International Journal of Digital Crime and Forensics
Volume 14 • Issue 2

Web Vulnerability Detection Analyzer Based on Python

Dawei Xu, Changchun University, China*
https://orcid.org/0000-0003-4422-0606

Tianxin Chen, Changchun University, China

Zhonghua Tan, Hainan Normal University, China

Fudong Wu, Changchun University, China

Jiaqi Gao, Changchun University, China

Yunfan Yang, Changchun University, China

ABSTRACT

In the information age, hackers will use web vulnerabilities to infiltrate websites, resulting in many security incidents. To solve this problem, security-conscious enterprises or individuals will conduct penetration tests on websites to test and analyze the security of websites, but penetration tests often take a lot of time. Therefore, based on the traditional web vulnerability scanner, the web vulnerability detection analyzer designed in this article uses vulnerability detection technologies such as sub-domain scanning, application fingerprint recognition, and web crawling to penetrate the website. The vulnerability scanning process of the website using log records and HTML output helps users discover the vulnerability information of the website in a short time and patch the website in time. It can reduce the security risks caused by website vulnerabilities.

KEYWORDS

Fingerprint Recognition, Network Security, Penetration Test, Subdomain Scanning, Vulnerability Scanning, Web Crawler, Web Security, Web Vulnerability

1. INTRODUCTION

With the continuous emergence of advanced Web application technologies in the Internet era, related Web vulnerabilities are also emerging. Web vulnerabilities may be due to lack of consideration of web security by website developers when developing websites, resulting in related security vulnerabilities in applications. Common web security vulnerabilities include SQL injection vulnerabilities, cross-site scripting vulnerabilities, and cross-site request forgery vulnerabilities. etc. (Yang Guofeng. 2019). Hackers can conduct penetration tests on target websites and use Web vulnerabilities to escalate privileges on website servers to achieve the purpose of invading websites. Based on these security threats, there is some value in using vulnerability scanners to detect vulnerabilities on websites.

DOI: 10.4018/IJDCF.302875 *Corresponding Author

This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

The scanning process of traditional scanners is generally to obtain the URL of the website through a crawler, send a request with attack parameters to the website to obtain the payload, and output the corresponding vulnerability report if the payload is successfully verified. If the verification fails, continue to send the next request. Due to the high concurrency between modules, the next task can only be started after the completion of the previous task. The Web vulnerability detection analyzer designed in this paper can collect website information in batches to achieve high concurrency between modules, and tasks can be processed between crawlers and plug-ins at the same time, improving the efficiency of scanning websites, and the vulnerability script of the system has Scalability is conducive to the improvement and upgrade of the system. The vulnerability detection analyzer adopts a callable plug-in framework, which can automate the scanning process, actively send a request with parameters to the target website, and detect website vulnerabilities according to the obtained response.

Contributions made in this paper include:

1. According to the process analysis of website vulnerability scanning, the overall architecture of the web vulnerability detection analyzer and the functional requirements of the four modules are designed.
2. According to the cross-platform operation requirements of vulnerability scanning, the system is written in Python language.
3. According to the requirements of vulnerability verification, this paper uses a custom PoC plug-in to verify website vulnerabilities, uses multi-process concurrent engine operation mode, uses logs to record the response information returned by website requests, and provides targets for vulnerability verification.
4. For the completed system, the vulnerability scanning test of the website is carried out to test whether the function of the system is complete and the efficiency of scanning website vulnerabilities. This paper conducts vulnerability scanning tests on hundreds of websites, and divides these websites into three different scales. Test the total scan time of the website and the accuracy of website vulnerability results.

2. BACKGROUND

The essence of Web application security problems stems from the quality of software. Compared with traditional software, web applications are usually considered to be enterprise-specific applications, and functions in them need to be changed frequently to maintain normal business, which leads to a longer development cycle for web applications; due to the communication between the client and the server. The process is cumbersome, and it is not easy for many development technicians to sort out the communication logic, which leads to problems in the security of Web applications.

For web application developers, common vulnerabilities in web security arise when developing web applications. Most programmers who develop websites have weak security awareness. They trust any data entered by users when writing application code. The input parameters are not tested or strictly tested, and the relevant technical personnel do not consider the filtering of special characters. This means that many applications have web vulnerabilities (Teng Fei. 2020). In the process of writing code, the programmer may use the interface incorrectly or the interface is not perfect, and there may be defects in the code function or logic loophole in the application program. Many web vulnerabilities can be avoided if the web application code takes into account the security of the website. Therefore, web application developers should strictly identify and filter the special characters of website HTTP headers, keyword queries and POST data entered by users in the process of code development. For web site administrators, failure to configure the type of software correctly, to patch vulnerable sites in a timely manner, and to properly handle abnormal problems in web applications are the main reasons for web security problems. Web site administrators can scan for vulnerabilities to identify the configuration parameters of various sites, install the latest security patches on the sites in a timely

manner, regularly check the logs of the Web server, and analyze whether there are Web vulnerabilities on the site, so as to ensure the security of the website (Zhang Bing, Ren Jiadong & Wang Jun. 2020).

Vulnerability scanning is divided into black box scanning and white box scanning in different application environments (Li Weifeng. 2021). Black box scanning tests the security of a website by simulating penetration testing attacks, reproducing real hacker attacks, dynamically testing every feature of a website, and judging the existence of web vulnerabilities on a website. White-box scanning is a periodic vulnerability scan of the host's operating system by the patch update program. When a security vulnerability is found on the host, the vulnerability in the operating system will be repaired in time (Sun Yongqing, Lu Zhen & Shen Liang. 2020). Because the host has protection software such as firewalls, the vulnerability results obtained by white-box scanning are of no value in penetration testing of the Internet. Therefore, the vulnerability scanner uses black-box scanning technology to scan the target website for security.

Web Vulnerability Scanner is a security testing tool used to detect web vulnerabilities of websites. According to the structure, it can be divided into browser/server (B/S) structure and client/server (C/S) structure (Liu Li. 2007). The working principle based on the B/S structure is that after the user uses the scanner to input the scanning command of the target site, the corresponding scanning module in the scanner will call other related function modules to send the request to the target server after obtaining the scanning command of the user. scanning. After the scanner analyzes the response data packet of the target site, it outputs the vulnerability result of the target site to the user. The working principle based on the C/S structure is to reproduce specific vulnerabilities through penetration testing, write a vulnerability test script according to the corresponding steps, use the script to conduct batch tests on the Internet, observe the influence and coverage of the vulnerability, and detect The service and port of the target site collect the detected information into the database, and then call other plug-ins to send requests to the target site, and save the results of the response packets in the database. This information can be submitted to other modules for testing and scripting.

The Web vulnerability detection analyzer implemented in this paper is an active testing tool that actively scans the target website by initiating HTTP requests. The steps of web vulnerability scanning are divided into crawlers to obtain web pages, using plug-ins to discover website injection points, and website vulnerability detection and verification (Hao Zixi. 2018). Through page crawling, crawling out the entire web application structure of the target website, constructing a special request to return response data packets, and finding exploitable website injection points in the response data packets, so as to verify the web vulnerabilities existing in the website, according to the website The result of vulnerability detection is output in HTML format, and information such as the risk level and type of website vulnerability is presented to the user.

3. IMPLEMENTATION OF WEB VULNERABILITY DETECTION ANALYZER

In this paper, the Web vulnerability detection analyzer based on Python language is designed. First, the crawler is used to collect information such as the subdomain name, port, and website fingerprint of the website. Through this information, the vulnerability of the website is detected, and the corresponding report is generated and presented to the user. When the system sends a scan request to the target website, there will be the following situations: the request resource does not return 404 status code, the request is intercepted by the Web Application Protection System (WAF) protection and returns 403 status code, and the request is redirected because of the default settings set by the server. Returns 301 or 302 status code. The WAF of the website will filter the scanning requests of the target website by the scanner through a single point of regularity. Since some subdomains of the website are not protected by WAF, the vulnerability scanning will collect the information of the subdomains of the website, send scanning requests to these subdomains, and further respond by responding The data packet judges the information of the target website, thereby detecting the web vulnerability existing in the website.

Web vulnerability scanning requests have obvious text features in SQL injection vulnerability scanning and cross-site scripting vulnerability scanning. These text features are helpful to distinguish the responses returned by normal requests, resulting in differences. The process of SQL injection attack is to change and merge the original URL, data packet or form field input parameters of the webpage into SQL statements. These SQL statements are first passed to the website server, and then passed to the back-end database to achieve the purpose of executing malicious SQL attack statements (Li Xin. 2020). The attacker uses the characteristics of the database to write malicious SQL statements to execute in the database, obtain sensitive data in the database, and then obtain the system permissions of the server. Cross Site Scripting (XSS) means that hackers use the dynamic content of the website to implant some malicious code in the web application. Because the website does not escape the parameters entered by the user or filter special characters, the website executes Corresponding malicious code, so that hackers can steal important information of users and modify user settings (Bai Wanjiao, Yang Jun & Zhao Yitong. 2021). XSS requests have obvious string characteristics, such as alert (1), prompt (1), confirm (1), etc.

After the system finds the detection point of the website, it will verify the vulnerability according to the Proof of Concept (PoC) in the system. PoC can prove the existence of the vulnerability. The specific process of writing a PoC is to find the program that affects the version according to the details of the vulnerability, reproduce the vulnerability, and analyze the proof steps of the vulnerability; through these steps, the code program of the PoC can be written, which is divided into three parts in the system, which are assign, audit, main, where the assign function is used to identify the fingerprint of the target website, the audit function is used to verify the payload audit vulnerability returned by the website request, and the main function is used for local testing. When scanning a target website, many PoCs need to be called, so the PoC framework is required in the system to manage PoCs in batches and schedule PoC programs. Writing PoC scripts requires some specifications, such as entry specifications and Application Programming Interface (API) specifications. These specifications are conducive to the framework calling PoC and network request tools, and have better fault tolerance.

The modules of the system are divided into scanning interface, information collection, vulnerability verification, and report generation. The overall workflow of the system is as follows: first, enter the target URL in the scanning interface, collect information and detect backup files based on the crawler, use the depth-first traversal algorithm to crawl the relevant pages and information of the target website, and then perform system port scanning, Port fingerprint analysis and website structure analysis, etc., the system uses logs to record the process of website vulnerability scanning, outputs website vulnerability results in HTML, and provides users with vulnerability information of the target website (Meng Qing, Lu Hejun, Liu Dui & Gao Yu. 2020). The overall architecture is shown in Figure 1.

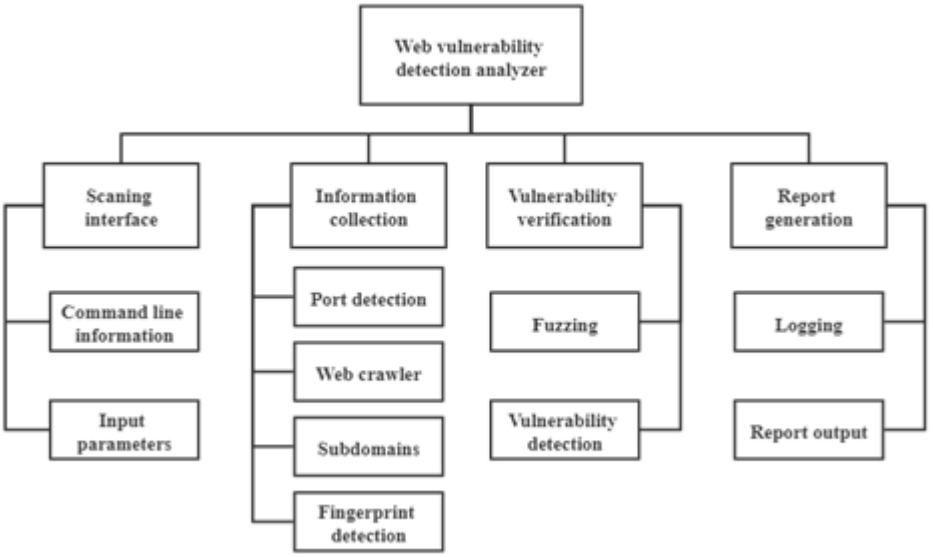
3.1 Scan Interface Module

The design of the scanning interface is very simple. Users can understand the command line prompts on the interface, and enter the target website to be detected according to the prompts, select the corresponding plug-in, the number of threads to be executed, and the traversal depth of the crawler. The system can hand over the tasks entered by the user to other modules for multi-threaded vulnerability scanning of the website.

3.2 Information Collection Module

Information gathering is critical for subsequent vulnerability verification. After collecting the information of the target website, the program can detect whether the website has the possibility of vulnerability. This module will obtain the information of the target website in many ways, such as crawler, port scan, port fingerprint detection and subdomain collection, which can be used to detect the insecure factors in Warnings, Cookies, and Header (Lang Zhizhe, Feng Xiaoyu & Dong Qifen. 2017). Information collection can obtain the following contents: basic information such as Internet

Figure 1. Overall architecture



Interconnection Protocol (IP for short), domain name, port, application information of each port, system information of operating system version, etc. (Deng Zelong. 2020).

3.2.1 Crawler-Based Website Scan

When scanning the target website, use the crawler to collect information on the target website and crawl the relevant URL of the target website. When the crawler sends the request to the website, it uses the constructed headers to simulate the browser header information, and finally encapsulates the object to request the website to obtain a response. The response can obtain the information of the page, and then parse the data one by one (Hou Meijing. 2018), and save the data in the In the database. It is convenient to call the vulnerability verification module later. Since the request sent by the system has no response during the scanning process, the web page error collection plug-in is added, and the system can detect different types of errors in web pages.

The pseudocode of the implementation is shown in Algorithms 1 and 2.

The website crawler process is shown in Figure 2.

3.2.2 Subdomain Scan

A subdomain is the next level of a top-level domain and belongs to a higher-level domain in the domain name system (Mo Huaihai & Li Xiaodong. 2019). Most websites with top-level domains will enable WAF attack protection, but websites with subdomains generally do not have WAF protection. Therefore, when the system scans website information, it will collect subdomains to expand the scope of website vulnerability detection and find breakthroughs in target sites. (Lian Bin & Liu Yongjian. 2017). The subdomain blasting of the system will be accessed through dictionary splicing, for example, www.example.com can be obtained by splicing www in the dictionary to the target website example.com. If the subdomain access is successful, it means that the subdomain of the target website exists. However, in this process, there will be a problem of subdomain pan resolution. Pan resolution will lead to the consequence that no matter what type of subdomain is used to scan the website, the IP address of the website can be accessed. In order to solve this problem, the project uses a randomly

Algorithm 1. Crawler scheduler

```
Data: urlManager urls,htmlDownloader downloader, html parser parser, total number
of pages crawled maxdeep, sequence of current url deep
Result: Control flow
1 while deep ⩽ maxdeep and maxdeep > 0 do
2         newurl ← urls.getnewurl();
3         html ← downloader.download(newurl);
4         newurls ← parser.parse(newurl, html);
5         deep ← deep + 1;
6 end
```

Algorithm 2. URL Manager

```
Data: URLs crawled oldurls,URLs not crawled newurls,A newly crawled URL url
Result: Deduplication of crawled urls
1 newurls ← set();// Use set() to deduplicate URLs
2 oldurls ← set();
3 newurls.pop();// Take a URL out of newurls
4 oldurls.add(newurl);// Add newurl to oldurls
5 if url is None then
6         return None;
7 end
8 if url not in newurls and url not in oldurls then
9         newurls.add(url);
10 end
```

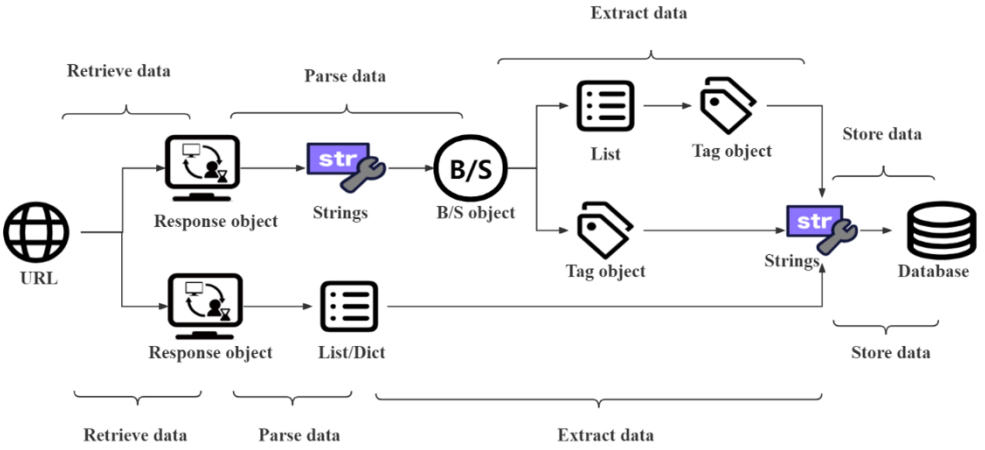
constructed subdomain name to verify the returned web page, obtains the MD5 encrypted feature of the web page, and then uses this feature to check whether the subdomain name has the problem of pan-resolution.

The relevant pseudocode for subdomain pan-resolution implementation is shown in Algorithm 3.

3.2.3 Fingerprint Recognition

Content Management System (CMS) is a software system that works between the front-end and back-end of the Web (Rao Lanxiang, Sun Dan, Shi Weili & Meng Shasha. 2020). Due to the difficulty and workload of manual development and maintenance of website content management, most companies or individuals now use open-source CMS codes when developing websites. These architectures are fully functional and beautiful, and users only need to use them. By adding your own text content and pictures, you can complete a website with good effect. Users can also regularly update data according to the CMS to maintain the website, which greatly reduces the workload.

Figure 2. Website crawler process



Algorithm 3. Subdomain

Data: *Defined ineligible strings "justtestjusttetnoanyothemeaning",the primary domain name of the website domain,Web services provided socket, subdomain of the website subdomain*

Result: *Solve subdomain pan resolution*

1 unablestr ← "justtestjusttetnoanyothemeaning";

2 hostnames ← unablestr + . + domain;// Add unable stings to the domain

3 hostnames ← hostnames.strip();// Remove leading and trailing spaces from hostnames

4 l ← socket.getNames(hostnames);

5 ***if*** *l! =0* ***then***

6 securityInfo("Existence of pan-analytics");

7 ***end***

Fingerprint identification is based on the corresponding feature codes in the files of the website, and the corresponding CMS can be identified according to different feature codes (Zhou Shuangfei. 2020). CMS is an open-source program and has been audited for vulnerabilities by many white hat hackers. At present, there are many CMSs on the Internet. If it can be determined that it is a certain CMS, the historical loopholes of the CMS can be inquired through a search engine, so as to try to exploit these loopholes (Tong Ying, Yao Huanzhang, Liang Jian, Wang Xigang & Zhou Yu. 2020). The Service parameter is the fingerprint feature of the service name in the target website. The corresponding plug-in can be called according to the fingerprint feature. The fingerprint content submitted by the Service is divided into two parts. The first part is the feature path, and the second part is the feature value of the path. The characteristic value of the image service is its corresponding MD5 value, and

the characteristic value of the text service is mostly a string. In the process of fingerprint recognition, first extract the MD5 from the fingerprint database of the system, start with a fingerprint feature, extract the MD5 of the first fingerprint, and compare it with the MD5 of the fingerprint storage path in the target URL, if they match each other If there is a match, the fingerprint feature of the website can be collected; if there is no successful match, start from the second fingerprint feature in the fingerprint database and repeat the above process.

3.2.4 Port Scan

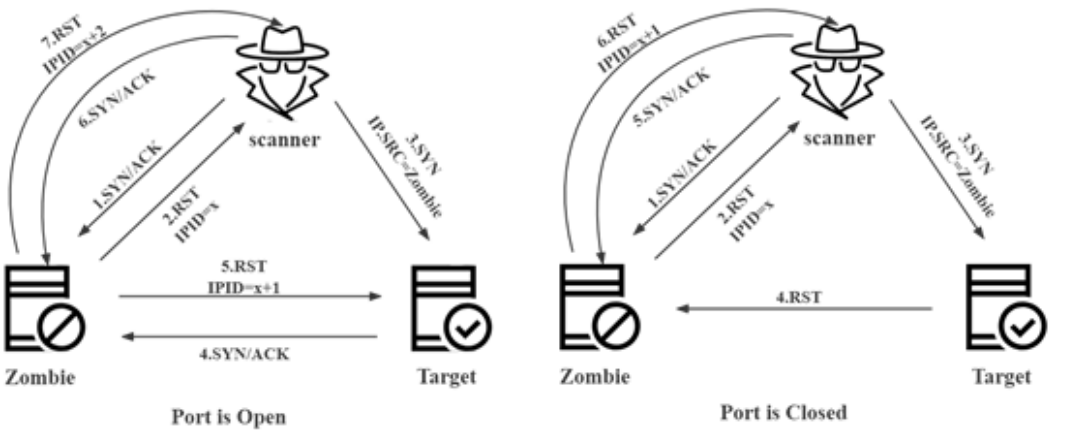
The ports correspond to various services of the website. The vulnerabilities of the server program can be found through the ports. If an open port is found in the port scan, the system can collect information such as application services running on the open port and the target operating system (Meng Bin, Zhiyun Lei & Zhong Fei. 2021), the collected port information can be identified according to the fingerprint, and the corresponding service of the website can be detected. The process of port scanning is to send a message to each port, and judge whether the port of the computer is open according to the information returned by the port. Port scanning usually detects the open ports of a host by sending TCP packets to a specific IP. There is zombie scanning, full connection scanning and stealth scanning in TCP port scanning. When the zombie port scan is performed, the system sends STN/ACK packets to the zombie machine, and gets the RST and IPID of the zombie machine, and then forges the IP as a zombie machine, and sends a SYN packet to the target server, and the target server sends an RST packet to the zombie machine., according to the RST packet returned by the zombie to the system to determine whether the port is open; when the system performs covert scanning, it establishes an incomplete TCP connection with the target server, and determines whether the port is open according to the returned RST packet; when the system performs a full connection scan, Determine whether the port on the target server is open according to the data packets after the system and the target server perform the TCP three-way handshake.

The zombie port scanning process is shown in Figure 3.

3.3 Vulnerability Verification Module

After the system collects the information of the target website, it finds the website's injection point interface through the website information obtained through the information collection, performs

Figure 3. Zombie port scan



SQL injection, cross-site scripting injection, and command injection on the interface, constructs an HTTP request for the target website, and sends the test Data packets, to determine whether the website responds to the data packets, if the system obtains the response from the website, then the target website has vulnerability characteristics (Zhang Bin. 2019).

The system will use the fingerprint identification result of the information collection module to detect the vulnerability fingerprint feature of the website, and determine whether the fingerprint feature is a CMS fingerprint. If the match is successful, the CMS fingerprint feature will be returned, and the vulnerability information of this website will be penetrated and verified. If it is not the CMS fingerprint feature, call all the fingerprint libraries in the project to match other service fingerprint features, and if the matching is successful, the vulnerability will be exploited. If it fails, then the website does not have the corresponding type of vulnerability. In the process of vulnerability verification, the engine of the system first transmits the obtained website fingerprint, and then loads all modules of the system. After traversing these modules, it determines whether the service name of the website is consistent with the obtained fingerprint, and if so, the system is started. Thread, load the corresponding plug-in, and exploit the specified vulnerability; if the service name and fingerprint are inconsistent, inject a delay code into the website to detect whether the website is delayed. Types of Web Vulnerabilities.

Fuzzing is a method of examining how a computer program or system responds to various inputs and information. This process involves generating some type of data, either completely random or randomly under certain constraints, that can be used as system events, keyboard input, simulated network signals, or even files to load, and then converting this data Enter into a program to test how it handles unexpected information. In its most basic form, fuzzing sends a random sequence of keystrokes or characters to the program and checks that the program handles them correctly. More sophisticated fuzzing uses random structured data manipulation and sending into the program, elements of the target program can be taken and manipulated to produce situations that could be exploited maliciously, possibly including changing the order of spawning processes, modification of permissions, or make changes to core data and library files.

Fuzz testing can automatically detect whether the target website has SQL injection vulnerabilities and XSS vulnerabilities involved in the project. The system will test the possible vulnerabilities in sequence according to the vulnerability verification rules written in advance (Ren Zezhong, Zheng Han, Zhang Jiayuan, Wang Wenjie, Feng Tao, Wang He & Zhang Yuqing. 2021). During the test, some type of data is generated, and then the data is input to test whether the WAF rules of the detected target URL are complete. There are a large number of vulnerability verification carriers in the system. When using plug-ins to detect vulnerabilities, select the payload for testing, and verify the data packets returned by the test. During fuzzing, the system can perform nondestructive scanning of the target website without causing damage to the target system.

The relevant pseudocode for fuzzing is shown in Algorithm 4.

The vulnerability verification process is shown in Figure 4.

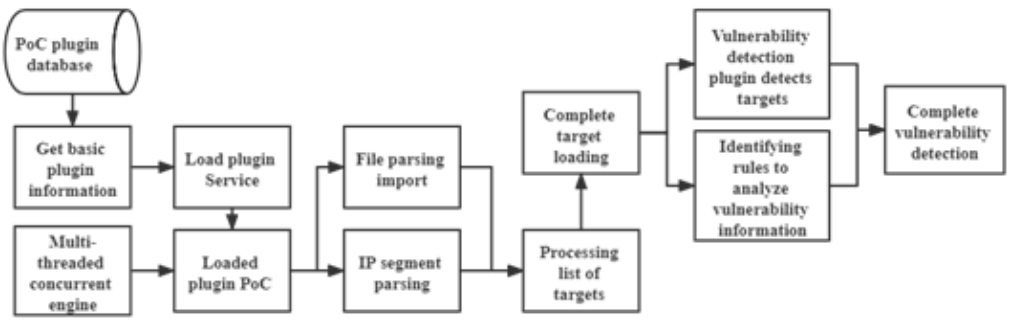
The system uses the plug-in by calling the method in the module, and after identifying the feature fingerprint, the scanning function of the plug-in is added to the thread pool. The PoC plug-in will make an integration to verify the specific vulnerabilities disclosed on the Internet. A thread pool is added to the plug-in of the system to improve the speed of website vulnerability detection, but there will be problems of program exceptions, so the thread engine plug-in is used in the thread pool. Waiting for the scanning task in the thread pool of the system. When starting the scanning task, first check whether the scanning task has all occupied the thread pool, and then check whether there is an empty queue. If there is an empty space in the queue, then check the scanning task. If there is no exception, start the scanning task; when the thread pool is full, block and wait for the scanning task, and finally repeat the above steps.

The thread pool process is shown in Figure 5.

Algorithm 4. Fuzzing

```
Data: http library hackhttp,target URL arg
Result: Get the elements of the target website
payload ← [];
len ← [];
for i in payload.strip().splitlines() do // Get a list with rows as elements
        url ← arg + i;
        code, header, body, redir, logging  ← hackhttp.http(url);
        if code! = 400andcode! =404 then
                payloaddict ← dict();
                payloaddict[" code"] ← code;
                payloaddict["url"] ← url;
                payloaddict[" len"] ← len(body);
                len.append(payloaddict[" len"]);
                payloads.append(payloaddict);
        end
end
```

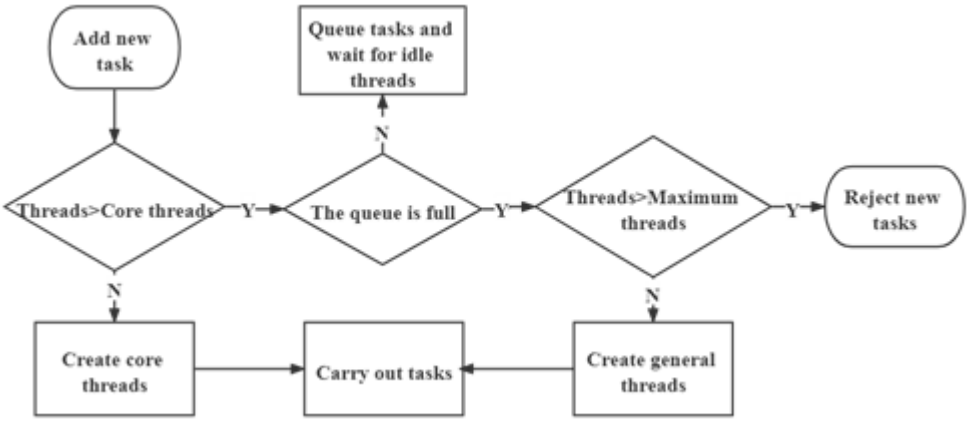
Figure 4. Process of vulnerability verification



3.4 Report Generation Module

The vulnerability detection analyzer will log the acquired vulnerability scanning process information and the content presented on the terminal and output the report. The log information includes the system request information and request parameters, the content of the website response, and the system plug-in detection, etc. After passing the vulnerability verification, the information recorded in the log will be reported in HTML format. The output HTML report contains the information of SQL injection detection and XSS detection, which is provided to users to analyze the vulnerability information of the website. The output HTML report will record the process of scanning website vulnerabilities and the content of information collection, and will display the level of website vulnerabilities. The content information will be clearly presented in the form of a table, so that the user can understand the

Figure 5. Thread pool flowchart



results of the website vulnerability scanned more clearly, and it is convenient for the user to analyze the cause of the vulnerability according to the content of the output report, and take corresponding security measures for the website.

4. SYSTEM TEST

4.1 Test Design

4.1.1 Test Purposes

The system is used to test whether there are loopholes in the website and help users to better detect and analyze the security of the website. The purpose of the test is to check whether the functions of each module are complete and the interoperability between functions, and whether the PoC plug-in integration in the project can successfully utilize and verify the Web vulnerabilities existing in the website.

4.1.2 Test Requirements

According to the functions that each module of the system needs to implement, it is divided into the following points for testing:

1. It is necessary to test whether the commands on the scanning interface can be executed and whether the startup function is normal, and a good boot interface is also required. Instructions for the use of the command line are essential, which can help users make relevant settings and execute scheduled tasks. Whether the user's command can be executed is the key to verifying whether the vulnerability detection analyzer can run normally.
2. It is necessary to test whether the functions of crawler, subdomain name collection, and fingerprint detection of the information collection module are complete, and conduct corresponding debugging and testing of these functions in the Python integrated environment.
3. It is necessary to test whether the vulnerability detection analyzer can realize the module function of vulnerability scanning, including plug-in loading, crawler, fuzzing, thread scheduling, etc., and observe whether the system can realize high concurrent scanning.
4. It is necessary to test whether data exchange can be achieved between the various functions of the Web vulnerability detection analyzer, and to check the interoperability between each

module. After entering the target website in the Web vulnerability detection analyzer, observe the information of each module in the scanning process and whether a report is generated at the end.

5. It is necessary to check whether the HTML report of the output vulnerability scan can be output normally and whether there are errors in the typesetting information on the page.

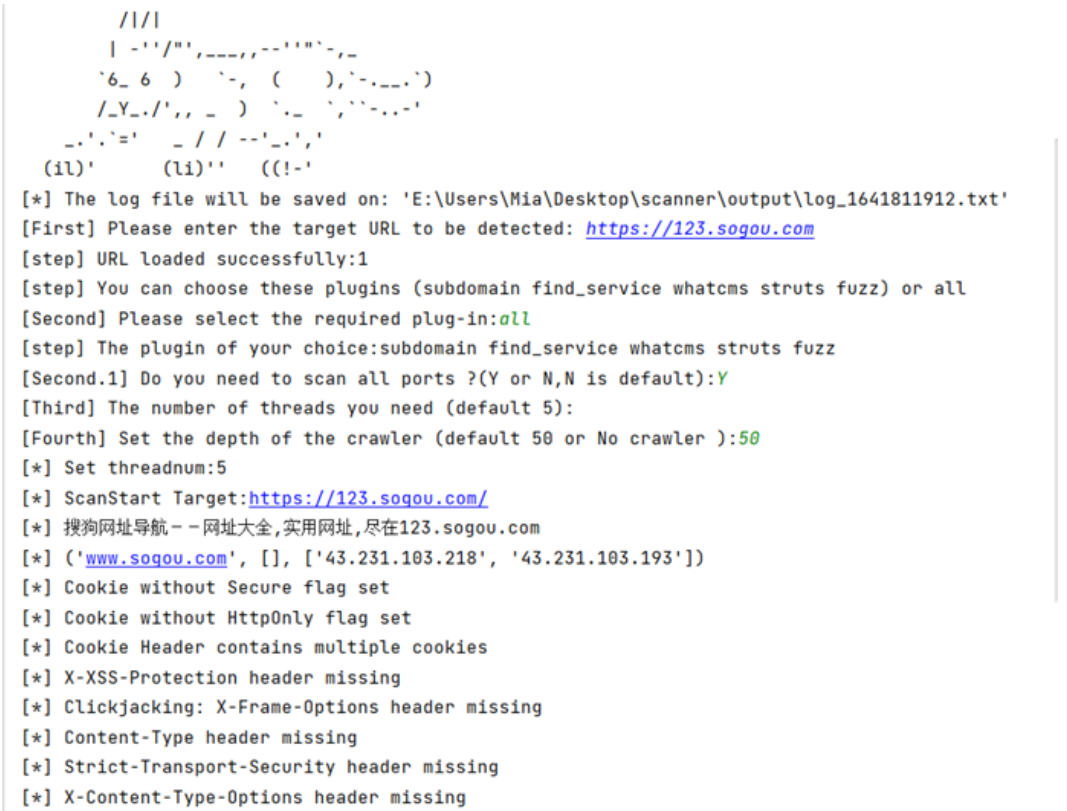
4.2 Functional Test

Enter scan.py -h in the python2.7 environment to display the command-line information guidance instructions for the Web vulnerability detection analyzer to scan vulnerabilities. The command-line help information includes system plug-in information, usage guidelines, and scan parameters.

During the test, the system scanned more than 100 websites to check whether the scanning function of the system is normal. After entering the target URL website, after setting the selected command parameters for information collection, port scanning, number of threads, and crawler traversal, there is no abnormality on the page. In the scanning interface, input parameters such as subdomain collection, fuzzing, and fingerprint detection, and no errors are reported during the scanning process. Through the output log records and HTML reports, it can be seen that each module of the system can work concurrently with the engine, and the user's scanning interface is shown in Figure 6.

The HTML report normally displays the website vulnerability information scanned by the Web Vulnerability Detection Analyzer. The displayed information includes the security level of the tested website, the SQL injection vulnerability module, the XSS vulnerability module, and the total scanning time. The HTML report can be tested. The output information does not display errors, and the page

Figure 6. Scan Interface



is concise, which is convenient for users to detect and analyze the vulnerabilities of the website. The information logged during the scanning of the website is also displayed normally. In the output HTML report, the layout is not disordered. The SQL injection and XSS vulnerability information displayed by the website's vulnerability information needs to be manually detected and analyzed. Some of the vulnerability information in the output HTML report has false positives.

During the process of scanning the detected target URL by the vulnerability detection analyzer, the system will record the information of the plug-in loaded during the scanning process, and the information is recorded according to the plug-in selected by the user. In the displayed result report, you can see that subdomain collection can bypass pan-resolution, and obtain the subdomain name and corresponding IP address of the target website.

Subdomain collection is shown in Figure 7.

4.3 Test Summary

After module design and testing, the vulnerability detection analyzer can scan out some vulnerabilities existing in the website, and the information of the vulnerabilities existing in the website can be seen in the output vulnerability scanning report. In the process of scanning for vulnerabilities, most of the functions of penetration testing are included, and each function can be performed normally. Backup files can be searched based on crawlers, subdomain scans, fingerprinted services, and port scans can all run normally. This article has tested hundreds of websites for this Web vulnerability detection analyzer. The scale, security policy and framework of each website are different. The total time scanned by the system and the accuracy of the vulnerability results are different, and there are great differences. sex. According to the number of split-screen pages and page views of the website, the more than 100 websites tested are divided into large websites, medium websites and small websites, and the three websites of different scales are scanned respectively, and the vulnerabilities of the websites of three different scales are counted. The average total scanning time and the average result accuracy, the Web vulnerability information displayed on the manual verification report, and the vulnerability results of the website after manual verification, the false positive rate of vulnerability scanning is about 55.96%.

The Web vulnerability detection analyzer in this paper is developed and improved based on the framework of the BugScan scanner. Therefore, the efficiency of scanning websites and the accuracy of the scanning results of the vulnerability detection analyzer and the BugScan scanner are also compared during the testing process. The difference between this system and the BugScan scanner is that a plug-in concurrency engine and a thread pool are added. The website vulnerability results are different. All plug-ins and the same number of threads are selected for vulnerability scanning.

Figure 7. Subdomain collection

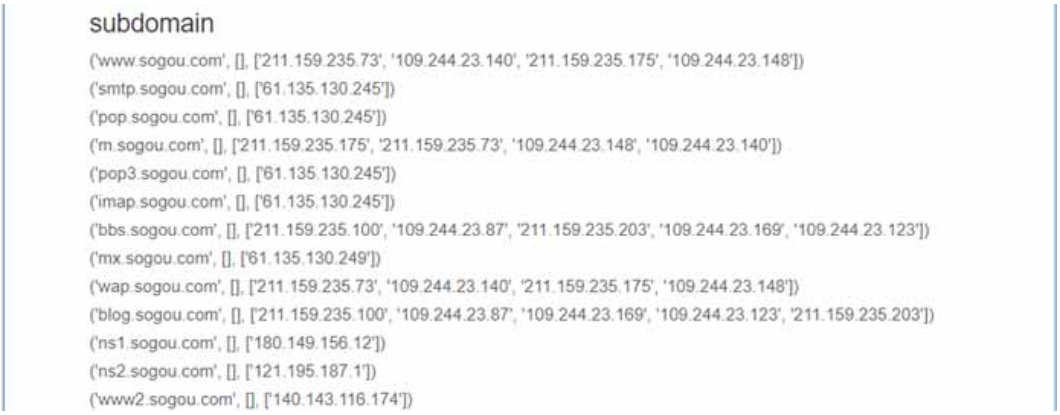
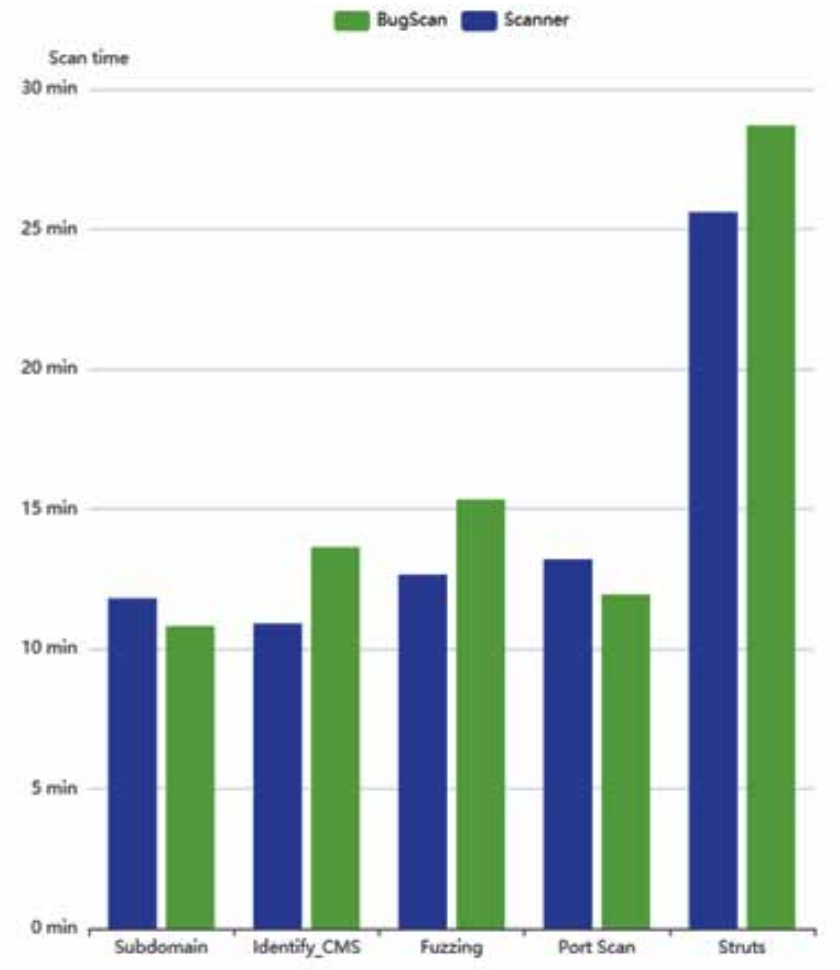


Table 1. Comparison of the test results

Result	Type	Large-sized website	Medium-sized website	Small-sized website	Average
Accuracy	**Vulnerability**	34.80	37.91	49.82	40.84
	Note	51.86	58.60	66.30	58.92
	Warning	51.67	57.87	60.11	56.55
	Information	55.30	69.90	77.43	67.54
Scan time /min	**Subdomain**	10.34	14.55	7.54	10.81
	Identify_CMS	15.1	8.97	4.82	9.63
	Fuzzing	20.34	7.9	5.8	11.34
	Port Scan	19.66	12.71	9.45	13.94
	Total time	65.44	44.13	27.61	45.72

Figure 8. Comparison of results



According to the vulnerability results report of each website, the scanning time of the two scanners is used to compare the scanning efficient of the two scanners.

5. CONCLUSION

The web vulnerability detection analyzer in this topic can scan the website for security vulnerabilities, and the built-in plug-in can detect web vulnerabilities through the information collection of the website by the crawler, including web fingerprint detection, port fingerprint detection, SQL injection detection, cross-site scripting After the detection of the website is completed, the result report in HTML format will be generated and stored in the file. By using the Web Vulnerability Detection Analyzer, the person in charge of the website can assess the security risk of the website based on the result report to find the Web vulnerability existing in the website, repair the existing Web vulnerability and modify the wrong settings in the website in time, and take precautions before the hacker attack., to avoid web vulnerabilities from damaging the information assets security of the website.

The system can realize the functions of four modules: scanning interface, information collection, vulnerability verification, and report generation, allowing users to detect and analyze the vulnerability information of the website. The system uses Python as the development language, and the project's operating environment is python2. 7. It does not need to rely on third-party libraries, improves the portability of the application, can realize cross-platform use, and is a universal plug-in compatible with the application. The information collection function of the system is relatively powerful, and a lot of information such as website services, operating systems, website frameworks and website files can be collected through plug-ins such as subdomain collection, port detection, fingerprint detection, and crawler, which are conducive to the detection of website vulnerabilities. In order to facilitate the system upgrade, the project uses a universal script extension tool to improve the PoC vulnerability script of the system.

However, there are still some deficiencies in this system. Since the vulnerability plug-ins in the project do not use third-party libraries, it is necessary to manually update the vulnerabilities disclosed on the Internet, and comprehensive vulnerability coverage is not achieved. The system's crawler is relatively basic. It uses the built-in plug-in to crawl the URL of the website. It does not call the JavaScript of the web page and the crawler in the website source code, and does not obtain enough URL pages. In addition, the system outputs the website in the HTML vulnerability report. The false positive rate of vulnerabilities exists, and these problems still need to be improved.

ACKNOWLEDGMENT

This research was supported by the Scientific research project of Education Department of Jilin Province (NO. JJKH20220602KJ).

REFERENCES

Bai, W., Jun, Y., & Zhao, Y. (2021). Analysis and discussion of XSS vulnerabilities. *Electronic World*, (20), 89–91. doi:10.19353/j.cnki.dzsj.2021.20.039

Bin, Z. (2019). *Research on the key technology of automatic mining and verification of software vulnerabilities* (Doctoral dissertation, National University of Defense Technology). https://kns-cnki-net-443.webvpn.jnu.edu.cn/KCMS /detail/detail.aspx?dbname=CDFDLAST2021&filename=1020386187.nh

Deng. (2020). Analysis of WEB penetration information collection. *Electronic Components and Information Technology*, (4), 24-25+32. .10.19772/j.cnki.2096-4455.2020.4.009

Hao, Z. (2018). *Design and Implementation of Web Application Vulnerability Scanner Based on Penetration Technology* (Master's Thesis, Donghua University). https://kns-cnki-net-443.webvpn.jnu.edu.cn/KCMS/detail/detail.aspx?dbname=CMFD201901&filename=1018839536.nh

Hou, M. (2018). *Research and implementation of network scanning technology based on intelligent crawling algorithm* (Master's thesis, Xidian University). https://kns-cnki-net-443.webvpn.jnu.edu.cn/KCMS/detail/detail.aspx?dbname=CMFD201901&filename=1019017938.nh

Lang, Feng, & Dong. (2017). Talking about the information collection of Web penetration testing. *Computer Age*, (8), 13-16. doi: .00410.16644/j.cnki.cn33-1094/tp.2017.08

Li. (2021). Web Application Penetration Testing Analysis. *Network Security Technology and Application*, (3), 5-6.

Li, L. (2007). *Design and Implementation of Network Vulnerability Scanner* (Master's Thesis, Xidian University). https://kns-cnki-net-443.webvpn.jnu.edu.cn/KCMS/detail/detail.aspx?dbname=CMFD2007&filename=2007049488.nh

Li. (2020). Research on SQL Injection Based on Web Penetration Testing. *Information and Computers (Theoretical Edition),* (3), 164-166.

Lian & Liu. (2017). Design and development of information collection tools for penetration testing. *Journal of Anhui Vocational College of Electronics and Information Technology*, (1), 30-34.

Meng, Lu, Liu, & Gao. (2020). Design and implementation of automatic proxy Web vulnerability scanner based on Python. *Science and Technology Vision*, (17), 41-45.

Meng, Zhiyun, & Zhong. (2021). Research on Port Scanning Technology Based on Python. *Network Security Technology and Application*, (1), 42-43.

Mo & Li. (2019). Research on Web Penetration Testing Information Collection Technology. *Communication World*, (3), 33-34.

Rao, Sun, Shi, & Meng. (2020). Research and Application of Web Application Vulnerability Detection System Based on Fingerprint Identification Technology. *Information and Communication*, (11), 97-100.

Ren, Zheng, Zhang, Wang, Feng, Wang, & Zhang. (2021). Overview of Fuzzing Testing Technology. *Computer Research and Development*, (5), 944-963.

Sun, Y., Zhen, L., & Liang, S. (2020). Research on Penetration Testing Technology of Hierarchical Protection Mobile Application Software. *Proceedings of 2020 China Conference on Hierarchical Protection of Network Security and Protection of Critical Information Infrastructure*, 43-47+68.

Teng. (2020). Discussion on the Countermeasures of Enterprise Information Security Management in the Network Environment. *Network Security Technology and Application*, (6), 124-125.

Tong, Yao, Liang, Wang, & Zhou. (2020). Application research based on cross-browser fingerprint recognition technology. *Network Security Technology and Application,* (11), 64-67.

Yang. (2019). Research status and development trend of information security at home and abroad. *Network Security Technology and Application*, (5), 1-2.

Zhang, Ren, & Wang. (2020). A review of research on network security risk assessment analysis methods. *Journal of Yanshan University*, (3), 290-305.

Zhou, S. (2020). *Web fingerprint recognition analysis research* (Master's degree thesis, Chongqing University of Posts and Telecommunications). https://kns-cnki-net-443.webvpn.jnu.edu.cn/KCMS/detail/detail.aspx?dbname=CMFD202101&filename=1020416916.nh

Dawei Xu is a Ph.D. student in the School of Cyberspace Science and Technology, Beijing Institute of Technology. He engages in scientific research and education work in the College of Cybersecurity, Changchun University. His current research interests include blockchain technology, anonymous communication, big data privacy protection, and machine learning.

Tianxin Chen is a graduate student with a major in cyberspace security, School of Changchun University. Her current research interest is blockchain and network security.

Zhonghua Tan is a lecturer at the School of International Education, Hainan Normal University. His research interest is cyberlinguistics.

Fudong Wu is a graduate student with a major in cyberspace security, School of Changchun University. His research interests include blockchain technology, and anonymous communication.

Jiaqi Gao is a graduate student majoring in cyberspace security, Changchun University. His research interests include blockchain technology, and anonymous communication.

Yunfan Yang is a graduate student with a major in cyberspace security, School of Changchun University. Her current research interest is machine learning.