

Development of Framework for Vulnerability Assessment activities

A Project Submitted
in Partial fulfilment of the Requirements
for the Internship.



by

Shreyansh Sancheti, 6th Sem, Walchand Institute of Technology

Under the guidance of

Sh. Ankit Sarkar

NATIONAL CRITICAL INFORMATION INFRASTRUCTURE PROTECTION CENTRE

New Delhi

CERTIFICATE

This is to certify that Project report entitled “**Development of Framework for Vulnerability Assessment activities**” submitted by **Shreyansh Sancheti** in partial fulfillment of the requirement for the award of internship at National Critical Information Infrastructure Protection Centre, New Delhi is a record of the candidate’s own work carried out by him under our supervision. The matter embodied in this report is analysis from open source and has not been submitted for the award of any other degree or program.

Date:

Sh. Ankit Sarkar

DECLARATION

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Name- Shreyansh Manish

Sancheti

Date -

Place - New Delhi

ACKNOWLEDGEMENT

It gives me a great sense of pleasure to present the report of the project undertaken by me during internship. I owe a special debt of gratitude to my mentor **Sh. Ankit Sarkar** at NCIIPC, New Delhi for his constant support and guidance throughout the course of my work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for me. It is only his cognizant efforts that my endeavours have seen light of the day.

I also take the opportunity to acknowledge the contributions of **Md. Talib Hasan Ansari**, NCIIPC, New Delhi for his full support and assistance during the development of the project and other members of the department for their kind assistance, cooperation and insights during the development of my project.

Last but not the least, I acknowledge the constant support of my family and friends, for always being there for me and constantly supporting me.

Shreyansh Sancheti

ABSTRACT

Attackers have developed an arsenal of tools and techniques to break into organization's networks and steal valuable information. These tools are mostly used for reconnaissance – information gathering, that can be done manually or in an automated way by using prewritten tools, but running every specific tool with different commands requires time and knowledge. A framework combines these tools with specific commands, and can execute it with desired output. This framework is built keeping in mind that it will be easy to use and upgrade, that's why it is written in python. There are many prewritten tools for performing reconnaissance in an automated way but those special tools need to be executed one by one. But, this framework solves the problem by providing a platform for those tools to be integrated in one frame and perform those same tasks with one hit.

TABLE OF CONTENTS

	Page Number
Certificate	II
Declaration	III
Acknowledgment	IV
Abstract	V
Table of Contents	VI
Chapter 1: Introduction	1
Chapter 2: Background	3
Chapter 3: Technologies used	4
Chapter 4: Description and working	6
Chapter 5: Reconstruction of Framework	11
Conclusion	16
References	17

CHAPTER 1: INTRODUCTION

The world wide web (www) has become ubiquitous with more advanced and upgraded features and services. Government and public organizations are utilizing the services of the internet to achieve their functionality. The rapid digitization and dependency on the internet has opened the doors for hackers to exploit the existing and undiscovered vulnerabilities due to insecure coding practices, during development. It is possible to find and fix the vulnerabilities post development, prior to deployment as well as post deployment using recommended vulnerabilities assessment techniques. The technologies are changing day by day and so are the vulnerabilities. Hence, it is a mandatory requirement of an organization to carry out periodic vulnerability assessments to stay protected from sophisticated and state-of-the-art cyber attacks.

Security bugs are incorporated unintentionally during development while focusing on features and aesthetics as well as the unawareness of security techniques. Sometimes third party tools and plugins are used to achieve better functionality and user-friendliness but unfortunately these sections are vulnerable to several attacks even though secure coding practices are taken care of. Vulnerability assessments are carried out in defined steps to collect the information about possible exploitations.

The users feel once they have set a strong password, they are completely secured whereas they are aware of the darker part of this internet. People with intentions to hurt, steal or harm in any of the ways called as the Hackers (crackers) constitute the darker part of this web. They intrude into the system using certain techniques. So, here comes the role of Ethical Hacker, the good ones. To stay secure from getting hacked, the most basic strategy would be to learn to think like a hacker. The attacks could be easily done once the vulnerabilities for any system are known. To be secure in this world of the internet, one must know how a hacker(cracker) can intrude into their system. [1]

Reconnaissance

[1] Reconnaissance is a set of processes and techniques used to secretly discover and collect information about a target system. Reconnaissance is possible in seven steps listed below:

- Gather preliminary information
- Identifying active machines
- Finding out open ports and access points
- OS fingerprinting
- Disclose all the services on ports
- Network mapping

Scanning and Enumeration

Scanning is a set of procedures for identifying live hosts, ports, and services, discovering Operating system and architecture of target system, Identifying vulnerabilities and threats in the network. Scanning refers to collecting more information using complex and aggressive reconnaissance techniques.[2]

Enumeration is defined as the process of extracting user names, machine names, network resources, shares and services from a system. In this phase, the attacker creates an active connection to the system and performs directed queries to gain more information about the target.[2]

There are two types of reconnaissance methods:

Active : Active reconnaissance is a type of computer attack in which an intruder engages with the targeted system to gather information about vulnerabilities.

Passive : Passive reconnaissance is an attempt to gain information about targeted computers and networks without actively engaging with the systems.[3]

Some vulnerabilities can be identified by using a scanning process which can be automated and that only involves passive methods. Passive methods don't require engaging with the target and thus scanning works differently than other methods. The biggest advantage of passive reconnaissance method over active reconnaissance is that it won't alert the system of the target user even if one performs a scan on the target user. In this framework such methods are used to keep the interaction with the targeted system to the minimum. It will scan for vulnerabilities which don't require any interaction and will be using the best methods available.[3]

Framework

The simple definition for a framework is that it takes care of redundant basic tasks so that sophisticated applications can be written on top of it in a simple manner. Framework handles all the required validations which are required in the application so that the programmer can focus mainly on the application logic. It gives a basic structure around which one can write code. A framework is very useful in building applications which need a sequence to follow.

Open-source intelligence (OSINT) is intelligence collected from publicly available sources. There are lots of tools available over the internet which include all publicly accessible sources of information such as social networking sites, video sharing sites. This framework will integrate all these tools for different purposes like Subdomain enumeration, port-scanning, Vulnerability scanning, SPF records, Zone transfers, SSL scanning, CMS scanning.

CHAPTER 2: BACKGROUND

Recon-ng is a full-featured Web Reconnaissance framework written in Python. Complete with independent modules, database interaction, built in convenience functions, interactive help, and command completion, Recon-ng provides a powerful environment in which open source web-based reconnaissance can be conducted quickly and thoroughly.[4]

This open source available framework needs its own modules to perform reconnaissance, scans and attacks. It is already available in the market and also comes pre-installed with Kali Linux (penetration testing OS).

This project of developing a reconnaissance framework uses a different idea and method to achieve the goal of collecting target information and finding vulnerabilities in the system. How's it different from recon-ng which is present in the market?

Firstly, the developed framework known as recon-tool uses external tools for various activities and then lists out the output in text files. It is simple and easy to use. One can modify the framework by adding more functions and tools. This framework integrates some well known tools and gives the best results.[4]

The Framework can also be further developed by adding some more functionalities like

1. having a proxy server.
2. running tools under tor tunnels.
3. Converting the cli based model to gui.
4. GUI will give the user visual experience.
5. Enable disable some features.

Features of Framework:

- This framework provides automated reconnaissance, all tools run simultaneously while gathering information.
- Manually typing the command and executing consecutively will consume time while having to wait for the output. Whereas this framework is easy to understand and saves time.
- This framework is easy to use as it is a menu driven framework. After running the tool one can easily operate the tool.
- It is very scalable if one wants to add new tools, they will have to create a new function and call that function that is similar to other integration in the source code. Hence it can be expanded with more tools if needed.
- It is written in python because python is much easier to write than other languages. It has easy syntax and modules which makes it easy for one to understand the code.
- The key feature of this framework is that it focuses more on passive reconnaissance which will lessen the engagement with the target system. Other available frameworks in the market used both methods so in reality they engage with the target.

CHAPTER 3: TECHNOLOGIES USED

To configure the proposed application, it needs api keys mentioned in the manual, Python 3.5 or > and Golang (latest or stable version) should be installed on the Linux system. One can directly install it from the shell script attached with a tool or can do it manually.

The framework uses some Python modules namely `os`, `system`, `subprocess`, `multithreading`, `termcolor` & `readline`. All the tools are performed using `subprocess` and `os` modules. Both these modules work similarly but have different methods. But the real difference between these two modules **os** and **subprocess** is the runtime of the processes.

The **multithreading** module in this framework creates threading objects which are useful for executing different functions at the same time.

The **readline** module helps in reading the input of stdin, the reason for using this module is that it reads input by line in a consecutive manner. While this framework is menu driven and input is always taken in a single line format, the readline module can enable the function where one can edit the input lines while writing. If this module is not implemented then python takes input as raw and one can't edit the written input.

The **termcolor** module is also being used here, which gives very interactive colours in the terminal. Various colour themes with settings such as bold, italics and many more. One can also change the foreground and background colour of the text that is being displayed. A very useful module for having a neat and simple looking framework.

The module **shodan-cli** is installed for using shodan through the api key of shodan account. It is useful for port scanning passively.

Tools used for framework:

1. Subfinder, amass, asset finder, wayback urls or archive urls and gau - are used for finding subdomains passively. Results are merged and uniquely arranged. A bunch of tools are used for getting unique results. Sometimes it may happen that one tool is not working properly or giving 50% results and to make sure that one can gather all the subdomains related to the root domain, these tools are used.
2. Findomain - It consists of different api keys like censys, virustotal, shodan, security trails, certspotter, cert.sh.
3. httpx - It is used for probing all active subdomains.
4. Shodan-cli – It is used for passive scanning on unique ip addresses of probed or active domains.
5. Nmap - For active scanning, Nmap is included with ping, service, and version

detection scan with top 1000 ports only because more than 1000 ports will generate load on the pc and will give RVVTAR error.

6. CORSy - Cross-Origin Resource Sharing (CORS) is an HTTP-header based mechanism that allows a server to indicate any other origins (domain, scheme, or port) than its own from which a browser should permit loading of resources.
7. CMSeeK - It scans for CMS of every domain and subdomains that are active.
- 8.
9. GitDorker – This is a tool which looks for sensitive information in Github repositories. It is written in python. Firstly, it collects all the public repositories of a specific organization and then makes a compiled list, it gathers all filenames listed in each repository and compares it with sensitive patterns of data or signatures.
10. De_googlehunter – This is a tool which looks for sensitive information that is publicly available. It is written in python. Firstly, it collects all the publicly available information of the given organization and then lists out the data in text format that matches the information.
11. Dnsrecon – This is a tool which looks for zone - transfer vulnerability. Zone transfer is a vulnerability that occurs when you bring up a new DNS server as a secondary DNS server. Sender Policy Framework is a standard for checking whether a certain IP is authorized to send mail from a given domain. It is designed to detect and block emails that are spam or blocking email spoofing. This tool detects whether the organization or target system has the SPF and DMARC records or not.
12. Nuclei - This is a fast tool that performs scans based on custom nuclei templates; This tool can be both passive and active depending on the use of templates.
13. Testssl - This is a free command line tool which checks a server's service on any port for the support of TLS/SSL ciphers, protocols as well as some cryptographic flaws.

CHAPTER 4: DESCRIPTION AND WORKING

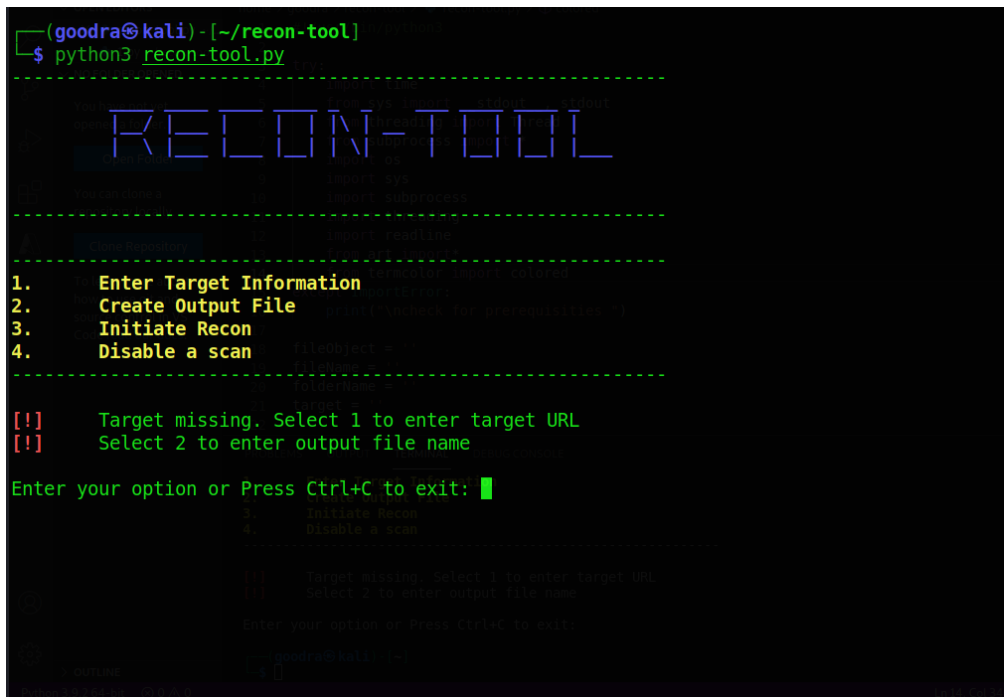
Description

Recon-tool, this framework is built with the purpose to automate the manual reconnaissance process. Throughout the years many researchers, security practitioners and experts have used the manual process to find the information about the target. After some time, tools were developed to smoothen manual processes, but this process still requires time, memory of the system to run simultaneous processes and manual work to type commands in a repeated order for attaining desired output. But nowadays, technology is growing rapidly in terms of automation.

This framework holds the same purpose of automating the desired tools into one frame and executing them simultaneously which will reduce the demerits held by the previous methodologies with some extra features.

Structure:-

- Menu driven with options as follows:
- Acquire target: input the target domain for scanning.
- Output folder : By default an output directory is set where the input name for specified folder will be created. E.g Output/<name of directory to save results>
- Initiate recon process : runs the recon process in sequential manner.



```
(goodra@kali) - [~/recon-tool]
$ python3 recon-tool.py

-----
RECON-TOOL
-----
1. Enter Target Information
2. Create Output File
3. Initiate Recon
4. Disable a scan

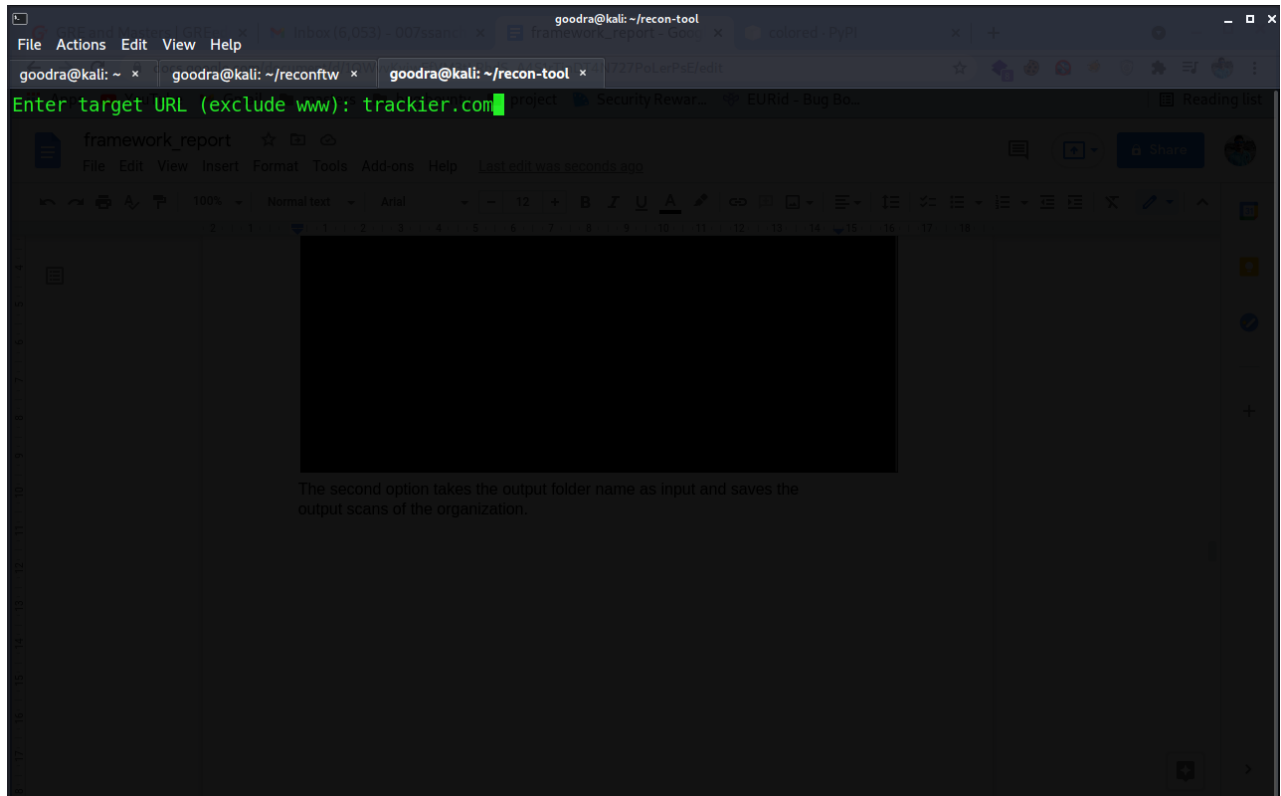
[!] Target missing. Select 1 to enter target URL
[!] Select 2 to enter output file name

Enter your option or Press Ctrl+C to exit: █
```

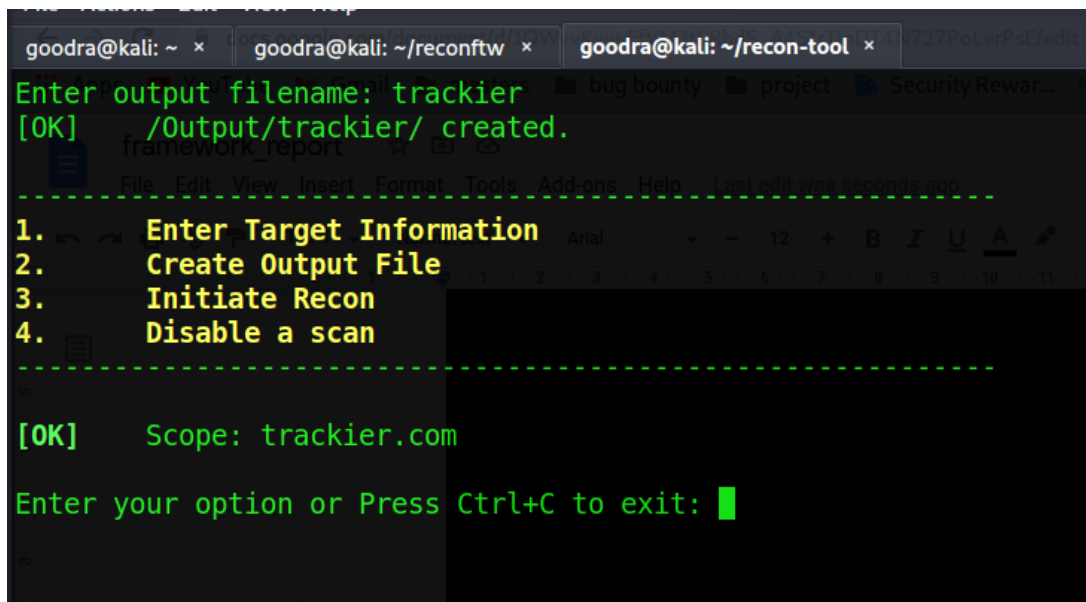
The above image is the initial screen, after turning on the recon-tool, one can see the interactive screen. It has a menu-driven and easy to understand user

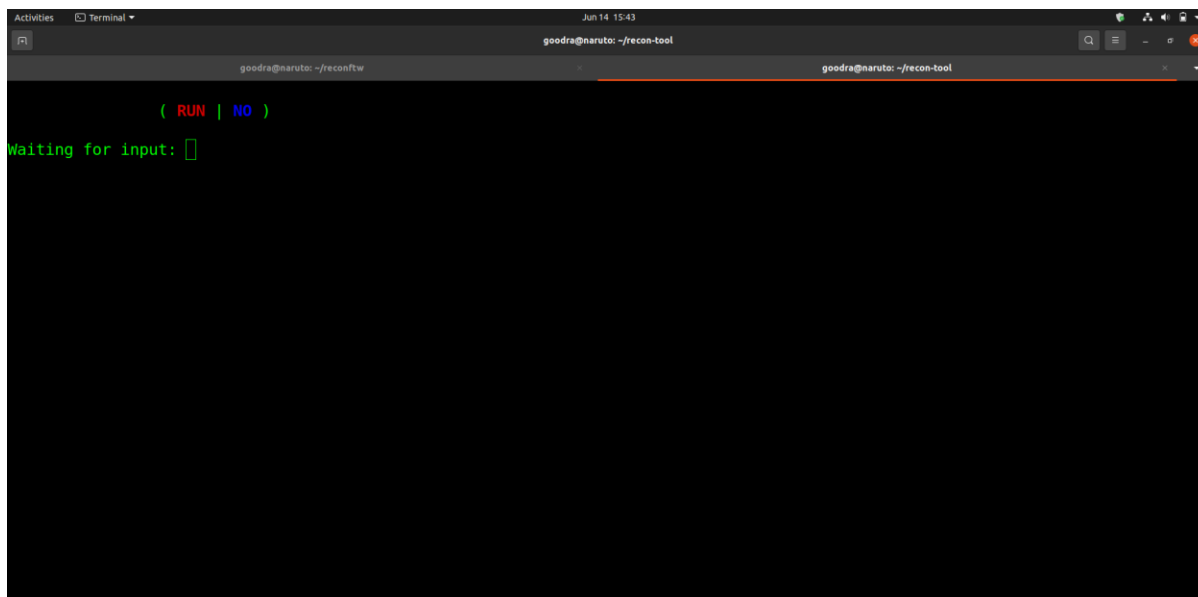
interface.

The first option takes the organization name as input without www just domain name.

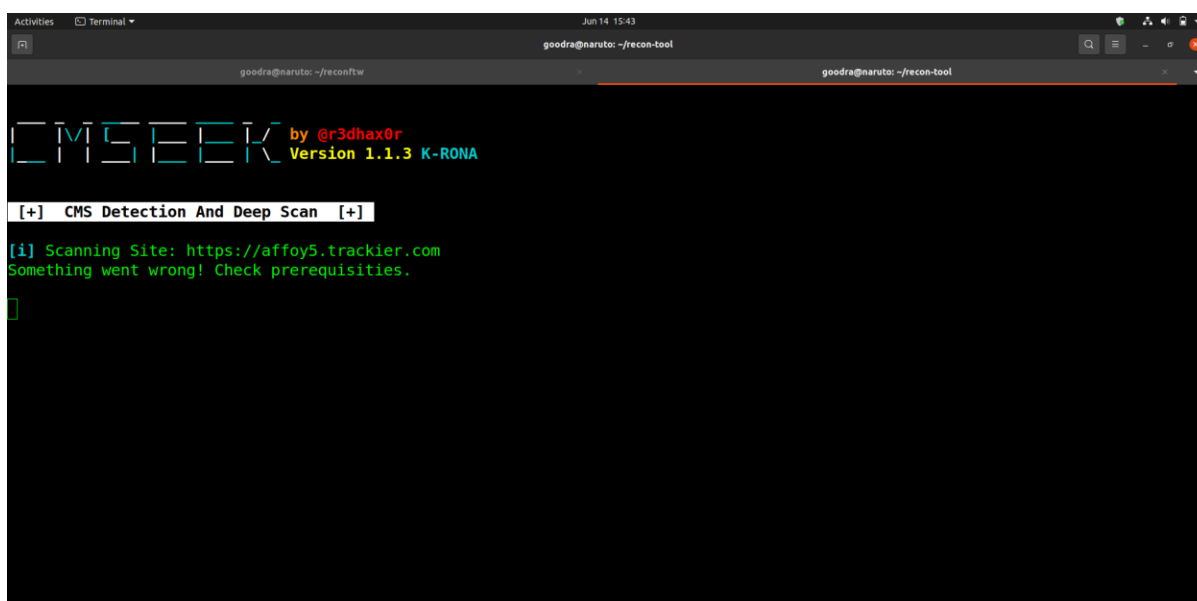


The second option takes the output folder name as input and saves the output scans of the organization.





The third option takes the command to execute all the tools that are listed in chapter 3.



The above image is the execution of the CMSeeK tool, all other tools are launched simultaneously. Only the subdomain enumeration tools launch alone at the initial stage and then all other tools use the output from enumerated and active domains.

The final output showing executed states that all the steps have been successfully executed and the output is saved in the directory that was given as input.

```

goodra@kali: ~/recon-tool
File Actions Edit View Help
goodra@kali: ~ x goodra@kali: ~/reconftw x goodra@kali: ~/recon-tool x

-----
RECON-TOOL
-----
1. Enter Target Information
2. Create Output File
3. Initiate Recon
4. Disable a scan

[!] Target missing. Select 1 to enter target URL
[!] Select 2 to enter output file name

Enter your option or Press Ctrl+C to exit: 4
0.portscan
1.emailspooft_ZT
2.nuclei_check
3.testssl
4.dorks
5.githubDorker
6.cms_scan
7.cors

-----
[!] Target missing. Select 1 to enter target URL
[!] Select 2 to enter output file name

Enter your option or Press Ctrl+C to exit:
Enter the scan number:

```

```

goodra@kali: ~/recon-tool
File Actions Edit View Help
goodra@kali: ~ x goodra@kali: ~/reconftw x goodra@kali: ~/recon-tool x

7.cors
Enter the scan number:
7
Press enter to disable another scan
or Type exit to continue scanning:
exit

These scans will be performed:
portscan
emailspooft_ZT
nuclei_check
testssl
dorks
githubDorker
cms_scan

-----
1. Enter Target Information
2. Create Output File
3. Initiate Recon
4. Disable a scan

-----
[!] Target missing. Select 1 to enter target URL
[!] Select 2 to enter output file name

Enter your option or Press Ctrl+C to exit:

```

The fourth option disables the scan which will be given as input and displays the scans which will be performed after disabling.

The working of the tool is easy to understand. For taking input, it has a separate function, then it uses if else for switching to the choices that will be given. For every scan it has one function and in that function there are terminal commands passed as list and called as system call in the terminal.

If there are multiple commands then a for loop is used for running commands one after another. The whole block of calling subprocess repeatedly, is written under try and except block so that if any error occurs it will stop the execution of that function rather than exiting the whole program and throwing an exception error. Coloured output is used in every print statement which makes it interactive.

At last, threading is used in the main option in which option for execution is given, threading combines the processes by running them simultaneously. The subdomain function is written under the if block because if the framework is being executed repeatedly then there will be no need of gathering subdomains respectively, also it will only overwrite the existing files creating more temporary memory. Therefore, the output is checked which is created by running the framework for the first time whether it is present or not. If it is present then the subdomain function will be skipped and all other remaining functions are joined by threading using thread, start & join command. In the thread.join command a timeout of 5 sec has been given so that every function starts and ends properly while executing side by side.

CHAPTER 5: Steps to Reconstruct

To reconstruct the framework the crucial thing is to import the required modules in the python source file. By following commands one can import desired modules that will be used in constructing the framework.

```
2
3  from sys import __stdout__, stdout
4
5
6  try:
7      from subprocess import *
8      import os
9      import sys
10     import subprocess
11     import threading
12     import readline
13     from termcolor import colored
14 except ImportError:
15     print("\ncheck for prerequisites ")
16
```

The first try and except block is to ensure that, even if the modules are not imported or aren't installed properly it won't exit the execution of framework instead it will print the above line (check for prerequisites). The modules are imported as shown in the above code.

The next parts are the different functions created for different vulnerability assessment activities. Every function has the same structures and methods, the only difference lies in the commands that are written for function. A function is a block of code which only runs when it is called.

Sample function code:

```
def acquireTarget():
    global target
    os.system('clear')
    target = str(input("Enter target URL (exclude www): "))
    print('\x1b[1;32m' + '\n[OK]\t' + '\x1b[0m' + 'Target Acquired')
```

```

def testssl():
    os.system('clear')
    p = './ssl_test/'
    try:
        os.mkdir(p)

    except FileExistsError as exc:
        print(exc)
        print(colored("\n-----",
            'red', attrs=['bold']))
        print(
            colored("[+] Starting SSL TEST", 'red', attrs=['bold']))
        print(colored("-----",
            'red', attrs=['bold']))
        print(colored("\n[*]\tRunning SSL TEST scans\n[*]", 'red', attrs=['bold']))
        global target
        cmd = [
            '~ /Tools/testssl.sh/testssl.sh --quiet --color 0 -U -il hosts/ips.txt | tee ssl_test/ssltest.txt'
        ]

        try:
            proc = subprocess.run(cmd, shell=True, stderr=STDOUT, stdout=PIPE)
            if(proc.returncode != 0):
                print(colored('Step 1 Failed! Check/Update prerequisite packages. \nError: ',
                    'blue', attrs=['bold']) + proc.stderr.rstrip())
                sys.exit(1)
            else:
                print(colored("[x] Executed", 'blue', attrs=['bold']))
        except:
            print("Something went wrong! Check prerequisites.\n")
            return 0

# ===== google dorking =====

```

#1 in code is acquire target function is written for taking input as target url. The input method is used there.

The #2 in sample testssl function - try and except block is to create a non-existent directory for saving the output of the scans, so that it will be easy to distinguish.

The #3 in the same function - Set target variable as global to use throughout the code and define cmd variable in which the desired linux command or command line arguments are given for the tool. Always define this variable in the list as it will be using the subprocess.run module to call this command on behalf of the system. After this, use try and except block for executing the scan given in the list of commands.

Case 1 -This example has only a single command that's why it is used by giving arguments directly. The Subprocess.run method takes a list of arguments. When the method is called, it executes the command and waits for the process to finish, returning a "CompletedProcess" object in the end.

Inside the subprocess.run command, it takes the list of commands as first argument, the environment variable to run it in shell environment is the second argument, The third and fourth arguments are for piping out the output and error code. So even if any error occurs it won't directly exit or stop the execution of other tools because it

is written inside try and except block.

Case 2 - Here the list of commands will have more than one command and every command depends on the previous command. The subprocess module takes only list objects or whole lists as arguments and to solve this problem one can use a for loop.

A for loop that is accessing a list of objects one by one (for c in cmd:). It will execute the same as case 1 only in a loop. All the functions are created using the same methods. More tools can be added using the same methods.

In the source code the main function is defined for all the functioning of the framework. Some variables are defined as global to use them in the whole code and after that they are not needed to define again. In print statements if one wishes to add more options then add the desired statement that will be printed on the screen. Then the input is taken as choices, using if condition to check the choice and if the choice is true then all the code written inside that choice will be executed.

For example:

```
global target
global fileName, folderName
print("-----")
print('\x1b[1;31m' + '\n\t\t\ RECON-TOOL /\ ' + '\x1b[0m')
print("-----")
try:
    while True:
        print("\n-----")
        print("1.\tEnter Target Information")
        print("2.\tCreate Output File")
        print("3.\tInitiate Recon")
        # print("4.\tEnable or Disable a scan")
        print("-----")
        if (target == ''):
            print('\x1b[1;31m' + '\n[!]\t' + '\x1b[0m' +
                  'Target missing. Select 1 to enter target URL')
        else:
            print('\x1b[1;32m' + '\n[OK]\t' +
                  '\x1b[0m' + 'Scope: %s' % target)

        if (fileObject == ''):
            print('\x1b[1;31m' + '[!]\t' + '\x1b[0m' +
                  'Select 2 to enter output file name')
        option = int(
            input("\nEnter your option or Press Ctrl+C to exit: "))
        if(option == 1):
            acquireTarget()

        elif(option == 3):
            if(target == '' or fileObject == ''):
                print('\x1b[1;31m' + '[!]\t' + '\x1b[0m' +
                      'Target or Filename missing!')
            else:
                os.system('clear')

                print('\n\t\t(' + '\x1b[1;31m' + ' RUN ' + '\x1b[0m' +
```

Here, if option 1 is given then the acquiretarget function runs where the target url name is taken as input. Other if blocks function the same. Choice 2 is for taking

input for saving files in a directory. Choice 3 is for executing the scans that are written in the form of functions.

In choice 3 to call the functions, threading is used to maintain simultaneous execution of scans. Python threading allows one to have different parts of the program run concurrently and can simplify the design. This means that the program will have two things happening at once. However, threading may not speed up all tasks. This is due to interactions with the GIL that essentially limit one Python thread to run at a time. Since the tasks that are given to the framework spend much of their time waiting for external events that's why these functions are good candidates for threading modules.

The target functions for starting in a thread can be taken as list without quotes and are executed in a loop

```
join_list=[]
for i,j in enumerate(scans):
    i=threading.Thread(target=j)
    i.start()
    join_list.append(i)
    # if i == len(scans):
    #     for i in range(len(scans)):
    #         i.join()
for i in join_list:
    i.join()

else:
    fileObject.close()
    sys.exit(0)
ion == 2):
system('clear')
lerName = str(input("Enter output filename: "))
ot os.path.exists("Output"):
```

In choice 4, present in the first menu, a scan method is written which disables the particular scan that has been called. The code for filtering out the scan is below.

```

elif(option == 4):
    for i in dummy:
        count=0
        for i in scans:
            print(str(count) + "." + str(i.__name__) )
            count+=1
        n=int(input(colored("\nEnter the scan number:\n", 'red', attrs=['bold'])))
        scans.pop(n)
        exit=str(input(colored("\nWant to exit type exit:", 'blue')))
        if exit == 'exit' or exit == 'EXIT':
            break
        elif exit == '':
            pass
    print(colored("\nThese scans will be performed:\n", 'yellow', attrs=['bold']))

    for j in scans:
        print(colored(str(j.__name__), 'magenta', attrs=['bold']))

```

Environment Variables:

The pre configurations that are required for this framework are as follows:

1. Install the latest golang or greater than 1.14, Python3 (latest recommended), upgrade the linux repositories after installing. Also export the environment variables to the bash_profile or bashrc file present in the home directory.
2. Installing the tools stated in the manual, report or run the install.sh file to install all the tools.
3. Install python modules through pip. Strip all the libraries present inside the go directory.

Requirements For installing all dependencies

1. Linux environment. (preferably Kali Linux or Parrot OS)
2. Ram min of 4gb
3. Stable internet connection to produce fast and accurate results. Around 5Mbps will be perfect.

For troubleshooting or finding an error:

1. Changes in the code block of the function which is throwing an error.
2. An error code block is already written in every function as a “ if ” condition. It will print some statement if something goes wrong with the tool.
3. To get any other error there's no choice but to use debugging in the framework.

CONCLUSION

In the present situation the government sector needs such automation frameworks which will reduce their time in manual testing. This report covers how the framework will be useful for finding out vulnerabilities in one's system without them being alerted. The objective of using passive methods is also covered in this report. All the well known tools that are used in this report are covered with explanations to use them. As the framework is user friendly and having been written in python it makes it easy for anyone to understand and modify the framework as per their requirement. The result of all the scans is generated automatically but should be checked manually. This concludes that this framework can be used for finding security vulnerabilities and can be used for the betterment of the country.

REFERENCES

- [1] S. Patil, A. Jangra, M. Bhale, A. Raina and P. Kulkarni, "Ethical hacking: The need for cyber security," 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), 2017, pp. 1602-1606, doi: 10.1109/ICPCSI.2017.8391982.
- [2] The OWASP Foundation, "A guide to information gathering, " **OWASP**,
https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/01-Information_Gathering/01-Conduct_Search_Engine_Discovery_Reconnaissance_for_Information_Leakage
- [3] [Sathish Kishore](#), "Reconnaissance – the Eagle’s Eye of Cyber Security," SISA infosec Blogs,

<https://www.sisainfosec.com/blogs/reconnaissance-the-eagles-eye-of-cyber-security/>
- [4] Offensive Security, "A framework included in **Kali Linux distro** and approved from **Offensive Security**," <https://tools.kali.org/information-gathering/recon-n>