BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2023
http://excel.fit.vutbr.cz

Tool for Automated Testing of Web Servers

Bc. Michal Rajecký*

Abstract
This thesis delves into the topic of cybersecurity, with an emphasis on the security of web servers. It covers the technologies that are employed in order to protect web servers from variety of common security threats. Additionally, the thesis explores multiple aspects of penetration testing, including stages of the process and the attacker kill chain. In the implementation part of this thesis, an automated tool for testing web servers is developed, focusing on expandability, configurability, maintainability, and user-friendliness. It is designed for the reconnaissance phase of the penetration testing process, in order to streamline the work of testers.

*xrajec01@fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

In the initial stage of penetration testing, known as the reconnaissance phase, the primary objective is to collect comprehensive information about the targeted system or network. The information gathering process consists of identifying potential targets, exploring software versions, open ports, web server security headers or detection of known vulnerabilities, among other relevant data.[1] The basic concept of the first stage is best described by the adage: "Reconnaissance time is never wasted time," well known among many military organizations, which underscores the critical importance of gathering as much information as possible before launching an attack. It is not unusual that penetration testers devote over 70 percent of their overall effort to the reconnaissance phase.[2] Consequently, any optimization or simplification of this phase can significantly enhance the efficiency and time effectiveness of penetration testing.

The aim of this master thesis is to develop a tool for automated testing of web servers that integrates the functionality of multiple Linux tools. The main motivation is to simplify the often time-consuming process of information gathering and identifying potential security issues during the reconnaissance phase of penetration testing, thereby improving the efficiency of testers.

2. Design and Implementation

While other existing solutions, such as Nessus or OpenVAS are available to address similar issues, they have several limitations. For example, higher cost of Nessus can make it less suitable option for smaller organization while also having a steep learning curve, making it challenging for new users to start with it.[3] Similarly, OpenVAS has some considerable limitations in its reporting capabilities and is known for its complexity, which can make it harder to configure and set up without a cretain level of technical skills.[4]

Based on the goals of the tool, the following functional requirements have been selected:

Host discovery & port scanning. The tool can evaluate whether the server is up, perform host discovery, and identify open ports and running services to help testers to identify potential points of interest.
Custom word lists. Ability to generate custom word lists, which can be used in dictionary attacks on folders,directories or user accounts.
SSL/TLS configurations. Analysis of the security headers and SSL/TLS configurations of servers.
Vulnerability scan. Provides the user with vulnerability scanning option.
DNS lookup. The tool provides the user with reverse DNS lookup option.
Directory enumeration. Recursive dictionary attack to enumerate the folders and files.

Regarding the chosen tools for fulfilling the functional requirements, the following have been selected: `Nmap`[1] for host discovery and port scanning, `CeWL`[2] for custom word list creation, both `Shcheck`[3] and `SSLScan`[4] in combination with the `Nmap ssl-enum-ciphers` script for SSL/TLS analysis, `Nikto`[5] for vulnerability scanning, `DNSenum`[6] for DNS enumeration, and finally, `Gobuster`[7] for directory enumeration.

The non-functional requirements include the **ability to configure parameters for the program's execution**, **simple usage** and **extensibility**.

The technologies chosen for this project are **Docker**[8], which allows for easy containerization of individual tools, and the high-level platform-independent programming language **Python**, which has a wide range of libraries suitable for a work in the field of cybersecurity.

One of the most significant advantages of this new tool is its highly configurable testing parameters (done through several configuration files), which allow users to customize each tool's switches and setting to their specific needs. It also provides extensibility, which enables users to add either a new combination of parameters to an already integrated tool, or a completely new tool. Another key advantage is that each integrated tool is actively developed and maintained, ensuring that they receive regular updates and patches. Furthermore, it isolates each individual tool in a separate docker container, which minimizes possibility of compatibility issues. Also, in case of an event when, due to an update or for any other reason, a given tool ceases to be functional or compatible with the parser, only that one part will stop working and the rest of the program will remain unaffected.

The tool is user-friendly, and the installation process is straightforward. Potential dependency issues are minimized thanks to the base building block, which is continually maintained Ubuntu docker image from Docker Hub.

The tool is also easy to install and configure, where the individual tools are based on the base docker image - Ubuntu. This greatly reduces potential dependency issues during installation, since the building block is free to download from docker-hub and is continually maintained.

[1] Nmap - https://nmap.org/
[2] CeWL - https://www.kali.org/tools/cewl/
[3] Shcheck.py - https://pypi.org/project/shcheck/
[4] SSLScan - https://www.kali.org/tools/sslscan/
[5] Nikto scanner - https://www.cirt.net/Nikto2
[6] DNSenum - https://www.kali.org/tools/dnsenum/
[7] Gobuster - https://www.kali.org/tools/gobuster/
[8] Docker - https://www.docker.com/

3. Conclusions

The practical part of this thesis focuses on designing and implementing a tool for automated penetration testing. The aim is to expand the repertoire of tools available to penetration testers, in order to simplify their work during the reconnaissance phase of the process. The tool is currently under development and ongoing testing, with the end goal of being ready for deployment in a real-world environment. Possible future improvements include adding the ability to configure a proxy, building a graphical user interface, as well as adding additional parameters or tools to enhance its functionality.

References

[1] J. Erickson. *Hacking: The Art of Exploitation, 2nd Edition*. No Starch Press Series. No Starch Press, 2008.

[2] V.K. Velu and R. Beggs. *Mastering Kali Linux for Advanced Penetration Testing: Secure Your Network with Kali Linux 2019. 1 - the Ultimate White Hat Hackers' Toolkit, 3rd Edition*. Packt Publishing, 2019.

[3] H. Kumar. *Learning Nessus for Penetration Testing*. Community experience distilled. Packt Publishing, 2014.

[4] Sagar Rahalkar. *Quick Start Guide to Penetration Testing: With NMAP, OpenVAS and Metasploit*. Apress, USA, 1st edition, 2018.