Penetration Testing Procedure using Machine Learning

1st Reevan Seelen Jagamogan
Razak Faculty of Technology and Informatics
Universiti Teknologi Malaysia
Kuala Lumpur, Malaysia
reevanseelen@graduate.utm.my

2nd Saiful Adli Ismail
Razak Faculty of Technology and Informatics
Universiti Teknologi Malaysia
Kuala Lumpur, Malaysia
saifuladli@utm.my

3rd Noor Hafizah Hassan
Razak Faculty of Technology and Informatics
Universiti Teknologi Malaysia
Kuala Lumpur, Malaysia
noorhafizah.kl@utm.my

4th Hafiza Abas
Razak Faculty of Technology and Informatics
Universiti Teknologi Malaysia
Kuala Lumpur, Malaysia
hafiza.kl@utm.my

2022 4th International Conference on Smart Sensors and Application (ICSSA) | 978-1-6654-9981-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICSSA54161.2022.9870951

***Abstract*—The main aim of this study is to determine the effectiveness of a penetration testing tool, GyoiThon as a Machine Learning tool by conducting penetration tests on websites, including the websites that have Content Management System (CMS) frameworks, to identify their vulnerabilities and assess the effectiveness of GyoiThon features in penetration testing. This experiment will determine how well the automation framework is executed for penetration testing. This research hypothesized, that if the feature of a penetration tool consists of any form of Machine Learning algorithm, the more effective the feature can search for more vulnerabilities. To achieve the aim of this paper, an experiment was conducted to evaluate the effectiveness of the two features of GyoiThon, the Default and Machine Learning modes. In the end, it was revealed that the Machine Learning mode of GyoiThon discovered more types of vulnerabilities than using the Default mode of GyoiThon, proving the hypothesis to be right.**

***Index Terms*—CMS, CVE, content management system, GyoiThon, penetration testing, Machine Learning, Naïve Bayes, vulnerabilities, web application**

I. Introduction

As technology advances for the greater good, adversaries will continue to try to compromise systems by hacking into them and stealing or destroying data or software. Penetration testing, often known as pen-testing, is a method of studying adversary behaviour and strategies to compromise a system [1]. There were approaches to penetration testing using various techniques, especially the ones using Machine Learning like the GyoiThon tool which has a Machine Learning feature that uses the Naïve Bayes algorithm to scan for vulnerabilities of the target URL [2].

This paper will document a penetration testing procedure that was carried out using GyoiThon as the penetration testing tool, to compare the effectiveness of searching for vulnerabilities between its default and Machine Learning feature. Section 2 describes the related works regarding using Machine Learning in the penetration testing procedure and the most common CVEs found in web applications. Section 3 explains the methodology used and the experimental setup of the penetration testing procedure. Section 4 outlines the results obtained from the experiment and highlights the findings of the experiment. Lastly, section 5 concludes the paper by outlining the limitations of the research project and the future work plan that could compensate those limitations.

II. Related Works

This section explains the past works involving researching penetration testing approaches using Machine Learning as the means to scan for vulnerabilities of the target machine or web services or websites. This section also includes the findings of the most common Common Vulnerabilities and Exposures (CVEs) found by previous researchers in web applications.

A. Penetration Testing Procedures using Machine Learning

The first tool is the HARMer tool, which was developed to launch simulated attacks for penetration testing purposes and to test the resilience of the system [3]. The tool includes features such as the three metrics-based attack planning: path-based metrics, composite metrics and the atom metrics. The researchers tested this tool on networks like the Amazon Web Services (AWS) networks to obtain realistic results when the networks are being attacked. In a nutshell, the tool was determined to be useful for penetration testers, especially the red team and also includes a cyber-defence assessment tool to evaluate the vulnerability status of the target system [4].

The next tool is the FUSE tool proposed by researchers to detect Unrestricted File Upload (UFU) and Unrestricted Executable File Upload (UEFU) vulnerabilities from web applications [5]. FUSE can generate upload requests to exploit the UEFU and UFU flaws. The authors also used FUSE to overcome two challenges: bypassing web-content filtering checks in web applications and preserving the execution semantics of uploaded files. The workaround to these issues is to 'mutate' the file uploads to overcome the challenges. From the

evaluated web apps, the author was able to find 30 undisclosed UEFU vulnerabilities and 15 Common Vulnerabilities and Exposures (CVEs) [4].

The final tool is the Intelligent Automated Penetration Testing System (IAPTS), which was proposed in two papers by the same authors [6] [7]. PEGASUS, PERSEUS, and the Generalised Incremental Pruning (GIP) algorithms are part of this sophisticated system that leverages Reinforcement Learning. The proposed system is modelled based on the Partially Observed Markov Decision Process (POMDP). The goal of this method is to conserve both time and human resources. The IAPTS works in tandem with the Penetration Testing module to collect data, learn from pen-testing experiences, and replicate tests in comparable scenarios [4].

Overall, the HARMer tool was developed to launch automated simulated attacks, which means it is specifically designed for professional red team attackers to find the vulnerabilities. It is unlike the FUSE and IAPTS tools that were designed not only to launch attacks but also to detect vulnerabilities according to the reported vulnerabilities like the ones in the National Vulnerability Database (NVD) [8], where the CVEs are listed.

B. Most Common CVEs found in Web Applications

Based on the findings as shown in Table I, the most mentioned vulnerabilities are SQL Injections and Cross-site scripting (XSS). According to one of the authors [9], SQL injection (SQLi) attacks pose a threat to web applications, where the adversaries will inject malicious SQL queries to compromise the web applications' databases. The author introduced a tool called SQL Block to identify the procedures that are used across the web scripts in the database, record the queries of the database procedures in training mode, assemble a database-access profile based on the information gathered earlier, and protects the web application from SQL injection attacks based on the information stored in the database-access profile. Overall, it is a tool that trains itself by learning the procedures used in the web applications databases and protects it from attacks that use similar queries to extract data.

Cross-site scripting (XSS) attack is similar to SQL injection whereby the adversary injects malicious codes into the system but on the client-side of the web application instead of the database like SQLi [10]. The author of the paper [10], proposed a grey box penetration testing tool, the XSS analyzer that has four steps, database traffic interception, payload identification exploit, recursive output parsing, and payload encoding verification. This tool is useful to detect XSS vulnerabilities in web applications.

The common content management system mentioned are WordPress, Joomla and Drupal. However, some papers did not specify the target web application because some experiments involve experimenting with a plethora of websites, like the paper [11], to study the usage of Sub-resource Integrity (SRI) and how the developers understand or consider it in web development through handing out surveys and it was revealed that SRI was not taken into consideration in web development, even though the developers were aware of its existence.

III. Methodology

This section explains the methodology used for the overall project and the experimental procedure for penetration testing using GyoiThon, together with a brief explanation of how GyoiThon works.

A. Project Methodology

The methodology used for this project is the exploratory methodology, which involves the investigation of a research question that was not 'explored' or studied in depth [19] [20]. This methodology was normally used in the social science field to study a new or niche topic that was obscure in the field of research [21]. In this project, the research question is 'How effective is the GyoiThon tool in detecting vulnerabilities?'. A working hypothesis is also needed for this kind of methodology because it strengthens the foundation, direction and consistency throughout the research process. The results of this experiment could either prove the hypothesis to be right wholly or partially, or it might prove otherwise [22]. Therefore this experiment hypothesized that if the feature of a penetration tool consists of any form of Machine Learning algorithm, the more effective the feature can search for more vulnerabilities compared to features with no form of Machine Learning algorithm.

The reason why this methodology is chosen is that no one has experimented on GyoiThon as a penetration testing tool other than the developer of the GyoiThon [2]. Another reason that this methodology is chosen is that the data can only be gathered only after the experiment is conducted using GyoiThon on target websites. Finally, this methodology has no preset steps, which allows flexibility in making new procedures, as long they are relevant in finding the solution to the research question.

B. GyoiThon

GyoiThon is a machine learning-based penetration testing tool developed by Masuya Masafumi that can use other penetration testing tools like Scrapy to identify the vulnerabilities found on the target websites [2]. Based on the learning data, GyoiThon recognises the software present on the web server like the operating system, middleware, framework, type of CMS and many more. According to the documentation of the tool on GitHub [23], GyoiThon has nine functions or modes of usage, excluding the default mode, that can be selected depending on the kind of penetration testing procedure that can be done.

The aforementioned processes are carried out automatically by GyoiThon. The Default mode of GyoiThon involves the following processes [23]:

i Gathering HTTP responses from target URLs by using the Web crawling feature.

ii Identifying product or version using string pattern matching.

TABLE I
COMMON VULNERABILITIES AND EXPOSURES RESEARCH MATRIX

Authors	Aim/Outcome of Research	Target System/CMS	Vulnerabilities Found
Cernica, Ionut Cosmin Popescu, Nirvana Tiganoaia, Bogdan [12]	The authors performed penetration testing on WordPress sites to highlight the mass usage of vulnerable backup plugins that can cause data leakage. The results were concerning as over 160,000 WordPress sites are using the vulnerable plugins highlighted in the paper.	WordPress Backup Plugins: • All-in-one-wp-migration • BackWPup • WP-DB Manager • BulletProofSecurity • Xcloner	• Weak cryptography algorithms • Path Traversal vulnerability • Can access CMS configuration files like '.htaccess' which has the metadata and control over the CMS site
Chapuis, Bertil Omolola, Olamide Cherubini, Mauro Humbert, Mathias Huguenin, Kévin [11]	The authors conducted large-scale research on the usage of Subresource Integrity (SRI) by analyzing a massive crawl of 3 billion URLs for 3 ½ years. From the results, it was determined the usage of SRIs is average and the web developers have comprehensive knowledge about SRI but neglect the essential aspect of their usage.	WordPress and 3 billion other URLs	• Buggy web application • Human error (due to the manual integration of SRI and being error-prone)
Dresen, Christian Ising, Fabian Poddebniak, Damian Kappert, Tobias Holz, Thorsten Schinzel, Sebastian [13]	The authors experimented with using CORSICA, by first analyzing the loopholes in the Same Origin Policy (SOP) that prevents access to HTTP responses in other origins, then using CORSICA to identify 74% of the tested web services running of various IoT devices and 4 content management systems' version numbers.	• WordPress • Joomla • Drupal • TYPO3	• Remote Code Execution vulnerability • SQL Injection vulnerability • XSS vulnerability • Cross-site request forgery (CSRF) vulnerability
Gałazkiewicz, Adam Wójtowicz, Adam [14]	The authors proposed a prototype to use Multiparty Computation (MPC) protocol in Enterprise Content Management (ECM) Systems which can be integrated easily. The MPC involves multiple processing units performing computations without having to disclose any data.	Enterprise Content Management (ECM) Systems	Weak Cryptography Protocols
Huang, Jin Zhang, Junjie Liu, Jialun Li, Chuang Dai, Rui [15]	The author introduced UFuzzer, a tool that can detect vulnerabilities in PHP server-side web applications and tested it, to discover 31 unknown vulnerabilities and 5 CVEs.	WordPress and Joomla together with their plugins	Unrestricted File Upload (UFU) vulnerability
Jahanshahi, Rasoul Doupé, Adam Egele, Manuel [9]	The authors developed a plugin tool, SQLBlock which limits malicious access to PHP web applications' databases and it discovered 11 vulnerabilities from the tested web applications.	• WordPress • Joomla • Drupal • Magneto	SQL Injection Vulnerabilities: • Tautologies • Piggy-backed Queries • Inference • Alternate Encodings • Stored Procedures • Union Query
Leithner, Manuel Garn, Bernhard Simos, Dimitris E [16]	The authors proposed a black-box testing tool, HYDRA, to perform injection attacks on web applications and it turns out to evade faulty filters that detected multiple Injection vulnerabilities.	Web Applications	Injection vulnerabilities: • XSS vulnerability • SQL Injection
Nguyen, Van-Linh Lin, Po-Ching Hwang, Ren-Hung [17]	The authors described how adversaries can earn money from exploiting vulnerable websites and suggested solutions to mitigate the attacks: Install attack prevention mechanisms, Perform cleanup tasks of malicious codes and Reinforce server security	• Joomla • Cryptocurrency sites • Online Banking sites	• SQL Injection • Cryptojacking vulnerability
Steinhauser, Antonín Tůma, Petr [10]	The authors proposed an XSS analysis approach to detect XSS vulnerabilities that are either stored or context-sensitive and managed to report 71 vulnerabilities that were found on 8 tested CMS.	• Joomla • Orchard CMS • SuiteCRM • OpenEMR • Jeesite • Mezzanine	XSS vulnerability
Wang, Meng Jung, Chijung Ahad, Ali Kwon, Yonghwi [18]	The authors introduced a tool, Spinner, which is a randomization-based input injection prevention method that prevents input injection attacks.	WordPress, Elementor and 25 more web applications	• SQL Injection • Shell Command Injection • XML external entity (XXE) injection

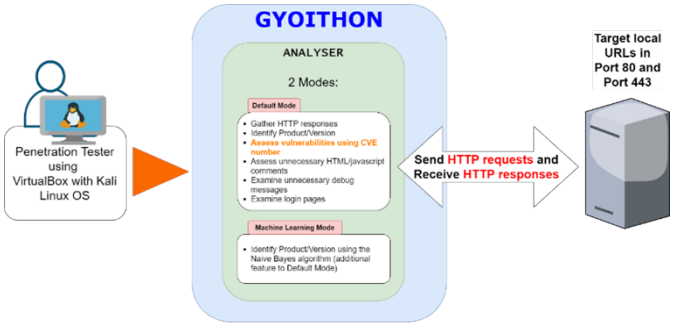

Fig. 1. GyoiThon Penetration Testing Procedure.

iii Identifying vulnerabilities based on the reported vulnerabilities in the National Vulnerability Database (NVD), by their CVE numbers.
iv Locating unnecessary HTML or JavaScript comments.
v Locating unnecessary debug messages.
vi Login pages assessment.

C. Experimental Procedure

Fig. 1 shows the procedure model designed based on the GyoiThon model developed by Masafumi [23], to conduct vulnerability scanning on the target websites using the GyoiThon information gathering tool. The penetration testing is conducted using the Kali Linux operating system with Virtualbox as the virtualization software.

For this experiment, GyoiThon will be used to identify the vulnerabilities in Port 80 and Port 443. Port 80 is used for Hypertext Transfer Protocol (HTTP) connection by default to allow data exchange between the web browser and the Internet. Port 443 is similar to Port 80 but it involves encrypted data exchange instead of the secured version of HTTP, the HTTPS and is used by most of the websites currently on the Internet [24]. They are the essential ports that needed to be assessed to inspect for vulnerabilities that could leak or damage the data exchange channel.

The target websites are set up from the OWASP Broken Applications server via Virtualbox, which was downloaded from the official website, Open Web Application Security Project (OWASP) [25]. Therefore, the target websites used for this experiment are localhost, as shown in Table II, the first three being CMS websites. The local IP address is censored with ”[localhost]” in Table II, to preserve the confidentiality of the machine used for this experiment.

IV. Results and Findings

This section explains the results obtained from the generated reports of GyoiThon and lists out the notable vulnerabilities labelled by GyoiThon.

A. Results

The results of the experiment are tabulated into two categories, the first one being the results gathered after using the Default mode of GyoiThon and the second one being the results gathered after using the ML mode of GyoiThon. The

TABLE II
List of Target Websites tested with GyoiThon

Target Websites	URLs
Broken WordPress	http://[localhost]/wordpress/
Broken Joomla	http://[localhost]/joomla/
TikiWiki	http://[localhost]/tikiwiki/tiki-index.php
Buggy Web Application (bWAPP)	http://[localhost]/bWAPP/login.php
Damn Vulnerable Web Application (DVWA)	http://[localhost]/dvwa/login.php
Ghost	http://[localhost]/ghost/
OWASP CSRF Guard Test Application	http://[localhost]/OWASP-CSRFGuard-Test-Application.html
OWASP Vicnum	http://[localhost]/vicnum/
Simple ASP.NET Forms	http://[localhost]/mono/
Wacko Picko	http://[localhost]/WackoPicko/

categories for each mode refer to the ports, Port 80 or Port 443, of the target websites being scanned. The numbers refer to the number of vulnerabilities found in GyoiThon. Most of the vulnerabilities were found but are not labelled with a CVE number because they are not reported in the National Vulnerability Database (NVD) [8], which is why the generated report of GyoiThon labelled it as ’Cannot Search’ as shown in the second last row in Table III.

The frequency of the vulnerabilities is categorized to reflect just the three CMS websites, WordPress, Joomla, TikiWiki, and all the 10 tested websites including the three CMS websites, as shown in Table III.

B. Discussion of Findings

According to the results in Table III, there are still many vulnerabilities that cannot be identified. This is due to those vulnerabilities are being not reported in the National Vulnerability Database (NVD) [8]. There is a major difference in vulnerabilities found using the Default mode and Machine Learning mode of GyoiThon. In default mode, there is a significant difference in the frequency of vulnerabilities found between Port 80 and Port 443. Port 80 has found 14 types of vulnerabilities while Port 443 has only 8 types of vulnerabilities being found. Port 80 has more vulnerabilities because the websites tested are deployed through ’localhost’, which means it does not have the latest version of hypertext transfer protocol, which is HTTPS, while Port 443 allows data transmission over a secured network. However, there are still some vulnerabilities discovered from Port 443, despite the websites only using HTTP. The difference between the frequency of vulnerabilities found between Port 80 and Port 443 is around 80.2%.

A similar pattern of results was found in machine learning mode, where there is a significant difference in the frequency of vulnerabilities found from Port 80 and Port 443. Port 443 has the exact pattern, where the frequency of the types of vulnerabilities found is equal, excluding the vulnerabilities that cannot be identified, labelled as ’Cannot Search’. However, the frequency of vulnerabilities found bumps up to 8.3% when

TABLE III
FREQUENCY OF VULNERABILITIES FOUND FROM THE EXPERIMENT

Vulnerabilities identified by its CVE number	Default Mode				Machine Learning (ML) mode			
	Port 80		Port 443		Port 80		Port 443	
	CMS Websites	*All Websites*	*CMS Websites*	*All Websites*	*CMS Websites*	*All Websites*	*CMS Websites*	*All Websites*
CVE-2006-5877	54	101	7	10	54	101	3	10
CVE-2007-2834	54	101	7	10	54	101	3	10
CVE-2008-1459	-	-	-	-	24	49	-	-
CVE-2008-4668	-	-	-	-	24	49	-	-
CVE-2008-5053	-	-	-	-	24	49	-	-
CVE-2010-0425	54	101	7	10	54	101	3	10
CVE-2010-0834	54	101	7	10	54	101	3	10
CVE-2010-2068	54	101	7	10	54	101	3	10
CVE-2010-4156	54	101	7	10	54	101	3	10
CVE-2011-0754	54	101	7	10	54	101	3	10
CVE-2011-5254	48	48	-	-	72	72	-	-
CVE-2012-2376	54	101	7	10	54	101	3	10
CVE-2012-3575	48	48	-	-	72	72	-	-
CVE-2015-4642	1	26	-	-	1	26	-	-
CVE-2016-7405	1	26	-	-	1	26	-	-
CVE-2016-8670	1	26	-	-	1	26	-	-
CVE-2020-26596	48	48	-	-	72	72	-	-
Unidentified vulnerabilities (*Cannot search*)	187	334	9	30	187	361	9	30
TOTAL	766	1,364	65	150	910	1,610	33	150

compared with the results from default mode. Furthermore, there are three more vulnerabilities were identified in Port 80 in ML mode.

1) Port 80 Vulnerabilities: As mentioned earlier, three more vulnerabilities were found in this port using ML mode, which is listed as the following [8]:

i **CVE-2008-1459** - A type of SQLi vulnerability
ii **CVE-2008-4668** - Directory traversal vulnerability
iii **CVE-2008-5053** - PHP remote file inclusion vulnerability

Furthermore, in ML mode, GyoiThon was able to highlight the three most common vulnerabilities found from the tested CMS websites, with their explanation according to the NVD [8]:

i **CVE2011-5254** – It is an unspecified vulnerability in the Connections plugins before 0.7.1.6 in WordPress (according to the CVE database) that has unknown attack and impact vectors
ii **CVE2012-3575** – Unrestricted File Upload (UFU) vulnerability
iii **CVE2020-26596** – Specified vulnerability for the Elementor Pro plugin that has Dynamic OOO widget through for WordPress.

2) Port 443 Vulnerabilities: The eight vulnerabilities found in the Default mode have the same pattern of the same types of vulnerabilities in the ML mode of GyoiThon. The vulnerabilities found were as follows, with its summary according to the NVD [8]:

i **CVE2006-5877** – Extension vulnerability due to improper handling of large, encrypted email file attachments that gives way for Denial-of-Service (DoS) vulnerability
ii **CVE2007-2834** – Vulnerability that allows remote adversaries to execute malicious code via a TIFF file.
iii **CVE2010-0425** – Vulnerability in Apache server that allows remote attackers to execute arbitrary codes.
iv **CVE2010-0834** – Lack of authentication for package installation that allows servers to be archived remotely and man-in-the-middle (MitM) adversaries to execute arbitrary code via a crafted package.
v **CVE2010-2068** – Vulnerability in Apache HTTP server where it does not properly detect timeouts, allowing remote attackers to exploit the HTTP request to obtain a potentially sensitive response intended for a different client.
vi **CVE2010-4156** – Vulnerability that allows adversaries to extract potentially sensitive data using a large value by executing the 'mb_strcut' function using PHP.
vii **CVE2011-0754** – A Symlink vulnerability that allows adversaries to place a link that redirects the victims in uploading or downloading a file from the malicious source.
viii **CVE2012-2376** - Buffer overflow vulnerability that allows remote adversaries to execute arbitrary code via crafted arguments that trigger incorrect handling.

Overall, there are still about 21.73% of the vulnerabilities found by GyoiThon are not reported in the National Vulnerability Database (NVD). This shows that this tool is still dependent on the secondary source when it comes to identifying the nature of new types of vulnerabilities. It still needs more room for improvement but it can be used as a learning tool to discover the common types of vulnerabilities found on the websites, or it can be used as one of but not

as the primary tool by professional penetration testers to find out the common vulnerabilities reported in the NVD. Since the Machine Learning mode of GyoiThon managed to detect three more vulnerabilities that the Default mode could not detect, the hypothesis stating that the Machine Learning algorithm works effectively has been proven to be true.

V. Conclusion

While the experiment was conducted successfully, there were some limitations faced in the beginning and during the completion of the project. Firstly, real websites were not used as targets for these experiments, to avoid any risks of being taken legal actions by the owner of the websites, the localhost websites are used as targets instead for this experiment. Another limitation is that GyoiThon is not tested to its full potential. GyoiThon has 9 modes to be utilized for penetration testing, but only 2 of them are tested for this penetration testing process. Due to brevity, only the Default and Machine Learning (ML) modes were evaluated and compared to see whether the ML algorithm (Naïve Bayes) used is more effective than using normal penetration testing tools. To improve this research in the future, the first step is to make the results to be more specific and realistic. To do that, real websites will be used as targets for the experiment. If the penetration testing involves a much more intrusive procedure, permissions will be requested to ensure that the owner of the target website acknowledges the penetration testing procedure that will be conducted at the specified time. To evaluate GyoiThon's full potential in penetration testing, other tools like the proposed tools in Section 2, will be included in the experiment to avoid bias during the evaluation process and obtain realistic findings to determine the maturity of machine learning algorithms in penetration testing. Overall, machine learning is a revolutionary solution when it comes to performing certain tedious processes, especially some of the penetration testing processes to save and allocate time for more crucial penetration testing processes. Hopefully, with this experiment, we can take one step closer to utilising machine learning to its fullest potential.

Acknowledgment

This research has received financial support from the Ministry of Higher Education under FRGS. Registration Proposal No: FRGS/1/2021/ICT07/UTM/02/2 & UTM.

References

[1] J. Firch. (2020, Oct) What Are The Different Types Of Penetration Testing? PurpleSec. [Online]. Available: https://purplesec.us/types-penetration-testing/

[2] M. Masafumi, I. Takaesu, T. Terada, T. Kudo, and T. Yoneyama, "GyoiThon - Next generation penetration test tool," *Black Hat Asia 2018 Arsenal*, pp. 2–3, 2018.

[3] S. Y. Enoch, Z. Huang, C. Y. Moon, D. Lee, M. K. Ahn, and D. S. Kim, "HARMer: Cyber-Attacks Automation and Evaluation," *IEEE Access*, vol. 8, pp. 129 397–129 414, 2020.

[4] R. S. Jagamogan, S. A. Ismail, N. Hafizah, and H. Hafiza Abas, "A Review: Penetration Testing Approaches on Content Management System (CMS)," in *2021 7th Int. Conf. Res. Innov. Inf. Syst.* IEEE, oct 2021, pp. 1–6.

[5] T. Lee, S. Wi, S. Lee, and S. Son, "FUSE: Finding File Upload Bugs via Penetration Testing," in *Proc. 2020 Netw. Distrib. Syst. Secur. Symp.*, no. April. Reston, VA: Internet Society, 2020.

[6] M. C. Ghanem and T. M. Chen, "Reinforcement Learning for Intelligent Penetration Testing," in *2018 Second World Conf. Smart Trends Syst. Secur. Sustain.*, vol. 11, no. 1. IEEE, oct 2018, pp. 185–192.

[7] ——, "Reinforcement learning for efficient network penetration testing," *Information*, vol. 11, no. 1, 2020.

[8] National Institute of Standards and Technology. Search Vulnerability Database. [Online]. Available: https://nvd.nist.gov/vuln/search

[9] R. Jahanshahi, A. Doupé, and M. Egele, "You Shall Not Pass: Mitigating SQL Injection Attacks on Legacy Web Applications," in *Proc. 15th ACM Asia Conf. Comput. Commun. Secur.*, ser. ASIA CCS '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 445–457.

[10] A. Steinhauser and P. Tůma, "Database Traffic Interception for Graybox Detection of Stored and Context-sensitive XSS," *Digit. Threat. Res. Pract.*, vol. 1, no. 3, pp. 1–23, sep 2020.

[11] B. Chapuis, O. Omolola, M. Cherubini, M. Humbert, and K. Huguenin, "An Empirical Study of the Use of Integrity Verification Mechanisms for Web Subresources," in *Proc. Web Conf. 2020*, no. May. New York, NY, USA: ACM, apr 2020, pp. 34–45.

[12] I. C. Cernica, N. Popescu, and B. Tiganoaia, "Security evaluation of wordpress backup plugins," *Proc. - 2019 22nd Int. Conf. Control Syst. Comput. Sci. CSCS 2019*, pp. 312–316, 2019.

[13] C. Dresen, F. Ising, D. Poddebniak, T. Kappert, T. Holz, and S. Schinzel, "CORSICA: Cross-Origin Web Service Identification," in *Proc. 15th ACM Asia Conf. Comput. Commun. Secur.* New York, NY, USA: ACM, oct 2020, pp. 409–419.

[14] A. Gałązkiewicz and A. Wójtowicz, "Multiparty Computation in Practice: Increasing Security of Documents in Enterprise Content Management Systems," in *2020 4th Int. Conf. Softw. E-bus.*, ser. ICSEB 2020. New York, NY, USA: ACM, dec 2020, pp. 1–6.

[15] J. Huang, J. Zhang, J. Liu, C. Li, and R. Dai, *UFuzzer: Lightweight Detection of PHP-Based Unrestricted File Upload Vulnerabilities Via Static-Fuzzing Co-Analysis*. New York, NY, USA: ACM, oct 2021, pp. 78–90. [Online]. Available: https://dl.acm.org/doi/10.1145/3471621.3471859

[16] M. Leithner, B. Garn, and D. E. Simos, "HYDRA: Feedback-driven black-box exploitation of injection vulnerabilities," *Inf. Softw. Technol.*, vol. 140, p. 106703, dec 2021.

[17] V.-L. Nguyen, P.-C. Lin, and R.-H. Hwang, "Web attacks: defeating monetisation attempts," *Netw. Secur.*, vol. 2019, no. 5, pp. 11–19, 2019.

[18] M. Wang, C. Jung, A. Ahad, and Y. Kwon, "Spinner: Automated Dynamic Command Subsystem Perturbation," in *Proc. 2021 ACM SIGSAC Conf. Comput. Commun. Secur.*, ser. CCS '21. New York, NY, USA: ACM, nov 2021, pp. 1839–1860.

[19] T. George. (2021, Dec) A guide to exploratory research. Scribbr. [Online]. Available: https://www.scribbr.com/methodology/exploratory-research/: :text=Exploratory

[20] R. Stebbins, *Exploratory Research in the Social Sciences*. 2455 Teller Road, Thousand Oaks California 91320 United States of America: SAGE Publications, Inc., 2001. [Online]. Available: https://methods.sagepub.com/book/exploratory-research-in-the-social-sciences http://methods.sagepub.com/book/exploratory-research-in-the-social-sciences

[21] M. Casula, N. Rangarajan, and P. Shields, "The potential of working hypotheses for deductive exploratory research," *Qual. Quant.*, vol. 55, no. 5, pp. 1703–1725, oct 2021.

[22] Gyoisamurai. (2019) GyoiThon: GyoiThon is a growing penetration test tool using Machine Learning. [Online]. Available: https://github.com/gyoisamurai/GyoiThon

[23] S. A. Baset and H. G. Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol," in *Proc. IEEE INFOCOM 2006. 25TH IEEE Int. Conf. Comput. Commun.*, vol. 00, no. c. IEEE, 2006, pp. 1–11.

[24] OWASP Broken Web Applications. OWASP. [Online]. Available: https://purplesec.us/types-penetration-testing/