|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | Network Penetration Testing  *CRN 50250* | |
| Password Attacks Exploitation Report | | | | |
|  | | | | |
| Date: April 15th, 2021  Project: 03-21  Version 1.0 | |  |  |  |

|  |
| --- |
|  |

Table of Contents

Table of Contents……………………………………………………………………………………………….2

Executive Summary……………………………………………………………………………………………….3

Phases of Penetration Test……………………………………………3

Findings Overview……………………………………………………… 4

Recommendations……………………………………………………… 4

Severity Scale………………………………………………………………4

Exploitation………………………………………………………………………………………………………....5

1.Target vsFTP……………………………………………………………..5

2.Target ProFTPD..........……………………………………………….11

3.Target DVWA............…………………………………………………16

Appendix.............................................................................................................................21

Table of Figures

[Figure 1.1: Nmap Scan Results 5](#_Toc64889312)

[Figure 1.2: vsFTP searchsploit results showing exploit. 6](#_Toc64889313)

[Figure 1.3: Metasploit Framework 6](#_Toc64889314)

[Figure 1.4: vsFTP Command Execution Exploit Show Info Page 7](#_Toc64889315)

Figure 1.5: Exploit Execution and Shell 8

Figure 1.6: Setting The Hashdump Post Module................................................................9

Figure 1.7: Using Run and Loot On The Hashdump Module.............................................9

Figure 1.8: Obtaining Hashed Access Credentials Using John.........................................10

Figure 2.1: Nmap Scan Displaying Information on Open Port 2121..................................11

Figure 2.2: Rapid7 Webpage Used To Gather Wordlist....................................................12

Figure 2.3: Starting CeWL and Offsite Link Message.......................................................13

Figure 2.4: Starting Apache2.............................................................................................13

Figure 2.5: Using wget to download the target Webpage..................................................14

Figure 2.6: Downloaded Webpage in html directory and accessed using localhost.........14

Figure 2.7: CeWL Successfully Creating Wordlist From Locally Hosted Webpage .........15

Figure 2.8: Hydra Successfully Retrieving Access Credentials For Login........................15

Figure 3.1: kayrrTolkienBruteForce.py.............................................................................16

Figure 3.2: DVWA Login Page With failed login attempt string.......................................17

Figure 3.3: Python Script Running Attack.........................................................................17

Figure 3.4: Brute Force Page Within The DVWA Web portal...........................................18

Figure 3.5: Intercepting Traffic To Get Current Cookie....................................................18

Figure 3.6: Failed Login Attempt Within Brute Force Page.............................................19

Figure 3.7: Successful Match From Brute Force...............................................................20

EXECUTIVE SUMMARY

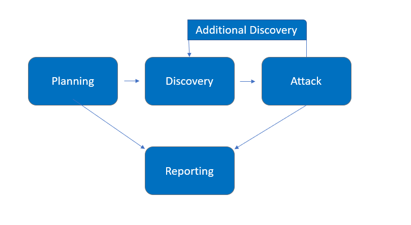
Akolade Adelaja was tasked to evaluate the Metasploitable 2 Virtual Machine’s security by engaging in a penetration test that was conducted on 10th April 2021. The goal of the “pentest” is to act as a malicious actor by performing cyber-attacks against the application with the aim of discovering any vulnerabilities that could lead to a breach and be leveraged to gain access to the system through the application.

This assessment harnessed testing based on the ***NIST SP 800-115 Technical Guide to Information Security Testing and Assessment***to provide documentation and proof of developing a working exploit. All issues discovered by Adelaja are achieved and verified through the evaluation of the network, vulnerability scanning and assessment on the system, and exploitation of found vulnerabilities using automated and manual processes where applicable..

PHASES OF PENETRATION TEST

Phases of penetration testing activities include the following:

* Planning – Goals are gathered, and rules of engagement obtained.
* Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
* Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
* Reporting – Document all found vulnerabilities and exploits, failed attempts, and recommendations.



FINDINGS OVERVIEW

While conducting the penetration test, there were several critical vulnerabilities discovered in the Metasploitable 2 network. A brief technical overview is listed below:

**Target: vsFTP 2.3.4(Critical)** – Low-privilege shell was obtained by performing an automated backdoor command execution exploit against the vulnerable FTP server found open on port 21. Once access was established, usernames and passwords files were extracted which were used to obtain access credentials.

**Target: proFTPD 1.3.1(Critical)** – Gained administrative/root access by performing a brute force attack against the FTP server found on port 2121 granting Adelaja access as the FTP service account **‘user’**.

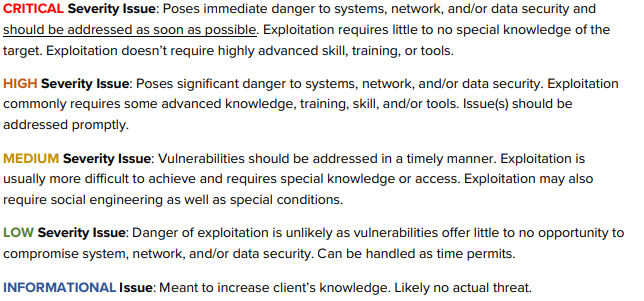
**Target: DVWA(High)**– Gained administrative/root access by performing a brute force attack against the Metasploitable 2 web app ‘DVWA’ found at <http://10.0.2.7/dvwa/> granting Adelaja access as the HTTP admin account **‘admin’**. Once access was established, further access into the admin protected area found at <http://10.0.2.7/dvwa/vulnerabilities/brute> was possible by performing a brute force attack against the internal login screen.

RECOMMENDATIONS

To increase security posture and prevent Password Attacks in Enterprises, Adelaja recommends the following mitigations and/or remediations be performed:

* **Multi Factor Authentication.** Calling on a Privileged Access Management (PAM) solution which deploys multifactor authentication(MFA) puts different layers of identity security on each account within the enterprise.
* **Implement Account Lockout.** These prevent any more login attempts for a period after a certain number of failed logins. Note that the counter should be associated with the account rather than the source IP Address. This is to prevent attackers from making login attempts from many different IP addresses.
* **Logging and Monitoring**. Enable logging and monitoring of authentication to detect attacks/failures on a real time basis by ensuring that all failures and account lockouts are logged and reviewed regularly.

SEVERITY SCALE



EXPLOITATION

During the exploitation phase, Adelaja will leverage a series of password attacks against the Metasploitable 2 machine. The end goal for the tester is to attempt to penetrate the target system gaining as much access privilege as possible. Adelaja will stay within the scope that was determined during pre-engagement.

1. **Target: vsFTP 2.3.4**

Before proceeding to develop the exploit. The application is checked to determine the security of services running on its open ports. To do this an Nmap service scan is run targeting every open port available. [Figure 1.1].

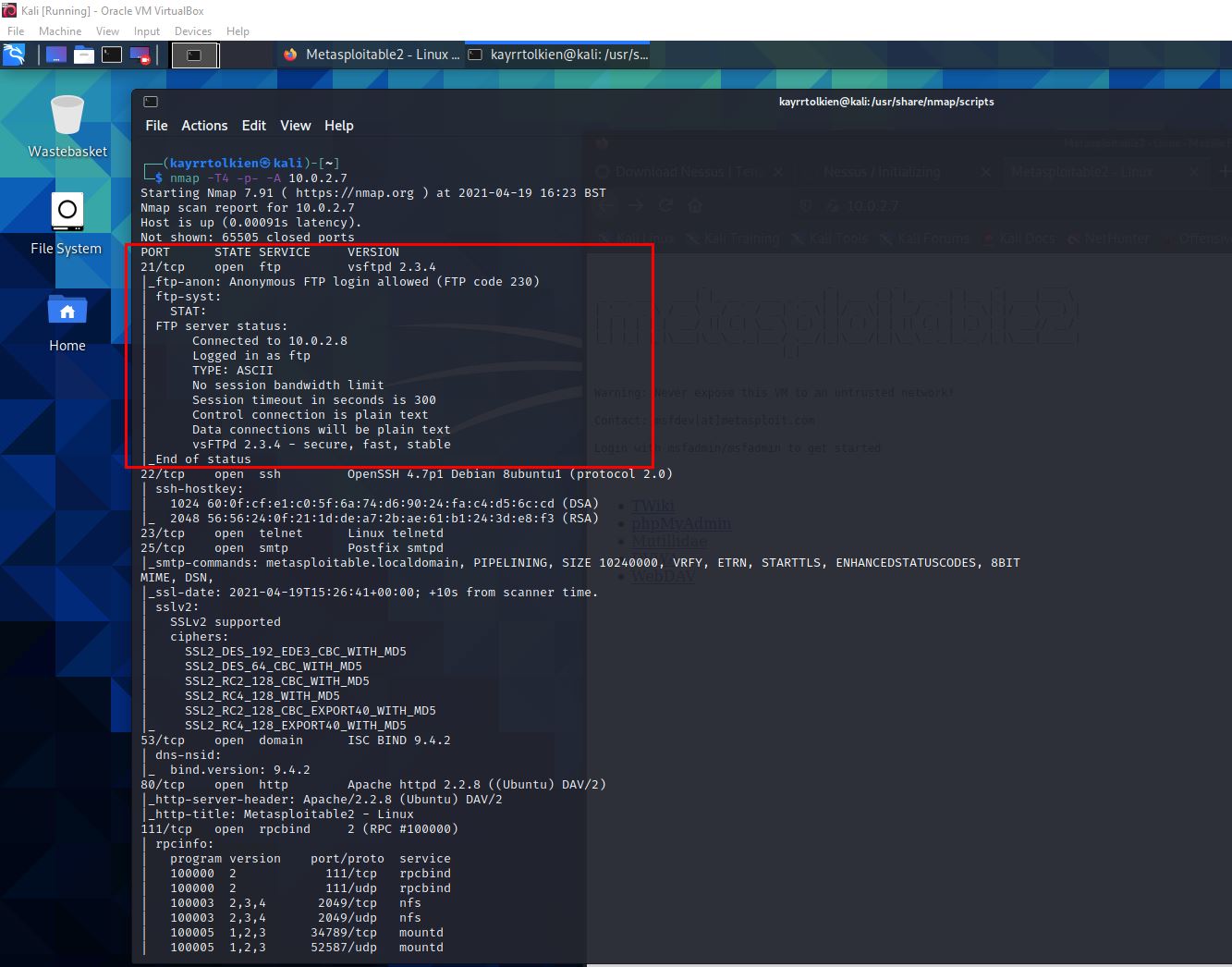


Figure 1.1: Nmap Scan Results

The resulting scan shows an open port 21 running an FTP server, specifically a vsFTP 2.3.4. A quick check on exploitDB using searchsploit reveals a backdoor command execution exploit for the version of vsFTP currently running on the Metasploitable 2 machine[Figure 1.2].

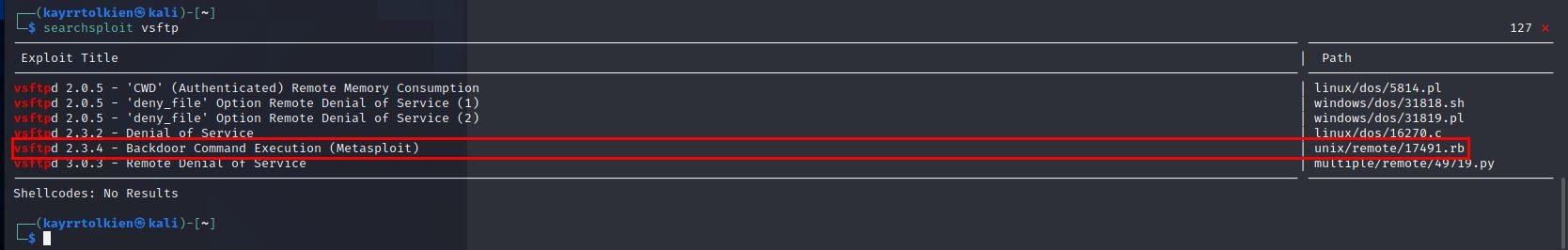


Figure 1.2: vsFTP searchsploit results showing exploit.

The Metasploit Framework contains a suite of tools and modules with different functionalities that can load, view, and run an exploit[Figure 1.3].

Text

Description automatically generated

Figure 1.3: Metasploit Framework.

As seen in Figure 1.3, the framework is first loaded up before the exploit needed is searched for using **“grep vsFTP search exploits”**. There is only one and it is selected using the **‘use’** keyword along with either the exploit name or number.

With the exploit selected, a few other parameters will have to be set for the exploit to work. The **‘show info’** keyword will display these parameters along with information relevant to the exploit[Figure 1.4].

Text

Description automatically generated

Figure 1.4: vsFTP Command Execution Exploit Show Info Page

As seen in Figure 1.4, the basic options listed under the Show Info page of the framework is edited to provide the target host and the target port where the exploit would be sent to and executed. Typing the **‘run’** keyword will run the exploit and create a shell and open a session indicating a successful exploit[Figure 1.5].

With the successful creation of the shell, and with it running in the background (Using CTRL D and sessions -l to put shell session to background and check on active sessions), username, and password extraction from the Metasploitable 2 machine is possible using the hashdump post module available on the Metasploit Framework[Figure 1.6].

The Hashdump module will gather password hashes dump for Linux systems. Figure 1.6 shows how this module is selected and the required module options needed for it to run successfully. With these set, the **‘run’** command is used to gather the password hashes and the **‘loot’** command is used to view the results. Here the passwd, shadow and unshadow files were obtained[Figure 1.7].

Text

Description automatically generated

Figure 1.5: Exploit Execution and Shell

Graphical user interface

Description automatically generated with low confidence

Figure 1.6: Setting The Hashdump Post Module

Graphical user interface, website

Description automatically generated

Figure 1.7: Using Run and Loot On The Hashdump Module

The obtained password files are still hashed and would need to be unhashed or decoded to reveal the actual passwords and usernames. The password cracking tool John The Ripper is used to crack the hashed files[Figure 1.8].

Text

Description automatically generated

Figure 1.8: Obtaining Hashed Access Credentials Using John

1. **Target: proFTPD 1.3.1**

Before proceeding to develop the exploit. The application is checked to determine the security of services running on its open ports. To do this an Nmap service scan is run targeting every open port available[Figure 2.1].

Text

Description automatically generated

Figure 2.1: Nmap Scan Displaying Information on Open Port 2121

The resulting scan shows an open port 2121 running an FTP server, specifically a ProFTPD 1.3.1. Checks show that there are no known vulnerabilities for this version of ProFTPD to exploit. Its security posture will be further tested by attempting a brute force attack on the port by using a created wordlist of usernames and passwords and passing all possible combinations from the wordlists until the correct one is found using the tool Hydra.

To create the wordlist, a ruby application called CeWL(pronounced as cool) is used to spider a given URL to a specific depth and return a list of words which can then be used to create a password or username list which can be used by tools like Hydra to automate the password combination process. CeWL will be used to spider the web page found on: https://docs.rapid7.com/metasploit/metasploitable-2-exploitability-guide[Figure 2.2].

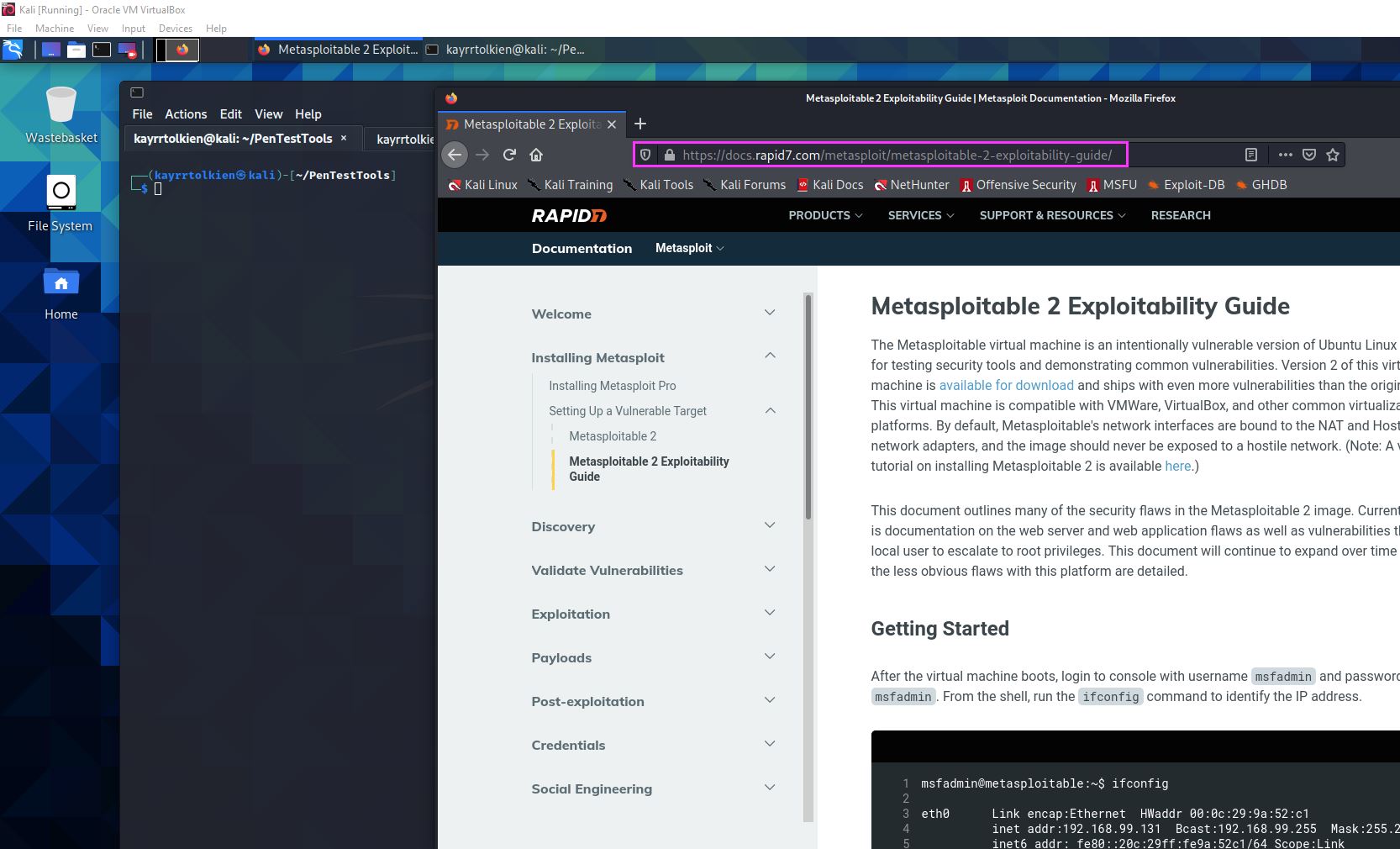


Figure 2.2: Rapid7 Webpage Used To Gather Wordlist

The webpage on inspection seems to have a lot of information on the Metasploitable 2 machine, maybe there are some keywords shared that could prove useful. Using CeWL with the -d option specifies the depth to spider, -m option specifies the minimum word length, which is 4 in this instance, and the -w specifies it write its output to the file dictionary.txt[Figure 2.3].

Unfortunately, CeWL does not work when it is redirected to another server, to use CeWL in this instance, the website will have to be downloaded to the local system and then hosted using Apache before calling CeWL to spider the locally hosted webpage. Start by calling the Apache2 service[Figure 2.4] and using wget to download the webpage in its entirety to the local machine[Figure 2.5].

Move the downloaded page to Kali’s /var/www/html folder where it should now be hosted on the local webserver[Figure 2.6]. CeWL can now be used successfully to spider the webpage and return the wordlist[Figure 2.7].

Text

Description automatically generatedFigure 2.3: Starting CeWL and Offsite Link Message

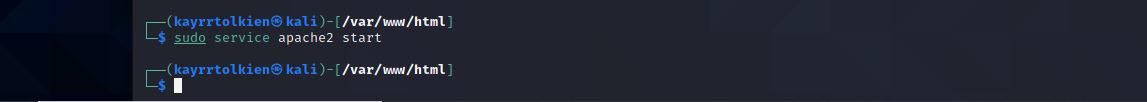


Figure 2.4: Starting Apache2

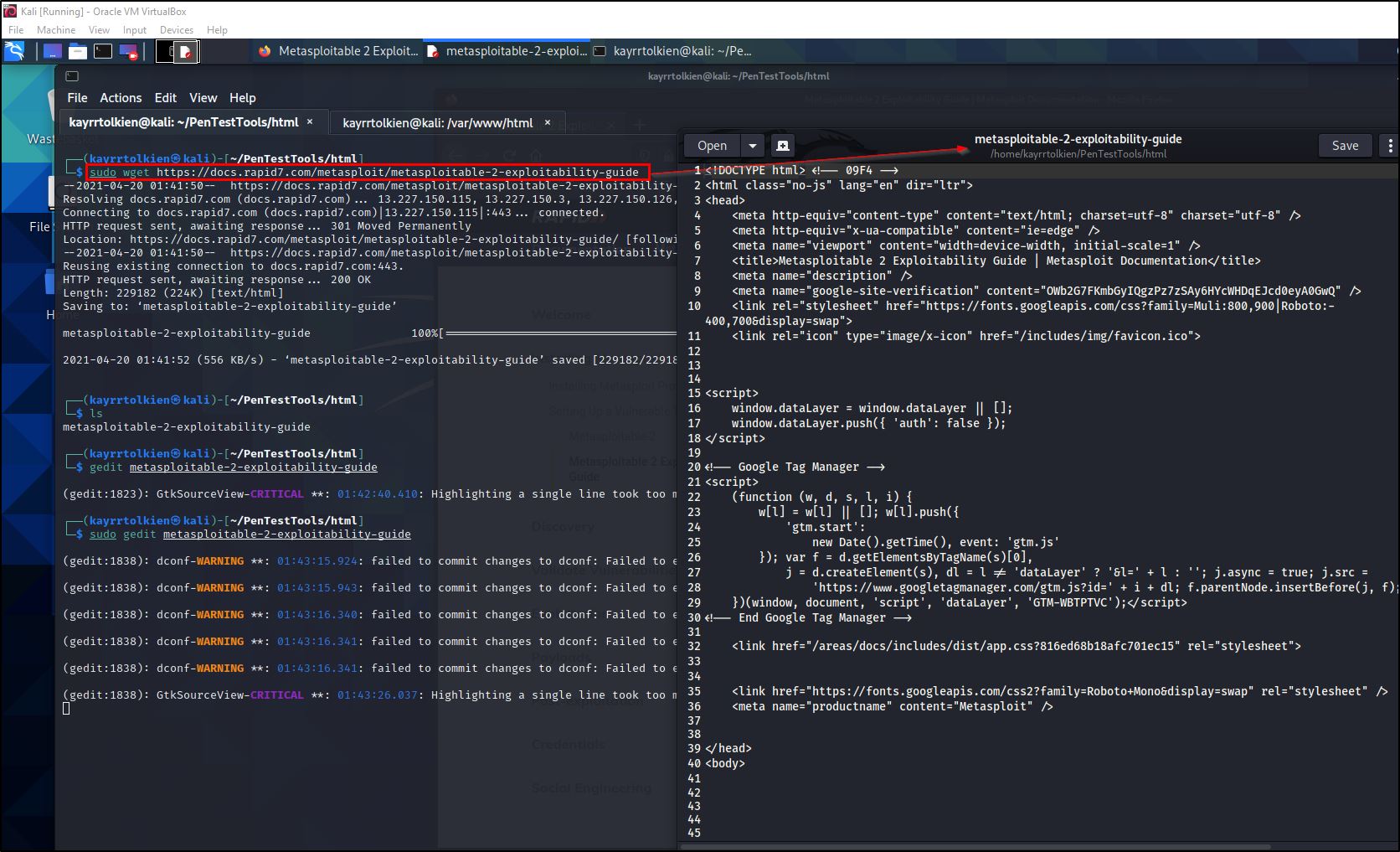


Figure 2.5: Using wget to download the target Webpage.

Graphical user interface, text, application

Description automatically generated

Figure 2.6: Downloaded Webpage in html directory and accessed using localhost.

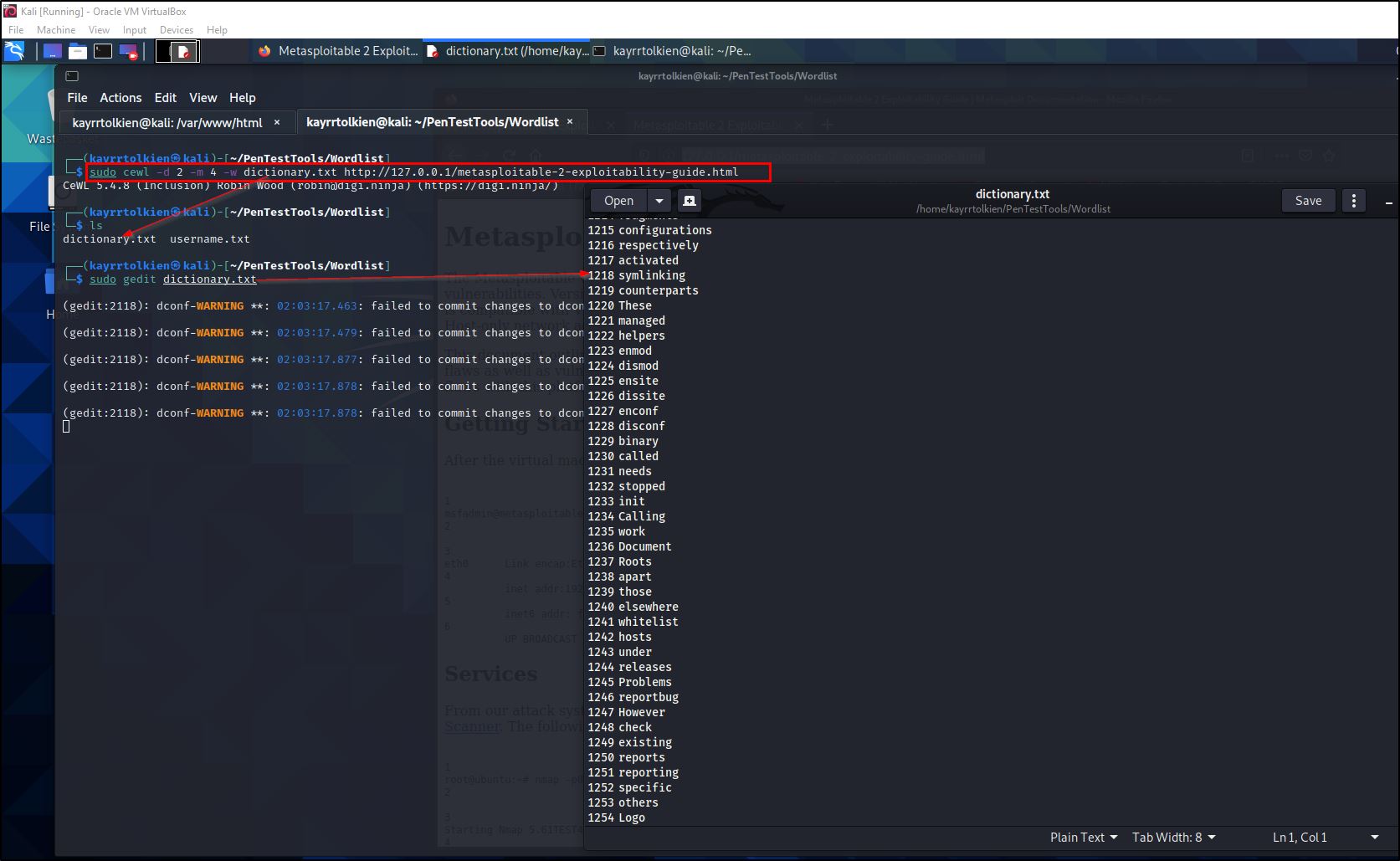


Figure 2.7: CeWL Successfully Creating Wordlist From Locally Hosted Webpage

With the generated wordlist, a brute force attack using Hydra can now be performed on the target ftp server on port 2121[Figure 2.8].

Text

Description automatically generated

Figure 2.8: Hydra Successfully Retrieving Access Credentials For Login

As highlighted above in Figure 2.8, It retrieves the access credentials for a successful login to the ProFTPD Service running on port 2121.

1. **Target: DVWA**

To gain access to the DVWA portal found at <http://10.0.2.7/dvwa/> , a python script is developed which will send possible matching combination from the generated wordlist gotten from CeWL to brute force the portals login page and the admin’s protected page within the DVWA web portal. The script below [Figure 3.1] will be using the variables:

* page\_url, which is the input value of the webpage to be brute force,
* username, which is the input value of a known account username,
* password\_file, which is the input value of a wordlist or password file.
* failed\_login\_string, which is the input value of the string displayed after a failed login attempt.
* Cookie\_Value, which is an optional input value of the current session.

Note that the imported requests library allows the script to make GET and POST requests, termcolor allows for printing certain string in red and green.

Graphical user interface, text

Description automatically generated

Figure 3.1: kayrrTolkienBruteForce.py

Running the script will present an input field, asking for the webpage URL to be brute forced, the username (in this case admin), the dictionary file generated previously, and the string displayed after a failed login attempt on the login form to be attacked[Figure 3.2]. It will test every possible combination against the admin username and if successful will display its results[Figure 3.3].

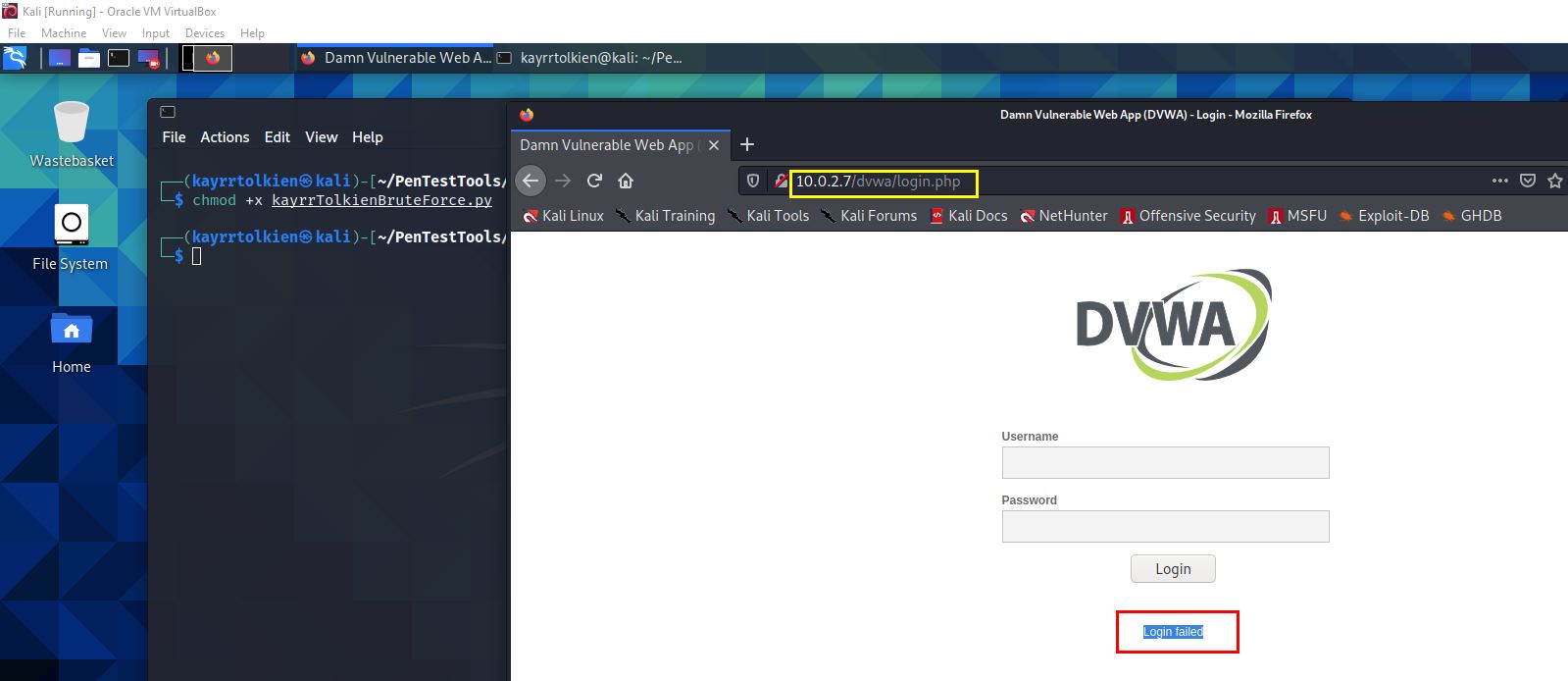


Figure 3.2: DVWA Login Page With failed login attempt string

Text

Description automatically generated

Figure 3.3: Python Script Running Attack

Can also use this python script on the Brute Force page within the DVWA website[Figure 3.4].

Graphical user interface, text, application, email

Description automatically generated

Figure 3.4:Brute Force Page Within The DVWA Web portal

This requires a different method of brute forcing as it uses a GET request, while the DVWA login page used a POST Request, to handle this the Python script has an if else clause which handles the GET request when a cookie value is passed to it As this page can only be accessed when a user logs in, the page can only be brute forced within a current session. This is where the optional cookie value acquired by intercepting the traffic using burp[Figure 3.5] come in.

Graphical user interface, text, email

Description automatically generated

Figure 3.5:Intercepting Traffic To Get Current Cookie

When the script runs, an input field is requesting the URL to be brute forced, here we provide the URL, the username, the dictionary file generated previously, the string displayed after a failed login attempt within the brute force page[Figure 3.6] and the current session cookie.

Graphical user interface, text, application, website

Description automatically generated

Figure 3.6: Failed Login Attempt Within Brute Force Page

The Script will attempt to match possible combinations of the given username and characters from the dictionary file and if it does it will output the results[Figure 3.7]

Graphical user interface

Description automatically generated

Figure 3.7: Successful Match From Brute Force

**Appendices**

**Definitions:**

1. An **Exploit** executes a sequence of commands that target a specific vulnerability found in a system or Application to provide an attacker with access to the system. is a register that contains the address of the next instruction for the program or command. Can be seen on the immunity Debugger
2. **Credentials** provide a gateway into various accounts and systems, which, potentially, can give access to additional targets on the network and lead to an extraction of confidential data from these targets.
3. A **Brute Force** attack automatically and systematically attempts to guess the correct username and password for a service. The goal is to find valid logins and leverage them to gain access to a network to extract sensitive data.
4. **The Metasploit Framework** is a ruby based modular penetration testing platform that enables the writing, testing and execution of exploit code.
5. **Hydra** is a parallelized login cracker which supports numerous protocols to attack making it possible for researchers and security consultants to show how easy it would be to gain unauthorized access to a system remotely.
6. The "**hashdump**" command is an in-memory version of the pwdump tool, but instead of loading a DLL into LSASS.exe, it allocates memory inside the process, injects raw assembly code, executes its via CreateRemoteThread, and then reads the captured hashes back out of memory.
7. **FTP stands for** File Transfer Protocol, and as the name suggests it is a standard for transferring files over the Internet.
8. **Hashes** are the output of a hashing algorithm such as MD5 (Message Digest 5) or SHA (Secure Hash Algorithm). These algorithms aim to produce a unique, fixed-length string called the hash value, or “message digest” – for any given piece of data or “message”. As every file on a computer is, ultimately, data that can be represented in binary form, a hashing algorithm takes that data runs a complex calculation on it and outputs a fixed-length string as the result of the calculation. The result is the file’s hash value.