Fully Flexible Pen-Testing Framework: An Approach to Automated Pen-Testing

Pulkesh Jindal[1], Atharva Gautam[2]

[3]**Dr. Hariom Upadhyay**, Computer Science and Engineering Department, ABES Engineering College, Ghaziabad, Uttar Pradesh, India

---

Abstract - *As we know such frameworks, it is basically a fully permissioned simulated cyberattack on a specific network or a web application or a particular computer system(s), which is performed with the sole purpose to evaluate the security implementation and measures taken for that Computer – System/Network/ Web-application. It should not be confused with Vulnerability assessment as vulnerability assessment itself is a part of Penetration Test. It is more of a Full-Risk-Assessment, including strengths and weaknesses of the given network/web-application under consideration(in-scope) [1]. Through this project, IE, Fully Flexible Penetrating Testing Framework, which is written in Python, we propose a different way of performing penetration tests, providing the pen-tester with a wide range of features, like the ability to select which commands to be automated, terminal-output automatically exported to a file and report generation of the pen-test performed, all these functionalities being automated with just a single run, IE, Our goal is to write a Pen-Testing Framework in python which will automate the complete Pen-test.*

Key Words: **NSA**, **NASA, PTES, TCP, SSL, HTTP, IMAP, SQL**

1. INTRODUCTION

At present, performing a penetration test is much of a hassle if a pen-tester is supposed to perform it manually. This means that pen-tester will have to manually execute all the commands of each phase of the pen-test, manually, and also would have to export all the output from terminal for the purpose of reporting. Through this research, we want to aim at solving the problem of performing the tedious task of performing a penetration test, where the pen-testes has to manually type a lot commands and extract all the results from the terminal to a separate file, for reporting and documentation purpose. Keeping in mind, that all that process should be automated while giving the user enough freedom to use their choice of tools (or methodology) and their choice of environment.

1.1 Proposed System

The main theme of this research, which makes it different than any other existing pen-testing automation frameworks, is that, firstly, this framework is written in **Python**. We chose python because it is really fast, supported by most of the Linux environments, can easily handle scripting as well as programming of this framework and because of availability of such a wide range of python scripts in cyber-sec community, it is one of the most loved language in the cyber-sec community. Also, being built in python, this framework will not only work in **KaliLinux** (Official Pen-Testing Distro), but in almost every operating system, like **Windows**, **Mac OS** or **Linux**, given that all the required tools that are chosen by the Pen-Testes, must also be installed already in that environment.

Secondly, this framework gives the pen-tester the ability to easily use commands of his/her choice, i.e., all the commands that we will be using, are **NOT HARDCODED** into the application, rather, we provide classification-wise files where the pen-tester can simply enumerate the commands of their choice[4]. They can change the implementation of any phase of the Pen-Test at any time they want in any manner they like. Hence, this automated pen testing framework will be **environment-friendly** as well as **user-friendly**. This gives our framework a Metasploit-Like structure where all the commands to be used, are already known to the framework and user just has to specify them rather than doing all the work manually.

1.2 Penetration Testing Execution Standard

A Pen-Test is a fully-authorized simulated cyber-attack, with the sol purpose to discover all the holes in the security implementations of the target (i.e., the company). According to the latest standards followed by the cyber-sec community, PTES is the current standard follow for performing Penetration Tests [5].

The Standard for Penetration Testing Execution (PTES), is composed of 7 phases:

1. Pre-Interactions with engagement
2. Intelligence Collection
3. Modelling the Hazard
4. Analyse vulnerability
5. Takeovers
6. Post-Operation
7. Reports

These 7 phases cover everything related to a pen-test, and PTES defines the guidelines for carrying out such tests,

from the initial communication and reasoning behind a pen-test, through intelligence gathering and threat modelling phases in which testers work behind the scenes to gain a better understanding of the organization tested, through recognition, exploitation.

2.1 Flow Chart

In our approach, we have adopted the following workflows for both the reconnaissance and attack phase.

Figure-1: Reconnaissance Phase Flow-Chart

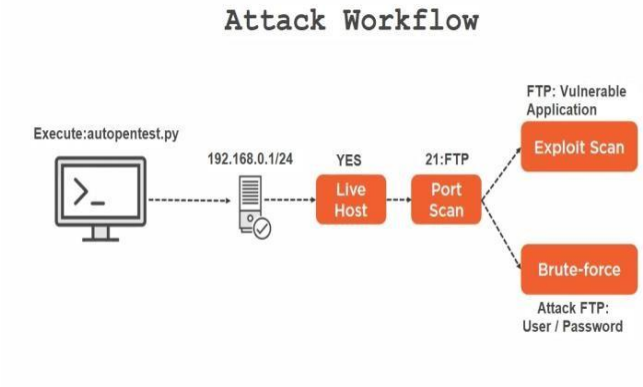

Figure-2: Attack Phase Flow-Chart

3. Use Case

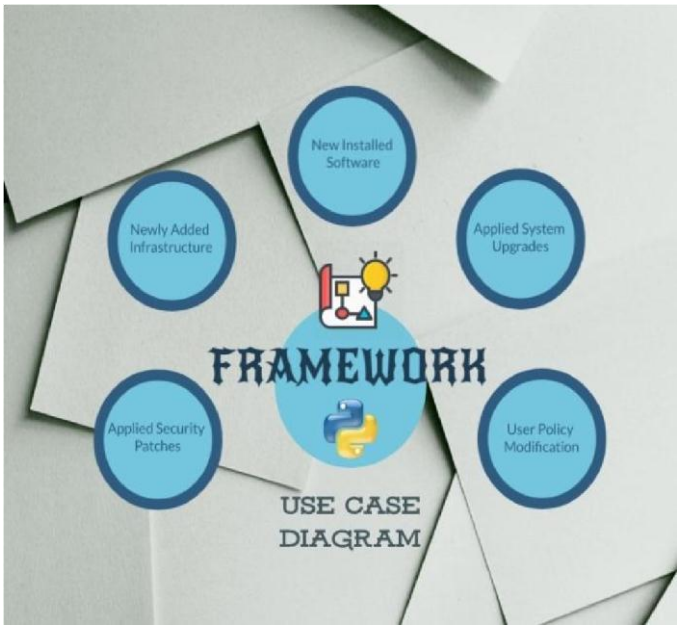

Figure-3: Use Case Diagram

4. System Modules

In this framework that we are building, we have automated our pen-test using the following tools.

Here is a flowchart for better understanding of the dependency of our framework on various tools:

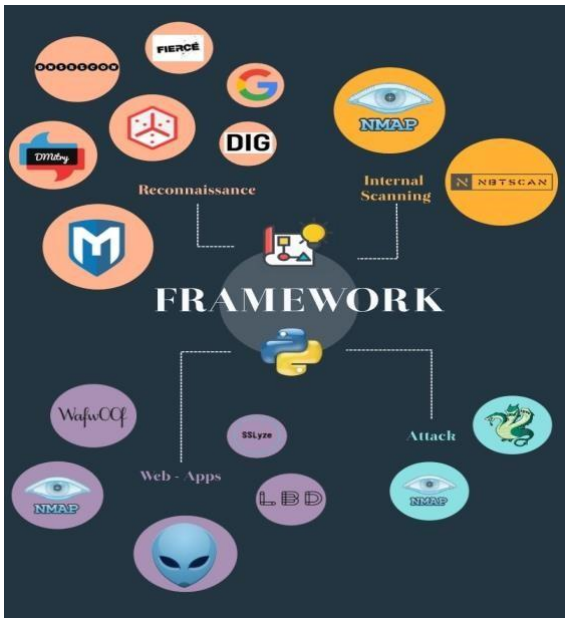

Figure-4: System Modules And Tools

4.1 NMAP

Nmap ("Network Mapper") is a free and open-source application for network exploration, network mapping as well as security-related auditing. According to network administrators, it is useful for tasks such as network

inventory, monitoring software, scheduling updates, and keeping track of uptime on host or server (just like persistent programs like 'pm2' for node). Nmap uses simple Internet-Protocol packets to determine which hosts are up in the network, what are the services those hosts provide, details of the operating systems they operate on, what kind of packet-filters / firewalls are deployed, etc.

4.2 THC-Hydra

Hydra is a simultaneous brute-forcing tool for authentication which supports numerous protocol attacks. It's very simple and adaptable, and new modules can easily be added. This approach lets researchers and safety experts show how simple it can be to gain unauthorized access to a computer remotely.

It supports: Cisco AAA, Cisco auth, Cisco enable, CVS, FTP,

HTTP(S)-FORM-GET, HTTP(S)-FORM-POST, HTTP(S)-GET,

HTTP(S)-HEAD, HTTP-Proxy, ICQ, IMAP, IRC, LDAP, MS-

SQL, MySQL, NNTP, Oracle Listener, Oracle SID, PCAnywhere, PC-NFS, POP3, PostgreSQL, RDP, Rexec, Rlogin, Rsh, SIP, SMB(NT), SMTP, SMTP Enum, SNMP v1+v2+v3, SOCKS5, SSH (v1 and v2), SSHKEY, Subversion, Teamspeak (TS2), Telnet, VMware-Auth, VNC and XMPP.

4.3 Metasploit

The Metasploit Project is an information security project that provides security vulnerabilities information and helps with penetration testing and development of IDS signatures. It is operated by Rapid7, a Cyber-Security company located in Boston, Massachusetts.

Its best established sub-project, Metasploit Platform, is an open-source resource to create and execute exploits against a specific target computer. Other significant sub-projects involve the Collection of Opcodes, the repository of shell codes and relevant work.

The Metasploit Project contains anti-forensic and evasive methods, both of which are implemented into the Metasploit System. Metasploit is pre-installed on operating system Kali Linux [11].

4.4 Wafw00f

Wafw00f is a fingerprinting framework from Python that helps you to recognize items from Web Application Firewall (WAF).When it actually connects to the web server, it is an active recognition tool, but it begins with a standard HTTP response and escalates if required. wafw00f has the following functionalities:

- Sends a standard HTTP request and the response is evaluated. Which identifies a variety of solutions for the WAF.
- If that fails, it sends out a number of (potentially malicious) HTTP requests and uses basic logic to predict which WAF it is
- If this is also not efficient, it analyses the previously returned responses and uses another basic algorithm to guess if a WAF or protection solution is reacting actively to our attacks

4.5 Nikto

Nikto is a free command line vulnerability scanner checking webservers for unsafe files / CGIs, obsolete server applications and other issues. It collects and prints any acquired cookies. It also conducts different tests for the generic and server category. Nikto itself is an open source tool but its libraries, which it requires to work with, are not free.

Nikto can detect over 6700 potentially harmful files, scans for obsolete/legacy versions of over 1250 servers and can identify version-specific issues on more than 270 servers. It also tests the existence of several index files and HTTP service choices for site setup objects, and will attempt to classify enabled web servers and applications.

4.6 Fierce

What's not Fierce, at first. Fierce isn't just a simple internetprotocol scanner, neither it's intended to search the Internet in its entirety or execute some untargeted assault. It is specifically intended to identify probable targets from both in-and-out a corporate network. This only discusses those goals (unless we use the -nopattern option). There is no abuse (unless we deliberately use -connect switch). Fierce is built in PERL, a script that uses multiple tactics to search domains quickly.

Fierce is a moderately lightweight scanner, which helps locate non-contiguous IP space and host names against specified domains. It's just realized as a nmap, unicornscan, nessus, nikto, etc. predecessor, because all these require you to know what IP space you 're already searching for. This won't hack because the whole Website isn't searched indiscriminately. This is designed primarily to identify possible targets both in-and-out of a corporate network. Since it uses Domain-Name-Service mainly, we can sometimes encounter networks that are configured improperly and leak the internal address space. It is particularly useful for targeted malware.

4.7 DMitry

DMitry (Deep Magic Information Gathering Tool) is a

UNIX/(GNU) Linux command-line program encoded in CLanguage. DMitry is capable to accumulate as much information as possible on a target. It's base-modules will capture potential subdomains, email addresses, uptime details, port search tcp, who is searches, and more.

A list of current features is set out below:

- An open source project.
- Does a who is lookup.
- Collects uptime for the data, system, and server.
- Scans the designated sub-domain.
- Search email address for a target host.
- On the target server run the TCP Port-scan.
- Modular software permitting the user to define modules

4.8 SSLyze

SSLyze is a Python application, which can evaluate the SSL configuration of a server by logging in. It's built to be fast and rigorous, and will help organizations and testers detect errors and bugs impacting their servers with SecuredSocket-Layer.

Main attributes of this tool, are:

- Multi-processed, (fast) multi-thread scanning;
- Compliance for SSL 2.0/3.0, and TLS 1.0/1.1/1.2
- Efficiency testing: restarting sessions and enabling TLS tickets
- Protection tests: poor chip sets, unstable renegotiation, Fraud, Heartbleed etc.
- Authentication and revocation of Application Credentials utilizing OCSP stapling
- Handshakes the SMTP, XMPP, LDAP, POP, IMAP, RDP and FTP service for starting TLS
- Supporting client certificates which authenticate while searching servers
- XML output to search reports for further process **the*H*arvester**

4.9 DNSRecon

DNSRecon provides the ability to perform:

- Check all records on NS Zone Transfers
- Enumerate Specific DNS records (MX, SOA, NS, A, AAAA, SPF, and TXT) for a specified namespace.
- Execute prevailing SRV data enumeration.

Top Level Domain Extend (TLD) Check Resolution on Wildcard • Bruteforces A and AAAA subdomains and host-records, provided with a domain and wordlist

- Search for a given PTR Record IP Range or CIDR
- Check the DNS registry Cached records for A, AAAA and CNAME records supported with a set of host records to be reviewed in a text file
- List can mDNS records over the Local Network Using Google to view host and subdomains

4.10 Nbtscan-unixwiz

This is a command line application that searches for free netbios-nameservers over a local/remote -TCP/IP network and is the first step towards finding available shares. It's based on the basic features of nbtstat-windows-application, which operates over a variety of addresses rather than only one address.

4.11 lbd

Load balance detection tool, that senses whether a target domain provided, employs a HTTP and/or DNS load balancer, or not (From the responses from server and differentiating on the basis of same through the 'Server:' and 'Date:' header in).

ACKNOWLEDGEMENT

We would like to present this research paper as a part of project-work done for our B.

Tech final year. We would like to thank our HOD, Prof. Pankaj Kumar Sharma, of our Computer Science Department at ABESEC, for his continuous guidance and support in our journey of building a flexible automated penetration testing framework.

At last but not the least, we would like to pay our gratitude towards the team of management, my colleagues and the whole staff of the ABESEC to help us during difficult times.

It is a really special feeling to mention all of them and for any name that skipped my mind during writing this, Gratitude is due.

5. CONCLUSIONS

This framework is built keeping in mind that it is not an automated application which would make a pen-test successful in its goal. It is purely dependent on the depth of knowledge and understanding of the individual (or group of individuals) who perform the pen-test, because the better they understand the risks of their supposed target

in a simulation, the better they are able to mitigate those risks. Hence, It is very important for an automation pen testing framework to give enough flexibility to its user so that they can choose what they want to automate, rather than how they want to automate the pen-test.

REFERENCES

[1] https://www.pgitl.com/blog/whats-thedifference-between-a-vulnerabilityassessmentand-a-penetration-test/

[2] https://pentestlab.blog [3] https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20140002617.pdf

[4] https://www.doi.gov/ocio/customers/penetration-testing

[5] http://www.pentest-standard.org/index.php