

AUTOMATIC WEB SECURITY SCANNER

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and
Engineering

By

GUHAN PRASAD.K (Reg. No. 39290027)

SAI KRISHNA.P (Reg. No. 38290088)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

**SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU**

MAY 22



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **GUHAN PRASAD.K (Reg. No. 38290104)** and **SAI KRISHNA.P (Reg. No. 38290104)** who carried out the project entitled “**AUTOMATIC WEB SECURITY SCANNER**” under my supervision from November 2021 to May 2022.

Internal Guide

Dr. M. SELVI, M.E., Ph.D.,

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.,

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **GUHAN PRASAD.K (Reg. No. 39290027)** hereby declare that the Project Report entitled "**AUTOMATIC WEB SECURITY SCANNER**" done by me under the guidance of **Dr. M. SELVI, M.E., Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Science degree in Computer Science.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D.**, Dean, School of Computing and **Dr. L. LAKSHMANAN, M.E., Ph.D.**, and **Dr. S. VIGNESHWARI, M.E., Ph.D.**, Head of the Department, Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. M. SELVI, M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Computer Science and Engineering who were helpful in many ways For the Completion of the project.

ABSTRACT

The world is exceedingly reliant on the Internet. Nowadays, web security is the biggest challenge in the corporate world. It is considered as the principal framework for the worldwide data society. Web applications are prone to security attacks. Web security is securing a web application layer from attacks by unauthorized users.

A lot of the issues that occur over a web application is mainly due to the improper input provided by the client. Finding the proper tools requires deep understanding and several steps, including analyzing the development environment, business needs, and the web applications' complexity. During the testing phase used this tool I developed to find real world Security vulnerability on the organization like US DOD And Apple.

Entering a web application can be vulnerable because of various vulnerabilities and attacks by hackers. Those vulnerabilities are: Clickjacking, Host Header Injection, Open Redirect, CORS, etc. Security tools take a major part in web applications which are used to check whether the web application is not vulnerable to the user. it works fine on the windows machine as well with some prerequisite to be installed on the windows machine this tool is which was developed to scan the security vulnerability for single host (Domains of the organization) on List faster with this tool I developed through this project period.

TABLE OF CONTENTS

Chapter No.	TITLE	Page
	ABSTRACT	
	LIST OF FIGURES	
1	INTRODUCTION	
	1.1 OVERVIEW OF PROJECT	
2	LITERATURE SURVEY	
3	AIM AND SCOPE OF PROJECT	
	3.1 AIM OF THE PROJECT	
	3.2 SCOPE AND OBJECTIVES	
	3.3 SYSTEM REQUIREMENTS	
	3.3.1 HARDWARE REQUIREMENTS	
	3.3.2 SOFTWARE REQUIREMENTS	
	3.4 SOFTWARE USED	
	3.4.1 PYTHON LANGUAGE	
	3.4.2 PYTHON CHARACTERISTICS	
	3.5 ANACONDA NAVIGATOR	
	3.5.1 ANACONDA	
	3.5.2 APPLICATIONS	
	3.5.3 VS CODE	

4	EXPERIMENTAL OR MATERIAL METHODS	12
	4.1 DESIGN METHODOLOGY	12
	4.1.1 EXISTING SYSTEM	12
	4.1.2 PROPOSED SYSTEM	12
	4.2 WEB SECURITY	13
	4.3 VULNERABILITY	14
	4.4 REVERSE IP	15
	4.5 NSLOOKUP	15
	4.6 CLICK JACKING	16
	4.7 OPEN REDIRECT	16
	4.8 HOST HEADER INJECTION	17
	4.9 CMS DETECT	17
	4.10 MODULE DESCRIPTION	18
	4.11 ARCHITECTURE DIAGRAM	20
	4.12 METHODOLOGY	20
5	RESULTS AND PERFORMANCE ANALYSIS	21
	5.1 RESULTS	21
6	CONCLUSION AND FUTURE ENHANCEMENT	22
	6.1 CONCLUSION	22
	6.2 FUTURE ENHANCEMENT	22
	REFERENCES	23
	APPENDIX	24
	A. SCREENSHOTS	24
	B. SOURCE CODE	28

LIST OF FIGURES

FIGURE NO:	FIGURE NAME	PAGE NO
3.1	Anaconda Distribution	10
3.2	Anaconda Navigator Home Page	11
4.1	Module Diagram	16
4.2	Architecture Diagram	17

CHAPTER 1

1. INTRODUCTION

1.1 OVERVIEW OF PROJECT

Web Security Scanner identifies security vulnerabilities in your App Engine, Google Kubernetes Engine (GKE), and Compute Engine web applications. It crawls your application, following all links within the scope of your starting URLs, and attempts to exercise as many user inputs and event handlers as possible. Currently, Web Security Scanner only supports public URLs and IPs that aren't behind a firewall.

In this study the working model was proposed and developed to find 3 type of vulnerabilities and 3 information gathering tools An Automatic Web Security Scanner. Which can be used on the real world to find this security vulnerability During the testing phase used this tool I developed to find real world Security vulnerability on the organization.

This web application vulnerability scanner is used to find the issue we discussed above i.e., HOST Header Injection to a single list of targets of any top organization to find the vulnerability with less false positive and faster and to solve real world problem.

CHAPTER 2

2. LITERATURE SURVEY

1. Studying Open-Source Vulnerability Scanners for Vulnerabilities in Web Applications.

Authors: Deepika Sagar; Sahil Kukreja; Jwngfu Brahma; Shobha Tyagi; Prateek Jain

During the past few decades, the digital market has seen an enormous growth in terms of cyber technologies and web applications. With the growth in digitalization the amount of risk is also mounting. A small mistake is capable of making the whole web application vulnerable to the attackers seeking it. Therefore, to save the developer's time, web application scanners are well placed to check for a group of known vulnerabilities all together. In our work, we have evaluated OWASP top 10 threats with three vulnerability scanners w3af, Skipfish and OWASP Zed Attack Proxy on vulnerable applications like DVWA. Scanning process starts with the insertion of the targeted vulnerable web application URL. A complete analyzed report is formed in each scenario that is further analyzed with the reports of other vulnerable web applications gathered through the application of the same process. At last, the resultant running time of each scanner is compared to obtain the final tool that work efficiently with minimal time consumption. From three different dataset gathered from the tested scanning tools we conclude that OWASP ZAP performed better than the other scanning tools mentioned in this paper.

2. Web Vulnerability Scanner by Using HTTP Method

Authors: Harshala P. Patil; Prof. Promod. B Gosavi,

Today website security is the most important feature of securing an organization and should be given higher priority. Day by day hackers are mainly targeting on informative websites and web-based applications like forms, sensitive area like login pages, shopping carts, dynamic pages etc. Insecure web applications cause uploading backdoors on the server which allows access to databases, website

hosted servers and also allow hackers to perform illegal activities using the host server like email spamming, proxy. A victim's website can be used for criminal activities, while illegally using website's bandwidth and making its owner liable for these unlawful acts. While developing the websites, many times developers/site owners forget to remove sensitive data from website which is not supposed to be exposed to public users. Such data consists of untested vulnerable forms, database backup, and site backup in compressed format. A hacker tries to search such kind of data and tries to collect important information from it like login detail from that day

3. Software Assurance Tools: Web Application Security Scanner Functional Specification Version 1.0

Authors: Paul E. Black; Elizabeth Fong; Vadim Okun; Romain Gaucher

Software assurance tools are a fundamental resource for providing an assurance argument for today's software applications throughout the software development lifecycle (SDLC). Software requirements, design models, source code, and executable code are analyzed by tools in order to determine if an application is secure. This document specifies the functional behavior of one class of software assurance tool: the web application security scanner tool. Due to the widespread use of the World Wide Web and profiler action of web application vulnerabilities, application-level web security and assurance requires major attention. This specification defines a minimum capability to help software professionals understand how a tool will meet their software assurance needs.

4. A Web Vulnerability Scanner

Authors: Stefan Kais; Engin Kirda; Christopher Kruegel; and Nenad Jovanovic,

As the popularity of the web increases and web applications become tools of everyday use, the role of web security has been gaining importance as well. The last years have shown a significant increase in the number of web-based attacks. For example, there has been extensive press coverage of recent security incidences involving the loss of sensitive credit card information belonging to millions of customers. Many web application security vulnerabilities result from generic input validation problems. Examples of such vulnerabilities are SQL

injection and Cross-Site Scripting (XSS). Although the majority of web vulnerabilities are easy to understand and to avoid, many web developers are, unfortunately, not security-aware. As a result, there exist many web sites on the Internet that are vulnerable. This paper demonstrates how easy it is for attackers to automatically discover and exploit application-level vulnerabilities in a large number of web applications. To this end, we developed SecuBat, a generic and Modular web vulnerability scanner that, similar to a port scanner, automatically analyzes web sites with the aim of finding exploitable SQL injection and XSS vulnerabilities. Using SecuBat, we were able to find many potentially vulnerable web sites. To verify the accuracy of SecuBat, we picked one hundred interesting web sites from the potential victim list for further analysis and confirmed exploitable flaws in the identified web pages. Among our victims were well-known global companies and a finance ministry. Of course, we notified the administrators of vulnerable sites about potential security problems. More than fifty responded to request additional information or to report that the security hole was close

CHAPTER 3

3. AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 AIM OF THE PROJECT

Aim of this project / study to develop an automatic web security scanner tool which can be used in the real-world penetration testing and red teaming scenario and find real world security issue so that it can be fixed by the organization/company before a bad or black hat hacker finds the issue and exploit it to do malicious activity with the security issue.

3.2 SCOPE AND OBJECTIVE

The Scope of the project is to scan multiple security vulnerability of a domain we use python as a base programming language. The future of this scanner can be great as well this can be used as a part of huge web security scanning framework. Which can be useful to all web testers and organization/company, when compare to other online.

3.3 SYSTEM REQUIREMENTS

3.3.1 Hardware Requirements:

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. The minimal hardware requirements are as follows,

1. Processor : Pentium IV
2. RAM : 8 GB
3. Processor : 2.4 GHz
4. Main Memory : 8GB RAM
5. Hard Disk Drive : 1tb
6. Keyboard : 104 Keys

3.3.2 Software Requirements:

Software requirements deals with defining resource requirements and prerequisites that needs to be installed on a computer to provide functioning of an application. The minimal software requirements are as follows,

1. Front end : python
2. Dataset : csv
3. IDE : anaconda, VS Code
4. Operating System : Windows

3.4 SOFTWARE USED:

3.4.1 Python Language

Python is an object-oriented programming language created by Guido Rossum in 1989. It is ideally designed for rapid prototyping of complex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies use the Python programming language include NASA, Google, YouTube, BitTorrent, etc. Python programming is widely used in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced fields of Computer Science. Python had deep focus on code readability & this class will teach you python from basics.

3.4.2 Python Characteristics

- It provides rich data types and easier to read syntax than any other programming languages
- It is a platform independent scripted language with full access to operating system API's
- Compared to other programming languages, it allows more run-time flexibility
- It includes the basic text manipulation facilities of Perl and Awk
- A module in Python may have one or more classes and free functions
- Libraries in Python are cross-platform compatible with Linux, Macintosh, and Windows
- For building large applications, Python can be compiled to byte-code
- Python supports functional and structured programming as well as OOP
- It supports interactive mode that allows interacting Testing and debugging of snippets of code
- In Python, since there is no compilation step, editing, debugging and testing is fast.

3.5 ANACONDA NAVIGATOR:

3.5.1 Anaconda:

Anaconda is a free and open source, easy to install distribution of Python and R programming languages. Anaconda provides a working environment which is used for scientific computing, data science, statistical analysis and machine learning.

The latest distribution of Anaconda is Anaconda 5.3 and is released in October, 2018. It has the conda package, environment manager and a collection of 1000+ open-source packages long with free community support.

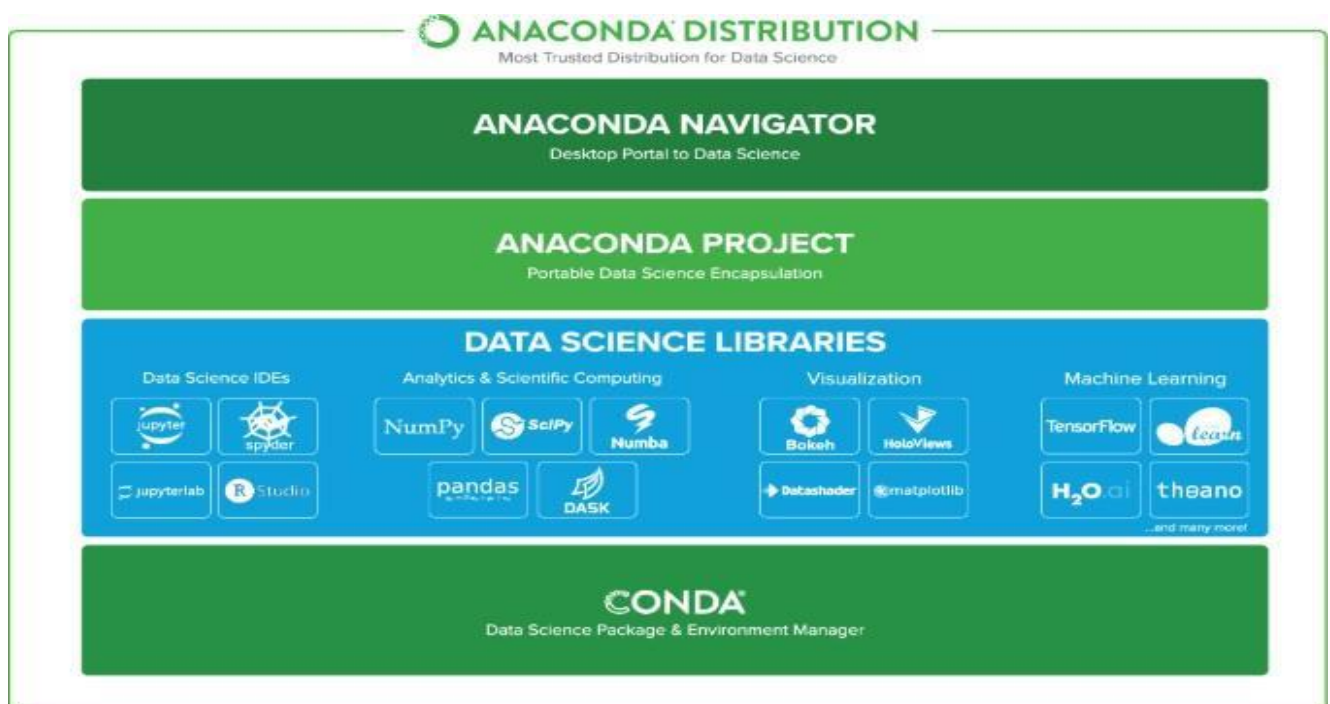
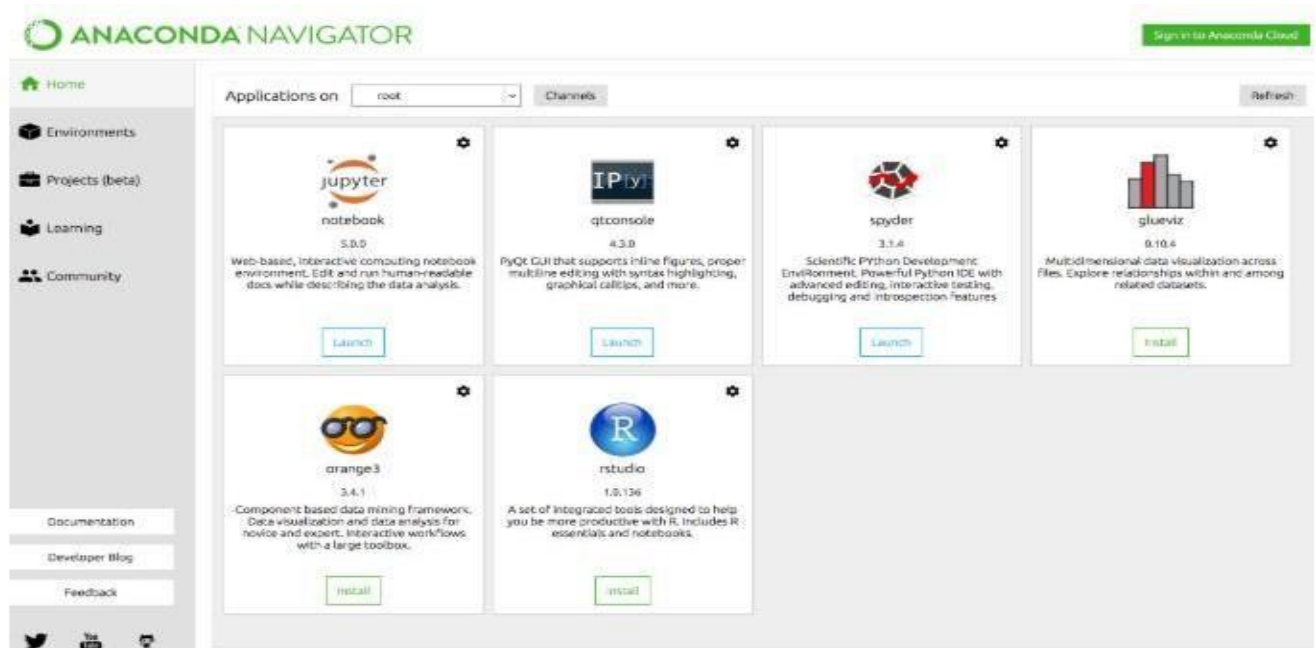


Fig 3.1: Anaconda Distribution

3.5.2 What is Anaconda Navigator?

Anaconda Navigator is a desktop graphical user interface (GUI) included in the Anaconda distribution. It allows us to launch applications provided in the Anaconda distribution and easily manage conda packages, environments and channels without the use of command-line commands. It is available for Windows, macOS and Linux.



Anaconda Navigator

Fig 3.2: Anaconda Navigator Home Page

3.5.3 Applications of Anaconda

The Anaconda distribution comes with the following applications along with Anaconda Navigator.

1. JupyterLab
2. Jupyter Notebook
3. Qt Console
4. Spyder
5. Glueviz
6. Orange3
7. RStudio
8. Vs Code

3.5.4 VS Code:

VS Code is a free, open source streamlined cross-platform code editor with excellent support for Python code editing, IntelliSense, debugging, linting, version control, and more. Additionally, the Python Extension for Visual Studio Code tailors VS Code into a Python IDE. It is a streamlined code editor with support for development operations like debugging, task running and version control.

VS Code is free for both private and commercial use, runs on Windows, macOS, and Linux, and includes support for linting, debugging, task running, version control and Git integration, IntelliSense code completion, and conda environments. VS Code is openly extensible and many extensions are available.

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE.

CHAPTER 4

4. EXPERIMENTAL OR MATERIAL AND METHODS

4.1 DESIGN METHODOLOGY

4.1.1 Existing System:

In existing system, we used k-nearest neighbor (k-NN) to classify the genres. This doesn't give an absolute reasonable correlation between learning strategies for classification of music genre. It uses filter modeling before Piecewise Gaussian Modeling. However, these improvements are not statistically significant. This procedure does not increase classification accuracy and it doesn't achieve the efficiency prediction.

4.1.2 Proposed System:

Support Vector Machine (SVMs) have been actively used for various music classification tasks such as music tag- ging, genre classification and user-item latent feature prediction for recommendation. SVMs and KNNs assume features that are in different levels of hierarchy and can be extracted by convolutional kernels. The hierarchical features are learned to achieve a given task during supervised training.

4.2 WEB SECURITY:

Web security must be a critical priority for every organization. Along with email, the web is one of the top vectors for cyberattacks. The web and the use of DNS services specifically are part of 91% of all malware attacks, and email and web together are a key part for 99% of successful breaches.

While the importance of web security is undisputed, protecting against web security threats grows more challenging each day. From thwarting attacks to dealing with limits in skills and resources, IT security departments face serious challenges when trying to secure the web. Web security is crucial to the smooth operation of any business that uses computers. If a website is hacked or hackers are able to manipulate your systems or software, your website—and even your entire network—can be brought down, halting business operations.

In the past, security teams have deployed a collection of on-premises solutions to manage email and web security. But increasingly organizations are turning to comprehensive email and web security solutions – via integrated, cloud-based technologies that simplify the task and reduce the cost of reducing risk. And because attackers often leverage email and web channels together, a seamless and scalable strategy for protecting both is essential.

4.3 VULNERABILITY:

In cybersecurity, a vulnerability is a weakness that can be exploited by cybercriminals to gain unauthorized access to a computer system. After exploiting a vulnerability, a cyberattack can run malicious code, install malware and even steal sensitive data. Vulnerabilities can be exploited by a variety of methods including SQL injection, buffer overflows, (XSS) and open-source exploit kits that look for known vulnerabilities and security weaknesses in web applications.

Many vulnerabilities impact popular software, placing the many customers using the software at a heightened risk of a data breach, or supply chain attack. Such zero-day exploits are registered by MITRE as a Common Vulnerability Exposure (CVE).

A security risk is often incorrectly classified as a vulnerability. The use of vulnerability with the same meaning of risk can lead to confusion. The risk is the potential of a significant impact resulting from the exploit of a vulnerability. Then there are vulnerabilities without risk: for example, when the affected asset has no value. A vulnerability with one or more known instances of working and fully implemented attacks is classified as an exploitable vulnerability—a vulnerability for which an exploit exists.

The window of vulnerability is the time from when the security hole was introduced or manifested in deployed software, to when access was removed, a security fix was available/deployed, or the attacker was disabled—see zero-day attack.

4.4 REVERSE IP:

Reverse IP Lookup is an incredibly powerful tool with many high-value business applications. Retrieve a list of all domains using the same IP address as you, and sharing the same resources Track down malicious behavior of phishing or scamming websites that reside on the same host.

Reverse IP lookup, also known as reverse DNS lookup, is the process of querying the Domain Name System (DNS) to determine the domain name associated with an Internet Protocol (IP) address. Put in simpler terms, this tool looks up the owners of IP addresses (the numerical label assigned to devices on a network). So, while your analytics software's IP tracking may tell you that your website has been visited by a user from IP address 73.50.51.77, reverse IP lookup can tell you that this visitor is from Acme Corporation. And yes, this is legal.

4.5 NsLookup:

nslookup is the name of a program that lets an Internet server administrator or any computer user enter a host name (for example, "whatismyip.com") and find out the corresponding IP address or domain name system (DNS) record. The user can also enter a command for it to do a reverse DNS lookup and find the host name for an IP address that is specified. Nslookup is typically a command-line tool, which means that it uses the command-line structure familiar to many users who have used older PC-DOS operating systems. In order to use command-line tools, users may have to shell out of a Windows-based environment in order to access the command-line interface. In addition to looking up server information, Nslookup can be used to test IP connections. Users can set items like retry and timeout, designate a root server, or get debugging information. Nslookup can also be used to check mail exchanger or MX records that designate routing for emails according to what servers are attached to a particular domain.

4.6 Click Jacking:

Clickjacking, also known as a “UI redress attack”, is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they were intending to click on the top-level page. Thus, the attacker is “hijacking” clicks meant for their page and routing them to another page, most likely owned by another application, domain, or both.

Users think they are using a web page’s normal UI, but in fact there is a hidden UI in control; in other words, the UI has been redressed. When users click something, they think is safe, the hidden UI performs a different action. Using a similar technique, keystrokes can also be hijacked. With a carefully crafted combination of stylesheets, iframes, and text boxes, a user can be led to believe they are typing in the password to their email or bank account, but are instead typing into an invisible frame controlled by the attacker.

4.7 Open Redirect:

Open redirection vulnerabilities arise when an application incorporates user-controllable data into the target of a redirection in an unsafe way. An attacker can construct a URL within the application that causes a redirection to an arbitrary external domain.

This behavior can be leveraged to facilitate phishing attacks against users of the application. The ability to use an authentic application URL, targeting the correct domain and with a valid SSL certificate (if SSL is used), lends credibility to the phishing attack because many users, even if they verify these features, will not notice the subsequent redirection to a different domain.

4.8 Host Header Injection:

A web server commonly hosts several web applications on the same IP address, referring to each application via the virtual host. In an incoming HTTP request, web servers often dispatch the request to the target virtual host based on the value supplied in the Host header. Without proper validation of the header value, the attacker can supply invalid input to cause the web server to:

- Dispatch requests to the first virtual host on the list.
- Perform a redirect to an attacker-controlled domain.
- Perform web cache poisoning.
- Manipulate password reset functionality.
- Allow access to virtual hosts that were not intended to be externally accessible.

4.9 CMS Detect:

A Content Management System, or CMS, is a piece of software designed to help users create and edit a website. It's important to note, however, CMS do much more than help manage the text and image content displayed on webpages. They have evolved to help design the look of websites, track user sessions, handle searches, collect visitor comments, host forums and much more. This CMS Detect shows the designed software of the input target host.

4.10 MODULE DESCRIPTION

Module 1: Getting Input Target Host and Port

Requesting User for input Target Host and Target Port number

A Target port number is always associated with an IP address of a host and the type of transport protocol used for communication. It completes the destination or origination network address of a message. Some Common Port Number are HTTP-80, HTTPS-443, DNS-53.

A Target Host, also known as a URL (uniform resource locator), is an Internet or intranet name that points to a location where a file, directory or website page is hosted. Website addresses can represent the home page of a web site, a script, image, photo, movie or other file made available on a server for viewing, processing or download. They can also be embedded into the code of web pages in the form of hyperlinks to direct the user to other locations on the Internet.

Module 2: Checking Whether the Host Is Alive

To check whether the given input Target host is alive or not we use socket. Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. The server forms the listener socket while the client reaches out to the server.

They are the real backbones behind web browsing. In simpler terms, there is a server and a client. socket instance and passed it two parameters. The first parameter is AF_INET and the second one is SOCK_STREAM. AF_INET refers to the address-family ipv4. The SOCK_STREAM means connection-oriented TCP protocol. Now we can connect to a server using this socket

Module 3: Working of Tools:

After The Host is alive six tools will started import from the sub programs. The Tools divided in two. Three Information gathering tools and Three Vulnerability detection tools. Information Gathering means gathering different kinds of information about the target. It is basically, the first step or the beginning stage of Ethical Hacking, where the penetration testers or hackers (both black hat or white hat) tries to gather all the information about the target. Vulnerability scanners are automated tools that scan web applications to look for security vulnerabilities. They test web applications for common security problems such as cross-site scripting (XSS), SQL injection, and cross-site request forgery (CSRF). The Tools Are:

- Reverse IP
- NsLookup
- Clickjacking
- Open Redirect
- Host Header Injection
- CMS Detect

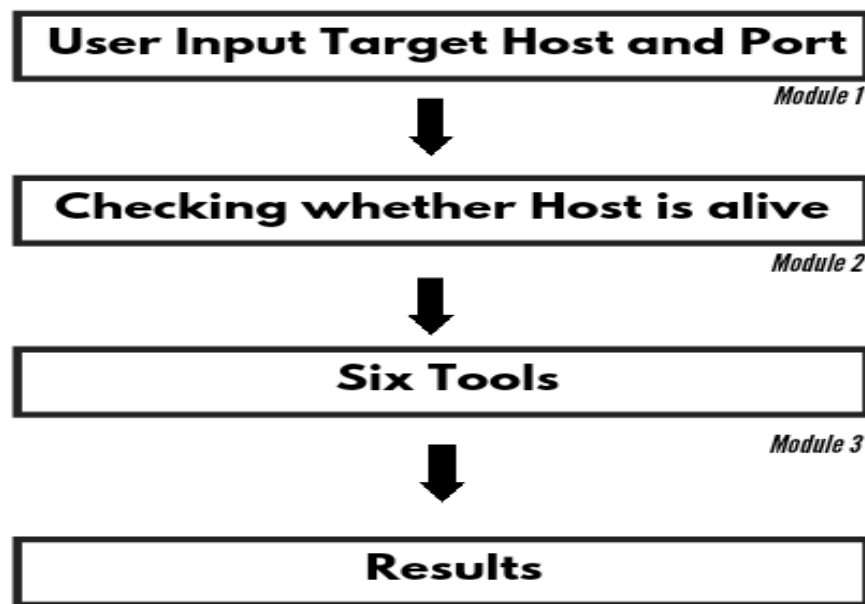


Fig 4.1: Module diagram

4.11 ARCHITECTURE DIAGRAM:

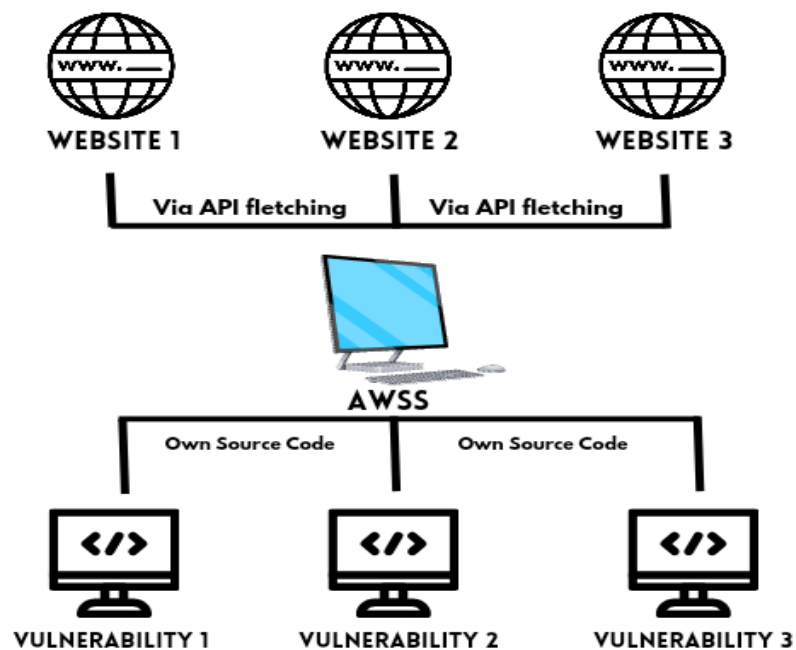


Fig 4.2: Architecture diagram

4.12 METHODOLOGY:

1. Open default browser (preferably chrome) and install Anaconda Navigator and VS code.
2. Ensure that all the software is correctly configured with your computer.
3. In the vs code editor, go to the extensions and install python extension
4. Create a folder in the computer named "AWSS-Project"
5. Open vs code and create a python file and named as "awss"
6. Import all sub program to the main program with import function.
7. Getting Target Host and Target port number as input from user.
8. Checking whether the host is alive or not.
9. If alive, six tools will pop up with numeric value
10. Click 1 for Reverse IP (The reverse Ip of the given host will be executed)
11. Click 2 for NsLookup (The DNS Lookup of the given host will be executed)
12. Click 3 for Clickjacking (Check whether the given host is vulnerable for Clickjacking)
13. Click 4 for OpenRedirect (Test payload of the given host is vulnerable for OpenRedirect)
14. Click 5 for Host Header Injection (Check Whether the given host id vulnerable for host Header Injection)
15. Click 6 for CMS Detect (The CMS of the given host will be executed)
16. Click 7 to exit the Program

CHAPTER 5

5. RESULTS AND PERFORMANCE ANALYSIS

RESULTS:

This tool was successfully developed and tested on the real-world target such as US Department of Defense and may big organization and managed to work perfectly without any false positive till now from the date of development of the tool this working is really smooth and fast in speed which can be used for real world pentesting purpose.

CHAPTER 6

6. CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION:

In the new normal post pandemic the usage of the internet has become a massive growth since we all have moved to the internet there are huge possibility of being victim of the cyber-attack, A cyberattack is any offensive maneuver that targets computer information systems, computer networks, infrastructures, or personal computer devices. An attacker is a person or process that attempts to access data, functions, or other restricted areas of the system without authorization, potentially with malicious intent. Depending on the context, cyberattacks can be part of cyber warfare or cyberterrorism. A cyberattack can be employed by sovereign states, individuals, groups, society, or organizations, and it may originate from an anonymous source. A product that facilitates a cyberattack is sometimes called a cyber weapon.

A cyberattack may steal, alter, or destroy a specified target by hacking into a susceptible system. Cyberattacks can range from installing spyware on a personal computer to attempting to destroy the infrastructure of entire nations. Legal experts are seeking to limit the use of the term to incidents causing physical damage, distinguishing it from the more routine data breaches and broader hacking activities. Cyberattacks have become increasingly sophisticated and dangerous this can't be solved entirely but this can be solved partially and organization can fix the security vulnerability and fix in order reduce the rate of being victim

6.2 FUTURE ENHANCEMENTS:

The scope of the tool is limited by now but it can be extended in many ways it can be used as function if we manage to create a huge vulnerability scanning framework

This can be extended and by making this tool to do an email/slack/discord

notification if the tool manages to find any kind of security vulnerability This project/tool can solve many real-world problems and help the society to be cyber safe.

REFERENCES

- [1]. Matteo, L., Luptak, P., & Mike, H. (2020). OWASP open web application security project, Web Security Testing Guide, 9(4), 290-296.
- [2]. Dafydd Stuttard, Marcus Pinto(2011, August). The Web Application Hackers Handbook: Finding and Exploiting Security Flaws. (pp. 137-140).
- [3]. Dafydd Stuttard, Marcus Pinto (2011, August). The Web Application Hackers Handbook: Finding and Exploiting Security Flaws. (pp. 137-140).
- [4]. Abdulkader A. Alfantookh. An automated universal server level solution for SQL injection security flaw. International Conference on Electrical, Electronic and Computer Engineering, pages 131—135, September 2004.
- [5]. Paolo Tonella and Filippo Ricca. A 2-Layer Model for the White-Box Testing of Web Applications. In IEEE International Workshop on Web Site Evolution (WSE), 2004.
- [6]. Paul E. Black, Michael Kass and Michael Koo, Source Code Security Analysis Tool Functional Specification Version 1.0, NIST Special Publication 500-268, May 2007.
- [7]. Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 2015 IEEE 8th International Conference on 1:399-402
- [8]. <https://en.wikipedia.org/>
- [9]. <https://portswigger.net/web-security/host-header>
- [10].https://owasp.org/www-project-web-security-testing-guide/latest/4Web_Application_Security_Testing/07-Input_Validation_Testing/17-Testing_for_Host_Header_Injection

APPENDIX

SCREENSHOTS:

Tool Output:

```
Help awss.py - Visual Studio Code
PROBLEMS OUTPUT Python: Current File
.d8888888 dP dP dP d8P888888b d8P888888b
d8' 88 88 88 88 88b 88b
88aaaaa88a 88 .8P .8P 'd8aaaaa8 'd8aaaaad8
88 88 88 d8' d8' d8' d8'
88 88 88.d8P8.d8P d8P d8P
88 88 8888'Y88' 888888888P 888888888P

Enter the Target Host : www.example.com
Enter the Target port : 80

Starting WAVES

Checking whether the Target is reachable

Target Alive

Host : www.example.com
Port : 80

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. Host Header Injection
6. CMS Detection
7. Exit

>> |
```

Reverse IP Output:

```
Help awss.py - Visual Studio Code
PROBLEMS OUTPUT Python: Current File

1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. Host Header Injection
6. CMS Detection
7. Exit

>> 1

096666.xyz
457135.xyz
7f000001.5db8d822.rbndr.us
acl.moe
adfs.cicd.ci.egsdlon.dk
adrianstott.com
agyleroster.com
alertmanager.bh.rxx.io
alertmanager.dev.egsdlon.dk
altmannserver.ddns.net
anthonyalazzetti.com
api.dartswiz.com
apple-pi.umi.ch
app03.dev.egsdlon.dk
app04.dev.egsdlon.dk
argocd.cf.rxx.io
artifactory.lach.pw
argand.space
```

NsLookup Output:

```
1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. Host Header Injection
6. CMS Detection
7. Exit

>> 2

A : 93.184.216.34
AAAA : 2606:2800:220:1:248:1893:25c8:1946
TXT : "v=spf1 -all"
TXT : "wgyf8z8cgvm2qmxpnbnldrcltvk4xqfn"
```

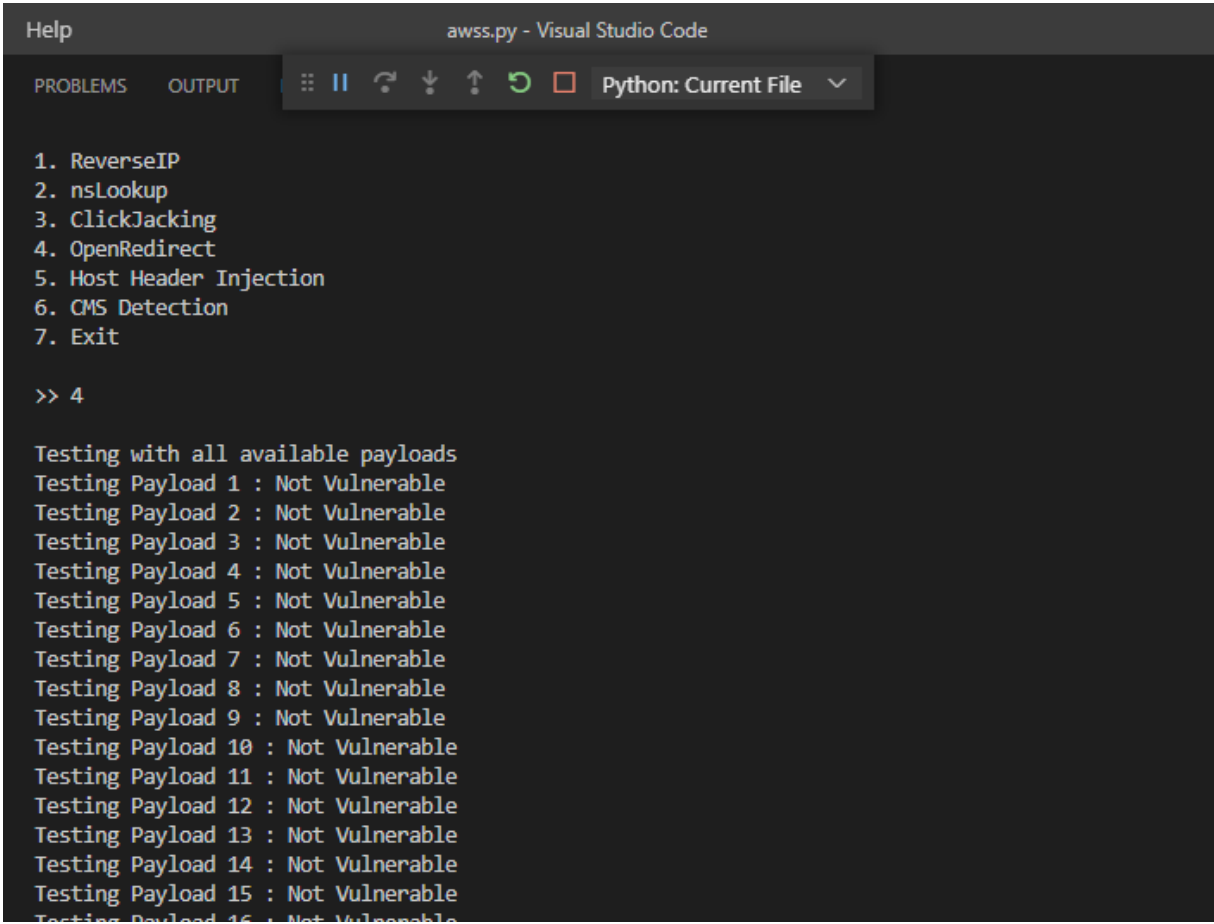
Clickjacking Output:

```
1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. Host Header Injection
6. CMS Detection
7. Exit

>> 3

Website is vulnerable to ClickJacking
```

OpenRedirect Output:

A screenshot of the Visual Studio Code editor interface. The title bar shows 'awss.py - Visual Studio Code'. The menu bar includes 'Help'. The toolbar shows icons for Explorer, Run and Debug, Source Control, Run and Test, and a dropdown menu currently set to 'Python: Current File'. The main editor area displays the following text:

```
1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. Host Header Injection
6. CMS Detection
7. Exit

>> 4

Testing with all available payloads
Testing Payload 1 : Not Vulnerable
Testing Payload 2 : Not Vulnerable
Testing Payload 3 : Not Vulnerable
Testing Payload 4 : Not Vulnerable
Testing Payload 5 : Not Vulnerable
Testing Payload 6 : Not Vulnerable
Testing Payload 7 : Not Vulnerable
Testing Payload 8 : Not Vulnerable
Testing Payload 9 : Not Vulnerable
Testing Payload 10 : Not Vulnerable
Testing Payload 11 : Not Vulnerable
Testing Payload 12 : Not Vulnerable
Testing Payload 13 : Not Vulnerable
Testing Payload 14 : Not Vulnerable
Testing Payload 15 : Not Vulnerable
Testing Payload 16 : Not Vulnerable
```

Host Header Injection:

```
1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. Host Header Injection
6. CMS Detection
7. Exit

>> 5

Not Vulnerable to Host header injection
```

CMS Detect:

```
1. ReverseIP
2. nsLookup
3. ClickJacking
4. OpenRedirect
5. Host Header Injection
6. CMS Detection
7. Exit

>> 6

CMS/Host/Theme not detected
Detected CMS : None
Detected Version : None
```

B. SOURCE CODE

REVERSE IP:

```
from requests import get

def ReverseIP(host, port):
    lookup = 'https://api.hackertarget.com/reverseiplookup/?q=%s' % host
    try:
        result = get(lookup).text
        print(result)
    except:
        print('Invalid IP address')
```

NSLOOK UP:

```
from requests import get

def nsLookup(host, port):
    result = get('http://api.hackertarget.com/dnslookup/?q=' + host).text
    print(result)

host = 'www.skcet.ac.in'
```

CLICKJACKING:

```
from urllib.request import urlopen

def ClickJacking(host, port):

    if port == 80:
        port = 'http://'
    elif port == 443:
        port = 'https://'
    else:
        print("Could'nt fetch data for the given PORT")

    url = (port+host)
```

```

data = urlopen(url)
headers = data.info()

if not "X-Frame-Options" in headers:
    print("Website is vulnerable to ClickJacking")

else:
    print("Website is not Vulnerable to ClickJacking")

```

OPEN REDIRECT:

```

from urllib.request import *

def OpenRedirect(host, port):
    if port == 80:
        port = 'http://'
    elif port == 443:
        port = 'https://'
    else:
        print("Could't fetch data for the given PORT")

    print('Testing with all available payloads')

    i = 2

    url = (port + host)
    q = Request(url)
    q.add_header('X-Forwarded-Host', 'https://google.com')
    a = urlopen(q).getcode()
    if a == 302:
        print("Testing Payload 1 : Vulnerable to X-Forwarded-Host Header")
    else:
        print('Testing Payload 1 : Not Vulnerable')

    f = open("redirectpayloads.txt", "r")
    f1 = f.readlines()

    for x in f1:

        while i < 34:

            try:

                url = (port + host + x)
                data = urlopen(url)
                x = data.getcode()

            except:
                print("Testing Payload %i : Error" % i)
                break

            if x == 302:
                print("Testing Payload %i : Vulnerable" % i)
                print(url)

```

```

        break

    elif x == 200:
        print("Testing Payload %i : Not Vulnerable" % i)
        break

    else:
        print("Testing Payload %i : Not Vulnerbale" % i)
        break

i+=1

```

HOST HEADER INJECTION:

```

import requests

def HostHeader(host, port):
    if port == 80:
        port = 'http://'
    elif port == 443:
        port = 'https://'
    else:
        print("Could'nt fetch data for the given PORT")
    url = (port + host)
    headers = {'Host': 'http://evil.com'}
    response = requests.get(url, headers=headers)
    if 'evil.com' in response.headers:
        print("Vulnerable to Host Header Injection")
    else:
        print("Not Vulnerable to Host header injection")

```

CMS DETECT:

```

import requests

def CMSdetect(domain, port):
    payload = {'key':
'1641c3b9f2b1c8676ceaba95d00f7cf2e3531830c5fa9a6cc5e2d922b2ed7165dcce66',
'url': domain}
    cms_url = "https://whatcms.org/APIEndpoint/Detect"
    response = requests.get(cms_url, params=payload)
    cms_data = response.json()
    cms_info = cms_data['result']
    if cms_info['code'] == 200:
        print('Detected CMS      : %s' % cms_info['name'])
        print('Detected Version : %s' % cms_info['version'])
        print('Confidence      : %s' % cms_info['confidence'])
    else:
        print(cms_info['msg'])
        print('Detected CMS : %s' % cms_info['name'])
        print('Detected Version : %s' % cms_info['version'])

```

MAIN PROGRAM:

```
import socket
from Plugins.reverseip import ReverseIP
from Plugins.nslookup import nsLookup
from Plugins.cmsdetect import CMSdetect
from Vulnerabilities.clickjacking import ClickJacking
from Vulnerabilities.hostheader import HostHeader
from Vulnerabilities.openredirect import OpenRedirect

def Banner():
    print('''
.d8888888 dP dP dP d8P88888b d8P88888b
d8' 88 88 88 88 88b 88b
88aaaaa88a 88 .8P .8P 'd8aaaaa8 'd8aaaaad8
88 88 88 d8' d8' d8' d8'
88 88 88.d8P8.d8P d8P d8P
88 88 8888'Y88' 888888888P 888888888P \n''')

host = None
port = None

# Checking whether the target host is alive or dead
def CheckTarget():

    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    result = s.connect_ex((host, port))

    if result == 0:
        return True
    else:
        return False

# Main Method
def Main():
    Banner()
    global host
    host = input("Enter the Target Host : ")
    global port
    port = int(input("Enter the Target port : "))
    print('')
    print("Starting WAVES \n")
    print("Checking whether the Target is reachable \n")
    # calling CheckTarget method
    if CheckTarget()==True:
        print("Target Alive \n")
        print("Host : " + host)
```

```

        print("Port : %s" % port)
    else:
        print("The Host is Unreachable \n")
        exit()

MainFunctions = {
    1: ReverseIP,
    2: nsLookup,
    3: ClickJacking,
    4: OpenRedirect,
    5: HostHeader,
    6: CMSdetect,
}

def Menu():
    Selection = 1
    while True:
        print('')
        print("1. ReverseIP")
        print("2. nsLookup")
        print("3. ClickJacking")
        print("4. OpenRedirect")
        print("5. Host Header Injection")
        print('6. CMS Detection')
        print("7. Exit")
        print('')
        Selection = int(input(">> "))
        print('')
        if (Selection >= 0) and (Selection < 7):
            MainFunctions[Selection](host, port)
        elif Selection == 7:
            exit()
        else:
            print("Please choose an Appropriate option")

WW

#calling main methods
if __name__ == "__main__":
    Main()
    Menu()

```