International Journal in IT & Engineering (IJITE)
Volume 9 Issue 5, May 2021 ISSN: 2321-1776 Impact Factor: 7.256
Journal Homepage: http://ijmr.net.in, Email: irjmss@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal

WAPT-LABS - A LAB ENVIRONMENT FOR LEARNING WEB APPLICATION PENETRATION TESTING

Anil Tom
MCA Student, Jain deemed-to-be University, Karnataka, India
Dr. M N Nachappa
Head of School of Computer Science & IT, Jain deemed-to-be University, Karnataka, India

ABSTRACT

Web Application Penetration Testing involves a methodological series of steps aimed at gathering information about the target system, finding vulnerabilities or faults in them, researching for exploits that will succeed against those faults or vulnerabilities and compromise the web application. The Open Web Application Security project (OWASP) is a community that focuses its efforts solely on discovering and reporting on web application security vulnerabilities. Their reputable list of top 10 security flaws is updated every four years to reflect the global trends in the security of web applications. OWASP also publishes articles and security tools that are most practiced.

The WAPT-LABS have more than 6 vulnerable applications running in docker containers which mainly focus the OWASP Top 10 and vulnerabilities. Which help people to learn more about Web Application Penetration Testing and how to identify the vulnerabilities in the web applications? Setting up the lab is an easy task so that everyone can learn. Once completed all the labs will have some real world challenges so that the students can check their knowledge.

INTRODUCTION

1.1 What is Penetration Testing?

Penetration testing (or pen testing) is a security exercise where a cyber-security expert attempts to find and exploit vulnerabilities in a computer system. The purpose of this simulated attack is to identify any weak spots in a system's defenses which attackers could take advantage of. These vulnerabilities may exist in operating systems, services and application flaws, improper configurations or risky end-user behavior. Such assessments are also useful in validating the efficacy of defensive mechanisms, as well as end-user adherence to security policies.

Penetration testing is typically performed using manual or automated technologies to systematically compromise servers, endpoints, web applications, wireless networks, network devices, mobile devices and other potential points of exposure. Once vulnerabilities are successfully exploited on a particular system, testers may attempt to use the compromised system to launch subsequent exploits at other internal resources, specifically by trying to

incrementally achieve higher levels of security clearance and deeper access to electronic assets and information via privilege escalation.

Information about any security vulnerabilities successfully exploited through penetration testing is typically aggregated and presented to IT and network system managers to help those professionals make strategic conclusions and prioritize related remediation efforts. The fundamental purpose of penetration testing is to measure the feasibility of systems or end-user compromise and evaluate any related consequences such incidents may have on the involved resources or operations.

1.2 Different types of Penetration testing

1. Targeted testing
2. External testing
3. Internal testing
4. Blind testing
5. Double-blind testing
6. Black box testing
7. White box testing
8. Grey box testing

1.3 Penetration Testing Phases

1. Reconnaissance
2. Scanning and Enumeration
3. Gaining Access
4. Maintaining Access
5. Covering Tracks And Report

Figure: 1 Phases of Penetration Testing

- **Reconnaissance**

The first stage involves gathering relevant information on a target system (how a target system works and its potential vulnerabilities). This information can be used to define the scope and goals of testing, and the testing methods to be used.

- **Scanning and Enumeration**

The next step is understanding the target application. How it responds to various intrusion. This scanning is done in two ways

- Static analysis
- Dynamic analysis

- **Gaining access**

Gaining access from the knowledge base collected earlier in reconnaissance and scanning phases helps the tester to intrude the targeted system and uncover target's vulnerabilities. This stage uses web application attacks, such as cross-site scripting, SQL injection, and back doors. Then the tester tries to damage the system to understand & measure the damage that a hacker can cause to the system.

- **Maintaining Access**

In this stage, the goal is to persistently stay within the target environment to gather as much data as possible. The purpose of this phase is to simulate advanced persistent threats that stays up in a system undetected for months to steal an organization's most sensitive data.

- **Covering Tracks and Report**

This phase is the final phase here the results of the penetration test were compiled into a report including

- Vulnerabilities of the target system.
- Data accessed and damage caused
- Amount of time a hacker could persist in system undetected.

1.4 Web Application Penetration Testing

The continued growth of internet usage as well as the development of Web applications foreground the challenges of IT security, particularly in terms of data confidentiality, data integrity and service availability. Thus, as stated in the annual barometer concerns of Information Technology Managers, for 72% of them, computer security and data protection are their primary concerns. This growth of risk arises from the mozaic of technologies used in current Web applications (e.g. HTML5), which increases the risk of security breaches. This situation has led to significant growth in application-level vulnerabilities, with thousands of vulnerabilities detected and disclosed annually in public databases such as the MITRE CVE - Common Vulnerabilities and Exposures. Web Application Penetration Testing is the process of using penetration testing techniques on a web application to detect its vulnerabilities. It is similar to a penetration test and aims to break into the web application using any penetration attacks or threats. Web Application Penetration Testing works by using manual or automated penetration tests to identify any vulnerability, security flaws or threats in a web application. The tests involve using/implementing any of the known malicious penetration attacks on the application. The penetration tester exhibits/fabricates attacks and environment from an attacker's perspective, such as using SQL injection tests. The Web Application Penetration Testing key outcome is to identify security weakness across the entire web application and its components (source code, database, back-end network). It also helps in prioritizing the identified vulnerabilities and threats, and possible ways to mitigate them.

The most common vulnerabilities found on these databases especially emphasize the lack of resistance to code injection of the kind SQL Injection or Cross-Site Scripting (XSS), which have many variants. They appear in the top list of current web applications attacks. Application-level vulnerability testing is first performed by developers, but they often lack the sufficient in-depth knowledge in recent vulnerabilities and related exploits. This kind of tests can also be achieved by companies specialized in security testing, in pen-testing (for penetration testing) as instance. These companies monitor the constant discovery of such vulnerabilities, as well as the constant evolution of attack techniques. But they mainly use manual approaches, making the dissemination of their techniques very difficult, and the impact of this knowledge very low.

To learn Web Application Penetration Testing there are many labs out there in the internet but collecting, installing and configuring the labs is a bit complicated things. For that we need to install other applications like XAMPP or WAMPP or other tools and softwares. Also configuring them is bit difficult for the bingers, as a solution for that we have created WAPT-LABS. Which have more than 6 vulnerable applications running in docker containers and also along with there is some challenges. So once a student finished all the vulnerable applications he/she can check their competency by solving the challenges. The challenges are based on real world vulnerabilities and exploits.

2. OBJECTIVE

The objective of this project is to help people to learn more about Web Application Penetration Testing using vulnerable applications and challenges, which further helps them with real time systems. It will help people who are still beginning their careers.

3. SYSTEM ANALYSIS

3.1 Existing System:

At present world, we have some vulnerable applications like DVWA, BWAPP, etc. These application are written in PHP and other languages and we need XAMPP or WAMPP for running the application.

3.2 Disadvantages of Existing System:

- The applications have to be downloaded one by one.
- After downloading the vulnerable application we have to install another software to run it.
- Configuring vulnerable application is also bit difficult for the beginner.

3.3 Proposed System:

In the proposed WAPT LAB all the vulnerable applications are running in the docker. By running the command it will automatically download all the vulnerable applications docker images and all the tools from the internet which is necessary for doing the lab and completing the challenges. And it will automatically configure and the vulnerable applications will list in the different ports.

3.4 Advantages of Proposed System:

- Easy to install
- Easy to configure.
- Multiple vulnerable applications running at the same time.

- Several real world web application challenges for practicing.
- No need of software's like XAMPP or WAMPP.

3.5 Software Environment

- **Docker**

Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels. Docker is an open source containerization platform. Docker enables developers to package applications into containers-standardized executable components that combine application source code with all the operating system (OS) libraries and dependencies required to run the code in any environment. While developers can create containers without Docker, Docker makes it easier, simpler, and safer to build, deploy, and manage containers. It's essentially a toolkit that enables developers to build, deploy, run, update, and stop containers using simple commands and work-saving automation.

- **Damn Vulnerable Web Application**

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment. The aim of DVWA is to practice some of the most common web vulnerabilities, with various levels of difficulty, with a simple straightforward interface.

- **OWASP Juice Shop**

OWASP Juice Shop is probably the most modern and sophisticated insecure web application. It can be used in security trainings, awareness demos, CTFs and as a guinea pig for security tools. Juice Shop encompasses vulnerabilities from the entire OWASP Top Ten along with many other security flaws found in real-world applications. Juice Shop is written in Node.js, Express and Angular. It was the first application written entirely in JavaScript listed in the OWASP VWA Directory. The application contains a vast number of hacking challenges of varying difficulty where the user is supposed to exploit the underlying vulnerabilities. The hacking progress is tracked on a score board. Finding this score board is actually one of the (easy) challenges.

- **bWAPP**

bWAPP, or a buggy web application, is a deliberately insecure web application. bWAPP helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities. It prepares one to conduct successful penetration testing and ethical hacking projects. What makes bWAPP so unique? Well, it has over 100 web bugs! bWAPP covers all major known web vulnerabilities, including all risks from the OWASP Top 10 project! It is for security-testing and educational purposes only.

- **OWASP Mutillidae II**

OWASP Mutillidae II is a free, open source, deliberately vulnerable web-application providing a target for web-security enthusiast. Mutillidae can be installed on Linux and Windows using LAMP, WAMP, and XAMMP. It is pre-installed on SamuraiWTF and OWASP BWA. The existing version can be updated on these platforms. With dozens of vulnerabilities and hints to help the user; this is an easy-to- use web hacking environment designed for labs, security enthusiast, classrooms, CTF, and vulnerability assessment tool targets. Mutillidae has been used in graduate security courses, corporate web sec training courses, and as an “assess the assessor” target for vulnerability assessment software.

- **Damn Vulnerable NodeJS Application**

DVNA is an intentionally vulnerable web application written in NodeJS. It can be used in learning to identify, attack and most importantly fix OWASP Top 10 vulnerabilities in NodeJS. Damn Vulnerable NodeJS Application (DVNA) is a simple NodeJS application to demonstrate OWASP Top 10 Vulnerabilities and guide on fixing and avoiding these vulnerabilities. The fixes branch will contain fixes for the vulnerabilities. Fixes for vulnerabilities OWASP Top 10 2017 vulnerabilities at fixes-2017 branch. The application is powered by commonly used libraries such as express, passport, etc.

- **OWASP WebGoat**

WebGoat is a deliberately insecure application that allows interested developers just like you to test vulnerabilities commonly found in Java-based applications that use common and popular open source components. Web application security is difficult to learn and practice. Not many people have full blown web applications like online book stores or online banks that can be used to scan for vulnerabilities. In addition, security professionals frequently need to test tools against a platform known to be vulnerable to ensure that they perform as advertised. All of this needs to happen in a safe and legal environment. Even if your intentions are good, we believe you should never attempt to find vulnerabilities without permission. The primary goal of the WebGoat project is simple: create a de- facto interactive teaching environment for web application security. In the future, the project team hopes to

extend WebGoat into becoming a security benchmarking platform and a Java-based Web site Honeypot.

- **Hackazon**

Hackazon is a free, vulnerable test site that is an online storefront built with the same technologies used in today's rich client and mobile applications. Hackazon has an AJAX interface, strict workflows and RESTful API's used by a companion mobile app providing uniquely-effective training and testing ground for IT security professionals. And, it's full of your favourite vulnerabilities like SQL Injection, cross-site scripting and so on. Today's web and mobile applications as well as web services have a host of new technologies that are not being adequately tested for security vulnerabilities. It is critical for IT security professionals to have a vulnerable web application to use for testing the effectiveness of their tools and for honing their skills. Hackazon enables users to configure each area of the application in order to change the vulnerability landscape to prevent "known vuln testing" or any other form of 'cheating. 'Since the application includes RESTful interfaces that power AJAX functionality and mobile clients (JSON, XML, GwT, and AMF), users will need to the latest application security testing tools and techniques to discover all the vulnerabilities. Hackazon also requires detailed testing of strict workflows, like shopping carts, that are commonly used in business applications.

4. SYSTEM STUDY

4.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

- **Economical Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and

development of the system is limited. The expenditures must be justified. Thus the developed system is within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

- **Technical Feasibility**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. Which will in turn lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

- **Social Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

5. IMPLEMENTATION

- **Create Docker images for Vulnerable Applications**

In the primary module, we build up each vulnerable application into docker images. Once we created the docker images then we can push it to the docker hub. Once we upload it to the docker hub, when we uploading to docker hub make sure we upload it as a public docker image. If it is not a public docker image it cannot download by any other person other than the docker hub owner. For creating docker images first clone the vulnerable application from the GitHub. After downloading the project, create the Dockerfile. Dockerfiles describe how to assemble a private filesystem for a container, and can also contain some metadata describing how to run a container based on this image. After creating Dockerfile build the application, after building the docker push it to the Docker Hub.

- **Create Challenges Docker image**

In this module, we will create vulnerable web application for practicing. In the WAPT LAB V1 we have two challenges. Both challenges are written in PHP. After creating the challenges we have to convert it to docker image and host it on dockerhub.

- **Automated script for download and configuring the applications**

After creating the docker images of vulnerable applications and challenges the next step is to create a bash script to auto install and configure the vulnerable application, challenges and tools.

6. CONCLUSION

WAPT-Labs - A Lab Environment for Learning Web Application Penetration Testing is a novel security structure that helps beginners to learn Web Application Penetration Testing. The lab setup is easy to install and configure so that everyone can learn the Web Application Penetration Testing. It helps a beginner to learn about Web Application Penetration Testing.

7. FUTURE ENHANCEMENT

In the next version of WAPT-Labs will include more web challenges where the students can practice and learn more about web application penetration testing.

8. REFERENCES

[1] Gajanan Shinje and Sanjhya S. Waghere, "Analysis of SQL Injection Using DVWA Tool" The Second International Conference on Research in Intelligent and Computing in Engineering, DOI: 10.15439/2017R66
[2] Franck Lebeau, Bruno Legeard, Fabien Peureux and Alexandre Vernotte, "Model-Based Vulnerability Testing for Web Applications" SECTEST'13, 4-th Int. Workshop on Security Testing. In conjunctionwith ICST'13, 6-th IEEE Int. Conf. on Software Testing, Verification and Validation, Jan 2013
[3] A. Dias-Neto and G. Travassos, "A Picture from the Model-BasedTesting Area: Concepts, Techniques, and Challenges,"Advances inComputers, vol. 80, pp. 45–120, July 2010, iSSN: 0065-2458.
[4] Renelada Kushe, "Security Assessment of Web Applications" (2017).UBT International Conference. 189.
[5] https://github.com/digininja/DVWA
[6] https://owasp.org/www-project-juice-shop/
[7] https://github.com/jehy-security/bwapp
[8] https://github.com/webpwnized/mutillidae
[9] https://github.com/appsecco/dvna
[10] https://owasp.org/www-project-webgoat/
[11] https://github.com/rapid7/hackazon