International Journal of Future Engineering Innovations

Secure Framework for Developing and Executing Exploits Against Vulnerable Systems

Firas Abdul Hadi AL-Khaledy [1*], Ameer Sameer Hamood Mohammed Ali [2]
[1] Presidency of the University of Babylon, University of Babylon Multimedia Center, Babylon, Iraq
[2] Presidency of the University of Babylon, University of Babylon TOEFL Center, Babylon, Iraq

* Corresponding Author: **Firas Abdul Hadi AL-Khaledy**

Article Info

ISSN (online): 3049-1215
Volume: 02
Issue: 03
May-June 2025
Received: 10-04-2025
Accepted: 09-05-2025
Page No: 161-170

Abstract
A secure framework for developing and executing exploits against vulnerable systems is essential in modern cybersecurity research and ethical hacking. Such a framework, like the Metasploit Framework, provides a controlled environment where security professionals can create, test, and execute exploits without compromising real-world systems. It enables safe vulnerability assessment and penetration testing by offering modular tools that support payload generation, exploit execution, and post-exploitation analysis. This paper explores the use of the Metasploit Framework to exploit vulnerabilities in the intentionally vulnerable virtual machine, Metasploitable. As one of the most widely utilized tools by cybersecurity professionals, Metasploit provides a comprehensive platform for developing, testing, and executing exploits. Through systematic reconnaissance and various scanning techniques, open ports were identified, leading to the discovery of underlying vulnerabilities within the target system. These vulnerabilities were then leveraged to successfully compromise the Metasploitable environment, demonstrating the practical application of Metasploit in penetration testing. The experiment highlights the framework's flexibility, efficiency, and effectiveness in real-world scenarios, showcasing its utility for vulnerability assessment, network enumeration, exploitation, and evasion. The findings reinforce Metasploit's role as an indispensable tool for ethical hackers and security researchers, and emphasize the importance of hands-on penetration testing in enhancing cybersecurity posture.

Keywords: Secure Framework, Metasploit, Penetration Testing, Vulnerabilities

1. Introduction

In the ever-evolving landscape of cybersecurity, the ability to identify, assess, and exploit vulnerabilities in information systems is a critical skill for both ethical hackers and security researchers [1]. As cyber threats continue to increase in complexity and frequency, organizations must adopt proactive security measures that go beyond traditional defenses [2]. One such measure is penetration testing a simulated cyberattack that helps uncover security weaknesses before they can be exploited by malicious actors [3]. However, performing these tests in uncontrolled environments poses significant ethical and legal risks, necessitating the development of secure, isolated frameworks for vulnerability research and exploit development [4].

The Metasploit Framework has emerged as one of the most powerful and widely used tools for penetration testing and exploit development. It provides a modular, open-source environment where security professionals can safely simulate real-world attacks against vulnerable systems. By enabling users to craft payloads, execute exploits, and perform post-exploitation tasks within a controlled setting, Metasploit serves as a vital resource for ethical hacking training and vulnerability assessment [5].

This paper presents a secure framework for developing and executing exploits using Metasploit, targeting an intentionally vulnerable virtual machine known as Metasploitable. The objective is to demonstrate how Metasploit can be effectively used for reconnaissance, vulnerability discovery, and system exploitation without causing harm to production environments. By analyzing the process of identifying open ports, detecting service vulnerabilities, and deploying exploits, we aim to highlight the practical applications and educational value of using such frameworks in cybersecurity practices.

The remainder of this paper is structured in the following way: Section 2 examines useful literature on exploit frameworks and penetration testing methodologies. In Section 3, we explain the methodology's proposed setup and the tools employed, including the reconnaissance and exploitation phases. Section 4 offers an analysis of the results from the simulation and the understanding gained from them. We articulate our final remarks and prospective research questions in Section 5.

2. Related Works

The field of ethical hacking and vulnerability assessment has seen significant growth in recent years, particularly with the advancement of frameworks such as Metasploit that support exploit development and penetration testing in controlled environments. Numerous studies have explored the role of automated tools, machine learning, and reinforcement learning in enhancing the efficiency, precision, and adaptability of security testing processes. This section reviews relevant literature that examines exploit frameworks, intelligent penetration testing systems, and the use of virtualized environments like Metasploitable for cybersecurity education and research.

The detectability of Metasploit's encrypted Command and Control (C2) traffic has been investigated in [6] by machine learning-based detectors. The authors identify distinctive traffic patterns in Metasploit's C2 communications and propose modifications to the framework to evade detection.

In [7] authors addressed the shortage of skilled cybersecurity professionals; they explored the application of model-free reinforcement learning to automate penetration testing. The authors design a simulator framing penetration testing as a Markov Decision Process, enabling agents to learn optimal attack strategies without prior environmental models. Their findings indicate that reinforcement learning can effectively identify attack paths in simulated environments, suggesting potential for scalable automated security assessments.

In [8] they introduced Raijū, a framework leveraging reinforcement learning to automate post-exploitation activities in network security assessments. Implementing Advantage Actor-Critic and Proximal Policy Optimization algorithms, Raijū trains agents to autonomously perform tasks such as privilege escalation and lateral movement using Metasploit modules. Experimental results demonstrate that the agents achieve high success rates in real-world environments, showcasing the framework's effectiveness in automating complex post-exploitation processes.

In [9] PentestGPT is introduced as an automated penetration testing tool powered by large language models (LLMs). The system comprises three self-interacting modules that handle sub-tasks of penetration testing, addressing challenges like context loss in LLMs. Evaluations reveal that PentestGPT significantly outperforms baseline LLMs in task completion and effectiveness, indicating its potential to revolutionize automated security assessments by integrating advanced language understanding capabilities.

In [10] authors addressed the process of conducting penetration testing on the Metasploitable2 virtual machine using the Metasploit Framework. The authors outline various tools and techniques employed to identify and exploit vulnerabilities within the system. Their work serves as a practical guide for security professionals and researchers aiming to understand the application of penetration testing methodologies in controlled environments.

3. The proposed methodology

The proposed system implemented based on the following steps:

3.1. The proposed system main design

This proposed system implemented the process of a cyberattack or penetration test using tools such as Metasploit and social engineering techniques. It shows how an attacker or security tester can infiltrate a network, exploit vulnerabilities, and move deeper into an organization's internal systems.

Step 1: Users (Targets)

- The process starts with regular users who operate computers within or connected to an organization's network.
- These users can be indirectly exploited through vulnerabilities in systems they use or directly via social engineering.

Step 2: Use of Metasploit

- Metasploit is a powerful penetration testing framework.
- It is used to scan, identify vulnerabilities, and exploit systems.
- Attackers or ethical hackers use it to create and deliver payloads that can control a system remotely.

Step 3: Firewall Bypass

- A firewall stands between the attacker and the internal network, acting as a security barrier.
- Attackers attempt to bypass the firewall using discovered vulnerabilities or through social engineering tactics that trick a user into allowing access.

Step 4: Scanning, Auditing, and Exploiting

Once past the firewall, the attacker performs

- **Scanning:** To detect active devices and open ports.
- **Auditing:** To identify weaknesses in system configurations or software.
- **Exploiting:** To gain unauthorized access using known vulnerabilities.

Step 5: Attacking Internal Systems

After successful exploitation, the attacker targets various internal components:

- **Web Server:** May contain publicly accessible services. Vulnerabilities here can be exploited and then used to launch pivoting attacks to other systems.
- **Database**: Often holds valuable data like credentials or internal records. Attackers can use this to gain deeper control.
- **Internal Server:** Central to internal operations. Compromising it gives wide access to the attacker.
- **Client Machines:** Computers used by employees. They can be accessed via technical exploits or social engineering.

Step 6: Social Engineering

- This involves tricking users into giving away access or sensitive information (e.g., phishing emails, fake software updates).
- Social engineering can bypass technical defenses by manipulating human behavior.

Step 7: Pivoting

- Once inside a system, attackers use it as a launch point to attack others on the same network a process called pivoting.
- This allows the attacker to move laterally within the network and gain control over multiple devices and services.

The main design of the used framework is showed in Figure 1

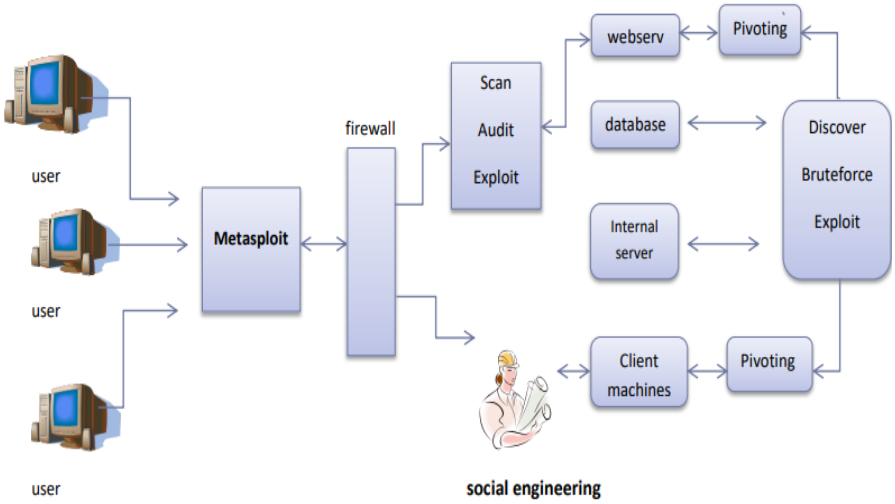

Fig 1: The main design of the used Metasploit framework

Once installed, organizing the file system and associated libraries is straightforward. Since Metasploit operates primarily through scripting, its script directory includes essential components such as Meterpreter and other necessary scripts. Users can choose between a graphical user interface (GUI) and a command-line interface (CLI) for interacting with the framework. Although all functions are available through the CLI, some users may find the GUI more user-friendly. The main design consists of user as the used personal computer which is tested the Metasploit framework which it considered firewall to encapsulate secure area and isolated the network from the outside attacks by providing network administrator a real time monitoring tool to discover, vulnerability scanning, and exploit.

3.2. The proposed system diagram

The proposed system based on the main steps showed in Figure 2.

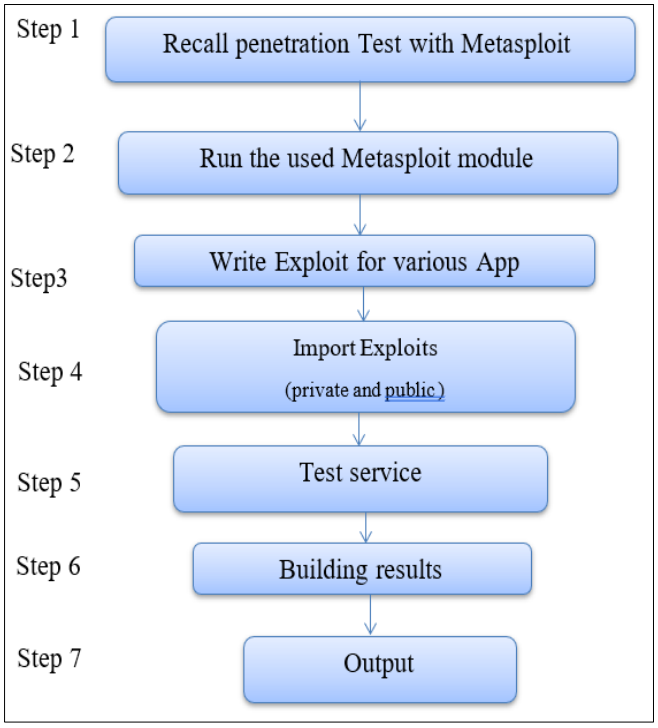

Fig 3: The used steps of the vulnerability testing system using Metaspoit framework.

3.3. Steps of building

The Metasploit framework is one of the most powerful penetration testing software that comes with multiple tools and modules to help in the discovery and exploitation of vulnerabilities of a target system. Here are the general steps to use Metasploit to find vulnerabilities:

- **Install Metasploit Framework:** Ensure that you have Metasploit installed on your system. Metasploit is available for various platforms, including Linux, Windows, and macOS.
- **Update Metasploit:** It's essential to keep Metasploit up to date to ensure you have the latest exploits and modules. You can update Metasploit using the msfupdate command.
- **Identify Target:** Determine the target system or network that you want to test for vulnerabilities. This could be a specific IP address, range of IP addresses, domain name, etc.
- **Perform Scanning:** Use Metasploit's scanning modules to gather information about the target system. Commonly used scanning modules include:
  auxiliary/scanner/tcp/
  auxiliary/scanner/http/
  auxiliary/scanner/ssh/
  auxiliary/scanner/smb/
- **Identify Vulnerabilities:** Analyze the results obtained from scanning to identify potential vulnerabilities in the target system. Metasploit will often highlight services, open ports, and potential vulnerabilities that you can exploit.
- **Select Exploit Module:** Based on the identified vulnerabilities, select an appropriate exploit module from Metasploit's extensive library. You can use the search command to find relevant exploit modules based on services, ports, or vulnerabilities.
- **Configure Exploit Module:** Once you've selected an exploit module, configure it with the necessary options such as target IP address, port, payload, etc. It can use the show options command to view and set the required options.
- **To run the exploit:** execute the exploit module with the exploit command. Metasploit will try to exploit the vulnerability that it will attempt to exploit in the system. If it succeeds, it will gain access to the system or execute the payload set.
- **Post-Exploitation:** After successfully exploiting the vulnerability, perform post-exploitation activities such as gathering information, escalating privileges, pivoting to other systems, etc. Metasploit provides various post-exploitation modules for these purposes.
- **Cleanup:** Once completed the testing and achieved the objectives, ensure to clean up any traces left on the target system. This might involve removing files, restoring configurations, or reverting changes made during the testing.
- **Documentation:** Document the findings, including the vulnerabilities discovered, the exploits used, and any recommendations for remediation. This documentation will be useful for reporting and further analysis.

Figure 3 showed the Metasploit installation steps.

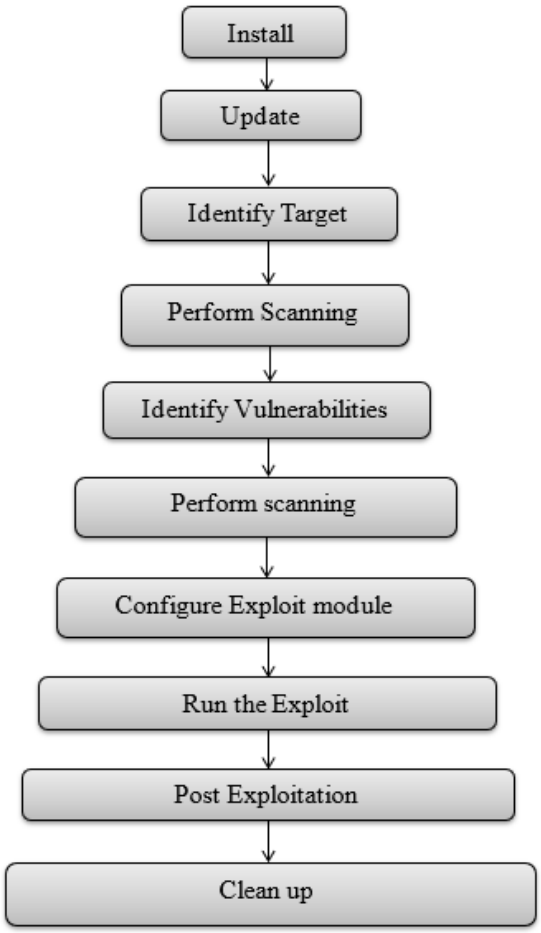

Fig 3: Metasploit installation steps.

Fig 4 showed the current options to run Metasploit framework.

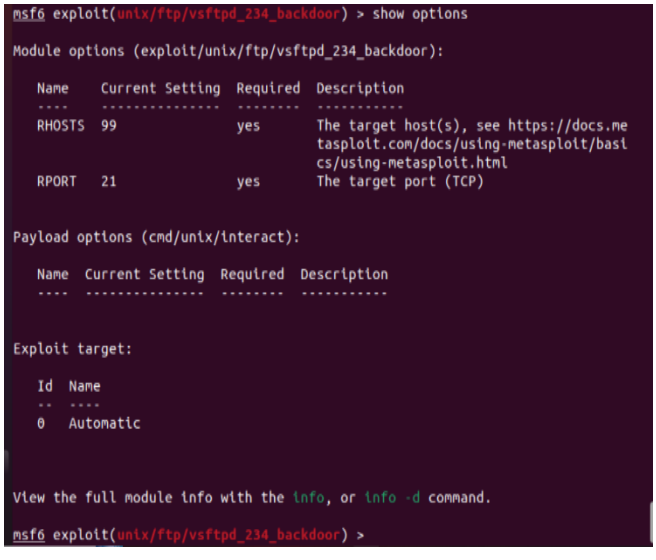

Fig 4: Running Metasploit framework in Linux-Ubuntu.

4. Implementation and Results

The Framework was as seems demonstrated the power of Metasploit as a penetration testing and vulnerabilities assessment tool. It provides a platform and excellent tools for exploiting an assortment of operating systems, applications, and devices; as well as performing deep system security audits.

4.1. Implementation

The proposed system is implemented in Linux operation system and the system configuration showed as follows:

Table 1: Characteristics of the device used to implement this tool

Processor	Intel(R) Core(TM) i5-5300U CPU @ 2.30GHz 2.29 GHz
Installed RAM	4.00 GB (3.88 GB usable)
Edition	Windows 10 Pro
Version	22H2
OS build	19045.4291
Experience	Windows Feature Experience Pack 1000.19056.1000.0
Version implement tool	Metasploit v6.4.4.1 –dev-

4.2 Results

The proposed system results are based on following steps to provide detailed instructions for installing the Metasploit Framework on Ubuntu 22.04 using the Nightly build installer.

Step 1: System Update

It make sure that your Ubuntu 22.04 operating system is fully updated. Go ahead and open your terminal and execute the command below.sudo apt update.

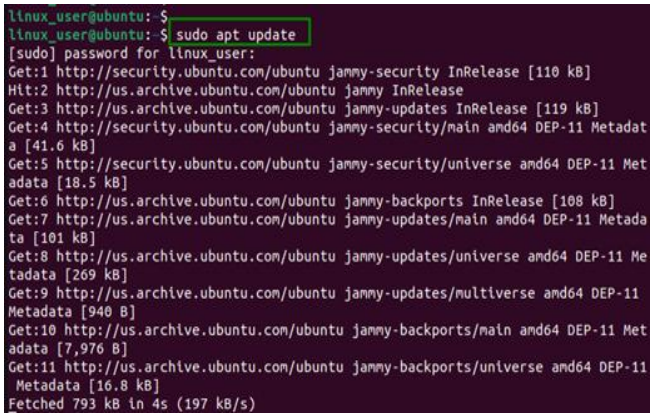

Fig 5: update ubuntu in virtualbox

Step 2: PostreSQL Installation

It can set up PostgreSQL from the terminal using the command down below on your Ubuntu 22.04 system:
sudo apt install postgresql postgresql-contrib -y

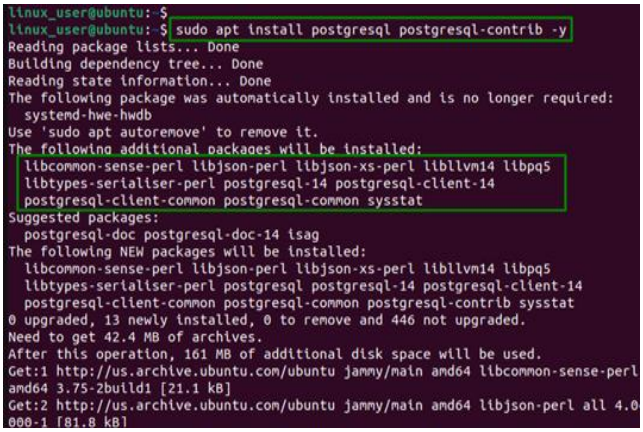

Fig 6: PostgreSQL installation on Ubuntu

Next, run this command for enabling the PostgreSQL service:
sudo systemctl enable --now postgresql

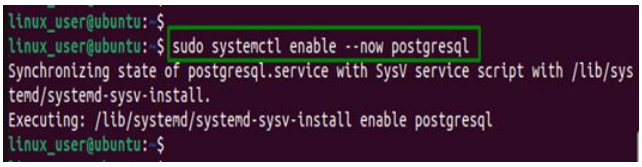

Fig 7: run this command for enabling the PostgreSQL service

Step 3: Download Metasploit Framework installer script
To set up the Metasploit Framework on Ubuntu 22.04, you can use the Nightly build installer script offered by Rapid7. This installer automatically configures the necessary repository and retrieves all required dependencies to complete the installation process.
Execute this command for downloading the installer script:
curl https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb > msfinstall

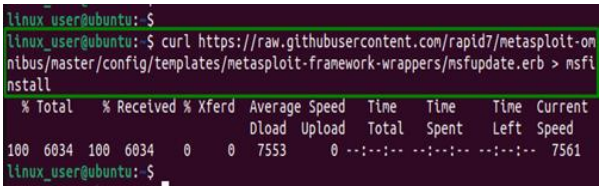

Fig 8: Download Metasploit Framework installer script

Step 4: Run Script

Then, use the following command to grant executable permissions to the installer script:
chmod +x msfinstall

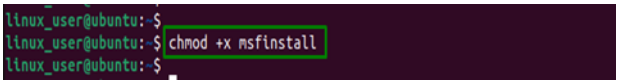

Fig 9: installer script executable

At this point, users can launch the Metasploit installer by executing:
sudo ./msfinstall

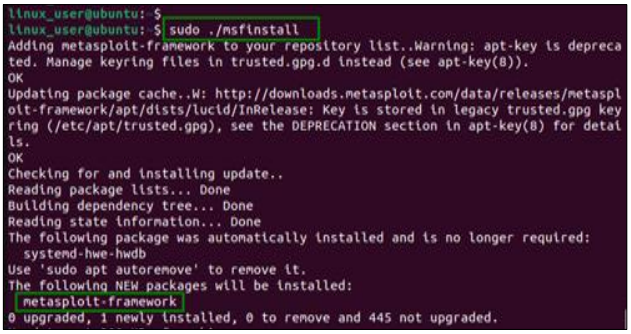

Fig 10: users can launch the Metasploit installer

The installation process may take a few minutes and will generate various messages in the terminal as it progresses. After it completes, the Metasploit Framework will be successfully set up on your system.
Launching Metasploit Framework on Ubuntu 22.04
After completing the installation, you can start the Metasploit Framework by executing the following command:
sudo msfconsole

Fig 11: Launch Metasploit Framework on Ubuntu

Step 5: Metasploit commands

To begin effectively using Metasploit for exploiting vulnerabilities in systems like Metasploitable 2, it's important to understand its core commands. These foundational commands include help, back, exit, and info, which are essential for navigating and managing modules within the framework. The use command is particularly significant, as it loads a specific module and shifts the msfconsole into the context of that selected exploit or auxiliary tool. Once activated, the name of the chosen module appears in red on the terminal prompt, signaling that the user is now operating within that module's environment. The back command exits the module context, while exit closes the console session entirely:

Fig 12: back and exit commands

In this scenario, we transitioned the command-line interface into the context of a specific exploit module, namely relance client. Once within this module, users can view detailed information about the exploit, configure necessary parameters, and execute the attack on a designated target system.

To exit the current exploit context and return to the main msfconsole interface, the back command should be used. This command reverts the console to its general state, allowing users to load a different Metasploit module by using the use command once again.

If the intention is to completely terminate the Metasploit session, the exit command will close the console and return the user to the Linux shell prompt.

The help command serves as a reference tool, displaying a list of available commands along with brief descriptions. When used in the main console or within a specific module, it provides relevant options, including module-specific commands when an exploit is active.

Fig 13: Metasploit exploit help command

There are different commands implemented to show system results as follows:

Info command

After selecting an exploit using the use command, detailed information about it—such as its name, supported platform, author, available targets, and more—can be obtained by running the info command. In the example shown in the screenshot, the info command is used on the exploit called vsftpd_234_backdoor:

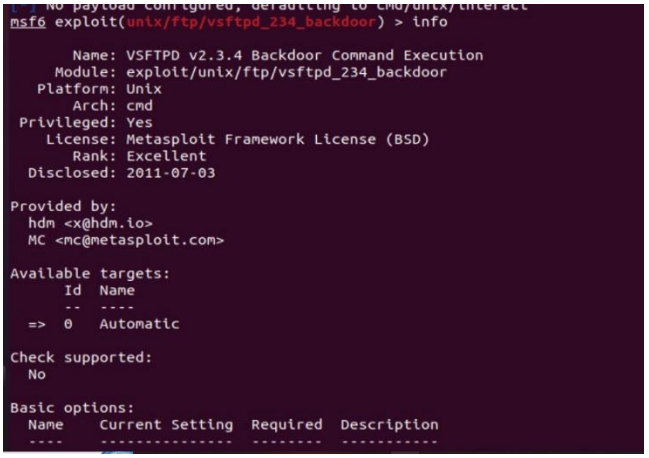

Fig 14: Info command

Search command

At the time of writing, the Metasploit Framework includes over 1,500 exploits, and new ones are frequently added. Given the large number of available modules, effectively using the search functionality becomes essential. The simplest method is to enter the search command followed by a keyword—for example, searching for vsftpd_234_backdoor will return related exploits. This command searches through module names and their

descriptions to find relevant results, as shown below:

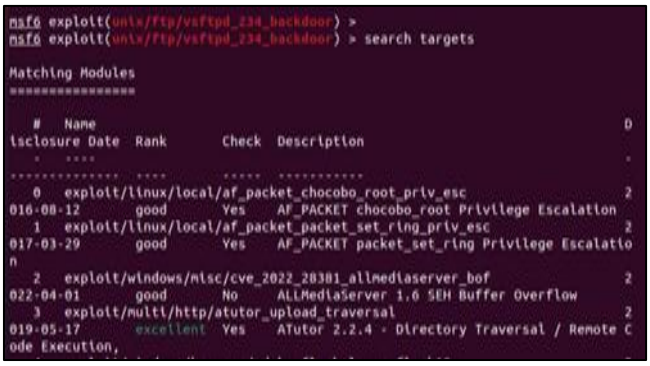

Fig 15: Search command

msf > search cve:2021

This command lists all exploits associated with CVE identifiers, including an auxiliary scanner module for the recent Fortinet firewall SSH backdoor vulnerability.:

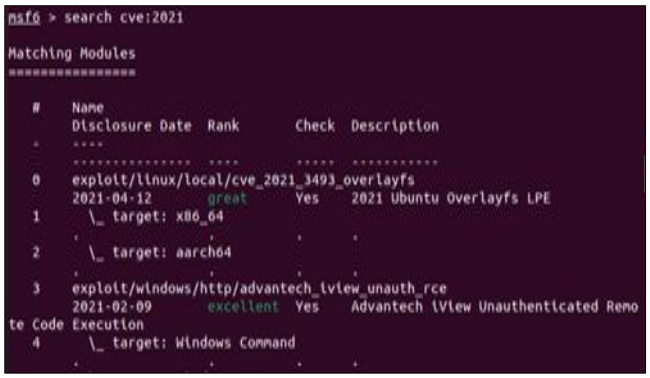

Fig 16: search cve:2021

Show options

When case related in the context of a specific exploit, the show options command displays the configurable parameters related to that module. For example, let's examine the available settings for the sitecore_xp_cve_42237 exploit using the following command.:

msf > Use exploit/windows/http/sitecore_xp_cve_2021_42237

Followed by the show options command:

msf > show options

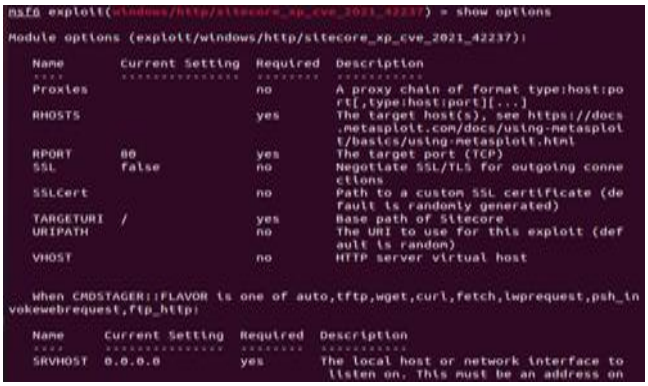

Fig 17: show options

The sitecore_xp_cve_2021_42237 exploit includes eight configurable options, but only one—RHOST—is mandatory. The full list of options includes: Proxies, RHOST (Required), RPORT, SSL, SSLCert, TARGETURI, URIPATH, and VHOST.

It's important to note that the show options command also displays the currently selected target below the module options. By default, the target is set to 0, which corresponds to a Windows platform for this specific exploit.

To modify any of these settings, use the set command followed by the option name and the desired value. For instance, to update the RHOST to 10.4.4.51, enter the following command: set RHOST 10.4.4.51

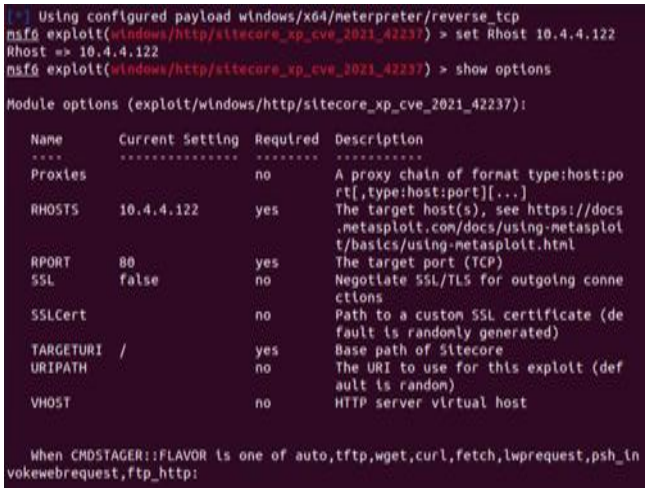

Fig 18: Use the set command followed by the option name and the new value

Show payloads

By executing the show payloads command, msfconsole will display all payloads that are compatible with the selected exploit. For example, when using the Flash Player exploit, it will list numerous suitable payload options:

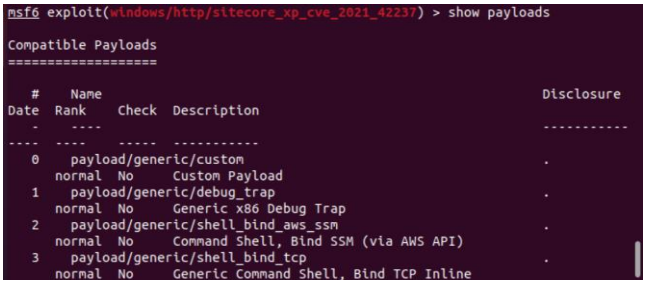

Fig 19: Show payloads

To select a specific payload, use the set command followed by the payload's name:

Set payload linux/x86/exec

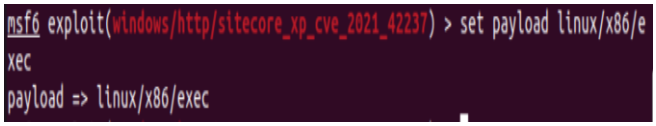

Fig 20: Set payloads

Show encoders

Running the show encoders command will list all encoders that are compatible with the chosen payload. Encoders help to bypass basic IDS/IPS detection by altering the payload's

byte patterns.

Fig 21: Show encoders

To apply an encoder, use the set command followed by the encoder's name.
As an initial step, you can perform a network scan with the following nmap command to detect open services on the target IP:
nmap –Pn –sV 10.4.4.51

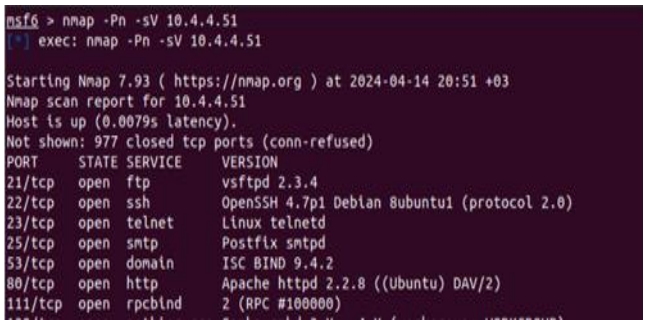

Fig 22: network scan using nmap

Exploiting VSFTPD 2.3.4
After successfully exploiting the service on port 21, the next step is to target the specific version of the FTP service. To do this, we'll look for an exploit related to VSFTPD version 2.3.4 by running the command search vsftpd
search vsftpd

Fig 23: Exploiting VSFTPD 2.3.4

First, let us execute the Metasploit framework to utilize the exploit which we have already noted. This module is focused on the malicious backdoor which was added to the VSFTPD download package. From the most recent intel, this backdoor was there in the vsftpd-2.3.4.tar.gz bundle from June 30 to July 1 in 2011, it was taken off shortly thereafter on 3rd July 2011.

msf > use exploit/unix/ftp/vsftpd_234_backdoor
msf exploit (unix/ftp/vsftpd_234_backdoor) > set rhost 192.168.1.103
msf exploit (unix/ftp/vsftpd_234_backdoor) > exploit

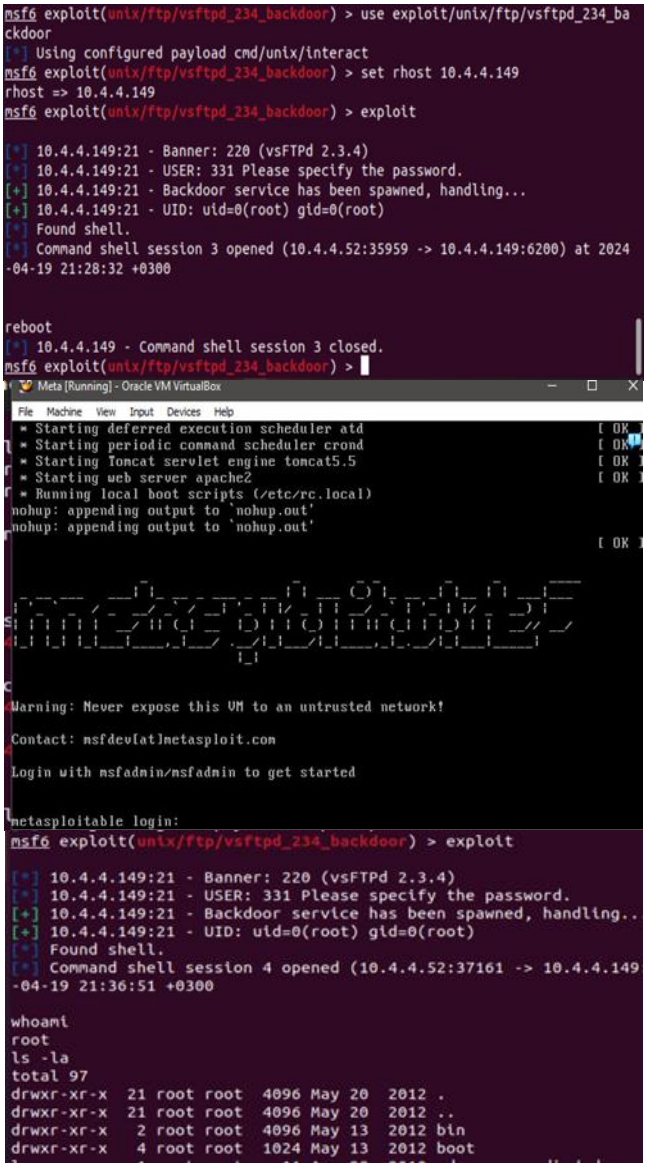

Fig 24: exploit VSFTPD

Exploiting Port 8080 (Java)
This module leverages the RMI Registry and RMI Activation services' weak points by exploiting their class loading configuration that accepts remote HTTP URLs. Because it invokes a method in the RMI Distributed Garbage Collector, which is reachable through every RMI endpoint, this can be executed on rmiregistry and rmid, as well as numerous other custom RMI endpoints. It does not apply, however, to Java Management Extension (JMX) ports as those do not permit remote class instantiation unless there exists another RMI endpoint in the same Java process. It is critical to note that calls to RMI methods do not and cannot be authenticated.
For this illustration, we utilize the Remote Method Invocation exploit against the service hosted on port 8080's Java container. The methodology is simple – pick the exploit, the IP address of the target, and proceed.

```
msf > use exploit/multi/misc/java_rmi_server
msf exploit(multi/misc/java_rmi_server) > set rhost 192.168.1.103
msf exploit(multi/misc/java_rmi_server) > exploit
```

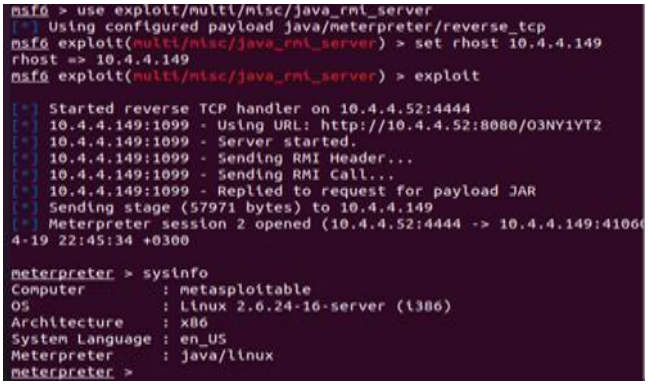

Fig 25: Exploiting Port 8080 (java)

Access Port 2121 (ProFTPD)
We will establish a connection to the target machine via Telnet on port 2121, using the default login credentials for Metasploitable 2.
telnet 10.4.4.149

Fig 26: Access Port 2121 (ProFTPD)

Table 2 table illustrates the step-by-step execution and resulting output of a Metasploit exploit targeting the well-known vulnerability in vsFTPd 2.3.4. The command use exploit/unix/ftp/vsftpd_234_backdoor loads the appropriate exploit module, followed by set rhost to specify the target IP. Executing the exploit command initiates a reverse TCP shell connection to the attacker's machine. Upon successful exploitation, the target system returns a root shell session, as confirmed by the id command. This validates the presence of a critical vulnerability that enables full system compromise.

Table 2: Metasploit Exploit Command Results (vsftpd_234_backdoor)

Command	Output / Result
use exploit/unix/ftp/vsftpd_234_backdoor	Using configured payload cmd/unix/interact
set rhost 192.168.1.103	rhost => 192.168.1.103
exploit	- Reverse TCP handler started on 192.168.1.100:4444- Target FTP Banner: 220 (vsFTPd 2.3.4)- Sending backdoor payload- Shell session opened on target IP and port
Post-exploit command id	uid=0(root) gid=0(root) groups=0(root) (root shell obtained)

Table 3 summarizes the open ports and running services on the target host (192.168.1.103), as discovered using Nmap's version detection. It identifies services such as vsFTPd, OpenSSH, and Apache HTTP, highlighting the FTP service (vsFTPd 2.3.4) as a known vulnerable entry point. This scan is critical for reconnaissance, helping security analysts or penetration testers identify which services might be exploitable.

Table 3: Nmap Scan Results (nmap -Pn -sV 192.168.1.103)

Port	Protocol	State	Service	Version	Extra Info
21/tcp	tcp	open	ftp	vsFTPd 2.3.4	Vulnerable FTP server
22/tcp	tcp	open	ssh	OpenSSH 7.6p1 Ubuntu 4ubuntu0.3	Protocol 2.0
80/tcp	tcp	open	http	Apache httpd 2.4.29	Ubuntu Linux server
111/tcp	tcp	open	rpcbind	2 (RPC #100000)	RPC port mapper service
139/tcp	tcp	open	netbios-ssn	Samba smbd 3.X - 4.X	SMB file sharing
445/tcp	tcp	open	microsoft-ds	Samba smbd 4.3.11	SMB over TCP
3306/tcp	tcp	closed	mysql	/	/

Table 4 presents the output of Nmap's OS fingerprinting feature, which attempts to determine the target's operating system based on TCP/IP stack characteristics. The scan indicates that the system is likely running a Linux 3.x kernel, possibly Ubuntu 16.04 LTS. Knowing the underlying OS is essential for tailoring the exploit strategy.

Table 4: OS Detection Summary

OS Guess	Accuracy	Details
Linux 3.x kernel	95%	Kernel 3.2 - 3.19
Ubuntu Linux 16.04 LTS	90%	Possible distribution
Device type	General purpose	Likely a server or VM

Table 5 and Fig 27 details the scan's performance metrics, including timeout values, probe parallelism, and overall scan duration. Such data is useful for understanding how fast the scan was conducted and if adjustments are needed in high-latency or stealth environments.

Table 5: Nmap Timing and Performance

Timing Parameter	Value	Description
Ping scan timeout	1000 ms	Time to wait for ping response
Connect scan timing	100 ms	Delay between TCP connect scans
Parallelism	10	Number of concurrent probes
Total scan time	45 seconds	Approximate duration for full scan

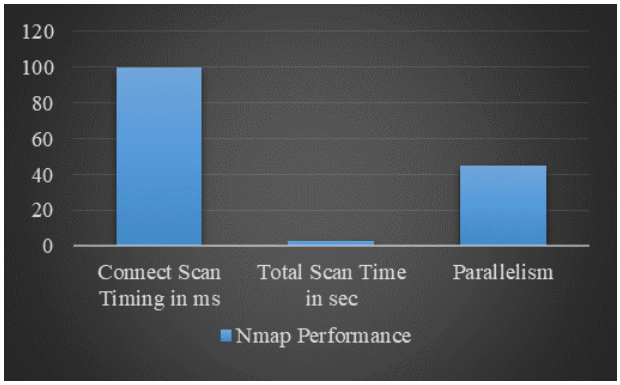

Fig 27: Nmap performance.

Table 6 reflects Nmap's ability to detect potential network defenses, such as firewalls or intrusion detection/prevention systems (IDS/IPS). A "Yes" under firewall detection indicates that some ports may be filtered, and latency anomalies suggest the possible presence of IDS/IPS. This insight is crucial for determining the level of difficulty an attacker may face when attempting to exploit the system.

Table 6: Nmap Firewall/Filter Detection

Feature	Status	Details
Firewall detected	Yes	TCP ports filtered
IDS/IPS detection	Possible	Some ports showed latency anomalies
Packet loss	3%	During scan

5. Conclusions

This walkthrough demonstrated how to leverage Metasploit Framework to identify and exploit vulnerabilities within a target system, specifically focusing on services such as FTP, Java RMI, and Telnet running on designated ports. By utilizing appropriate exploits, payloads, and encoders, it is possible to gain unauthorized access or execute code remotely, highlighting the importance of maintaining updated security configurations and monitoring for known vulnerabilities. Tools like Metasploit provide powerful capabilities for penetration testing and security assessments, emphasizing the need for defenders to stay informed and proactively secure their systems against such attack vectors.

6. References

1. Hussain SM, Tummalapalli SRK, Chakravarthy ASN. Cyber Security Education: Enhancing Cyber Security Capabilities, Navigating Trends and Challenges in a Dynamic Landscape. Adv Cyber Secur Digit Forensics. 2024;9-33.
2. Tahmasebi M. Beyond defense: Proactive approaches to disaster recovery and threat intelligence in modern enterprises. J Inf Secur. 2024;15(2):106-33.
3. Safitra MF, Lubis M, Fakhrurroja H. Counterattacking cyber threats: A framework for the future of cybersecurity. Sustainability. 2023;15(18):13369.
4. Goni A, Jahangir MUF, Chowdhury RR. A study on cyber security: Analyzing current threats, navigating complexities, and implementing prevention strategies. Int J Res Sci Innov. 2024;10(12):507-22.
5. Judijanto L, Hindarto D, Wahjono SI. Edge of enterprise architecture in addressing cyber security threats and business risks. Int J Softw Eng Comput Sci (IJSECS). 2023;3(3):386-96.
6. Xavier G, Novo C, Morla R. Tweaking Metasploit to Evade Encrypted C2 Traffic Detection [Internet]. 2022 [cited 2025 Jun 13]. Available from: https://arxiv.org/abs/2209.00943
7. Schwartz J, Kurniawati H. Autonomous penetration testing using reinforcement learning [Internet]. 2019 [cited 2025 Jun 13]. Available from: https://arxiv.org/abs/1905.05965
8. Pham VH, Do Hoang H, Trung PT, Quoc VD, To TN, Duy PT. Raiju: Reinforcement learning-guided post-exploitation for automating security assessment of network systems. Comput Netw. 2024;253:110706.
9. Deng G, Liu Y, Mayoral-Vilches V, Liu P, Li Y, Xu Y, *et al*. Pentestgpt: An llm-empowered automatic penetration testing tool [Internet]. 2023 [cited 2025 Jun 13]. Available from: https://arxiv.org/abs/2308.06782
10. Singh M, Kumar S, Garg T, Pandey N. Penetration testing on metasploitable 2. Int J Eng Comput Sci. 2020;9(05):25014-22.