Project Report

on

Advanced Web Penetration Testing Tool

Submitted by

Project Members

Indraneel Tilloo (1032191334)

Nirmal Vyas (1032191456)

Akshit Langeh (1032192137)

Nabil Barbhuyan (1032190871)

Under the Guidance of

Dr. Vinayak Musale

Under the External Guidance of ***Mr. Prateek Kuber (HACK-X Security)***

School of Computer Engineering and Technology

Dr. Vishwanath Karad MIT World Peace University, Kothrud,

Pune 411 038, Maharashtra - India

2022-2023

MIT-WPU
॥ विश्वशान्तिर्ध्रुवं ध्रुवा ॥

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

SCHOOL OF COMPUTER ENGINEERING AND TECHNOLOGY

C E R T I F I C A T E

This is to certify that,

Indraneel Tilloo

Nirmal Vyas

Akshit Langeh

Nabil Barbhuyan

of BTech. (Computer Science & Engineering) have completed their project titled "*Advanced Web Penetration Testing Tool*" and have submitted this Capstone Project Report towards fulfillment of the requirement for the Bachelor of Computer Science and Engineering (BTech-CSE) for the academic year 2022-2023.

[Dr. Vinayak Musale]
Project Guide
School of CET
MIT World Peace University, Pune

[Dr. Vrushali Kulkarni]
Program Head
School of CET
MIT World Peace University, Pune

Internal Examiner : _____

External Examiner : _____

Date:

Acknowledgement

Firstly, we would like to thank my university Dr. Vishwanath Karad MIT World Peace University for giving us this opportunity to do a project as a part of the academic curriculum which helped us in getting a lot of experience.

We would also like to thank the School of Computer Engineering and Technology (SCET) and the head of the school Dr. Vrushali Kulkarni for providing us with all the necessary infrastructure and facilities needed for the project.

Also, we are highly indebted and would like to acknowledge and give our warmest thanks to our guide Dr. Vinayak Musale who made the accomplishment of this project possible. His valuable guidance, support and advice carried us through all the steps of the project.

We also acknowledge and thank Mr. Yash Vyas and Mr. Prateek Kuber for their guidance throughout the project. I appreciate the confidence you gave us and for always being there to clear all our doubts.

Name of the Students:

Indraneel Tilloo (1032191334)

Nirmal Vyas (1032191456)

Akshit Langeh (1032192137)

Nabil Barbhuyan (1032190871)

Abstract

Penetration testing is a critical component in ensuring the security of digital assets. It entails simulating an assault on a system in order to find weaknesses that attackers could exploit. It is critical to have a tool that can detect many sorts of vulnerabilities and automate the process of detecting risk factors in order to conduct efficient penetration testing.

The purpose of this project is to develop an advanced penetration testing tool capable of detecting feature and endpoint vulnerabilities, automating the process of identifying potential risks, and improving testing efficiency.

The tool will include feature detection and endpoint detection capabilities, which will allow it to identify potential vulnerabilities and risks connected with a certain asset. It will also include CVSS (Common Vulnerability Scoring System) automation to help prioritize vulnerabilities based on severity. This will allow security teams to prioritize the most serious vulnerabilities.

The creation of an advanced penetration testing tool will have several advantages. To begin with, it will eliminate human error from the testing process, resulting in better outcomes. Second, it will increase testing efficiency, allowing security professionals to perform more tests in less time. Ultimately, it will enable security professionals to more correctly detect potential risks and vulnerabilities, resulting in improved digital asset security.

Keywords - Common Vulnerability Scoring System, Endpoint Detection, Feature Detection, Penetration Testing, VAPT.

List of Figures

List of Tables

List of Abbreviations

VAPT - Vulnerability Assessment and Penetration Testing

CVSS - Common Vulnerability Scoring System

SQL - Structured Query Language

SSTI - Server Side Template Injection

XSS - Cross Site Scripting

CRLF - Carriage Return and Line Feed

SIEM - Security information and event management

SOAR - Security orchestration, automation, and response

AV - Attack Vector

AC - Attack Complexity

PR - Privilege Requirement

UI - User Interaction

SVM - Support Vector Machines

SVR - Support Vector Regression

RFR - Random Forest Regression

MSE - Mean Squared Error

Contents

Appendices

Chapter 1
Introduction

1.1 Project Statement

ADVANCED WEB PENETRATION TOOL

To create an advanced web penetration testing tool. It would include Feature detection, Endpoint detection, CVSS automation and would consist of detecting various injections and vulnerabilities.

1.2 Project Domain

Cybersecurity, Vulnerability Assessment and Penetration Testing

1.3 Motivation

The need for vulnerability assessment and penetration testing arises due to the increasing sophistication and frequency of cyberattacks that can potentially compromise an organization's digital assets, including sensitive data, financial resources, reputation, and legal compliance. Our project’s primary objective is to automate the VAPT process thus reducing the testing tool and also try to eliminate the human errors that occur while testing.

1.4 Project Introduction and Aim

Penetration testing is essential to ensure the security of digital assets. The need for penetration testing arises due to the increasing sophistication of cyberattacks, which target vulnerabilities in an organization's systems, software, or network. These attacks can result in the loss of sensitive data, financial loss, damage to reputation, and legal repercussions.

Vulnerability Assessment and Penetration Testing (VAPT) are proactive approaches that help organizations identify potential security risks before they can be exploited by attackers. VAPT identifies weaknesses in the system, network, or software and provides a detailed report on the vulnerabilities found, along with recommendations to address them.

Although web applications can be secured in a variety of ways, the Vulnerability Assessment and Penetration Testing (VAPT) methodology is a unique technique for protecting web applications from both logical and technological vulnerabilities. This approach can be used to audit web application security and also secure connected layers. VAPT entails auditing the system for vulnerabilities that may exist, exploiting flaws in the same way that an attacker would, and producing data that represents the system's risk rating.

Penetration testing involves simulating a real-world attack on an organization's systems and applications to test the effectiveness of existing security measures. It helps in identifying weaknesses in the security architecture and validates the effectiveness of existing security controls. The need for penetration testing and VAPT is essential for organizations to stay ahead of cyber threats, secure their digital assets, and maintain the trust of their stakeholders. A proactive approach to security testing can prevent attacks and minimize the impact of any potential security breaches.

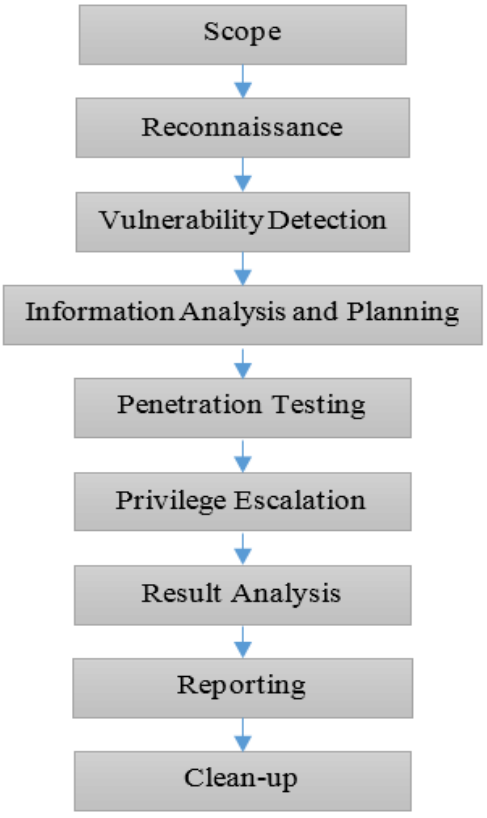


Fig 1.1. VAPT Lifecycle

Chapter 2
Literature Survey

VAPT, which stands for Vulnerability Assessment and Penetration Testing, is a process of identifying and evaluating the vulnerabilities in a system or network and attempting to exploit them to assess the level of security of the system. The process involves two main components: vulnerability assessment, which is the process of identifying vulnerabilities, and penetration testing, which involves attempting to exploit these vulnerabilities to gain unauthorized access to the system or network. The Vulnerability Assessment and Penetration Testing (VAPT) process is a process which has evolved over the years.

The history of VAPT can be traced back to the early days of computer security when hackers were first attempting to penetrate computer systems. At that time, security professionals began developing techniques to identify and address vulnerabilities in their systems. As technology has evolved, so has the VAPT process. Today, VAPT is an essential component of any comprehensive security program.

The importance of VAPT lies in its ability to identify and address vulnerabilities before they can be exploited by attackers. By proactively testing systems and networks, organizations can detect and remediate vulnerabilities before they are exploited, minimizing the risk of a security breach. This is particularly important in today's increasingly complex and interconnected world, where cyberattacks are becoming more sophisticated and frequent.

The authors discuss and analyze the life cycle of the VAPT process and VAPT tools for finding vulnerabilities in a system. Although the paper thoroughly explains all the injections and scopes, it is just an overview and no actual testing has been done[3].

The main task is to identify several challenges and limitations associated with security testing of web applications[4], such as the lack of a comprehensive testing framework and the limited effectiveness of automated testing tools. But it lacks analysis on the effectiveness and efficiency of different security testing techniques for web applications.

Below is the table that shows the literature review of the papers we reviewed and understood what the existing approaches are present and what are the research gaps associated with them -

Table 2.1. Literature Survey of relevant research papers

Sr. No.	Paper Title	Publication Details	Existing Approaches	Research Gap
1	Perusal of web application security approach	2017, International Conference on Intelligent Communication and Computational Techniques (ICCT)	The paper discusses a comprehensive overview of the various security approaches that can be used to protect web applications.	The paper does not address the most recent trends and advancements in web application security, which may limit its usability.
2	Web Application Safety by Penetration Testing	2018, International Journal of Advanced Studies of Scientific Research	The paper elaborates why the process of VAPT is important. In addition, various vulnerabilities have been discussed, giving us a general picture of how the testing is carried out.	The vulnerabilities have not been covered by going into depth. The research is just an overview and no actual testing has been done to find the efficiency of the tools.
3	Analysis and Impact of Vulnerability Assessment and Penetration Testing	2019, International Conference on Machine Learning, Big Data, Cloud and Parallel Computing	The paper discusses and analyzes the life cycle of the VAPT process and VAPT tools for finding vulnerabilities in a system.	Although the paper thoroughly explains all the injections and scopes, it is just an overview and no actual testing has been done.

4	Security testing of web applications: A systematic mapping of the literature	2022, Journal of King Saud University - Computer and Information Sciences	The authors identify several challenges and limitations associated with security testing of web applications, such as the lack of a comprehensive testing framework and the limited effectiveness of automated testing tools.	The paper lacks an analysis on the effectiveness and efficiency of different security testing techniques for web applications.
5	VAPT & Exploits, along with Classification of Exploits	2022, SSRG International Journal of Computer Science and Engineering	This paper helps us understand the working of remote attacks and the classification of exploits based on their type.	No clear algorithmic approach that detects the exploits as well as how the exploits work is mentioned.
6	Cybersecurity Threats and Vulnerabilities : Systematic Mapping Study	2020, Arabian Journal for Science and Engineering	The research aims to identify the relationship between the vulnerabilities and their frequency of occurrence.	The paper does not provide a detailed analysis to examine the quality, the consistency of the findings, or the limitations of the research.
7	An Empirical Comparison of Pen-Testing Tools for Detecting Web App Vulnerabilities	2022, New Advances and Challenges in Communication Networks	The design of a Framework for Evaluation Criteria aids in the analysis of each parameter.	The paper talks very less on the analysis of false positives and false negatives which can have significant implications for the efficiency and effectiveness.

8	A Deep Learning Approach For Classifying Vulnerability Descriptions Using Self Attention Based Neural Network	2021, Journal of Network and Systems Management	There is a proposed system utilizes a self-attention deep neural network (SA-DNN) model and text mining approach to identify the vulnerability category from the description text contained within a report	Inconsistencies and errors in the dataset/ reports
9	An automatic method for CVSS score prediction using vulnerabilities description	2016,Journal of Intelligent and Fuzzy Systems	This paper stated how with the help of various text mining techniques we can extract the feature vectors and then can predict CVSS score by using SVM, random forests and fuzzy systems.	The OSVDB database which consisted of CVSS scores that were used in this paper is not active anymore.
10	Using neural networks to aid CVSS risk aggregation - An empirically validated approach	2016, Journal of Innovation in Digital Ecosystems	A method for automated CVSS risk aggregation is proposed which also reduces ambiguity and variability in CVSS assessments	Large dataset will be required to train a neural network
11	Design of Web Security Penetration Test System Based on Attack and Defense Game	2022, Hindawi Scientific Programming Journal	The paper highlights an advanced and innovative methodology to web penetration testing with the help of a web crawler module and a breadth-first search strategy.	There is a requirement for a large dataset in order to implement the data layer.

Chapter 3

Problem Statement

3.1 Project Scope

The scope of the project is to automate the VAPT process. Automated Vulnerability Assessment and Penetration Testing (VAPT) can provide several benefits. Automated tools can ensure consistent testing across multiple systems and networks, perform testing more quickly and efficiently than manual testing, and scale up or down based on the size and complexity of the environment. They can also identify vulnerabilities more accurately and provide more comprehensive testing coverage, generating detailed reports with valuable insights into vulnerabilities and remediation actions. Overall, VAPT automation can save time and resources, while improving accuracy and effectiveness in identifying and addressing vulnerabilities.

3.2 Project Limitations

The VAPT process has several limitations, including the potential for false positives and false negatives generated by automated tools, incomplete testing due to time or resource constraints, human error in manual testing, limited coverage of potential vulnerabilities, lack of context, high cost, and potential impact on systems. Therefore, VAPT testing should be considered as a part of a comprehensive security program that includes multiple layers of defense, regular patching and updates, ongoing monitoring, and user awareness training.

3.3 Project Objectives

The objective of the project is to include feature detection and endpoint detection capabilities, which will allow it to identify potential vulnerabilities and risks connected with a certain asset. It will also include CVSS (Common Vulnerability Scoring System) automation to help prioritize vulnerabilities based on severity. This will allow security teams to prioritize the most serious vulnerabilities.

Chapter 4

Project Requirements

4.1 Resources

In this project we will be using programming languages such as python, bash, c++ to develop the injection scripts. Flask will be used to integrate the backend code with the UI. Feature Detection and Endpoint Detection algorithms are developed using python and selenium is used for web scraping. CVSS automation is carried out in the jupyter notebook with libraries like pandas, numpy and scikit learn.

4.2 Reusable Software Components

- Web Crawlers: These are software components that can be used to traverse web applications and extract relevant data, such as URLs, form inputs, and cookies. This can be done with the assistance of the feature and endpoint detection module
- Vulnerability Scanners: The scripts used to detect the injection vulnerabilities as mentioned in the problem statement can also be referenced to develop scripts to develop other injection vulnerabilities.
- Reporting Engine: It can be customized in accordance with other penetration testing tools to create reports in accordance to developer
- ML Libraries: The CVSS automation module can be used to generate vectors and score for other vulnerabilities and penetration testing tools
- Integrations with other Security Tools: These are software components that can be used to integrate the web penetration testing tool with other security tools, such as vulnerability management systems, SIEMs, or SOAR platforms.

4.3 Software and Hardware Requirements

- **Hardware:**
    1. Intel i3 8th generation and above
    2. RAM: 8GB (Minimum)
    3. Graphics Card: 4GB (Minimum)

- **Software:**
    1. Operating System: Open Source Linux Distribution
    2. Programming Languages: Python, Bash, C++
    3. IDE: Visual Studio, Jupyter Notebook
    4. Flask

4.4 Requirements Rationale

Table 4.1. Requirements Rationale

Robust Access Controls	Scripts can bypass weak access controls and can gain unauthorized access
Rigorous Testing	Injection scripts can exploit undiscovered vulnerabilities
Scalability	Scripts and algorithms can handle large scale websites efficiently
In depth Explanation	We also checked and compared each data entry

4.5 Risk Management

Table 4.2. Risk Factors

Factor	Cause	Impact	Chances
Insufficient Input Validation	Lack of input validation mechanisms	SQL injection, XSS attacks, Command Injection	High
Third Party Dependencies	Insecure or compromised dependencies	Malicious code injection, data manipulation	Medium
Unexpected Page Structure	Unexpected html structure of target website	Failure in extracting form features from the website	Medium
Data Size	Due to early development, there is lack of data	Impacts the overall accuracy of the CVSS automation module	Medium

4.6 Functional Specifications

- Scanning Capabilities: The tool should be able to scan a web application for vulnerabilities, including common vulnerabilities such as SQL injection, cross-site scripting (XSS), and CRLF vulnerabilities.
- Reporting and Analysis: The tool should be able to generate detailed reports that highlight the vulnerabilities detected, their severity, and recommendations for remediation. The reports should be easy to understand and navigate, with the ability to export the reports to various formats.
- User Interface: The tool should have an intuitive user interface that is easy to navigate and use.
- Performance: The tool should perform efficiently and accurately, with minimal false positives or false negatives.

Chapter 5
Proposed Architecture

5.1 Design Considerations

- Scanning Techniques: The tool must incorporate a range of scanning techniques, including vulnerability scanning, input detection, endpoint scanning to detect potential vulnerabilities in web applications. The tool should also be capable of detecting hidden vulnerabilities that are not easily discoverable through traditional scanning techniques.
- Machine Learning: The tool should leverage machine learning algorithms and models to predict the likelihood of a vulnerability being exploited and provide insights into the potential impact of a vulnerability on the system.
- Regular Updates: The tool should be regularly updated to ensure that it is up-to-date with the latest scanning and testing techniques and can effectively detect the latest vulnerabilities.

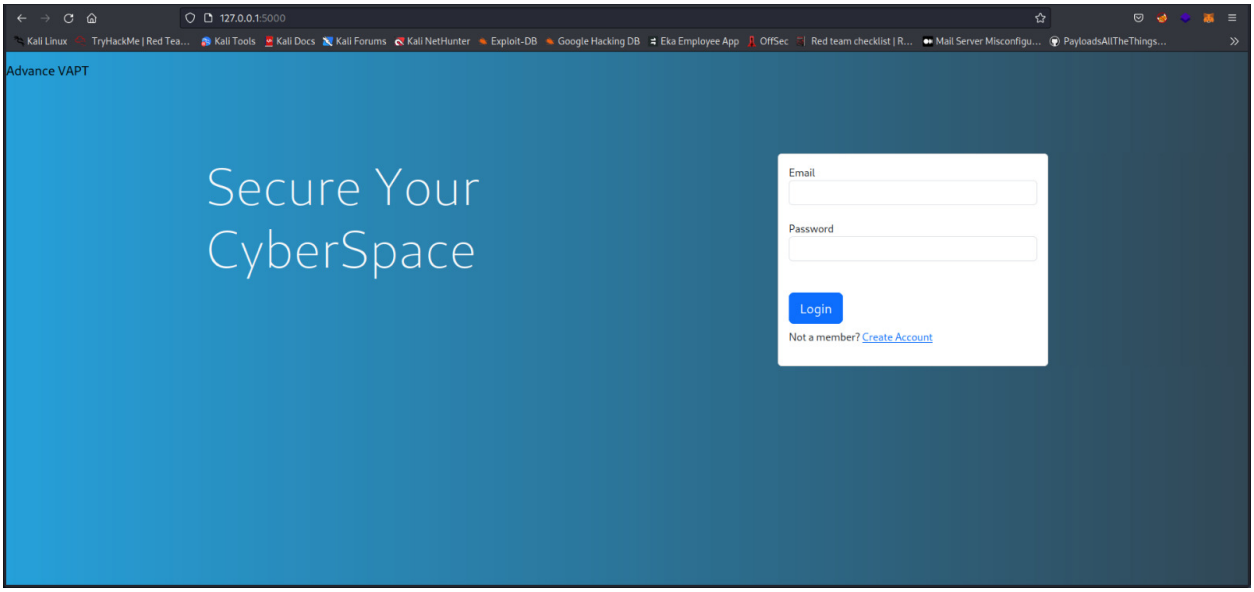


Fig. 5.1. Login Page

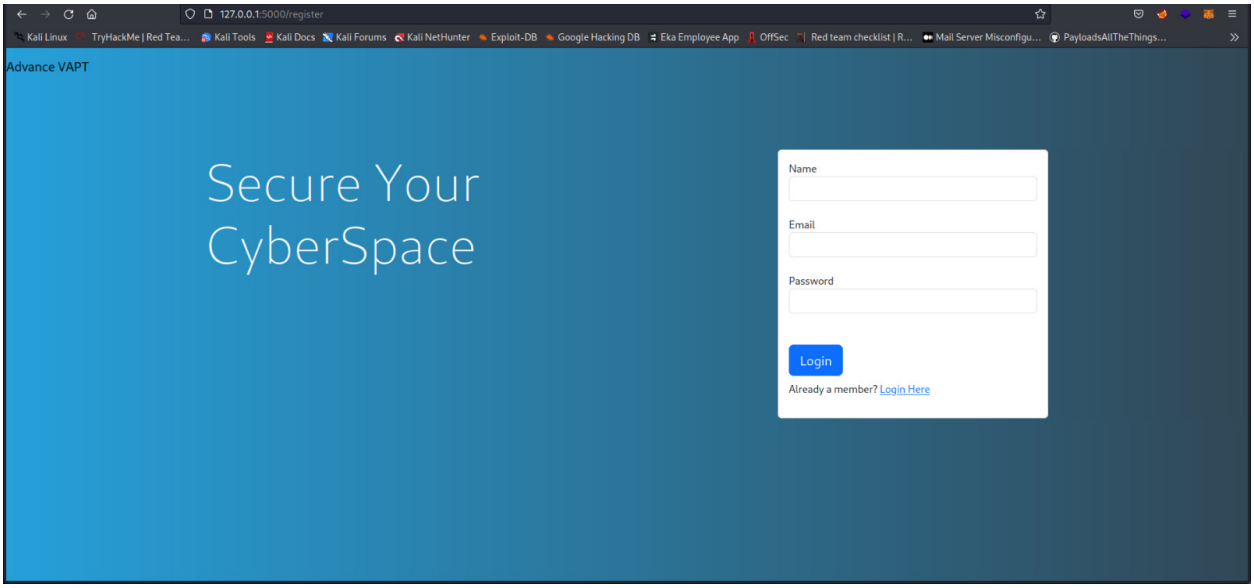


Fig. 5.2. Registration Page

5.2 Assumptions and Dependencies

- Regular Updates: It is assumed that the tool would be regularly updated as per the latest security benchmarks
- The admin has sufficient proficiency to utilize the tool
- Network access: The tool assumes that it has sufficient network access to scan and test the web application. This includes the ability to access the target system, as well as any underlying infrastructure, such as databases and network devices.

5.3 General Constraints

- Time Constraint: The overall testing period is very time consuming with some injections taking over 7 hours
- Resource Constraints: Testing can be resource-intensive, requiring significant computational power and memory. The tool may need to operate within resource constraints, such as limited processing power or memory.
- Regulatory Constraints: The tool may need to operate within regulatory constraints, such as compliance with data privacy laws and regulations

5.4 System Flow Diagram

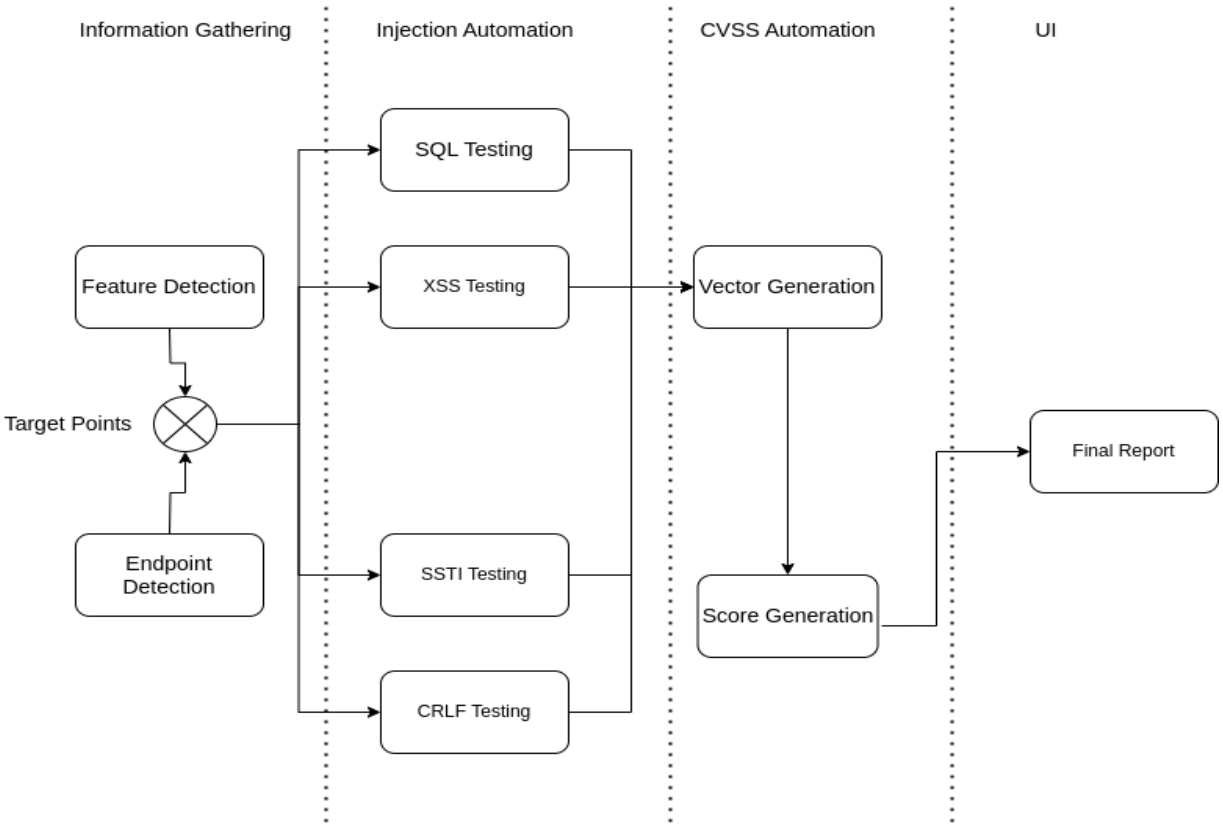


Fig. 5.3. System Flow diagram of the system

The above diagram explains the overall flow of functions and tasks performed in the tool. The tool begins by gathering general information about the target system and also the open input and endpoints. This would be required to implement the automated injection scripts. The scripts are run simultaneously and look for the respective vulnerabilities as they are developed. After the running time of the scripts is completed, the CVSS vector of the found vulnerabilities is generated and the vectors are then imputed through a calculator and the score along with the final report is generated.

5.5 System Architecture

The System architecture illustrates the modules which contributed to the functionality of the tool and how they are interconnected to each other. The modules consist of the scripting module,

feature and endpoint detection modules, the ML module and the UI module. Apart from the modules, the external interaction i.e. The client has also been shown in the above diagram.

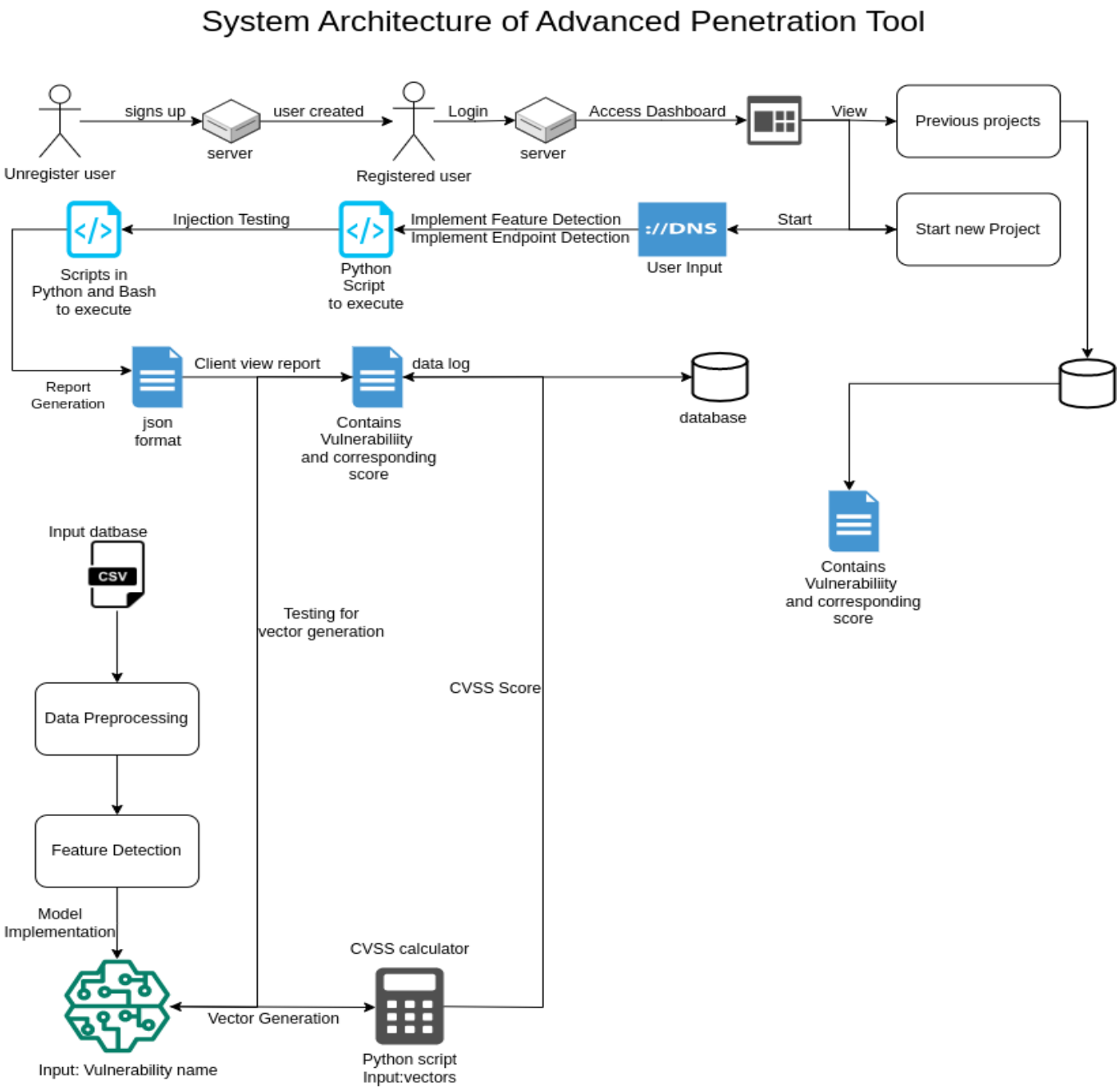


Fig. 5.4. System Architecture Diagram

5.6 Modules of the Project

- **Feature and Endpoint Detection Module:** This module is responsible for the detection of input points and endpoints. Without this module it would be impossible to run the scripting module.
- **Scripting Module:** It pertains to the automated injection scripts that are utilized. The basic architecture is of a vulnerability scanner and scripting interface.
- **ML Module:** Here ML algorithms are trained on a CVSS repository in order to determine appropriate vectors to a corresponding vulnerability.
- **UI Module:** The function of the UI module to provide a simple graphical representation of the projects being run and also give a status check of the same. The final reports along with the mitigation and score will be stored here.

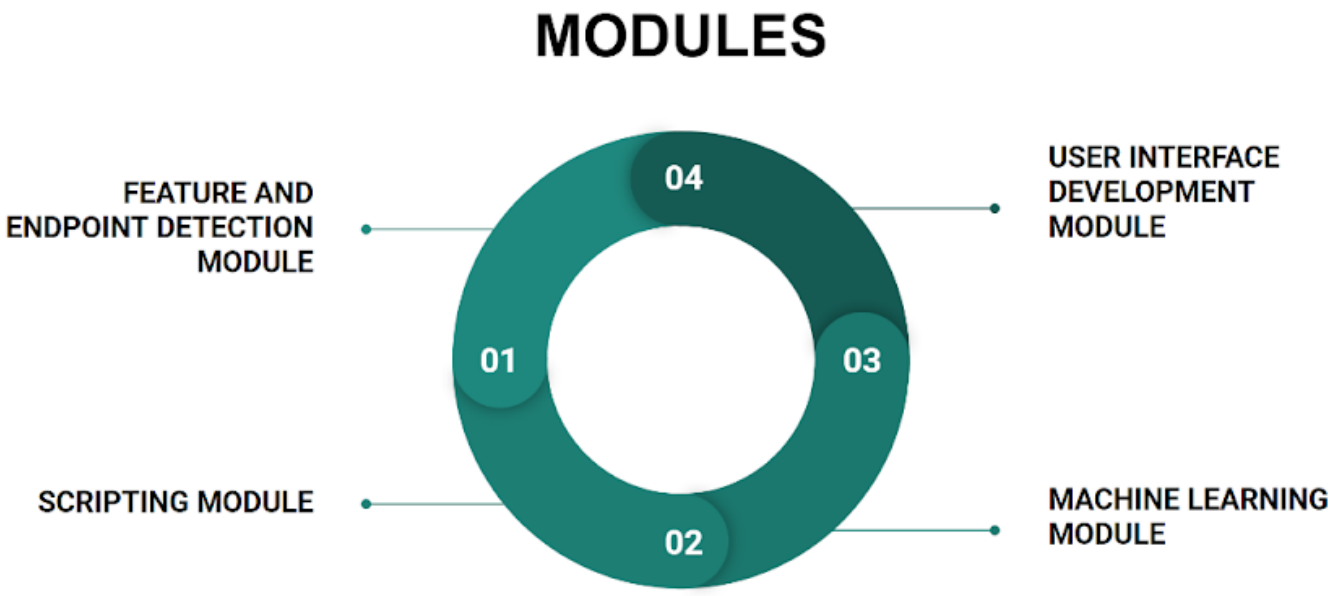


Fig. 5.5. Modules of the Project

5.7 Low Level Design

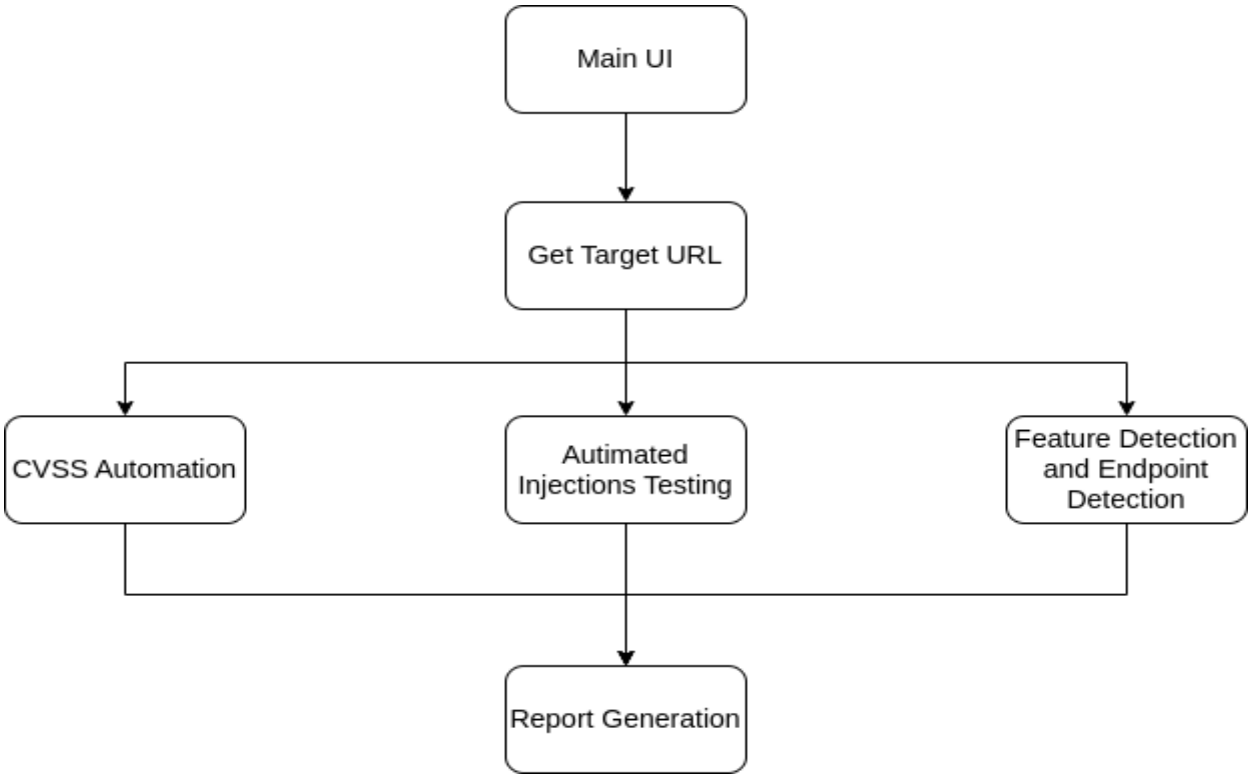


Fig. 5.6. Low Level Design of the System

This low-level design diagram depicts the various components of the advanced web penetration testing tool and their interactions with each other. The main UI serves as a platform for viewing current projects and their reports. After getting the target URL, The following tasks are performed in a chronological order: Feature Detection, Endpoint Detection, Injection vulnerability testing and CVSS automation. After the CVSS automation is done, the end report is generated. The report would consist of all the found vulnerabilities with their mitigation suggestions and a CVSS score that would determine the severity of the vulnerability.

5.8 UML Diagrams

1) Use Case Diagram

Fig. 5.7. Use Case Diagram of the System

The use case diagram focuses on the interaction between the actors and the system. It also identifies all the actions each actor can do based on the privilege and the response from the backend.

2) Activity Diagram

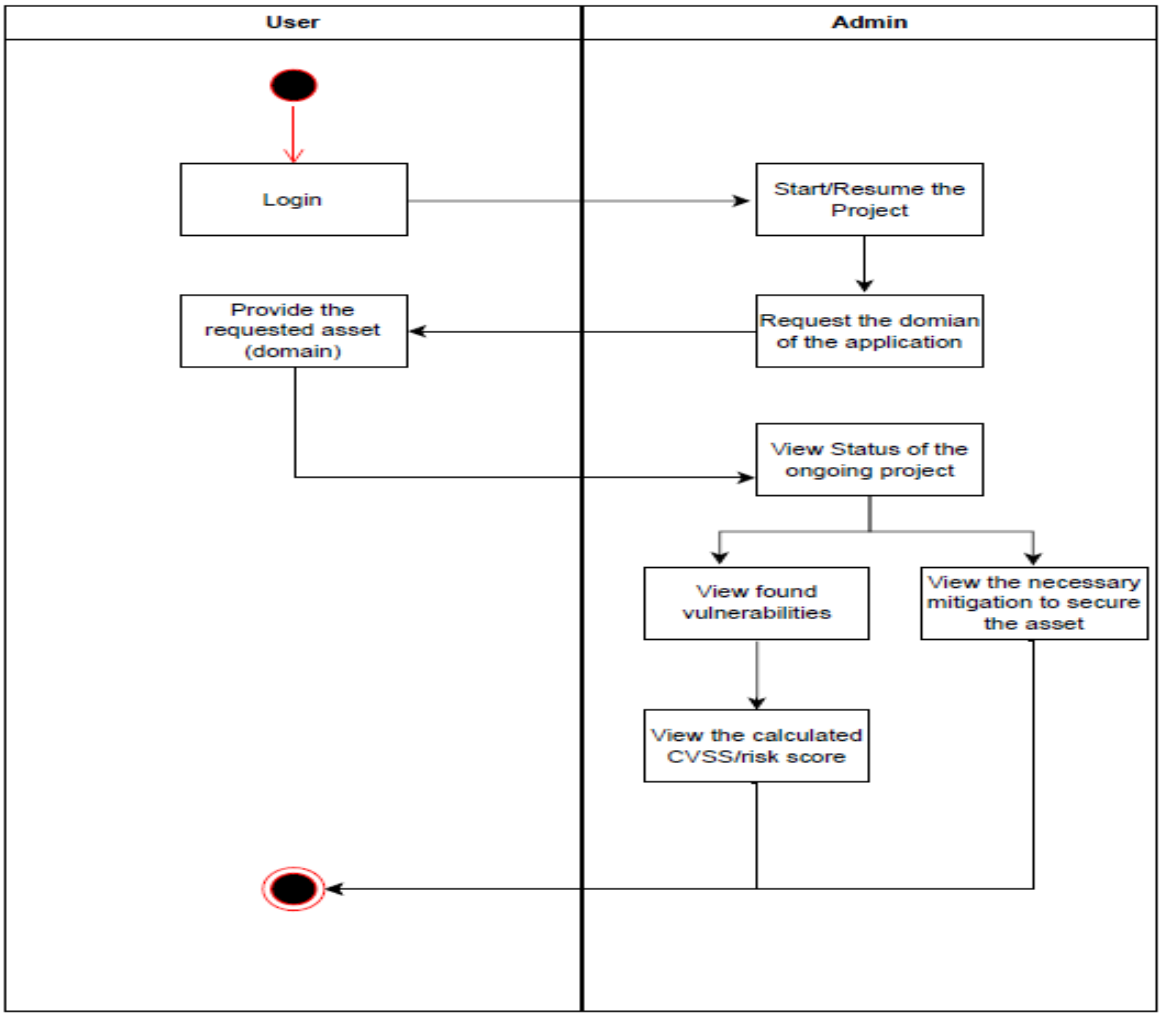


Fig. 5.8. Activity Diagram of the System

The activity diagram is a graphical representation of workflows of stepwise activities and actions associated with the tool. The Activity diagram is designed from the perspective of the client/user. Where the client can choose to start a new project or view an existing project and check its status where they can see the vulnerability found, score associated with it and the suggested mitigations.

3) Class Diagram

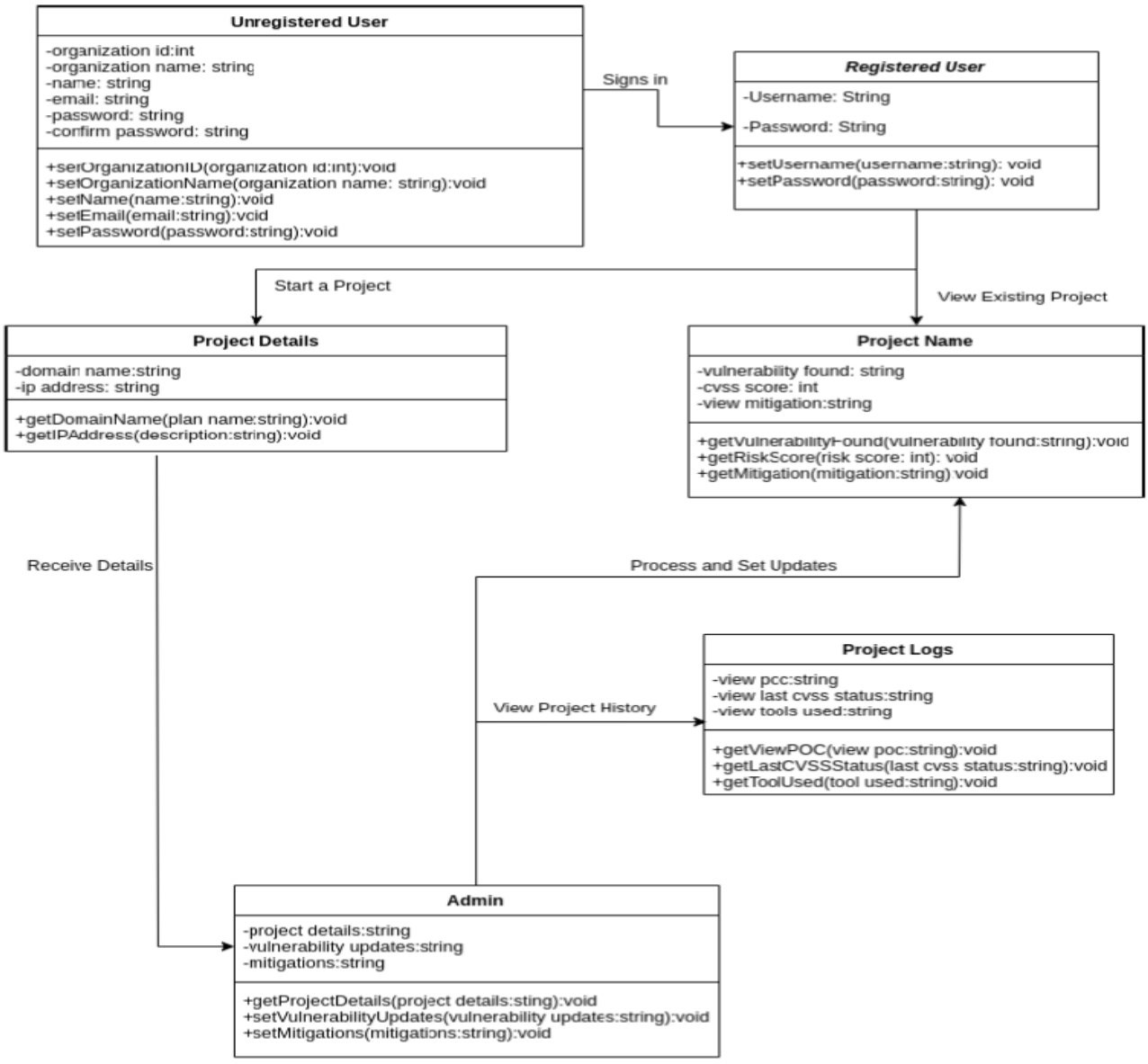


Fig 5.9. Class Diagram of the System

The class diagram depicts all the project components, their attributes and the overall relationship with each other. An unregistered user will need to be verified and sign up with the portal in order to log in. As a registered user, it can view existing projects or decide to start a new project, while starting a new project they are required to provide the domain of the asset. The duration of the project is done and the logs are saved for future reference.

Chapter 6

Project Plan

The above diagram shows the overall timeline of the project. In January, the focus was on researching the current vulnerability detection tool in the domain. Also the time was spent planning on how to proceed with the project and develop the required modules.The feature and endpoint detection tool was developed in February, It would cover all the available input points of the target where we would be able to check whether an injection vulnerability exists or not. Without this module, injection automation would not be possible.

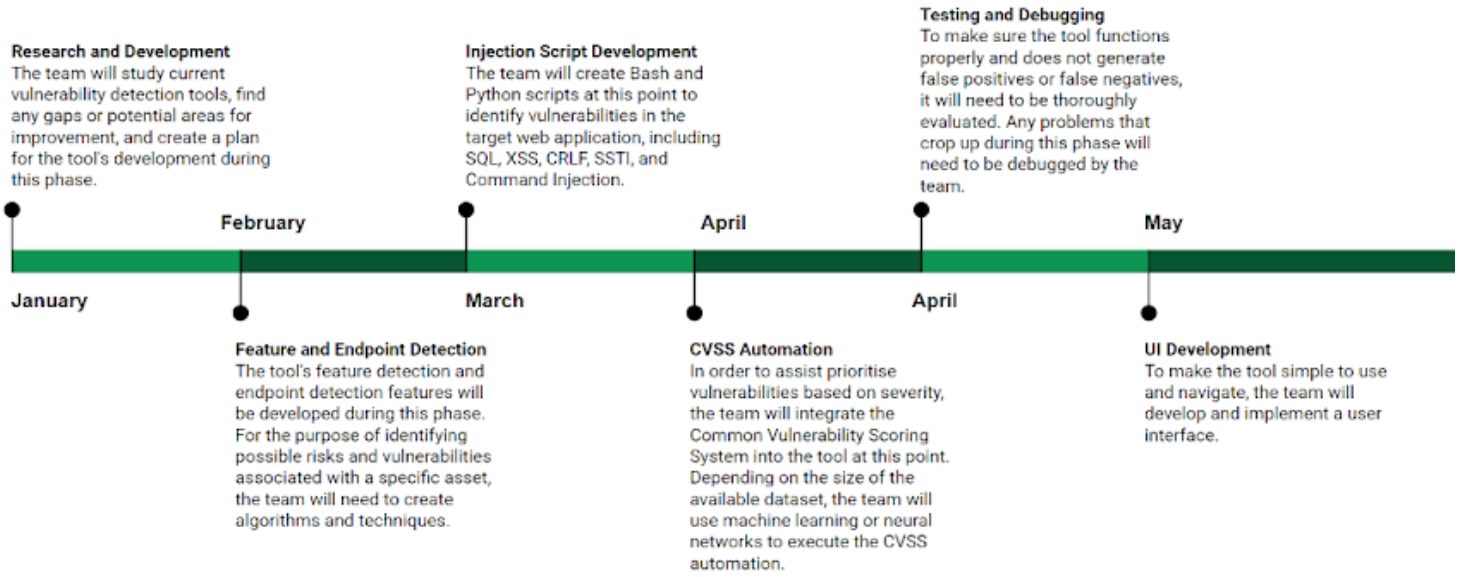


Fig. 6.1. Timeline Diagram of the SDLC

The Injection script development was worked upon in the following month. This would cover preparing scripts to cover various injections such as XSS, SQl, SSTI, CRLF and command injection. In order to assist prioritize vulnerabilities based on severity, CVSS automation was integrated into the tool at this point. Depending on the size of the available dataset, the team will use machine learning or neural networks to execute the CVSS automation. Since the project is about to be concluded, To make sure the tool functions properly and does not generate false positives or false negatives, it will need to be thoroughly evaluated. Any problems that crop up during this phase will need to be debugged by the team.

Chapter 7
Implementation

7.1 Methodology

Bash and python scripts for different vulnerabilities such as SQL, XSS, CRLF, SSTI and Command Injection would be developed to find existing vulnerabilities as specified above in the target web application(ip would be required).

In order for the injection scripts to function properly and prevent bad outputs, feature detection would be required to be done as a prerequisite. It is done in order to find the input via which it can be checked for injection attacks. The same would be added to the injection scripts so it can check for vulnerabilities in all the preexisting input points.

An independent CVSS (Common Vulnerability Scoring System) calculator would be developed to output an appropriate score to the corresponding vulnerability. This would be built on Python. In order to determine the accurate vulnerability score, different parameters would be needed to be considered and give corresponding weightages. Those parameters would include Attack Vector(AV), Attack Complexity(AC), Privilege Requirement(PR), User Interaction(UI), Scope and the impacted CIA triad.

The AV refers to the medium through which the vulnerability can be exploited and the base score is inversely proportional to the proximity of the vulnerable component.
AC refers to the difficulty/complexity of the procedure of the attack and the base score is inversely proportional to the complexity. PR refers to the privilege that an actor must possess to exploit the vulnerability. The base score is inversely proportional to the privilege.

The user interaction means if another user other than the attacker is required to participate in order for the exploit to be successful. The base score is higher if no user interaction is required and vice versa. The scope means that other components of the web component are affected apart from the intended one. The base score; along with the CIA triad. will increase if scope is changed. The CIA triad refers to the Confidentiality, Integrity and Availability of a web app.

The base score is directly proportional to the triad. And it is mandatory for at least one parameter of the triad to be impacted for an application to be 'exploited'. The CVSS Automation would be done with the assistance of machine learning/neural networks depending on the size of dataset at hand. The dataset would consist of a list of injection vulnerabilities and the impacted CVSS parameters and ml would be applied to predict impacted vectors when a vulnerability is inserted.

7.2 Algorithms

FEATURE DETECTION (get_forms_info)

1. create a function called get_forms_info that takes in two parameters, url and res
2. create a new instance of a web driver
3. navigate to the url
4. find all form elements on the page
5. create an empty list called forms
6. iterate over each form element
7. extract the form ID, action URL, and HTTP method
8. create a dictionary called form_info with the extracted form information
9. find all input elements within the current form element
10. create an empty list called input_elements
11. iterate over each input element and extract its name and type
12. create a dictionary called input_info with the extracted input information
13. add the input information dictionary to the input_elements list
14. add the input_elements list to the form_info dictionary
15. add the form_info dictionary to the forms list
16. close the webdriver
17. add the forms list to the res dictionary
18. return the res dictionary

ENDPOINT DETECTION

1. Import the necessary module: from urllib.parse import urlparse
2. Define the function detect_endpoints that takes a URL as input.
3. Parse the URL using urlparse to extract its components.
4. Extract the path and query components from the parsed URL.
5. Split the path component into its segments.
6. Remove any empty segments from the list.
7. If the URL has a query component, append it to the last path segment.
8. Create an empty list to store the endpoints.
9. Iterate over the segments and build the endpoints by joining the path segments progressively.
10. Return the list of endpoints.

7.3 Other Implementation Details

INJECTION SCRIPTING

- **SQL -** SQL injection is a web security vulnerability that occurs when an attacker can manipulate user-supplied input to execute unauthorized SQL queries. By injecting malicious SQL code, attackers can manipulate the database, extract sensitive information, modify or delete data, or even gain unauthorized access to the underlying system.
- **XSS** - Cross-Site Scripting is a vulnerability that allows attackers to inject malicious scripts into web pages viewed by other users. It occurs when a web application fails to properly validate or sanitize user input, allowing the injection of harmful scripts. When other users visit the affected page, the malicious scripts execute in their browsers, enabling the attacker to steal sensitive information, perform actions on behalf of the user, or deface the website.
- **CRLF** - CRLF injection is an attack technique that exploits improper handling of carriage return and line feed characters by web applications. Attackers inject these characters into user-supplied input to manipulate the formatting or add malicious content

in HTTP headers or other response outputs. CRLF attacks can lead to various security issues, such as HTTP response splitting, session hijacking, or content injection.

- **SSTI** - Server-Side Template Injection is a vulnerability that occurs when an application allows user-supplied input to be embedded directly into server-side templates. If not properly validated or sanitized, this can enable an attacker to inject template syntax or code that gets executed on the server. This can lead to remote code execution, data leakage, or other types of attacks depending on the capabilities of the underlying template engine.
- **Command Injection** - Command injection is a vulnerability that arises when an application allows user-controlled input to be directly included in system command execution calls without proper validation or sanitization. Attackers exploit this vulnerability to execute arbitrary commands on the targeted system, which can result in unauthorized access, data loss, or further compromise of the system.

RANDOM FOREST REGRESSION

- Random Forest is a prominent machine learning method that belongs to the supervised learning technique.Adopted for classification as well as regression issues in ML, it is established on the notion of ensemble learning which is the procedure of merging numerous classifiers to resolve a complicated difficulty and to upgrade the efficiency of the method.
- As the name implies, Random Forest is a classifier that holds a number of decision trees on several subgroups of the provided dataset and then considers the average to enhance the prediction accuracy of the dataset.
- Rather than depending on one decision tree this algorithm collects the output from each tree and predicts the concluding result.
- Higher the number of trees , greater is the accuracy of the algorithm.

SUPPORT VECTOR REGRESSION

- SVR is a supervised learning technique used for regression tasks to predict continuous numerical values.
- It applies the principles of Support Vector Machines (SVM) and aims to find a hyperplane that maximizes the margin while allowing for a specified tolerance level (epsilon) around the predicted values.
- SVR maps the input data into a high-dimensional feature space using a kernel function, enabling the capture of nonlinear relationships.
- The training process involves solving a quadratic optimization problem to minimize the error between predicted and actual values, making SVR suitable for handling both linear and nonlinear regression problems.

7.4 Discuss Dataset

The dataset is provided by the company and is only used in CVSS automation module. The provided dataset is of .csv format and has only two columns or features i.e CVSS and total. Here under the CVSS column all the CVSS scores extracted by the injection scripts are present. The total column acts as a target variable and depicts how severely the application is being affected.

	CVSS	total
0	7.6,7.9,4.6,5.4,6.9,6.9,4.2,3.5,2.7,3.9	7.8
1	9.1,8.2,8.3,8.4,8.7,8.9,6.5,6.9	9.3
2	8.2,8.9,7.2,7.11,6,6.91,3.9	7.9
3	5.5,7.8,6.5,4.3,2.5,2.8,3.9	7.9
4	8.3,5.6,5.3,6.6,3.7	8.4

Fig. 7.1. Image of Pandas Dataframe consisting of CVSS and their total

With the help of feature engineering and CVSS vulnerability metric we can add more features to the dataset which can further add up to the accuracy of our dataset.

	total_vulnerabilities	critical_no	high_no	moderate_no	low_no	total
total_vulnerabilities	1.000000	0.268958	0.457164	0.621959	0.616740	0.279187
critical_no	0.268958	1.000000	0.277238	-0.069415	-0.087652	0.553347
high_no	0.457164	0.277238	1.000000	0.034359	-0.090923	0.477513
moderate_no	0.621959	-0.069415	0.034359	1.000000	0.128178	0.099063
low_no	0.616740	-0.087652	-0.090923	0.128178	1.000000	-0.225915
total	0.279187	0.553347	0.477513	0.099063	-0.225915	1.000000

Fig 7.2 Correlation between the engineered features and the total CVSS

To add more feature diversity we find the maximum count of CVSS values in the provided dataset and pad every CVSS score whose count is less than the maximum value with ‘0’.

	CVSS_a	CVSSS_b	CVSS_c	CVSS_d	CVSS_e	CVSS_f	CVSS_g	CVSS_h	CVSS_i	CVSS_j	CVSS_k	CVSS_l	CVSS_m	CVSS_n	CVSS_o
0	7.6	7.9	4.6	5.40	6.9	6.90	4.2	3.5	2.7	3.9	0.0	0.0	0.0	0.0	0.0
1	9.1	8.2	8.3	8.40	8.7	8.90	6.5	6.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	8.2	8.9	7.2	7.11	6.0	6.91	3.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	5.5	7.8	6.5	4.30	2.5	2.80	3.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	8.3	5.6	5.3	6.60	3.7	0.00	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Fig 7.3. Padded CVSS values added as features

Chapter 8

Performance Evaluation and Testing

8.1 Performance Evaluation

Table 8.1. Time Complexity for the respective algorithms

Algorithm	Best Case	Average Case	Worst Case
Feature Detection	O(1)	O(m * k)	O(m * k_max)
Endpoint Detection	O(1)	O(n^2)	O(n^2)
Random Forest Regression	O(n * m * log(m))	O(n_estimators * n * m * log(m))	O(n_estimators * n * m * log(m))
Support Vector Regression	O(n^2 * m)	_	O(n^3 *m)

- **Feature Detection -** In the best-case scenario, when there are no forms on the webpage, the function has a constant time complexity of O(1). In the average case, it depends on the number of forms (m) and the average number of input elements per form (k), resulting in a time complexity of approximately O(m * k). In the worst case, with a large number of forms and input elements, the time complexity is approximately O(m * k_max), where k_max represents the maximum number of input elements per form.
- **Endpoint Detection -** In the best-case scenario, when the URL has no path or query components, the function has a constant time complexity of O(1). In the average case, where the length of the URL path is denoted as n, the function has an average time complexity of O(n^2) due to the iteration over each segment and the concatenation operation. In the worst case, with a long path and a large number of segments, the function's time complexity is approximately O(n^2) since it iterates over all segments and performs string concatenation in each iteration.

- **Random Forest Regression -** In the best case, the time complexity is determined by the time complexity of training a single decision tree, which is typically O(n * m * log(m)), where n is the number of training instances and m is the number of features. In the average case, the time complexity is approximately O(n_estimators * n * m * log(m)) for both training each decision tree and making predictions, where n_estimators is the number of trees in the forest. In the worst case, the time complexity remains the same as the average case.
- **Support Vector Regression -** Best-case, average-case, and worst-case time complexity for SVR typically range from O(n^2 * m) to O(n^3 * m), depending on the specific implementation and kernel function chosen. The time complexity primarily depends on the number of training instances (n) and the number of features (m). The exact time complexity can vary based on factors such as the kernel type (linear, polynomial, radial basis function, etc.) and the optimization algorithm used. It's worth noting that SVR can be computationally expensive for large datasets, particularly when the number of training instances (n) is high. Techniques such as kernel approximation or stochastic gradient descent are commonly used to mitigate the computational burden and reduce the time complexity.

8.2 Type of Testing

Table 8.2. Injections scripts and respective testing techniques

Injection Type	Testing Techniques
SQL Injection	Input validation and sanitization - Testing for escape characters and quotes - UNION-based and boolean-based testing - Error-based testing - Time-based testing - Blind SQL injection testing - Out-of-band testing
XSS (Cross-Site Scripting)	Input validation and sanitization - Testing for script tags, event handlers, and HTML encoding - Testing different contexts (e.g., HTML, attributes, JavaScript) - Payload

	variations (e.g., <script>alert('XSS')</script>, <img src=x onerror=alert('XSS')>) CRLF Injection - Input validation and sanitization - Testing for newline characters (%0D and %0A) - HTTP response splitting - HTTP header injection - User-controlled HTTP responses
SSTI (Server-Side Template Injection)	Input validation and sanitization - Testing for template syntax and placeholders - Injecting template directives and expressions - Testing different template engines (e.g., Jinja, Twig)
CRLF Injection	Input validation and sanitization - Testing for newline characters (%0D and %0A) - HTTP response splitting - HTTP header injection - User-controlled HTTP responses
Command Injection	Input validation and sanitization - Testing for command terminators (;, &,

- **SQL Injection** - Input validation and sanitization involve ensuring that user input is properly validated and sanitized before being used in SQL queries. Testing for escape characters and quotes helps identify vulnerabilities where an attacker can bypass input sanitization and inject malicious SQL code. UNION-based and boolean-based testing involve crafting specific queries to exploit vulnerabilities and retrieve unauthorized data. Error-based testing focuses on generating SQL errors to gather information about the database. Time-based testing involves injecting delays to infer information about the database. Blind SQL injection testing aims to extract information by using conditional queries. Out-of-band testing involves leveraging alternate communication channels to extract data.

- **XSS (Cross Scripting)** - Input validation and sanitization are crucial to prevent XSS attacks. Testing for script tags, event handlers, and HTML encoding helps identify vulnerabilities where user input can be rendered as code in a web page. Testing different contexts, such as HTML, attributes, and JavaScript, helps cover various injection points. Payload variations, including different forms of XSS payloads, are tested to ensure comprehensive coverage against XSS vulnerabilities.
- **SSTI (Server-Side Template Injection)** - Input validation and sanitization play a significant role in preventing SSTI attacks. Testing for template syntax and placeholders helps identify potential injection points. Injecting template directives and expressions allows for testing the behavior of the template engine and detecting vulnerabilities. Testing different template engines is essential as each may have its own syntax and behavior regarding injection attacks.
- **CRLF Injection** - Input validation and sanitization are important to prevent CRLF injection attacks. Testing for newline characters and HTTP response splitting helps identify vulnerabilities where an attacker can manipulate the HTTP response or inject additional headers. HTTP header injection testing focuses on injecting malicious headers to manipulate the server's behavior. User-controlled HTTP responses aim to manipulate the server's response to perform unauthorized actions.
- **Command Injection** - Input validation and sanitization are vital in preventing command injection attacks. Testing for command terminators like semicolons, ampersands, and pipes helps identify vulnerabilities where user input can be used to execute unauthorized commands on the system. Proper input validation and the use of secure APIs can help mitigate command injection vulnerabilities.

Chapter 9

Result and Analysis

The Client gives the URL/Domain which to be tested. That domain is entered here and then the actual testing starts. This domain will be tested for all the possible vulnerabilities and the result of the found vulnerabilities will be shown in a tabular format which can be easily understood by the client.

Once the client submits the request of a domain to be tested it would go to the project manager. The project manager will initiate the testing from the admin side.

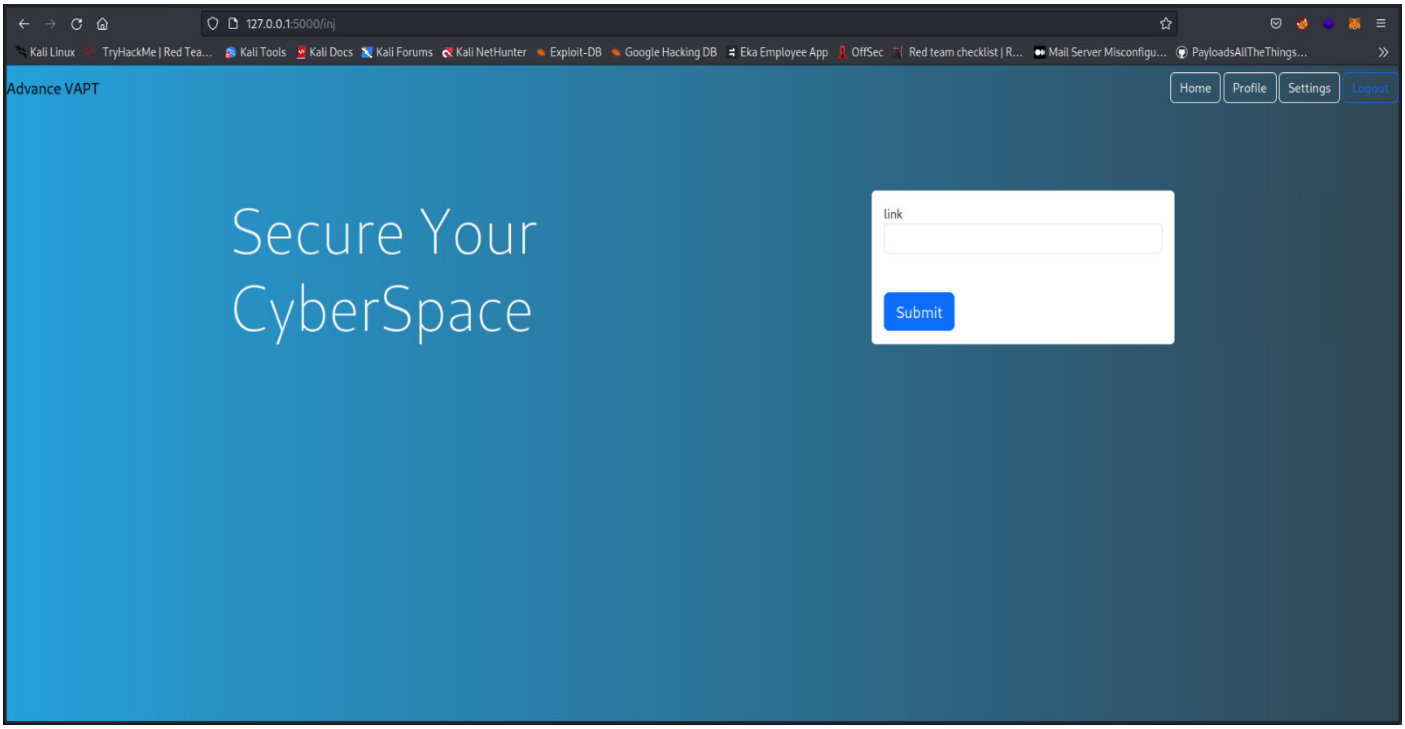


Fig. 9.1. Client side page

Next the request will go to the project manager and the project manager will verify the request and the documentation required before starting the testing process.

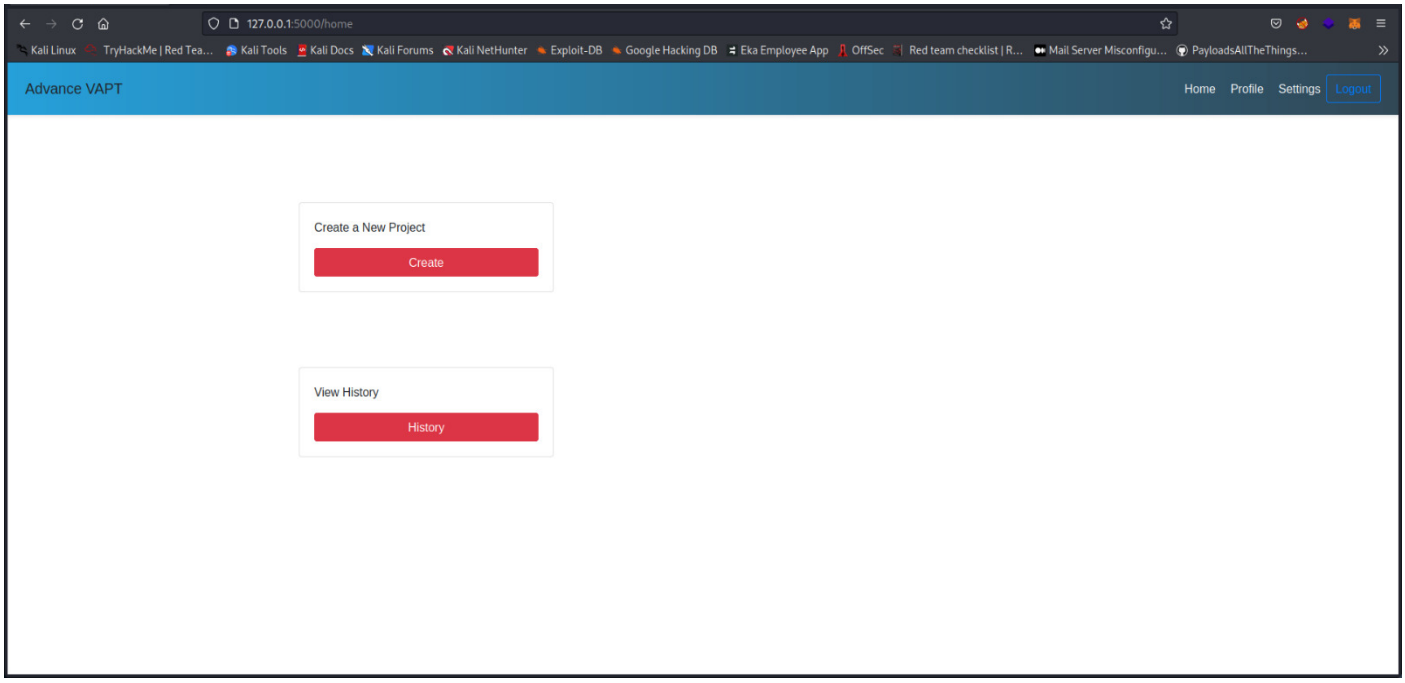


Fig. 9.2. Admin page

After getting notified that the client wants to start the testing process, the admin creates a new project and initiates the testing process. Also, the admin can view the history of the previous testing results as well as the current testing results with the exact date and time at which the testing occurred.

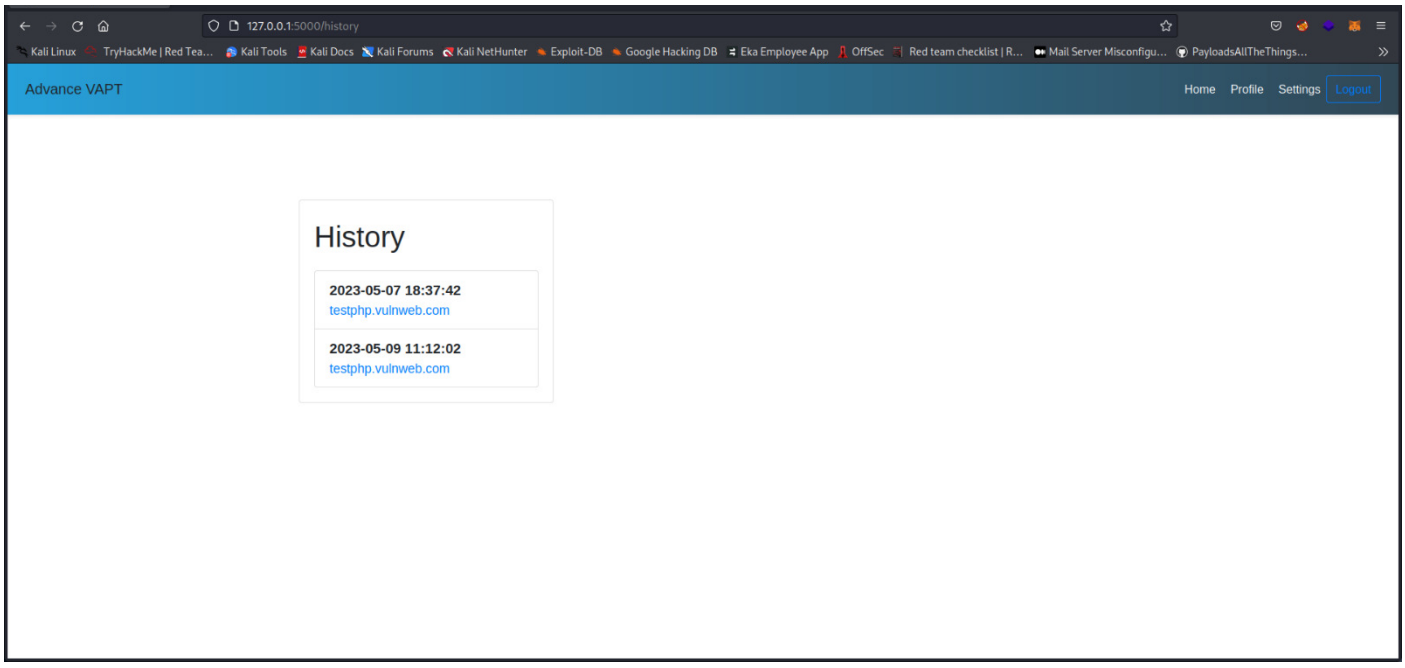


Fig. 9.3. History page containing the test results

Once the testing process is initiated a loading screen will appear as the testing might take several hours to complete depending on the number of features, endpoints and sub - domains present. The client will be notified once the test results are available.

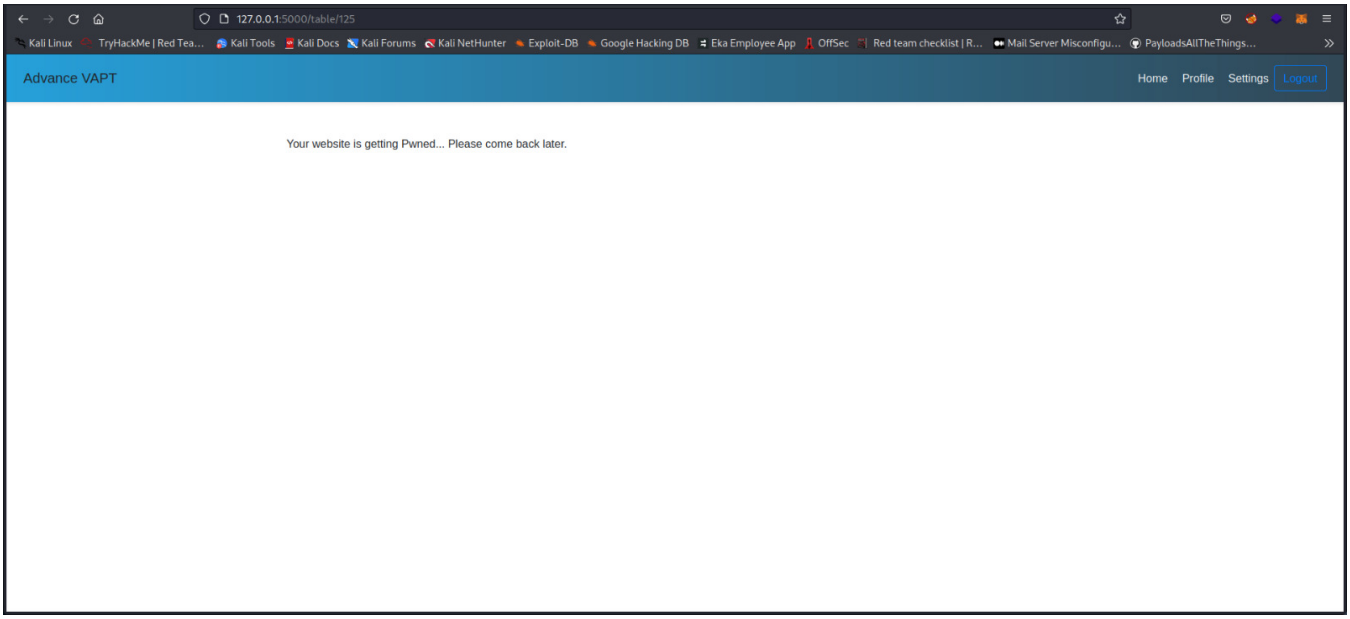
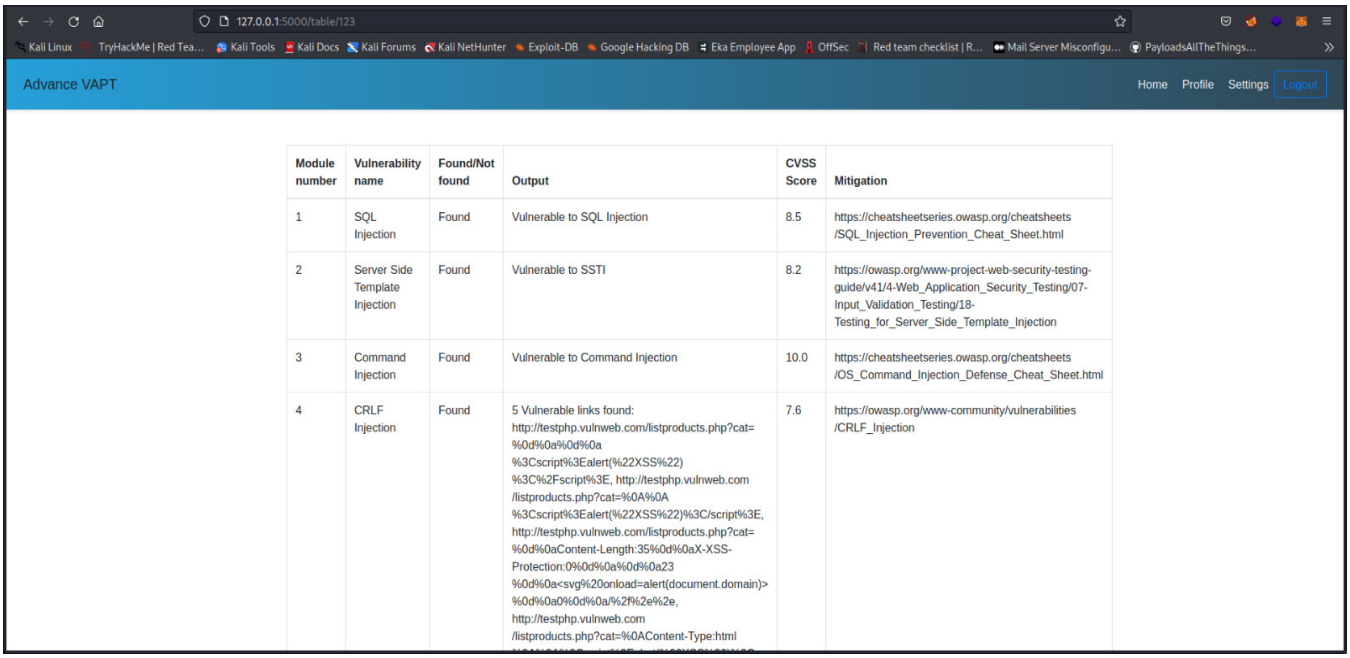


Fig. 9.4. Testing in process screen



Module number	Vulnerability name	Found/Not found	Output	CVSS Score	Mitigation
1	SQL Injection	Found	Vulnerable to SQL Injection	8.5	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
2	Server Side Template Injection	Found	Vulnerable to SSTI	8.2	https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/07-Input_Validation_Testing/18-Testing_for_Server_Side_Template_Injection
3	Command Injection	Found	Vulnerable to Command Injection	10.0	https://cheatsheetseries.owasp.org/cheatsheets/OS_Command_Injection_Defense_Cheat_Sheet.html
4	CRLF Injection	Found	5 Vulnerable links found: http://testphp.vulnweb.com/listproducts.php?cat=%0d%0a%0d%0a%3Cscript%3Ealert(%22XSS%22)%3C%2Fscript%3E, http://testphp.vulnweb.com/listproducts.php?cat=%0A%0A%3Cscript%3Ealert(%22XSS%22)%3C/script%3E, http://testphp.vulnweb.com/listproducts.php?cat=%0d%0aContent-Length:35%0d%0aX-XSS-Protection:0%0d%0a%0d%0a23%0d%0a<svg%20onload=alert(document.domain)>%0d%0a0%0d%0a/%2f%2e%2e, http://testphp.vulnweb.com/listproducts.php?cat=%0AContent-Type:html	7.6	https://owasp.org/www-community/vulnerabilities/CRLF_Injection

Fig. 9.5.1. Report containing found vulnerabilities

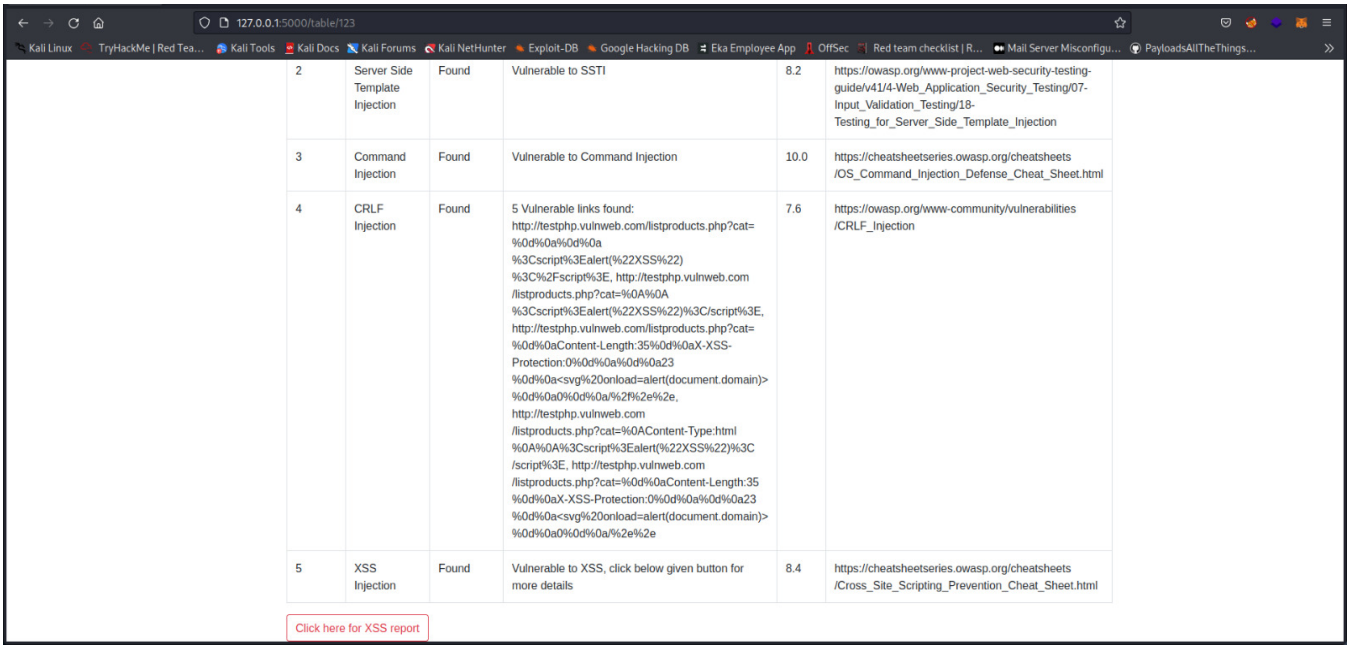


Fig. 9.5.2. Report containing found vulnerabilities

A detailed view of the results would be available to the client once the testing has been completed. The report would contain the Module number, Vulnerability name, Vulnerability found or not found, Output, CVSS score and the best mitigation suggestions to resolve the vulnerabilities. Also, the client can view the detailed report of the individual vulnerabilities found with the exact URL where it is occurring and can also view the Endpoints and Features found through endpoint detection and feature detection modules present.

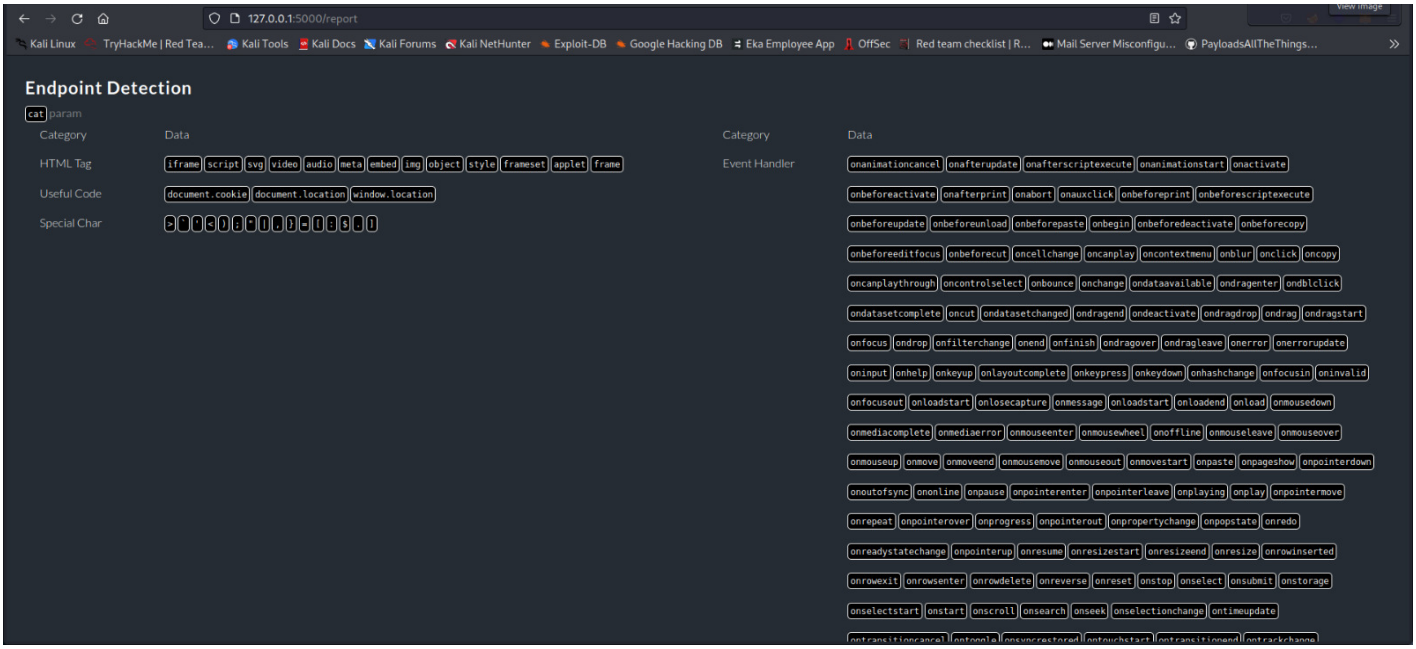


Fig. 9.6.1 Results page - Endpoint Detection

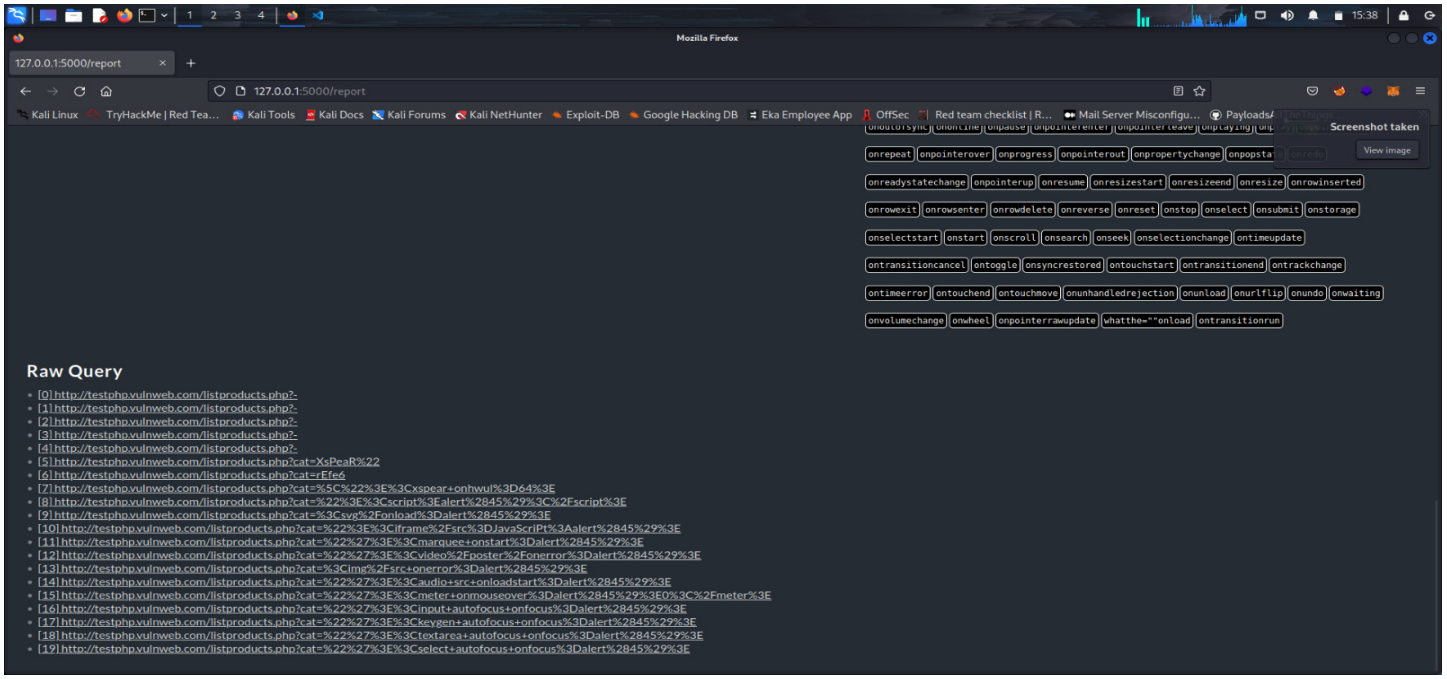


Fig. 9.6.2 Results page - Endpoint Detection

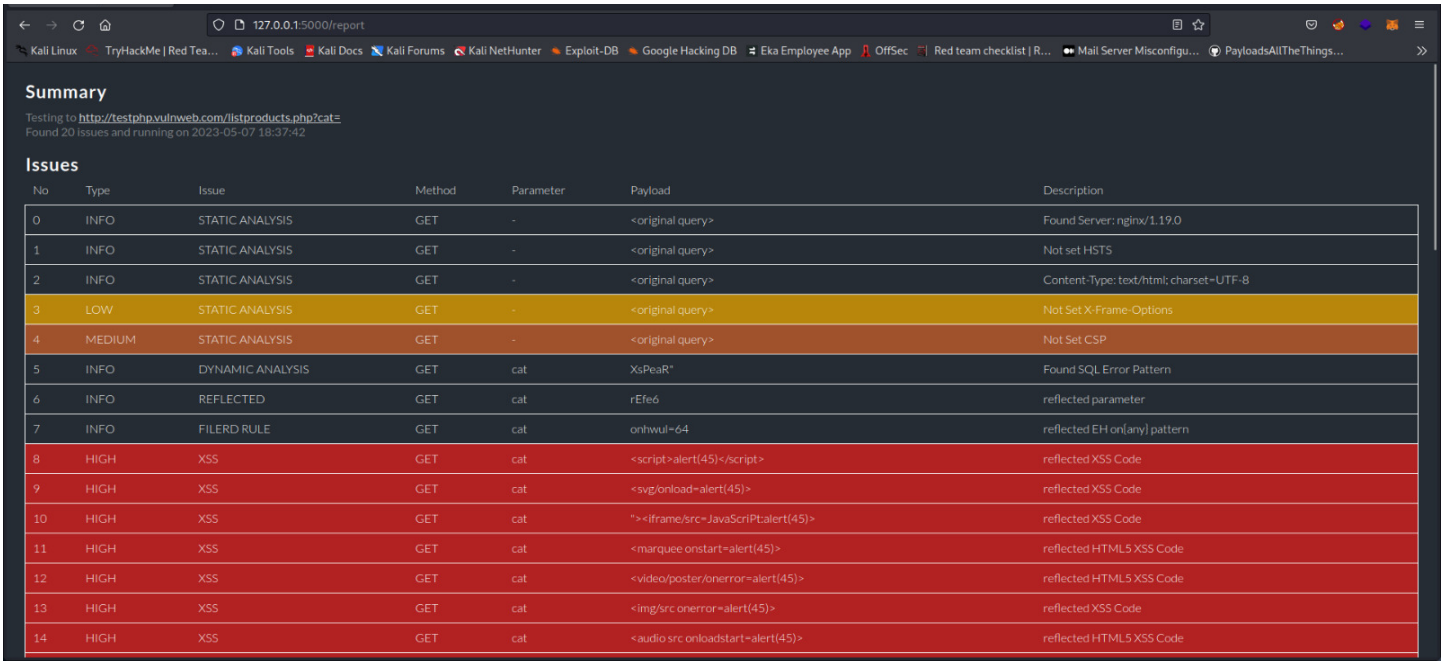


Fig. 9.6.3 Results page - XSS report

Here the results containing all the issues in detail would be visible to the client so that the client also understands the various methods and parameters present in the report. Next the client can proceed to patch the found vulnerabilities with the help of mitigation suggestions present in the detailed report as shown in Fig. 9.5.1 and 9.5.2.

To calculate the total impact of the vulnerabilities on our website the CVSS scores for each vulnerability are grouped and treated as a vulnerability vector and a score between 0.0 and 10.0 is given. This score will follow the CVSS range metric i.e score in range of 0.1 to 3.9 will represent low, 4.0 to 6.9 will represent moderate, 7.0 to 8.9 will be considered high and the CVSS score in the range of 9.0 to 10.0 will be taken as a critical vulnerability.

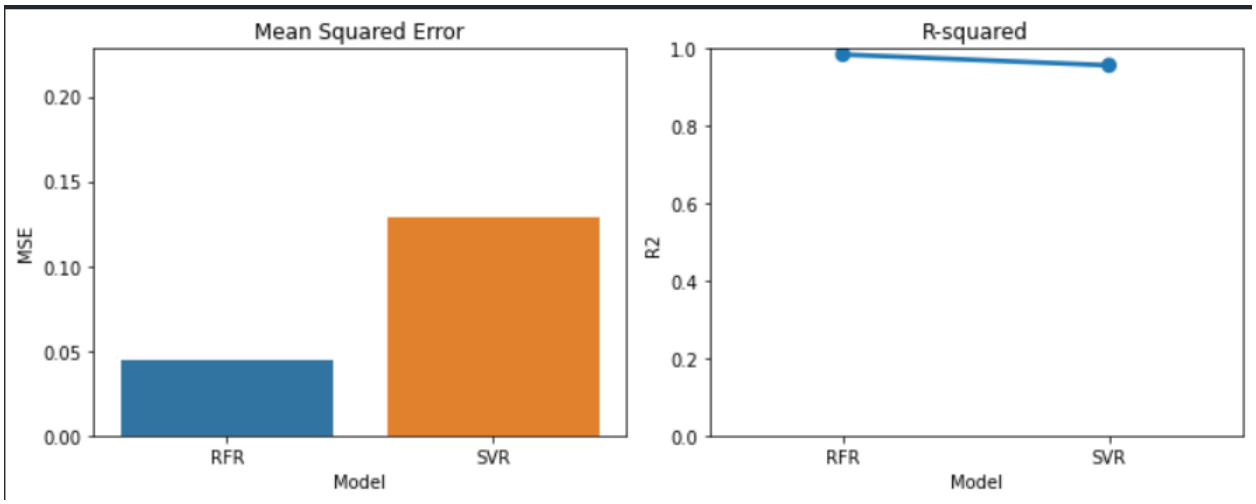


Fig 9.7. Comparison of SVM and RFR model

The models on which the data is trained are Support Vector Regression and Random Forest Regression. These models are then evaluated using two metrics - Mean Squared Error and R-Squared. From Fig 9.7 we can conclude that RFR is a better option since the mse is less than SVR and R-Squared value is closer to 1.

Now for the report given in Fig 9.5.1 and 9.5.2, for every vulnerability a respective CVSS is assigned so to get the total CVSS score for the whole application involved we extract this CVSS score as a vector and feed it to our trained model.

The CVSS vector for the given report will be [8.5,8.2,10.0,7.6,8.4]. This vector will be engineered into a suitable array (Fig 7.2 and 7.3) and will be fed to the trained model (Random Forest Regression).

Enter the vulnerability vector (values separated by commas):8.5,8.2,10.0,7.6,8.4

Predicted vulnerability level: 8.6

Chapter 10
Applications

1. **Intrusion Detection:** Feature Detection and Endpoint Detection can be used to detect intrusions into a system or network. Feature detection algorithms can identify unusual patterns in network traffic, while endpoint detection can identify suspicious activity on individual devices.
2. **Malware Detection:** Both Feature Detection and Endpoint Detection can be used to detect malware on a system or network. Feature Detection algorithms can identify patterns associated with known malware signatures, while Endpoint Detection can detect malicious activity such as the creation or deletion of files, changes to system settings, or network connections.
3. **Threat Hunting:** Feature Detection and Endpoint Detection can be used to proactively search for signs of malicious activity. Threat hunters can use these techniques to search for unusual patterns or behavior that may indicate the presence of a threat.
4. **Incident Response:** Feature Detection and Endpoint Detection are crucial in incident response, as they can help security teams quickly identify the source of a security incident and respond appropriately. By detecting suspicious activity in real-time, security teams can prevent the incident from escalating.
5. **Compliance Monitoring:** Feature Detection and Endpoint Detection can help organizations ensure compliance with security regulations and standards. By monitoring for unusual activity, organizations can identify potential compliance violations and take action to address them.
6. **Network Security:** VAPT is widely used to assess the security of a computer network, including routers, switches, firewalls, and servers. The assessment involves identifying vulnerabilities in the network infrastructure and testing them by simulating real-world attack scenarios.

7. **Web Application Security:** VAPT is used to assess the security of web applications, including web servers, web-based applications, and e-commerce websites. The assessment involves identifying vulnerabilities in the web application code and testing them by simulating real-world attack scenarios.
8. **Mobile Application Security:** VAPT is used to assess the security of mobile applications, including Android and iOS apps. The assessment involves identifying vulnerabilities in the mobile application code and testing them by simulating real-world attack scenarios.
9. **Cloud Security:** VAPT is used to assess the security of cloud-based infrastructure and services, including Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). The assessment involves identifying vulnerabilities in the cloud infrastructure and testing them by simulating real-world attack scenarios.
10. **Social Engineering:** VAPT is also used to assess the effectiveness of an organization's security awareness training program by simulating social engineering attacks. This type of assessment involves attempting to trick employees into providing sensitive information or performing actions that can compromise the organization's security.

Chapter 11
Conclusion

VAPT helps organizations identify vulnerabilities and weaknesses in their systems and applications, allowing them to take action to mitigate risks and improve overall security. Also, many industries have regulations that require regular security assessments, including VAPT. By conducting VAPT, organizations can ensure they are meeting compliance requirements and avoid potential penalties for non-compliance.

The early identification of vulnerabilities and flaws in systems and apps can help avert costly security breaches in the future. VAPT can be a low-cost method of identifying and addressing vulnerabilities before they are exploited by attackers. By demonstrating a commitment to security and the protection of sensitive data, VAPT may also assist in creating confidence with customers, partners, and stakeholders.

If an organization is called as "SECURE", a strong security posture can be a competitive advantage for businesses, especially those in industries where security is a top concern for customers and partners.

Overall, VAPT is an essential component of a comprehensive security program and can help organizations proactively identify and address vulnerabilities and weaknesses in their systems and applications.

Future Prospects of the Project

With the advancements in AI and machine learning, we can expect more automation in VAPT and penetration testing. Automated tools can quickly scan systems for vulnerabilities and identify potential threats.

Furthermore, the growing use of IoT devices in homes and organizations poses additional security challenges. VAPT and penetration testing can be used to assess IoT device security and discover potential vulnerabilities.

We can expect to see more automated threat response systems that reduce response times and minimize the impact of security incidents. Integration with DevOps can identify and remediate security issues earlier in the development process, and orchestration and automation platforms can streamline security operations and provide a comprehensive security solution.

There will be a greater focus on cloud security automation, including automated threat detection, configuration management, and compliance monitoring. By embracing automation technologies, organizations can improve their security posture and reduce the risk of security incidents.

The scope of our project can be further expanded by accommodating more injection-based vulnerabilities such as code injection, email header injection, host header injection, LDAP injection (Lightweight Directory Access Protocol), and Xpath injection.

In summary, VAPT and penetration testing will continue to play a critical role in cybersecurity. As technology advances, we can expect new challenges and threats to emerge, making it essential to adapt VAPT and penetration testing to address these new challenges.

References

[1] A. M. Hasan, D. T. Meva, A. K. Roy and J. Doshi, "Perusal of web application security approach," 2017 International Conference on Intelligent Communication and Computational Techniques (ICCT), Jaipur, India, 2017, pp. 90-95.

[2] A. Hasan and D. Meva, “Web application safety by penetration testing,” Web Application Safety by Penetration Testing International Journal of Advanced Studies of Scientific Research, Volume 3, Issue 9, 2018.

[3] Y. Khera, D. Kumar, Sujay and N. Garg, "Analysis and Impact of Vulnerability Assessment and Penetration Testing," 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 525-530.

[4] M. Aydos, Ç. Aldan, E. Coşkun, and A. Soydan, “Security testing of web applications: A systematic mapping of the literature,” Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 9, pp. 6775–6792, 2022.

[5] S. Shukla and T. B. Rehman, “Vapt & Exploits, along with classification of exploits,” International Journal of Computer Science and Engineering, vol. 9, no. 3, pp. 1–4, 2022.

[6] M. Humayun, M. Niazi, N. Z. Jhanjhi, M. Alshayeb, and S. Mahmood, “Cyber security threats and vulnerabilities: A systematic mapping study,” Arabian Journal for Science and Engineering, vol. 45, no. 4, pp. 3171–3189, 2020.

[7] M. Albahar, D. Alansari, and A. Jurcut, “An empirical comparison of pen-testing tools for detecting web app vulnerabilities,” Electronics, vol. 11, no. 19, p. 2991, 2022.

[8] P. R. Vishnu, P. Vinod, and S. Y. Yerima, “A deep learning approach for classifying vulnerability descriptions using self attention based neural network,” Journal of Network and Systems Management, vol. 30, no. 1, 2021.

[9] A. Khazaei, M. Ghasemzadeh, and V. Derhami, “An automatic method for CVSS score prediction using vulnerabilities description,” Journal of Intelligent & Fuzzy Systems, vol. 30, no. 1, pp. 89–96, 2016.

[10] A. Beck and S. Rass, “Using neural networks to aid CVSS risk aggregation - an empirically validated approach,” Journal of Innovation in Digital Ecosystems, vol. 3, no. 2, pp. 148–154, 2016.
[11] Bing Song, Li Sun, Zhihong Qin, "Design of Web Security Penetration Test System Based on Attack and Defense Game", Scientific Programming, vol. 2022, Article ID 8645969, 2022.

Publication Details

ADVANCED WEB PENETRATION TESTING TOOL

Vinayak Musale[1], Yash Vyas[2], Indraneel Tilloo[3], Nirmal Vyas[4], Akshit Langeh[5], Nabil Barbhuyan[6]

[1]School of Computer Engineering & Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India
vinayak.musale@mitwpu.edu.in
[2]HACK-X Security, Pune, Maharashtra, India
yash@hackxsecurity.com
[3]School of Computer Engineering & Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India
indraneeltilloo@gmail.com
[4]School of Computer Engineering & Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India
nirmalvyas85@gmail.com
[5]School of Computer Engineering & Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India
langehakshit@gmail.com
[6]School of Computer Engineering & Technology, Dr. Vishwanath Karad MIT World Peace University, Pune, Maharashtra, India
barbhuyan.nabil@gmail.com

Abstract - **Penetration testing is a vital component in assuring the security of digital assets. It entails simulating an assault on a system in order to find weaknesses that attackers could exploit. It is critical to have a tool that can detect many sorts of vulnerabilities and automate the process of detecting risk factors in order to conduct efficient penetration testing. The purpose of this project is to develop an advanced penetration testing tool capable of detecting feature and endpoint vulnerabilities, automating the process of identifying potential risks, and improving testing efficiency. The tool will include feature detection and endpoint detection capabilities, which will allow it to identify potential vulnerabilities and risks connected with a certain asset. It will also include CVSS (Common Vulnerability Scoring System) automation to help prioritize vulnerabilities based on severity. This will allow security teams to prioritize the most serious vulnerabilities. The creation of an advanced penetration testing tool will have several advantages. To begin with, it will eliminate human error from the testing process, resulting in better outcomes. Second, it will increase testing efficiency, allowing security professionals to perform more tests in less time. Ultimately, it will enable security professionals to more correctly detect potential risks and vulnerabilities, resulting in improved digital asset security.**

Keywords - Common Vulnerability Scoring System, Endpoint Detection, Feature Detection, Penetration Testing, VAPT.

1. INTRODUCTION

Penetration testing is essential to ensure the security of digital assets. The need for penetration testing arises due to the increasing sophistication of cyberattacks, which target vulnerabilities in an organization's systems, software, or network. These attacks can result in the loss of sensitive data, financial loss, damage to reputation, and legal repercussions.

Vulnerability Assessment and Penetration Testing (VAPT) are proactive approaches that help organizations identify potential security risks before they can be exploited by attackers. VAPT identifies weaknesses in the system, network, or software and provides a detailed report on the vulnerabilities found, along with recommendations to address them.

Appendices

A. Base Papers

Title - Web Application Safety by Penetration Testing

Year of Publication - 2018

Special Issue based on proceedings of 4TH International Conference on Cyber Security (ICCS) 2018

Web Application Safety by Penetration Testing

Ashikali Hasan[a], Dr. Divyakant Meva[b]

[a]*Research Scholar, Marwadi University, Gujarat, India*
[b]*Associate Professor, Marwadi University, Gujarat, India*

Abstract: By taking advantage of vulnerability, Cyber criminals is easily able to steal confidential data of the ICT, results in heavy loss. Vulnerability Assessment and penetration testing is a special approach to eliminate various security threats from the web application. By focusing high risk vulnerability such as SQL Injection, Cross Site Scripting, Local File Inclusion and Remote File Inclusion, in this paper, we have surveyed literatures to study the general mechanics of VAPT process and gather tools which can be useful during VAPT process.

Keywords—component; Vulnerability Assessment and Penetration Testing, Web Application Security Testing, SQL Injection, Cross Site Scripting, Local File Inclusion, Remote File Inclusion.

1. Introduction

The web application can be affected by various logical and technical vulnerabilities. SQL injection, cross site scripting, remote file inclusion and local inclusion are the examples of technical vulnerability (Ashikali M. Hasan; Divyakant T. Meva; Anil K. Roy; Jignesh Doshi, Dec 2017). These vulnerabilities affect the security of web application. Vulnerability can be occurred due to various reasons such as due to poor programing or due to an outdated system (Hossain Shahriar, Mohammad Zulkernine, June 2012).

Although web application can be secure by various ways wherein Vulnerability Assessment and Penetration Testing (VAPT) process is a special approach to secure web application from both logical and wide range of technical vulnerability. This approach audit web application security and also can be used to secure associated layers. VAPT includes to audit system for finding vulnerabilities, which may be existing in the system, exploit vulnerability same as an attacker exploit and produce data which represent the risk level of the system (Ashikali M. Hasan; Divyakant T. Meva; Anil K. Roy; Jignesh Doshi, Dec 2017).

In purpose of dive little deeper in the area of vulnerability assessment and penetration in this paper we have analyzed overview of the penetration testing process and its limitations and includes various tools which are helpful to conduct VAPT process of these high-risk vulnerabilities.

The paper is presented as follows: section 2 provides background information on our study, section 3 describe literature survey, section V present the general mechanism of VAPT, section VI presents the tools used in VAPT section VII summarizes the contributions of our research and proposes the future work.

2. Background

Researcher considers that SQL Injection, Cross Site Scripting, Remote File Inclusion and Local file inclusion are the high-risk vulnerabilities (Dimitris Mitropoulos; Panagiotis Louridas; Michalis Polychronakis; Angelos D. Keromytis, 2017,). OWASP also included these threats in Top 10 High Risk of web application. These technical vulnerabilities occurred when the web application processes data without proper filtration or validation.

2.1. SQL Injection

SQL Injection vulnerability may affect to dynamic web application which stored data in the associated database. Through SQL Injection, attacker passes malicious code to SQL Server through inserting it in the strings. This malicious code is commonly known as payloads that instruct the database server to retrieve specific information from database (Rahul Johari; Pankaj Sharma, 2012).

By taking advantage of an SQL injection vulnerability attacker can download the entire database on his computer machine and enumerate important information such as database version, database user name, table information and sensitive data available in database column such as password, username etc.

In several cases, attackers can perform various operations such as add, modify and delete records in a database or attacker may able to execute system-level commands and can be successful to launch additional attacks such as denial of service. These additional attacks are depended on the role and privilege set in SQL server of target machine (OWASP, 2016).

2.2. Cross Site Scripting

Cross site scripting also known as XSS is scripting attack in which attacker injects or execute code through the browser at user side for the purpose to steal information of the user's credential. Attacker attempt to steal the user's credential through the vulnerable web application by executing the payloads at client side. The typical example scenario is attacker may inject the payload to the vulnerable field of web application and when the user visits the page at the time payload placed in the page steal other user's cookie and send it to the attacker or may redirect users to phishing sites (M. Ridwan Zalbina; Tri Wanda Septian; Deris Stiawan; Moh. Yazid Idris; Ahmad Heryanto; Rahmat Budiarto, 2017). There are three types of XSS attack known which are persistent, not persistent and DOM based cross site scripting (OWASP Testing Guide v2, n.d.).

Electronic copy available at: https://ssrn.com/abstract=3315587

Title - Design of Web Security Penetration Test System Based on Attack and Defense Game

Year of Publication - 2022

Hindawi
Scientific Programming
Volume 2022, Article ID 8645969, 11 pages
https://doi.org/10.1155/2022/8645969

Research Article

Design of Web Security Penetration Test System Based on Attack and Defense Game

Bing Song, Li Sun, and Zhihong Qin

Department of Network Security, Henan Police College, Zhengzhou 450000, China

Correspondence should be addressed to Bing Song; songbing@hnp.edu.cn

Received 22 April 2022; Revised 10 May 2022; Accepted 18 May 2022; Published 10 June 2022

Academic Editor: Lianhui Li

Copyright © 2022 Bing Song et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

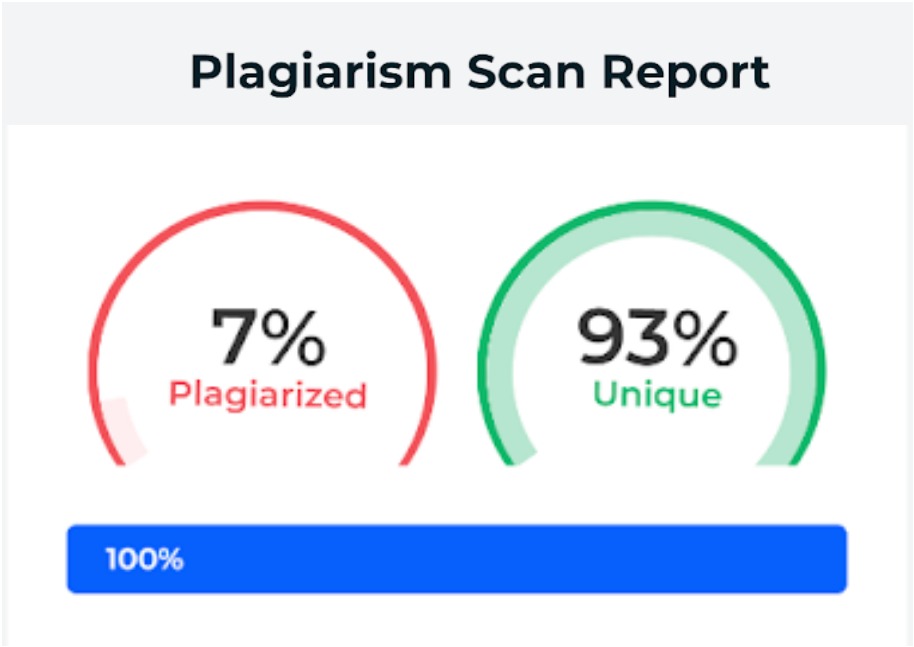
Some sensitive data in the network will be leaked due to the loopholes or weaknesses of the web system itself, which will bring potential harm to the society or the public. Aiming at this, this study carries out the design of web security penetration test system. A test scheme comparing single method with an automatic comprehensive test method is designed. Based on this scheme, an automatic penetration test system script used under the terminal operation page is tested and designed. A security evaluation algorithm based on the prediction results of the game between attack and defense is proposed. Through this algorithm, different website systems are evaluated and scored, and the test results are compared through scoring. The automatic penetration test integration system designed and implemented in this study can meet the main objectives of web security and the protection requirements of websites against general, routine, and universal security attacks. The proposed evaluation algorithm is more detailed, accurate, and reference in scoring.

1. Introduction

With the popularization of Internet and the rapid development of web application technology, Internet provides an important basic platform for web applications, on which more and more web applications are set up [1–6]. Common online shopping malls, online banking, and other web applications have greatly changed people's lifestyle [7–10]. They can easily shop or deal with financial problems without leaving home. However, these new technologies not only bring convenience to our work life and even learning but also bring great risks that we have never had before. Due to the maturity of network technology, the threshold of web application attack technology is getting lower and lower. Hackers gradually transfer the attack object from the network server to the web application [11–14]. According to Gartner's survey data, 75% of information security attacks occur on Web applications, not on the network level. At the same time, it is also found that two-thirds of web applications are very vulnerable and vulnerable to attacks. However, it is a pity that many enterprises spend a lot of energy and financial resources on network security and server security and do not pay attention to the security problems of web applications, leaving an opportunity for hackers [15–17].

The main reason for the vulnerability of web applications is that users can submit data arbitrarily, but the server-side does not carry out reasonable verification. From the perspective of the software itself, the main reason is that the cycle of web application development is getting shorter and shorter, and the level of developers is uneven, which leads to the incomplete consideration of security issues in the process of software development. From the perspective of software deployment and configuration management, the staff may be careless, so there are great security risks. A comprehensive penetration test must be conducted on the web application before the attacker launches an attack to ensure the security of the web application and prevent it from happening [16, 17]. Due to the uneven level of web application developers and the shorter development cycle, it is inevitable for web applications to have security vulnerabilities. The traditional way to ensure network security is through firewall and IDS/IPS. It works on the network layer, while the security penetration test process works on HTTP protocol. It can make up for the deficiency of firewall relative static

B. Plagiarism Report



C. Individual Contribution

Problem Statement: Advanced Web Penetration Testing Tool

Name of the Student: Indraneel Tilloo

Module Title: Scripting Module, User Interface Development Module

Project's Module Objectives - To develop the scripts for various injections and to work on the GUI part of the application.

Project's Module Scope - The scripts would be able to identify the vulnerabilities without any manual intervention, thus increasing the efficiency and reducing the testing time. A user-friendly GUI would help new clients to easily understand the tool and have a smooth interaction with it.

Project's Module(s) - Feature and Endpoint Detection Module, Scripting Module, Machine Learning Module, User Interface Development Module.

Problem Statement: Advanced Web Penetration Testing Tool

Name of the Student: Nirmal Vyas

Module Title: Scripting Module, User Interface Development Module

Project's Module Objectives - To develop the scripts for various injections and to work on the GUI part of the application.

Project's Module Scope - The scripts would be able to identify the vulnerabilities without any manual intervention, thus increasing the efficiency and reducing the testing time. A user-friendly GUI would help new clients to easily understand the tool and have a smooth interaction with it.

Project's Module(s) - Feature and Endpoint Detection Module, Scripting Module, Machine Learning Module, User Interface Development Module.

Problem Statement: Advanced Web Penetration Testing Tool

Name of the Student: Akshit Langeh

Module Title: Machine Learning Module, Feature and Endpoint Detection Module

Project's Module Objectives - To predict the impact of the found vulnerabilities and to extract all the features/input points present in the domain or the website.

Project's Module Scope - The CVSS model will be able to predict the combined impact of the vulnerabilities when trained on the vulnerability vector extracted from the report. The feature detection algorithm will be able to extract all the input points present in the domain and will help the tester to send payloads to these input points. The Endpoint Detection algorithm will extract all the available endpoints in the given web domain.

Project's Module(s) - Feature and Endpoint Detection Module, Scripting Module, Machine Learning Module, User Interface Development Module.

Problem Statement: Advanced Web Penetration Testing Tool

Name of the Student: Nabil Barbhuyan

Module Title: Machine Learning Module, Feature and Endpoint Detection Module

Project's Module Objectives - To predict the impact of the found vulnerabilities and to extract all the endpoints present in the domain or the website.

Project's Module Scope - The CVSS model will be able to predict the combined impact of the vulnerabilities when trained on the vulnerability vector extracted from the report. The feature detection algorithm will be able to extract all the input points present in the domain and will help the tester to send payloads to these input points. The Endpoint Detection algorithm will extract all the available endpoints in the given web domain.

Project's Module(s) - Feature and Endpoint Detection Module, Scripting Module, Machine Learning Module, User Interface Development Module.

D. Project to Outcome mapping

Objectives -

1. To create a tool capable of detecting various vulnerabilities and injections which could compromise an asset.
2. To build a feature detection script.
3. To build an endpoint detection script.
4. To develop an independent CVSS calculator.
5. To develop a CVSS automation system.

Sr. No.	PRN No.	Student Name	Individual Project Student Specific Objectives	Learning Outcomes mapped (To be filled by Guide)
1.	1032191334	Indraneel Tilloo	Developing the scripts for injections (SQL injection, XSS injection and CRLF injection). To work on developing the GUI.	
2.	1032191456	Nirmal Vyas	Developing the scripts for injections (SSTI injection, Command injection & CRLF injection). To work on developing the GUI.	
3.	1032192137	Akshit Langeh	Feature Detection and CVSS automation with the help of ML algorithm (Support Vector Machine).	
4.	1032190871	Nabil Barbhuyan	End-point Detection and CVSS score calculator with the help of ML algorithm (Random Forest).	