CSE 3024 - Web Mining

Digital Assignment - I

Web Crawling and Vulnerability Detection

By

20BCE1421	Krishna Prasad Y V S Purama
20BCE1309	Santhosh Narayanan B
20BCE1844	Tathagata Biswas

B.Tech CSE (Core)

Submitted to

Dr.A.Bhuvaneswari,
Assistant Professor Senior,
SCOPE, VIT, Chennai

School of Computer Science and Engineering

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

January 2023

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computing Science and Engineering

VIT Chennai

Vandalur - Kelambakkam Road, Chennai - 600 127

WINTER SEM 22-23

Worklet details

Programme	B.Tech CSE (Core)
Course Name / Code	Computer Science and Engineering
Slot	A2 + TA2
Faculty Name	Dr. A. Bhuvaneswari
J Component Title	Web Crawling and Vulnerability Detection
Team Members Name | Reg. No	Krishna Prasad Y V S Purama — 20BCE1421; Santhosh Narayanan B — 20BCE1309; Tathagata Biswas — 20BCE1844

Krishna Prasad Y V S Purama	20BCE1421
Santhosh Narayanan B	20BCE1309
Tathagata Biswas	20BCE1844

Team Members(s) Contributions – Tentatively planned for implementation:

Worklet Tasks	***Contributor's Names***
Research analysis	Krishna Prasad Y V S Purama
Algorithm design	Tathagata Biswas
Evaluation and testing	Krishna Prasad Y V S Purama
Deployment	Santhosh Narayanan B
Technical Report writing	Tathagata Biswas
Presentation preparation	Santhosh Narayanan B

ABSTRACT

The project intends to create a web crawling and vulnerability detection system, to assist businesses in securing their websites and safeguarding the data of their users . The system scans and analyzes webpages for various sorts of vulnerabilities using a combination of automatic and manual procedures while being programmed in Python. These flaws might involve regular expression DoS flaws, SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), remote file inclusion vulnerability (RFI), and other popular web-based attack vectors.

The idea for our project was inspired by the open-source programme SQLMap, which automates the process of finding and exploiting SQL injection flaws in online applications. It makes it possible for penetration testers and security researchers to conduct security analyses of web applications with SQL databases as the backend and find flaws.

The system begins by crawling the target website to gather pertinent data, including URLs, input forms, and other significant information. The website is then thoroughly scanned using this data to look for any potential security issues. In the event that a vulnerability is found, the system generates a thorough report with information on the type of vulnerability, its location, and the possible effects it may have on the website and its visitors.

The scans' findings are provided to website owners in a straightforward and comprehensive way, making it simple for them to comprehend the dangers and take the necessary precautions to resolve them. The initiative offers businesses a practical means of enhancing the safety of their websites and safeguarding the information of their visitors. The method may be used on a daily basis to continually check for vulnerabilities on the website and monitor its security.

1. Introduction

An increase in the number of websites and online applications that are susceptible to security risks like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF) has been caused by people's increased dependence on the internet. The reputation of firms may suffer significantly as a result of these security risks, which may lead to the loss of sensitive data including personal and financial information.

The goal of this project is to create a system that can automatically scan websites and find any security flaws that might endanger the users and data on the website. In order to make it simple for website owners to comprehend the dangers and take the necessary steps to address them, the system should be able to do a thorough scan of the website and show the results in a clear and straightforward manner.

The creation of a system for web crawling and vulnerability detection is not without difficulties. Some of the challenges faced in this project include:

- The capacity to precisely detect and exploit numerous sorts of vulnerabilities, such as SQL injection and XSS, is one of the difficulties encountered in this project.
- The requirement for ongoing system monitoring and updating to make sure it can recognise newly identified vulnerabilities and assaults.
- The findings of the scans must be presented in a clear and succinct way so that website owners can quickly comprehend the dangers and take the necessary steps to resolve them.
- The system's capacity to scale to accommodate a large number of websites and manage enormous volumes of data.

In conclusion, creating a web crawling and vulnerability detection system is a complicated and difficult process that calls for both technological proficiency and a solid grasp of online security. This project seeks to offer an efficient solution for businesses wishing to enhance the security of their websites and safeguard the data of their customers by using Python as the programming language and utilizing some of its automatic procedures.

2. Literature Survey

Sl no	Title	Author / Journal name / Year	Technique	Result
1	Deep Learning Algorithm for Threat Detection in Hackers Forum (Deep Web)	Victor Adewopo, Bilal Gonen, Nelly Elsayed, Murat Ozer and Zaghloul Saad Elsayed. arXiv 2022	Deep Learning, Threat Prediction.	Identify existing security threats. Predict potential vulnerabilities.
2	Deep Learning for Vulnerability and Attack Detection on Web Applications	Rokia Lamrani Alaoui and El Habib Nfaoui. mdpi 2022	Deep Learning.	Explore advanced DL models in the field of web attacks detection Secure learning
3	Design and Implementation of Dynamic and Efficient Web Crawler for XSS Vulnerability Detection	Ao Chai Atlantis Press 2017	Dynamic parallel crawler system.	The XSS vulnerability detection tool has certain advantages in scanning websites, which involves speed and system resource occupancy.
4	A Web Second - Order Vulnerabilities Detection Method	Miao Liu and Bin Wang. IEEE 2018	Penetration testing, second-order vulnerability detection.	The presented algorithm can detect second-order web vulnerability.
5	Design and Implementation of an Automatic Scanning Tool of SQL Injection Vulnerability Based on Web Crawler	Xiaochun Lei, Jiashi Qu, Gang Yao, Junyan Chen & Xin Shen. Springer 2019	Web Crawler, SQL injection, Automatic scanning	Automatic detection tool for SQL injection vulnerability based on web crawler

6	Web Vulnerability Detection Analyzer Based on Python	Dawei Xu, Tianxin Chen, Zhonghua Tan, Fudong Wu, Jiaqi Gao, and Yunfan Yang. IGI Global 2022	Penetration testing, Vulnerability detection, Crawler based Scan	Website and its vulnerability classification.
7	Design and Implementation of High-performance Web Vulnerability Scanner Based on Python Intelligent Crawler	Jianxun Tang; Fang Zhou IEEE 2021	CMS recognition, port, plug in technology.	High accuracy and scalability.
8	Design and Implementation of Security Vulnerability Sharing Platform Based on Web Crawler	Zhiqiang Wang, Ziyi Wang, Zhuoyue Wang, Zhirui Zhang & Tao Yang Springer 2021	Web crawler, Vulnerability detection, Network security tools.	High automation and low cost in labor compared to traditional techniques.
9	Evaluation of Black-Box Web Application Security Scanners in Detecting Injection Vulnerabilities	Muzun Althunayyan, Neetesh Saxena , Shancang Li and Prosanta Gope mdpi 2022	Injection vulnerability	Scanners have a limited ability to crawl dynamic modern web applications. Thus, providing scanners with a proxy component helps them detect more existing vulnerabilities
10	Web Crawler: A Review	Md. Abu Kausar, V. S. Dhaka and Sanjeev Kumar Singh IJCA 2013	Web Crawler Survey, Search engine, Parallel Crawler	Web Crawler is thus vital information retrieval which traverses the Web and downloads web documents that suit the user's need.

11	Vulnerability Detection with Deep Learning on a Natural Codebase for Python	Laura Wartschinki, Yannic Noller, Thomas Vogel, Timo Kehrer and Lars Grunske Elsevier 2022	Deep Learning, Vulnerability Detection.	Better at vulnerability detection.
12	A Static Analysis Tool for Detecting Security Vulnerabilities in Python Web Applications	Stefan Micheelsen and Bruno Thalmann mdpi 2016	Static Analysis, Web Application Security	Specialized to find vulnerabilities in Flask using a modified reaching definitions analysis, but it is the intention that both aspects can be expanded on.

3. **Dataset and Tool to be used**:

Python: With its simplicity and high-level programming capabilities, we utilize Python to create systems for web crawls and vulnerability detection. We use numerous libraries and modules that it provides, such as Scrapy and BeautifulSoup, which may be used for site crawling and data extraction. Furthermore, Python's regular expression processing capabilities can be used to search and extract critical data from websites, such as URLs, input fields, and other information that can be used to identify security issues.

Requests: The requests package is in Python for making HTTP requests. It hides the difficulties of making requests behind a beautiful, simple API, allowing you to concentrate on interacting with services and consuming data in your application.

re: This is the Regular Expression library in Python. It is used to find all the <a> tags in the HTML content, which represent the links on the website.

socket: This is a low-level library for working with sockets, which are a common way of establishing connections between two systems on a network. In the code, it is used to resolve the host name of the website to an IP address.

nmap: This is a library for using the Nmap Security Scanner, which is used to scan networks and hosts for open ports. In the code, it is used to scan the IP address of the website and find any open ports on the host.

set: This is a built-in data structure in Python that represents a collection of unique elements. In the code, it is used to keep track of the URLs that have already been visited during the website crawl, to avoid visiting the same URL multiple times.

BeautifulSoup: Beautiful Soup is a Python package used for web scraping to extract data from HTML and XML files. It generates a parse tree from the page source code, which can be used to extract data in a more hierarchical and legible format.

4. Algorithms / Techniques description

Pseudocode:

```
import requests, re, socket, nmap, BeautifulSoup

#function to crawl website:

def  crawl_website(url,n)
    visited = set[]
    to_visit = [url]
    level=0
    while to_visit and level <= n
        current_url = to_visit.pop(0)
        visited.add(current_url)
        try:
            response = requests.get(current_url)
            soup = BeautifulSoup(response.text, "html.parser")
        level += 1
    return visited


#function to detect sql vulnerabilities:

def detect_sql_injection(url)
    payloads = [] #all types of sql injection payloads
    for payload in payloads
        injection_url =  url + payload
        try:
            response = requests.get(injection_url)
        if  error
            return "sql vulnerability in URL"
        else
            return "no sql vulnerability"
```

#function to detect open ports:

```
def detect_open_ports(url):
        host = url.split("//")[1].split("/")[0]
        try:
                ip = socket.gethostbyname(host)
                nm = nmap.PortScanner()
                nm.scan(ip, arguments="-T4 -F")
                open_ports = [
                        str(port) + "/" + nm[ip]["tcp"][port]["name"]
                        for port in nm[ip].all_tcp()
        i               if nm[ip]["tcp"][port]["state"] == "open"
                        ]
                if  open_ports
                        return "Open ports in URL"
                else
                        return "no open ports"
```

#function to detect XSS vulnerabilities:

```
def detect_xss(urls):
   for url in urls:
        response = requests.get(url)
        if "<script>" in response.text.lower():
           return "XSS vulnerability in url"
```

#function to detect remote file inclusion:

```
def detect_remote_file_inclusion(urls):
   for url in urls
        response = requests.get(url)
        if (
           "http://" in response.text.lower()
            or "https://" in response.text.lower()
        )
           return "Possible RFI vulnerability in url"
```

#function to detect regular expression dos:

```
def detect_regular_expression_dos(urls):
   for url in urls
        response = requests.get(url)
        if re.search(r"(a+){10,}", response.text)
           return "Possible regular expression DoS vulnerability in url"
```

Techniques:

crawl website:
A function that can "crawl" or traverse through a website and all of its pages, which is commonly used for indexing the website for search engines or assessing its structure and contents.

detect sql injection:
A function for detecting SQL injection vulnerabilities in a website. SQL injection is a sort of security hack in which an attacker injects malicious SQL code into a website's database, potentially gaining access to sensitive data.

detect open ports:
A function that scans a target system to determine whether network ports are open and accepting connections. This data can be used to establish which services are operating on the system and to discover potential vulnerabilities.

detect XSS:
A function for detecting cross-site scripting (XSS) vulnerabilities on a website. XSS is a sort of security vulnerability in which an attacker injects malicious JavaScript code into a website, which is then executed by unwitting visitors.

detect remote file inclusion:
A function that detects vulnerabilities in a website's Remote File Inclusion (RFI). RFI is a sort of security exploit in which an attacker can add remote files from a third-party website, possibly jeopardizing the targeted website's security.

detect regular expression:
A function for detecting regular expressions in strings. A regular expression is a string of letters that establishes a search pattern and is used for activities like pattern matching, data extraction, and text manipulation. This function can be used to check user input, extract data from text, and so on.

5. Github Repository Link:

https://github.com/krishnaprasad12/Web-Mining-DA

REFERENCES

[1] Lei, X., Qu, J., Yao, G., Chen, J., Shen, X. (2020). Design and Implementation of an Automatic Scanning Tool of SQL Injection Vulnerability Based on Web Crawler. In: Yang, CN., Peng, SL., Jain, L. (eds) Security with Intelligent Computing and Big-data Services. SICBS 2018. Advances in Intelligent Systems and Computing, vol 895. Springer, Cham.

[2] B. Wang, L. Liu, F. Li, J. Zhang, T. Chen and Z. Zou, "Research on Web Application Security Vulnerability Scanning Technology," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 2019, pp. 1524-1528, doi: 10.1109/IAEAC47372.2019.8997964.

[3] Wang, Z., Wang, Z., Wang, Z., Zhang, Z., Yang, T. (2022). Design and Implementation of Security Vulnerability Sharing Platform Based on Web Crawler. In: Liu, Q., Liu, X., Chen, B., Zhang, Y., Peng, J. (eds) Proceedings of the 11th International Conference on Computer Engineering and Networks. Lecture Notes in Electrical Engineering, vol 808. Springer, Singapore.

[4] E. İ. Tatli and B. Urgun, "WIVET—Benchmarking Coverage Qualities of Web Crawlers," in The Computer Journal, vol. 60, no. 4, pp. 555-572, March 2017, doi: 10.1093/comjnl/bxw072.

[5] C. Saini and V. Arora, "Information retrieval in web crawling: A survey," 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, India, 2016, pp. 2635-2643, doi: 10.1109/ICACCI.2016.7732456.

[6] S. Shiaeles, N. Kolokotronis and E. Bellini, "IoT Vulnerability Data Crawling and Analysis," 2019 IEEE World Congress on Services (SERVICES), Milan, Italy, 2019, pp. 78-83, doi: 10.1109/SERVICES.2019.00028.

[7] K. J. Koswara and Y. Dwi Wardhana Asnar, "Improving Vulnerability Scanner Performance in Detecting AJAX Application Vulnerabilities," 2019 International Conference on Data and Software Engineering (ICoDSE), Pontianak, Indonesia, 2019, pp. 1-5, doi: 10.1109/ICoDSE48700.2019.9092613.

[8] L. Chen, J. Li and B. Zhang, "Intelligent Penetration Technology of Power Web System Vulnerability Based on Deep Learning," 2021 International Conference on Computer, Blockchain and Financial Development (CBFD), Nanjing, China, 2021, pp. 209-213, doi: 10.1109/CBFD52659.2021.00048.

[9] Peng, S., Liu, P. & Han, J. A Python Security Analysis Framework in Integrity Verification and Vulnerability Detection. Wuhan Univ. J. Nat. Sci. 24, 141–148 (2019).

[10] P. Lewandowski, M. Janiszewski and A. Felkner, "SpiderTrap—An Innovative Approach to Analyze Activity of Internet Bots on a Website," in IEEE Access, vol. 8, pp. 141292-141309, 2020, doi: 10.1109/ACCESS.2020.3012969.