Web Vulnerability in 2021: Large Scale Inspection, Findings, Analysis and Remedies

Borka Jerman Blažič
International Postgraduate School Jožef Stefan
IJS; Laboratory for Open systems and networks
Ljubljana, Slovenia
e-mail: borka@e5.ijs.si

Primož Cigoj
Institute Jožef Stefan
Laboratory for Open systems and Networks
Ljubljana, Slovenia
e-mail: primoz@e5.ijs.si

Abstract: **The main focus of the cyber-security community has been to make operating systems and communication networks more secure and harder for attackers to penetrate. The most frequently used web application and user web pages are developed today with the Web Content Management System (WCMS), as it allows user-friendly access, easy development and operation. Any malware that can penetrate the WCMS can significantly affect the system itself and the service the web pages offer. This paper presents the approach for identifying the vulnerabilities of the majority of Internet sites with WCMS applications and the remedies to be applied with the use of an automated, fast and dynamic vulnerability detection tool. The state of the web sites vulnerability in Europe and the impact factors that influence the vulnerability to be present are presented and discussed.**

Keywords- cybersecurity; scanners and crawlers; WCMS vulnerability; security state of European web space.

I. INTRODUCTION

We do business, pay through the Internet, store documents and share our personal information, card numbers and identification data online very frequently. There is no doubt that this data is private and should be stored as safely as possible. The vision of the future Internet involves building a new generation of applications made by merging services and data from different providers and organizations [1]. Web services provide the basic interface between the provider and the consumer, supported by complex software components like the operating system, the server application and many additional systems like databases, shops and selling systems, appliances for different services, etc. Web services are subject to several unique security concerns, due to their pervasiveness, seamless interoperability and operations that can be remotely invoked by the user, and they need to be carefully considered in view of the envisioned architecture of the future Internet. The major web-security concern is related to the differences between the web applications that do not have embedded security protection and the security solutions present in traditional messaging techniques applied within other Internet services. For instance, the SOAP protocol used in web-service communications does not address the security itself and can be bypassed by a firewall [2].

This article provides a brief overview of the tools used for inspecting the web vulnerability at large and the newly developed tool called VulNet that scan the Internet web space at large for identifying vulnerability within websites built with WCMS (Web Content Management System) application. The tool is an advancement in the field when compared with other known proprietary or open-access tools. Its major improved properties are fast scanning at large, ethical search of vulnerability, acceptable scoring mechanism enabling comparison of the security between the web spaces in different regions of the world. The paper is divided into five sections. After the introduction, the second section introduces the reader to the area and describes the problem being addressed. The third section presents the tool components and its functionality. The next section provides an overview of the results and informs about the factors that impact the appearance of higher presence of vulnerability among particular web spaces. The paper ends with a conclusion and points to the limitation of the tool and the presented study results.

II. OVERVIEW OF THE AREA

A. WCMS

The continuous evolution of networks based on Internet technology has made its services very attractive and many different new applications appeared with the use of WCMS. A modern Content-Management System (CMS) like WordPress simplifies website creation as it allows the functionality of the site to be extended with additional applications known as plug-ins that are available for downloading from known databases. Currently, the estimated number of plug-ins is close to 54,000 and the total number of downloads is close to 900 million. Public web applications are usually accessible from anywhere in the world, but many corporate web applications that are set on networks with restricted access are also accessible. Web applications handle very sensitive information, ranging from banking to health directories, as well as personal images and photographs that are of interest to criminals and attackers. According to a survey carried out by W3Tech, about 52.9% of Internet websites use some kind of web-content management system [3]. The most popular open-source Web-Content Management Systems (WCMSs) are

 ISBN: 978-1-61208-880-8

WordPress, Drupal and Joomla [4]. A technology survey by BuiltWith Pty Ltd in 2017 concluded that about 46% of the top one million websites use WordPress [5]. The reason is that WCMSs allow users, even without an in-depth knowledge of web technology, to deploy and offer system content to users. Due to the popularity of these systems, they have become an interesting target for malicious attackers, and, therefore, the importance of the security features and the overall vulnerability of these applications have become very important, especially in cases where the web-content owners do not possess the necessary knowledge and understanding of the possible threats to the system. Writing computer programs is a complex task and modern software development usually involves combining many libraries and frequently not all the bugs have been removed in the developed software. Design errors become a risk, especially when the security of the program has not been taken into consideration from the beginning of the design process. The architecture and the design of a computer system are expected to be coherent and to follow the security principles, but this is not the case in the current web space [5]. Knowing the system's vulnerability represents the most vital and precious information for malicious parties. The removal of the vulnerability increases the resilience of the underlying system. That is why the operation and management of the web system should be actively monitoring and removing the vulnerable parts of the system to prevent possible attacks. The vulnerability testing of websites can be performed using two approaches. One is called white-box testing, in which the testing software has access to the source code of the application and this source code is then analyzed to track down defections and vulnerabilities in the code. These operations are expected to be integrated into the web-development process with the help of add-on tools within the development environments, but they are usually not used, especially when the system is upgraded with new plug-ins to enhance the service and user satisfaction. The other approach is called black-box testing, where the tool has no direct access to the source code, but instead it tries to find vulnerabilities and bugs with special input test cases that are generated by the tool and then sent to the application. Responses are then analyzed for unexpected system behaviors that indicate the errors or vulnerabilities of the system. A black-box security scanner typically uses a mixture of passive (typically, during the crawl) and active (typically, post-crawl) vulnerability- testing techniques like code execution [6].

B. Crawlers and scanners

Identifying the vulnerabilities across the whole web space of the Internet is not an easy task, though this information is extremely valuable, helpful and required by the website owners. The available vulnerability-testing tools are either restricted to internal use by the owners of corporate or organization networks, as they use software mimicking real attacks, or they just scan the basic web server's vulnerability, without providing sufficient information about the whole WCMS system and the associated plug-ins. A common approach for inspecting the web vulnerability is scanning the Internet sites and associated domains with the use of a web crawler in combination with a search engine, such as Google. Web crawlers, however, have multiple problems. Some crawlers access the same URLs [7] more than 1000 times, as there is no intelligence in-built into the crawler that help in avoiding repeating accesses to a web site. Any web-vulnerability inspection of the web application, besides the crawler, needs additional software, as the information sought beyond the port data is located in the plug-ins and in the web pages applications. The detection of infinite loops and the actual depth of crawling in the web space are difficult as the websites are not static and the web pages change over time due to user intervention. A more difficult problem is related to the large number of pages and the amount of data included. As a consequence, the crawling process can take a long time and the results are frequently not a snapshot of the system, as multiple pages might have changed during the scan. However, in recent years some improvement to Internet-wide scanning was achieved with tools such as ZMap and MasScan [8]. ZMap was developed by the University of Michigan and is now the main tool of the Internet-search service known as Censys [9]. Shodan is a similar service that uses behavior or grab techniques to identify the vulnerabilities of sensors and similar devices. Shodan collects data mostly on web servers, but it is supplied with applications to access FTP servers and other known Internet ports such a Telnet (virtual terminal), SNMP (mail), IMAP (encrypted mail) and the Real Time Streaming Protocol. The latter are used to access web cameras and their video stream. However, Shodan does not conduct a deep review of the sites. Despite the popularity of these tools, they are not real crawlers, but rather ports scanners looking for the HTTP type of servers that are usually extended with additional applications. The collected port information using these tools does not include information about the vulnerability of the applications and the plug-ins as they operate only with an IP address and do not crawl links within the website's content. The systems are proprietary, but the service is publicly available. A similar publicly available tool is called Nmap [8], which requires multiple machines and weeks to complete any horizontal scan of the public address space, making it rather slow [5]. Running regular web vulnerability scanners against numerous websites is time consuming and, if exploit techniques are used, the scan is considered as illegal if browsing permission is not granted by the owners. Another way of detecting the vulnerability without breaking the law [6] is to detect the application and then identify its issuing version or its fingerprint and then look in a database with the identified vulnerabilities of that particular version. The most well-known vulnerability database with vulnerable plug-ins is the National Vulnerability Database (NVD) that is hosted by the National Institute of Standards and Technology [10] and is used by most of the known scanners. The capacity of scanning web

 ISBN: 978-1-61208-880-8

sites differs among the scanners. The first group of known scanners usually scans data sets between 20,000 and 200,000 websites [11]–[13], but forgetting the rest of the web, and they are focusing on specific vulnerability, such as XSS, SSL, SQL injection, phishing, Heartbleed and search-redirection attacks, instead of covering all of them at once. In addition, their methods are also time-consuming: they need more than 9 days to measure a dataset of 200,000 websites. They focus on determining whether a given input propagates, rather than efficiently finding the propagating inputs, for arbitrary vulnerabilities [12] [14]. The second group performs the scanning of the Internet IPv4 protocol for a specifically defined subject area, such as hosted services, SSL/TLS, vulnerabilities or specific software or protocol vulnerabilities by using mass scan tools such as ZMAp, Nmap and Massscan [15]–[17]. This technique is good for the fast TCP/IP stack-fingerprinting technique to identify the OS's type, port range scan, and basic web, but not for a detailed overview of the online content vulnerabilities. In this case the CMS's core and plug-in vulnerabilities are not inspected.

TABLE I. TOOLS COMPARISON

Characteristics / Tools		Shodan	Censys	WPScan	[5]	[11]	[12]	[13]	VulNET
General Characteristics	OpenSource	No	No	Yes	No	No	No	No	Yes (A)
	URL or IP	IP	IP	URL	IP	URL	URL	URL	Both
	Results are Freely Accessible on the Internet	Yes (P)	Yes	No	No	No	No	No	Yes
	Internet-connected Devices	Yes	Yes	No	Yes	No	No	No	Yes (E)
	Automatic Scanning	Yes	Yes	No	Yes	Yes	No	No	Yes
	Ethical	Yes	Yes	Yes (P)	Yes	No	Yes	Yes	Yes
	Web UI and Command Line (CL)	Both	Both	CL	CL	No	No	No	Both
	Real-time Visualization while Scanning	No	No	No	No	No	No	No	Yes
	Free API	Yes (P)	No	No	No	No	No	No	Yes
	More than 1 million scanned IPs or Websites	Yes	Yes	No	Yes	No	No	No	Yes
WP Specific	CMS Scan	No	No	Yes	Yes	Yes	Yes	Yes	Yes
	CVE Exposer	Yes	No	Yes	Yes	No	No	No	Yes
	Plugins Scan	No	No	Yes	No	No	No	Yes (L)	Yes
	Scoring	No	No	No	No	No	No	No	Yes

P = Partly, E = Extension, A = Attended, L = Limited.

IV. VULNET AND ITS FUNCTIONALITY

The general methodology for web scanning and data collecting consists of six steps: a) collecting the IP addresses of the web targets, b) accessing and c) getting responses, d) sending queries for application patterns, e) collecting vulnerability information and f) saving and validating the results. The last three steps in current web scanners differ very much due to their capability to catch relevant objects, the range of the collected data, the speed of the provision of answers and the vulnerability analysis provided. The VulNet tool provides effective search for vulnerable servers on a large scale by scanning multiple web pages on the same host with different IP addresses.

The VulNet tool is built from four modules presented on Figure 1. The first module is the local Signature database, which stores all the known WordPress plug-ins and different core versions of the WCMS. The second module is the Common Vulnerabilities and Exposures (CVE) database, built up from different public resources available online (CVE databases, Exploit databases, etc.). In the production of these two blocks a specific methodology for scoring the vulnerability was used that indexes each known plug-in or version of the WCMS. The value of a particular index depends on the assessment of the potential threat if the vulnerability is exploited. The CVE database contains 300 identified vulnerabilities of the WordPress core versions and more than 1300 WordPress plug-ins exploits. The index value is assigned based on the developed scoring mechanism that enables easier verification of the potential damage and the vulnerability assessment. The index values for seven known vulnerabilities run from 3 to 9, but the group of all the vulnerability types is scored to 10.

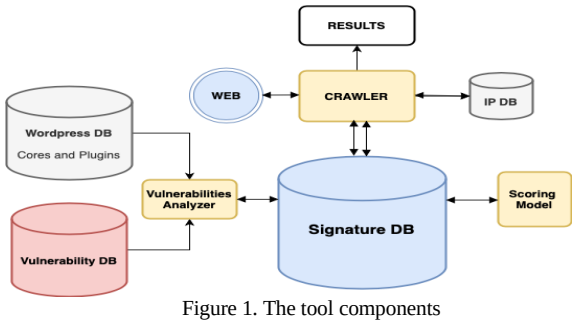

Figure 1. The tool components

The search mechanism is fast as the likelihood of false or repeated access to web site is reduced feedback information received from the crawler and the matching with the data of the temporary set data base that is set up to prevent repeating access to already visited web server. Enabled parallel computation of the code as well reduce the time for scanning. Specific scoring mechanism of insecurity based on the found vulnerability enables a rapid and reliable analysis and assessment. Since there is no list of WordPress sites on the Internet, the first step in the search process is to scan the entire web space and to find

 ISBN: 978-1-61208-880-8

the sites with a WCMS. According to Verisign's data (verysign.com) the web space is enormous, as there are more than 350 million web domains registered on the Internet. Unfortunately, not all DNS (Domain Name Server) zones are accessible (due to private networks) and for that reason a list of root domains is created in the initiation operational phase of the tool. To speed up the process, all the services that allow the use of shortened URLs, and large sites like Facebook, YouTube, and Instagram are removed from the search list as they are not operating any significant device. After identifying the presence of a WCMS the query part of the tool looks for the presence of the file under the name ''robots.txt'', used to verify whether the site allows browsing, meaning that the reviewing of the web applications is legal. The next step is sending queries to the site and collecting information about the content.

Figure 2. The vulnerability of one affected domain.

The tool looks for the web meta tag generator (<meta name ''generator'' content ''WordPress 5.1.1''>), which usually contains information about the version of the plat- form. If the data is missing, then the tool looks for CSS and JS files, (/wp-includes/js/wp-emoji-release.min.js?ver 5.2); this information is provided at the end of the file. The tool then parses the plug-ins (wp-content/plugins/wp-hide-post/public/js/wp-hide-post-public.js?ver2.0.10) to obtain the version applied. If the version of the plug-ins is not avail- able in the CVE database, then the plug-ins are stored in a separate folder for further analysis and a search of exploits. The server vulnerability of the affected domain (e.g., the country top-level domain) is presented in a way that does not show the ownership of the web site type (e.g. Apache/2.2.15 CentOS), and the IP address are also stored and used later to identify the server's location and the country of origin. The scoring assessment for the vulnerability of the website's page is based on two sets of parameters: the first set is used for the risk assessment of the website's core C(s), and the second set is used for a risk assessment of the attached plug-ins P(s). The version of the website's core is first verified and then the tool looks for the potential vulnerability of the core in the signature database. In the case of several identified vulnerabilities in a web page, the score with the highest value is selected for the website core's risk parameter. The same approach is applied to the plug-ins: the first match is found for each of the plug-ins and then the vulnerability with the highest risk parameter is selected for the plug-in's risk value.

The calculated vulnerability-risk score for the web WCMS is calculated as an aggregated score from the score obtained of the web-server core and the highest found vulnerability found among the plug-ins. The scan speed and the answers that provide the data are very important characteristics of any scanning tool. VulNet is capable of scanning 90,000 web pages in 15 minutes. The answering speed is variable, as the response of the WCMS pages depends on how fast the web server is at delivering the responses. Some servers need up to 15 seconds to respond and that timing influences the speed of the data collection. The program's logic is written in the Python programming language.

IV. THE STUDY RESULTS

A. *General findings*

In the first scan with Vulnet 115 million randomly scanned web addresses from around the world (over 194 countries) were accessed. The tool established 126,086,633 links, but some of the visited web servers were found to not be active as the waiting time response of 15 seconds was exited. Among these sites there were 16,274,980 valid WordPress installations and 14,887,047 plug-in installations. In the identified web set, more than 5,018,262 were found to be vulnerable, representing 31% of all the WordPress installations on the Internet, where 2,475,337 had a higher score than 5. A total of 4,356,067 vulnerabilities were detected among the detected plug-ins and 2,795,855 had a score higher than 5. The analysis of the results revealed that there are very vulnerable core versions of WordPress, recent versions of WordPress accounting for over 1 million vulnerable pages tool of found vulnerability in an affected domain. The repeated scan that was implemented six months later showed that the number of identified top-10 vulnerable plug-ins as well the top-10 outdated vulnerable plug-ins were not changed very much, neither in frequency nor in plug-in type. However, some other changes were noticed in the general scan. The status of 12,865,441 websites from the first run and 10,330,577 among them retained the same vulnerability status. There was found lower number of unknown core version and higher version of secure web sites. This transitions from hidden core versions with the implementation of new WP core versions release in 2020 contributed the percentage of the overall security to be higher.

B. *Study results from the inspected European web spaces*

The exploratory study with the Vulnet tool started in the middle of September 2019. The scanning of the

European web space provided a set of 23,131,336 websites, among which 3,738,654 with WordPress applications (16%) The websites belong to the following European countries: Germany (DE), Netherlands (NL), France (FR), Great Britain (GB), Italy (IT), Denmark (DK), Poland (PL), Spain (ES), Sweden (SE), Switzerland (CH), Czech Republic (CZ), Ireland (IE), Finland (FI), Austria (AT), Romania (RO), Belgium (BE), Hungary (HU), Bulgaria (BG), Norway (NO), Slovakia (SK), Estonia (EE), Slovenia (SI), Portugal (PT), Croatia (HR), Lithuania (LV), Luxembourg (LU), Greece (GR), Iceland (IS), Latvia (LT), and Cyprus (CY). The selection of these countries was based on the availability of data regarding the digital development provided from credible sources like the International Telecommunication Union (ITU) (ITU, 2019) and Eurostat (Eurostat, 2019). The number of WordPress sites that were vulnerable in the European sample of 3,738,654 WordPress websites was 1,339,325 and the number of websites without vulnerabilities was 1,187,085. The rest of the websites did not provide vulnerability information as the versions of the core and plug ins were hidden. A website was considered to be secure if no vulnerability was detected in the core and in the attached plug-ins. The percentage of insecure WordPress sites in a particular EU country ranged between 30 and 47 %, with the average of the whole set being 38%.

The macro-analysis of the collected data provided a good insight into how plug-ins and the web core state influence the overall state of a website's vulnerability. The correlation between the plug-ins vulnerability and the overall web site vulnerability was high (r = 0.91). So it concluded that the number of insecure websites primarily depends on the number of insecure plug-ins (at least one) in the web sites. All found insecure-core websites were critically unsafe, as their score was, for the majority, above 5 from maximum score of 10. Plug-ins overall insecurity is the greatest risk for a website to become insecure, similar conclusion was presented in the study presented by Vases & Moore [18] but on much less entries in their sample. The comparison of the level of digital development with the level of web insecurity among was intended to discover whether different parameters that measure the digital economy and social advancement provide an impact on the appearance of higher insecurity. Two indexes were considered Cost of the fixed access to Internet normalized with GNI data (Gross National Income) and the level of Digital skills among country population. Eurostat recognize three levels of DS: low, middle and high. Figure 3 shows the relationship between the percentage of secure and insecure websites in a particular country with indication of the DS levels. DS index higher than 75 is colored in green, orange is used for a DS index between 75 and 50, and the lowest DS country index being below 50 is colored blue.

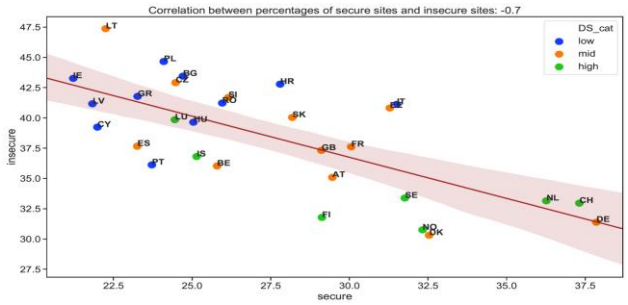

Figure 3. Percentage of secure and insecure sites in a country with different level of Digital skills.

The following countries: Finland, Sweden, Norway, the Netherlands, Denmark, Switzerland, Germany with high digital skills, have high percentage of secure websites and a low percentage of insecure websites are grouped in the bottom-right corner in Fig 3. They have high DS index as well. In the group of countries with low digital skills and a high percentage of insecure websites, the following countries are found: Lithuania, Poland, Bulgaria, Greece, Latvia, Portugal, Ireland, Czech, Romania, Slovenia, Croatia, Hungary and Cyprus. In the group of countries with a middle level of digital skills and a moderate percentage of insecure websites are Belgium, Austria, Slovakia, Great Britain, France, Austria, Iceland, Estonia, and Italy. The findings suggest that high DS contributes to the higher security of the country web space.

The correlation of the fixed-cost access to the Internet normalized with the country's GNI, as shown in Fig. 4, provides another insight into the influential factors affecting the presence of insecurity with the Cost of access to fixed Internet normalized with GNI (the Gross Income of a Country). At a lower fixed-access cost rate, the percentage of insecure websites is also the lowest, and the countries that belong to this group have the highest level or middle DS level, similar to the group in Fig 3. In this group, the following countries can be found: Denmark, Germany, Switzerland, Norway, Finland, Sweden, the Netherlands and Austria. The other group of countries with a much higher cost of fixed access have higher percentage of insecure websites. They are as follows: Hungary, Bulgaria, Italy, Croatia, Slovenia, Poland, Latvia, Romania, Czech Republic, Estonia, Italy, Iceland, Luxembourg and Belgium. They form the middle group. These findings are also in line with the linear dependency between the level of DS and the percentage of insecurity sites (r = -0.68) and imply that a low fixed-access price for the internet is a positive factor for higher security. Both factors have indirect influence on the awareness about the security provision in the web sites.

 ISBN: 978-1-61208-880-8

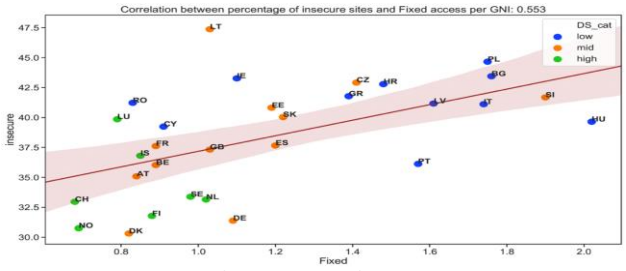

Figure 4. Impact of the Fixed cost for Internet access per GNI

The Vulnet tool was further reshaped as publicly available notification service for better management of the web site vulnerability. Developed platform is offered as a public service available to any individual web owner or web administrator for regular monitoring of the websites and discovering the vulnerabilities. By entering the website's URL and the owner's e-mail address on the platform portal, the platform sends information about this URL's inspection regarding the potential presence of vulnerability. The vulnerabilities of the website are revealed only to the owner on the platform screen or by notification with an e-mail.

V. CONCLUSION AND LIMITATIONS OF THE STUDY

The current known services and tools for measuring the WCMS vulnerabilities lack to provide real insight in the security of the websites operating with WordPress applications [18] as they usually provide only raw data, the address of the accessed server and the associated ports. The effectiveness of these known security tools is low as they do not provide information about the security holes within the web applications supported by the installed plug-ins. The presented tool and the study results show that known vulnerabilities and potential exploits can be found among the great part of the web space on the Internet. The advantage is in the applied scanning approach that allows a fast and reliable overview of almost all accessible web space and enables safe patch fingerprinting to be applied for improving the web security. Evidence that this is happening was notified in 2020 when the security of WordPress site installations rose sharply after the release of the new version of WordPress applications by the WordPress organization, that happen for a first time from 2007 year on. However, the approach and the tool have some limitations. The study was carried on websites with WCMS WordPress applications only. Although they represent the largest part of open-source WCMS installations in the whole web space, other systems like Joomla were not inspected. An additional limitation comes from the collected data sample for the further stud, with sites having in their URL the Top Level Domain of a particular EU country and accessible zone files. The presence of other websites in the country with TLD different from the TLD of that country were not collected as affiliation to the country population was not known. Despite that, the size of the obtained samples contained enough data for the carried exploratory analysis to be credible. The applied and improved scoring system for insecurity follows the approaches in other studies with added vulnerabilities of the plug-ins. Due to the ethical requirements of the applied web scanning, the search for vulnerabilities was limited only to websites with an accessible web core and a displayed core version, which can be considered as another limitation of the study. In case of hidden core version, data were not obtained.

ACKNOWLEDGEMENT

The support of the ARRS under contract P2-0037 is appreciated.

REFRENCES

[1] S. Karumanchi and A. C. Squicciarini, ''A large scale study of Web service vulnerabilities,'' J. Internet Service Inf. Secure, vol. 5, no. 1, pp. 53–69, 2015.

[2] The Hague Security Delta. (Oct. 2018). The Hague Security Delta. Accessed: May 2018. [Online]. Available: https://www.thehague securitydelta.com/news/newsitem/976

[3]W3Techs. Accessed: May 2019. [Online]. Available: https://w3techs.com

[4] M. Hassan, K. Sarker, S. Biswas, and H. Sharif, ''Detection of wordpress content injection vulnerability,'' 2017, arXiv:1711.02447. [Online]. Avail- able: https://arxiv.org/abs/1711.02447

[5] P. Laitinen, ''Vulnerabilities in the wild: Detecting vulnerable Web applica- tions at scale,'' M.S. thesis, Univ. Jyväskylä, Dept. Comput. Sci. Inf. Syst., Jyväskylä, Finland, 2018.

[6] H. Trunde and E. Weippl, ''WordPress security: An analysis based on publicly available exploits,'' in Proc. 17th Int. Conf. Inf. Integr. Web-Based Appl. Services, vol. 81, 2015, pp. 1–7

[7] Uniform Resource Location—Identification of the Web Site by Its Inter- net Address (IP). Accessed: May 2019. [Online]. Available: https://www. w3.org

[8] A. Tundis, W. Mazurczyk, and M. MÜhlhäuser, ''A review of network vulnerabilities scanning tools: Types, capabilities and functioning,'' in Proc. 13th Int. Conf. Availability, Rel. Secur., vol. 65, 2018, pp. 1–10

[9] Z. Durumeric, E. Wustrow, and A. J. Halderma, ''ZMap: Fast Internet-wide scanning and its security applications,'' presented at the 22nd USENIX Secur. Symp. (USENIX Secur.), 2013.

[10] NIST. National Vulnerability Database. Accessed: May 2019. [Online] Available: https://nvd.nist.gov

[11] T. V. Goethem, P. Chen, N. Nikiforakis, L. Desmet, and W. Joosen, ''Large- scale security analysis of the Web: Challenges and findings,'' in Proc. Int. Conf. Trust Trustworthy Comput. New York, NY, USA: Springer-Verlag, 2014, pp. 110–126.

[12] B. Stock, G. Pellegrino, F. Li, M. Backes, and C. Rossow, ''Didn't you hear me?—Towards more successful Web vulnerability notifications,'' in Proc. Netw. Distrib. Syst. Secur. (NDSS) Symp., 2018.

[13] M. Vasek, J. Wadleigh, and T. Moore, ''Hacking is not random: A case- control study of Webserver-compromise risk,'' IEEE Trans. Dependable Secure Comput., vol. 13, no. 2, pp. 206–219, Mar./Apr. 2015.

[14] N. Schagen, K. Koning, H. Bos, and C. Giuffrida, ''Towards automated vulnerability scanning of network servers,'' in Proc. 11th Eur. Workshop Syst. Secur., vol. 5, 2018, pp. 1–6

[15] A. Nappa, Z. M. Rafique, J. Caballero, and G. Gu, ''CyberProbe: Towards Internet-scale active detection of malicious servers,'' in Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS), 2014, pp. 1–15.

[16] H. Kim, T. Kim, and D. Jang, ''An intelligent improvement of Internet- wide scan engine for fast discovery of vulnerable IoT devices,'' Symmetry, vol. 10, no. 5, pp. 151–166, 2018.

 ISBN: 978-1-61208-880-8

[17] F. Li, Z. Durumeric, J. Czyz, M. Karami, D. McCoy, S. Savage, and V. Paxson, ‘‘You’ve got vulnerability: Exploring effective vulnerability notifications,’’ in Proc. 25th USENIX Secur. Symp. (USENIX Secur., 2016, pp. 1033–1050.

[18] M. Vasek and T. Moore, Identifying risk factors for webserver compromise, 18th International conference on financial cryptography and data security, Springer, LNCS, (V) 8437, 2014, pp.326-345