AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
E-ISSN:3069-0102
VOL.6, NO. 2 (2026)

FILE UPLOAD VULNERABILITIES AND REMOTE COMMAND EXECUTION USING DVWA

[1]Mrs T.Tejaswi, [2] P Srinithya,[3] H Neeraja, [4] P Lokesh,[5] M Sripathi
[1]AssistantProfessor, [2345]Students
Department of Cyber Security
Siddhartha Institute of Technology & Sciences, Narapally
thadaka.tejaswi@siddhartha.org.in, 24TQ1A6217@siddhartha.co.in, 24TQ1A6206@siddhartha.co.in, 24TQ1A6216@siddhartha.co.in, 25TQ5A6203@siddhartha.org.in

Abstract

File upload functionality is a common feature in modern web applications, allowing users to share images, documents, and other data. However, if not properly secured, it can introduce serious security risks. This project focuses on analyzing file upload vulnerabilities and demonstrating how they can lead to Remote Command Execution (RCE) using Damn Vulnerable Web Application. The main objective is to understand how improper validation mechanisms can allow attackers to upload malicious files and execute them on the server.

The study begins by examining how web applications process file uploads and identifying common weaknesses such as lack of proper server-side validation and over-reliance on client-side checks. Attackers can exploit these flaws using techniques like file extension manipulation, MIME type spoofing, and code obfuscation to bypass security restrictions and upload harmful scripts.

Once a malicious file is successfully uploaded, it can be executed through the web interface, enabling attackers to perform unauthorized actions such as executing system commands, accessing sensitive information, modifying files, or gaining control over the server. The project demonstrates these attacks across different security levels in DVWA, including low, medium, and high, to better understand both vulnerabilities and defense mechanisms.

Furthermore, the project highlights important mitigation strategies, including strict server-side validation, limiting file types, secure file storage, and content inspection. Overall, this study provides a comprehensive understanding of file upload vulnerabilities, their exploitation, and prevention techniques, emphasizing the importance of secure coding practices in protecting web applications from critical threats.

I. Introduction

Web applications play a vital role in modern digital systems, enabling users to perform activities such as communication, online transactions, file sharing, and data storage. Among these features, file upload functionality is widely used, allowing users to upload images, documents, and other files to a server. While this feature enhances usability and flexibility, it also introduces significant security risks if not properly implemented and validated.

One of the most critical issues associated with file upload functionality is the file upload vulnerability. This occurs when a web application fails to properly validate

Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026

uploaded files, allowing attackers to upload malicious content instead of legitimate files. These malicious files, often written as server-side scripts like PHP web shells, can be executed on the server, leading to serious security breaches. Attackers can exploit such vulnerabilities to gain unauthorized access, steal sensitive data, manipulate system files, or even take full control of the server. This type of exploitation is commonly referred to as Remote Command Execution (RCE), which is considered one of the most severe threats in web application security.

In this project, file upload vulnerabilities and their exploitation are demonstrated using Damn Vulnerable Web Application, a purposely insecure platform designed for learning cybersecurity concepts. DVWA provides multiple security levels—low, medium, and high—allowing users to understand how vulnerabilities exist under weak security conditions and how they can be mitigated through stronger defenses. This controlled environment makes it ideal for practical experimentation and learning.

The project explores various techniques used by attackers to bypass security mechanisms, including file extension manipulation, MIME type spoofing, and embedding malicious code within seemingly harmless files. After successfully uploading a malicious file, the project demonstrates how it can be executed through a web interface to perform unauthorized actions on the server, highlighting the real-world impact of insecure implementations.

II. Literature Survey

Web application security has become a major area of research due to the rapid growth of internet-based services and increasing cyber threats. Among various vulnerabilities, file upload vulnerability is considered one of the most critical because it allows attackers to directly interact with the server and potentially execute malicious code. Several researchers and cybersecurity experts have studied this vulnerability and proposed different detection and prevention techniques to mitigate its impact.

According to OWASP, file upload vulnerabilities are categorized among the most severe web application security risks. These vulnerabilities typically arise when applications fail to properly validate user inputs, especially uploaded files. Many studies point out that relying only on client-side validation is insufficient, as it can be easily bypassed by attackers. Therefore, researchers strongly emphasize the importance of robust server-side validation mechanisms to prevent unauthorized file uploads.

Previous research highlights multiple techniques used by attackers to bypass file upload restrictions. These include file extension manipulation, use of double extensions (such as .php.jpg), MIME type spoofing, and embedding malicious scripts within seemingly harmless files. Studies also reveal that weak filtering mechanisms and improper server configurations significantly increase the likelihood of successful exploitation.

To better understand and demonstrate such vulnerabilities, several tools and platforms have been developed. One of the most widely used platforms is Damn Vulnerable Web Application, which provides a controlled environment for learning and practicing web application security. Researchers and students use DVWA to analyze

Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026

vulnerabilities across different security levels and gain practical insights into real-world attack scenarios.

Recent research has focused on enhancing security measures through techniques such as file content inspection, sandboxing, and secure file storage outside the web root directory. Advanced approaches, including machine learning-based detection systems, are also being explored to automatically identify malicious file patterns. However, despite these advancements, file upload vulnerabilities continue to persist due to poor implementation practices and lack of awareness among developers.

III. System Analysis

System analysis focuses on understanding how file upload functionality in web applications can become a security risk when not properly implemented. In this project, the analysis examines how users upload files to a server and how improper validation mechanisms can allow malicious files to bypass security checks. The system studies the behavior of web applications under different security levels using Damn Vulnerable Web Application. It identifies weaknesses such as lack of server-side validation, improper file handling, and insecure storage. The analysis also evaluates how attackers exploit these weaknesses to execute malicious code on the server. This helps in understanding both the attack process and the importance of secure implementation.

Existing System

In many traditional web applications, file upload functionality is implemented with minimal security checks. These systems often rely on basic validation techniques such as checking file extensions or using client-side validation methods. Such approaches are not sufficient to prevent malicious uploads.

Existing systems do not properly verify the content or type of uploaded files and may store them directly in accessible directories. This allows attackers to upload harmful scripts and execute them through the web browser. As a result, these systems are highly vulnerable to attacks such as Remote Command Execution (RCE), leading to unauthorized access and data compromise.

Disadvantages of Existing System

- Lack of proper server-side validation
- Over-reliance on client-side validation
- Easy bypass of file upload restrictions
- Vulnerability to malicious file uploads
- Risk of Remote Command Execution (RCE)
- Poor file type verification mechanisms
- Insecure storage of uploaded files
- Increased chances of data breaches
- Weak filtering and validation rule
- Lack of awareness in secure coding practices

Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026

Proposed System

The proposed system aims to demonstrate and analyze file upload vulnerabilities in a controlled environment using Damn Vulnerable Web Application. It provides a structured approach to understanding how attackers exploit insecure file upload mechanisms and how such vulnerabilities can lead to Remote Command Execution.

The system evaluates different security levels (low, medium, high) in DVWA to understand how security measures impact vulnerability. It also demonstrates attack techniques such as file extension manipulation, MIME type spoofing, and embedding malicious code. Additionally, the system highlights secure practices like strict server-side validation, restricting file types, and secure file storage to prevent such attacks.

Advantages of Proposed System

- Provides practical understanding of file upload vulnerabilities
- Demonstrates real-world attack techniques
- Helps in learning secure coding practices
- Improves awareness of web security risks
- Supports analysis across different security levels
- Identifies weaknesses in file handling mechanisms
- Encourages implementation of strong validation methods
- Helps prevent Remote Command Execution attacks
- Useful for students and cybersecurity learners

IV. Methodology

The methodology of this project involves analyzing and exploiting file upload vulnerabilities using Damn Vulnerable Web Application in a controlled environment. Initially, the file upload functionality of the web application is studied to understand how files are processed and validated. The process then involves identifying weaknesses such as insufficient server-side validation and reliance on client-side checks. Various attack techniques, including file extension manipulation, MIME type spoofing, and embedding malicious scripts, are applied to bypass security restrictions. Once a malicious file is successfully uploaded, it is executed through a web interface to demonstrate Remote Command Execution and its impact on the server. The results are analyzed to understand how vulnerabilities occur and how they can be mitigated. Finally, secure coding practices and preventive measures such as strict validation, file filtering, and secure storage are discussed to reduce risks.

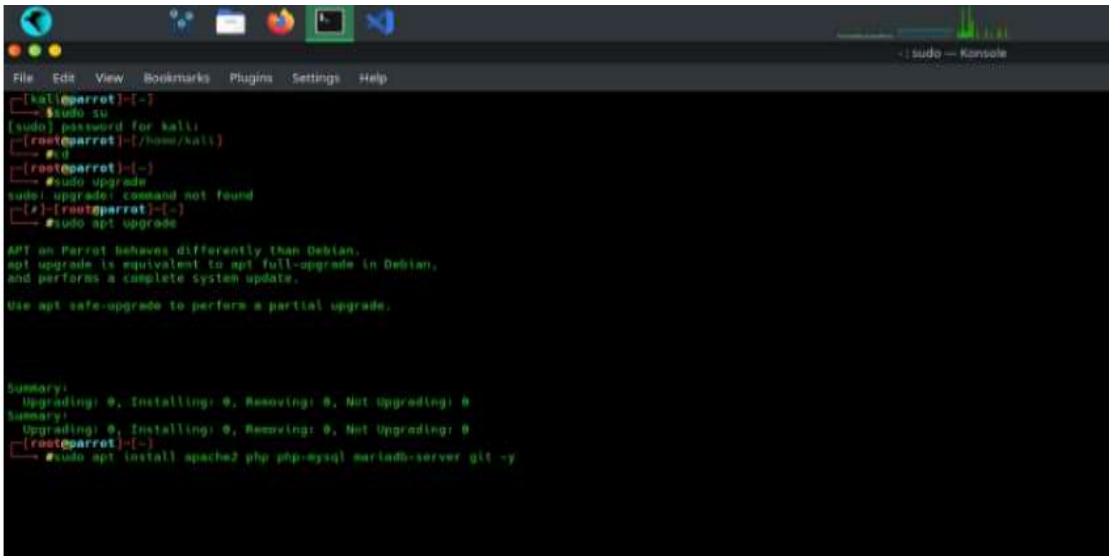
System Architecture

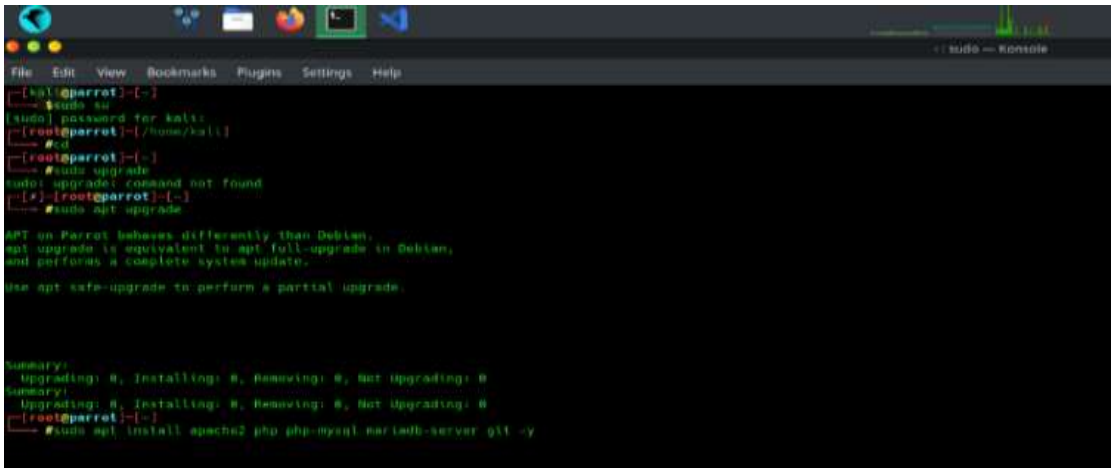
The system architecture for this project is designed to illustrate how file upload vulnerabilities lead to Remote Command Execution using Damn Vulnerable Web Application. The architecture consists of multiple components, including the user interface, web server, file upload module, storage system, and execution environment. The user interacts with the web application by uploading files through the interface. The file upload module processes the file and performs validation checks. In insecure systems, weak validation allows malicious files to pass through and be stored on the server. Once uploaded, these files can be accessed and executed via the web server,

Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026

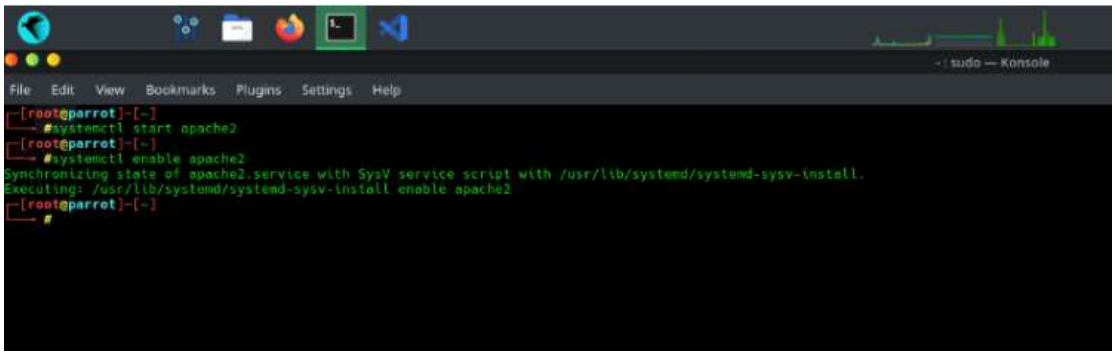
leading to unauthorized command execution. The system also includes an analysis component to study vulnerabilities and their impact. This architecture highlights the flow of data and demonstrates how improper validation can compromise the entire system, emphasizing the need for secure implementation.

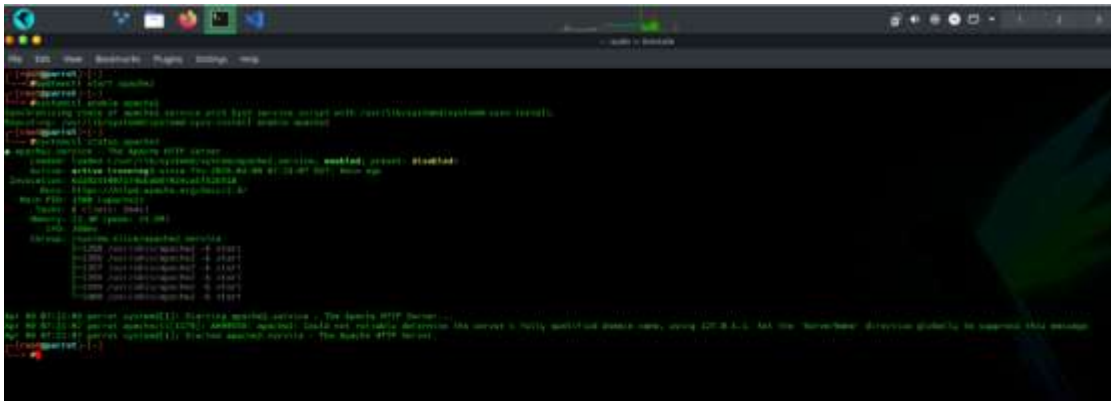
V. Result and Output

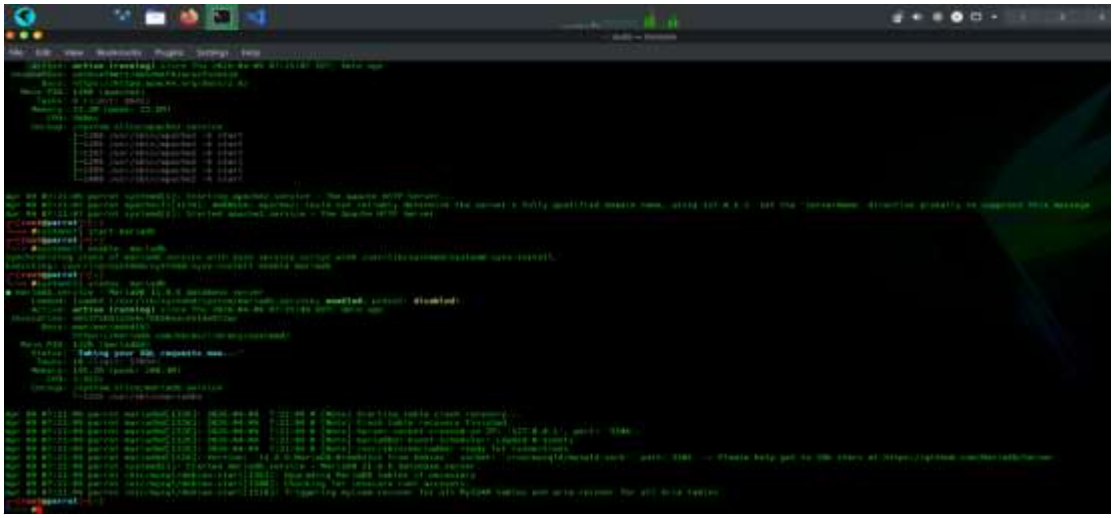


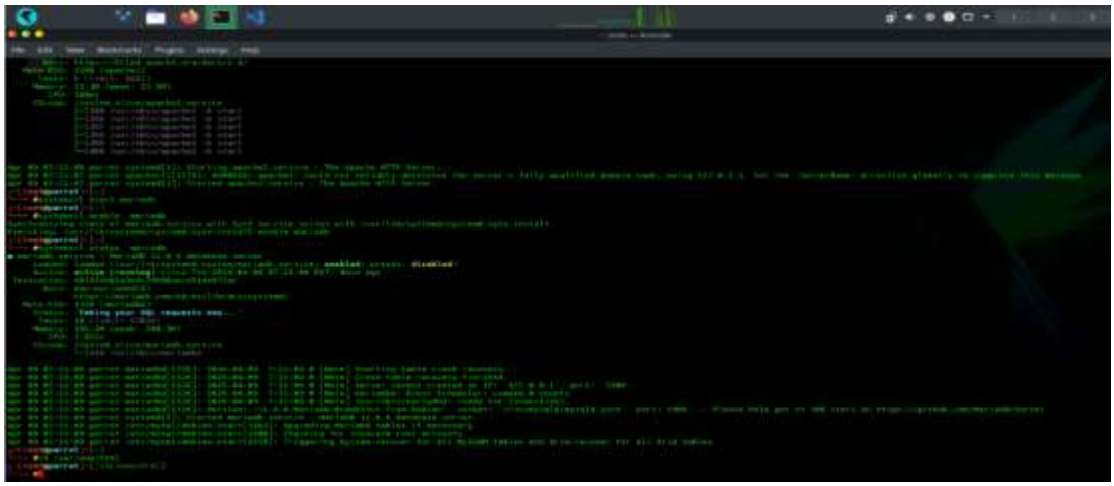


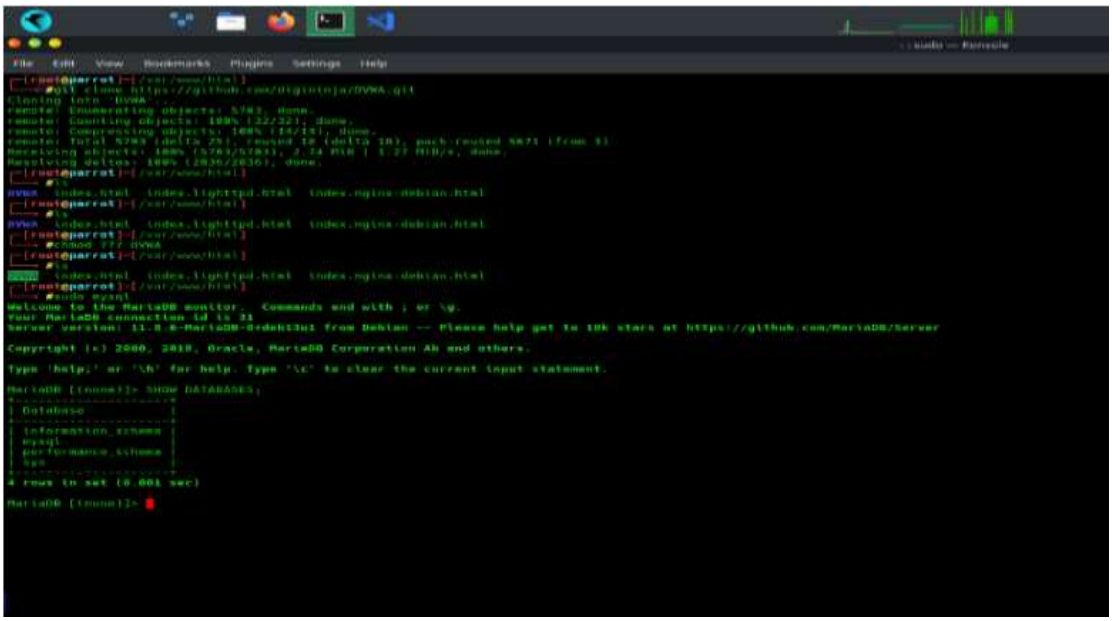
Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026

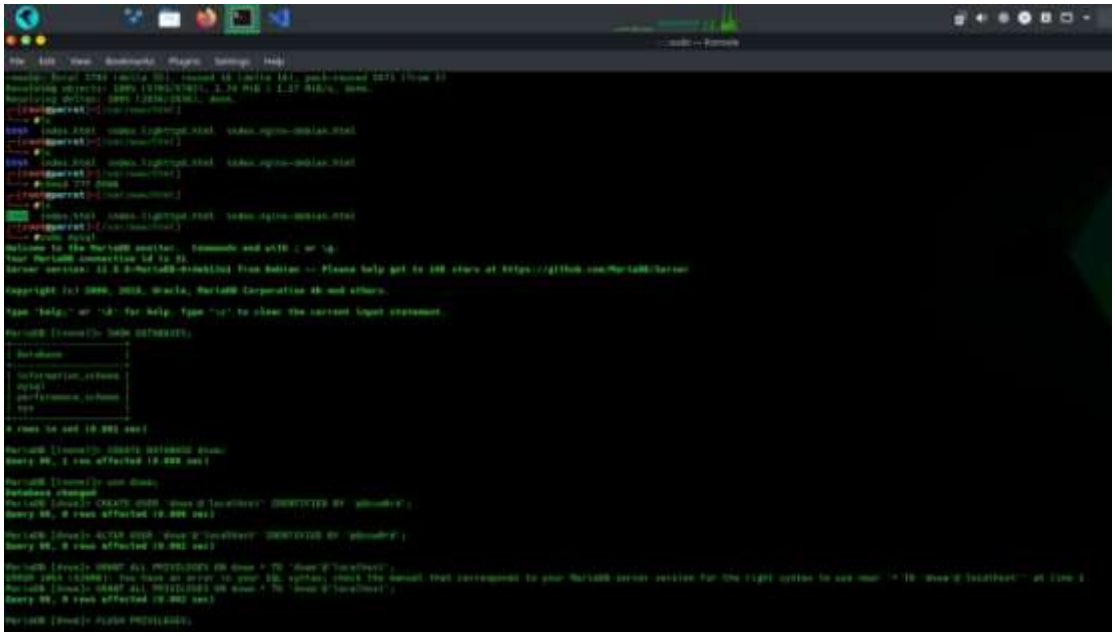

- : sudo — Konsole
File Edit View Bookmarks Plugins Settings Help
[root@parrot]-[~]
#systemctl start apache2
[root@parrot]-[~]
#systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
[root@parrot]-[~]
#

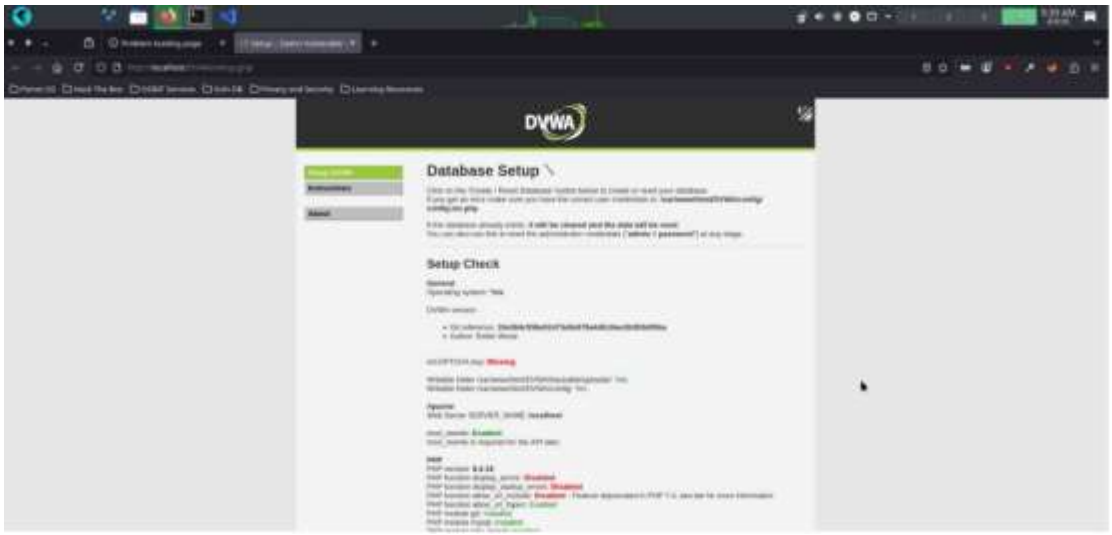






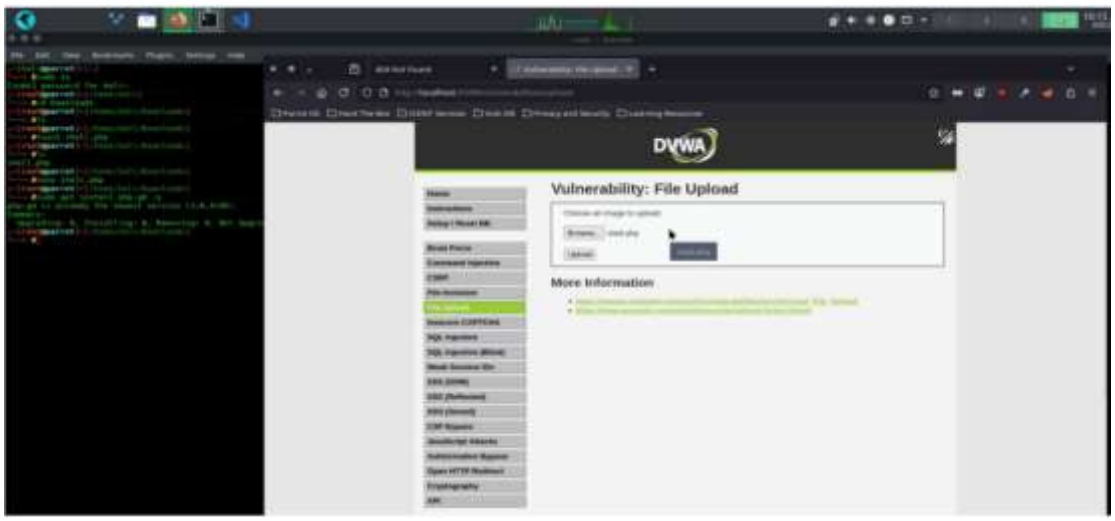


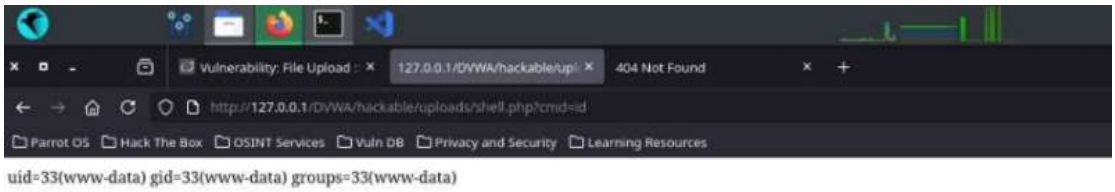



DVWA
Database Setup
Setup Check


DVWA
Welcome to Damn Vulnerable Web Application!
General Instructions
WARNING!


DVWA
DVWA Security
Security Level


DVWA
Vulnerability: File Upload
More Information



VI. Conclusion

The experiment on file upload vulnerabilities using Damn Vulnerable Web Application clearly demonstrates how insecure file handling can lead to serious security breaches in web applications. By exploiting weak validation mechanisms, a malicious file can be successfully uploaded and executed, resulting in Remote Command Execution (RCE) on the server. This highlights that even a simple file upload feature, if not properly secured, can become a critical entry point for attackers to gain unauthorized access and control over a system.

The study identifies that the primary causes of such vulnerabilities include inadequate server-side validation, improper file storage practices, and lack of execution restrictions. Accepting user-uploaded files without strict verification of file type and content allows attackers to bypass security controls. Additionally, storing files in publicly accessible directories and permitting direct execution significantly increases the risk of exploitation.

This project emphasizes the importance of secure coding practices and implementing multiple layers of security. Measures such as strict server-side validation, restricting executable permissions, renaming uploaded files, and storing them outside the web root are essential to prevent attacks. Regular security testing, code reviews, and awareness of common vulnerabilities further strengthen application security.

In conclusion, file upload vulnerabilities are critical threats that can lead to complete system compromise if not properly addressed. This experiment not only demonstrates

Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026

how such attacks are carried out but also reinforces the need for proactive and secure development practices. Ensuring proper validation and secure file management is crucial for building robust and resilient web applications capable of defending against real-world cyber threats.

References

[1] Kumar, R. D., Prudhviraj, G., Vijay, K., Kumar, P. S., & Plugmann, P. (2024). Exploring COVID-19 through intensive investigation with supervised machine learning algorithm. In Handbook of Artificial Intelligence and Wearables (pp. 145-158). CRC Press.

[2] Swathi, B., Vijay, K., Sushanth Babu, M., & Dinesh Kumar, R. (2024, November). Machine Learning Techniques in Cloud Based Intrusion Detection. In The International Conference on Artificial Intelligence and Smart Environment (pp. 557-564). Cham: Springer Nature Switzerland.

[3] Sv satyakrishna, shirisha rangu ,bhargavi nalacheruve.(2024) Prospective investigation on colorectal cancer with SMOTE on machine learning Algorithm

[4] Dr.G.Vishnu Murthy, BhargaviNalacheruve 1Professor, Department of computer Science & engineering, Anurag University, TS, India. 2Student, Department of computer Science & engineering, Anurag University, TS, India.

[5] V. N. S. Manaswini, K. K, C. Nigam, S. S. Ali, R. Niranjana, and Suman, “Real-Time Object Detection in Drone Surveillance Using YOLOv5,” in Proc. 2025 3rd Int. Conf. IoT, Communication and Automation Technology (ICICAT), Gorakhpur, India, 2025, pp. 1–6, doi: 10.1109/ICICAT68430.2025.11414670.

[6] B. Soundarya, V. N. S. Manaswini, M. Ayyakrishnan, R. D. Kumar, “Contextual Analysis of Big Data Analytics in Intelligent Transportation Frameworks,” in Intersection of Artificial Intelligence, Data Science, and Cutting-Edge Technologies: From Concepts to Applications in Smart Environment, Lecture Notes in Networks and Systems, vol. 1353, Cham: Springer, 2025, doi: 10.1007/978-3-031-88304-0_79.

[7] R. D. Kumar, V. N. S.Manaswini, “Applications of blockchain in smart cities: detecting fake documents from land records using blockchain technology,” in Blockchain for Smart Cities, Elsevier, 2021, pp. 105–117, doi: 10.1016/B978-0-12-824446-3.00017-X.

[8] Tejavath Veeramma, Badarla Anil, Guguloth Ravinder, “An advanced movie recommender using collaborative filtering and sentiment analysis,” International Research Journal of Modernization in Engineering Technology and Science, vol. 7, no. 7, July 2025, doi: 10.56726/IRJMETS81618.

[9] Ravi Kumar Banoth, Ramana Murthy B V, “Automatic crop recommendation system using LightGBM and decision tree machine learning models,” Journal of Machine and Computing, vol. 5, no. 1, pp. 343, Jan. 2025, doi: 10.53759/7669/jmc202505026.

Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026

[10] Ravi Kumar Banoth, Dr. B.V. Ramana Murthy, “Smart agriculture through IoT and machine learning for analyzing carbon footprints,” in Proc. Int. Conf. Computer Science and Communication Engineering (ICCSCE), Apr. 2025.

[11] Ravi Kumar Banoth, B. V. Ramana Murthy, “Soil image classification using transfer learning approach: MobileNetV2 with CNN,” SN Computer Science, vol. 5, art. no. 199, 2024, doi: 10.1007/s42979-023-02500-x.

Received: 28-04-2026 | Accepted: 02-06-2026 | Published: 11-06-2026