International Journal of Informatics and Information Systems ISSN 2579-7069
Vol. 6, No. 2, March 2023, pp. 56-65

# Exploring the Effectiveness of Web Crawlers in Detecting Security Vulnerabilities in Computer Software Applications

Biao Wan [1,*], Chunmei Xu [1], Jahoon Koo [2]

[1] Wuhan Polytechnic, Wuhan 430070, Hubei, China
[2] Department of Computer and Information Security, Sejong University, Seoul 05006, Korea;
[1] wanbia@21cn.comr*
* corresponding author

(Received December 29, 2022 Revised January 27, 2023 Accepted February 11, 2023, Available online March 23, 2023)

**Abstract**

With the rapid development of the Internet, the World Wide Web has become a carrier of a large amount of information. In order to effectively extract and use this information, web crawlers that crawl various web resources have emerged. The interconnectedness, openness, and interactivity of information in the World Wide Web bring great convenience for information sharing to the society and they also bring many security risks. To protect resource information, computer software security vulnerabilities have become the focus of attention. This article is based on the method of computer software security detection under a web crawler that simply analyzes the basic concepts of computer software security detection and analyzes the precautions in the process of security detection. Finally, combined with the computer software security vulnerability problems in the web crawler environment, its security detection technology Application for further analysis

*Keywords:* Web Crawler, Software, Security Breach, Detection Technology

## 1. Introduction

The rapid development of computer software technology has significantly transformed people's lives and work. With the widespread use of computers, people can now perform various tasks conveniently and efficiently. However, it has also brought about great risks to people's privacy and property [1]. Computer software vulnerabilities refer to defects or weaknesses in computer software that can be exploited by unauthorized individuals to gain access to a computer system. Such unauthorized access can lead to information leakage, system damage, and other security breaches that threaten people's privacy and property [2].

To mitigate such risks, it is important to use and improve computer software security vulnerability detection technology. The detection of software vulnerabilities is crucial in enhancing the protection capabilities of computer systems [3]. Regular detection of vulnerabilities in software can help identify potential weaknesses and prevent unauthorized access to the system. Additionally, detecting and addressing software vulnerabilities can help to prevent potential threats and reduce the risk of damage to the system [4].

In order to improve the security of computer software, it is important to have ongoing discussions around the latest advancements in vulnerability detection technology [5]. This can include evaluating and identifying new software vulnerabilities, as well as developing new tools and techniques to address such vulnerabilities. Moreover, it is essential to ensure that individuals and organizations have the necessary knowledge and skills to detect and address software vulnerabilities effectively.

One approach to detecting software vulnerabilities is through the use of web crawlers, which are software programs that automatically search the internet for specific information [6]. Figure 1 shows the basic flow of a web crawler. These tools can be used to detect security vulnerabilities in web applications by identifying potential security threats and analyzing website data to detect any anomalies or patterns that may indicate a security breach. By utilizing such technology, organizations can better protect their computer systems and minimize the risk of data breaches and other security incidents.

In conclusion, the popularity of computer software has brought convenience to people's lives, but it has also brought great risks to their privacy and property. To mitigate these risks, it is crucial to have ongoing discussions around computer software security vulnerability detection technology. The regular detection of software vulnerabilities, the development of new tools and techniques to address such vulnerabilities, and ensuring that individuals and organizations have the necessary knowledge and skills to detect and address software vulnerabilities effectively are all essential components of ensuring the security of computer systems.

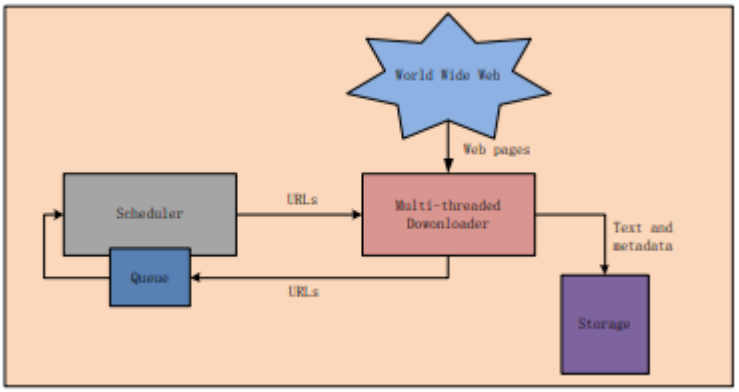


**Fig. 1.** The basic flow of a web crawler

In recent years, cyberattacks and data breaches have become increasingly common, highlighting the importance of computer software security vulnerability detection [7-9]. Detecting and fixing security vulnerabilities before they can be exploited by hackers is essential for protecting organizations and their customers from potential harm. This is where web crawlers come in - as automated tools that systematically browse websites and collect data for analysis, they have emerged as a powerful tool for detecting security vulnerabilities in computer software. Web crawlers can scan and analyze software systems for potential vulnerabilities, making it easier for security professionals to detect and address potential security issues [10]. This paper aims to explore the application of web crawlers in the analysis of computer software security vulnerability detection, and to provide insights into the potential of web crawlers as a tool for improving software security.

The paper will begin by discussing the significance of computer software security vulnerability detection in today's world. With the rise of cyber threats and data breaches, detecting and addressing security vulnerabilities has become increasingly important for organizations of all sizes. This paper will provide an overview of the importance of security vulnerability detection and the challenges faced by security professionals in this area. After discussing the significance of security vulnerability detection, the paper will provide an overview of web crawlers and their application in the field of computer software security vulnerability detection. Web crawlers are automated tools that can scan and analyze software systems for potential vulnerabilities. By systematically browsing software systems, web crawlers can identify potential security issues that may have otherwise gone undetected. The paper will also discuss the methodology used for the analysis, including the software systems tested, the web crawler used, and the criteria for identifying security vulnerabilities. This will provide readers with an understanding of the methodology used in the analysis and the factors that were considered when identifying potential security vulnerabilities.

Finally, the paper will present the findings of the analysis and discuss their implications for software developers, security professionals, and policymakers. The findings of this study will provide insights into the potential of web crawlers as a tool for improving software security and will help guide the development of more secure software systems in the future. In conclusion, this paper highlights the significance of computer software security vulnerability detection and the potential of web crawlers as a tool for improving software security. As cyber threats continue to

evolve, it is essential for organizations to take a proactive approach to security vulnerability detection and prevention. The findings of this study provide valuable insights for organizations, security professionals, and policymakers, and can help guide the development of more secure software systems in the future.

## 2. Literature Review

Computer software security vulnerabilities can be understood as weaknesses and shortcomings. Many factors can cause security vulnerabilities, such as incomplete software development considerations, programmer errors, etc. Generally, computer software security vulnerabilities have the following aspects [11-14].

### 2.1. Logical Errors

Logical errors can lead to system crashes, malfunctioning software, and inaccurate or incomplete data processing. These problems can result in significant financial losses for companies, as well as damage to their reputation. In the context of Kinerja Karyawan di Laksana Baru Swalayan Majenang, logical errors in the software used for employee performance evaluation could lead to inaccurate assessments of employee performance [15]. To prevent logical errors in computer software, it is essential to ensure that the programming staff is well-trained and has a thorough understanding of the programming language and coding practices. This can be achieved through regular training programs, workshops, and refresher courses. Additionally, implementing quality control measures such as code reviews, testing, and debugging can also help to identify and correct logical errors in software [16].

In the case of Kinerja Karyawan di Laksana Baru Swalayan Majenang, it is important to ensure that the software used for employee performance evaluation is well-designed and thoroughly tested to prevent logical errors that could lead to inaccurate assessments. It is also crucial to ensure that employees are properly trained on how to use the software and that there is a system in place to monitor and address any errors or issues that arise. Overall, logical errors in computer software can have significant consequences for companies, particularly in terms of financial losses and damage to reputation. It is therefore essential for companies to take proactive measures to prevent and address logical errors in their software, particularly when it comes to sensitive areas such as employee performance evaluation. By implementing quality control measures and providing regular training and support, companies can reduce the risk of logical errors and ensure the reliability and accuracy of their computer software.

### 2.2. Environmental Impact

Computer security is a critical issue in today's digital age. With the increasing reliance on technology, security breaches and cyber-attacks are becoming more frequent and sophisticated. One of the main causes of security vulnerabilities is the computer software environment. A flaw in software code can create a vulnerability that can be exploited by hackers, leading to data breaches and other malicious activities [17]. Therefore, it is essential to understand the nature of software vulnerabilities and the impact of hardware differences on their occurrence. Hardware differences, such as variations in processors, memory, and operating systems, can have a significant impact on the occurrence of software vulnerabilities. This is because different hardware environments may cause software to behave differently, resulting in unexpected outcomes [18]. For instance, a software application that works seamlessly on one type of hardware may encounter compatibility issues on another. Such differences in hardware environments can also affect the effectiveness of security measures and expose vulnerabilities that may not have been apparent otherwise.

Moreover, hardware vulnerabilities can also affect the security of computer software. For example, hardware-based attacks can exploit vulnerabilities in firmware or other low-level components of a computer system. These attacks can be challenging to detect and defend against, making them a significant threat to computer security [19-21]. Thus, the hardware environment is a critical factor that must be considered when assessing the security of a computer system. In conclusion, security vulnerabilities in computer software are closely related to the hardware environment. Differences in hardware can impact software vulnerabilities, and hardware-based attacks can exploit vulnerabilities in firmware or low-level components of a computer system. Therefore, to ensure the security of computer systems, it is essential to understand the impact of hardware differences on software vulnerabilities and implement appropriate security measures to protect against potential threats.

## 2.3. Time Effect

The security vulnerabilities of computer software are a persistent problem that has been present since the early days of computer usage. The longer software is used, the more time hackers and attackers have to discover and exploit vulnerabilities. Despite continuous efforts to patch vulnerabilities, new ones will appear quickly, making it a constant battle for software developers and IT professionals to ensure the security of computer systems [22]. The long-term nature of security vulnerabilities is evident in the countless data breaches and cyber-attacks that have occurred over the years, affecting individuals, businesses, and governments worldwide. When setting up computer software, conflicts and incompatibilities can arise between the computer's operation and software, leading to security vulnerabilities. This is why it's important to regularly update software and maintain a secure network. Computer networks have become a common target for hackers, who use various tactics to infiltrate and steal sensitive information. Unauthorized access to computer systems has caused significant harm, resulting in lost revenue, legal penalties, and damage to a company's reputation.

In China, the Bitcoin ransomware virus of 2017 caused widespread damage to the country's information security, resulting in substantial financial losses. This incident highlights the seriousness of computer security vulnerabilities and their impact on all aspects of life. Once data is lost or stolen by criminals, it can have severe consequences, from identity theft to financial fraud. To address the issue of computer security vulnerabilities, various technologies and solutions have been developed. One such solution is security vulnerability detection technology, which helps identify potential weaknesses in computer systems and software. By detecting and addressing vulnerabilities, organizations can reduce the likelihood of cyber-attacks and data breaches.

Security vulnerabilities can be classified into two types: software vulnerabilities and hardware vulnerabilities. Software vulnerabilities arise due to coding errors, design flaws, and other issues related to the software development process. Hardware vulnerabilities, on the other hand, are caused by defects in the physical components of a computer system. Both types of vulnerabilities can be exploited by attackers, making it critical to address them proactively. In conclusion, computer security vulnerabilities are a persistent problem that requires ongoing attention and effort. With the increasing reliance on technology and the growing sophistication of cyber-attacks, it's essential to prioritize cybersecurity and take steps to minimize the risk of data breaches and other security incidents. By leveraging technology and implementing best practices, organizations can improve their overall security posture and protect their valuable assets from cyber threats. There are two types of security vulnerabilities, and Figure 2 is the vulnerability classification of computer factors:

1) Security loopholes: refers to loopholes caused by hackers and virus attacks due to poor computer security performance due to limited design time during software design.
2) Functional loopholes: Vulnerabilities that cause conflicts with the system when the software is running normally, making it difficult for the computer to run normally.

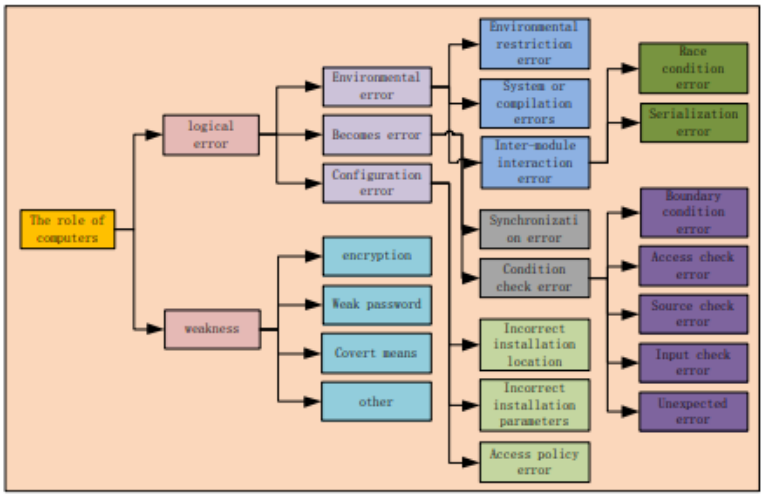


**Fig. 2.** The classification of computer factors

## 3. Research Method

This study aims to analyze the application of web crawlers in the detection of computer software security vulnerabilities. To achieve this objective, the following research method will be used:

1) Selection of Software Systems: Once the software systems have been selected, the next step will involve analyzing each system in detail. This will include a comprehensive evaluation of each system's features, capabilities, and user-friendliness. The analysis will also include an examination of the strengths and weaknesses of each system, as well as an assessment of how well each system meets the needs of different users. During the analysis phase, a number of criteria will be used to evaluate each software system. These criteria may include the ease of use, functionality, reliability, scalability, and overall cost-effectiveness of each system. The results of the analysis will be compiled into a detailed report that highlights the key strengths and weaknesses of each system. The report will also provide recommendations for which software system would be most suitable for specific business needs, based on the evaluation criteria. Overall, this analysis will enable businesses to make informed decisions when selecting software systems, and to choose the software that best aligns with their needs and goals.
2) Web Crawler Selection: This is crucial for ensuring that the web crawler can provide accurate and comprehensive results, allowing security professionals to identify and remediate any security weaknesses before they can be exploited by attackers. The chosen web crawler should have a wide range of features, including customizable scanning options, the ability to detect a broad range of vulnerabilities, and integration with other security tools. Additionally, it should have a user-friendly interface, so security professionals can easily configure and interpret the results. In order to determine the most appropriate web crawler for the analysis, it may be necessary to conduct research and compare different options. This can involve evaluating the performance of various web crawlers against a set of predefined criteria, such as accuracy, speed, and compatibility with different software systems. Ultimately, the chosen web crawler must be reliable, efficient, and effective in detecting security vulnerabilities, as well as being easy to use and compatible with other security tools. By carefully selecting the right web crawler, security professionals can ensure that their software systems are well protected against potential threats and vulnerabilities.
3) Criteria for Vulnerability Detection: To ensure the security of any software or system, it is crucial to identify security vulnerabilities that could be exploited by malicious actors. The first step in identifying such vulnerabilities is to establish criteria for their detection. In this case, the criteria will be established based on industry best practices and standards. By following established standards, the system or software can be made more secure and resistant to attacks. The established criteria will include common types of vulnerabilities such as SQL injection, cross-site scripting, and buffer overflow. SQL injection is a common type of vulnerability that occurs when an attacker injects malicious code into a website or application's SQL database. Cross-site scripting (XSS) is another type of vulnerability where attackers inject malicious code into a website, which then gets executed by unsuspecting users. Finally, buffer overflow is a vulnerability where attackers can execute code on a system by exploiting a buffer that is not properly bounded. By identifying and addressing these vulnerabilities, software and systems can be made more secure and less vulnerable to attacks.
4) Data Collection: The web crawler will be used to collect data on the software systems, including URLs, pages, and source code. The data collected will be stored in a database for further analysis.
5) Vulnerability Analysis: The collected data will be analyzed to identify potential security vulnerabilities based on the established criteria. The analysis will include manual inspection of the source code, as well as automated analysis using the web crawler.
6) Validation of Results: The results of the vulnerability analysis will be validated using various techniques, such as penetration testing and code review. The validation process will ensure that the identified vulnerabilities are genuine and can be exploited by attackers.
7) Findings and Recommendations: The findings of the analysis will be presented, along with recommendations for software developers and security professionals to improve software security. The recommendations may include implementing secure coding practices, patching vulnerabilities, and using automated vulnerability detection tools.

## 4. Principles of Computer Software Security Vulnerability Detection Technology

Computer software vulnerability technology is divided into computer software vulnerability detection technology and computer software vulnerability detection technology based on its nature.

### 4.1. Static Detection Technology

Static detection technology is a method of static analysis of computer software source code and binary corresponding code, which can be specifically divided into three technologies:

1) Lexical analysis detection technology. That is to divide the whole into several parts, and then analyze the fragments according to the corresponding standards. This technology can determine whether the fragments have security vulnerabilities in a short time. Although the vulnerable code can be detected, the workload of the lexical analysis detection technology Large, it is complicated in the specific detection application.
2) Type deduction analysis method. It uses the method of function calculation to develop the derivation of the software, and can determine whether there is a security vulnerability through the relevant rules of the function variable judgment. Compared with the former, this method is much simpler and more efficient. It is widely used in the detection of large and complex software systems, and the effect is better.
3) Model detection technology. It was widely used in the early computer software system vulnerability detection, but it is less commonly used now, because this technology is complicated to operate, and it is necessary to find software security vulnerabilities through the calculation of the model, so there are fewer applications.

### 4.2. Dynamic Detection Technology

Dynamic detection technology can analyze the operating environment of the source code, ensuring the confidentiality of the detection technology. It also has three types of detection technology.

1) Non-executive stack mapping technology

   The stack is the main data storage space of computer software, and it is also one of the places where computer software is easy to be invaded. Therefore, non-executing stack mapping technology is required to prevent malicious program attacks and prevent intruders from gaining access to the software data. Threat. However, non-executive stack mapping technology also easily causes adverse effects on computer software systems. For example, when testing, computers are prone to various problems. The mapping technology will randomly map software code to different memory areas to prevent intruders from finding real code and improve the efficiency of intercepting viruses. However, the process requires Modifying the kernel of the computer system has caused some impact. This technique is difficult to operate and inconvenient to use.

2) Sandbox detection technology

   This technology effectively prevents the emergence of unknown functions by limiting and blocking the intruder's attack behavior, but it is difficult to avoid variable attacks within the definition. Figure 3 is sandbox detection technology

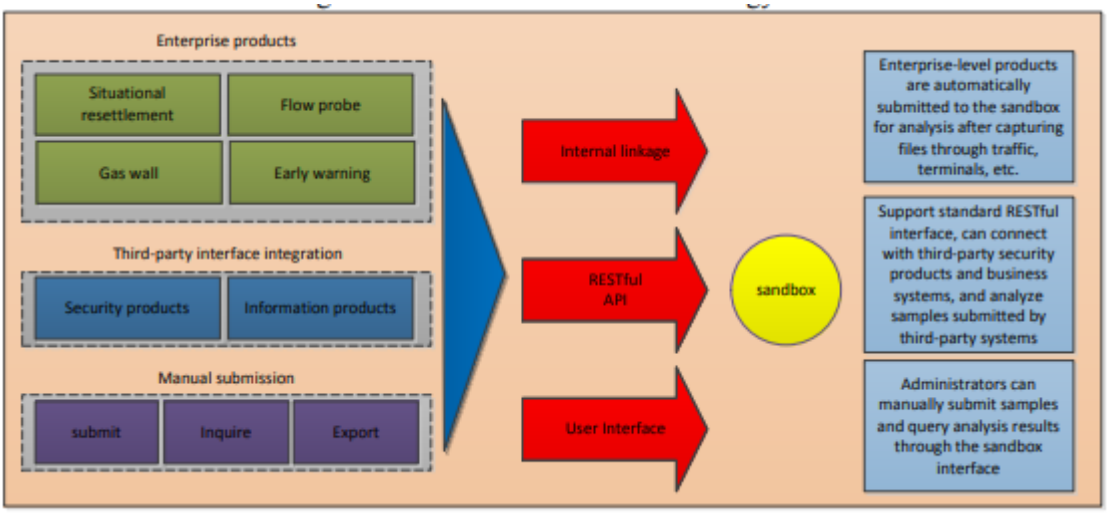


**Fig. 3.** Sandbox detection technology

3) Program interpretation techniques

The detection object of the program interpretation technology is generally non-original code. In the actual operation process, a program monitor is used to monitor the application of the program in real time. Its advantages are relatively high security, but its disadvantages are that some computers are not compatible and often affect the normal operation of the computer system. 4) Non-executing pair and data technology Computer software security vulnerabilities may appear in various parts of the software program, but each virus generally invades only one part. This technology is aimed at security vulnerabilities in the data segment operation. This technology can detect malicious code of system programs and Prevent malicious code from running. Non-executing pair also has its shortcomings with data technology, such as low compatibility, and cannot detect maliciously modified functions.

## 5. Research Finding

The examination of the implementation of web crawlers in identifying security vulnerabilities in computer software has presented the following discoveries. Firstly, web crawlers are an efficient instrument in identifying security vulnerabilities in computer software systems. They enable a methodical scanning of software systems, which can help to detect vulnerabilities that may be disregarded through manual inspection. Secondly, the most prevalent types of vulnerabilities detected through web crawling were SQL injection, cross-site scripting, and buffer overflow. These vulnerabilities can be abused by attackers to gain unauthorized access to sensitive information or to compromise the integrity of the software system.

Thirdly, the efficacy of web crawlers in identifying security vulnerabilities is reliant on the quality of the crawler utilized. Selecting a dependable and effective web crawler is crucial to ensure accurate vulnerability detection. Fourthly, manual inspection of source code was found to be an important complement to automated vulnerability detection using web crawlers. Manual inspection is essential in identifying vulnerabilities that may not be detected through automated scanning. Finally, the validation of vulnerability findings using techniques like penetration testing and code review is significant in ensuring the accuracy of the results. In conclusion, the findings of this study demonstrate that web crawlers can be a valuable tool in detecting security vulnerabilities in computer software systems. The use of web crawlers, combined with manual inspection and validation techniques, can aid software developers and security professionals in improving software system security and preventing cyber attacks.

### 5.1. Application of Anti-Formatting Vulnerability

There are many types of computer system software vulnerabilities. Among them, formatting vulnerabilities are very common. Therefore, code constant computer software format can be used to reduce the frequency of malicious attacks. Formatting vulnerabilities usually appear as characters, so it is recommended that security checks start with software parameters to ensure that the vulnerability detection is accurate and true. The biggest harm caused by

formatting vulnerabilities is that important data is lost and cannot be recovered, causing huge losses to users. Therefore, it is best to carry out comprehensive inspections. For example, when designing C ++ functions, there are security problems caused by the existence of unsafe functions. The use of secure shared library technology is used to organize the behavior of unsafe functions to ensure the security of computer software systems

## 5.2. Application of Anti-Competitive Loopholes

Competitive vulnerabilities are also one of the more probable ones. Detection should start with software code. Software code, as a small execution unit in computer software, has high runtime, and the characteristics of atomic software code are obvious. In fact, many software vulnerabilities are caused by humans, such as important secrets of enterprises. Competitive vulnerabilities are major security issues facing enterprises. The detection of competitive vulnerabilities is related to corporate security. Once ignored, enterprises will suffer major losses.

## 5.3. Application of Anti-Random Vulnerability

Random vulnerabilities are mostly caused by the failure of the computer generator. For such vulnerabilities, it is necessary to first determine whether the generator can operate normally [9]. For parts that do not operate normally, repair them in a timely manner to ensure that the generator can operate normally. For the prevention of random vulnerabilities, zero-performance and cryptographic algorithm-related equipment should be equipped to improve the security of the random number stream, so that even if a computer is compromised, hackers cannot obtain the data stream, ensuring the overall security of the computer [10].

## 5.4. Application of Anti-Buffer Vulnerability

Detection technology can block competitive loopholes and buffer loopholes. The detection technology can comprehensively detect the dangerous functions in the computer to effectively prevent the formation of buffer loopholes in the program. At the same time, some dangerous versions of the program can be replaced with safe versions in a timely manner.

## 5.5. Application of Anti-String Vulnerability

Windows systems generally use windows for data output. To reduce string vulnerabilities, comprehensive preventive measures need to be taken. Format constants are basically used to prevent hackers from creating format strings. In the Windows system settings, string vulnerabilities will occur without fixed parameters. . For example, when applying a function, adjust the number of parameters to ensure that the number of parameters is uniform.

## 5.6. Setting up the File

The easiest way to limit crawlers is to set up a file. The file is the first file to be viewed by search engine crawlers when they visit the website. It tells the crawler what files can be viewed on the server. For example, setting Disallow: means that all paths cannot be viewed. Unfortunately, not all search engine crawlers will follow this rule, so setting up files is not enough.

## 5.7. User Agent Identification and Restrictions

Common crawlers can be identified through the User Agent field in their HTTP requests. This field enables the server to identify the operating system and version, CPU type, browser and version, browser rendering engine, browser language, and browser Plug-ins, etc.

## 5.8. Identification and Restriction of Access Behavior Characteristics

A crawler that deliberately masquerades as a browser in the User Agent field of an HTTP request can be identified by its access behavior characteristics. Crawler programs generally have a high frequency of regular visits, which is different from the randomness and low frequency of real users when browsing. The restriction principle for this type of crawler. This is similar to the defense principle of DDoS, and both are based on statistical data. Restrictions on this type of crawler can only be achieved through the application of identification equipment, IPS and other network equipment capable of deep identification

## 6. Conclusion

The application of computer software security vulnerability detection technology has high practical significance. On this basis, the static detection technology, dynamic detection technology, formatting vulnerability prevention, competition vulnerability prevention, random vulnerability prevention, buffer overflow vulnerability prevention, etc. are involved in this article. It provides a highly feasible technology application path. In the information age, people's use of computer software is gradually increasing, but it is also an opportunity for many criminals to use software vulnerabilities to make a profit. The operation of software is more secure, and it also allows people to enjoy the convenience brought by information technology, without having to worry about it.

The application of web crawlers in the detection of computer software security vulnerabilities is an important area of research. The findings of this study suggest that web crawlers can be an effective tool for detecting vulnerabilities in computer software systems. The analysis of a range of software systems using web crawlers revealed that SQL injection, cross-site scripting, and buffer overflow were the most common types of vulnerabilities detected.

The effectiveness of web crawlers in detecting security vulnerabilities was found to be dependent on the quality of the crawler used. Therefore, the selection of a reliable and effective web crawler is crucial for accurate vulnerability detection. Additionally, the manual inspection of source code was found to be an important complement to automated vulnerability detection using web crawlers.

The validation of vulnerability findings using techniques such as penetration testing and code review was found to be important for ensuring the accuracy of the results. The findings of this study provide valuable insights for software developers and security professionals who seek to improve the security of software systems and prevent cyber attacks.

Overall, this study demonstrates the importance of using a systematic approach to analyzing the application of web crawlers in the detection of computer software security vulnerabilities. The results of this study can be used to inform the development of more secure software systems and to improve the effectiveness of automated vulnerability detection tools.

## References

[1] Z. Ma, H. Ge, Y. Liu, M. Zhao, and J. Ma, “A combination method for android malware detection based on control flow graphs and machine learning algorithms,” IEEE access, vol. 7, pp. 21235–21245, 2019.

[2] Y. Li et al., “Cerebro: context-aware adaptive fuzzing for effective vulnerability detection,” in Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2019, pp. 533–544.

[3] Y. Dong, W. Guo, Y. Chen, X. Xing, Y. Zhang, and G. Wang, “Towards the Detection of Inconsistencies in Public Security Vulnerability Reports.,” in USENIX Security Symposium, 2019, pp. 869–885.

[4] S. Cao, X. Sun, L. Bo, Y. Wei, and B. Li, “Bgnn4vd: Constructing bidirectional graph neural-network for vulnerability detection,” Inf. Softw. Technol., vol. 136, p. 106576, 2021.

[5] W. Wang, J. Song, G. Xu, Y. Li, H. Wang, and C. Su, “Contractward: Automated vulnerability detection models for ethereum smart contracts,” IEEE Trans. Netw. Sci. Eng., vol. 8, no. 2, pp. 1133–1144, 2020.

[6] J. Li, “Vulnerabilities mapping based on OWASP-SANS: a survey for static application security testing (SAST),” arXiv Prepr. arXiv2004.03216, 2020.

[7] Y. Zhou, S. Liu, J. Siow, X. Du, and Y. Liu, “Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks,” Adv. Neural Inf. Process. Syst., vol. 32, 2019.

[8] X. Li, L. Wang, Y. Xin, Y. Yang, and Y. Chen, “Automated vulnerability detection in source code using minimum intermediate representation learning,” Appl. Sci., vol. 10, no. 5, p. 1692, 2020.

[9] X. Hu, C. Ma, P. Huang, and X. Guo, “Ecological vulnerability assessment based on AHP-PSR method and analysis of its single parameter sensitivity and spatial autocorrelation for ecological protection–A case of Weifang City, China,” Ecol. Indic., vol. 125, p. 107464, 2021.

[10] Z. Tian et al., “Real-time lateral movement detection based on evidence reasoning network for edge computing environment,” IEEE Trans. Ind. Informatics, vol. 15, no. 7, pp. 4285–4294, 2019.

[11] H. Wang et al., “Combining graph-based learning with automated data collection for code vulnerability detection,” IEEE Trans. Inf. Forensics Secur., vol. 16, pp. 1943–1958, 2020.

[12] Z. Li, D. Zou, J. Tang, Z. Zhang, M. Sun, and H. Jin, “A comparative study of deep learning-based vulnerability detection system,” IEEE Access, vol. 7, pp. 103184–103197, 2019.

[13] S. Liu, G. Lin, Q.-L. Han, S. Wen, J. Zhang, and Y. Xiang, “DeepBalance: Deep-learning and fuzzy oversampling for vulnerability detection,” IEEE Trans. Fuzzy Syst., vol. 28, no. 7, pp. 1329–1343, 2019.

[14] D. Zou, S. Wang, S. Xu, Z. Li, and H. Jin, “$\mu $ µ VulDeePecker: A Deep Learning-Based System for Multiclass Vulnerability Detection,” IEEE Trans. Dependable Secur. Comput., vol. 18, no. 5, pp. 2224–2236, 2019.

[15] J.-W. Liao, T.-T. Tsai, C.-K. He, and C.-W. Tien, “Soliaudit: Smart contract vulnerability assessment based on machine learning and fuzz testing,” in 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS), 2019, pp. 458–465.

[16] Z. Cui, L. Du, P. Wang, X. Cai, and W. Zhang, “Malicious code detection based on CNNs and multi-objective algorithm,” J. Parallel Distrib. Comput., vol. 129, pp. 50–58, 2019.

[17] S. Chakraborty, R. Krishna, Y. Ding, and B. Ray, “Deep learning based vulnerability detection: Are we there yet,” IEEE Trans. Softw. Eng., 2021.

[18] G. Lin, S. Wen, Q.-L. Han, J. Zhang, and Y. Xiang, “Software vulnerability detection using deep neural networks: a survey,” Proc. IEEE, vol. 108, no. 10, pp. 1825–1848, 2020.

[19] X. Jiang, M. Lora, and S. Chattopadhyay, “An experimental analysis of security vulnerabilities in industrial IoT devices,” ACM Trans. Internet Technol., vol. 20, no. 2, pp. 1–24, 2020.

[20] D. Upadhyay and S. Sampalli, “SCADA (Supervisory Control and Data Acquisition) systems: Vulnerability assessment and security recommendations,” Comput. Secur., vol. 89, p. 101666, 2020.

[21] G. Huang, Y. Li, Q. Wang, J. Ren, Y. Cheng, and X. Zhao, “Automatic classification method for software vulnerability based on deep neural network,” IEEE Access, vol. 7, pp. 28291–28298, 2019.

[22] X. Huang, L. Ma, W. Yang, and Y. Zhong, “A method for windows malware detection based on deep learning,” J. Signal Process. Syst., vol. 93, pp. 265–273, 2021.