Securing web content and services in open source content management systems

*H. Jerković, *P. Vranešić, **S. Dadić
Zagreb School of Economics and Management,
*IT department, Zagreb, Croatia,
**MBA graduate student,
(hrvoje, pvranesi, stipe.dadic) @zsem.hr

Abstract - Content management systems (CMS) are information systems designed dominantly for managing different types of publicly available web content although they could be used for various other purposes. Today, CMS open source solutions are one of the most popular platforms for developing web sites, portals, web shops and other publicly available content and services. Main problem in this area is lack of proper understanding of security issues and procedures which are frequently leaving content and services vulnerable for various types of attacks. This work focuses on risk analysis of main CMS open source systems from the point of security of data and services. Usual "points of failure" are analyzed, compared to similar solutions and final measures for full protection are proposed.

I. INTRODUCTION

Most used CMS in world today are WordPress, Joomla and Drupal which are all open source software built on PHP with MySQL database support primarily. WordPress alone is used by more than 23.3% of the top 10 million websites as of January 2016 with around 60% of CMS market share and more than 60 million websites altogether [1]. Joomla and Drupal together take around 10% of CMS market share [2]. We'll analyze main causes of security problems in CMS in general and compare their security with enterprise CMS solutions. Since CMS security dominantly depends on security of underlying systems (server, database server and web server) we will briefly cover comparison between usual underlying technology of three most used CMS (Linux server, MySQL,MariaDB or PostgreSQL as database servers and Apache or Nginx as web servers) with underlying Microsoft technology for running CMS (Windows Server, MS SQL as database server and .NET based CMS running on IIS[1] as web server). We'll give analysis of most abused and most severe cases of vulnerabilities and try to identify most common ways in which they are being exploited. Chapter 9. analyses importance of community in preventing widespread abuse of newly discovered vulnerabilities. Chapter 10. analyses effectiveness of CDN[2] and their ability to eliminate standard threats. Final goal of paper is to define overall best practices for securing open source CMS.

II. CURRENT RESEARCH

Research from 2013 [3] compared security of three most used CMS (WordPress, Drupal and Joomla) and another from same year [4] explored security and performance of main CMS. Other researchers are usually focused on general accessibility, usability and security [5]. Responsible development of CMS with security in mind is topic of research for researchers in [6]. Quality of plugins and their effect on security of CMS is another popular topic of research [7],[8]. Researcher also explored about software vulnerability markets [9] and also general comparisons researches are done without special focus on security related to three main CMS [10]. Some researchers also focused on various tool for testing security of CMS [11],[12]. CMS are also frequently analyzed from point of application usefulness for various data integration purposes in LMS[3] and SIS[4] [13] or as comparison tool for support in learning environment where they could take role of Learning CMS (LCMS) [14]. We can conclude that there is no recent research in area of CMS security that covers influence of CDN networks and responses of developer community 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.

III. CMS FEATURES AND GENERAL PROBLEMS

CMS usually supports modular and extensible framework for installation of various plugins so common CMS core functionalities could be easily extended with many additional features. WordPress.org directory contains 42.807 plugins which are downloaded 1.16 million times (as of January 2016) [15]. CMS core functionalities are usually secure if updated regularly. Most common problems with such a large plugin base is that majority of plugins are out of date, not properly working with current release of CMS, they are bloated with additional content and many plugins are inherently unsecure because of poor outdated or reused code. Attacker identification of website with outdated CMS and or outdated vulnerable plugins can lead to security problem. Unfortunately all major CMS could be successfully identified remotely: WordPress, Joomla and Drupal are all in that category. Experts agree that security by obscurity is not true security but still there is agreement

[1] Internet Information Services
[2] Content Delivery Networks
[3] Learning Management Systems
[4] Student Information Systems

that obscurity is extremely helpful in situation where users are running major CMS with number of publicly disclosed vulnerabilities. So general security problems with CMS are: easy remote identification of system, poor programming practices while creating plugins, lack of oversight while submitting plugins, poor inspection of potential security issues with plugins and lack of auto-update of CMS installations and plugins which are leaving system open for knows CMS and plugins vulnerabilities.

IV. SERVER VULNERABILITIES

Experts agree that securing application itself is futile without server-side validation measures[16]. So in this chapter we'll compare general security of Linux servers (which are most commonly used for running three most used CMS) with security of other servers (mainly Windows servers). This is important so we can get clear understanding of overall security of supporting CMS technologies. Table 1. shows statistics based on data from NVD[5] [17]. We can conclude that Linux servers have approximately the same number of high level

TABLE 1. OPERATING SYSTEMS VULNERABILITY THREATS

	Number of vulnerabilities			
OS	**Total**	**High**	**Medium**	**Low**
Apple Mac OS X	147	64	67	16
Linux Kernel	119	24	74	21
MS Win Server 2008	38	26	12	0
MS Win Server 2012	38	24	14	0

vulnerabilities as Windows Server 2012 (which is still one of the most used Windows Server OS). Linux Kernel has significantly larger number of medium size threats detected (74) in comparison with Windows Server 2012 and 2008. Since those are lower level threats, we can argue that with proper maintenance all main server types offer adequate level of security for hosted services and applications. Experts agree that although proper server maintenance is important, main cause of security problems are usually caused by outdated applications [18].

V. WEB SERVER VULNERABILITIES

Three major open source CMS have to be hosted on web servers that support PHP/MySQL. So security of application greatly depends on security of web server. This chapter brings comparison of general security of web servers commonly used to host three major CMS and other currently used web servers. Apache and Nginx are usual choice for hosting major open source CMS so comparing their vulnerabilities with other web servers could give us better understanding of overall security of major CMS. We've tried to find relevant statistics that will show us usage of current most used web servers. Netcraft Web Server Survey explores usage statistics of current most used web servers. Survey has run since August 1995, exploring the Internet to find new websites. Every month, an HTTP request is sent to each site, determining the Web server software used to support the site through careful inspection of the TCP/IP characteristics of the response [19, 20]. Statistics shows that Apache web server is still most used web server (51%) and if we exclude Google (8%) and Windows servers (11%) next in line suitable for PHP/MySQL open source applications is Nginx with 14% of market share and steadily growing in usage.

If we compare vulnerabilities of Apache HTTP server, Nginx and IIS number of reported CVE[6] [21] since 2010 is: Nginx – 15 , Apache HTTP - 97 , IIS– 27. If we analyze number of reported CVE just for 2015 results are: Apache – 9, IIS – 2 and Nginx – none. So we could conclude that Nginx could be more secure choice for hosting any popular CMS.

VI. DATABASE SERVER VULNERABILITIES

Final goal of attacker is usually access to database so securing database is probably most important issue in overall security of web application. This chapter brings comparison of general vulnerabilities of databases used by three most used CMS (MySQL, MariaDB and PostgresSQL) and MS SQL Database as usual database of choice for .NET based CMS. Most popular CMS usually are using MySQL databases but since Oracle acquired it many users are turning to MariaDB which is a community-developed fork of the MySQL and it intends to remain free under the GNU GPL [22]. Researchers in [23] made comparison of both DBMS in 2012. but no recent study was published. PostgreSQL could also be viewed as viable alternative since many CMS are supporting it. Microsoft SQL Server in this case is proprietary alternative mainly used as database server for .NET based CMS and other web applications. *TABLE 2.* shows total vulnerabilities for each DBMS in last three years. In 2015. MySQL accounted for growing 76

TABLE 2. TOTAL VULNERABILITIES PER YEAR FOR MAIN DATABASE VENDORS

Total vulnerabilities per year	**Oracle MySQL**	**PostgreSQL**	**MariaDB**	**MS SQL Server**
2013.	66	6	3	0
2014.	63	9	1	2
2015.	76	3	0	3

detected vulnerabilities but it's also worth noting that none of those vulnerabilities have score higher than 8/10. Number of total vulnerabilities for MariaDB is 4 and for MS SQL is 5 in last 3 years.

So we can conclude that MariaDB and PostgreSQL are more secure alternatives when it comes to choice of database support for CMS.

VII. CONTENT MANAGEMENT SYSTEMS VULNERABILITIES

Main cause of security problems are CMS application itself. We'll analyze security situation for main most used CMS: WordPress, Joomla and Drupal and compare that to DotNetNuke (.NET based CMS) and phpBB (PHP based CMS with market share equal to DotNetNuke). Table 3. shows number of detected vulnerabilities for compared CMS. Results vary from year to year but we can see that in 2015. situation is similar for all three main CMS. WordPress has much larger share of the CMS market than

[5] National Vulnerability Database

[6] Common Vulnerabilities and Exposures

TABLE 3. TOTAL VULNERABILITIES FOR LAST THREE YEARS FOR CMS COMPARISON

Total vulnerabilities	Word Press	Joomla	Drupal	DotNet Nuke	phpBB
2013.	20	11	14	2	0
2014.	29	10	35	3	0
2015.	11	13	10	1	2

Joomla or Drupal [2], nevertheless number of discovered vulnerabilities is aligned with those discovered in other popular CMS in year 2015. Figure 1. shows WordPress common types of exploits for period 2004-2015. Cross Site Scripting (XSS) with 73 CVE has leading number of vulnerabilities but after analysis of each vulnerability for 2015 we can see that CVSS[7] [21] score for all of them is between 3,5 and 4,3. Common example of such CVE is CVE-2015-5734 : Cross-site scripting (XSS) vulnerability in the legacy theme preview implementation in *wp-includes/theme.php* in WordPress before 4.2.4 allows remote attackers to inject arbitrary web script or HTML via a crafted string[24]. In regard with integrity impact of this threat and threats of similar level; we should add that modification of some system files or information is possible using this attack, but the attacker does not have control over what can be modified. Also scope of what the attacker can affect is limited and there is no impact to the availability of the system.

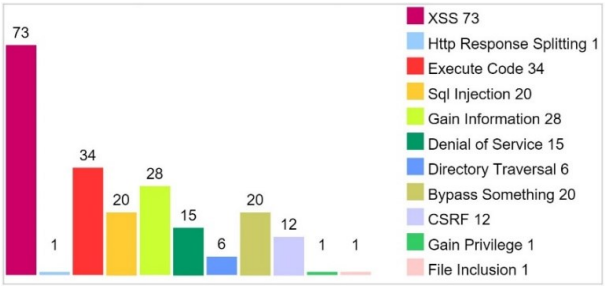

FIGURE 1. WORDPRESS VULNERABILITIES BY TYPE [25]

Most dangerous types of attacks are related to SQL injection vulnerability because these attacks have higher availability impact on the system. Usually there is reduced performance or interruptions in resource availability. In year 2015. only one CVE of such type is reported for WordPress with score 7.5 and situation is similar with Drupal. All other CVE discovered for WordPress are rated below 5.0. Joomla statistics for 2015 are showing 8 CVE registered with severity of 7.5, majority of them are SQL injection vulnerability which puts Joomla on the 1st place as most vulnerable CMS.

General hacking probability of CMS is defined in Figure 2. Hackers dominantly focus on CMS with greater market share, than on existing CMS exploits and server vulnerabilities.

[7] Common Vulnerability Scoring System

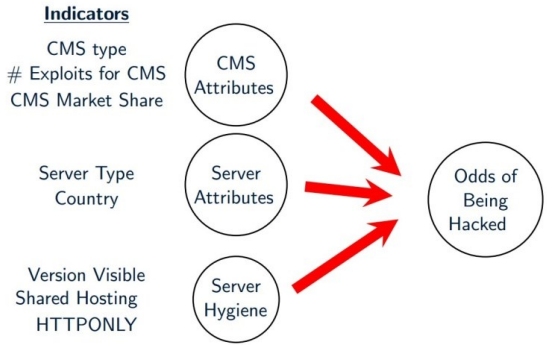

FIGURE 2. CALCULATION OF HACKING PROBABILITY

DotNetNuke CMS is most used .NET based CMS but it takes only 0,6% of market share and with only 6 vulnerabilities discovered in last 3 years. Intensity of usage greatly influences number of discovered vulnerabilities so we cannot easily conclude that DotNetNuke is safe choice. To prove that we have taken into comparison phpBB - PHP based CMS which has almost equally low market share as DotNetNuke (0.5%). *TABLE 3.* clearly shows that point is proven since total number of vulnerabilities in last 3 years is 6 for DotNetNuke and only 2 for phpBB.

We can conclude that overall security of main most used PHP based CSM applications is equal or better than of those .NET based.

VIII. ANALYSIS OF HIGH SCORE CMS VULNERABILITIES

In order to propose adequate security procedures we must analyze common vulnerabilities and common ways in which they are being exploited.

There are many ways in which CMS vulnerabilities could be exploited. Besides manual methods, recently many automated solutions appeared in forms of web kits for exploiting common vulnerabilities. Analysis of

TABLE 4. WORDPRESS VULNERABILITIES RANKING BY CVSS SCORE

CVSS Score	Number Of Vulnerabilities	Percentage
0-1		0.00
1-2		0.00
2-3	6	6.90
3-4	4	4.60
4-5	40	46.00
5-6	12	13.80
6-7	16	18.40
7-8	7	8.00
8-9		0.00
9-10	2	2.30
Total	**87**	

vulnerability severity for WordPress for period February 2012. to February 2016. shows 9 threats with score higher than 9 (see Table 4.). Severe threats that have CVSS score higher than 7 are falling in following types of attacks: SQL injection, code execution, denial of service (DoS) and Cross-site request forgery (CSRF).

SQL injection allows attacker injection of destructive SQL code in order to misuse database of CMS application [26]. This is a code injection technique that exploits a security vulnerability occurring in the database layer of an application and service. This is most often found within web pages with dynamic content [27]. In 2015 serious SQL injection vulnerability was discovered in WordPress. SQL injection vulnerability in the *wp_untrash_post_comments* function in *wp-includes/post.php* in WordPress before 4.2.4 allows remote attackers to execute arbitrary SQL commands via a comment that is mishandled after retrieval from the trash [28]. This vulnerability could for example allow attacker to change default email address of all users or only of administrator of web site to whatever he wants. But still attacker have to get database access in order to modify meta value of post metatag and execute injection code.

Second common scenarios include compromised plugins. Yoast SEO WordPress plugin [29] is used on millions of WordPress sites for improving sites presence on search engines [30]. In 2015. blind SQL injection vulnerability was discovered in that plugin. WPScan Vulnerability Database, which is Vulnerability Database for WordPress its plugins and themes, issued an advisory after responsibly disclosing the vulnerability to the plugins author. By having a logged-in author, editor or admin, and by visiting malformed URL, a malicious hacker could change sites database. This is a case where CSRF [8] vulnerability allowed blind SQL injection. Blind SQL injection is a type of SQL Injection attack that asks the database true or false questions and determines the answer based on the applications response. This attack is often used when the web application is configured to show generic error messages, but has not mitigated the code that is vulnerable to SQL injection.Hosting companies that are dominantly hosting WordPress sites (like Pressable and Siteground) reacted in two ways: Pressable initiated automatic update of all affected sites and Siteground has added a temporary fix to tide customers over in the meantime before they have the chance to update. The company added new security rules to its WAF[9], which actively filtered any possible incoming hacking attempts that try to exploit the vulnerability.

This is classical situation where flaw in one plugin can compromise millions of WordPress sites. In this chapter we are going to analyze response of the various actors and show how joint efforts of the members of community can produce excellent results. Team from WPScan Vulnerability Database promptly alerted Yoast plugin team about issue first and waited for release and deployment of update before releasing details of the issue. Immediately after update for compromised plugin was released, WordPress pushed that update as platform security update for all WordPress sites that were using Yoast plugin. Users that were using cloud based Wordpress.com plugin called Jetpack could automatically update all critical plugins. Users can turn this option on in Jetpack dashboard administration [15]. Obviously choosing a professional hosting service can greatly improve on security of the site as demonstrated in this case. Site owners are usually concerned with bandwidth speed and space on servers without desire to spend extra money on securing their web sites. This is common problem and it's frequently neglected. Many realize potential and scope of threats only when they feel consequences on their own situation.

We can conclude that most frequently reported types of attacks are mainly using SQL injection vulnerabilities or CSRF in combination with SQL injection. Also we can conclude that having organized and interactive community (like one present around WordPress development) and professional hosting is essential for security of CMS solution.

IX. CLOUD PROTECTION FROM CONTENT DELIVERY NETWORKS

Additional layer of security can be achieved with CDN. CDN is a system of distributed servers that deliver webpages and other Web content to a user based on the geographic locations of the user, the origin of the webpage and a content delivery server [31]. CDN WAF offers protection against XSS and SQL injection attacks described in TABLE 4. WAF is frequently updated in cooperation with leading CMS providers so it can prevent new attacks on sites in CDN [32, 33]. Such secure delivery networks can help prevent or mitigate the most common attacks against websites. A case study from a leading provider of content delivery services (Akamai Technologies) illustrates CDN effectiveness, as authors in [34] described it. Akamai Technologies reported that distributed denial-of-service (DDoS) attacks against its customers are increasing in terms of both the bandwidth and the number of requests generated by the attackers. From 2009 to 2014, the size of the largest attack grew year by year from 48 to 68 to 79 to 82 to 190 to 321 in Gbit/s. Another global leader in CDN technologies, CloudFlare, confirmed that they mitigated the largest-ever recorded DDoS attack, which peaked at 400 Gbit/s in February 2014 [35]. At the same time, the number of packets per second in the largest attack grew from 29 million to 169 million. These kind of web application attacks are the most common cause of data breaches today [34].

In light of that, 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 [36]. 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 [37].

Unfortunately cloud solutions do not yet offer ideal environment. Some researchers have explored novel "origin-exposing" attack vectors which can be used to discover the IP address of the server where website is hosted [38]. To assess the impact of the origin-exposing vectors on the security of CDN protected websites

[8] Cross-Site Request Forgery
[9] Web Application Firewall

researchers consolidate all vectors into CloudPiercer. This is automated origin-exposing tool, which they use to conduct the first large-scale analysis of the effectiveness of the origin-exposing vectors. Results show that the problem is severe: 71.5% of the 17,877 CDN protected websites tested, expose their real IP address through at least one of the evaluated vectors [38].

X. Best practices and Content Management System protection discussion

Based on research from previous chapters we've defined list of best practices for securing CMS. Most important of them are:

- Regularly updating CMS, programming and database support, web server and server itself. If all of that is done regularly it's almost impossible to hack major CMS applications,
- Limiting logging attempts, obscuring admin page and installation itself, enforcing strong passwords, use discreet error messages, regularly check for proper permissions, limit IP and country access if necessary.
- Deleting known IDs and names for administration access (like Admin) and use random IDs for admin accounts,
- Using professional hosting services with security teams specially dedicated to monitor security issues with CMS of a choice,
- Using professional CDN service with option to automatically add new and recommended firewall rules.

Using web firewall and security plugins (like iThemes Security or Wordfence) is not effective in case previously defined important factors are not covered [39]. Experts demonstrated how fully patched systems with all security plugins in place can still be compromised [40]. This could be countered if hosting is moved in cloud environment. Since attackers are using standardized exploit kits and standardized codes massively [41] it's easy to spot breach in cloud environment if cloud hosting service is hosting thousands of WordPress installations for example. If there's communication with development team of CMS application than rules can be quickly added or changed easily. For specific most used web applications attacks are detected daily and rules are added immediately to prevent them [42].

XI. Conclusion

Based on findings of this research we can conclude that servers, web servers and database servers used to host most popular CMS discussed in article are not less secure than those used to host other CMS if proposed alternatives are used. We also concluded that number of discovered CMS vulnerabilities in three major CMS is correlated with their market share (chapter 7.). Analysis of most frequently reported types of attacks showed that attacks on three major CMS are mainly using SQL injection vulnerabilities or CSRF in combination with SQL injection. Analysis of community responses in case of security incidents showed that it's important to choose CMS which has live and interactive community. WordPress development team is good example of such community where automated security updates are created and deployed as soon as threat is discovered. Research also showed that CDN WAF offers high quality protection against main vulnerabilities described in chapter 7. All other recommendations for further securing CMS are summed up in chapter 11. where we concluded that even if CMS is fully updated it can still be compromised when it's not hosted in CDN with cloud based protection like WAF. Current research confirms that hosting CMS without cloud web protection is becoming increasingly vulnerable even if all other measures are taken.

References

[1] J. Ji, J. Kim, Y. Kim, S. Jung, C. Lee, D. Kim, *et al.*, "Design and Implementation of the Korean Style Plug-In using the Wordpress."

[2] S. a. t. i. t. u. o. c. m. s.-. W3Techs. (2016). *Usage Statistics and Market Share of Content Management Systems for Websites*. Available: http://w3techs.com/technologies/overview/content_management/all/

[3] S. K. Patel, 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, pp. 344-349.

[4] A. Onishi, "Security and Performance," in *Pro WordPress Theme Development*, ed: Springer, 2013, pp. 297-332.

[5] R. Ismailova, "Web site accessibility, usability and security: a survey of government web sites in Kyrgyz Republic," *Universal Access in the Information Society,* pp. 1-8, 2015.

[6] S. Mansfield-Devine, "Taking responsibility for security," *Computer Fraud & Security,* vol. 2015, pp. 15-18, 2015.

[7] J. C. Coelho Martins da Fonseca 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, pp. 21-30.

[8] T. Koskinen, 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, pp. 834-837.

[9] A. Algarni and Y. Malaiya, "Software vulnerability markets: Discoverers and buyers," *International Journal of Computer, Information Science and Engineering,* vol. 8, pp. 71-81, 2014.

[10] A. Mirdha, A. Jain, and K. Shah, "Comparative analysis of open source content management systems," in *Computational Intelligence and Computing Research (ICCIC), 2014 IEEE International Conference on*, 2014, pp. 1-4.

[11] P. J. Costa Nunes, 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, pp. 299-306.
[12] T. Jensen, H. Pedersen, M. C. Olesen, and R. R. Hansen, "Thaps: automated vulnerability scanning of php applications," in *Secure IT Systems*, ed: Springer, 2012, pp. 31-46.
[13] P. Orduna, S. Botero Uribe, N. Hock Isaza, E. Sancristobal, M. Emaldi, A. Pesquera Martin, *et al.*, "Generic integration of remote laboratories in learning and content management systems through federation protocols," in *Frontiers in Education Conference, 2013 IEEE*, 2013, pp. 1372-1378.
[14] A. Iglesias, L. Moreno, P. Martínez, and R. Calvo, "Evaluating the accessibility of three open‐source learning content management systems: A comparative study," *Computer Applications in Engineering Education,* vol. 22, pp. 320-328, 2014.
[15] A. Singh, "How to Secure Your WordPress Site From Brute Force Attack," *watermark,* 2016.
[16] M. Pistoia, O. Segal, and O. Tripp, "Detecting security vulnerabilities in web applications," ed: Google Patents, 2015.
[17] A. o. t. U. S. D. o. Commerce. (2.2.2016). *National Vulnerability Database*. Available: https://nvd.nist.gov/
[18] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," *Journal of Internet Services and Applications,* vol. 4, pp. 1-13, 2013.
[19] G. K. Chaudhary, "Development Review on Phishing: A Computer Security Threat," 2014.
[20] M. Yağci and M. Ünal, "Designing and implementing an adaptive online examination system," *Procedia-Social and Behavioral Sciences,* vol. 116, pp. 3079-3083, 2014.
[21] E. Weintraub and Y. Cohen, "Continuous Monitoring System Based on Systems' Environment," in *Proceedings of the Conference on Digital Forensics, Security and Law*, 2015, pp. 151-160.
[22] D. Bartholomew, *MariaDB Cookbook*: Packt Publishing Ltd, 2014.
[23] D. Bartholomew, "MariaDB vs. MySQL," *Dostopano,* vol. 7, p. 2014, 2012.
[24] CVE_Details. (2016). *Vulnerability Details : CVE-2015-5734*. Available: http://www.cvedetails.com/cve/CVE-2015-5734/
[25] http://www.cvedetails.com/. (2016, 3.32016). *Wordpress: Vulnerability Statistics*. Available: http://www.cvedetails.com/product/4096/Wordpress-Wordpress.html?vendor_id=2337
[26] V. Bittal and S. Banerjee, "Prevention Guidelines of SQL Injection Database Attacks: An Experimental Analysis," in *Emerging Research in Computing, Information, Communication and Applications*, ed: Springer, 2016, pp. 23-32.
[27] I. Lee, S. Jeong, S. Yeo, and J. Moon, "A novel method for SQL injection attack detection based on removing SQL query attribute values," *Mathematical and Computer Modelling,* vol. 55, pp. 58-68, 2012.
[28] CVE_Details, "Security Vulnerabilities Published In 2015 (SQL Injection)," 2015.
[29] J. Aull, *WordPress SEO Success: Search Engine Optimization for Your WordPress Website Or Blog*: Pearson Education, 2014.
[30] S. Fernandes and A. Vidyasagar, "Digital Marketing and Wordpress," *Indian Journal of Science and Technology,* vol. 8, p. 61, 2015.
[31] E. V. J. Aikas, A. Agarwal, and B. N. Bershad, "Content delivery network," ed: Google Patents, 2015.
[32] T. L. Longren, *Wordpress Multisite Administration*: Packt Publishing Ltd, 2013.
[33] J. Stearns, "Create a Faster WordPress Website & Blog," 2013.
[34] D. Gillman, Y. Lin, B. Maggs, and R. K. Sitaraman, "Protecting Websites from Attack with Secure Delivery Networks," *Computer,* vol. 48, pp. 26-34, 2015.
[35] N. Mosharraf, A. P. Jayasumana, and I. Ray, "A Responsive Defense Mechanism Against DDoS Attacks," in *Foundations and Practice of Security*, ed: Springer, 2014, pp. 347-355.
[36] J. A. Dilley, P. Laghate, J. Summers, and T. Devanneaux, "Cloud based firewall system and service," ed: Google Patents, 2015.
[37] M. B. Prince, L. H. Holloway, S. N. Rao, and I. G. Pye, "Supporting secure sessions in a cloud-based proxy service," ed: Google Patents, 2015.
[38] T. Vissers, T. Van Goethem, W. Joosen, and N. Nikiforakis, "Maneuvering Around Clouds: Bypassing Cloud-based Security Providers," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1530-1541.
[39] A. K. Kyaw, F. Sioquim, and J. Joseph, "Dictionary attack on Wordpress: Security and forensic analysis," in *2015 Second International Conference on Information Security and Cyber Forensics (InfoSec)*, 2015, pp. 158-164.
[40] S. Sreedharan, "Does your WordPress Security Plugin really secure your site?," 2014.
[41] T. Taylor, X. Hu, T. Wang, J. Jang, M. P. Stoecklin, F. Monrose, *et al.*, "Detecting Malicious Exploit Kits using Tree-based Similarity Searches," in *Proceedings of the Sixth ACM on Conference on Data and Application Security and Privacy*, 2016, pp. 255-266.
[42] S. Prandl, M. Lazarescu, and D.-S. Pham, "A Study of Web Application Firewall Solutions," in *Information Systems Security*, ed: Springer, 2015, pp. 501-510.