Web App Pen testing on Metasploitable 2

1. Project Overview

This project details my process of conducting a basic web application penetration test against the Metasploitable 2 virtual machine (IP: 192.168.1.80). My goal was to identify common web vulnerabilities using a combination of automated scanners and manual testing.

Tools Used: Nmap, Nikto, Burpsuite, Dirb and Sqlmap

2. Initial Setup And Reconnaissance

First, I ensured both my Metasploitable 2 and Kali Linux VMs were running and could communicate. I found Metasploitable 2's IP address to be 192.168.1.80.

I then ran a quick Nmap scan to see what services were running:

Command: nmap -sV 192.168.1.80

```
┌──(paxton㉿kali)-[~]
└─$ nmap -sV 192.168.1.80
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-30 13:14 +0545
Nmap scan report for 192.168.1.80
Host is up (0.000087s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login       OpenBSD or Solaris rlogind
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:55:6E:B5 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CP
E: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.60 seconds
```

Figure 1: Nmap showing open ports and services on Metasploitable 2

This confirmed that web services (Apache on port 80, Tomcat on 8180) were active, which would be my focus.

3. Tool Specific Findings

i. Nikto Scan

- **What it does:** Nikto is a web server scanner that checks for known vulnerabilities, outdated software, and common misconfigurations.
- **Command Used:** nikto -h http://192.168.1.80/
- **What I Found:**
  - ➔ The anti-clickjacking X-Frame-Options header is not present.
  - ➔ The X-Content-Type-Options header is not set.
  - ➔ Apache/2.2.8 appears to be outdated
  - ➔ HTTP TRACE method is active which suggests the host is vulnerable to XST.

- **Proof/Screenshot:**

```
---------------------------------------------------------------------------
+ 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. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.co
m/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index' were found: index.php. See: htt
p://www.wisec.it/sectou.php?id=4698ebdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross_Site_Tracing
+ /phpinfo.php: Output from the phpinfo() function was found.
+ /doc/: Directory indexing found.
+ /doc/: The /doc/ directory is browsable. This may be /usr/doc. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0678
+ /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /phpMyAdmin/changelog.php: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ /phpMyAdmin/ChangeLog: Server may leak inodes via ETags, header found with file /phpMyAdmin/ChangeLog, inode: 92462, size: 40540, mtime: Tue Dec  9 23:09:00 2008. See: http://cve.mitre.o
rg/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ /phpMyAdmin/ChangeLog: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ /test/: Directory indexing found.
+ /test/: This might be interesting.
+ /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information. See: CWE-552
+ /icons/: Directory indexing found.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /phpMyAdmin/: phpMyAdmin directory found.
+ /phpMyAdmin/Documentation.html: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ /phpMyAdmin/README: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts. See: https://typo3.org/
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8910 requests: 0 error(s) and 27 item(s) reported on remote host
+ End Time:           2025-05-30 13:41:55 (GMT5.75) (13 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figure 2: Nikto scan results

- **Impact:**
  - ➔ Enables Clickjacking attacks, allowing attackers to trick users into performing unintended actions (e.g., unauthorized clicks, data submission) by embedding the site in a malicious frame.
  - ➔ Allows MIME-sniffing attacks, which could lead to Cross-Site Scripting (XSS) or arbitrary code execution by tricking browsers into misinterpreting content types.
  - ➔ Susceptible to numerous known vulnerabilities, including potential for denial-of-service, information disclosure, directory traversal, and remote code execution, severely compromising server security.
  - ➔ Vulnerable to Cross-Site Tracing (XST), which can be combined with XSS to bypass HTTPOnly protection and steal sensitive session cookies, leading to session hijacking and unauthorized access.

- **Recommendations:**
  - ➔ Implement X-Frame-Options and X-Content-Type-Options HTTP headers, update the outdated Apache server to the latest version, and disable the active HTTP TRACE method.

ii. Dirb Scan

- **What it does:** Dirb is a web content scanner that tries to find hidden directories and files on a web server using a wordlist.
- **Command Used:** dirb http://192.168.1.80
- **What I Found:**
  - ➔ Discovered common directories like /dav, /phpMyAdmin, /test, /twiki and /phpinfo and the subdirectories inside them.

- **Proof/Screenshot:**

```
┌──(paxton㉿kali)-[~]
└─$ dirb http://192.168.1.80

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Fri May 30 15:26:11 2025
URL_BASE: http://192.168.1.80/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612
```

Figure 3: dirb scan info

```
---- Scanning URL: http://192.168.1.80/ ----
+ http://192.168.1.80/cgi-bin/ (CODE:403|SIZE:293)
==> DIRECTORY: http://192.168.1.80/dav/
+ http://192.168.1.80/index (CODE:200|SIZE:891)
+ http://192.168.1.80/index.php (CODE:200|SIZE:891)
+ http://192.168.1.80/phpinfo (CODE:200|SIZE:48062)
+ http://192.168.1.80/phpinfo.php (CODE:200|SIZE:48074)
==> DIRECTORY: http://192.168.1.80/phpMyAdmin/
+ http://192.168.1.80/server-status (CODE:403|SIZE:298)
==> DIRECTORY: http://192.168.1.80/test/
==> DIRECTORY: http://192.168.1.80/twiki/
```

Figure 4: dirb scan result

```
---- Entering directory: http://192.168.1.80/twiki/ ----
==> DIRECTORY: http://192.168.1.80/twiki/bin/
+ http://192.168.1.80/twiki/data (CODE:403|SIZE:295)
+ http://192.168.1.80/twiki/index (CODE:200|SIZE:782)
+ http://192.168.1.80/twiki/index.html (CODE:200|SIZE:782)
==> DIRECTORY: http://192.168.1.80/twiki/lib/
+ http://192.168.1.80/twiki/license (CODE:200|SIZE:19440)
==> DIRECTORY: http://192.168.1.80/twiki/pub/
+ http://192.168.1.80/twiki/readme (CODE:200|SIZE:4334)
+ http://192.168.1.80/twiki/templates (CODE:403|SIZE:300)
```

Figure 5: dirb scan result

```
---- Entering directory: http://192.168.1.80/phpMyAdmin/ ----
+ http://192.168.1.80/phpMyAdmin/calendar (CODE:200|SIZE:4145)
+ http://192.168.1.80/phpMyAdmin/changelog (CODE:200|SIZE:74593)
+ http://192.168.1.80/phpMyAdmin/ChangeLog (CODE:200|SIZE:40540)
==> DIRECTORY: http://192.168.1.80/phpMyAdmin/contrib/
+ http://192.168.1.80/phpMyAdmin/docs (CODE:200|SIZE:4583)
+ http://192.168.1.80/phpMyAdmin/error (CODE:200|SIZE:1063)
+ http://192.168.1.80/phpMyAdmin/export (CODE:200|SIZE:4145)
+ http://192.168.1.80/phpMyAdmin/favicon.ico (CODE:200|SIZE:18902)
+ http://192.168.1.80/phpMyAdmin/import (CODE:200|SIZE:4145)
+ http://192.168.1.80/phpMyAdmin/index (CODE:200|SIZE:4145)
+ http://192.168.1.80/phpMyAdmin/index.php (CODE:200|SIZE:4145)
==> DIRECTORY: http://192.168.1.80/phpMyAdmin/js/
==> DIRECTORY: http://192.168.1.80/phpMyAdmin/lang/
==> DIRECTORY: http://192.168.1.80/phpMyAdmin/libraries/
+ http://192.168.1.80/phpMyAdmin/license (CODE:200|SIZE:18011)
+ http://192.168.1.80/phpMyAdmin/LICENSE (CODE:200|SIZE:18011)
+ http://192.168.1.80/phpMyAdmin/main (CODE:200|SIZE:4227)
+ http://192.168.1.80/phpMyAdmin/navigation (CODE:200|SIZE:4145)
+ http://192.168.1.80/phpMyAdmin/phpinfo (CODE:200|SIZE:0)
+ http://192.168.1.80/phpMyAdmin/phpinfo.php (CODE:200|SIZE:0)
+ http://192.168.1.80/phpMyAdmin/phpmyadmin (CODE:200|SIZE:21389)
+ http://192.168.1.80/phpMyAdmin/print (CODE:200|SIZE:1063)
+ http://192.168.1.80/phpMyAdmin/readme (CODE:200|SIZE:2624)
+ http://192.168.1.80/phpMyAdmin/README (CODE:200|SIZE:2624)
```

```
---- Entering directory: http://192.168.1.80/twiki/bin/ ----
+ http://192.168.1.80/twiki/bin/attach (CODE:200|SIZE:4356)
+ http://192.168.1.80/twiki/bin/changes (CODE:200|SIZE:21785)
+ http://192.168.1.80/twiki/bin/edit (CODE:200|SIZE:5345)
+ http://192.168.1.80/twiki/bin/manage (CODE:302|SIZE:0)
+ http://192.168.1.80/twiki/bin/passwd (CODE:302|SIZE:0)
+ http://192.168.1.80/twiki/bin/preview (CODE:302|SIZE:0)
+ http://192.168.1.80/twiki/bin/register (CODE:302|SIZE:0)
+ http://192.168.1.80/twiki/bin/save (CODE:302|SIZE:0)
+ http://192.168.1.80/twiki/bin/search (CODE:200|SIZE:3542)
+ http://192.168.1.80/twiki/bin/statistics (CODE:200|SIZE:1194)
+ http://192.168.1.80/twiki/bin/upload (CODE:302|SIZE:0)
+ http://192.168.1.80/twiki/bin/view (CODE:200|SIZE:10039)
+ http://192.168.1.80/twiki/bin/viewfile (CODE:302|SIZE:0)
```

Figure 7: dirb scan result

- **Impact:**
  - ➔ Finding hidden directories can reveal parts of the application that aren't meant to be public, or provide clues for further attacks

- **Recommendation:**
  - ➔ Review discovered directories for sensitive information and restrict access or remove them if not needed

iii. Burp Suite – Manual Testing

- **What it does:** Burp Suite is an intercepting proxy that let us see and modify web traffic between our browser and the web server.
- **Actions Taken:**
  - ➔ I configured my browser (in Kali) to use Burp's proxy (127.0.0.1:8080).
  - ➔ I browsed to http://192.168.1.80/mutillidae/ and observed the requests in Burp's Proxy history.
  - ➔ I sent a request for the user-info.php page (after a failed login attempt) to Burp's Repeater.
  - ➔ In Repeater, I modified the username parameter by adding a single quote (') to test for SQL injection.

- **What I Found:**
  - ➔ When I added a single quote to the username parameter, the server returned a detailed SQL error message. This is a strong indicator of a SQL Injection vulnerability.

- **Proof/Screenshot:**

Figure 8: burpsuite request using incorrect credential.

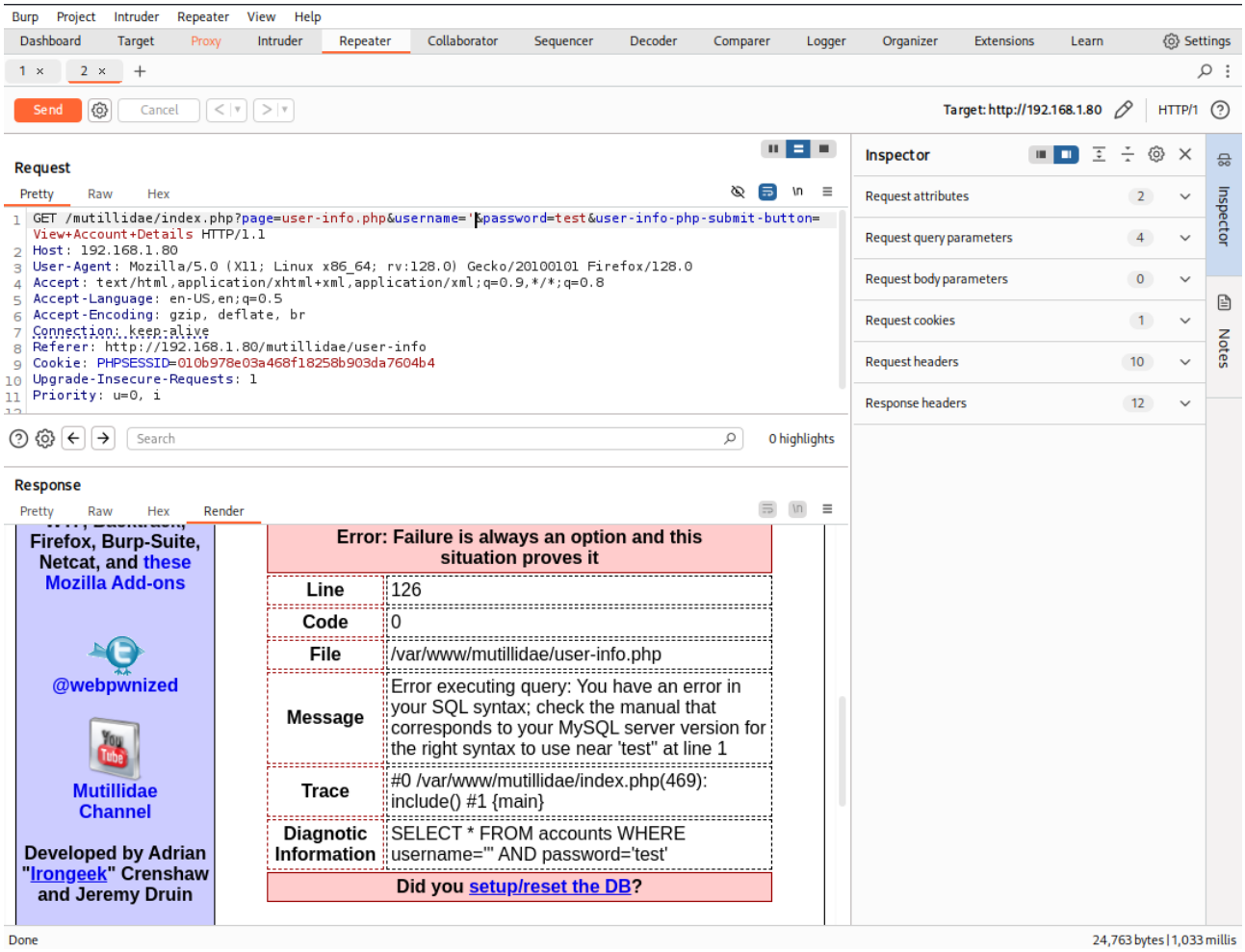

Figure 9: Burpsuite repeater showing SQL error after injecting a single quote in username field

- **Impact:**
  - → **SQL Injection:** Allows attackers to potentially access, modify, or delete database information.
- **Recommendation:**
  - → **For SQL Injection:** Implement parameterized queries or prepared statements for all database interactions.

iv. Sqlmap Exploitation:

- **What it does:**
  - → SQLMap automates finding and exploiting SQL injection vulnerabilities.
- **Command Used:**
  - → sqlmap -u http://192.168.1.80 --batch –crawl 3
  - → sqlmap -u http://192.168.1.80/mutillidae/user-info.php --dbs --level=5 --risk=3 --tamper=space2comment
- **What I Found:**
  - → Despite manual confirmation of a SQL Injection vulnerability in the username and password parameter of the login form on user-info.php using Burp Suite, SQLMap was unable to identify this specific SQL Injection vulnerability.
- **Proof/Screenshot:**

```
[12:51:49] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[12:51:49] [WARNING] POST parameter 'username' does not seem to be injectable
[12:51:49] [INFO] testing if POST parameter 'password' is dynamic
```

Figure 10: sqlmap showing username filed not inejectable

```
[12:51:51] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[12:51:51] [WARNING] POST parameter 'password' does not seem to be injectable
[12:51:51] [INFO] testing if GET parameter 'page' is dynamic
[12:51:51] [INFO] GET parameter 'page' appears to be dynamic
```

Figure 11: sqlmap showing showing password field not injectable

```
[12:51:54] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[12:51:54] [WARNING] GET parameter 'page' does not seem to be injectable
[12:51:54] [CRITICAL] all tested parameters do not appear to be injectable.
```

Figure 12: sqlmap showing all tested parameter not injectable

- **Impact:**
  - ➔ While SQLMap did not confirm the SQL Injection in the targeted login form, its inability to detect a known vulnerability highlights that automated tools are not foolproof and often require manual verification and sophisticated configuration.

- **Recommendation:**
  - ➔ Prioritize manual testing alongside automated scans. Automated tools like SQLMap have limitations and may miss critical vulnerabilities (as seen with the SQL Injection), making hands-on assessment crucial for full coverage.

4. Lessons Learned:

This project shows the importance of using a variety of tools for web application testing. Automated scanners like Nikto and Dirb proved useful for initial reconnaissance and quick overviews. Burp Suite was invaluable for its detailed manual analysis capabilities, allowing for precise vulnerability identification and confirmation, as seen with the SQL Injection flaw. And sqlmap inability to automatically detect the manually confirmed SQL Injection in a POST request highlighted that even advanced automated tools have limitations and can miss critical vulnerabilities.

5. Disclaimer:

This project was conducted for educational purposes only, within a controlled lab environment. Unauthorized penetration testing is illegal and unethical. Always ensure you have explicit permission before conducting any security assessments on systems you do not own.