Failure to sanitize allows attackers to bypass client-side checks.
AFFECTED URL	https://demo.testfire.net/bank/transfer.jsp
REFERENCE URL FOR MITIGATION	<p>OWASP Input Validation Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html</p> <p>CWE-20: Improper Input Validation: https://cwe.mitre.org/data/definitions/20.html</p>
RECOMMENDATIONS	

Whitelist input patterns. Use parameterized queries. Implement library-based sanitization (e.g., OWASP ESAPI). Test with fuzz vectors. Apply strict input validation on server-side, use whitelisting, sanitize and encode input, avoid relying solely on client-side validation

STEPS TO REPRODUCE / PROOF OF CONCEPT

Image 1: Setting Up the Attack

- Attacker enters a legitimate small transfer of **10,000** from account 800002 to 800003
- Burp Suite intercepts the HTTP request
- At the bottom (line 24), you can see the request parameters ready to be manipulated

The screenshot shows the Burp Suite interface with a captured request and its corresponding response.

Request:

```

POST /bank/doTransfer HTTP/1.1
Host: demo.testfire.net
Content-Type: application/x-www-form-urlencoded
From: 800002 Savings
To: 800003 Savings
TransferAmount: 10000

```

Response:

The response shows a successful transfer message: "10000.0 was successfully transferred from Account 800002 into Account 800003 at 11/5/25 5:08 AM."

Image 2: Manipulating the Request

- In Burp Suite's Proxy tab, the attacker modifies the intercepted request
- Changes transferAmount=10000 to transferAmount=**10000000000** (10 billion!)
- The request shows: fromAccount=800002&toAccount=800003&transferAmount=10000000000
- Attacker then forwards this modified request to the server

The screenshot shows a web browser window for 'https://demo.testfire.net/bank/doTransfer'. The page has a header with the AltoroMutual logo, navigation links like 'Sign Off', 'Contact Us', 'Feedback', and 'Search', and a 'DEMO SITE ONLY' button. The main content area is titled 'Transfer Funds' and contains fields for 'From Account' (800002 Savings), 'To Account' (800002 Savings), and 'Amount to Transfer' (\$10). A success message at the bottom states: '1.0E11 was successfully transferred from Account 800002 into Account 800003 at 11/5/25 5:11 AM.' Below the message, a note says 'This web application is open source! Get your copy from GitHub and take advantage of advanced features.' At the bottom, there's a footer with links to 'Privacy Policy', 'Security Statement', 'Server Status Check', 'REST API', and copyright information: 'Copyright © 2008, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd. All rights reserved.'

Successful Exploitation

- The server accepts the manipulated request without proper validation
- Success message appears: "1.0E11 was successfully transferred from Account 800002 into Account 800003"
- $1.0E11 = 100,000,000,000$ (100 billion in scientific notation)
- The attacker successfully transferred a massive amount despite only entering \$10 initially

1.10 Plaintext Credentials

Plaintext Credentials	
SEVERITY	High
STATUS	OPEN
CVSS SCORE	9.1
CWE-ID	CWE-319
OWASP CATEGORY	Cryptographic Failures

DESCRIPTION
The application transmits user credentials in plaintext without encryption. If an attacker intercepts the communication or gains access to the storage, they can directly read usernames and passwords.
IMPACT
Credentials transmitted without encryption can be intercepted via MITM attacks. Network sniffing reveals passwords in transit. Violates PCI DSS and other compliance standards.
AFFECTED URL
https://demo.testfire.net/login.jsp
REFERENCE URL FOR MITIGATION
<p>OWASP Password Storage Cheat Sheet https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html</p> <p>CWE-312 – Cleartext Storage of Sensitive Information https://cwe.mitre.org/data/definitions/312.html</p>
RECOMMENDATIONS
Enforce HTTPS with HSTS header. Redirect all HTTP to HTTPS. Use secure cookies with 'Secure' flag. Implement certificate pinning for critical applications. Use HTTPS for all communications, store passwords using strong hashing (bcrypt/argon2), avoid hardcoding credentials
STEPS TO REPRODUCE / PROOF OF CONCEPT
Step 1: Go to affected URL and open burpsuite then capture /doLogin request fill the Credentials and login in to the website..

Step 2: Username and password are sent **unencrypted** in the POST body. Even though it's HTTPS, the parameter names reveal sensitive data structure.

uid=admin&passw=admin&btnSubmit=Login

The screenshot shows the Burp Suite interface with a captured POST request and a corresponding browser response. The request details the parameters sent to the server, including 'uid=admin&passw=admin&btnSubmit=Login'. The browser screenshot shows a login page for 'Hello Admin User' where the user has been pre-approved for a credit limit of \$10000!

1.11 Click-Jacking

Click-Jacking	
SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	6.1
CWE-ID	CWE-451
OWASP CATEGORY	Insecure Design
DESCRIPTION	

Clickjacking (UI redressing) occurs when an attacker embeds a target page inside a transparent or hidden frame on an attacker-controlled site, and tricks a victim into interacting with UI elements on the framed page (buttons/links/forms) by presenting decoy content on top. The application fails to restrict or control whether it may be framed by other origins, enabling these attacks

IMPACT

Attackers create invisible frames forcing users to click hidden UI elements. Victims unknowingly perform actions like funds transfer or changing account settings. This exploits user trust in the visible website content.

AFFECTED URL

<https://demo.testfire.net/search.jsp?query=>

REFERENCE URL FOR MITIGATION

OWASP Clickjacking Defense Cheat Sheet. [cheatsheetseries.owasp.org](https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html)

OWASP Web Security Testing Guide — Testing for Clickjacking. [owasp.org](https://owasp.org/www-project-web-security-testing-guide/docs/attacks/client-side/clickjacking/)

MDN / X-Frame-Options and CSP frame-ancestors guidance. [MDN Web Docs+1](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options)

RECOMMENDATIONS

Implement X-Frame-Options: DENY header. Use Content Security Policy with frame-ancestors directive. Consider JavaScript frame-busting scripts for legacy browsers. Visual cues for sensitive actions.

STEPS TO REPRODUCE / PROOF OF CONCEPT

go to the Affected URL...

Add iframe on query= <iframe src="http://www.testfire.net/sensitive-page" width="500" height="500">

<https://testfire.net/search.jsp?query=<iframe src='http://www.testfire.net/sensitive-page' width='500' height='500'>>

<https://testfire.net/search.jsp?query=<iframe src='http://www.testfire.net/sensitive-page' width='500' height='500'>>

1.12 AccessibleByIPAddress

SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	5.5
CWE-ID	CWE-200
OWASP CATEGORY	Security Misconfiguration
DESCRIPTION	<p>The web application or service is reachable directly via its IP address (for example <code>http://1.2.3.4</code>) rather than only by an expected hostname. This can allow attackers or scanners to: discover services that were intended to be hidden, bypass host-based routing or virtual-host name checks, encounter default or management pages, cause TLS certificate name mismatches (leading to user confusion or ignoring warnings), or probe for internal/legacy interfaces.</p>
IMPACT	Bypasses WAF/DNS policies, exposing admin interfaces or APIs. Attackers scan IP ranges to find unprotected endpoints.
AFFECTED URL	http://demo.testfire.net
REFERENCE URL FOR MITIGATION	<p>OWASP – Security Misconfiguration (Top 10) – guidance on avoiding misconfigurations. https://owasp.org/www-project-top-ten/ (see A05:2021 – Security Misconfiguration)</p> <p>OWASP Host header / Host-based logic – validate Host headers and avoid trusting client-supplied hostnames for access control. https://cheatsheetseries.owasp.org/cheatsheets/Host_Header_Attacks_Prevention_Cheat_Sheet.html</p>
RECOMMENDATIONS	

Restrict access to whitelisted domains. Disable IP-based access. Use Host header validation. Configure load balancers to filter IP requests. Restrict access to application via domain name only; configure firewall to block direct IP access

STEPS TO REPRODUCE / PROOF OF CONCEPT

Try to access using this url : <http://65.61.137.117>

The screenshot shows a web browser displaying the 'AltoroMutual' demo site at the URL 65.61.137.117. The page features a green header bar with links for 'Sign In', 'Contact Us', 'Feedback', and 'Search'. Below the header, there's a banner for 'DEMO SITE ONLY' featuring several small images of people. The main content area is divided into several sections:

- ONLINE BANKING LOGIN**: Includes links for Personal (Deposit Product, Checking, Loans, Cards, Investments & Insurance, Other Services) and Small Business (Deposit Products, Lending Services, Cards, Business, Retirement, Other Services).
- PERSONAL**: Shows a couple hugging in front of a house with a 'SELL' sign, with text about Online Banking with FREE Online BILL Pay.
- SMALL BUSINESS**: Shows a stack of credit cards with text about Business Credit Cards.
- INSIDE ALTORO MUTUAL**: Shows a large group photo of employees with text about Privacy and Security.
- Real Estate Financing**: Text about real estate financing.
- Business Credit Cards**: Text about business credit cards.
- Retirement Solutions**: Text about retirement solutions.
- Win a Samsung Galaxy S10 smartphone**: Text about a contest.

At the bottom, there's a note about the site being open source and a copyright notice for HCL Technologies, Ltd. from 2008, 2017, 2025.

1.13 HTML Injection

HTML Injection	
SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	6.4
CWE-ID	CWE-80

OWASP CATEGORY	Identification and Authentication Failures
DESCRIPTION	HTML injection occurs when an application includes untrusted input in HTML pages without proper validation or output encoding, allowing an attacker to inject arbitrary HTML (and often JavaScript) that is rendered in other users' browsers
IMPACT	Unescaped HTML alters page structure, creates fake UI elements, or delivers phishing content. Often precursor to XSS.
AFFECTED URL	https://demo.testfire.net/search.jsp?query=
REFERENCE URL FOR MITIGATION	OWASP Web Security Testing Guide – Testing for HTML Injection / XSS. owasp.org PortSwigger – Injecting into direct HTML & XSS resources. portswigger.net+1
RECOMMENDATIONS	Implement output encoding. Use CSP to restrict inline HTML. Sanitize with libraries like DOMPurify. Sanitize and encode all user inputs; use security libraries to handle HTML content safely.
STEPS TO REPRODUCE / PROOF OF CONCEPT	Go to affected url and type is payload <h1><i>HACKED</i></h1>

AltoroMutual

[Sign Off](#) | [Contact Us](#) | [Feedback](#) | [Search](#) | [Go](#)

  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

Hello John Smith

Welcome to Altoro Mutual Online.

View Account Details:

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](#) | © 2025 Altoro Mutual, Inc.

This web application is open source! Get your copy from [GitHub](#) and take advantage of advanced features.

The Altoro2 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/jeoscan/>.

Copyright © 2008, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved.

AltoroMutual

[Sign Off](#) | [Contact Us](#) | [Feedback](#) | [Search](#) | [Go](#)

  DEMO SITE ONLY

MY ACCOUNT **PERSONAL** **SMALL BUSINESS** **INSIDE ALTORO MUTUAL**

PERSONAL

- Deposit Product
- Checkbook
- Loan Products
- Cards
- Investments & Insurance
- Other Services

SMALL BUSINESS

- Deposit Products
- Lending Services
- Cards
- Insurance
- Retirement
- Other Services

INSIDE ALTORO MUTUAL

- About Us
- Contact Us
- Locations
- Investor Relations
- Press Room
- Careers
- Subscribe

Search Results

No results were found for the query:

Hacked

The Altoro2 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/jeoscan/>.

Copyright © 2008, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved.

1.14 Unvalidated URL Redirection

Unvalidated URL Redirection	
SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	5.0
CWE-ID	CWE-601
OWASP CATEGORY	Security Misconfiguration
DESCRIPTION	The application allows redirection to external URLs without validation, enabling attackers to craft malicious links that redirect users to untrusted websites.
IMPACT	Attackers can trick users into visiting malicious websites by exploiting unvalidated redirects, leading to phishing, credential theft, or malware distribution.
AFFECTED URL	https://demo.testfire.net/search.jsp?query=
REFERENCE URL FOR MITIGATION	CWE-601 — URL Redirection to Untrusted Site. https://cwe.mitre.org/data/definitions/601.html

OWASP – Unvalidated Redirects and Forwards.

https://owasp.org/www-community/attacks/Unvalidated_Redirects_and_Forwards

RECOMMENDATIONS

Validate and restrict redirect URLs to a trusted allowlist or internal paths only, and avoid using user-controlled input directly for redirection.

STEPS TO REPRODUCE / PROOF OF CONCEPT

Step 1 : go to the affected url

Step 2 : type the payload on the url

payload : <p> Google Home </p>

The screenshot shows a web browser window with the URL <https://testfire.net/search.jsp?query=<a href%3Dhttps%3A%2F%2Fgoogle.com><p>+Google+Home+<%2fp>+<%2fa>>. The page title is "Altoro Mutual". The top navigation bar includes links for "Sign In", "Contact Us", "Feedback", "Search", and "Go". A banner on the right says "DEMO SITE ONLY". The main content area has tabs for "ONLINE BANKING LOGIN", "PERSONAL", "SMALL BUSINESS", and "INSIDE ALTORO MUTUAL". The "PERSONAL" tab is active, showing a sidebar with categories like "Deposit Product", "Checkings", "Loan Products", etc., and a "Search Results" section with the message "No results were found for the query: Google.Home". At the bottom, there's a footer with links for "Privacy Policy", "Security Statement", "Server Status Check", "REST API", and copyright information from 2008-2017.

1.15 2FA Not Implemented

2FA Not Implemented	
SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	6.8
CWE-ID	CWE-307
OWASP CATEGORY	Identification and Authentication Failures
DESCRIPTION	<p>The application relies solely on single-factor authentication (username and password) and does not implement Two-Factor Authentication (2FA) or Multi-Factor Authentication (MFA). Without a second verification factor, compromised</p>

credentials (through phishing, brute force, or credential stuffing) can directly grant attackers access to user accounts or administrative interfaces.

IMPACT

Accounts vulnerable to credential stuffing. No defense against password reuse or phishing.

AFFECTED URL

<https://demo.testfire.net/login.jsp>

REFERENCE URL FOR MITIGATION

OWASP Authentication Cheat Sheet –

https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

CWE-287: Improper Authentication – <https://cwe.mitre.org/data/definitions/287.html>

RECOMMENDATIONS

Implement TOTP (Google Authenticator) or WebAuthn. Provide backup codes. Educate users. Implement two-factor authentication (2FA) for all sensitive user accounts and critical functionality

STEPS TO REPRODUCE / PROOF OF CONCEPT

step 1: Go to the affected url

Enter valid credentials (e.g., admin:admin) and capture the request using **Burp Suite Repeater**.

step 2 :

Access Granted:

User is successfully logged in as **Admin** and redirected to /bank/main.jsp, confirming that no multi-factor authentication mechanism is implemented.

Observation:

The system authenticates users using **only username and password**, without verifying user identity through **2FA methods** like email OTP, SMS token, or authenticator app.

1.16 CAPTCHA Not Found

CAPTCHA Not Found	
SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	6.5
CWE-ID	CWE-799
OWASP CATEGORY	Identification and Authentication Failures
DESCRIPTION	<p>The application's login, registration, or password reset page does not implement a CAPTCHA mechanism.</p> <p>Without CAPTCHA, attackers can automate requests such as brute-force login attempts, credential stuffing, or spam submissions using bots, leading to account compromise or service disruption.</p>
IMPACT	<p>Login/registration forms without CAPTCHA allow unlimited automated attempts.</p> <p>Enables credential stuffing attacks.</p>
AFFECTED URL	https://demo.testfire.net/login.jsp
REFERENCE URL FOR MITIGATION	<p>Google reCAPTCHA Documentation</p> <p>https://developers.google.com/recaptcha</p>

OWASP Authentication Cheat Sheet

https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

RECOMMENDATIONS

Implement reCAPTCHA v3 or hCAPTCHA. Monitor for abnormal traffic patterns. Combine with rate limiting. Implement CAPTCHA or reCAPTCHA to prevent automated bots from abusing authentication and input forms

STEPS TO REPRODUCE / PROOF OF CONCEPT

Step 1: Navigate to the login page (<https://demo.testfire.net/login.jsp>) and observe that no CAPTCHA or verification mechanism is present on the login form.

Step2 : Attempt multiple login submissions with different credentials – the application allows repeated requests without any CAPTCHA challenge or rate limiting.

The screenshot shows a web browser window with the URL <https://demo.testfire.net/login.jsp>. The page title is "Online Banking Login". A red error message at the top right says: "Login Failed: We're sorry, but this username or password was not found in our system. Please try again." Below the message are fields for "Username" and "Password", and a "Login" button. To the left, there's a sidebar with links for "PERSONAL" (Deposit Product, Checking, Loan Products, Cards, Investments & Insurance, Other Services), "SMALL BUSINESS" (Deposit Products, Lending Services, Cards, Insurance, Retirement, Other Services), and "INSIDE ALTORO MUTUAL" (About Us, Contact Us, Locations, Investor Relations, Press Room, Careers, Subscribe). At the bottom, there are links for Privacy Policy, Security Statement, Server Status Check, REST API, and a copyright notice for 2025 Altoro Mutual, Inc. A green banner at the top right says "DEMO SITE ONLY".

The screenshot shows the Burp Suite Professional interface. On the left, the "Repeater" tab is selected, showing a captured request and response. The request is a POST to <https://demo.testfire.net/login.jsp> with form data including "username=admin" and "password=123456". The response is a 200 OK page titled "Hello Admin User" with a congratulatory message. The main Burp Suite window shows the captured session with various requests and responses listed.

1.17 Missing X-Frame-Option Header

Missing X-Frame-Options Header	
SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	5.8
CWE-ID	CWE-1021
OWASP CATEGORY	Security Misconfiguration
DESCRIPTION	<p>The X-Frame-Options HTTP response header is used to control whether a web page can be embedded into a <frame>, <iframe>, or <object> tag on another site. If this header is missing, attackers can load your web page within a malicious site using an iframe and perform clickjacking attacks, tricking users into clicking hidden elements like “Transfer” or “Delete Account” buttons without realizing it.</p>
IMPACT	Pages can be embedded in malicious frames to trick users into unwanted actions. Particularly dangerous for authentication pages.
AFFECTED URL	https://demo.testfire.net
REFERENCE URL FOR MITIGATION	<p>OWASP Clickjacking Defense Cheat Sheet:</p> <p>https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html</p>

MDN Web Docs – X-Frame-Options:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options>

RECOMMENDATIONS

Set X-Frame-Options: DENY or SAMEORIGIN. Combine with CSP frame-ancestors for modern browsers. Add X-Frame-Options: DENY or SAMEORIGIN to prevent framing by unauthorized sources

STEPS TO REPRODUCE / PROOF OF CONCEPT

Step 1 : Open kali terminal and type this command :

nikto -h https://demo.testfire.net

It give The X-content-Type-Option header is not set...

```
(Kali㉿kali)-[~]
$ nikto -h https://demo.testfire.net
- Nikto v2.5.0

+ Target IP:      65.61.137.117
+ Target Hostname: demo.testfire.net
+ Target Port:    443

+ SSL Info:       Subject: /CN=demo.testfire.net
                  Ciphers: ECDHE-RSA-AES256-GCM-SHA384
                  Issuer: /C=GB/ST=Greater Manchester/L=Salford/O=Sectigo Limited/CN=Sectigo RSA Domain Validation Secure Server CA
+ Start Time:    2025-11-04 12:07:37 (GMT-5)

+ Server: Apache-Coyote/1.1
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion
  to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS .
+ HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.
+ HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
```

1.18 Hardcoded Version Disclosure

Hardcoded Version Disclosure	
SEVERITY	Medium
STATUS	OPEN
CVSS SCORE	6.1
CWE-ID	CWE-200
OWASP CATEGORY	Vulnerable and Outdated Components
DESCRIPTION	<p>The application or server reveals software version details (e.g., Apache, Nginx, PHP, or framework versions) within HTTP response headers, error messages, or HTML comments.</p> <p>This information can help attackers identify specific versions and exploit known vulnerabilities associated with them.</p>
IMPACT	Exact software versions allow attackers to search for known CVEs. Patch timelines become predictable. Common in HTTP headers and error pages.
AFFECTED URL	https://demo.testfire.net
REFERENCE URL FOR MITIGATION	<p>OWASP – Information Leakage Cheat Sheet:</p> <p>https://cheatsheetseries.owasp.org/cheatsheets/Information_Leakage_Cheat_Sheet.html</p>

Apache Server Hardening Guide:

https://httpd.apache.org/docs/2.4/misc/security_tips.html

RECOMMENDATIONS

Disable Server/X-Powered-By headers. Obfuscate version info in responses. Patch promptly when updates are available. Remove version banners, headers, and comments; avoid exposing software version info in responses

STEPS TO REPRODUCE / PROOF OF CONCEPT

Step 1: Open kali terminal.

Step 2: Using Curl tool to get information about the server or other information about the website..

```
└─(kali㉿kali)-[~]
└─$ curl -I https://demo.testfire.net
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=E3F6B39CC5E1926E0C8ACDEEA65DA877; Path=/; Secure; HttpOnly
Content-Type: text/html;charset=ISO-8859-1
Transfer-Encoding: chunked
Date: Tue, 04 Nov 2025 19:06:57 GMT
```

Step 3: Step 2: Using Whatweb tool to get information about the server or other information about the website.

```
(kali㉿kali)-[~]
└─$ whatweb -v -a 3 https://demo.testfire.net
whatWeb report for https://demo.testfire.net
Status : 200 OK
Title  : Altore Mutual
IP    : 65.61.197.117
Country : UNITED STATES, US

Summary  : Apache, Cookies[JSESSIONID], HTTPServer[Apache-Coyote/1.1], HttpOnly[JSESSIONID], Java

Detected Plugins:
[ Apache ]
The Apache HTTP Server Project is an effort to develop and
maintain an open-source HTTP server for modern operating
systems including UNIX and Windows NT. The goal of this
project is to provide a secure, efficient and extensible
server that provides HTTP services in sync with the current
HTTP standards.

Google Dorks: (3)
Website   : http://httpd.apache.org/

[ Cookies ]
Display the names of cookies in the HTTP headers. The
values are not returned to save on space.

String     : JSESSIONID

[ HTTPServer ]
HTTP server header string. This plugin also attempts to
identify the operating system from the server header.

String     : Apache-Coyote/1.1 (from server string)

[ HttpOnly ]
If the Httponly flag is included in the HTTP set-cookie
response header and the browser supports it then the cookie
cannot be accessed through client side script - More Info:
http://en.wikipedia.org/wiki/HTTP\_cookie

String     : JSESSIONID

[ Java ]
Java allows you to play online games, chat with people
around the world, calculate your mortgage interest, and
view images in 3D, just to name a few. It's also integral
to the intranet applications and other e-business solutions
that are the foundation of corporate computing.

Website   : http://www.java.com/

HTTP Headers:
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=FC2CF1F40C587F8C5BB32B735333CC7; Path=/; Secure; HttpOnly
Content-Type: text/html;charset=ISO-8859-1
Transfer-Encoding: chunked
Date: Tue, 04 Nov 2025 19:10:28 GMT
Connection: close
```

1.19 Missing X-Content-Type-Options Header

Missing X-Content-Type-Options Header	
Severity	Low
Status	Open
CVSS Score	3.7

CWE-ID	CWE-16
OWASP CATEGORY	Security Misconfiguration
DESCRIPTION	
<p>The HTTP security header X-Content-Type-Options prevents browsers from MIME-sniffing a response away from the declared Content-Type.</p> <p>If this header is missing, browsers may try to guess the content type, allowing attackers to trick browsers into executing malicious files (e.g., a script disguised as an image or text file).</p>	
IMPACT	
Browsers may interpret files as executable content even when served with incorrect MIME types. Can lead to XSS via uploaded files.	
AFFECTED URL	
https://demo.testfire.net	
REFERENCE URL FOR MITIGATION	
<p>OWASP Secure Headers Project: https://owasp.org/www-project-secure-headers/</p> <p>MDN Web Docs – X-Content-Type-Options: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options</p>	
RECOMMENDATIONS	
Set X-Content-Type-Options: nosniff for all responses. Validate Content-Type headers match file content. Add X-Content-Type-Options: nosniff header to all responses serving content like scripts, styles, etc.	
STEPS TO REPRODUCE / PROOF OF CONCEPT	

Step 1: Go to the kali terminal

Step 2: Use Nikto tool to find information about the header or other technology used by the website..

```
(kali㉿kali)-[~]
$ nikto -h https://demo.testfire.net
- Nikto v2.5.0

+ Target IP:      65.61.137.117
+ Target Hostname: demo.testfire.net
+ Target Port:    443

+ SSL Info:       Subject: /CN=demo.testfire.net
                  Ciphers: ECDHE-RSA-AES256-GCM-SHA384
                  Issuer: /C=GB/ST=Greater Manchester/L=Salford/O=Sectigo Limited/CN=Sectigo RSA Domain Validation Secure Server CA
+ Start Time:    2025-11-04 12:07:37 (GMT-5)

+ Server: Apache-Coyote/1.1
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS .
+ HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.
+ HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
```

1.20 Missing Permissions-Policy Header

Missing Permissions-Policy Header	
SEVERITY	Low
STATUS	OPEN
CVSS SCORE	3.2
CWE-ID	CWE-16
OWASP CATEGORY	Security Misconfiguration
DESCRIPTION	The Permissions-Policy HTTP header (formerly <i>Feature-Policy</i>) allows websites to control which browser features and APIs (like camera, microphone, geolocation, fullscreen, etc.) can be used by the site or embedded content.

If this header is missing, attackers may exploit embedded frames or injected scripts to abuse sensitive browser features without the site owner's control.

IMPACT

Allows unrestricted access to camera, mic, geolocation without user consent.
Privacy violations possible.

AFFECTED URL

<https://demo.testfire.net>

REFERENCE URL FOR MITIGATION

OWASP Secure Headers Project:

<https://owasp.org/www-project-secure-headers/>

MDN Web Docs – Permissions-Policy Header:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Permissions-Policy>

RECOMMENDATIONS

Implement Permissions-Policy with strict directives (e.g., geolocation=(), camera=()). Disable unused features. Add a proper Permissions-Policy header to restrict access to sensitive browser features

STEPS TO REPRODUCE / PROOF OF CONCEPT

Step 1: Open kali terminal and run command:

```
curl -I -s https://demo.testfire.net | egrep -i 'Permissions-Policy|Feature-Policy|X-Frame-Options|Content-Security-Policy' || echo 'permissions-Policy: <missing>'
```

```
(kali㉿kali)-[~]
$ curl -I -s https://demo.testfire.net | egrep -i 'Permissions-Policy|Feature-Policy|X-Frame-Options|Content-Security-Policy' || echo "Permissions-Policy: <missing>"
```

- Output showed: Permissions-Policy: <missing>
- Verified via scan — header not found in HTTP response.
- Confirms site doesn't restrict browser features like camera, mic, or geolocation.

Steps 2 :

- Go to <https://securityheaders.com>.

- Enter the target URL – <https://demo.testfire.net/> – and click **Scan**.
- This confirms the **Permissions-Policy header is missing** from the server response.

Security Report Summary



Site: <https://demo.testfire.net/>
 IP Address: 65.61.137.117
 Report Time: 05 Nov 2025 05:36:58 UTC
 Headers: ✖ Strict-Transport-Security ✖ Content-Security-Policy ✖ X-Frame-Options ✖ X-Content-Type-Options
✖ Referrer-Policy ✖ Permissions-Policy

Advanced: Ouch, you should work on your security posture immediately! Start Now

Missing Headers

Strict-Transport-Security	HTTP Strict Transport Security is an excellent feature to support on your site and strengthens your implementation of TLS by getting the User Agent to enforce the use of HTTPS. Recommended value "Strict-Transport-Security: max-age=31536000; includeSubDomains".
Content-Security-Policy	Content Security Policy is an effective measure to protect your site from XSS attacks. By whitelisting sources of approved content, you can prevent the browser from loading malicious assets.
X-Frame-Options	X-Frame-Options tells the browser whether you want to allow your site to be framed or not. By preventing a browser from framing your site you can defend against attacks like clickjacking. Recommended value "X-Frame-Options: SAMEORIGIN".
X-Content-Type-Options	X-Content-Type-Options stops a browser from trying to MIME-sniff the content type and forces it to stick with the declared content-type. The only valid value for this header is "X-Content-Type-Options: nosniff".
Referrer-Policy	Referrer Policy is a new header that allows a site to control how much information the browser includes with navigations away from a document and should be set by all sites.
Permissions-Policy	Permissions Policy is a new header that allows a site to control which features and APIs can be used in the browser.

1. Checklist (Test Cases Performed)

Sr. No.	Vulnerability Name	Vulnerable (Yes/No)
1	Account Lockout Policy Not Implemented	No
2	An application sent Sensitive data in URL (GET)	No
3	Anti-CSRF Token Missing	No
4	Application Accepts Arbitrary Methods	No
5	Application does not display Last Login Date and Time	No

6	Application is accessible by IP address	yes
7	Application is Vulnerable to BEAST Attack	No
8	Application is Vulnerable to BREACH Attack	No
9	Application is Vulnerable to LOGJAM Attack	No
10	Application is Vulnerable to LUCKY13 Attack	No
11	Application is vulnerable to Non-HTML page accessible	No
12	Application is Vulnerable to POODLE Attack	No
13	Application is Vulnerable to RC4 Attack	No
14	Application is Vulnerable to ROBOT Attack	No
15	Application is Vulnerable to Secure Renegotiation	No
16	Application is Vulnerable to SWEET32 Attack	No
17	Application is Vulnerable to TLS_FALLBACK_SCSV Attack	No
18	ASP.NET Debugging Enabled	No
19	Authentication Bypass (Blind SQL Injection)	No
20	Authentication Bypass by Response Manipulation	No
21	Auto-complete is Enabled	No
22	Back Button Policy Enabled (Browser Back Refresh Attack)	No
23	Banner Grabbing	No
24	Broken Access Control (Improper Authorization)	No
25	Broken Authentication (Forced Browsing)	No

26	Broken Authentication (Improper Session Management)	No
27	Broken Links	No
28	Buffer Overflow	No
29	Business Logic Error (Business Logic Flaw)	No
30	CAPTCHA Bypass	No
31	Click-Jacking Attack	yes
32	Content Spooking Attack	No
33	CORS Misconfiguration	No
34	Credentials Transmitted to Server in Plain Text	No
35	Cross Site Scripting (XSS)	yes
36	CSRF (Cross-Site Request Forgery)	yes
37	Database Dump using sqlmap	No
38	Database is Unencrypted	No
39	Default Credentials been Used	yes
40	Directory Listing	No
41	Directory Traversal	No
42	Documentation File Found	No
43	Email Harvesting	No
44	File Upload - Content-Type Restriction Bypass	No
45	File Upload - Double Extension File Upload Vulnerability	No

46	File Upload - Magic Number File Upload	No
47	File Upload - Malicious File Upload	No
48	File Upload - No Size Restriction	No
49	File Upload - Null Byte Extension	No
50	File Upload - Unrestricted File Upload	No
51	Functionality Issue	No
52	Hardcoded Secret	No
53	HTTP Parameter Pollution	No
54	HTTP Request Smuggling	No
55	IDOR	yes
56	Improper Error Handling	No
57	Improper Input Validation	No
58	Information Disclosure	No
59	Information Disclosure (Hardcoded Version)	yes
60	Information Exposure through Query Strings in URL	No
61	Injection - CSS Injection	No
62	Injection - CSV Injection	No
63	Injection - Host Header Injection	No
64	Injection - HTML Injection	yes
65	Injection - iFrame Injection	No

66	Injection - Link Injection	No
67	Injection - OS Command Injection	No
68	Injection - Server-side Template Injection (SSTI)	No
69	Injection - XML Injection	No
70	Insecure Change Password Functionality	No
71	Insecure Communication (SSL is Not Implemented)	No
72	Internal Full Path Disclosure	No
73	Internal IP Disclosure	No
74	Local File Inclusion (LFI)	No
75	Method Interchange Attack (POST to GET)	No
76	Missing Cache-Control Header	No
77	Missing Content Security Policy (CSP) Header	No
78	Missing Expires Header	No
79	Missing HTTP Strict-Transport-Security (HSTS) Header	No
80	Missing Permissions-Policy Header	yes
81	Missing Pragma Header	No
82	Missing Referrer-Policy Header	No
83	Missing X-Content-Type-Options Header	yes
84	Missing X-Frame-Options Header	yes
85	Missing X-Permitted-Cross-Domain-Policies	No

86	Missing X-XSS-Protection Header	No
87	OTP can be Brute Force	No
88	Out-Of-Date Component Version	No
89	Parameter Tampering	No
90	Password Revealed in Response	No
91	Possible Brute Force Attack – CAPTCHA Not Found	No
92	Privilege Escalation	No
93	Race Condition	No
94	Rate Limit is Not Implemented	yes
95	Remote Code Execution (RCE)	No
96	Remote File Inclusion (RFI)	No
97	Second Factor Authentication (2FA) Not Implemented	No
98	Security Misconfiguration	No
99	Sensitive Files Disclosure	No
100	Sensitive Information Disclosed in Response	No
101	Sensitive Page Disclosure	No
102	Server Returns 403 Forbidden Response or Error	No
103	Server-Side Validations are not in Place	No
104	Session Cookie HttpOnly Attribute Not Set	No
105	Session Cookie SameSite Attribute Not Set	No

106	Session Cookie Secure Attribute Not Set	No
107	Session Fixation	No
108	Session ID remains constant before login and after logout	No
109	Session Timeout is High or Not Implemented	No
110	Simultaneous Login Enabled (Concurrent Login Allowed)	No
111	SQL Injection (Boolean-based Blind SQL Injection)	No
112	SQL Injection (Error Based SQL Injection)	No
113	SQL Injection (Time-based Blind SQL Injection)	No
114	SQL Injection (Union Based SQL Injection)	No
115	SQL Wildcard Attack	No
116	SSL Version 2.0 Protocol Detection	No
117	SSL Version 3.0 Protocol Detection	No
118	SSRF	No
119	Stack Trace Error	No
120	Test Script Page Available on Server	No
121	TLS Version 1.0 Protocol Detection	yes
122	TLS Version 1.1 Protocol Detection	yes
123	Unrestricted Field Length	No
124	Unwanted HTTP Methods Enabled	No
125	User can set New Password as Old Password	No

126	User Enumeration	No
127	Version Disclosure	No
128	Vulnerable Component Used	No
129	Weak Ciphers Enabled	No
130	Weak Encoding is Used	No
131	Weak Password Policy	No
132	XML External Entity (XXE)	No
133	XMLRPC.php File Found	No

Table 4

1. General References

Application Security Standard –

<https://owasp.org/Top10/>

<https://www.sans.org/top25-software-errors/>

<https://cert-in.org.in/>

Hardening of Servers –

<https://geekflare.com/apache-web-server-hardening-security/>

<https://geekflare.com/apache-tomcat-hardening-and-security-guide/>

<https://geekflare.com/nginx-webserver-security-hardening-guide/>

<https://geekflare.com/ibm-http-server-security-guide/>

1. Appendices

Table 1: SEVERITY LEVEL INFORMATION AND DESCRIPTION (CVSS SCORE)

Table 2: OWASP TOP 10 AND SANS CWE TOP 25

Table 3: VULNERABILITY SUMMARY KEY FINDINGS

Table 4: CHECKLIST (TEST CASES SUMMARY)

Figure 1: ASSESSMENT APPROACH

Figure 2: RISK LEVEL INFORMATION AND NECESSARY ACTIONS

Figure 3: GRAPHICAL REPRESENTATION OF VULNERABILITY SUMMARY

THE END OF DOCUMENT