Vulnerability analysis of most popular open source Content Management Systems with focus on WordPress and proposed integration of artificial intelligence cyber security features

HRVOJE JERKOVIC, BRANKO SINKOVIC
IT Department
Zagreb School of Economics and Management
Jordanovac 110, Zagreb
CROATIA
hrvoje@zsem.hr, bsinkovi@zsem.hr

Abstract: Web sites are major sources of information today and Internet is dominating platform for deployment of various applications built for worldwide audience. Modern Content Management Systems (CMS) play major role in that situation since they enable technical users to build various standard and custom web applications but they also enable non-technical users to build various web sites and applications using standard and extended set of tools provided by CMS developers. This paper will analyze latest major CMS vulnerabilities, effectiveness of various security communities' responses with propositions for improvements.

Key-Words: - Content Management Systems, web site security, cyber security, web filtering, web firewall, vulnerabilities, artificial intelligence

1 Introduction

CMS play major role today in creating and publishing web content online. From era of Web 1.0 creation and publishing of web content has rapidly changed. Today non-technical users can easily create sophisticated web applications with various available web platforms and additional tools.

Besides creating just web sites users can now quickly learn how to create and run web shops and similar interactive business applications.

Simplification of such procedures has positive influence on sharing of knowledge, information and boosting of business activities.

But at same time security threats are rapidly rising as well, allowing various malicious activities associated with known vulnerabilities in most commonly used CMS platforms.

Market analysis shows that three major CMS platforms are dominating market today – WordPress, Joomla and Drupal.

This research analyses CMS architecture and advanced CMS features from perspective of security trends and put them in appropriate context of current security trends in web development.

Research will also identify and classify common CMS security vulnerabilities and current vulnerability trends.

Main part of the research will focus on latest vulnerabilities, common scenarios and contexts in which attackers exploit them.

Research will also analyze response procedures of security communities involved in mitigating newly discovered vulnerabilities. Fast response is crucial today especially in case of heavily used CMS like WordPress, Joomla and Drupal.

Final chapters of this research will give proposition for improvements of standard security measures in case of discovery of new vulnerabilities.

2 Current research

General problems in areas of web application security threats are very well covered by various researchers. Core of the problem, according to researchers in [1], is that web security needs to address the research challenges in Javascript execution environment, safe inclusion of third party contents and regular audits of embedded contents in web applications. Those are also some of the core CMS related security issues.

However, increasing growth of malicious websites and systems for distributing malware through websites is urging adoption of effective techniques for timely detection of web security threats. So researchers in [2] propose a set of features extracted from the content and the structure

of webpages, which could be used as indicators of web security threats.

Many researchers covered topic of web application testing and prevention methods against different security flaws. Researchers also focused on various tool for testing security of CMS [3],[4],[5] and various corporate solutions for testing web applications for security vulnerabilities in general are continually being patented as well [6].

However, there is lack of latest CMS security research analysis. Researchers in [7] compared security of three most used CMS (WordPress, Drupal and Joomla) and another from same year (2013) [8] explored security issues.

Responsible development of CMS with security in mind is topic of research for researchers in [9].

Quality of plugins and their effect on security of CMS is another popular topic of research [10],[11].

We can conclude that there is no recent research in area of CMS security that analyzes responses of various communities and organizations involved when vulnerability is discovered.

This research also covers analysis of all other relevant major security factors with analysis of latest statistics that is of relevance to overall security of CMS [12].

3 Market analysis

According to [13] more than 53% websites do not use CMS at all, rest of the market is divided by WordPress which has taken 58,7% (27,5% of all websites) followed by Joomla's 7,1% (3,3% of all) and Drupal's 4,8% (2,2% of all websites) as of February 2017. According to the market share development of these market leaders, these positions are stable for last two years [14].

Besides popularity, choice of CMS is driven by user's preference towards flexibility of features versus ease of use [15] that's why we see a lot of users using CMS with most versatile and most comprehensive list of plugins (free or premium).

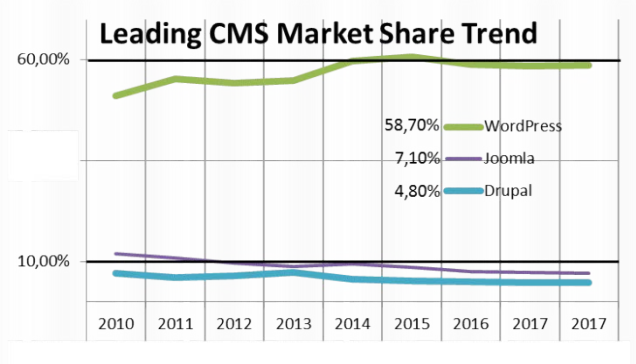

Figure 1. Market trends for tree leading open source CMS

4 CMS architecture and advanced features

Unlike static web, pages used in CMS don't exist as files on the web server, but are generated „on the fly" as answer on client's HTTP request. Page content (title, text, links etc.) supplied by author is retrieved from the database and merged with the selected page templates into html/css/xml file which is then sent to client.

Generic CMS consists of database server used to store content, user information and site metadata, such as link between content and page templates. Application server which generate pages and execute programs like page editing, user rights administration as well as various plugins and extensions. Web server handles client HTTP requests and web based, usually WYSIWYG (What You See Is What You Get) content authoring tool.

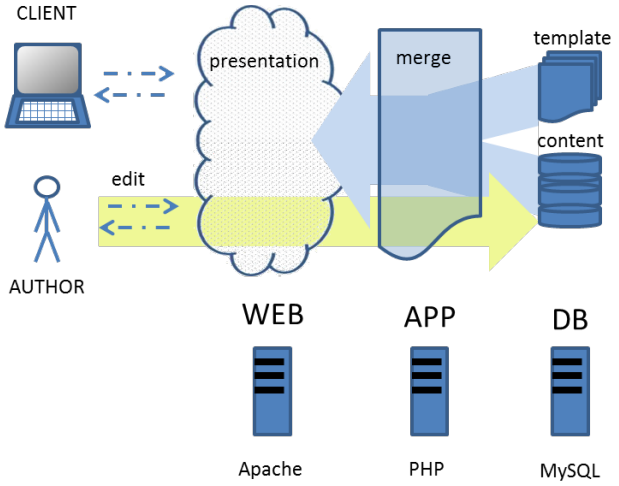

Figure 2. Usual architecture for open source CMS

Most used choice of components for open source CMS is MySQL database server, PHP programming language based application server and Apache web server which can all be run on Linux or Windows operating system [16].

Standard CMS features include web, blog and social media content editing and publishing, user rights administration, site and page template selection and some basic statistics. Besides well designed and search engine optimized pages which are displayed to the client at the resulting site, appropriate navigation structure is strongly correlated with CMS popularity [17].

More advanced features are available under different names for different CMS [18]: plugins for WordPress [19], extensions for Joomla! [20] , modules for Drupal [21]. Example of these features

are SEO (Search Engine Optimization), CAPTCHA security feature, gallery, backup, forms, event scheduler and Google Maps integration.

Using advanced add-on features became more widely spread security issue. Generally web sites that are using a CMS suffer about three times [22] more attacks than non-CMS web sites.

The main security issues in open-source CMS are:

- Data integrity compromise using SQL injections and parameter manipulation in order to bypass validation mechanisms
- Accessing confidential data (gaining unauthorized access using SQL injections or XSS attacks)
- Phishing (gaining confidential information via e-mails or malicious input forms).
- Code execution (exploiting input improperly validated) [22]

5 Latest vulnerabilities report on WordPress CMS

General situation with CMS security is concerning. According to statistics from more than 40,000 WordPress websites in Alexa top 1 million [23], more than 70% of WordPress installations are vulnerable to hacker attacks [24]. Latest news from February 2017. announced that WordPress (most used CMS) newly discovered vulnerability allowed hackers to infiltrate and vandalize around 2 million web sites [25]. Security experts reported average growth in defaced pages per campaign of 44%. Largest hacking campaigns are shown on Figure 3. Total number of defaced pages for all campaigns, as indexed by Google has grown from 1,4 million to 1,9 million in single day in critical week.

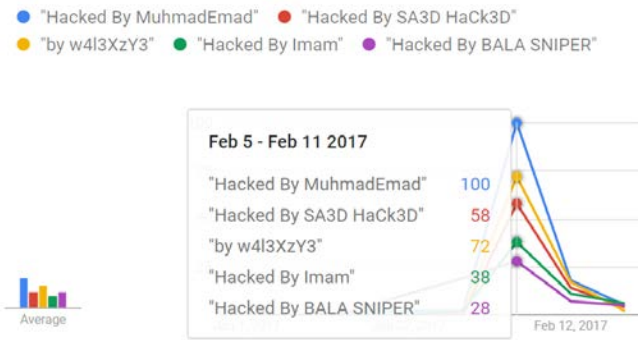

Figure 3. Largest hacking campaigns participating in abuse of WordPress 4.7 REST-API vulnerability

The exploited vulnerability was in the Representational State Transfer (REST) Application programming interface (API) – REST API. The weakness was only present in versions 4.7.0 and 4.7.1, with the issue fixed in 4.7.2. Unlike its closest competitors, Drupal and Joomla, WordPress knows how to apply security updates to itself automatically so version 4.7.2 was pushed through automatic updating system and majority of site owners didn't even notice it. In initial release WordPress team didn't mention this most critical vulnerability but only three less critical ones [26].

The vulnerability, which affects the WordPress REST API added in the 4.7 release, allows attackers to modify the content on any affected website remotely. When vulnerability like this is discovered it represents a massive opportunity for criminals and automated attacks can start within hours [27].

5.1 WordPress 4.7 REST-API vulnerability

Information about this severe WordPress vulnerability was added week later to public announcement of 4.7.2 security update release. Announcement stated "An unauthenticated privilege escalation vulnerability was discovered in a REST API endpoint".

WordPress REST API allows authorized computer programs access to WordPress content data so they can perform CRUD operations over WordPress content (blogs, pages, images). Websites have to check that the people or programs sending requests to their APIs are authenticated and authorized to perform those commands.

Discovered REST API vulnerability enables attackers to bypass those checks. Figure 4. shows how attacker could update existing post. Attacker can forge POST request with nonexistent ID like "*67istarget*" while actually targeting existing post with existing post ID "*67*" (*/wp/v2/posts/67?id=67istarget*). Instead of being stopped there, request is forwarded to "*update_item*" WordPress core function which is in charge for allowing updating of post. It would be expected that attack would fail at this point since post with ID "*67istarget*" doesn't exist since all WordPress posts have to have numerical IDs. Casting in integer is explicitly not needed here since "*update_item*" requires integer as argument. So attack would be successful with or without casting since a value will be automatically converted if an operator, function or control structure requires an integer argument, which is the case with "*update_item*" function.

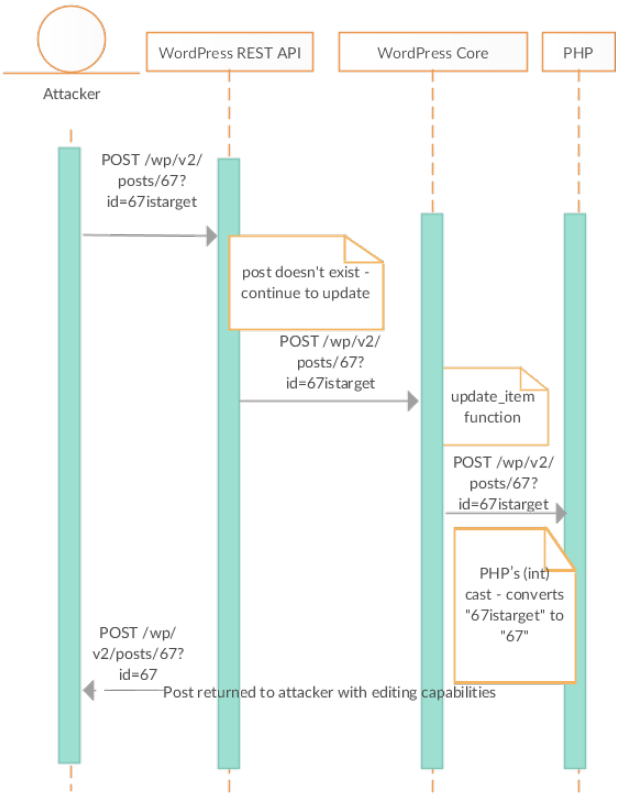

Figure 4. Wordpress REST API attack in 4.7.0 and 4.7.1 versions

5.2 Security response of parties involved

In order to propose adequate strategy for combating situations like this first of all we have to be able to analyze responses of all parties involved in this critical security situation.

Security flaw was first discovered by web security company Sucuri which core business is focused on network based integrity monitoring system for websites [24]. Sucuri alerted the WordPress team and they began work on the fix while monitoring for activity and alerting security vendors and web hosts about the problem. When the update was ready, six days after discovery, WordPress team took the decision to deploy the fix without disclosing details of it first, giving themselves an opportunity to stay ahead of the criminals who might exploit it [27].

Meanwhile, Sucuri added rules to their Web Application Firewall (WAF) to block exploit attempts against their clients [26] and WordPress team alerted other companies with WAFs including SiteLock, Cloudflare, and Incapsula and worked with them to create a set of rules that could protect more users. Wordpress team also contacted leading hosting companies privately with information on the vulnerability and ways to protect users. Data from all WAFs and WordPress hosts showed no indication that the vulnerability had been exploited by that time. Therefore, Wordpress team decided to delay disclosure of this particular issue to give time for automatic updates to run and ensure as many users as possible were protected before the issue was made public.

Millions of websites where updated without their owners knowing about importance of the update since initial disclosure didn't have details of most important vulnerability.

But also 2 million websites where hacked and attackers started to deface websites quickly and in few days later monetization on black SEO market started [28].

6 Analysis of security community response in case of security incidents

Various companies and communities are taking part in responding to security issues like this. In first place we have WordPress.org team, web security testing companies like Sucuri, web applications protection companies that provide WAF services to their clients (like SiteLock and Incapsula), CDN (Content Delivery Networks) companies that provide content caching and web security services (like CloudFlare and Akamai), web hosting companies, web site owners and popular search engines like Google.

On January 20th vulnerability was discovered, on January 26th update was released and on February 1st full disclosure of most important REST-API vulnerability was made.

Although all important security and hosting companies were informed about issues still rapid defacement of large number of web sites clearly demonstrated that many hosts were left uninformed and without reliable information on how to protect their web servers in case their clients are still running vulnerable version of WordPress.

There was no organized way of informing web hosting companies about seriousness of the issue at hand. Only information posted was update to initial security release information that came few days after initial release. WordPress team admitted that there is dedicated slack channel for the WordPress hosting community but it's only just in it's infancy.

In comments section of posted 4.7.2. release users were exchanging various information on how to protect their servers because there were no public release about that, at least there were no known publicly disclosed information about how to protect

most popular web servers used for hosting WordPress (Apache and nginx) [26].

Disclosing such highly sensitive information to only few privately held web security and content companies is not reliable practice especially when some of those companies haven't taken proper actions regarding WordPress security in past [29].

Few days after rapid defacement of millions of web sites Google sent email to many web site owners who allegedly run vulnerably version of WordPress. Reason – thousands of hacked pages started to appear on search results.

Some webmasters who received notices thought the email was a phishing attempt as WordPress is misspelled using a lower-case p. Others expressed confusion and anxiety receiving notices despite having already updated their sites. In many cases Google incorrectly detected WordPress version of web site because many installations are behind load balancer which is removing WordPress meta information from HTTP headers of web requests.

Also WordPress powered sites contain a meta generator that Google uses to detect which version is running: *< meta name="generator" content="WordPress 4.7.1" />*. However, Google does not monitor pages in real-time. If a site owner updates to WordPress 4.7.2 but the page indexed by Google is running 4.7.1, they'll receive a notice.

Very nature of the message that Google sent to web site owners were confusing leading many owners to believe that they will be penalized for not updating their web sites. Google spokesperson admitted they will need to work on the alert's wording to make the messages "more accurate and less confusing" [28].

7 Proposition for improvement of security community response

Security experts agree that WordPress security team properly reacted when delaying full disclosure of such severe vulnerability. But after all parties were informed, sites updated and firewall rules in place; WordPress should have informed public about vulnerability with full disclosure of how to protect web servers as well not only WordPress installations.

We propose better collaboration with communities and companies that supply and develop Web servers (open source and proprietary), Web Application Firewalls (hardware and application) and also with all (not just leading) Content Delivery Networks.

Such vulnerability should never be disclosed without publicly available procedures in place for securing all levels of server; operating system, web server, database server and not only application level.

All hosting companies should be able to protect themselves based on publicly disclosed information without seeking professional WAF solutions even if their clients don't update their installations.

Leading search providers should be informed and warned about how abuse of vulnerability could impact their services so they could prepare themselves.

These are all relatively easy to implement procedures if only there were agreement in place about them.

8 Proposition of integration of AI security services

What if hackers find new way to exploit computer platform or application before security experts? Extremely serious vulnerability lay undiscovered at the heart of much of the cloud platforms for seven years [30]. Same scenario could be happening with most used CMS at this moment.

Artificial intelligence applied in cybersecurity could be answer to this problem.

MIT (Massachusetts Institute of Technology) Computer Science and Artificial Intelligence Lab (CSAIL) [31] has led one of the most notable efforts in this regard, developing a system called AI^2, an adaptive cybersecurity platform that uses machine learning and the assistance of expert analysts to adapt and improve over time. The platform was tested during a 90-day period, crunching a daily dose of 40 million log lines generated from an e-commerce website. After the training, AI^2 was able to detect 85 percent of the attacks without human assistance [32].

In similar way IBM artificial intelligence platform Watson is used for analyzing unstructured data to read and learn from thousands of cybersecurity documents per month, and apply that knowledge to analyze, identify and prevent cybersecurity threats [33].

Google focused some of its AI efforts in trying to find and predict cyber security breaches in patent called "Pervasive, domain and situational-aware, adaptive, automated, and coordinated analysis and control of enterprise-wide computers, networks, and applications for mitigation of business and operational risks and enhancement of cyber

security" [34]. Google also joined this battle recently by filing patent request for "Cloud based firewall system and service". System will provide protection to customers' sites from attacks, leakage of confidential information and other security threats. Such firewall system can be implemented in conjunction with a CDN [35]. In time to come cooperation between Google and major CDN providers like Akamai or CloudFlare will probably intensify. Collaboration recently extended beyond just business partnership and into joint collaboration on new patents like "Supporting secure sessions in a cloud-based proxy service" which is Google patent but original assignee of the document is CloudFlare, Inc. [36].

8.1 Proposed integration of AI services in CMS

Leading CMS providers could benefit as well from AI cyber security solutions. From previously described incident it's obvious that Google don't have strong collaboration with CMS communities and on other hand is seriously focused on mitigation of business and operational risks and enhancement of cyber security.

In example of recent WordPress vulnerability, simple AI algorithms could detect unusual behavior and Google or IBM AI algorithms could easily detect suspicious behavior.

Latest vulnerability analysis described in previous chapters shows that current security responses is not sufficient to stop massive damage to millions of websites.

Proposed AI security features wouldn't use signature ("regular expression") rules to identify web based attacks. Using rules is a reactive approach and detection is only as good as the rules that are created, managed and updated.

Data science based approach which AI implements must be adopted for automatic creation and updating of CMS security tactics.

Detection could be based on analysis of remote access to any REST-API endpoints and other core functionalities. Parameters analyzed could include HTTP requests, WordPress Core functions accessed, user credentials supplied, origin of request, content generated, SEO discrepancy etc.

Such rules could be stored in security tactics profile for corresponding CMS, in our case WordPress. Figure 5. shows how whole mechanism could work. WP security tactics stores history as well as current security tactics. In short, attack is compared with WordPress security goals and consequences of attack/request are being compared with security goals. If no current tactic can mitigate attack, then new security sub-goals are generated to avoid compromise of security rules and existing rules are upgraded.

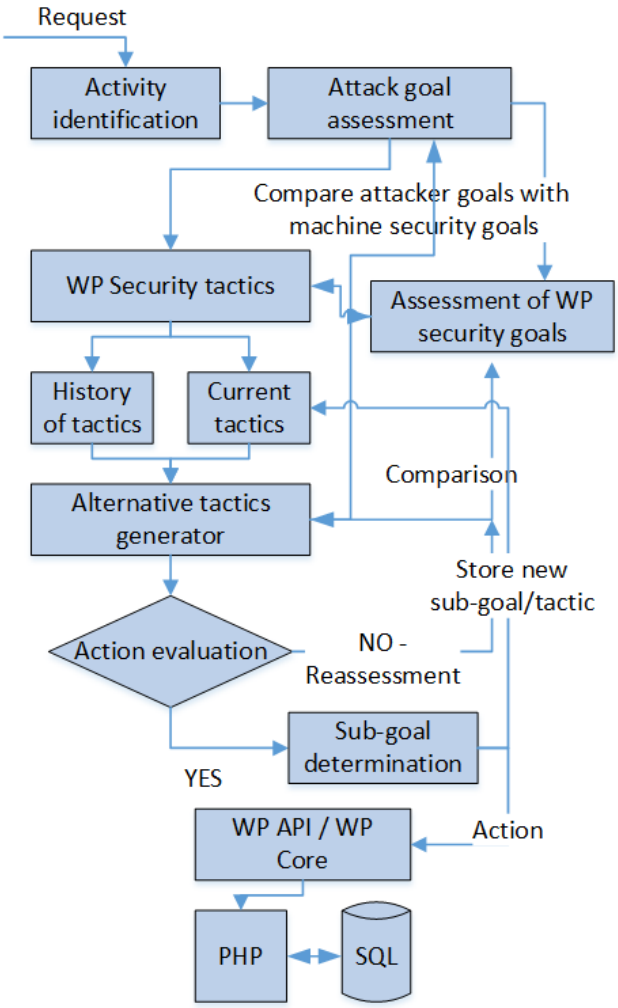

Figure 5. Proposition of AI security request profiling and protection

Based on this proposition WordPress 4.7.0 and 4.7.1 vulnerability could be easily detected even if it wasn't previously discovered by security experts.

On top of integrating proposed algorithms in WordPress, reporting feature should also by default be "on". This would enable notifications sent to Wordpress security team each time suspicious behavior is detected. In collaboration with research centers like one lead by MIT, specialized software like AI^2 could be utilized for analysis of received logs. Learning algorithms could also be improved over time for better and more efficient detection of possible vulnerabilities in CMS.

On top of that, users should be able receive notification about suspicious behavior on their sites. Visual dashboard with reporting features could enable them to access following proposed features:

- List and utilization of all core CMS functionalities by services, users, agents, programs with details for filtering (origin, endpoints, functions, sessions, content analysis)
- Information and representation of how plugins are using specific core functionalities
- Information and representation of how database is accessed
- List of unused but active functionalities that could be disabled
- Visual geographical map of users, services and programs accessing site and database
- Analyze CRUD operations per various properties and details and filter them out by country, service, program, plugin, user agent etc.
- Allowing users to create customized rules with simple rules editor.

9 Conclusion

Based on findings in this research we can conclude that CMS (especially leading ones like WordPress, Joomla and Drupal) play critical role in web content access and creation today.

This research also shows that major parties involved in handling web security don't have properly organized responses in case of newly discovered vulnerabilities.

While analyzing responses and actions of various parties involved in example of WordPress security vulnerability we managed to propose improvements that could lead to better and more organized management of security community responses in case of discovered vulnerabilities.

Exploration of latest available AI cyber security researches clearly shows positive advancements in detection of new vulnerabilities. Such features could be used and/or customized in CMS as well since they are already specialized in analyzing structured and unstructured content generation in similar systems.

Based on that we have proposed set of features that could enable early detection of various suspicious activities and enable worldwide detection of vulnerabilities in CMS platforms.

11 References

1. Patil, S., *Hare Hunting in the Wild Web: A Study of Web Security Threats and Solutions*. 2016.

2. Canfora, G. and C.A. Visaggio, *A set of features to detect web security threats*. Journal of Computer Virology and Hacking Techniques, 2016. **12**(4): p. 243-261.

3. Costa Nunes, P.J., J. Fonseca, and M. Vieira. *phpSAFE: A Security Analysis Tool for OOP Web Application Plugins*. in *Dependable Systems and Networks (DSN), 2015 45th Annual IEEE/IFIP International Conference on*. 2015. IEEE.

4. Jensen, T., et al., *Thaps: automated vulnerability scanning of php applications*, in *Secure IT Systems*. 2012, Springer. p. 31-46.

5. Sethi, S. and V. Singhal. *ICTS2016-SS27-07: A Peek into Web Applications Security*. in *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*. 2016. ACM.

6. Pistoia, M. and O. Tripp, *Testing WEB Applications For Security Vulnerabilities With Metarequests*. 2016, Google Patents.

7. Patel, S.K., V.R. Rathod, and J.B. Prajapati. *Comparative analysis of web security in open source content management system*. in *Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on*. 2013. IEEE.

8. Onishi, A., *Security and Performance*, in *Pro WordPress Theme Development*. 2013, Springer. p. 297-332.

9. Mansfield-Devine, S., *Taking responsibility for security*. Computer Fraud & Security, 2015. **2015**(12): p. 15-18.

10. Coelho Martins da Fonseca, J.C. and M.P. Amorim Vieira. *A Practical Experience on the Impact of Plugins in Web Security*. in *Reliable Distributed Systems (SRDS), 2014 IEEE 33rd International Symposium on*. 2014. IEEE.

11. Koskinen, T., P. Ihantola, and V. Karavirta. *Quality of WordPress plug-ins: an overview of security and user ratings*. in *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*. 2012. IEEE.

12. Jerković, H., P. Vranešić, and S. Dadić. *Securing web content and services in open source content management systems*. in *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2016 39th International Convention on*. 2016. IEEE.

13. Surveys, W.T.-W.T. *Usage of content management systems for websites*. 2017; Available from: https://w3techs.com/technologies/overview/content_management/all.

14. Surveys, W.T.-W.T. *Market share yearly trends for content management systems for websites*. 2017; Available from: https://w3techs.com/technologies/history_overview/content_management/ms/y.

15. Ogunrinde, M.A. and T.H. Yoosuf, *Performance Analysis on Content Management Systems: A Case Study of Drupal and Joomla.* Auditing. **4**: p. 6.

16. Tripathi, D., *Open Source Content Management System for content development.* 2015.

17. Gilani, S., et al., *A Navigational Evaluation Model for Content Management Systems.* Nucleus, 2016. **53**(2): p. 82-88.

18. Dombrowski, Q., *Drupal and other content management systems.* Doing Digital Humanities: Practice, Training, Research, 2016.

19. WordPress.org. *WordPress.org - Plugin Directory*. 2017; Available from: https://wordpress.org/plugins/browse/popular/.

20. Joomla! *Joomla! Extensions Directory*. 2017; Available from: https://extensions.joomla.org/browse/top-rated/.

21. Drupal. *Download & Extend.* 2017; Available from: https://www.drupal.org/project/project_module.

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

23. Mehrotra, S. and S. Kohli. *The Study of the Usage of Data Analytic and Clustering Techniques for Web Elements*. in *Proceedings of the ACM Symposium on Women in Research 2016*. 2016. ACM.

24. Design, I., *How to Improve Wordpress Security For Your Website| Security Plugins.* 2016.

25. Ball, T. *WordPress security weak spot lets hackers infiltrate and vandalise*. February 2017; Available from: http://www.cbronline.com/news/cybersecurity/breaches/wordpress-security-weak-spot-lets-hackers-infiltrate-and-vandalise/.

26. Campbell, A.D. *WordPress 4.7.2 Security Release*. January 26, 2017 Available from: https://wordpress.org/news/2017/01/wordpress-4-7-2-security-release/.

27. Stockley, M. *Critical WordPress update fixes zero-day flaw unnoticed*. 2017; Available from: https://nakedsecurity.sophos.com/2017/02/03/critical-wordpress-update-fixes-zero-day-flaw-unnoticed/.

28. Cimpanu, C. *Google Makes WordPress Site Owners Nervous Due to Confusing Security Alerts*. 2017; Available from: https://www.bleepingcomputer.com/news/security/google-makes-wordpress-site-owners-nervous-due-to-confusing-security-alerts/.

29. Design, W.F. *GoDaddy and SiteLock Make a Mess of a Hack Cleanup (And Drop The Ball on Security As Well)*. 2017; Available from: https://www.whitefirdesign.com/blog/2016/09/14/godaddy-and-sitelock-make-a-mess-of-a-hack-cleanup-and-drop-the-ball-on-security-as-well/.

30. Stockley, M. *Critical Xen vulnerability went undiscovered for seven years*. 2015.

31. Jones, M.T., *Artificial Intelligence: A Systems Approach: A Systems Approach*. 2015: Jones & Bartlett Learning.

32. Dickson, B. *Exploiting machine learning in cybersecurity*. 2016.

33. Palmer, C., et al. *Cognitive Cyber Security Assistants—Computationally Deriving Cyber Intelligence and Course of Actions*. in *2016 AAAI Fall Symposium Series*. 2016.

34. Ray, P.D., et al., *Pervasive, domain and situational-aware, adaptive, automated, and coordinated analysis and control of enterprise-wide computers, networks, and applications for mitigation of business and operational risks and enhancement of cyber security*. 2014, Google Patents.

35. Dilley, J.A., et al., *Cloud based firewall system and service*. 2015, Google Patents.

36. Prince, M.B., et al., *Supporting secure sessions in a cloud-based proxy service*. 2015, Google Patents.