WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Defending Against Third-Party File Injection: Uncovering and Eliminating Security Vulnerabilities

S.S. Tamilarasu[1] ; V.Vaidehi[2]

sstamilarasan07@gmail.com ; vaidehi.mca@drmgrdu.ac.in

[1]. PG Student; [2.] Professor

Department of Computer Application,

Dr.MGR Educational and Research Institute Chennai-600095

ABSTRACT:

The security of web applications is seriously threatened by third-party file injection, which can result in sensitive data compromise, illegal access, and data breaches. This study's main goal is to create preventative measures and mitigation solutions to protect web applications from file injection vulnerabilities introduced by outside parties. A thorough examination of popular third-party file injection attack vectors, including those that take advantage of external resources, libraries, and dependencies, is part of the research process. To find potential vulnerabilities, a thorough analysis of input validation techniques, security procedures, and coding practices will be carried out. In order to improve the identification of injection places, the study will also investigate the application of static and dynamic code analysis methods. Building on the vulnerabilities found, the study will suggest and put into practice defense methods, such as the application of Content Security Policy (CSP) directives, code sanitization, and input validation. Via simulated attack scenarios and penetration testing, the efficacy of these defense systems will be assessed.

KEYWORDS: Unauthorized access, Injection attack vectors, Libraries, External resources, Coding practices, Input validation mechanisms, Security protocols, Static code analysis, Dynamic code analysis, Injection points, Simulated attack scenarios, Vulnerability detection.

I.INTRODUCTION :

A growing number of embedded devices, such as printers, routers, and webcams, are being connected to the Internet, and an adversary can attack those devices by taking advantage of known security holes. Security patches are typically linked to the firmware of the device, which is dependent upon the manufacturers and products of the device. Many embedded devices are still utilizing outdated firmware with known vulnerabilities or faults because of compatibility and release-time difficulties [1].

For network function deployment and communication facilities, network function virtualization (NFV) has shown promise in terms of availability, programmability, and flexibility. Since cloud technologies have advanced, there has been a trend of outsourcing network operations to a cloud service provider via virtualization in order to reduce the local load of managing and providing such hardware resources. Redirecting communication traffic to a third-party service provider, despite its potential, has raised issues around security and privacy [2].

The wireless air interface is open and available to both authorized and unauthorized users because radio propagation is broadcast in nature. In contrast, a wired network is one in which all communication devices are physically connected to one another via cables, making it impossible for a node not directly associated to access the network for unauthorized purposes. Because of the open communications environment, malicious assaults, such as active jamming to disrupt legitimate transmissions or passive eavesdropping for data interception, are more likely to affect wireless transmissions than cable communications [3].

Since block chains are decentralized systems, security is essential to their operation. But despite their growing acceptance and appeal, standardized models that analyze security risks associated to block chain technology are lacking. Our effort aims to close this gap by systematizing and expanding the body of knowledge on the security and privacy aspects of block chains, as well as by promoting uniformity in this field [4].

Deep learning is being used in many safety-critical contexts thanks to its notable results and quick advancement in a wide can weaken deep neural networks (DNNs). Adversarial perturbations are undetectable to humans, but during testing and deployment, they can readily trick DNNs. When implementing DNNs in situations where safety is paramount, one of the main dangers is the susceptibility to adversarial examples. As such, there is considerable interest in assaults and defenses against adversarial instances [5].

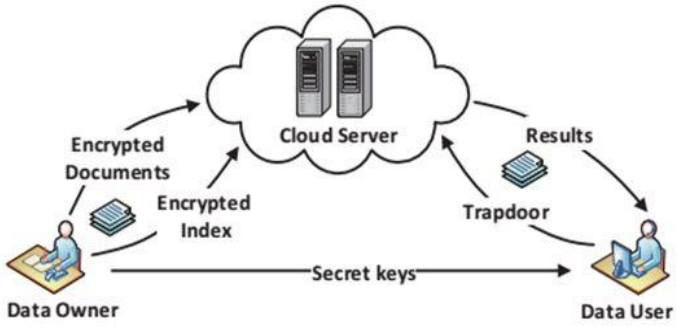
II. Literature Survey:

According to Maliheh Monshizadehet al.,(2015), Content that combines Flash and JavaScript has grown in popularity due to its ability to deliver dynamic features specific to each platform, which is useful for a wide range of web development projects. While the security of Flash and JavaScript

have been well studied, much less research has been done on the security of untrusted content that combines the two. In-depth discussion of this fusion is provided in this article, along with a number of real-world examples that put web application security at risk [6].

According to Muktar Yahuza et al., (2020), A promising paradigm that expands on cloud computing's potential is edge computing. Maintaining a positive environment free from security and privacy violations is crucial if you want to keep using the computer services. The edge computing environment's security and privacy concerns have limited the technology's general acceptance as a trustworthy paradigm. While many academics have examined privacy and security concerns in edge computing, not all have thoroughly examined the necessary security and privacy measures [7].

According to Rafiq Ahmad Khan et al., (2022) say that, Software quality is heavily dependent on security. In software development, integrity, confidentiality, and availability of code, data, and services are guaranteed by development processes. Security is an afterthought for software development companies, which means that security threats persist in their operations. The Software Development Life Cycle (SDLC) has made it imperative to incorporate security at every stage. When it comes to software security, many approaches, plans, and models have been devised and put



out, but very few of them provide solid proof for developing safe software [8].

According to Yonhap Xiao et al., (2019) say that, the development of edge computing has been greatly aided by the recent rapid growth of the Internet of Things (IOT) and smart mobile devices. Although edge computing has greatly aided lightweight devices in efficiently completing complex tasks, its rapid development has resulted in a significant disregard for security threats in edge computing platforms and the applications they enable. In this work, we present a thorough overview of the most common and impactful assaults along with the accompanying mitigation mechanisms that are tailored to edge computing and may be used in real-world edge computing systems [9].

III. Proposed System:

Web applications can have vulnerabilities, especially during the development and debugging stages. This can be found using a technique called vulnerability detection. The application of vulnerability scanners automates this process, or developers might carry it out manually. An auditor examines the source code by hand and/or makes genuine attempts to compromise the web application while using the manual technique. The auditor needs to be knowledgeable about the program architecture and source code in order to find vulnerabilities. Alternatively, they can try successful attacks specifically designed for their online application by becoming an expert in computer security. Long hours are needed for a thorough audit, and the auditor's expertise is the only factor that can determine its outcome. Moreover, errors and omissions can occur during human verification. Conversely, vulnerability scanners automate the process of finding vulnerabilities without requiring the auditor to be well-versed in the security aspects of web applications. The errors and omissions that are usually committed when detecting vulnerabilities manually are eliminated by automated vulnerability scanners. Websites can be static, dynamic, or sometimes a mix of the two. For websites to guarantee security, their databases must be protected. Interactive online applications that offer database services are vulnerable to SQL injection attacks. These programs use information from the user to generate a SQL query on the fly. An attacker may introduce a malicious SQL query as input to carry out an illegal database operation in a SQL injection attack.

An attacker can obtain or alter sensitive and confidential data from the database by using SQL.

ADVANTAGES OF PROPOSED SYSTEM:

- Enhanced File Upload Security
- User Authentication and Authorization
- Educated Develop Balanced Dynamic and Static Content Handling.

Fig: The System Architecture.

Existing System:

SQL injection is a technique maliciously used to obtain unrestricted access to databases by inserting maliciously crafted strings into SQL queries via a web application. It allows an attacker to spoof his identity, expose and tamper with existing data in databases, and control the database server with the privileges of its administrator. There is a variable SQL injection scanner, but it prolongs the connection time if it wants to detect,

prevent, or both. The popular solutions for the prevention of SQL injection attacks can't solve the problem of legacy systems, and the software developers are not easy to adapt. In this paper, we try to improve the execution time and find ways to improve the legacy system to be able to prevent injection attacks. Try to make our technique easy to adapt for software developers and make it more efficient in the future

IV. EXPLANATION

In this project, the main objectives are to implement URL and SQL Injection concepts. When a data provider uploads a file, it is shared with the receiver. The receiver must provide the correct OTP and file credentials to access the file. However, if a user attempts to access the file without proper credentials via URL or through a SQL attack, access to the file will be denied.

1.User [Sender]: In this phase, data's are uploaded and encrypted using AES algorithm whereas this kind of encoding is done to avoid unauthorized person to view the context of the data and only the authorized person can able to view through secret key which is provided to corresponding user. Block diagram of source phase is as shown in figure.1.

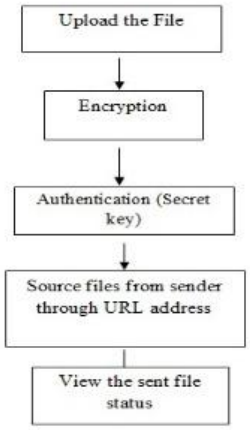


Figure.1 Source Phase

2.User [Receiver]:

In this phase, the source data is received in the destination end. In order to confirm whether the destination receiver is an authenticated person to view the data, the respective user has to provide the URL address and port number to receive the data. Block diagram of source end phase is as shown in figure.2.

- **Registration page:** A *registration form* is a list of fields that a user will input data into and submit to an individual. There are many reasons why you would want a person to fill out a *registration form*. Companies use *registration forms* to sign up customers for subscriptions, services, or other programs or plans.
- **Login page:** Logins are used by websites, computer applications, and mobile apps. They are a security measure designed to prevent unauthorized access to confidential data. When a login fails (i.e, the username and password combination does not match a user account), the user is disallowed access. Many systems block users from even trying to log in after multiple failed login attempts.
- **Provider:** Data controller used to control and monitor all users. Data controller needs to login with username and password. View request from the data provider and then controller will send key to user. Generate graph based on the overall user requests and generate graph based on the all the uploaded files.

V. Result & Discussion:

Admin:

Admin Login: This refers to the process of accessing an administrative account within a system or platform. Admin login grants elevated privileges and access to administrative features that regular users may not have. Admins typically use specific credentials (username/email and password) designated for administrative access.

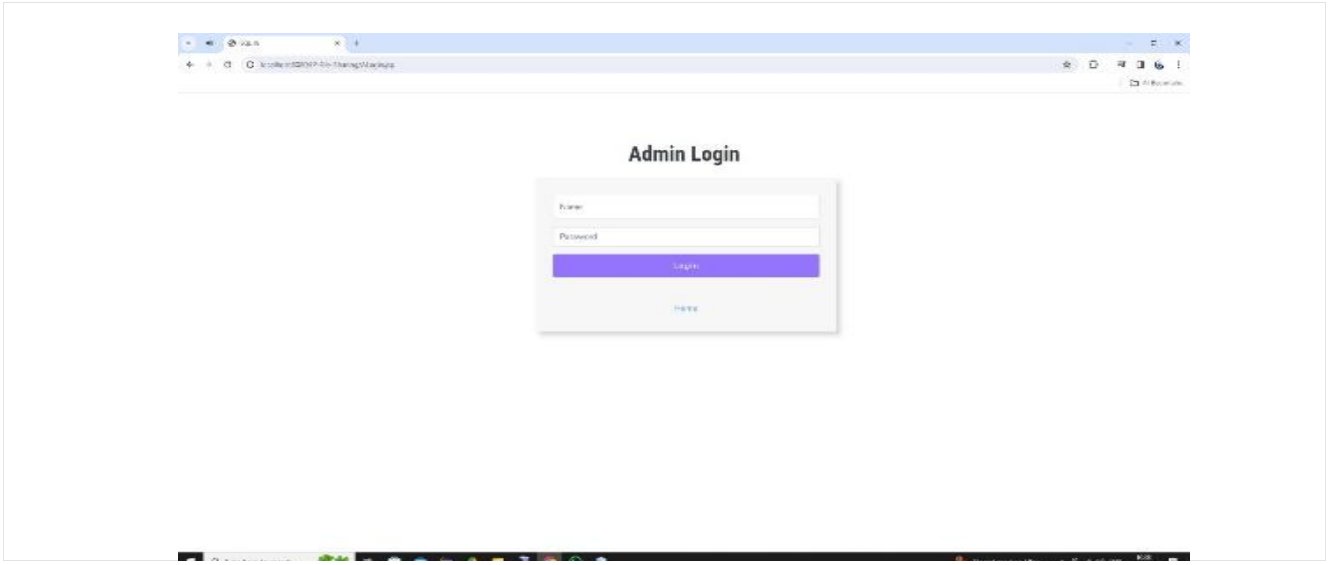


Fig 2: Admin Login Page.

View All User Status: After logging in as an admin, you have the capability to view the status of all users within the system. This includes information such as user accounts' activity status (active, inactive), login history, last login timestamp, account permissions, and any status flags related to user accounts (e.g., locked, disabled).

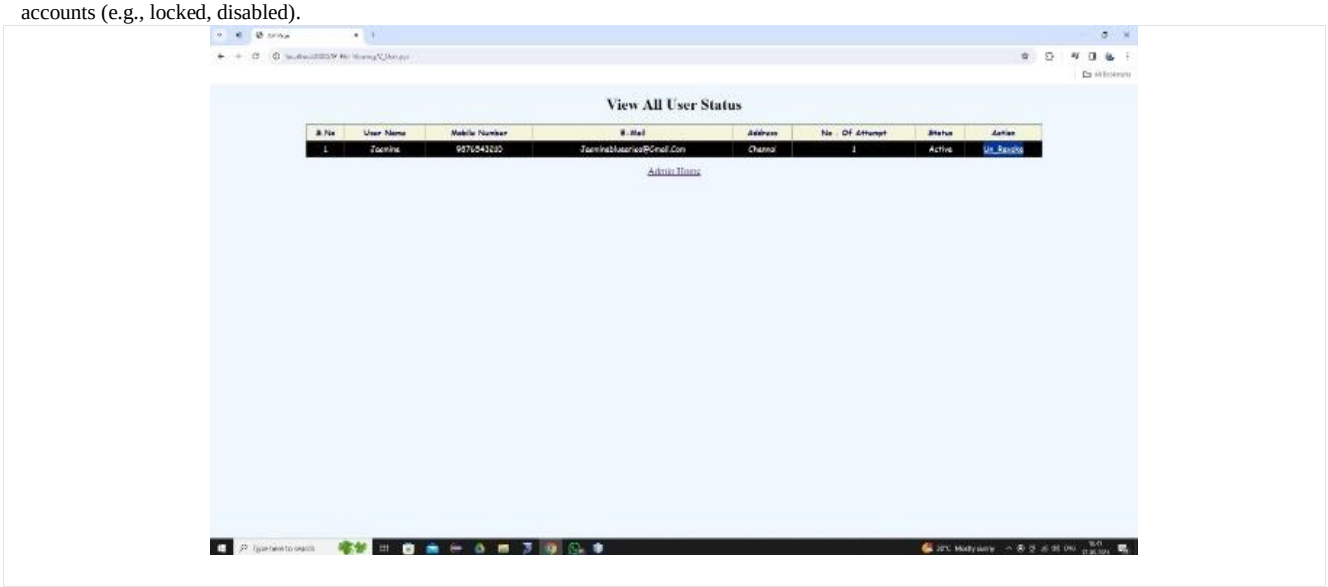


Fig 3: View All User Status.

View All Users' Uploaded Files: As an admin, you can access and view all the files uploaded by users within the system. This includes documents, images, videos, or any other types of files uploaded by users. Admins can typically browse through the files, view file details (such as size, type, upload date), and manage files as necessary (e.g., download, delete, move).

Admin login grants access to administrative functions and controls within the system, allowing admins to perform tasks such as viewing all user statuses and accessing all users' uploaded files. This level of access is crucial for system management, monitoring user activities, ensuring compliance, and handling administrative tasks efficiently. Admins play a vital role in maintaining the integrity, security, and functionality of the system while overseeing user activities and uploaded content

User:

User Registration: Allows users to create new accounts on the platform, providing essential information for identification and access.
User Login: Grants access to registered users, verifying their identity and enabling them to utilize the platform's features based on their permissions

User File Uploading: Enables users to upload files to the platform, facilitating data sharing, collaboration, and storage of user-generated content

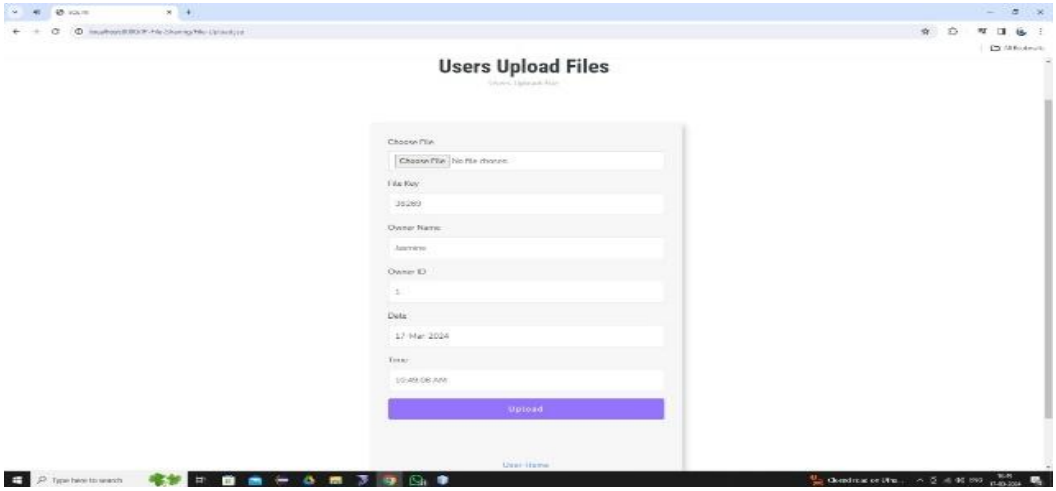


Fig 4: User Upload Files.

User OTB Verification: It seems like you're referring to "OTB verification" in the context of user verification or authentication. However, the term "OTB" doesn't have a universally recognized meaning in this context. Could you please clarify what "OTB" stands for or provide more context about the verification process you're asking about? This will help me provide a more accurate explanation or guidance.

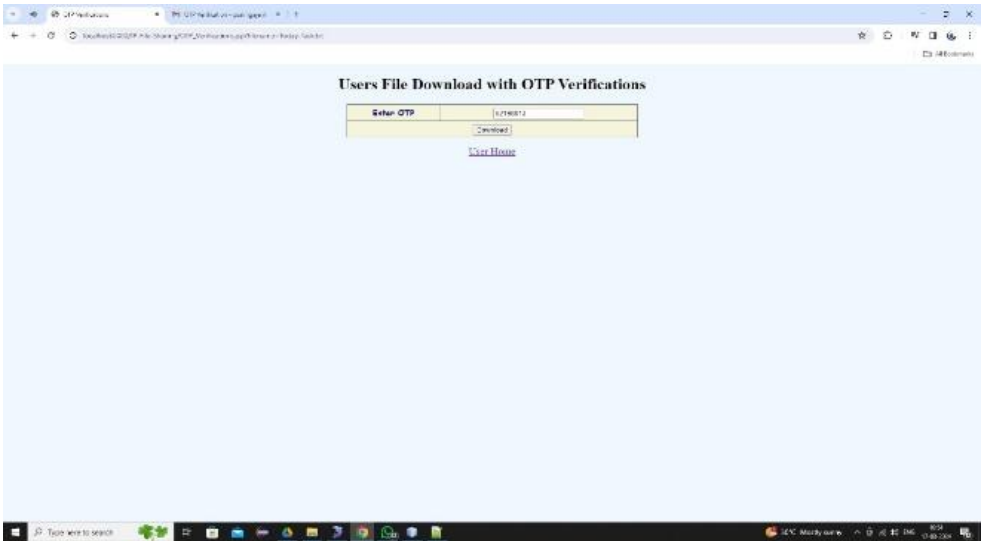


Fig 5: User OTB Verification.

SQL injection: is a technique maliciously used to obtain unrestricted access to databases by inserting maliciously crafted strings into SQL queries via a web application. It allows an attacker to spoof his identity, expose and tamper with existing data in databases, and control the database server with the privileges of its administrator. There is a variable SQL injection scanner, but it prolongs the connection time if it wants to detect, prevent, or both. The popular solutions for the prevention of SQL injection attacks can't solve the problem of legacy systems, and the software developers are not easy to adapt. In this paper, we try to improve the execution time and find ways to improve the legacy system to be able to prevent injection attacks. Try to make our technique easy to adapt for software developers and make it more efficient in the future.

Fig 6: SQL injection page.

VI.CONCLUSION:

We have presented an effective user-defined access control safe deduplication mechanism in this research. In particular, our technique achieves the permitted deduplication without requiring the usage of hybrid cloud architecture or the deployment of an extra authorized server. Only the CSP is able to oversee access privileges on behalf of data owners in our scheme without jeopardizing the privacy of the data. Additionally, our plan adds the Bloom filter to effectively finish the dual-category check. Thorough security evaluations show that our plan is capable of simultaneously achieving tag consistency, data secrecy, access control, and resistance against brute-force attacks. Our approach is efficient in terms of deduplication efficacy, computational cost, communication overhead, and storage cost, as demonstrated by comprehensive performance assessments on file-level and chunk-level deduplication. Time and efficiency measures are used to evaluate the performance progression of the proposed AES and the current DES. The results show that the AES algorithm performs better overall, in terms of robustness and time consumption, than the DES.The file at the nodes does not crash during the full security code encryption and decryption operation. As a result, until the 4-digit security key is matched and validated, the data stays safe and unaltered.

FUTURE ENHANCEMENTS:

It shows the original IP address. Sender and Receiver does not communicate through transmission medium. Instead the secret is generated in the form of OTP. Filename should be hidden to avoid intruder from hacking.

VII. REFERENCES:

1. Qiang Li & Dawei Tan & Xin Ge & Haining Wang & Zhi Li & Jiqiang Liu(2022) "Understanding Security Risks of Embedded Devices Through Fine-Grained Firmware Fingerprinting" Volume 19.
2. Peipei Jiang & Qian Wang & Muqi Huang & Cong Wang & Qi Li & Chao Shen & Kui Ren (2021) "Building In-the-Cloud Network Functions: Security and Privacy Challenges" Volume 109.
3. Yulong Zou & Jia Zhu & Xianbin Wang & Lajos Hanzo (2016) "A Survey on Wireless Security: Technical Challenges, Recent Advances, and Future Trends" Volume 104.
4. Ivan Homoliak & Sarad Venugopalan & Daniël Reijsbergen & Qingze Hum & Richard Schumi & Pawel Szalachowski (2021) "The Security Reference Architecture for Block chains: Toward a Standardized Model for Studying Vulnerabilities, Threats, and Defenses" Volume 23.
5. Xiaoyong Yuan & Pan He & Qile Zhu & Xiaolin Li (2019) "Adversarial Examples: Attacks and Defenses for Deep Learning" Volume 30.
6. Phu H. Phung & Maliheh Monshizadeh & Meera Sridhar & Kevin W. Hamlen & V.N. "Venkat" Venkatakrishnan (2015) "Between Worlds: Securing Mixed JavaScript/ActionScript Multi-Party Web Content" Volume 12.
7. Muktar Yahuza & Mohd Yamani Idna Bin Idris & Ainuddin Wahid Bin Abdul Wahab & Anthony T. S. Ho & Suleman Khan & Siti Nurmaya Binti Musa & Azni Zarina Binti Taha (2020) "Systematic Review on Security and Privacy Requirements in Edge Computing: State of the Art and Future Research Opportunities" Volume 8.
8. Rafiq Ahmad Khan & Siffat Ullah Khan & Habib Ullah Khan & Muhammad Ilyas (2022) "Systematic Literature Review on Security Risks and its Practices in Secure Software Development" Volume 10.
9. Yinhao Xiao & Yizhen Jia & Chunchi Liu & Xiuzhen Cheng & Jiguo Yu & Weifeng Lv (2019) "Edge Computing Security: State of the Art and Challenges" Volume 107.
10. María B. Jiménez & David Fernández &Jorge Eduardo Rivadeneira & Luis Bellido & Andrés Cárdenas "A Survey of the Main Security Issues and Solutions for the SDN Architecture" Volume 9.