GRM

Empowering and Building Trust

Web Application Security Assessment Report

Co-operative Department Government of Puducherry, India

18/07/2025

Document and Record Control

Version Control

Document Control ID	GRM-WEB-APPSEC-CO-OPERATIVE DEPARTMENT GOVERNMENT OF PUDUCHERRY-JULY-2025
Issued Date:	18th July 2025
Effective Date:	18th July 2025
Owner:	Mr. Babu Gopinathan

Revision Table

Date	Version	Brief Description	Author
08th July 2025	Draft	Co-operative Department Government of Puducherry Web Application Security Assessment Draft Report	Ms. Deepika Sukumar
18th July 2025	Final	Co-operative Department Government of Puducherry Web Application Security Assessment Final Report	Ms. Deepika Sukumar

Release Authorization

Task	Author	Title
Prepared by:	Ms. Deepika Sukumar	Cyber Security Consultant
Reviewed by:	Ms. Bhavani P	Assistant Manager-Cyber Security

Approval Authorization

Name	Title	Signature	Date
Mr. Babu Gopinathan	Director-Cyber Security	Babu G	18th July 2025

Document Distribution List

Name	Designation	Organization
Mr. K. Mahila	Head of Office, Junior Accounts Officer	Co-operative Department of Puducherry

About Us

GRM Technologies Private Limited

GRM Technologies (GRM) is a **CERTIN Empaneled (The Indian Computer Emergency Response Team (CERT–In)** Cyber Security & Information Security Audit, Advisory and Consulting company Founded in 2013, our company was launched with a view to protect and preserve the integrity of client organizations' systems and data, Head Quartered in Chennai, India and having offices in United Arab Emirates (Dubai and Abu Dhabi) and United Kingdom, which is now a trusted partner serving clients in different Industry verticals including Public Sectors, IT & ITES, Oil and Gas, Banking, Finance, NBFC, Manufacturing, Logistics, Consulting, Outsourcing, Retail, Ecommerce, Hospitality and other verticals, we operate in different geographies such as India, United States, Caribbean Islands, United Arab Emirates, United Kingdom, Canada and Europe. GRM's team served more than 1000+ Cyber Security and Information Security Engagements.

GRM Technologies Professionals have vast experience in providing a full range of Cyber and Information security services ranging from Cert–in guidelines, ISO 27001, ISO 22301, ISO 27701, ISO 27017, ISO 27018, ISO 14001, ISO 31000, SOC 1, SOC 2, GDPR, PCI DSS, HIPAA, SOX Consulting, Vulnerability Assessment and Penetration Testing, Application Security Assessments, Forensics, Incident Management, Regulatory compliance (RBI, SEBI, BSE, NSE, NSDL, CDSL,CIRCA), Privacy Framework. Our professionals provide customized / Tailored solutions to help organizations in different industry verticals to meet their Information and Cyber security needs.

Our Top Management and Core security Professionals are experienced working closely with client's board members to converse technology terms to business language which will reach all audience in an effective manner. We are proud to establish a long–term relation with all our existing clients by partnering with their management and execution team as one among their family by building Trust, Integrity and Confidence to recommend and establish right security solutions are provided at right intervals to combat Internal and External threats and vulnerabilities and applying right controls to secure their data, environment, and their processes.

GRM Contact Details

Name	**Babu Gopinathan**
Title	Director - Cyber Security
Company	GRM Technologies Pvt. Ltd.
Office address	No 9, 2nd floor, Shoba Homes, Gokul Nagar, West Tambaram, Chennai- 600045
Mobile	+971529623748, +91-9042000525
Email	babug@grmtechnologies.com

Required Details

S. No	Particulars	Details to be furnished
1	Name of the Information System Auditor/Consultant/Company	GRM Technologies (P) Limited
2	Location of Registered Office	2/123 (New No 2/127) Mani Sethupattu, Manimangalam, Sriperumbudur Taluk, Kancheepuram Dist., TN 601301 IN
3	Year of Establishment	2013
4	Mailing Address	No 9, 2nd floor, Shoba Homes, Gokul Nagar, West Tambaram, Chennai- 600045, Tamilnadu, India
5	Registered Company Number	U72900TN2013PTC091063
6	Official Contact Numbers	+91-9042000525, +91-44-22261489
7	E-mail address of the contact person	babug@grmtechnologies.com
8	Description of business and business background	• Cyber Security Consulting Services • Advisory Services • Managed Security Services • Privacy Services • Forensics Services • Training Services

Table of Contents

1. Introduction

1.1 Objective

The Cyber Security team of "GRM Technologies" was engaged to understand Co-operative Department Government of Puducherry application security posture by conducting security testing of the Co-operative Department Government of Puducherry Web Application. The objective of this security assessment was to identify vulnerabilities/weaknesses that Co-operative Department Government of Puducherry Web Application may possess and to provide recommendations for risk mitigation. Based on the outcome of the security assessment, the Co-operative Department Government of Puducherry team would develop an action plan to minimize risk in the Web application.

1.2 Scope

The security assessment was conducted on 07th July 2025 on the Co-operative Department Government of Puducherry Web Application.

The security reassessment was conducted on 18th July 2025 on the Co-operative Department Government of Puducherry Web Application.

The following activities were completed as part of the testing:

- Black Box security testing of Co-operative Department Government of Puducherry Web Application.
- The security team was provided with the Co-operative Department Government of Puducherry URL.

Details of the application tested are mentioned in the below table.

Web Application Security Assessment					
Application URL	**Application Name**	**Hash Value**	**Application Version**	**Application Released Timestamps**	**Type**
http://61.2.143.168/cod/	Co-operative Department	NA	NA	NA	Black box Assessment

1.3 Exceptions and Limitations

During the assessment, no exceptions and limitations were observed.

1.4 Manpower Involved

S. No	Name of Employee	E-mail ID		Certifications (Yes/No)												Experience in Cyber Security Audit (No. of Years)
			Masters in Cyber Security	CISA	CISM	CRISC	CGEIT	DISA	OSCP	CEH	ISO 7001	PCIDSS	ISO 2301	CompTIA	CCNA	
1	Babu Gopinathan	babug@grmtechnologies.com		Y	Y	Y		Y	Y	Y	Y	Y	Y		Y	20
2	Bhavani P	bhavanip@grmtechnologies.com								Y	Y					3
3	Deepika Sukumar	deepika@grmtechnologies.com								Y						1

1.5 Approach and Methodology

The following approach and methodology were followed for Security Assessment of Co-operative Department Government of Puducherry Web application.

PHASE I: Project Planning, Initiation and Pre-Assessment

Establish initial information, coordination, timelines, and methods of communication. In general, every type of assessment possesses a common phase of planning.

The objectives of this phase are:

- Define Objectives: Clearly define the goals and objectives of the web application security assessment, such as identifying vulnerabilities, assessing the effectiveness of security controls, or meeting compliance requirements. Incorporate OSSTMM3 control metrics to ensure the security assessment's objectives are measurable and align with OWASP Top 10 2024 vulnerabilities OWASP Web Security Testing Guide and OWASP ASVS.
- Establish Communication: Coordinate with stakeholders, including project managers, developers, and system administrators, to ensure their understanding of the assessment process and to gather relevant information about the web application.
- Obtain Necessary Permissions: Obtain authorized permissions to conduct the security assessment on the web application. Ensure verification of trust levels for all entities involved in the assessment, as suggested by OSSTMM3.

PHASE II: Information Gathering and Threat Modelling

The objective of this phase Includes Identify Application Components: Application's architecture, Secure Deployment and Configuration, Provision to Patch and update, Secure Development of Update, Patch & Release to Mitigate Against Supply Chain Risk from Developers, technology stack, frameworks, and libraries used.

Enumerate Functionalities: Understand the various functionalities and features of the web application. Overview and walk through, understanding application, fingerprinting of OS layer, application layer, web server layer, crawling, domain name lookup, port scanning, service identification

Threat Modeling: Define the Attack Surface by identifying all entry points, interfaces, and potential vulnerabilities within the web application. Use OWASP Threat Modeling techniques to assess risks and ensure alignment with OSSTMM3's multi-layer security approach. Analyze Threats and Risks associated with the identified attack surface and correlate them across various layers as per OSSTMM3.

Control Verification: Prioritize threats, determine the impact and likelihood of each threat, and verify the trustworthiness of components across different layers (OSSTMM3's trust level verification), ensuring that OWASP's API security guidelines and data validation practices are applied.

OWASP Top 10 Proactive Controls: 1. Implement Access Control, 2. Use Cryptography to Protect Data, 3. Validate all Input & Handle Exceptions, 4. Address Security from the Start, 5. Secure By Default Configurations, 6. Keep your components Secure, 7. Secure Digital Identities, 8. Leverage Browser Security Features, 9. Implement Security Logging and Monitoring, 10. Stop Server Sider Request Forgery.

PHASE III: Vulnerability Assessment and Penetration Testing

Vulnerability Assessment and Penetration Testing: - The activities performed in this phase includes vulnerability detection by using Automated Scanning: Utilize automated scanning tools to identify common vulnerabilities, including outdated software versions, misconfigurations, and known web application vulnerabilities. Manual Testing: Conduct manual penetration testing to discover complex vulnerabilities that automated scanners might miss. This involves attempting to exploit identified vulnerabilities and testing for potential attack scenarios as per both OWASP Top 10 2024 vulnerabilities OWASP Web Security Testing Guide, OWASP ASVS and OSSTMM3 guidelines. Authentication and Authorization Testing: Verify the robustness of authentication and authorization mechanisms, such as weak passwords, insecure session management, and privilege escalation. Input Validation and Output Encoding: Validate and sanitize input data to prevent common web application vulnerabilities such as Cross-Site Scripting (XSS), SQL injection, and Command Injection. Session Management and Error Handling. Secure Communication. Data Protection and Storage. Web application security assessment team followed Grey Box testing approach. Any unexpected reaction from the Application(s) was noted and investigated. The goal of this method was to identify the vulnerabilities in the application(s) and underlying server(s).

PHASE IV: Analysis and Risk Assessment

The activities performed in this phase are:

- Analyze Findings: Consolidate and analyze the results of vulnerability assessments and penetration tests. Correlate findings to ISO/IEC, Cyber Security Audit Baseline Requirements, Open Source Security Testing Methodology Manual (OSSTMM3) multi-layer security principles to ensure comprehensive coverage, OWASP Web Security Testing Guide, Method of verifications to compliance with CERT-In Directions issued by CERT-In, OWASP Top 10 2024 vulnerabilities and OWASP ASVS.
- Identify Risks: Evaluate the impact and likelihood of identified vulnerabilities and prioritize them based on their severity.
- Risk Mitigation: Recommend appropriate mitigation strategies and countermeasures to address identified risks and vulnerabilities.

PHASE V: Report Generation and Deliverable

The deliverables consist of a report describing the activities performed, identified vulnerabilities and recommendations for mitigating the vulnerabilities and the areas that needs improvement, with the leading practice recommendations. The project management is performed throughout the engagement duration.

Penetration testing is essentially carried out to generate proof of concepts (POC) for the identified vulnerabilities in the application.

Following is the methodology:

- Perform security assessment using tools such as Acunetix, Burp Suite Professional, Nmap etc. to identify the vulnerabilities.
- Identify vulnerabilities through manual techniques.
- Identify and remove false positives

- Assign severity ratings to identified vulnerabilities and generate Proof of Concept (POC), with the focus on measurable outcomes in line with OSSTMM3 control metrics and OWASP guidelines.

1.6 CERT-In Guidelines followed during the assessment

GRM Technologies Private Limited includes the verification of compliance to CERT-In direction:

- "Directions under sub-section (6) of section 70B of the Information Technology Act, 2000 relating to information security practices, procedure, prevention, response and reporting of cyber incidents for Safe & Trusted Internet" dated 28 April 2022.
- "Method of verifications to compliance with CERT-In Directions issued on 28.04.2022".
- "Guidelines for Secure Application Design, Development, Implementation & Operations" issued by CERT-In.

1.7 Risk Rating

The scale mentioned below has been used to rate the risk of the observed vulnerabilities:

Risk Rating	Description
Critical	Any vulnerability that has a high potential of causing program failure (inability to achieve minimum acceptable requirements) with high likelihood of occurrence and low attack complexity due to which, a threat actor or attacker could gain privileged access and disrupt the normal flow of operations or for which there is a publicly available mechanism to exploit the Vulnerability.
High	Any vulnerability that has a high potential of impacting business operations leading to the downtime or the disruption and provides an attacker with the privileged access, customer service, or SLA breach resulting in a significant outage or penalty to any of the customer services
Medium	Any vulnerability that has the potential of indirectly giving access to an intruder and/or does not have the features enabled for collecting evidence or notifying the attacker of unauthorized use for taking legal action in case the vulnerability gets exploited. This type of vulnerability upon exploitation might result in the elevation of privileges or slowing down the operations.
Low	Any vulnerability that has the potential of revealing the information about the device may lead to unauthorized access to the system leading to the compromise. The higher work factor would be involved in exploiting this type of vulnerability.
Informational	IT security risk is the potential harm to processor-related information resulting from some purposeful or accidental event that negatively impacts the process or the related information.

2. Executive Summary

The following sections in the report highlight the key findings as well as the analysis of the vulnerabilities identified in the Co-operative Department Government of Puducherry Web Application.

It is highly recommended that the recommendations should be applied to the complete application based on issues reported in this report.

2.1 Analysis

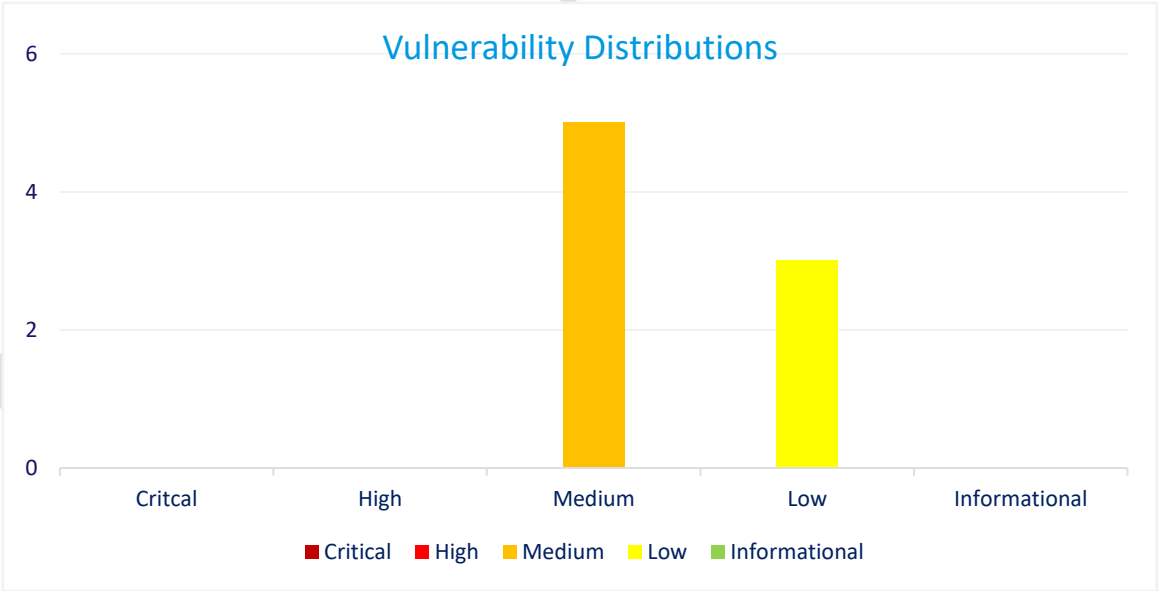
From the above-mentioned Co-operative Department Government of Puducherry Web Application in scope, **08** vulnerabilities were observed post-assessment for which breakup is mentioned below.

Critical	High	Medium	Low	Informational	Total
0	0	5	3	0	8

The table given below highlights the distribution of **08** vulnerabilities by components/platform.

#	Critical	High	Medium	Low	Informational	Total
Web Application	0	0	5	3	0	8
Total	0	0	5	3	0	8

The following graph shows the distribution of identified vulnerabilities with Critical, High, Medium, Low and Informational severity.



2.2 Vulnerability to Risk Mapping

The below table provides the mapping of each identified vulnerability along with the corresponding risk rating to vulnerable components.

S. No	Vulnerability	Risk Rating	OWASP Rating	Initial Status as on 08th July 2025	Final Status as on 18th July 2025
1.	Sensitive Information Disclosure	Medium	A05:2021	Open	Closed
2.	Unencrypted Communications	Medium	A02:2021	Open	Closed
3.	Outdated Version (PHP)	Medium	A06:2021	Open	Closed
4.	Host Header Injection	Medium	A05:2021	Open	Closed
5.	XMLRPC.PHP Vulnerability	Medium	A010:2021	Open	Closed
6.	Webpage Default Detected	Low	A05:2021	Open	Closed
7.	CSP Misconfiguration	Low	A05:2021	Open	Closed
8.	Lucky13	Low	A05:2021	Open	Closed

2.3 Prioritization

No Critical and High vulnerabilities were identified.

2.4 Overall Security Measure

The below table provides the overall security measure for the Co-operative Department Government of Puducherry web application.

Project Name	Description	Security Measure			Comments
		Poor	Controlled	Good	
Co-operative Department Government of Puducherry	The objective of this security assessment was to identify vulnerabilities/weaknesses that Co-operative Department Government of Puducherry Web Application may possess and to provide recommendations for risk mitigation http://61.2.143.168/cod/				Current Security posture is Good.

3. Detailed Technical Report

This section provides a detailed list of the observed vulnerabilities with the following information:

- Definition
- Description
- Risk Rating
- OWASP Rating
- Affected URL
- Root cause
- References
- Impact
- Recommendation(s) to minimize the risk
- Proof of Concept & Steps to reproduce Vulnerability

3.1 Assessment Findings

Below are the vulnerabilities identified during the assessment and are listed according to the risk associated with them.

3.1.1 Sensitive Information Disclosure

Parameters	Description
Vulnerability Severity	Medium
OWASP Rating	A05:2021 – Security Misconfiguration
Vulnerability Definition	phpinfo.php is a PHP script that displays detailed configuration information about the PHP environment. If exposed publicly, it may disclose sensitive server data such as paths, variables, enabled modules, and software versions. This can aid attackers in planning further exploitation or reconnaissance on the target server.
Vulnerability Description	The phpinfo() function outputs complete details about the PHP configuration, including system paths, loaded extensions, HTTP headers, environment variables, and more. When phpinfo.php is accessible to external users, it can leak sensitive configuration data that should remain internal, increasing the risk of targeted cyberattacks and information gathering.
Root Cause	Security Misconfiguration
Affected URL	http://61.2.143.168/dashboard/phpinfo.php
Impact	Public exposure of phpinfo.php can help attackers gain insights into the server's structure, software versions, and enabled components. This can lead to exploitation through known vulnerabilities, privilege escalation, or lateral movement. Even without immediate vulnerability, the disclosed data aids in crafting sophisticated and targeted attacks against the application.
References	https://cwe.mitre.org/data/definitions/200.html
Recommendation	It is recommended to: • Immediately remove or restrict access to phpinfo.php on production servers. • Use access controls (e.g., IP whitelisting) if needed for internal diagnostics. • Avoid deploying test/debug scripts to public-facing environments. • Regularly audit web root directories for unnecessary files. • Monitor access logs for attempts to access phpinfo.php.
Proof of Concept	

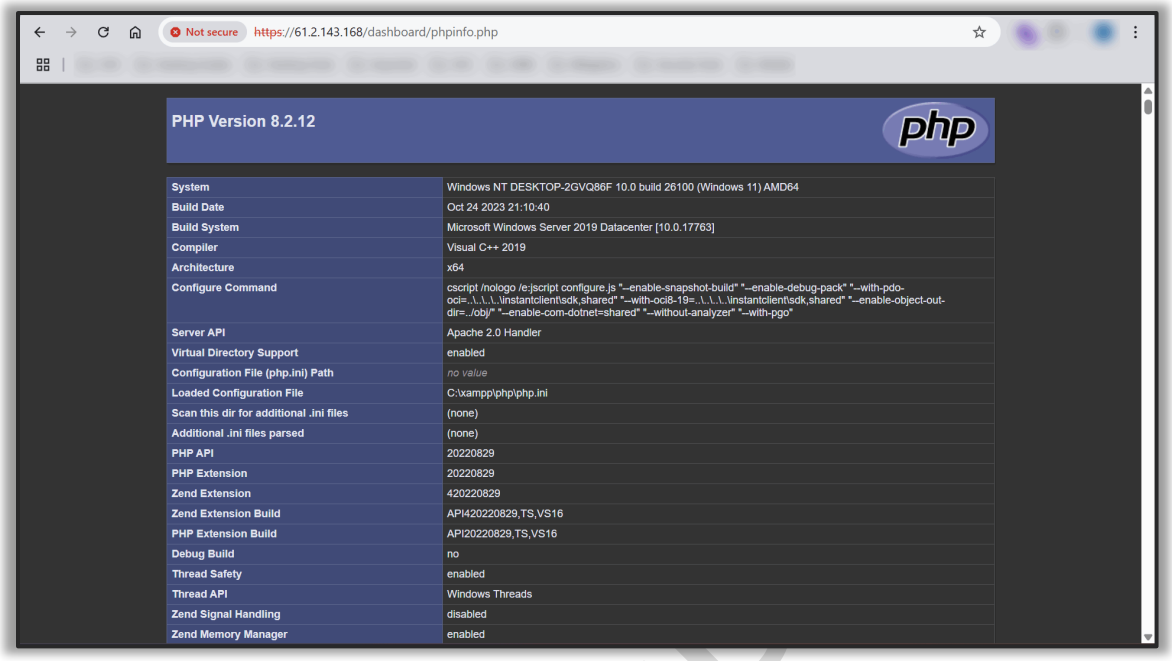


FIGURE 1-SENSITIVE INFORMATION DISCLOSURE 01

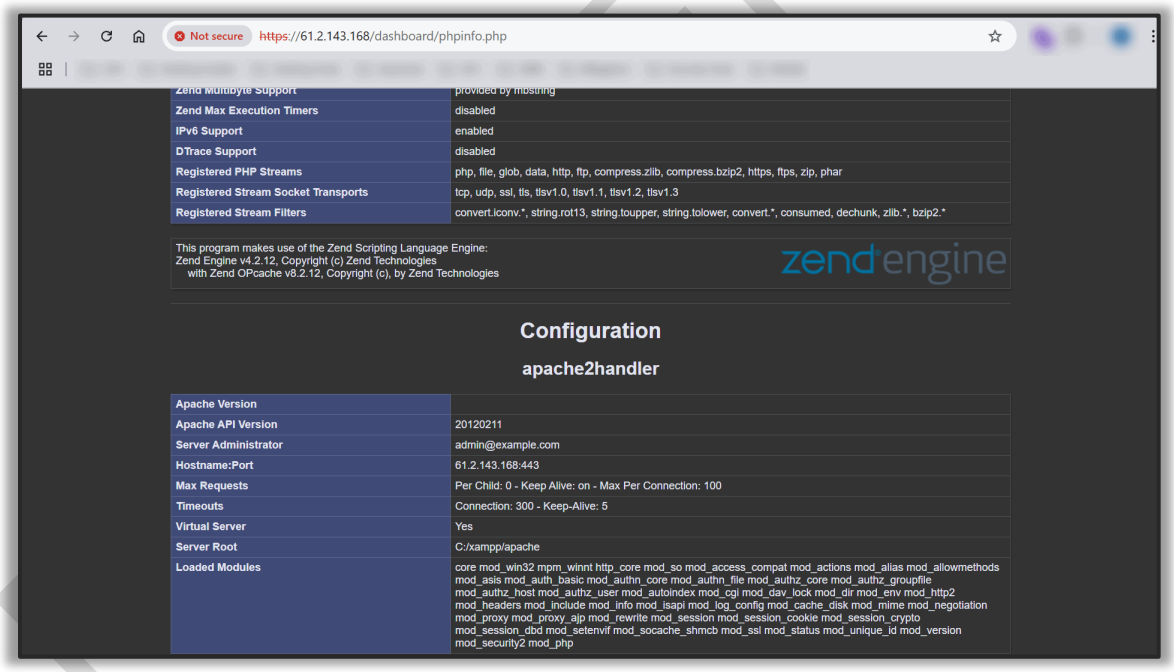


FIGURE 2-SENSITIVE INFORMATION DISCLOSURE 02

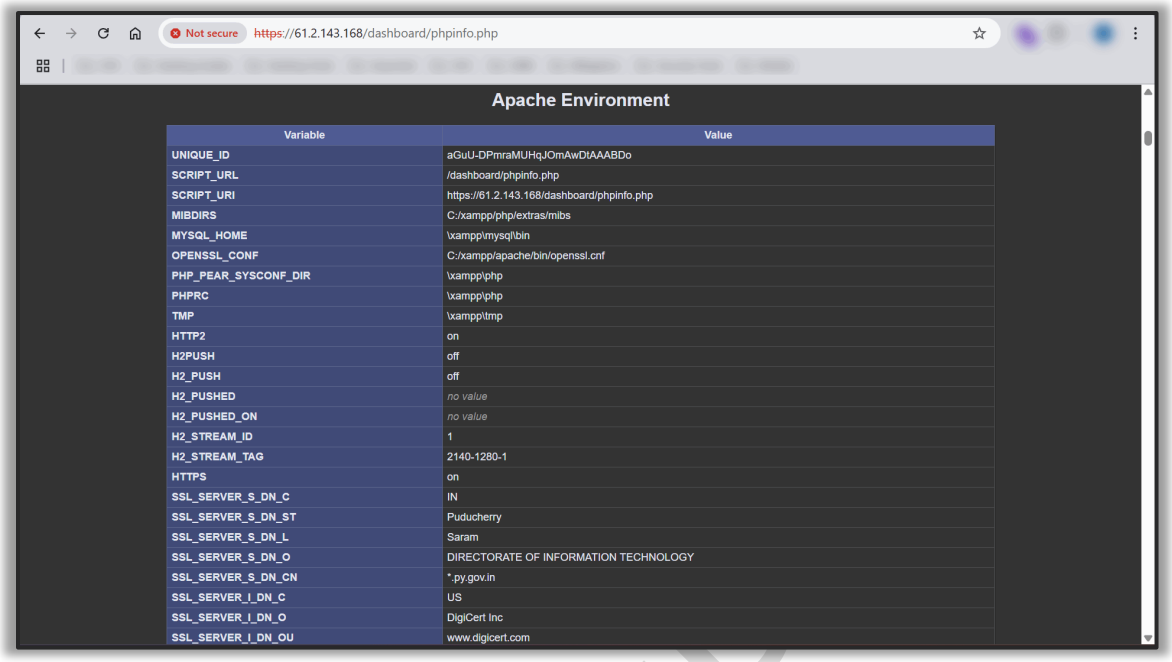


FIGURE 3-SENSITIVE INFORMATION DISCLOSURE 03

3.1.2 Unencrypted Communications

Parameters	Description
Vulnerability Severity	**Medium**
OWASP Rating	A02:2021 – Cryptographic Failures
Vulnerability Definition	Unencrypted communication refers to data transmitted over a network without encryption (e.g., HTTP, FTP, Telnet). This allows attackers to intercept and view the data in plaintext using sniffing tools. Sensitive information such as credentials, session tokens, or personal data may be exposed during transmission.
Vulnerability Description	When applications or servers transmit data without using secure protocols like HTTPS, SSH, or TLS, communication can be intercepted by malicious actors on the same network. This is especially risky on public or shared networks. Attackers can perform Man-in-the-Middle (MitM) attacks to steal or alter transmitted data.
Root Cause	Cryptographic Failures
Affected URL	http://61.2.143.168/cod/
Impact	Unencrypted traffic may expose user credentials, personal information, or session tokens, leading to account compromise, identity theft, or unauthorized access. In enterprise environments, it could result in data breaches, regulatory violations (e.g., GDPR, HIPAA), and loss of customer trust. Attackers can also inject malicious content during transmission.
References	https://cwe.mitre.org/data/definitions/319.html
Recommendation	It is recommended to: • Enforce HTTPS and disable HTTP using HSTS headers. • Use encrypted protocols like FTPS/SFTP instead of FTP, and SSH instead of Telnet. • Implement TLS 1.2 or higher for all communications. • Use valid SSL/TLS certificates from trusted Certificate Authorities (CAs). • Regularly scan for unencrypted endpoints.
Proof of Concept	

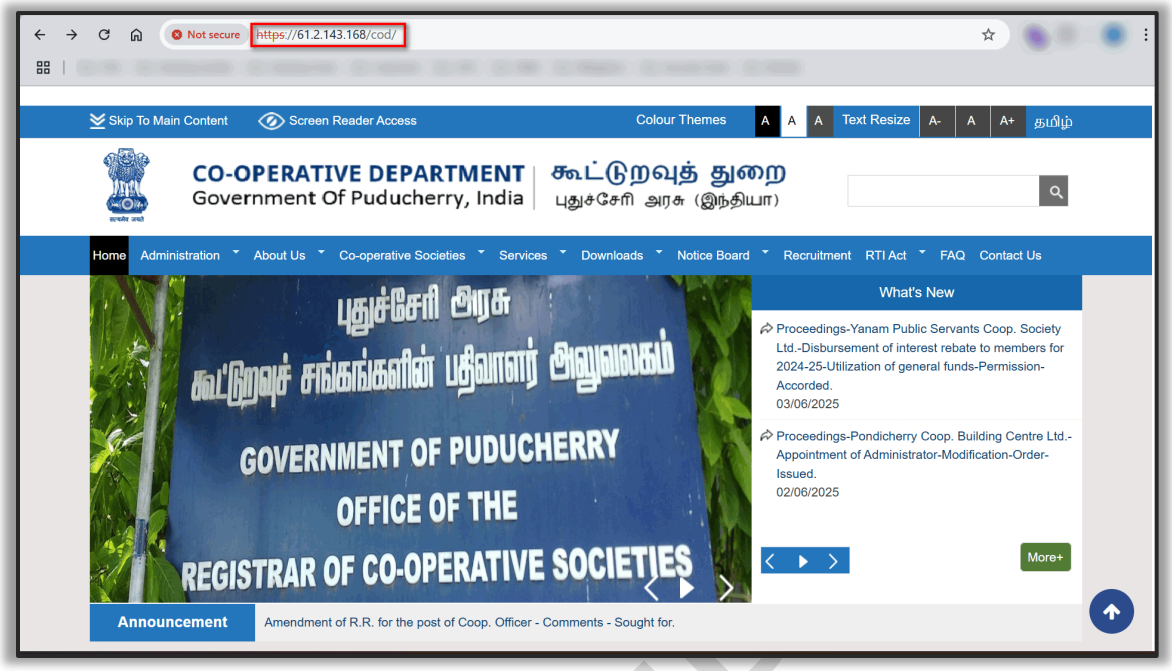
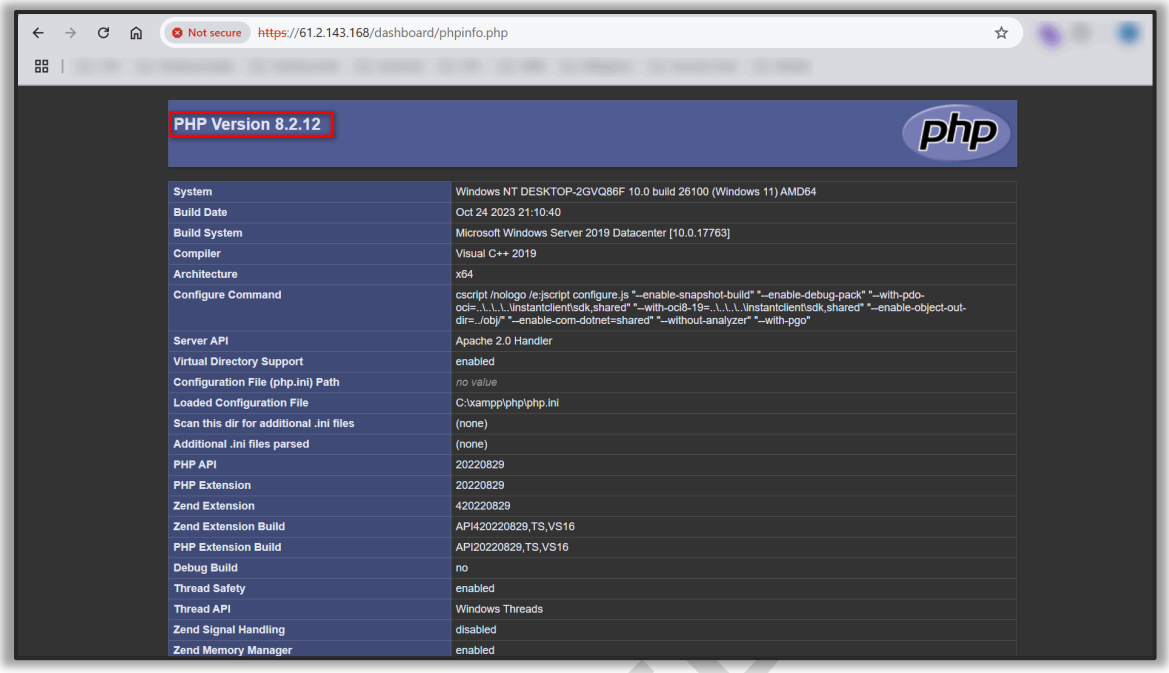


FIGURE 4-UNENCRYPTED COMMUNICATIONS

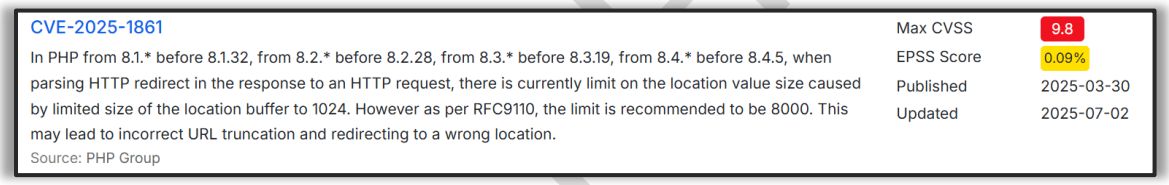
3.1.3 Outdated Version (PHP)

Parameters	Description
Vulnerability Severity	**Medium**
OWASP Rating	A06:2021 – Vulnerable and Outdated Components
Vulnerability Definition	Running an outdated version of PHP means using a PHP release that is no longer supported or maintained. These versions may contain publicly known vulnerabilities that have been fixed in newer releases, putting the application at risk of exploitation through remote code execution, information disclosure, or privilege escalation.
Vulnerability Description	Outdated PHP versions often lack security patches and contain known vulnerabilities that are actively exploited by attackers. These may include memory corruption, insecure deserialization, or arbitrary code execution flaws. Continued use of unsupported versions also signals poor maintenance practices, increasing the attack surface of the application or server.
Root Cause	Vulnerable and Outdated Components
Affected URL	http://61.2.143.168/dashboard/phpinfo.php
Impact	An outdated PHP version can be exploited to compromise the server, access or modify data, or disrupt service availability. It may also lead to non-compliance with security standards (e.g., PCI-DSS, ISO 27001). Attackers can use public exploits to target known CVEs, increasing the risk of successful breaches.
References	https://cwe.mitre.org/data/definitions/1104.html
Recommendation	It is recommended to: • Upgrade to the latest stable and supported PHP version. • Monitor PHP's official website for end-of-life (EOL) notices. • Regularly patch and test applications in a staging environment before production deployment. • Use automated tools to detect outdated software versions. • Harden the server by disabling unnecessary PHP functions.
Proof of Concept	



System	Windows NT DESKTOP-2GVQ86F 10.0 build 26100 (Windows 11) AMD64
Build Date	Oct 24 2023 21:10:40
Build System	Microsoft Windows Server 2019 Datacenter [10.0.17763]
Compiler	Visual C++ 2019
Architecture	x64
Configure Command	cscript /nologo /e:jscript configure.js "--enable-snapshot-build" "--enable-debug-pack" "--with-pdo-oci=..\..\..\instantclient\sdk,shared" "--with-oci8-19=..\..\..\instantclient\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet=shared" "--without-analyzer" "--with-pgo"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	*no value*
Loaded Configuration File	C:\xampp\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20220829
PHP Extension	20220829
Zend Extension	420220829
Zend Extension Build	API420220829,TS,VS16
PHP Extension Build	API20220829,TS,VS16
Debug Build	no
Thread Safety	enabled
Thread API	Windows Threads
Zend Signal Handling	disabled
Zend Memory Manager	enabled

FIGURE 5-OUTDATED VERSION (PHP) 01



CVE-2025-1861

In PHP from 8.1.* before 8.1.32, from 8.2.* before 8.2.28, from 8.3.* before 8.3.19, from 8.4.* before 8.4.5, when parsing HTTP redirect in the response to an HTTP request, there is currently limit on the location value size caused by limited size of the location buffer to 1024. However as per RFC9110, the limit is recommended to be 8000. This may lead to incorrect URL truncation and redirecting to a wrong location.

Source: PHP Group

Max CVSS	9.8
EPSS Score	0.09%
Published	2025-03-30
Updated	2025-07-02

FIGURE 6-OUTDATED VERSION (PHP) 02

3.1.4 Host Header Injection

Parameters	Description
Vulnerability Severity	**Medium**
OWASP Rating	A05:2021 – Security Misconfiguration
Vulnerability Definition	Host Header Injection occurs when an application relies on the Host header value in an HTTP request without properly validating or sanitizing it, potentially allowing attackers to manipulate how the server generates links, performs redirects, or constructs responses.
Vulnerability Description	During the assessment, it was observed that the application accepts arbitrary Host header values without validation. This could allow attackers to perform cache poisoning, password reset poisoning, web cache deception, or facilitate phishing attacks by crafting malicious links that appear legitimate.
Root Cause	A05:2021 – Security Misconfiguration
Affected URL	http://61.2.143.168/cod/
Impact	Attackers could exploit this vulnerability by manipulating links in email poison caches, or redirect users to malicious domains, resulting in data theft, phishing, or session hijacking.
References	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/17-Testing_for_Host_Header_Injection
Recommendation	It is recommended to validate the Host header against a whitelist of expected values on the server-side. Avoid using Host headers for generating URLs or redirects where not necessary. Use server configurations or frameworks that enforce strict host checking.
Proof of Concept	

FIGURE 7-HOST HEADER INJECTION 01

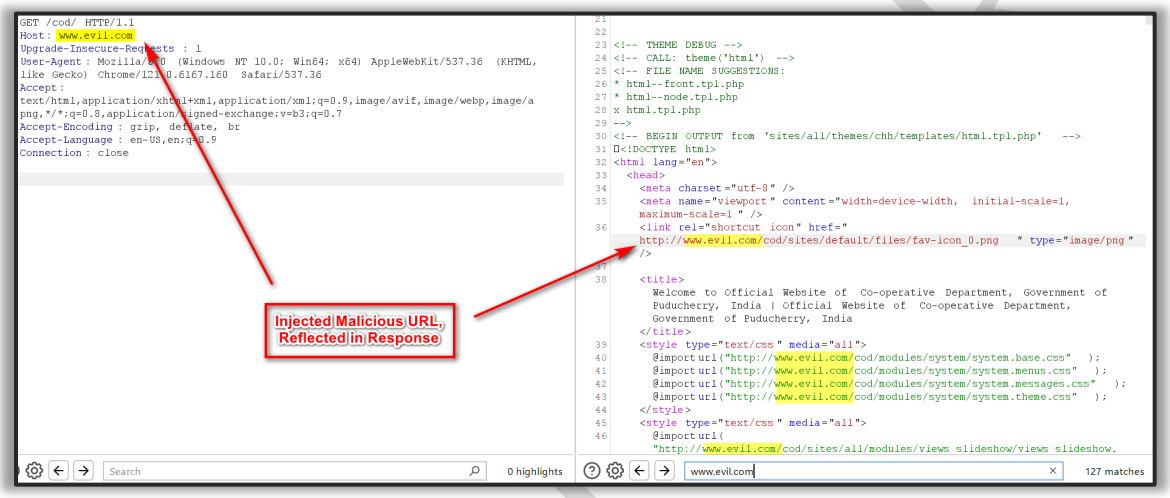


FIGURE 8-HOST HEADER INJECTION 02

3.1.5 XMLRPC.PHP Vulnerability

Parameters	Description
Vulnerability Severity	**Medium**
OWASP Rating	A10:2021 - Server-Side Request Forgery
Vulnerability Definition	XML-RPC is a protocol that allows for remote procedure calls, but misconfigurations or flaws can expose systems to attacks like unauthorized access, information disclosure, or service exploitation.
Vulnerability Description	An XML-RPC endpoint can be abused to perform remote actions on the server, such as unauthorized access or DoS attacks, often due to insecure configurations or exposed interfaces.
Root Cause	A10:2021 - Server-Side Request Forgery
Affected URL	http://61.2.143.168/cod/
Impact	Unauthorized access, DoS (Denial of Service) attacks, remote command execution, or data exfiltration.
References	https://owasp.org/www-community/attacks/Server_Side_Request_Forgery
Recommendation	It is recommended to disable or properly secure XML-RPC services are not required. Implement access controls and validate inputs to prevent unauthorized access.

Proof of Concept

FIGURE 9-XMLRPC.PHP VULNERABILITY

3.1.6 Webpage Default Detected

Parameters	Description
Vulnerability Severity	Low
OWASP Rating	A05:2021 – Security Misconfiguration
Vulnerability Definition	A default webpage or welcome page is presented by the web server when accessed, often indicating default or incomplete configuration.
Vulnerability Description	The server displays a default page which can leak information about the underlying technology stack and suggest a lack of hardening.
Root Cause	Security Misconfiguration
Affected URL	https://61.2.143.168/root/
Impact	Can disclose server software and version; may signal unmaintained or misconfigured systems to attackers.
References	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/02-Testing_for_Default_Credentials
Recommendation	It is recommended to replace or remove default pages, customize error and index pages, and harden web server configurations.

Proof of Concept

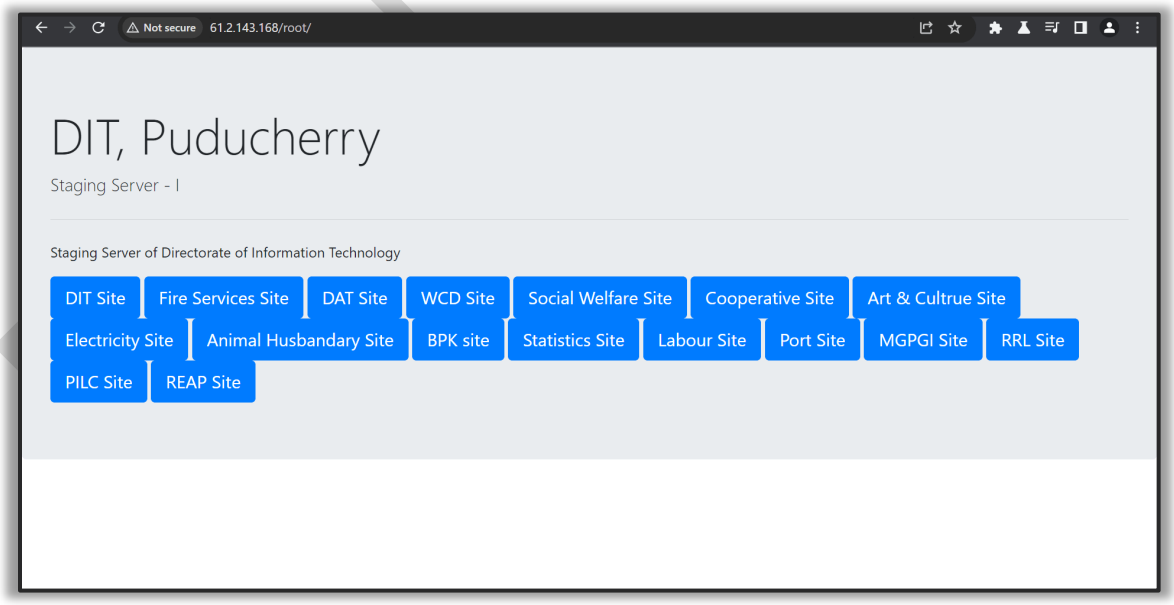


FIGURE 10-WEBPAGE DEFAULT DETECTED

3.1.7 CSP Misconfiguration

Parameters	Description
Vulnerability Severity	**Low**
OWASP Rating	A05:2021 – Security Misconfiguration
Vulnerability Definition	CSP (Content Security Policy) misconfiguration occurs when the security headers intended to restrict resource loading are improperly defined. This allows attackers to bypass restrictions and execute malicious scripts, leading to vulnerabilities such as Cross-Site Scripting (XSS), data injection, or content spoofing—even when CSP is intended as a security control.
Vulnerability Description	A misconfigured CSP may use overly permissive directives like unsafe-inline, wildcards (*), or omit important policies like script-src or object-src. Such configurations fail to effectively block malicious scripts from executing or loading from untrusted domains, thereby negating the intended security benefits of CSP headers and increasing XSS risks.
Root Cause	Security Misconfiguration
Affected URL	http://61.2.143.168/cod/
Impact	Improper CSP settings can leave web applications vulnerable to script injection, clickjacking, data theft, or session hijacking. Attackers can exploit weak or missing directives to bypass browser-enforced security, making even modern browsers ineffective at protecting users from XSS and other client-side attacks. This could lead to account or data compromise.
References	https://www.invicti.com/blog/web-security/content-security-policy/?_gl=1*1eyyd8a*_gcl_au*MTQ4MjQ0NzE3LjE3NDU5OTQyNjc.
Recommendation	It is recommended to: • Define a strong and specific Content-Security-Policy header. • Avoid using unsafe-inline, unsafe-eval, or wildcard * unless absolutely necessary. • Whitelist only trusted domains for script-src, style-src, img-src, etc. • Use browser testing tools (like Google CSP Evaluator) to validate policies. • Monitor CSP violations using report-uri or report-to.
Proof of Concept	

FIGURE 11-CSP MISCONFIGURATION

3.1.8 Lucky13

Parameters	Description
Vulnerability Severity	**Low**
OWASP Rating	A02:2021 – Cryptographic Failures
Vulnerability Definition	Lucky13 is a cryptographic timing attack targeting the implementation of TLS/SSL protocols, specifically when using CBC (Cipher Block Chaining) mode. It exploits subtle timing differences during decryption and padding verification processes, enabling attackers tog recover plaintext from encrypted TLS traffic under certain conditions.
Vulnerability Description	Lucky13 attacks take advantage of how TLS implementations process padding in CBC-mode encryption. The attacker repeatedly sends modified ciphertexts and measures server response times. Variations in processing time reveal information about the decrypted plaintext. This side-channel attack can potentially lead to session hijacking, data leakage, or credential recovery over time.
Root Cause	Cryptographic Failures
Affected URL	http://61.2.143.168/
Impact	If successful, the Lucky13 attack can allow an attacker to decrypt sensitive information like session cookies, authentication tokens, or other confidential data transmitted over TLS. Although complex and timing-sensitive, it compromises the confidentiality of TLS sessions, especially when older or unpatched libraries are used (e.g., OpenSSL pre-1.0.1g).
References	https://wiki.crashtest-security.com/prevent-ssl-lucky13
Recommendation	It is recommended to: • Use TLS 1.2 or higher with AEAD ciphers (like GCM or ChaCha20-Poly1305) instead of CBC mode. • Ensure all cryptographic libraries (e.g., OpenSSL, NSS) are updated. • Disable legacy TLS versions and weak cipher suites.
Proof of Concept	

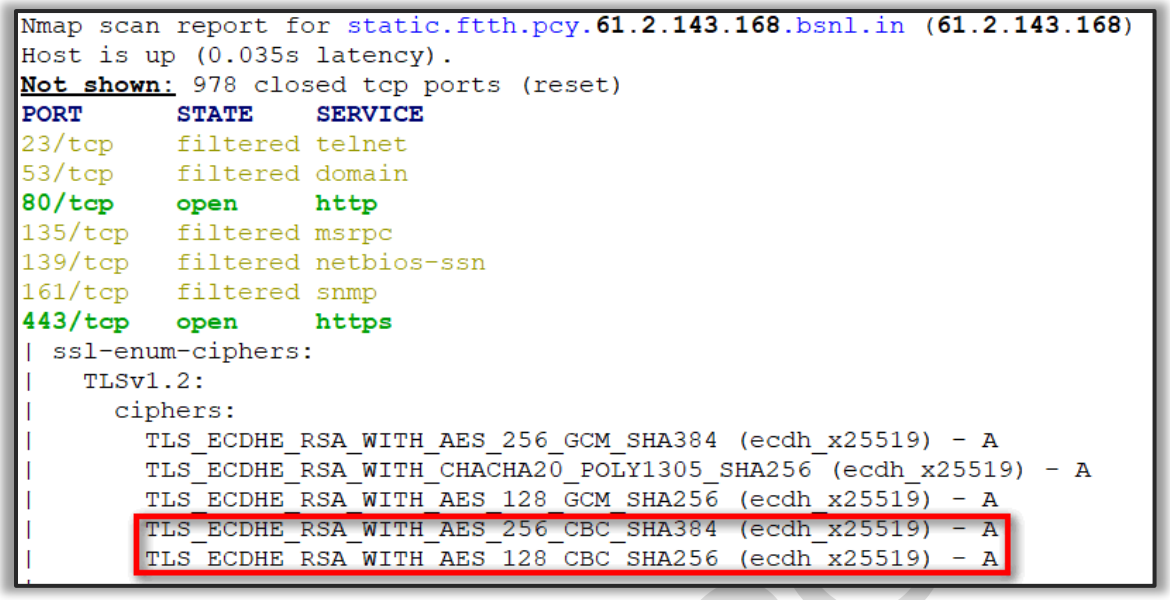


FIGURE 12-LUCKY13

Annexure A – Security Assessment Test Case Sheet

The following sheet contains information of the various test cases used while conducting Web application security assessment.

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
1	Reflected Cross Site Scripting	In Reflected XSS attack, the injected code is reflected off the web server, such as in popup from executed script, error message, or any other response that includes some or all the input sent to the server as part of the request. When a user is tricked into clicking on a malicious link or submitting a specially crafted form, the injected code travels to the vulnerable web server, which reflects the attack back to the user's browser.	Not Vulnerable
2	Stored Cross Site Scripting	In Stored XSS attacks, the injected malicious code is permanently stored on the target servers, such as in a database, in a message forum, visitor log, comment field, etc. when victim requests to the stored information, the malicious script gets executed on his browser.	Not Vulnerable
3	DOM based Cross Site Scripting	In the DOM Based XSS attack, malicious payload is executed as a result of modifying the DOM (Document Object Model) "environment" in the victim's browser used by the original client-side script, so that the client-side code runs in an "unexpected" manner.	Not Vulnerable
4	Injections - SQL, LDAP, Code Injections	Injection is an attack where user input is directly appended to the query without any validation. The attacker may be able to inject control characters into the query and change the entire structure of the query. Additional conditions can be added to the query, or parts of the query can be commented out. This can lead to leakage of data that the attacker is not authorized to see by allowing the attacker to run any arbitrary SQL statement against the database.	Not Vulnerable
5	HTML Injection	HTML Injection refers to injecting HTML code into a web server's response to alter the content to the end user.	Not Vulnerable
6	Vertical & Horizontal Privilege Escalation	Privilege escalation happens when it is possible to access resources granted to more	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
		privileged account via changing some parameters.	
7	Cross Site Request Forgery Testing	A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim's browser to generate requests the vulnerable application thinks are legitimate requests from the victim.	Not Vulnerable
8	HTTP Response Splitting	HTTP response splitting occurs when the data is included in an HTTP response header sent to a web user without being validated for malicious characters.	Not Vulnerable
9	Forced Browsing	In this attack an attacker tries to enumerate and access resources that are not referenced by the application but are still accessible. This attack is performed manually when the application index directories and pages are based on number generation or predictable values or using automated tools for common files and directory names.	Not Vulnerable
10	Session Fixation	Session Fixation is an attack technique that forces a user's session ID to an explicit value. The attacker then causes the victim to authenticate against the server using the same session identifier, giving the attacker access to the user's account through the active session.	Not Vulnerable
11	Arbitrary File Upload Possible	In this attack, the affected application allows user to upload any arbitrary file through upload module. An attacker may possibly upload malicious ".exe" files on to the server.	Not Vulnerable
12	Installed application platforms are vulnerable	In this attack, the installed application platform is already vulnerable to several vulnerabilities. An attacker can use those to craft an attack on the application.	Vulnerable
13	File Inclusion Vulnerability	File Inclusion attack possible When web applications take user input (URL, parameter value, etc.) and pass them into file include commands, the web application might be tricked into including remote files with malicious code.	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
14	Unrestricted URL Access	In this attack, an attacker can get authenticated pages by just accessing the URL through browser without having credentials.	Not Vulnerable
15	Bypassing client-side validation for change password	The applications, which rely on client-side validations, are prone to this attack. In this attack, an attacker tries to change the password while bypassing the old password field for validation.	Not Vulnerable
16	Query Parameter in GET request	The get request is known as unsecure method as all the parameters through this request transmitted in URL. If application transfers any sensitive information through GET request, an attacker may exploit it to get sensitive information.	Not Vulnerable
17	Unhandled Error	Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application errors. Attackers may use this weakness to steal sensitive data or conduct more serious attacks.	Not Vulnerable
18	Parallel session and authorization check	An attacker would be unnoticed, if application allows concurrent user session to login and perform the actions.	Not Vulnerable
19	Application Shows Last User Login Date & time	As per the security best practices, it is recommended to show last user login information for End user's notice.	Not Vulnerable
20	Parameter Tampering	The Web Parameter Tampering attack is based on the manipulation of parameters exchanged between client and server in order to modify application data, such as user credentials and permissions, price, and quantity of products, etc. Usually, this information is stored in cookies, or URL Query Strings, and is used to increase application functionality and control.	Not Vulnerable
21	Hidden Parameter validation	In this attack, an attacker tries to change the values in hidden parameter through proxy or another tool. If application does not validate hidden parameter at server level, attacker may use this to modify sensitive hidden field's values.	Not Vulnerable
22	Change "To Mail Id" while send mail	If application does not validate "To Mail Id" parameter while sending mail through	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
		application, then an attacker may be able to send mails to anyone from any email id.	
23	Check for password history	If application allows setting the last used passwords for reuse, it would be easy for an attacker to guess the credentials.	Not Vulnerable
24	Password Complexity	As per the security best practices, an application must check for complexity of password.	Not Vulnerable
25	Weak SSL or Cipher testing	If the target server supports the use of weak SSL ciphers or weak algorithm, an attacker may deprecate the communication between the affected application and its client.	Vulnerable
26	Backup and unreferenced files used	If an application contains unreferenced and/or forgotten files on production server that can be used to obtain critical information about either the infrastructure or the credentials.	Not Vulnerable
27	Access to Admin interfaces	If application does not restrict open admin interface to authorized IPs, it may possible that an attacker may access it remotely and preform malicious activity.	Not Vulnerable
28	Unnecessary HTTP Methods enabled	If application allows to unnecessary HTTP methods like PUT, Delete, Trace etc. an attacker can misuse these methods to take control on server.	Not Vulnerable
29	Credential's transport over an unencrypted channel	If SSL is not implemented on the web server, sensitive information such as Username, password goes across the wire in clear text. This allows determined attacker to perform man in the middle attack on web application for capturing the credentials.	Not Vulnerable
30	User enumeration possible	If application through different error message for wrong username and wrong password, an attacker would be able to get a list of users on system.	Not Vulnerable
31	Guessable user account found	If default and guessable credentials are being used in the application, then an attacker can get access to them and may lead to compromise the application.	Not Vulnerable
32	Weak Password Forgot Functionality	If applications forgot password functionality is properly secure, an attacker may be able to get the password of another user by using weakness in it.	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
33	By-passing authentication Mechanism through Injections	If application's login page is affected from Injection attacks, the authentication mechanism can be bypassed.	Not Vulnerable
34	AutoComplete is not disabled	If the application does not prohibit autocomplete feature, the victim's browser memorizes the username and passwords in clear text, which can be easily compromised.	Not Vulnerable
35	Improper Cache Management	Web caching is the caching of web documents (e.g., HTML pages, images) in order to reduce bandwidth usage, server load, and perceived lag. A web cache stores copies of documents passing through it; subsequent requests may be satisfied from the cache if certain conditions are met. Caching should only be used on static content that does not require "real-time" access.	Not Vulnerable
36	Weak Captcha implementation on Login pages	If Captcha is not implemented properly, an attacker can easily guess the Captcha and can bypass it.	Not Vulnerable
37	Bypass Multiple Factors Authentication	If the Process of validating the second factor authentication is not properly implemented, an attacker may be able to bypass it.	Not Vulnerable
38	Weak Session generation algorithm	Session algorithm is used for creating unique and random session for each login session, which used for maintaining authenticated session. If the session algorithm is weak then an attacker may guess new session id and may access to authenticated pages.	Not Vulnerable
39	Session cookie mechanism for Pre & Post Authentication	If web application does not change the Session ID that is assigned to user's browser after successfully login and use it as authenticated session id. Then an attacker with pre-authenticated session id would be able to access the authenticated pages.	Not Vulnerable
40	Cookies attributes not set properly	Each cookie contains some attribute as secure, path, domain etc. with it. If application does not set these attributes in secure manner, an attacker may be able to get access the cookie information.	Not Vulnerable
41	Session Cookies in Clear text	If cookies are not set secure; an attacker may access it whether application using http or https.	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
42	Path Traversal testing	A Path Traversal attack aims to access files and directories that are stored outside the web root folder. By manipulating variables that reference files with “dot-dot-slash (../)” sequences and its variations, it may be possible to access arbitrary files and directories stored on file system, including configuration and critical system files etc.	Not Vulnerable
43	Buffer overflow Testing	Buffer overflow errors are characterized by the overwriting of memory fragments of the process, which should have never been modified intentionally or unintentionally. Overwriting values of the IP (Instruction Pointer), BP (Base Pointer) and other registers causes exceptions, segmentation faults, and other errors to occur. Usually, these errors end execution of the application in an unexpected way.	Not Vulnerable
44.	Credential Lockout Policy	If application does not lock the wrong credentials account attempts at login page, an attacker can brute force for the valid credentials and if gets succeed, may access authenticated pages.	Not Vulnerable
45	XPath Injection	XPath Injection attacks occur when a web site uses user-supplied information to construct an XPath query for XML data. By sending intentionally malformed information into the web site, an attacker can find out how the XML data is structured, or access data that he may not normally have access to.	Not Vulnerable
46	WSDL authentication testing	If application is not using authentication for Web services, an attacker may get access to exposed methods and may lead to further exploits.	Not Vulnerable
47	Banner Disclosed Sensitive Information	The default banners leak the version information of the application, platform etc. This information may prove vital for launch further attacks.	Not Vulnerable
48	Directory Indexing Enabled	If the web server was configured improperly, it may be possible to retrieve a directory listing by sending a request for a specific directory.	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
49	Cryptographic Storage	If web application does not properly protect sensitive data, such as SSNs, and authentication credentials, with appropriate encryption or hashing. Attackers may steal or modify such weakly protected data to conduct identity theft, or other crimes.	Not Vulnerable
50	Other Business Logic Flaws	If application has functional flaws like logout module is not implemented etc., an attacker may use such flaws to lead further attacks.	Not Vulnerable
51	HTTP Security headers	HTTP security headers are a set of lines that one can add to your website's code. It helps protect it from malicious attacks. They tell the browser what is allowed and what isn't. They can also serve as a warning against possible threats. One can use these headers alone or in conjunction with other security measures such as an SSL certificate for extra protection.	Vulnerable
52	HTTPS Connection	HTTPS, the communication protocol is encrypted using Transport Layer Security (TLS) or, formerly, Secure Sockets Layer (SSL). The protocol is therefore also referred to as HTTP over TLS	Not Vulnerable
53	Content Negotiation Flaws	The Content-Type header is used to indicate the media type of the resource. The media type is a string sent along with the file indicating the format of the file. The HTTP Accept header is a request type header. The Accept header is used to inform the server by the client that which content type is understandable by the client expressed as MIME-types. By using the Content-negotiation the server selects a proposal of the content type and informs the client of its choice with the Content-type response header.	Not Vulnerable
54	Hardcoded Sensitive Information	The software contains a hard-coded password, which it uses for its own inbound authentication or for outbound communication to external components. The use of a hard-coded cryptographic key significantly increases the possibility that encrypted data may be recovered.	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
55	Sensitive information submitted as URL Parameter	Information exposure through query strings in URL is when sensitive data is passed to parameters in the URL. This allows attackers to obtain sensitive data such as usernames, passwords, tokens (authX), database details, and any other potentially sensitive data. Simply using HTTPS does not resolve this vulnerability.	Not Vulnerable
56	Server DEBUG Mode Enabled	Debug Mode, debuggers are software tools which enable the programmer to monitor the execution of a program, stop it, restart it, set breakpoints, and change values in memory.	Not Vulnerable
57	Fingerprinting HTTP Headers Enabled	Web server fingerprinting is one of the critical tasks for a penetration tester. By knowing the version and type of a running web server allows testers to determine known vulnerabilities and the appropriate exploits to use during testing.	Not Vulnerable
58	Broken Authentication	Broken authentication is an umbrella term for several vulnerabilities that attackers exploit to impersonate legitimate users online. Broadly, broken authentication refers to weaknesses in two areas: session management and credential management.	Not Vulnerable
59	Broken Authorization	Broken Authorization happens when an application does not correctly confirm that the user performing the request has the required privileges to access a resource of another user.	Not Vulnerable
60	Excessive Data Exposure	Looking forward to generic implementations, developers tend to expose all object properties without considering their individual sensitivity, relying on clients to perform the data filtering before displaying it to the user.	Not Vulnerable
61	Lack of Rate Limiting	APIs or applications do not impose any restrictions on the size or number of resources that can be requested by the client/user. Not only can this impact the API server performance, leading to Denial of Service (DoS), but also leaves the door open to authentication flaws such as brute force	Not Vulnerable
62	Mass Assignment	Binding client provided data (e.g., JSON) to data models, without proper properties filtering based on an allowlist, usually leads to Mass Assignment. Either guessing objects	Not Vulnerable

Sr. No.	Test cases	Description	Test Result (Vulnerable / Not Vulnerable)
		properties, exploring other API endpoints, reading the documentation, or providing additional object properties in request payloads, allows attackers to modify object properties they are not supposed to.	
63	Improper Assets Management	APIs tend to expose more endpoints than traditional web applications, making proper and updated documentation highly important. Proper hosts and deployed API versions inventory also play an important role to mitigate issues such as deprecated API versions and exposed debug endpoints.	Not Vulnerable
64	Insufficient Session or Token Expiration	Insufficient Session Expiration is when a web site permits an attacker to reuse old session credentials or session IDs for authorization	Not Vulnerable
65	Insecure Direct Object References	Insecure direct object references (IDOR) are a type of access control vulnerability that arises when an application uses user-supplied input to access objects directly.	Not Vulnerable
66	Unvalidated Redirects and Forwards	This issue occurs when a web application redirects or forwards users to untrusted locations based on unvalidated user input, which could be used for phishing attacks.	Not Vulnerable
67	CSRF Token Missing or Incorrect Implementation	Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. Check whether the application uses CSRF tokens to prevent CSRF attacks and if the token implementation is secure.	Not Vulnerable
68	Test for Default Credentials	Applications or services that use default credentials (username and password) left unchanged may be exploited by attackers to gain unauthorized access.	Not Vulnerable
69	Insufficient Logging and Monitoring	This vulnerability occurs when web applications lack proper logging mechanisms, making it difficult to detect and respond to malicious activities or security breaches.	Not Vulnerable
70	Cross-Origin Resource Sharing (CORS)	Improper CORS configurations may allow an attacker to retrieve restricted resources from a web application using unauthorized domains	Not Vulnerable

Annexure B - Tools Used for Assessments

The following tools were used for this Security Assessment:

- Acunetix/ Burp Suite Pro - is a licensed comprehensive vulnerability-scanning program. Its goal is to detect potential vulnerabilities on the tested systems. For example:
- Vulnerabilities that allow a remote cracker to control or access sensitive data on a system
- Default passwords, a few common passwords, and blank/absent passwords on some system accounts. Nessus can also call Hydra (an external tool) to launch a dictionary attack
- Acunetix runs on multiple platforms and support all Operating Systems for technical/application vulnerability assessment.
- Burp Suite Professional is an integrated platform for performing security testing of web applications. Its various tools work seamlessly together to support the entire testing process, from initial mapping and analysis of an application's attack surface, through to finding and exploiting security vulnerabilities.

END OF THE REPORT