ICRRD Journal
INTERNATIONAL CENTER FOR RESEARCH AND RESOURCES DEVELOPMENT article
ISSN Number: 2773-5958, https://doi.org/10.53272/icrrd, www.icrrd.com

Security Analysis of CMS based Websites through CMSPY

Siddharth Bose[1*], Ambili Kakkad Narayanan[2]

[12] *TIFAC-CORE in Cyber Security Amrita School of Computing, Coimbatore Amrita Vishwa Vidyapeetham, India*

**Corresponding author; Email: siddharthbs067@gmail.com*

Received: 10 March 2023 **Accepted:** 12 June 2023
Revision: 20 May 2023 **Published:** 07 December 2023. **Volume**-4, **Issue**-4

Cite as: Bose S., Narayanan AK., (2023). Security Analysis of CMS based Websites through CMSPY *ICRRD Journal, 4*(4), 162-174.

ABSTRACT*:* This paper presents a method of security analysis of websites around 3 content management systems (CMS), namely WordPress, Joomla, and Drupal. The analysis aims to provide key inputs into CMS configuration reconnaissance of websites and assess the overall security posture of CMS-based websites depending on various criteria, including the version of the CMS, the presence and use of security plugins or extensions, the complexity of website customization, and frequency of software updates applied. The aim of this intel and analysis is to bridge the gap between CMS users and security by providing them with a sense of understanding of the security implications around the way their CMS is implemented. Additionally, the solution also helps in mitigating these security risks and provides an insight on what kind of exploits can harm the website and how. Further research could focus on developing more advanced security plugins or extensions for CMS-based sites, identifying the most common types of vulnerabilities across different versions and configurations of CMS-based websites, and exploring the efficacy of various cybersecurity measures for addressing CMS-related security risks. This paper contributes to the growing body of the latest research and literature on CMS security and provides a command line-based tool as a solution for securing web CMS. Overall, this security analysis can aid in improving the overall security posture of CMS-based websites and reducing instances of cyberattacks targeting these platforms.

Keywords: *Web Application Security, Content Management Systems, WordPress, Drupal, Joomla, Reconnaissance, ExploitDB, Vulnerability Assessment.*

Introduction

A software content management system designed especially for managing web content is called a web content management system. It offers website authoring, collaboration, and administration tools that let users build and manage web- site material even if they have little to no experience with web programming or markup languages. These systems offer an intuitive user interface and neces- sitate little to no coding expertise, making website administration available to those outside of the web development industry. However, the ease of creating and managing a website with CMS comes

at the cost of security vulnerabil-ities. [1] To mitigate these risks and bolster security measures, it is essential to conduct a thorough CMS security analysis on a regular basis. This analysis should be conducted by qualified professionals who have a deep understanding of the underlying technology and potential vulnerabilities associated with CMS platforms. Such analysis typically involves a comprehensive review of the CMS platform, the kind of default settings that are implemented, directories currently hosted and present on the site along with what kind of theme, version, plugins, components, modules, and templates have been used in order to search for any vulnerabilities that could play a role based on the reconnaissance results of the proposed tool ”CMSPY”. Additionally, CMS security analysis should include regular monitoring of the site for any suspicious activity or unauthorized access attempts. Overall, conducting a CMS security analysis is a critical step towards maintaining the integrity and confidentiality of website content, protecting user data from potential breaches, and ensuring the secure functioning and availability of the website for its users. Furthermore, regular security analyses are recommended as new vulnerabilities continue to be discovered in CMS platforms and cyber threats constantly evolve. [2] Website owners or operators who use CMS platforms should stay informed about security updates and vulnerabilities associated with their system, as well as keep in mind the importance of regularly conducting CMS security analyses to ensure that their website remains secure and protected against potential cyber-attacks. Thus this study is focused on important intel analysis of best practices and proper implementation of web-based CMS on websites through a command line tool ”CMSPY” as a solution which will provide reconnaissance information of the configuration of the CMS being used as well as perform relevant Vulnerability Assessment through ExploitDB.

Literature Review

To begin with, CMS showed up in the late 1990s, however, it turned out to be well-known in center 2000s There are more than 200 CMS created in different programming dialects, for example, PHP, Java, Perl, .NET, and others, isolated in open source and exclusive CMS, everyone with its bolstered databases like eg. MySQL, SQLite, Oracle, PostgreSQL. Out of all CMS, WordPress is the most widely used with more than 60 percent market share while Drupal and Joomla come 2nd and 3rd with 5 and 8 percent each. This exclusive suite of an all-in-one web technology system made it easy for users with very little or no technical knowledge to create, manage and maintain dynamic and feature-friendly interactive websites. But the entire dependency being solely on a CMS of a website can be dangerous, creating loopholes and vulnerabilities through the internal and external resources used by the CMS such as modules, plugins, themes, templates, etc. These resources are said to be the top-most source of attacks and exploitation of all CMS-based websites, especially in the case of WordPress, Drupal, and Joomla. The 2nd source of vulnerabilities arrives through default configurations and faulty setup of the CMS which leaves information or access to the CMS almost open to attackers. It is said that at any given point 18-20 percent of CMS-related attacks and issues remain unpatched and open to exploitation for the main reason being the users and admins of the sites are unaware of the issue the entire time. Due to this scenario, solutions have been brought up in the past in the form of tools that perform the needed security analysis and vulnerability assessment, the famous ones being WPScan for WordPress, Drupwn for Drupal, and Joomscan for Joomla. Similarly, more scanning tools across platforms have emerged in the recent past but with their specific

functionality and restricted limitations, particularly catering to just one CMS primarily. This is where a common open- source web CMS scanning tool is still the need of the hour. Web CMS Security has seen a surge of vulnerabilities and CVEs being reported in the past few years, primarily attacks like Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF) and Injection attacks like SQLi, code injections are the most found issues which are mainly discovered through faults within the plugins or modules being used in the CMS. 92 percent of vulnerabilities in the case of WordPress are observed as plugin-based issues since there are over 5000 plus currently vulnerable plugins solely in WordPress, namely the most recent ones including Drupal are ”Stored cross-site scripting in Link Juice Keeper Plugin for WordPress” (02/05/2023)[3] and ”Security restrictions bypass in Drupal - Improper Access Control” (19/04/2023) [4] The Fig 1. fetched from a security blog report further shows how more than half of the CMS based websites still run on the outdated versions thus opening doors for vulnerabilities and threats still actively being exploited and which can be fixed by updating the versions but unfortunately remain present due to unawareness. [5]

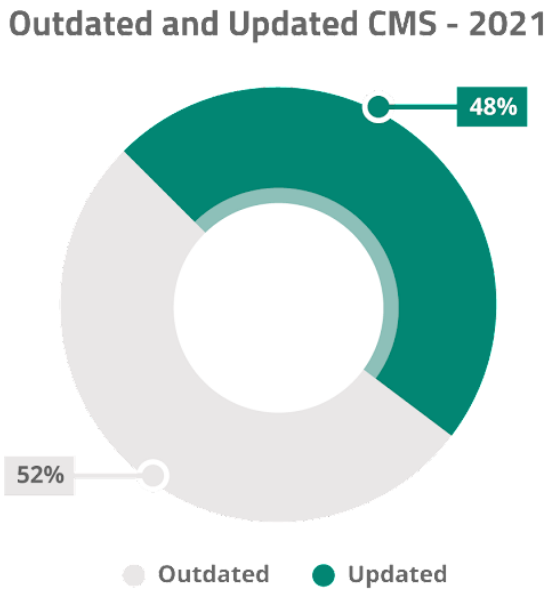

Fig. 1. 52 percent of sites run on an outdated CMS

Proposed Work

I have proposed a python-based command line tool which first accepts user input URL and performs analysis on it via parsing an argument based on 6 features of the tool provided as options:

-u for target URL that needs to be tested.
-b for brute-forcing directories with a user-provided list as input.
-x for checking if XML-RPC services are enabled or not and if the targetsite is vulnerable to pingback DOS and/or Login Brute Force attacks.
-e for enumerating admin usernames of the target URL if available.

-wpp for enumerating all plugins along with their version.
-edb for performing ExploitDB search for the url and prints out relevantEDB-IDs.

On receiving the target from the user, The URL is pinged for availability,and once established that the target is valid and online, we begin with CMS Identification. This takes place first in the case of WordPress, where we check for 4 standard Word- Press indicators in the target as factors to judge whether the target runs on WordPress or not. Similarly, the same is carried forward for Drupal for 3 specific factors and then Joomla! for a single factor (admin page) ina similar manner by pinging the target using the request.get() function to search and check for specific indications known for those particular CMS. In addition to the type of CMS present, its current running version is also checked for via Regex and fetched which would provide a better pinpointed assessment. Once the CMSis successfully identified, (if it is out of those three CMS) then we can display appropriate results accordingly. The Directory fuzzing option requests a path list where custom-made options are provided for all 3 web cms and users can createtheir own path list too for their specific CMS-related directory fuzzing to check for sensitive or interesting webpages or files that may be present on the target based on the type of CMS being used. The next function is a sensitive PHPfile vulnerability check for XML-RPC. [6] Initially, the presence and activity of XML-RPC is checked by sending an XML-based request to verify that it exists on target and that it is enabled. If that is the case then 2 vulnerability checks are done by parsing their XML-based requests to check for any specificresponse that is received through regular expression. [7] If the target responds in amanner that indicates the issue of either Pingback DOS Vulnerability or Login Brute Force Attack, then accordingly the same would be printed out thus successfully performing a vulnerability assessment and identification.

Next, the username enumeration is planned in various ways which include searching for the author parameter via url redirection, then for WordPress we have a dedicated /wp-json/wp/v2/users/ path [8], and lastly crawling over jet- pack public api. All these methods store usernames of admins in plaintext and if enumerated might be considered as sensitive data which should be removed from the target. The plugin enumeration feature would again make use of Regular expression [9] and crawls over wp-content/plugins/ path of the target to identifyall plugins being used as well as fetch the version of the plugin.

All this reconnaissance information is used as vital input to enable the main final feature of CMSPY which is ExploitDB static analysis. [10] The CMS type, version, plugins, modules, templates, themes, and their respective versions, all this data will be used to identify any exploits or threats corresponding which maybe present in ExploitDB which is the largest and rapidly growing crowd sourced Exploit database which contains thousands of working exploits and attack scriptsthat can be used against the target if any of the reconnaissance data matches with exploits present in the database. [11] Plan is to use Searchsploit, which is an exploit searching module that will crawl through the entire database and tryto look for a match for the data that has been received from the target site. Searchsploit will work locally and access the database via /usr/share/exploitdb/ path which contains the entire database. If a match for an exploit is found thenit will collect it and print it out along with its EDB-ID which is a unique identification number given to each exploit

present in the database. This feature,if successfully implemented will provide an effective means of static VulnerabilityAssessment of the target site thus helping in identifying potential issues and threats by verifying the exploit further through thorough Penetration Testing.

3.1 Algorithm applied for the script:

Import requests, re, sockets, argparse, os, sys.
Set parser = argparse.ArgumentParser() function:
add argument for the URL.
add argument for directory fuzzing
add argument for XML-RPC assessment
add argument for username enumeration
add argument for plugin enumeration
add argument for ExploitDB search for EDB-IDs
Define and assign all necessary variables and their initial values
Take input of the URL from the user.
Using sockets, sort the URL into the desired format.
Get a response from the URL to check if it's available via:requests.get() (main function)

WORDPRESS:

use request.get() for /wp-login, (looking for user-login)
use request.get() for /wp-admin,
use request.get() for wp-json/wp/v2/
use request.get() for /robots.txt (looking for 'wp-admin')

DRUPAL:

use request.get() for /readme.txt (looking for 'drupal')
use request.get() for Generator (looking for 'drupal')
use request.get() for /modules/README.txt (looking for 'drupal')

JOOMLA:

same request.get() for /administrator/ (looking for mod-login-username)
Display result according to findings
Provide the target overview as well such as HTTP Status, Server and IP.
Display HTTP header response via req.headers (may contain other CMSname)

Directory Fuzzing:

open the wordlist file and read the contents storing it in a variable
(f=open(wordlist, 'r')
run a for loop using request.get() adding / to the end of the URL andevery word from the user-provided path list after that.
if status code=200, print SUCCESS followed by the URL path
discard all other status code results

XML-RPC check:

First, check and validate the presence of XML-RPC on target by sendinga test XML code and analyzing the response via regex.

If XML-RPC is present, then check for a pingback DOS attack by sendingpingback.ping payload XML and analyzing response via Regex.

Then check for Login Brute Force Attack by sending wp.GetUsersBlogsto check for username admin and analyze response via Regex.

Username Enumeration:

Find the author parameter and look for username entries.

Crawl over /wp-json/wp/v2/users/ path and identify usernames listed

Try accessing the jetpack public api and look for the same then print allresults.

Plugin Enumeration:

Crawl over /wp-content/plugins to identify and fetch all plugins used.

Also use Regular Expression to identify plugin versions and print both.

ExploitDB Analysis:

Collect all the reconnaissance data of the target such as CMS type, ver-sion, plugins, themes, templates, modules etc.

Use Searchsploit to find for exploits in ExploitDB via /usr/share/exploitdb/

Collect all matches for the data pointing to exploits in ExploitDB andprint then out in a format including the EDB-IDs.

Results

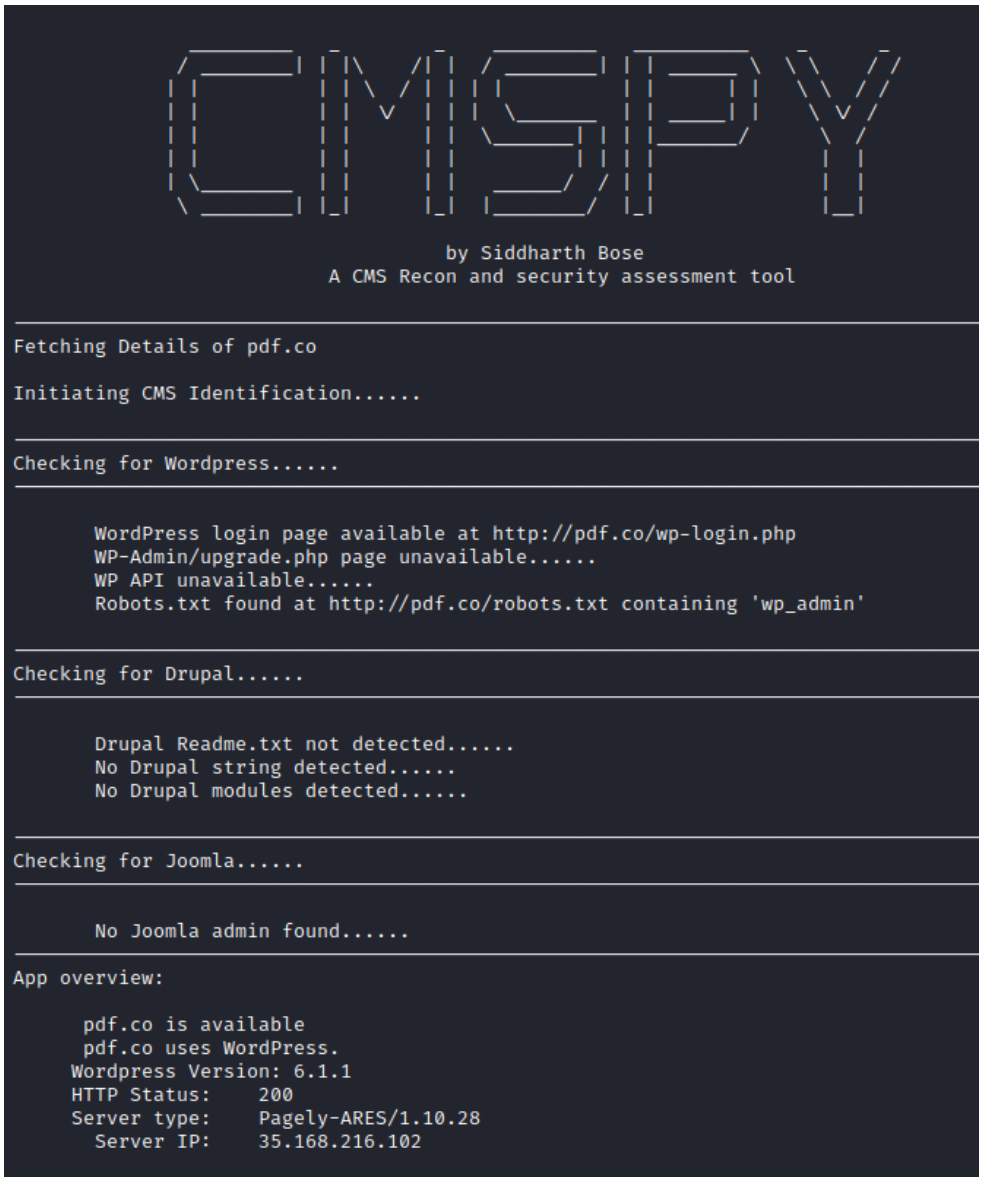

Fig. 2. CMS Detection Example

As observed in Fig 2., in this use case, target site being " pdf.co"checks two out of the four factors responsible for WordPress detection, hence concluded that it is running on WordPress cms. Thus, the primary functions of detecting the cms as well as its version work perfectly showing the results in theApp overview section

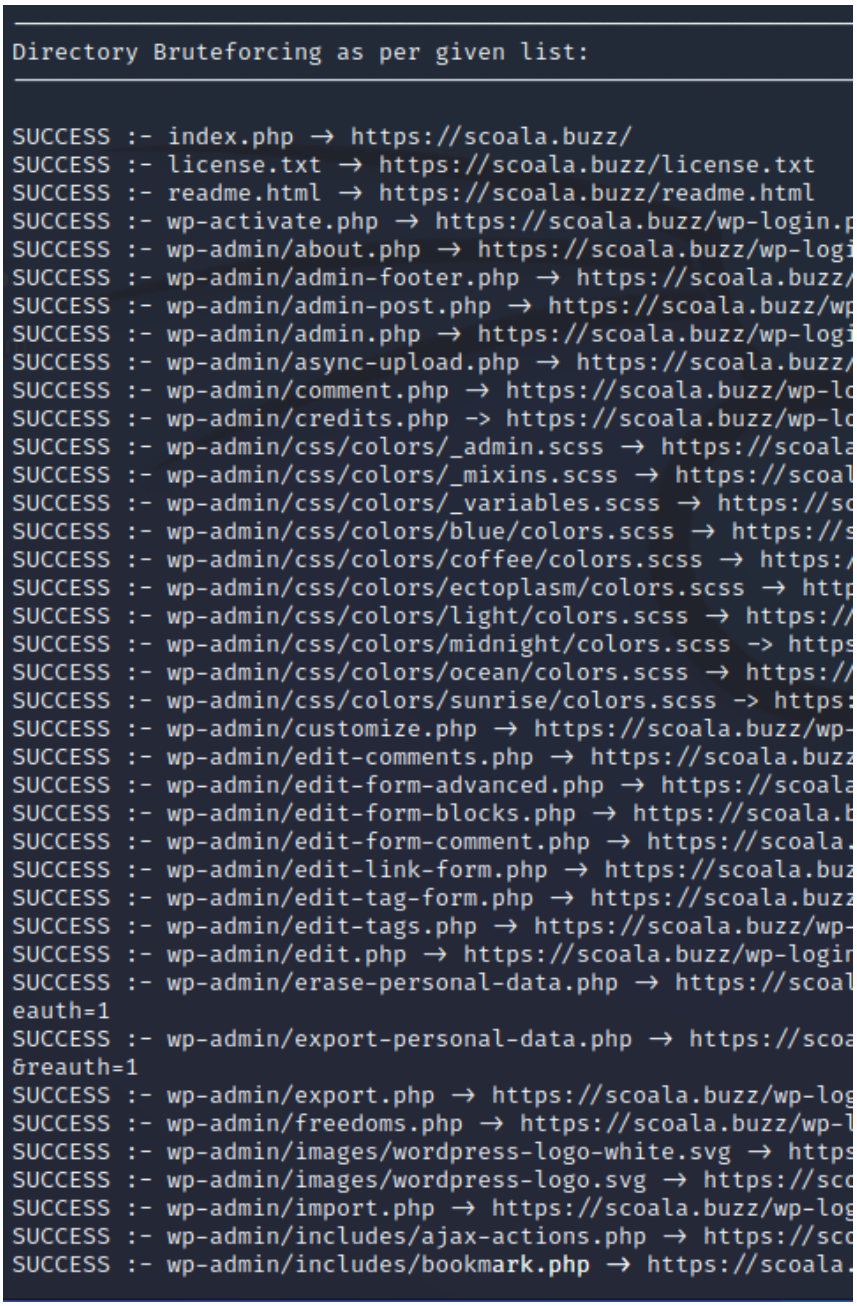

Fig. 3. Directory Fuzzing output

In this example of Directory Fuzzing, ”scoala.buzz” was a WordPress sitewhere as seen the WordPress directory list provided was able to fetch many interesting webpages in a precise manner.

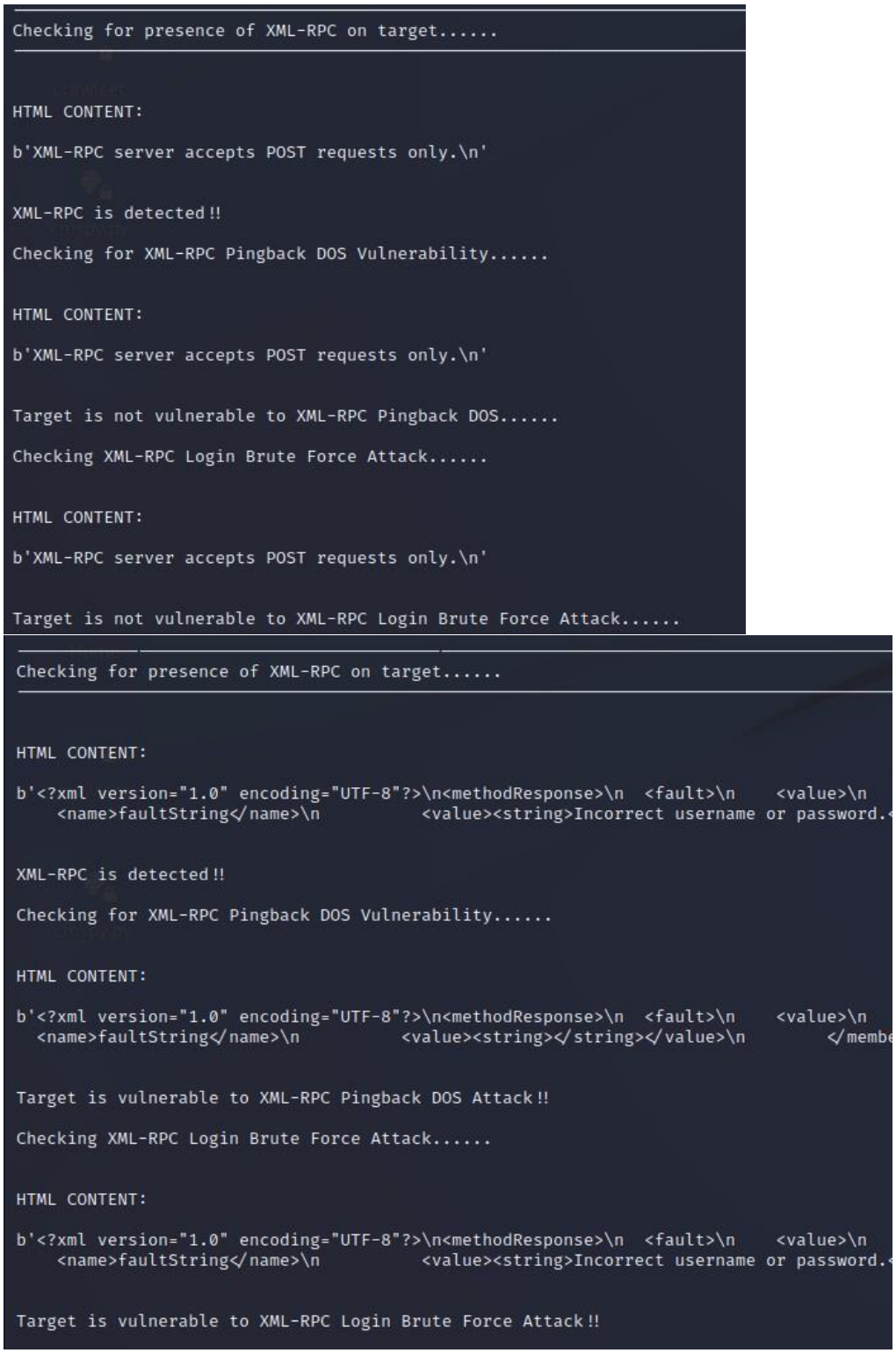

Fig. 4. XML-RPC output test cases

Here we intentionally test 2 use cases of XML-RPC vulnerable and non- vulnerable sites as proof of concept of both our vulnerability checks being matched by analyzing the response.

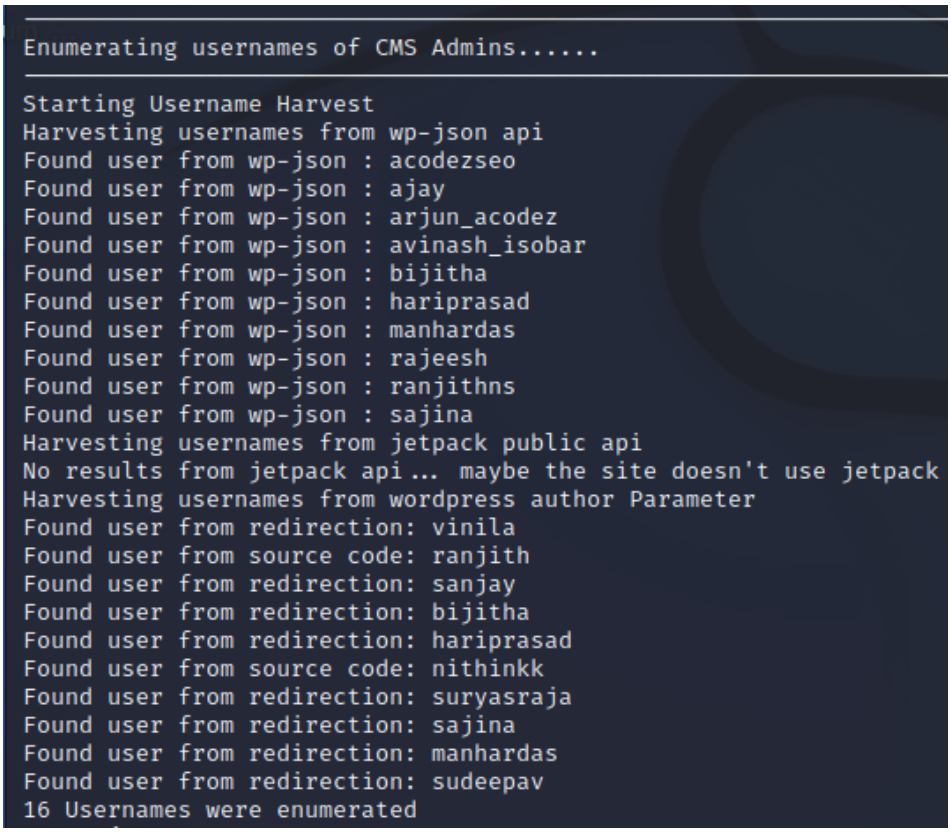

Fig. 5. Successful Username Enumeration

As shown in this example for the site ”amrita.edu” 2 out of 3 factors namely ”wp-json api” and author parameter yielded a total of 16 usernames associated with the cms. Such kind of data could be used in brute force attacks or user authentication bypasses.

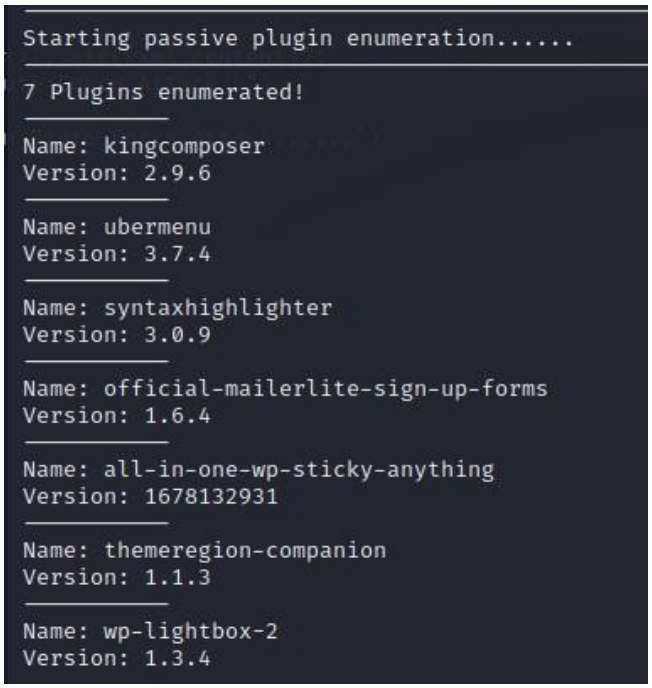

Fig. 6. Plugin Enumeration Example

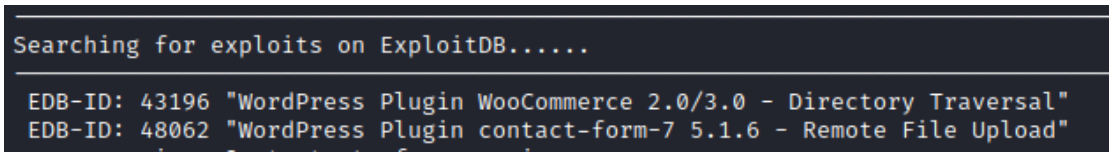

Fig. 7. Output for WordPress site ExploitDB scan

This is the example for a WordPress site where as shown for the target site, Search- sploit was able to find and detect 2 exploits based on plugins as mentioned which would lead to Directory Traversal and Remote File Upload possibilities on the site.

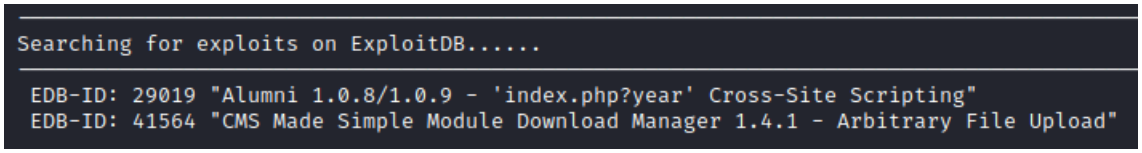

Fig. 8. Output for Drupal site ExploitDB scan

Referring to the use case of Drupal, on successfully fetching the modules and templates and feeding them to Searchsploit, 2 exploits with their EDB-IDs were enumerated leading to a particular Cross-Site Scripting attack and Arbitrary File Upload vulnerability on the target site.

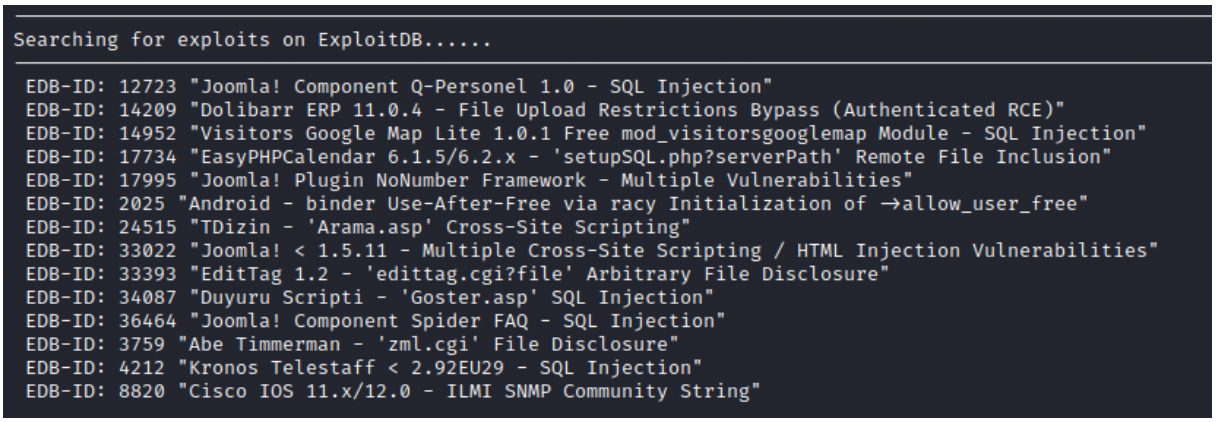

Fig. 9. Output for Joomla site ExploitDB scan

Final case of a Joomla cms based target site yielded 14 exploits via Searchsploit ranging from various critical attacks specifically targeting Joomla artifacts suchas SQL Injection, Cross-Site Scripting, Remote File Inclusion and Sensitive File Disclosure.

If these outcomes are known to hackers or adversaries then it will be a piece of cakefor them to use the exploits by vising ExploitDB and searching for them via their EDB-IDs and just using the script and following the steps as mentioned to initiate the corresponding attacks, which would result in severe direct business impact of the website and the organization.

Conclusion

Based on an overall understanding we can conclude that open-source web CMS (WordPress, Joomla, and Drupal) play a critical role in web content access and creation today. The work and

findings of CMSPY show that just standard static crawling over a CMS-based website reveals so much data and the unawareness ofsecuring your CMS environment needs utmost attention. [12] Guaranteeing us- ability and security in web CMS is the top priority for web administrators as this informs the use and exhaustive utilization of web-based CMS applications and their integration with other applications running in an organization. While most web CMS provides easy access to web services to the users, they are vulnerable to security breaches and threats. This paper emphasizes the security vulnerabilities in web CMS as perceived in the past and present and the level of awareness of these vulnerabilities and control measure, and propose a security solution for theproper assessment and identification of such web CMS security vulnerabilities. Exploration of the latest available AI cyber security Research clearly shows pos- itive advancements in the detection of new vulnerabilities. Hence there is a needto bridge the gap between CMS developers and security by bringing and devel- oping more such tools and stronger alternatives that automate the entire testing,validating, and patching of the web CMS by integrating penetration testing andAI into the solution.

Declarations

The manuscript has not been submitted in any other journal or conference.

Conflicts of Interest

There are no conflicts to declare.

Funding: The study received no funding from any source.

Acknowledgment: The authors extend their sincere appreciation to the editor and the anonymous reviewers for providing valuable feedback that significantly contributed to the improvement of this paper.

References

Maraga, A., Awuor, F. M., Ogalo, J. (2022). Model for security controls in web content management system. Journal of Internet and Information Systems, 11(1), 1-12.

Kumar, R. Kumar, K.. (2017). Exploitation of content management system vulnera- bilities to launch large scale cyber attacks. International Journal of Civil Engineering and Technology. 8. 1381-1395.

HRVOJE JERKOVIC, BRANKO SINKOVIC. (2017). Vulnerability analysis of most popular open source Content Management Systems with focus on WordPressand proposed integration of artificial intelligence cyber security features. Interna- tional Journal of Economics and Management Systems Volume 2, pp. 66-74, 2367- 8925.

Conţu, C.A., et al. (2016) Security issues in most popular content management systems. in Communications (COMM), International Conference on. 2016. IEEE.

Cimpanu, C. (2017) Google Makes WordPress Site Own- ers Nervous Due to Confusing Security Alerts.

M. Meike, J. Sametinger and A. Wiesauer, (2009) Security in Open-Source Web Content Management Systems. IEEE Security Privacy, vol. 7, no. 4, pp. 44-51, July- Aug. 2009, doi: 10.1109/MSP.2009.104.

Alghofaili, Rasha, (2018) Security Analysis of Open-Source Content Management Systems Wordpress, Joomla, and Drupal. Master Thesis at California State Poly- technic University, Pomona Martinez-Caro, Jose-Manuel, Antonio-Jose Aledo-Hernandez, Antonio Guillen- Perez, Ramon Sanchez-Iborra, and Maria-Dolores Cano. (2018) A Comparative Study of Web Content Management Systems. Information 9, no. 2: 27.

Short, C. (2010) Web content management: CMS for competitive advantage. Journal of Direct Data and Digital Marketing Practices 11, 198–206.

Meike, M., Sametinger, J., Wiesauer, A. (2009). Security in Open Source Web Content Management Systems. IEEE Security Privacy, 7.

Vishnu, V., Praveen, K. (2022). Identifying Key Strategies for Reconnaissance in Cybersecurity. In: Agrawal, R., He, J., Shubhakar Pilli, E., Kumar, S. (eds) Cy- ber Security in Intelligent Computing and Communications. Studies in Computa- tional Intelligence, vol 1007. Springer, Singapore.

Suresh, M., Amritha, P. P., Mohan, A. K., Kumar, V. A. (2018). An investigationon HTTP/2 security. Journal of Cyber Security and Mobility, Vol. 7, Issue 1-2, pp.161- 180.

Remya S., Praveen K. (2016) Protecting the Augmented Browser Extension from Mutation Cross-Site Scripting. In: Satapathy S., Raju K., Mandal J., Bhateja V. (eds) Proceedings of the Second International Conference on Computer and Com- munication Technologies. Advances in Intelligent Systems and Computing, Vol. 379, pp. 215-223. Springer, New Delhi.

Aravind V., Sethumadhavan M. (2014) A Framework for Analysing the Security of Chrome Extensions. In: Kumar Kundu M., Mohapatra D., Konar A., Chakraborty

Nath H.V. (2011) Vulnerability Assessment Methods – A Review. In: Wyld D.C., Wozniak M., Chaki N., Meghanathan N., Nagamalai D. (eds) Advances in Net-work Security and Applications. CNSA 2011. Communications in Computer and Information Science, Vol 196. pp. 1-10. Springer, Berlin, Heidelberg.