Scanning and Exploitation of Vulnerabilities using Nessus & Metasploit

Khalil Mellouk

December 24, 2024

Abstract

The project focuses on conducting a comprehensive security assessment using two widely recognized cybersecurity tools: Nessus and Metasploit. The goal is to identify and exploit vulnerabilities within a deliberately vulnerable virtual machine, Metasploitable, which serves as the target environment for testing.

In the first phase, Nessus is used to perform an in-depth vulnerability scan on the target system. This process involves identifying software flaws, configuration weaknesses, and network vulnerabilities. The generated report is thoroughly analyzed to pinpoint exploitable vulnerabilities, categorized by their severity and potential impact.

In the second phase, Metasploit Framework is employed to exploit at least three identified vulnerabilities, varying in severity. Each exploitation scenario demonstrates the attack vector, execution process, and resulting system compromise. The tests cover different services, including network protocols, application layers, and operating system weaknesses, providing insights into the risks posed by each vulnerability.

The project not only highlights the capabilities of Nessus and Metasploit as essential tools in cybersecurity but also emphasizes the importance of proactive vulnerability management and the implementation of effective mitigation strategies. The findings underscore the need for regular security assessments to safeguard systems from potential threats.

Contents

Abstract **1**

List of Figures **5**

List of abbreviations and acronyms **6**

Introduction **7**

1 Vulnerabilities **8**

1.1 Introduction to Vulnerabilities 8

1.2 Types of Common Vulnerabilities 8

1.2.1 Software Vulnerabilities 8

1.2.2 Configuration Vulnerabilities 9

1.2.3 Network Vulnerabilities 9

1.3 Impacts of Vulnerabilities 9

2 Vulnerability Scanning **10**

2.1 Introduction to Vulnerability Scanning 10

2.2 How Vulnerability Scanning Works 10

2.3 Vulnerability Scan Types 10

2.4 Advantages and Limitations 11

3 Vulnerability Scanning with Nessus **12**

3.1 Introducing Nessus 12

3.1.1 What is Nessus? 12

3.1.2 Key features of Nessus 13

3.1.3 Benefits of Nessus Vulnerability Scanner 14

3.1.4 Installing Nessus on Kali Linux 14

3.1.5 Start the Nessus Service 18

3.2 Nessus Scan Vulnerability on Metasploitable 2 19

3.2.1 Metasploitable 2 19

3.2.2 Downloading and Setting Up Metasploitable 2 19
3.2.3 Powering on Metasploitable 2 20
3.2.4 Logging in to Metasploitable 2 20
3.2.5 Identifying Metasploitable 2's IP Address 20
3.2.6 Scanning for Vulnerabilities 21
3.2.7 Some Detected Vulnerabilities 25

4 Exploiting Vulnerabilities Using Metasploit 26
4.1 Introduction to Metasploit 26
4.1.1 The History of Metasploit 26
4.1.2 Uses and Advantages of Metasploit 26
4.1.3 Installing the Metasploit Framework on Kali Linux 27
4.2 Attack Scenarios 29
4.2.1 Critical Vulnerabilities 29
4.2.1.1 Velnurabilities in UnrealIRCd Backdoor (Port 6667) . . . 29
4.2.1.2 Vulnerabilities in the VNC Service (Port 5900) 33
4.2.1.3 Vulnerabilities in Apache Tomcat (Port 8180) 37
4.2.2 High Vulnerabilities 41
4.2.2.1 Vulnerabilities in the MySQL Service (Port 3306) 41
4.2.2.2 Vulnerabilities in HTTP Services (Port 80) 44
4.2.3 Medium Vulnerabilities 50
4.2.3.1 Vulnerabilities in PostgreSQL (Port 5432) 50
4.2.4 Low Vulnerabilities 53
4.2.4.1 Vulnerabilities in Java Rmi (Ports 1099 49104) 53

Conclusion 56

Webography & Bibliography 57

List of Figures

3.1 Nessus Logo 12
3.2 Nessus génère des scores VPR. 13
3.3 Tenable's Nessus platform 13
3.4 Kali Linux Logo 14
3.5 Downloading the Nessus package for Debian 15
3.6 Downloaded Nessus file 15
3.7 Accessing Nessus via `https://kali:8834/` 16
3.8 Selecting Nessus Essentials 17
3.9 Downloading Nessus plugins 17
3.10 Nessus Essentials Login Page 18
3.11 Nessus main page 19
3.12 Logging in to Metasploitable 2 20
3.13 Identifying Metasploitable 2's IP Address 21
3.14 New Scan 21
3.15 Advanced Scan 22
3.16 Target 22
3.17 Lunch Scan 23
3.18 Scan Details : Hosts 23
3.19 Scan Details : Vulnerabilities 24
3.20 Scan Details : Remidiations 24
3.21 Save Report 25

4.1 Metasploit Logo 26
4.2 Metasploit Uses and Benefits 27
4.3 Metasploit architecture 27
4.4 Install Metasploit from Kali Repositories 28
4.5 Launching Metasploit 29
4.6 Search for Unreal 30
4.7 Set RHOSTS Unreal 31
4.8 Set Payload Unreal 31

4.9 Exploiting Unreal 32
4.10 Port 5900 is open 34
4.11 Launch Metasploit 35
4.12 VNC authentication test module 35
4.13 Configure the module 36
4.14 Attack execution 36
4.15 Open 192.168.93.129:8180 38
4.16 Search tomcat 5.5 38
4.17 Use tomcat_mgr_upload 39
4.18 Set RHOSTS RPORT .. Tomcat 39
4.19 Exploiting Tomcat on Port 8180 40
4.20 Scan opened port 3306 42
4.21 Scan script mysql brute 43
4.22 MySQL Empty Password Script 43
4.23 Nmap scan showing HTTP on port 80. 45
4.24 Accessing the server via the IP address. 45
4.25 Mutillidae password page. 46
4.26 PHP information page on the server. 46
4.27 Mutillidae robots.txt file. 47
4.28 Mutillidae configuration page. 47
4.29 HTTP version detection using Metasploit. 48
4.30 Setting RHOSTS for the Metasploit scanner. 48
4.31 Searchsploit results for Apache 2.2.8 and PHP vulnerabilities. 49
4.32 Attempted exploitation of the HTTP server. 49
4.33 Use Postgress Login 51
4.34 Set USERNAME USER_AS_PASS RHOSTS Postgress 51
4.35 Exploiting Postgress 52
4.36 Search JAVA RMI 53
4.37 Using java_rmi_server 54
4.38 Exploiting Java RMI 54

List of abbreviations and acronyms

VM	*Virtual Machine*
VNC	*Virtual Network Computing*
MySQL	*My Structured Query Language*
HTTP	*Hypertext Transfer Protocol*
R	*Remote Access*
NFS	*Network File System*
IP	*Internet Protocol*
OS	*Operating System*
SQL	*Structured Query Language*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
SSH	*Secure Shell*
DOS	*Denial of Service*
IDS	*Intrusion Detection System*
IPS	*Intrusion Prevention System*
CVE	*Common Vulnerabilities and Exposures*
SYN	*Synchronize (TCP synchronization segment)*
RCE	*Remote Code Execution*
XSS	*Cross-Site Scripting*
CSRF	*Cross-Site Request Forgery*

Introduction

The security of computer systems is a crucial issue in the current technological context, where cyberattacks are increasing and becoming more sophisticated. In this context, assessing the vulnerability of IT infrastructures is an essential step to prevent possible intrusions and ensure data integrity. This project aims to explore two major steps in the assessment and exploitation of vulnerabilities: detection by vulnerability scanning and targeted exploitation of these vulnerabilities.

The first part of this project will consist of performing a vulnerability scan using Nessus software. Nessus is a reference tool in the field of security analysis, capable of detecting a wide range of vulnerabilities, ranging from weak configurations to critical system vulnerabilities. This analysis will make it possible to draw up a detailed report of the vulnerabilities present on the target machine.

The second part will focus on the exploitation of identified vulnerabilities using Metasploit, a comprehensive framework for exploiting security flaws. Metasploit is pre-installed on Kali Linux and allows you to run penetration tests to validate the presence and impact of vulnerabilities. Three vulnerability exploitation tests of different severity levels will be conducted to illustrate the diversity of exploitable flaws.

For this project, the Metasploitable 2.0 virtual machine will be used as a test platform. Metasploitable is a deliberately vulnerable machine, designed specifically for the practice and demonstration of security tools.

The goal of this project is to put vulnerability analysis and exploitation into practice in order to develop a better understanding of potential threats and methods for strengthening the security of computer systems.

Chapter 1

Vulnerabilities

1.1 Introduction to Vulnerabilities

Vulnerabilities are weaknesses or flaws in computer systems, applications, networks, or configurations that can be exploited by cyber attackers to compromise security. They can lead to data loss, unauthorized access, or other negative impacts to organizations. Understanding and managing these vulnerabilities is essential to strengthening overall security. To fully understand the scope of the risks, it is crucial to identify the types of vulnerabilities that exist and how they can affect systems.[1]

1.2 Types of Common Vulnerabilities

1.2.1 Software Vulnerabilities

Software vulnerabilities are flaws in the code of a program or application. They can allow an attacker to take control of a system or cause damage. Some examples include:

- **Buffer Overflow:** An attack where a program writes data beyond the allocated memory, which can cause unpredictable behavior or the execution of malicious code.
- **SQL Injection:** An injection technique that allows attackers to submit malicious SQL queries, thereby gaining access to sensitive data, bypassing authentication, or even compromising the entire database.
- **Cross-Site Scripting (XSS):** Allows the injection of malicious scripts into web pages to steal user sessions, modify displayed content, or redirect users to unsecured sites.

1.2.2 Configuration Vulnerabilities

Configuration errors are a significant and often overlooked risk. These include:

- **Default Passwords:** Leaving default credentials in place after installation exposes systems to unauthorized access.
- **Excessive Permissions:** Assigning permissions that are too broad can allow unintended access to critical resources.
- **Inadequate Firewall Configurations:** Misconfigured firewall rules can expose vulnerable ports or services.

1.2.3 Network Vulnerabilities

Network vulnerabilities compromise communications security and data integrity. Examples:

- **Denial-of-service (DoS) attacks:** Overwhelm network resources to make a service unavailable.
- **Man-in-the-Middle (MitM) attacks:** Allow for the interception and modification of communication between two parties.
- **Insecure protocols:** Using outdated protocols like Telnet or HTTP exposes sensitive data to theft.

1.3 Impacts of Vulnerabilities

The consequences of vulnerabilities can be devastating for businesses and individuals, including data loss, reputational damage, significant financial losses, and even legal penalties. Exploitation of vulnerabilities can also facilitate targeted attacks such as ransomware, crippling operations until a ransom is paid.

Chapter 2

Vulnerability Scanning

2.1 Introduction to Vulnerability Scanning

Vulnerability scanning is an automated process that scans systems, applications, and networks to identify potential security vulnerabilities. It is a crucial part of cybersecurity strategies that aim to prevent cyberattacks by detecting and remediating weaknesses before they are exploited. Through these scans, security teams can better understand their attack surface and prioritize corrective actions.[2]

2.2 How Vulnerability Scanning Works

The vulnerability scanning process consists of several steps:

- **Information Gathering:** Identifying assets to scan (IP addresses, applications, critical systems).
- **Automated Scanning:** Using specialized software to scan for vulnerabilities by comparing them to a database of known vulnerabilities.
- **Results Report:** Identified vulnerabilities are categorized by criticality (low, moderate, high, critical) for efficient prioritization.
- **Manual Validation:** Some findings require manual review to eliminate false positives.

2.3 Vulnerability Scan Types

- **Network Scan:** Scans the network infrastructure for potential vulnerabilities.

- **Web Application Scan:** Checks for vulnerabilities specific to web applications, such as SQL injection and XSS vulnerabilities.
- **Configuration Scan:** Checks configurations to ensure they follow security best practices.
- **Authenticated Scan:** Uses credentials to log into systems and provide a deeper view of internal vulnerabilities.

2.4 Advantages and Limitations

Vulnerability scans have several advantages, including early detection of vulnerabilities and continuous improvement of security. However, they also have limitations: the possibility of false positives and the reliance on the vulnerability database that must be regularly updated. Additionally, a scan cannot always identify unlisted vulnerabilities or those that require further contextual analysis.

Chapter 3

Vulnerability Scanning with Nessus

3.1 Introducing Nessus

3.1.1 What is Nessus?

Figure 3.1: Nessus Logo

Nessus is a computer security tool. It reports potential or proven weaknesses on tested machines. This includes, among others:

- services vulnerable to attacks allowing control of the machine, access to sensitive information (reading confidential files for example), denials of service...
- configuration errors (open mail relay for example)
- security patches not applied, whether or not the corrected flaws are exploitable in the tested configuration
- default passwords, some common passwords, and the absence of passwords on some system accounts. Nessus can also call the external program Hydra (de) to attack passwords using a dictionary.
- services deemed weak (e.g., replacing Telnet with SSH is suggested)
- denials of service against the TCP/IP stack.

3.1.2 Key features of Nessus

Vulnerability scores and categories

SCORE RANGE	SEVERITY CATEGORY
0.0	None
0.1–3.9	Low
4.0–6.9	Medium
7.0–8.9	High
9.0–10.0	Critical

Figure 3.2: Nessus génère des scores VPR.

Nessus est reconnu pour sa base de données de plugins étendue. Ces plugins sont dynamiquement et automatiquement intégrés à l'outil pour améliorer ses performances d'analyse et réduire le temps nécessaire à l'évaluation, à la détection et à la résolution des vulnérabilités. Ils peuvent également être personnalisés pour créer des contrôles spécifiques adaptés à l'écosystème applicatif d'une organisation.

Nessus includes a feature called Predictive Prioritization, which uses algorithms to rank vulnerabilities based on their severity to help IT teams determine the most urgent threats to address. Each vulnerability is assigned a Vulnerability Priority Rating (VPR), which uses a scale of 0 to 10, with 10 being the highest risk, to rate its severity: critical, high, medium, or low. IT teams can also use pre-built policies and templates to quickly find vulnerabilities and understand the threat landscape.

In parallel, the Nessus packet capture feature allows teams to quickly debug and resolve scanning issues. In this way, it minimizes disruptions and ensures continuous protection of the enterprise IT environment.

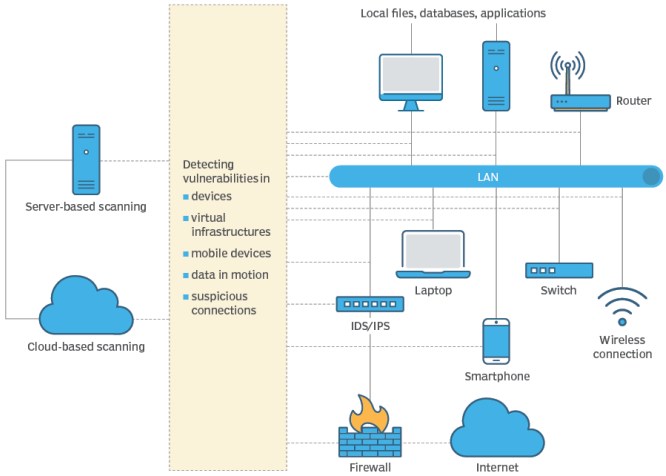


Figure 3.3: Tenable's Nessus platform

3.1.3 Benefits of Nessus Vulnerability Scanner

Nessus is a user-friendly vulnerability scanner that supports over 76,000 CVEs and provides access to a library of 185,000+ plugins, updated weekly by Tenable Research. It includes 450+ pre-configured templates for scanning IT assets, such as cloud infrastructures and virtualized environments. The platform offers simple remediation guidance, compliance checks, and low false positives (0.32 defects per 1 million scans), reducing alert fatigue. Nessus is portable, making it ideal for penetration testers and consultants. With updates within 24 hours of new vulnerability disclosures, it helps organizations proactively secure their systems against emerging threats.[5]

3.1.4 Installing Nessus on Kali Linux

1. **Kali Linux**

Figure 3.4: Kali Linux Logo

Kali Linux is a Debian-based GNU/Linux distribution released on March 13, 2013, and maintained by Offensive Security. Designed for penetration testing and ethical hacking, it integrates tools for network analysis, vulnerability assessment, and forensic investigation. With multiple pre-installed desktop environments (GNOME, KDE, XFCE, etc.), Kali Linux offers flexibility and convenience for security professionals.

2. **Downloading the Nessus package for Debian**

   - Go to the official Nessus website: `https://www.tenable.com/products/nessus/select-your-operating-system`.
   - In the download section, select the platform `Linux-Debian-amd64`.
   - Make sure to choose the version appropriate for your operating system.
   - After downloading, verify the integrity of the file using the provided checksum to ensure that it has not been tampered with.

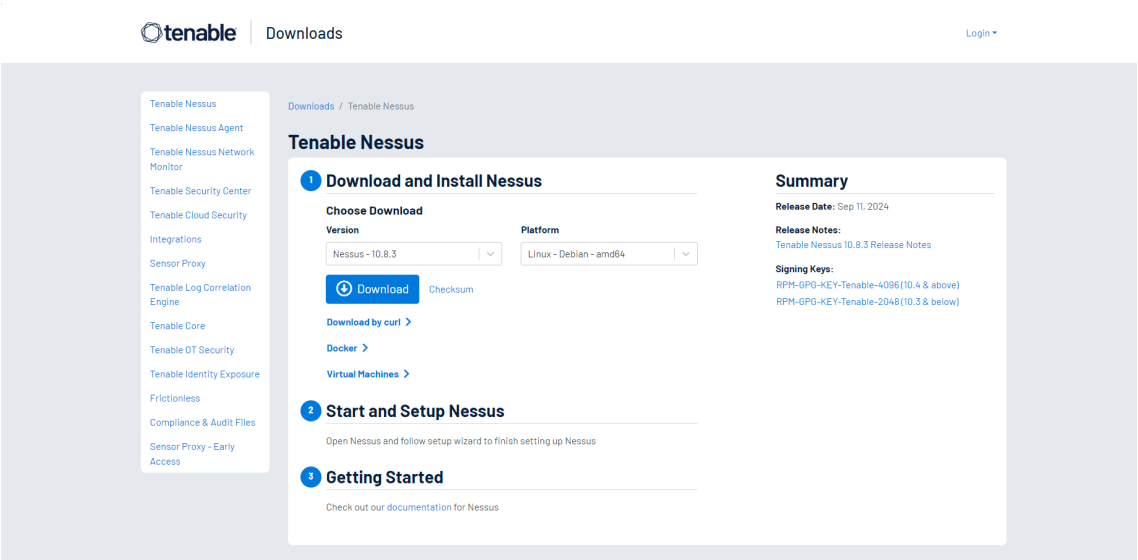


Figure 3.5: Downloading the Nessus package for Debian

3. **Navigation to the Nessus file directory**

   After completing the download, I launched my Linux terminal and navigated to the directory where the Nessus file was saved.

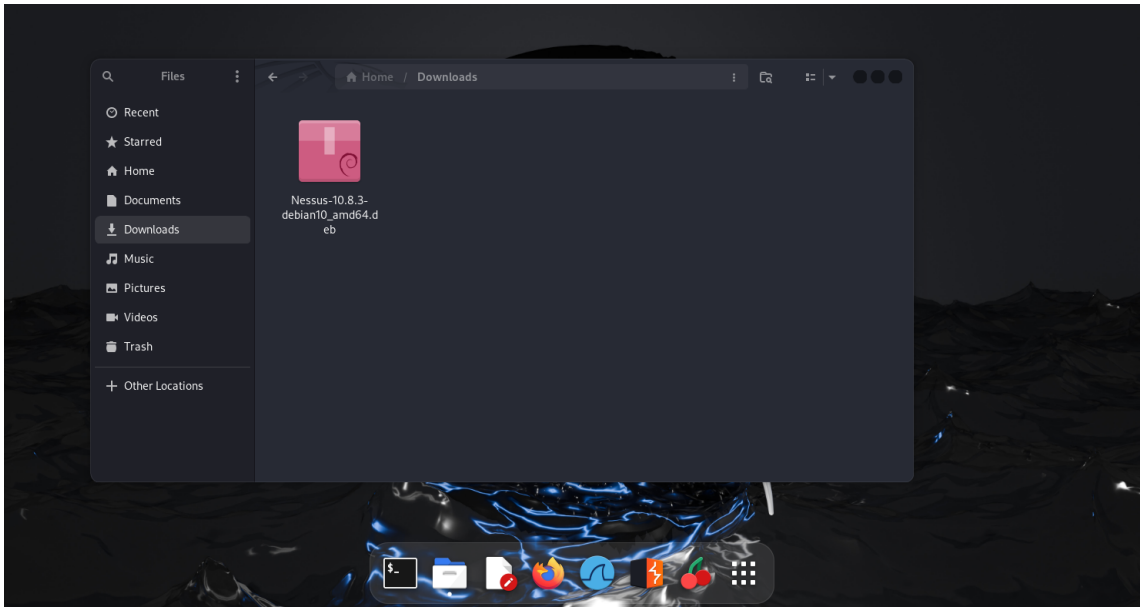


Figure 3.6: Downloaded Nessus file

4. **Installing Nessus**

   To install Nessus, I used the following command in the terminal:

```
dpkg -i Nessus-10.4.1-debian9_amd64.deb
```

5. **Starting the Nessus service**

   After installation, I started the Nessus service with the following command:

```
systemctl start nessusd
```

6. **Accessing the Nessus service via the browser**

   In my browser, I went to the following address: `https://kali:8834/`. A warning page was displayed.

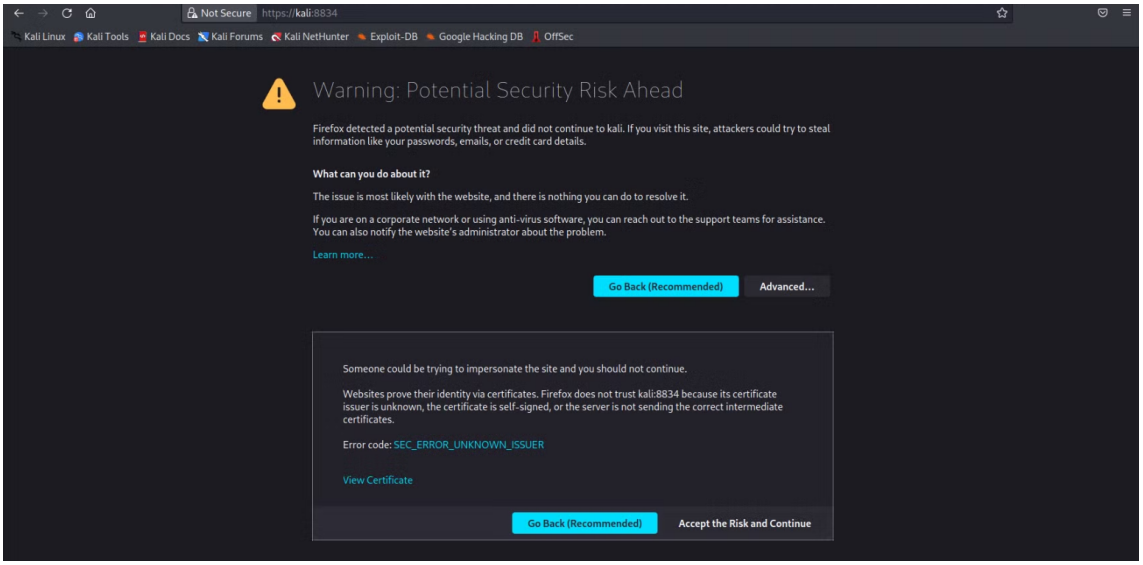


Figure 3.7: Accessing Nessus via `https://kali:8834/`

   I clicked on `Advanced`, then on `Accept Risk and Continue` to access the Nessus interface.

7. **Nessus Product Selection**

   I then selected the desired Nessus product. To use the free version, I clicked on `Nessus Essentials`.

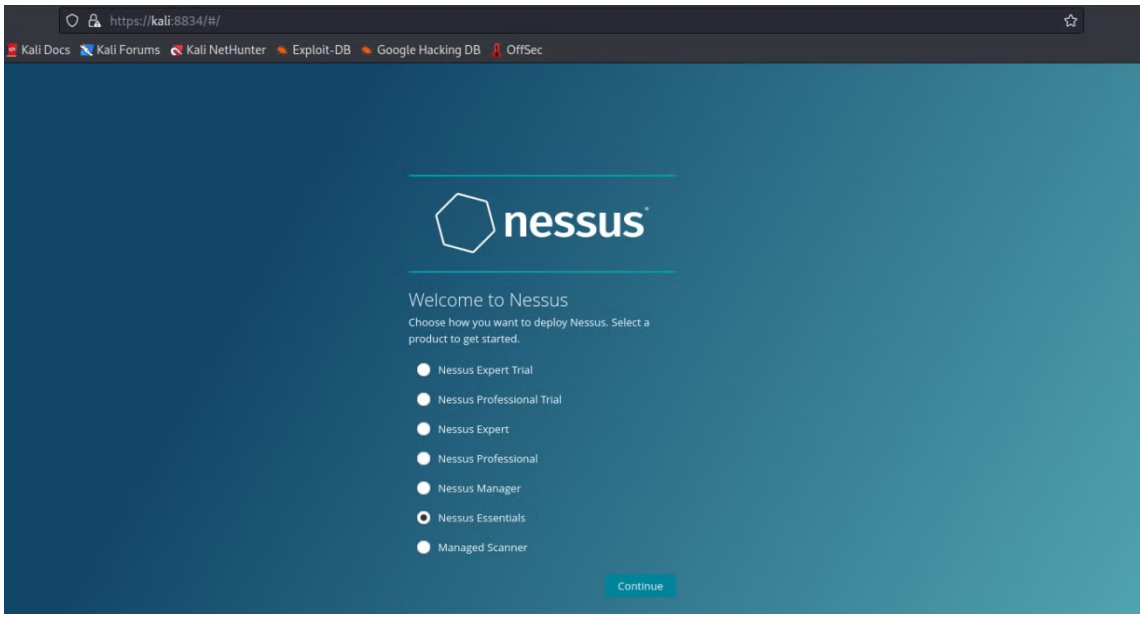


Figure 3.8: Selecting Nessus Essentials

8. **Activating and setting up Nessus**

   I entered my name and email address to receive an activation code. After pasting the code into the space provided, I chose a username and password to set up my account.

9. **Downloading the necessary plugins**

   Nessus then downloaded the plugins needed for its operation.

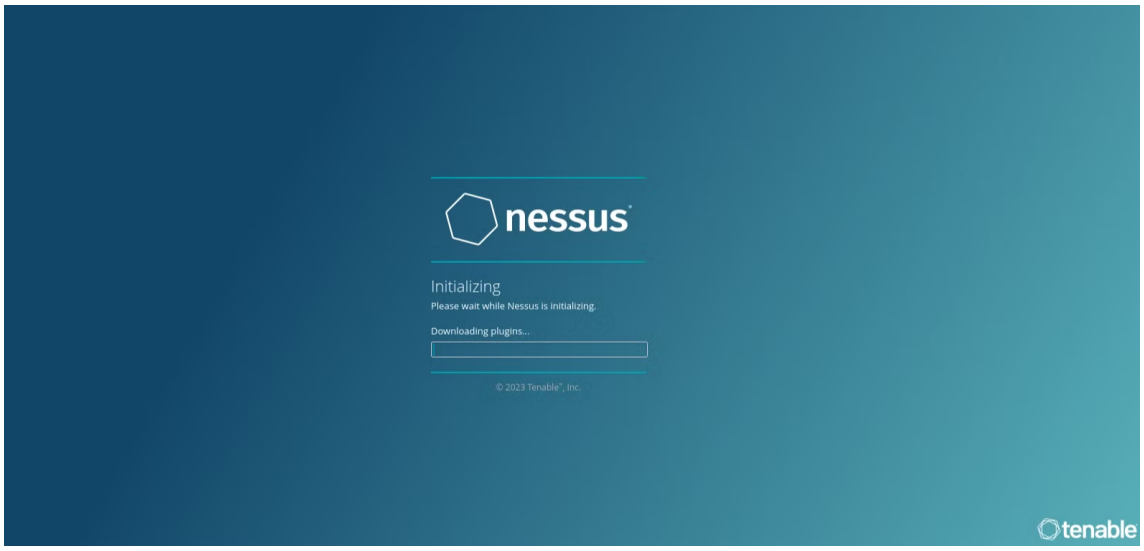


Figure 3.9: Downloading Nessus plugins

10. **Using Nessus**

Once the plugins were downloaded, the Nessus service was ready to be used to perform vulnerability scans and security tests.

3.1.5 Start the Nessus Service

1. **Starting the Nessus service**

   To start the Nessus service, I ran the following command in the terminal:

```
systemctl start nessusd
```

   This command starts the Nessus daemon in the background, making the service accessible via a browser.

2. **Accessing the Nessus Web Interface**

   Next, I opened a browser and entered the following URL to access the Nessus web interface:

```
https://kali:8834/
```

   Once logged in, I was redirected to the Nessus Essentials login or signup page.

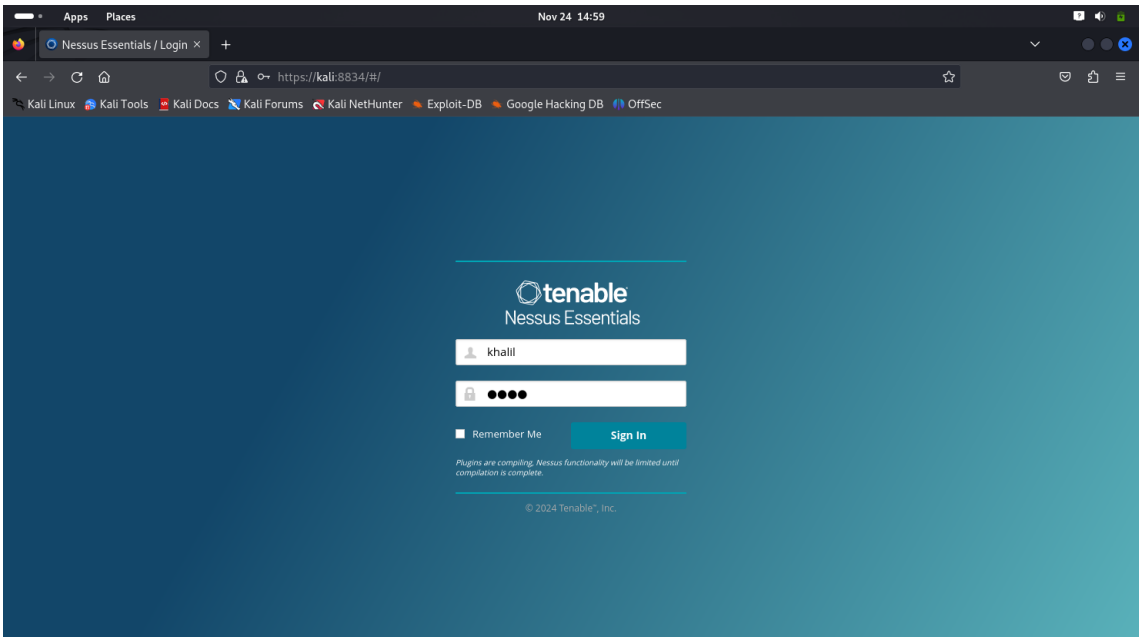


Figure 3.10: Nessus Essentials Login Page

3. **Nessus Main Page**

After completing the setup and login process, I accessed the Nessus main page. This page features a dashboard that allows you to manage vulnerability scans, view results, and configure security options.

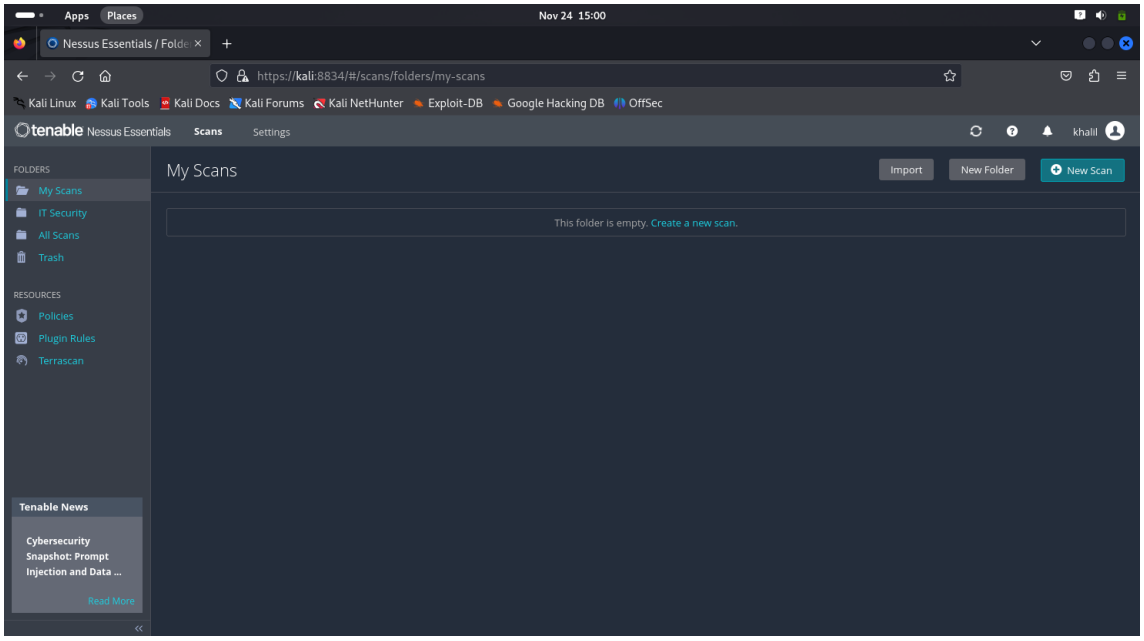


Figure 3.11: Nessus main page

3.2 Nessus Scan Vulnerability on Metasploitable 2

3.2.1 Metasploitable 2

A test environment provides a secure place to perform penetration testing and security research. For your test environment, you need a Metasploit instance that can access a vulnerable target. The following sections describe the requirements and instructions for setting up a vulnerable target.

3.2.2 Downloading and Setting Up Metasploitable 2

The easiest way to get a target machine is to use Metasploitable 2, which is an intentionally vulnerable Ubuntu Linux virtual machine that is designed for testing common vulnerabilities. This virtual machine (VM) is compatible with VMWare, VirtualBox, and other common virtualization platforms.[4]

Metasploitable 2 is available at:

`https://information.rapid7.com/metasploitable-download.html`

`https://sourceforge.net/projects/metasploitable/`

The compressed file is about 800 MB and can take a while to download over a slow connection. After you have downloaded the Metasploitable 2 file, you will need to unzip the file to see its contents.

3.2.3 Powering on Metasploitable 2

Once the VM is available on your desktop, open the device, and run it with VMWare Player. Alternatively, you can also use VMWare Workstation or VMWare Server.

3.2.4 Logging in to Metasploitable 2

The login for Metasploitable 2 is

```
msfadmin:msfadmin
```

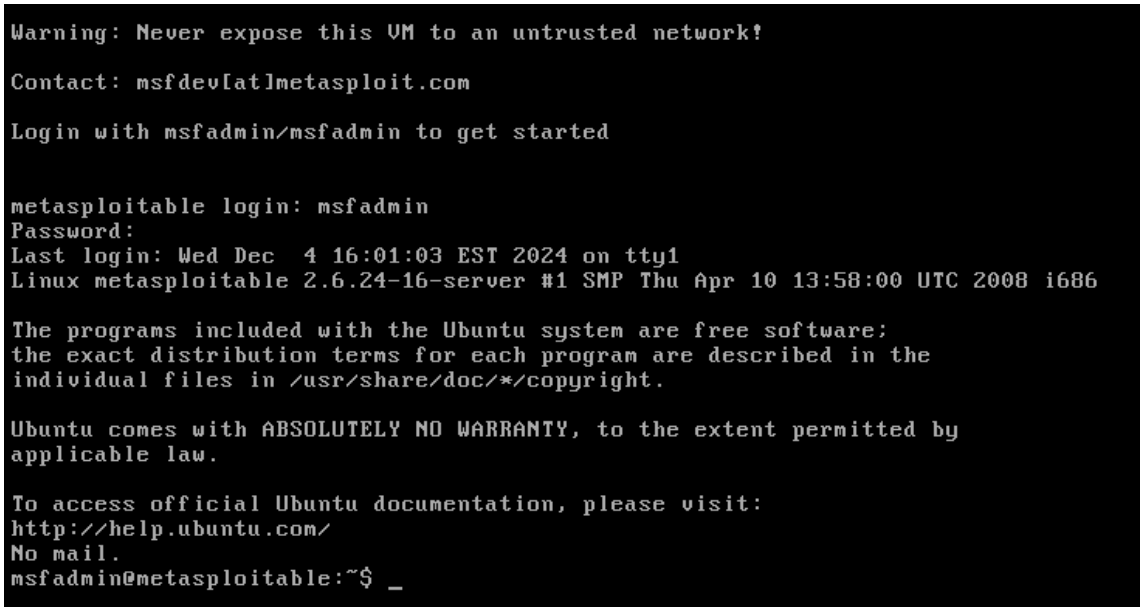
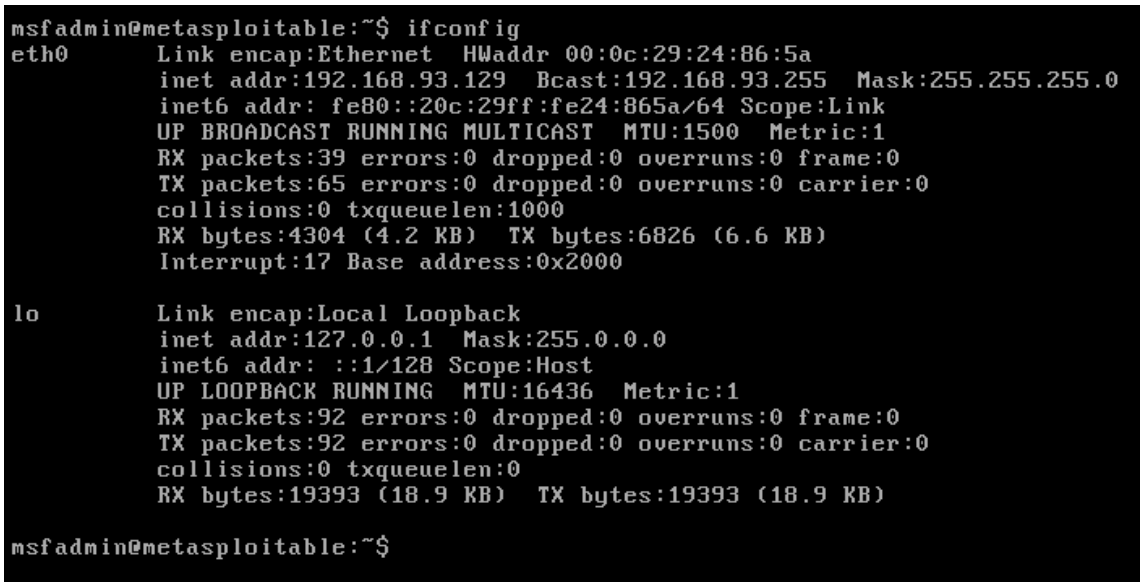


Figure 3.12: Logging in to Metasploitable 2

3.2.5 Identifying Metasploitable 2's IP Address

After you log in to Metasploitable 2, you can identify the IP address that has been assigned to the virtual machine. Just enter `ifconfig` at the prompt to see the details for the virtual machine.

```
msfadmin@metasploitable: $ ifconfig
```



```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:24:86:5a
          inet addr:192.168.93.129  Bcast:192.168.93.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe24:865a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:39 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4304 (4.2 KB)  TX bytes:6826 (6.6 KB)
          Interrupt:17 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:92 errors:0 dropped:0 overruns:0 frame:0
          TX packets:92 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19393 (18.9 KB)  TX bytes:19393 (18.9 KB)

msfadmin@metasploitable:~$
```

Figure 3.13: Identifying Metasploitable 2's IP Address

3.2.6 Scanning for Vulnerabilities

I know my Metasploitable IP address is ‘‘`192.168.93.129`‘‘. So, go back to the Nessus panel and enter the **New Scan** button.

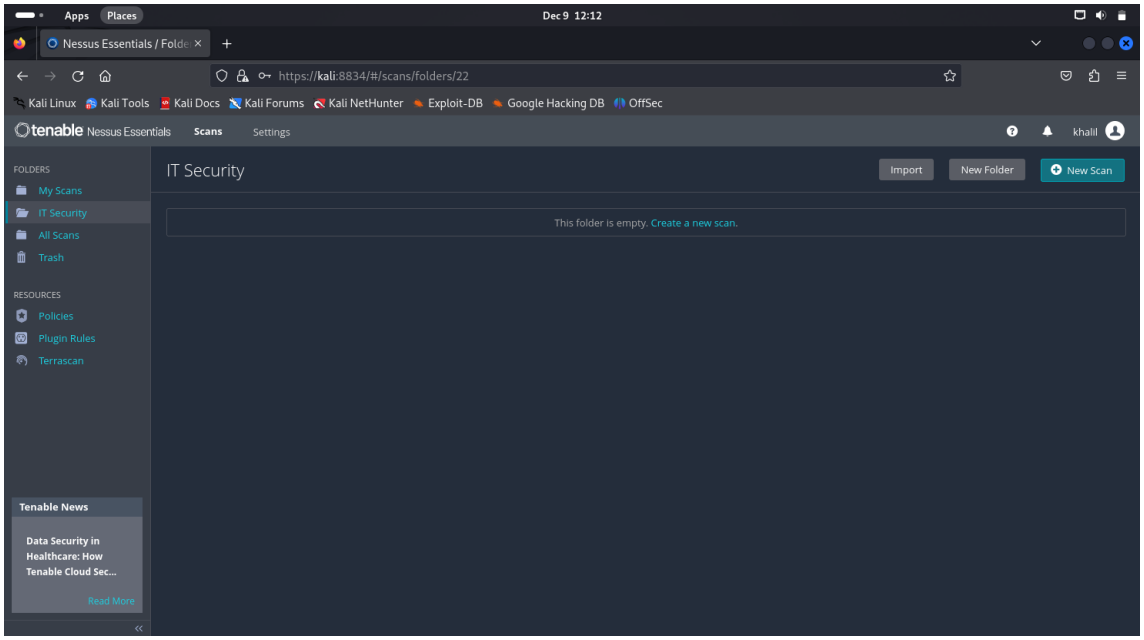


Figure 3.14: New Scan

You will see many scan options here. I have chosen Advanced Scan for this Machine Metasploitable 2.

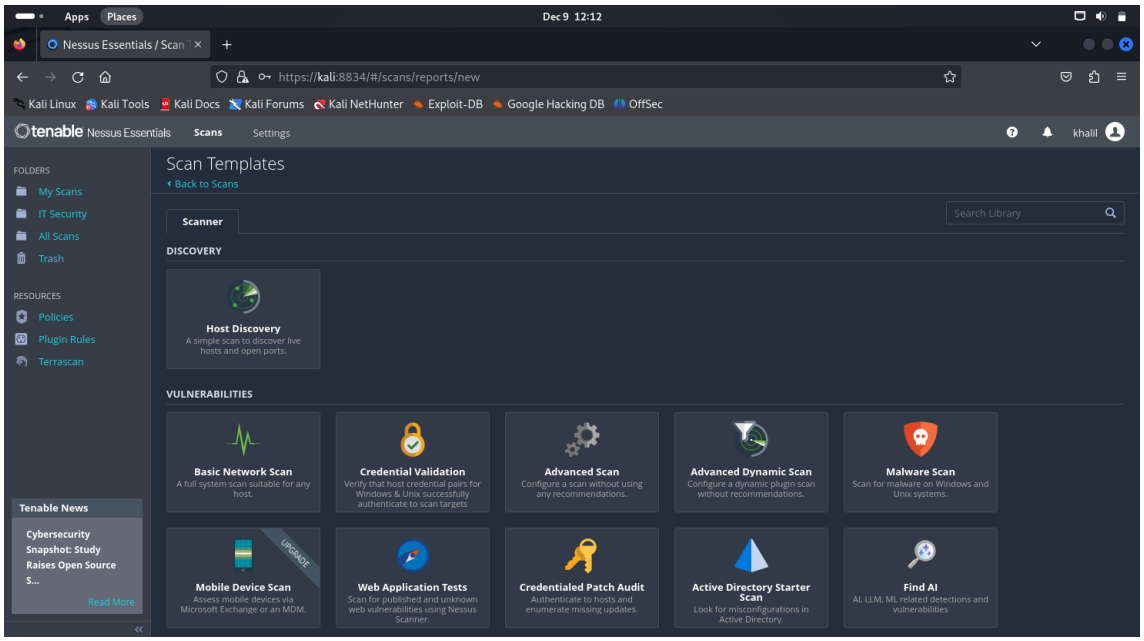


Figure 3.15: Advanced Scan

I applied the scan settings as follows. The important thing here is the **Targets section**, you need to type the **target machine's IP address** in that box.

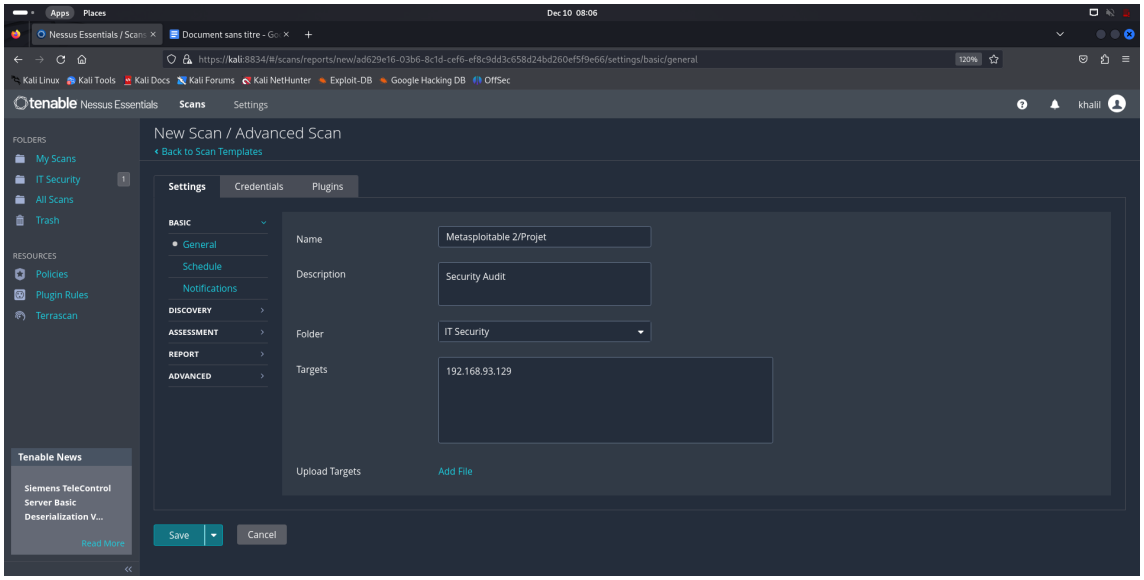


Figure 3.16: Target

Once you save your scan, you will see it in your scan lists. Here we need to click on the play ikon which means Launch. And now, the scan has started.

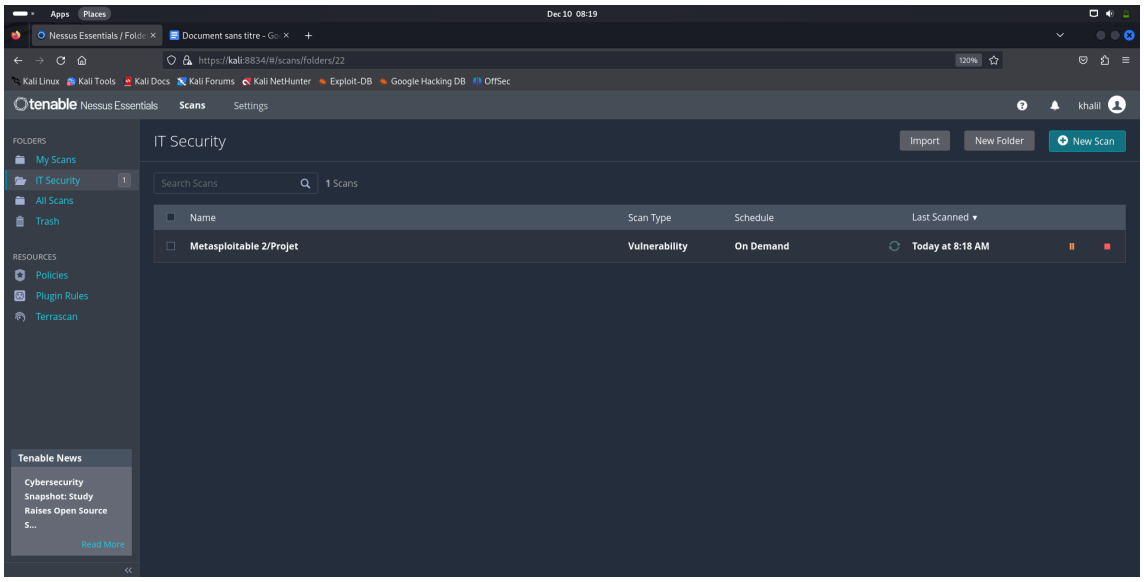


Figure 3.17: Lunch Scan

When you double click on your Scan, you can see this screen as follows. You will have know when your scan is finished at the right side below the Scan Details.

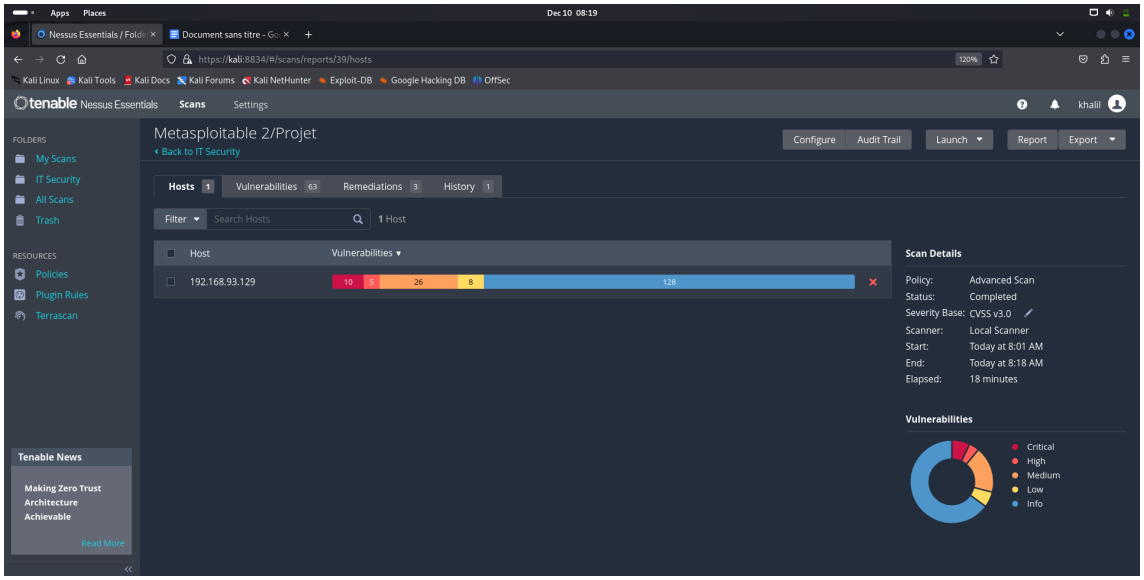


Figure 3.18: Scan Details : Hosts

Once the scan has completed, we can see all vulnerabilities in the Vulnerabilities section. It found a total of **63 security vulnerabilities** as you can see

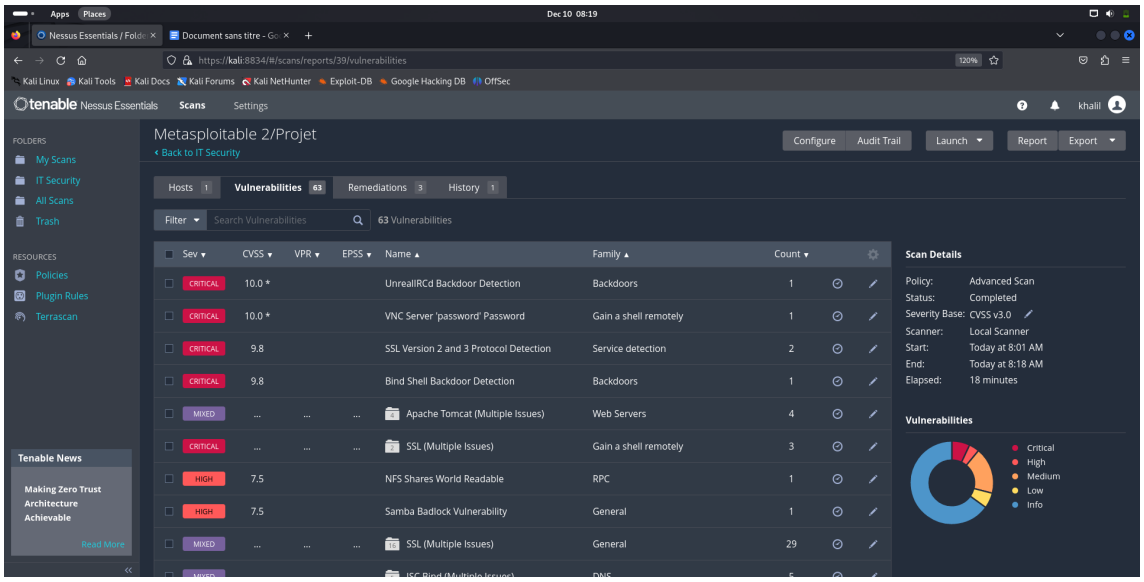


Figure 3.19: Scan Details : Vulnerabilities

The breakdown of the vulnerabilities is as follows:

Severity	Number of Vulnerabilities
Critical	10
High	5
Medium	36
Low	8
Informational	128

Table 3.1: Vulnerability Counts by Severity

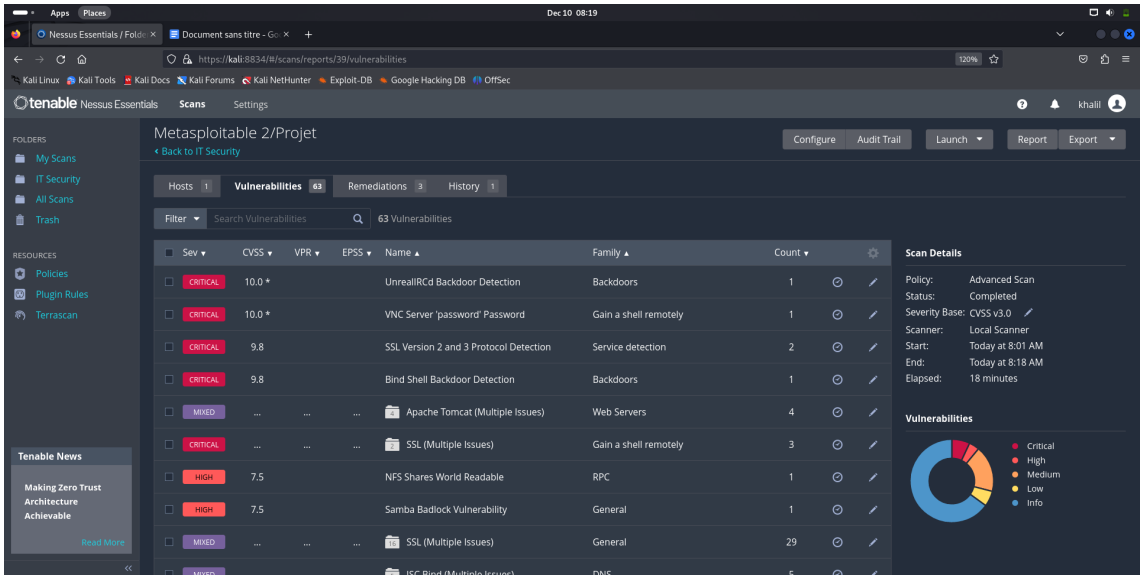


Figure 3.20: Scan Details : Remidiations

and Now export to generate a pdf report

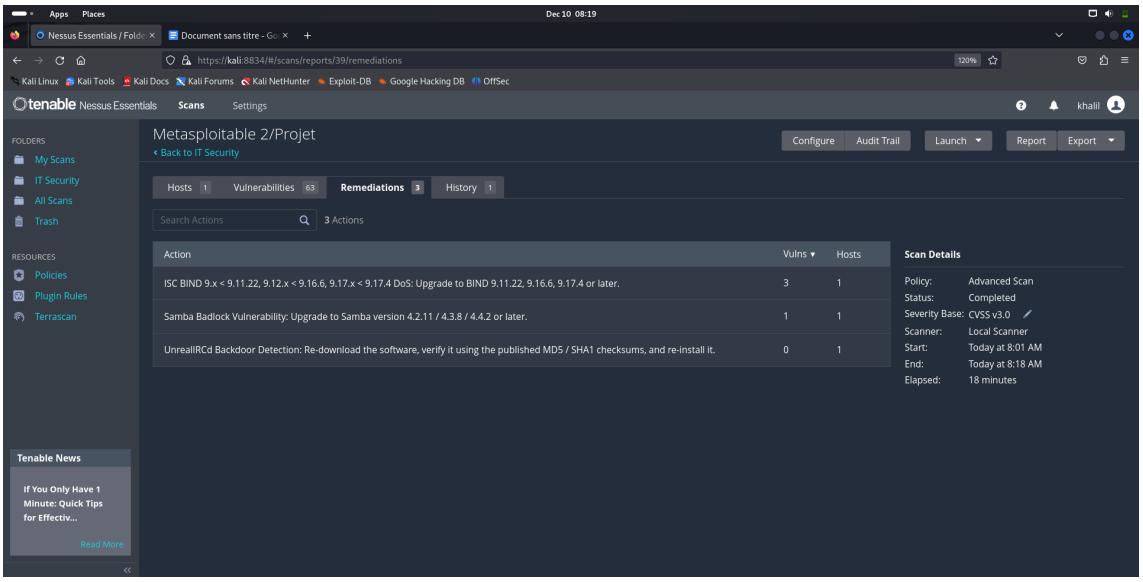


Figure 3.21: Save Report

3.2.7 Some Detected Vulnerabilities

ID	Description	Related CVE
134862 - Apache Tomcat AJP Connector Request Injection (Ghostcat)	This vulnerability affects the Apache JServ Protocol (AJP) connector in Tomcat. It allows an attacker to access sensitive files or execute injected requests, which may lead to data disclosure or server-side request forgery (SSRF) attacks.	CVE-2020-1938
171340 - Apache Tomcat SEoL ($\leq 5.5.x$)	Vulnerability affecting outdated versions (*End of Life, SEoL*) of Apache Tomcat, where security patches are no longer applied, leaving instances exposed to various threats.	None
61708 - VNC Server 'password' Password	This vulnerability occurs when the VNC server uses the default password `password`, allowing an unauthorized attacker to gain access to the system.	None
136769 - ISC BIND Service Downgrade / Reflected DoS	Vulnerability affecting the ISC BIND DNS server, allowing a downgrade attack (*service*) or a reflected denial of service (*Reflected DoS*) by exploiting malicious DNS queries.	None

Chapter 4

Exploiting Vulnerabilities Using Metasploit

4.1 Introduction to Metasploit

Figure 4.1: Metasploit Logo

4.1.1 The History of Metasploit

The Metasploit Project, launched in 2003 by H.D. Moore and later converted to Ruby, became a cornerstone of penetration testing.[4] Acquired by Rapid7 in 2009, it now serves as a versatile framework for exploit development, IDS signature creation, and evasion tools, with proprietary versions like Metasploit Pro and Express. Integrated into Kali Linux, it revolutionized penetration testing by automating probes and enabling remote testing, saving time and resources compared to the manual, localized methods previously required.[3]

4.1.2 Uses and Advantages of Metasploit

Metasploit simplifies penetration testing by enabling users to gather target information (via port scanning, OS fingerprinting, or vulnerability scanning), select an exploit, and deploy a payload to exploit system weaknesses. It includes tools like msfconsole, msfcli, the Armitage GUI, and a web interface for remote testing.

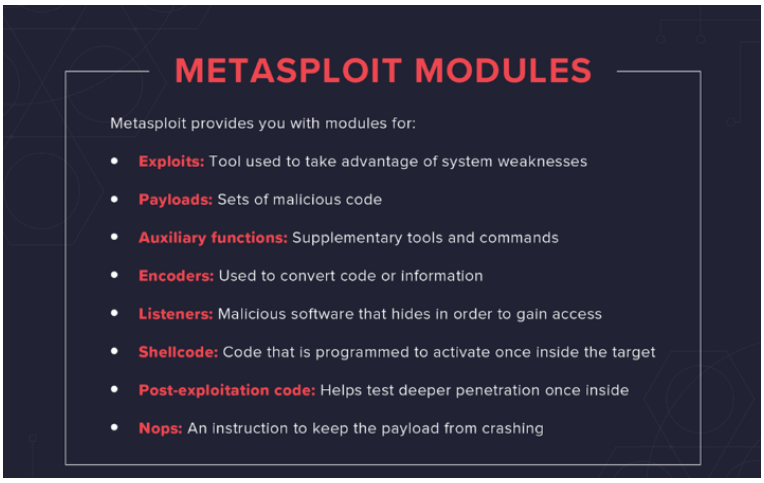


Figure 4.2: Metasploit Uses and Benefits

Both ethical and malicious hackers often use privacy tools like VPNs and dedicated VPS servers to mask their activities. Metasploit offers a range of features, including exploits, payloads, encoders, and post-exploitation tools.

Professionals can enhance their credentials by obtaining the Metasploit Pro Specialist Certification, which requires an 80% passing score in a two-hour open-book exam ($195). Recommended prerequisites include knowledge of Windows/Linux, network protocols, vulnerability management, and pen-testing basics. The certification boosts marketability for aspiring pen-testers and security analysts.

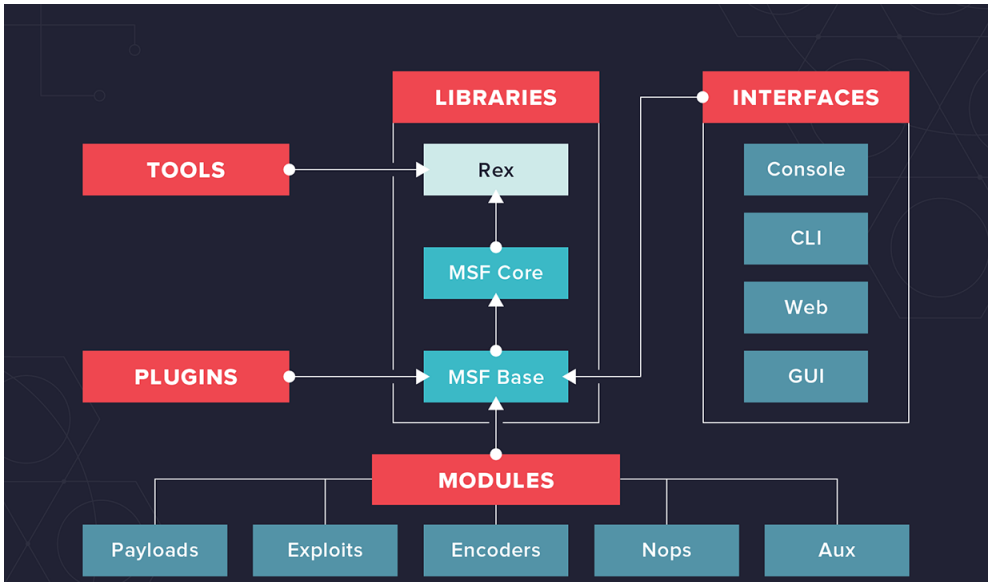


Figure 4.3: Metasploit architecture

4.1.3 Installing the Metasploit Framework on Kali Linux

1. Update the System Before starting, update the package lists and existing packages:

```
sudo apt update && sudo apt upgrade -y
```

2. Install Metasploit from Kali Repositories Metasploit can be installed directly from Kali Linux's official repositories:

```
sudo apt install metasploit-framework -y
```

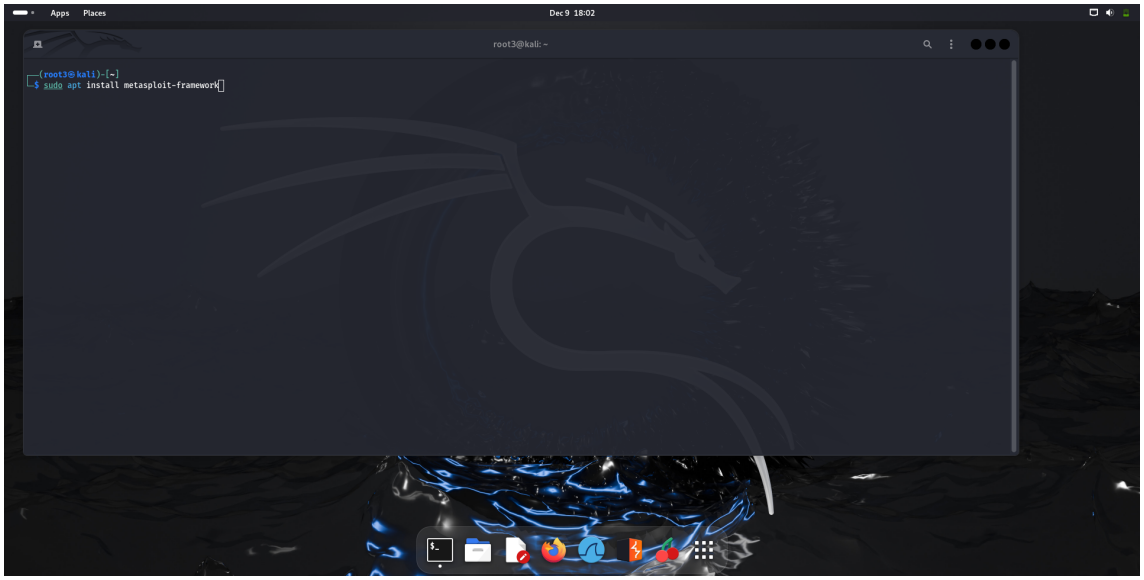


Figure 4.4: Install Metasploit from Kali Repositories

3. Verify the Installation After the installation is complete, verify that Metasploit is correctly installed by running:

```
msfconsole
```

This will launch the Metasploit console.

4. Install from Source (Optional) If you prefer the latest version or the Kali repositories are outdated, you can install Metasploit directly from the source:

   (a) Install dependencies:

```
sudo apt install curl gpg -y
```

   (b) Add the Metasploit repository:

```
curl https://raw.githubusercontent.com/rapid7/metasploit-framework/master/scripts/metasploit-framework.gpg.key | gpg –dearmor > metasploit.gpg sudo mv metasploit.gpg /etc/apt/trusted.gpg.d/ echo "deb https://apt.metasploit.com/ buster main" | sudo tee /etc/apt/sources.list.d/metasploit-framework.list
```

(c) Update and install Metasploit:

```
sudo apt update && sudo apt install metasploit-framework -y
```

5. Launching Metasploit Run the following command to start the Metasploit console:

```
msfconsole
```

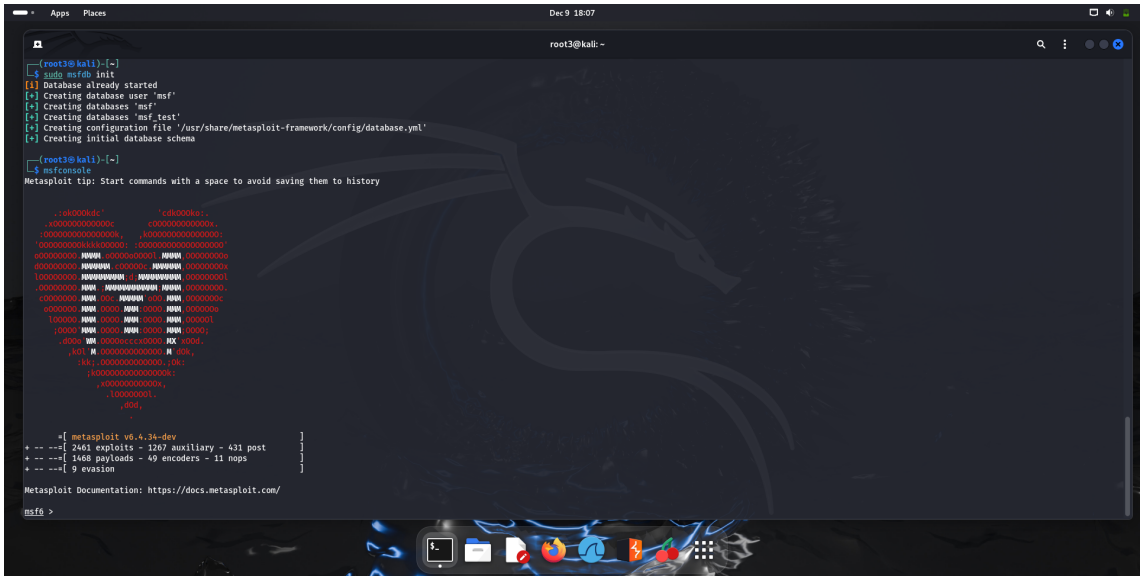


Figure 4.5: Launching Metasploit

4.2 Attack Scenarios

4.2.1 Critical Vulnerabilities

4.2.1.1 Velnurabilities in UnrealIRCd Backdoor (Port 6667)

1. **Context: Attack Scenario** In this attack scenario, the attacker exploits a backdoor vulnerability in UnrealIRCd version 3.2.8.1, a popular Internet Relay Chat (IRC) server software. This backdoor allows remote execution of arbitrary

commands due to the attacker embedding a malicious command within the server's code.

The attacker leverages the Metasploit Framework to exploit this vulnerability and gain unauthorized access to the vulnerable server, which is running on the default IRC port, 6667. Once the backdoor is triggered, the attacker uses a bind shell payload to establish a listening service on a specified port, enabling them to run arbitrary commands remotely on the compromised server.

2. **Attack Steps:**

   (a) Selecting the Exploit Module:

   - I select the unreal_ircd_3281_backdoor exploit in Metasploit, which targets the specific vulnerability in UnrealIRCd version 3.2.8.1.
   - The backdoor vulnerability allows remote code execution when a client connects to an affected UnrealIRCd server.

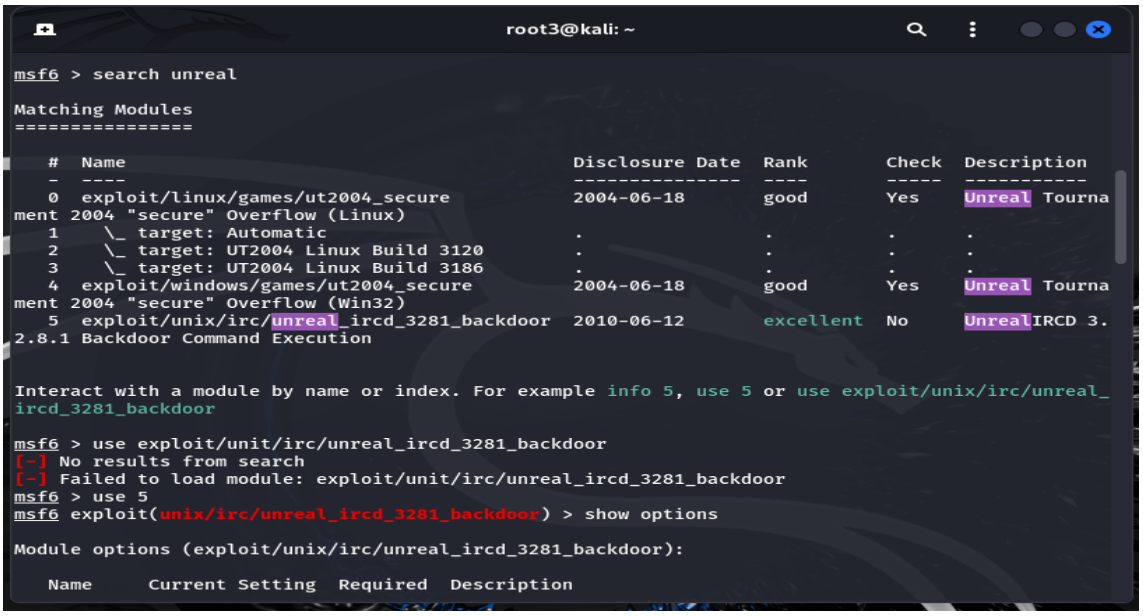


Figure 4.6: Search for Unreal

   (b) Setting Exploit Options:

   - RHOSTS (Remote Host) is set to 192.168.93.129, the target server's IP address.
   - RPORT (Remote Port) is set to 6667, the default port for IRC communication, which is where UnrealIRCd is listening.
   - Payload: I select the cmd/unix/bind_ruby payload, which binds a command shell to a port on the target machine, allowing the attacker to interact with it remotely.

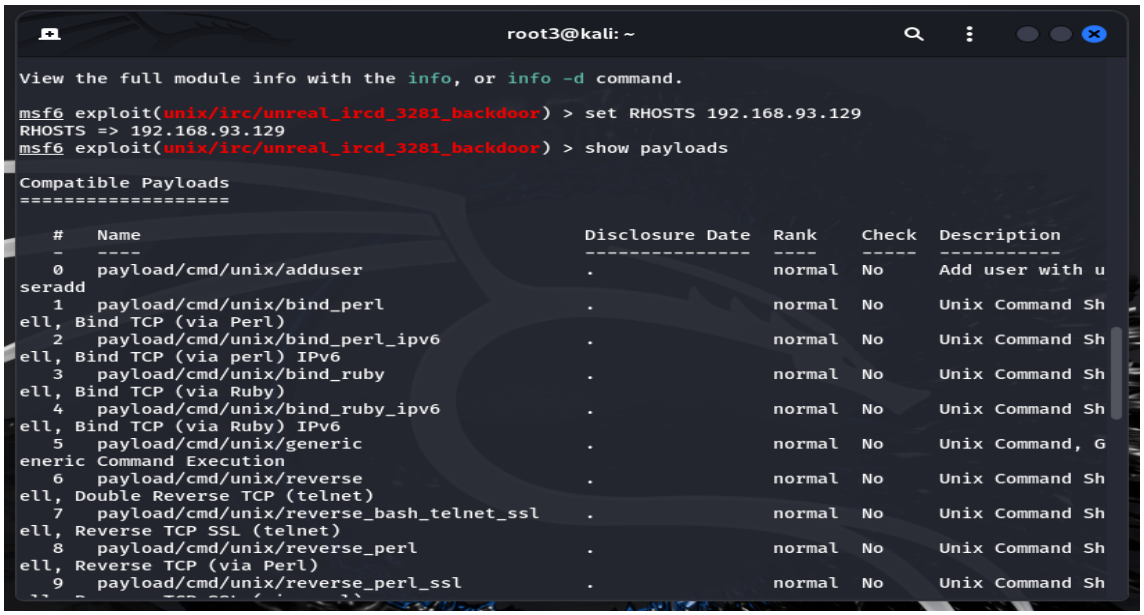


Figure 4.7: Set RHOSTS Unreal

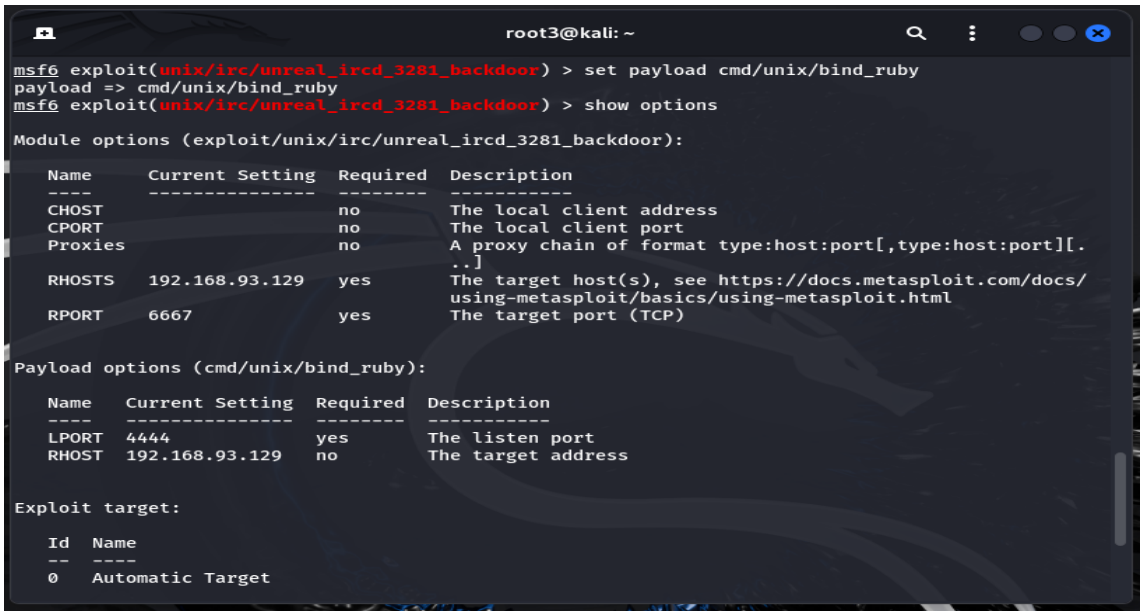


Figure 4.8: Set Payload Unreal

(c) Launching the Exploit:

- I use the exploit command in Metasploit, which triggers the exploitation process.
- The Metasploit framework connects to the UnrealIRCd server on port 6667 and sends the backdoor command to trigger remote command execution.
- A bind shell is opened on the target system, specifically on port 4444.

(d) Achieving Command Execution:

- Once the bind shell is active on the target system, the attacker can establish

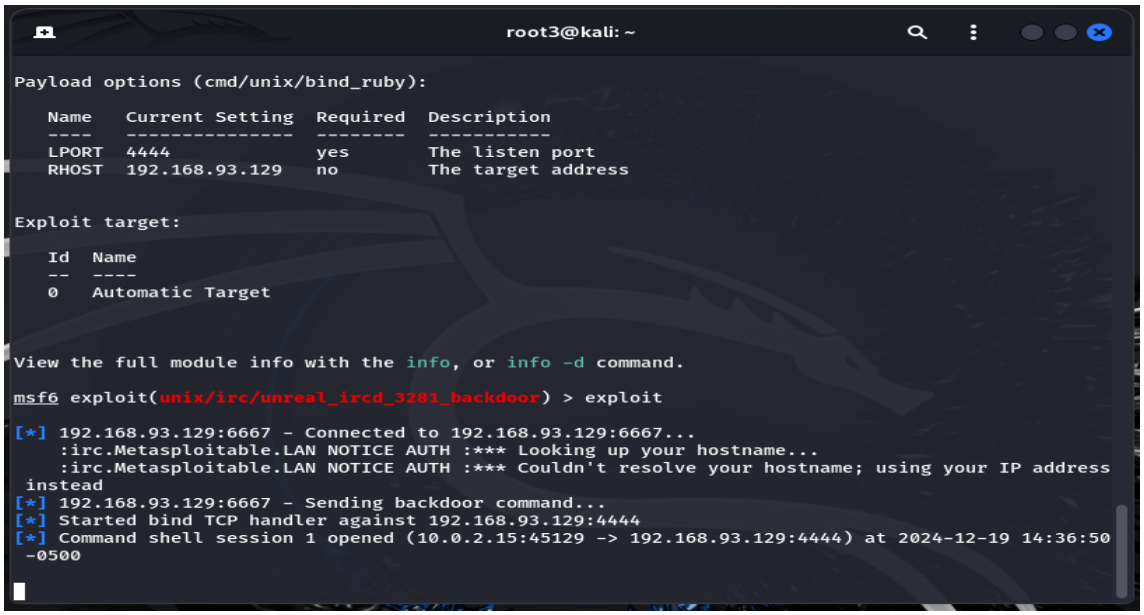


Figure 4.9: Exploiting Unreal

a connection by connecting to the target server on port 4444.

- I successfully open a command shell session, providing them with control over the target server.

3. **Results Analysis and Interpretation:**

   (a) Exploit Success:

   - I successfully exploited the backdoor vulnerability in UnrealIRCd 3.2.8.1, establishing remote access to the target system (192.168.93.129) through port 4444. A command shell was opened, allowing the attacker to run commands on the compromised server.

   (b) Backdoor Mechanism:

   - The backdoor embedded in UnrealIRCd bypasses authentication and security measures, giving any attacker who connects to the server full control and the ability to execute arbitrary commands.

   (c) Bind Shell Payload:

   - I used a bind shell payload (cmd/unix/bind_ruby), causing the compromised system to listen on port 4444 for incoming connections. This method helps evade detection by outbound firewall rules, but can still be identified by monitoring for unusual inbound traffic.

   (d) Attack Impact:

- Remote Command Execution (RCE): The attacker can run any command on the server, leading to potential further exploits like privilege escalation, data theft, or malware installation.
- Potential Lateral Movement: If the compromised system is part of a larger network, the attacker could attempt to move laterally to other systems.

(e) Network Traffic:

- The unusual connection to port 4444 may be detected by network administrators or security tools. Proper network monitoring can help identify and block such activity.

4. **Security Measures for Protecting UnrealIRCd from Exploitation:**

- Upgrade the Software: The most important step is to upgrade UnrealIRCd to a version that is not vulnerable to this backdoor. UnrealIRCd versions after 3.2.8.1 fix this issue.
- Regularly apply security patches to all software, including UnrealIRCd, to protect against known vulnerabilities.
- Configure UnrealIRCd to use SSL/TLS encryption (port 6697) for secure communication. This helps protect against man-in-the-middle (MITM) attacks and makes it more difficult for an attacker to intercept and exploit traffic.
- Disable unencrypted communication on port 6667 to prevent exploitation of the vulnerability over plain text.

4.2.1.2 Vulnerabilities in the VNC Service (Port 5900)

The **port 5900** is used by default by the Virtual Network Computing (VNC) protocol, which allows remote control of a computer via a graphical interface. VNC is a basic implementation of the Remote Frame Buffer (RFB) protocol, which transmits graphical information and keyboard/mouse events between a server (controlled computer) and a client (controller).

1. **Context: VNC Service and Its Vulnerabilities**

   I have identified that port 5900/tcp is open on the target host 192.168.93.129 and that it is using VNC protocol version 3.3. This port is used to establish remote graphical connections.

2. **Attack Scenario: VNC Authentication Scan**

The goal was to test whether the VNC service uses weak authentication mechanisms (such as a default password or an empty password).

Detailed Steps:

(a) Scan the service with nmap:

```
nmap 192.168.93.129 -sV -p 5900
```

Result: Port 5900 is open, and the VNC service (protocol version 3.3) is detected.

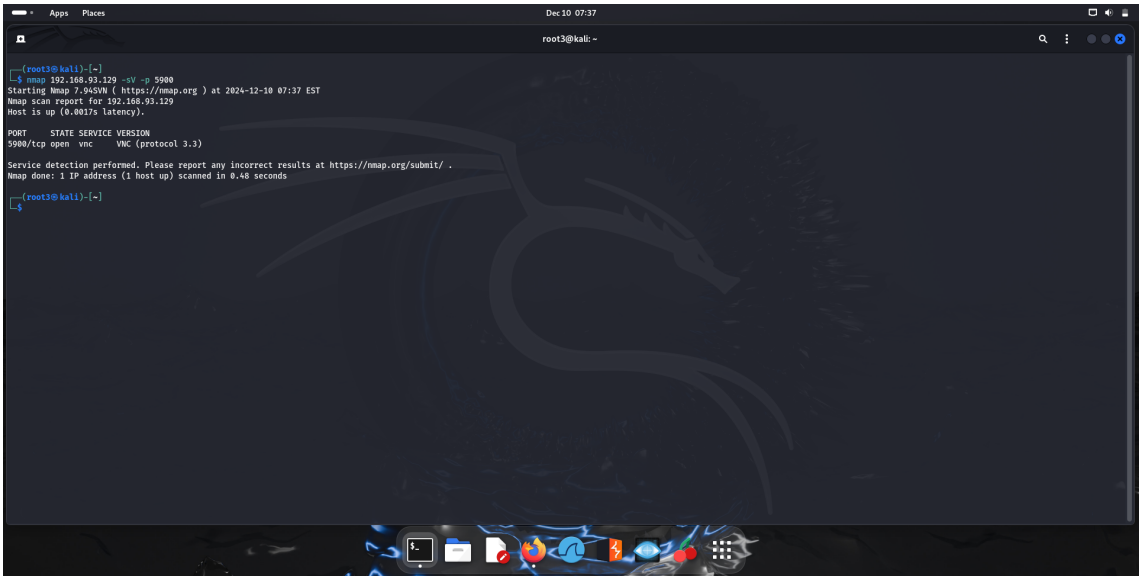


Figure 4.10: Port 5900 is open

(b) Launch Metasploit:

Starts the Metasploit Framework, an exploitation platform for testing vulnerabilities.

(c) Select the VNC authentication test module: Command used:

```
msf6 > use auxiliary/scanner/vnc/vnc_login
```

This module tests different password combinations to access a VNC server.

(d) Configure the module:

I set the target host with:

```
set RHOSTS 192.168.93.129
```

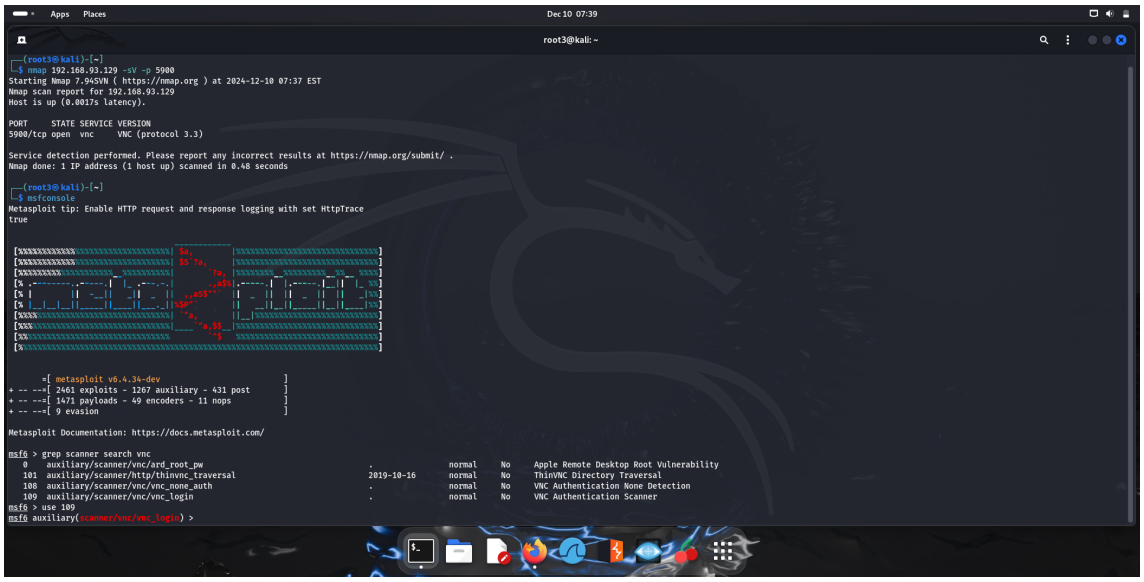


Figure 4.11: Launch Metasploit

Figure 4.12: VNC authentication test module

By default, the module uses a predefined password file located at: `/usr/share/metasploit-framework/data/wordlists/vnc_passwords.txt`.

Figure 4.13: Configure the module

(e) Execute the attack:

I launched the exploit with:

```
exploit
```

Result: The VNC service password is successfully found: password.

Figure 4.14: Attack execution

3. Results Analysis and Interpretation

- The command revealed that VNC authentication was vulnerable because:

  - No specific user was required (USERNAME was left blank).
  - The default password was used.
- This shows a security misconfiguration on the VNC service.

4. **Recommendations for Securing Port 5900** To prevent this type of attack:
   (a) Configure a strong password for the VNC service.
   (b) Disable "None" authentication if it is enabled.
   (c) Restrict access to port 5900 with a firewall (e.g., limit access to trusted IP addresses).
   (d) Use an SSH or VPN tunnel to secure the VNC connection.
   (e) Update the VNC server to fix possible vulnerabilities.

4.2.1.3 Vulnerabilities in Apache Tomcat (Port 8180)

1. **Context: Attack Scenario :**

   In this scenario, I am attempting to exploit a vulnerability in Apache Tomcat running on port 8180. The targeted Tomcat installation is likely misconfigured or using default credentials, which can be exploited to gain unauthorized access.

   Exploit Overview:

   I use the Metasploit Framework module tomcat_mgr_upload to exploit an authentication vulnerability in the Apache Tomcat Manager application. The Tomcat Manager allows authorized users to upload and deploy web applications. If an attacker can authenticate with valid credentials, they can upload malicious files to the server and execute code, potentially leading to remote code execution.

   This exploit targets the Tomcat Manager Application Deployer, which is a service accessible by default in some Tomcat configurations (especially with default settings like tomcat:tomcat as credentials). If the attacker is able to upload a malicious .war file to Tomcat's server, the malicious application can be deployed and executed, potentially granting the attacker a remote shell

2. **Steps of the Attack:**
   (a) **Identify Exploit Module:**
   I select the tomcat_mgr_upload exploit from Metasploit, which is designed to exploit the Tomcat Manager's authenticated file upload functionality.

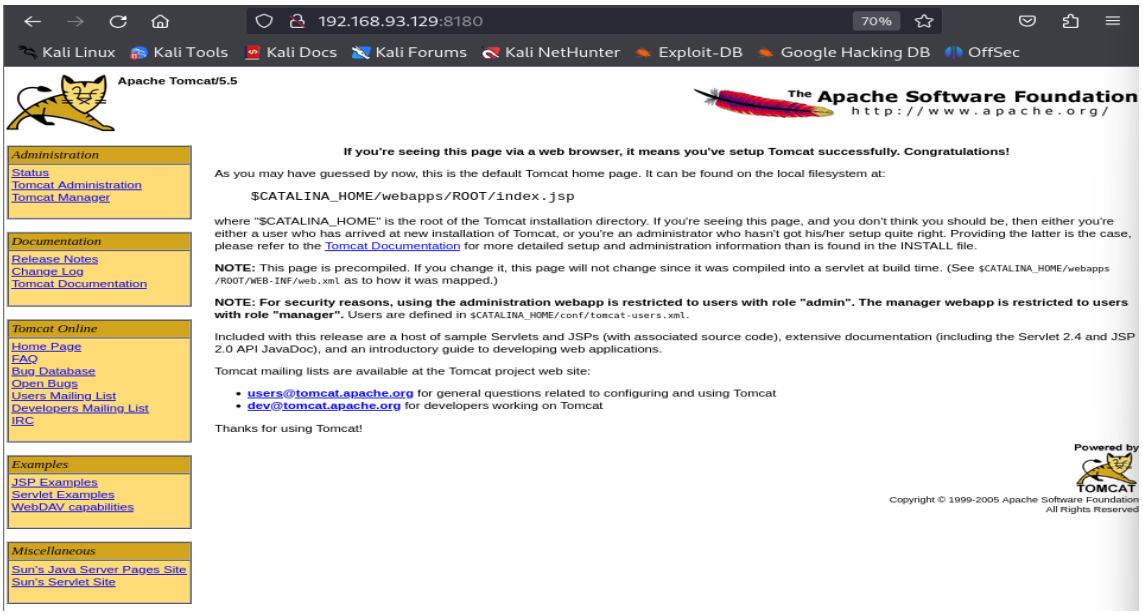


Figure 4.15: Open 192.168.93.129:8180

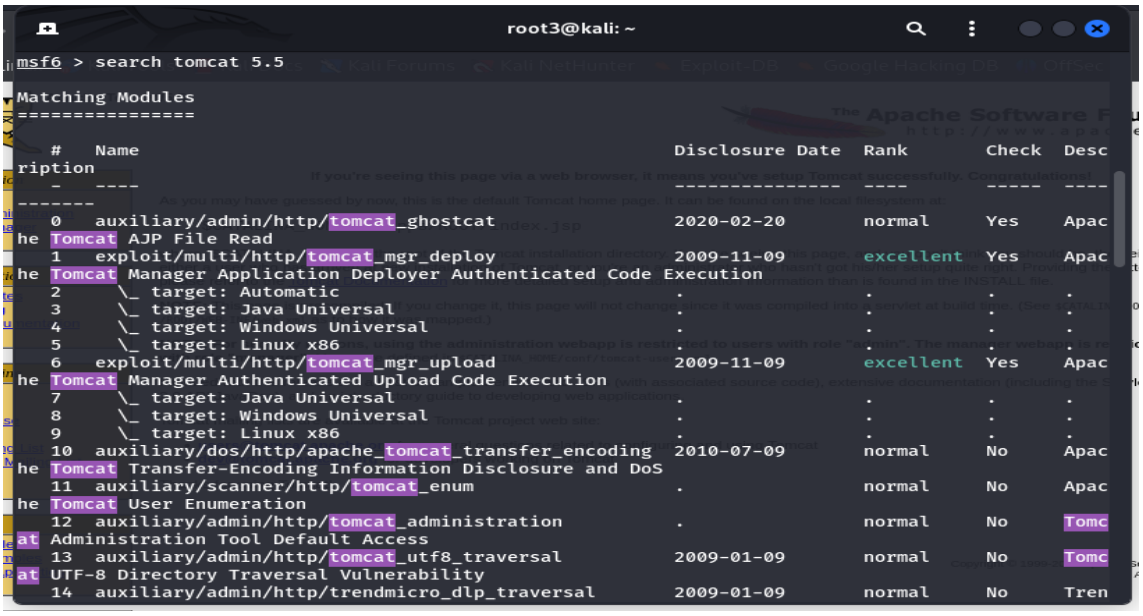


Figure 4.16: Search tomcat 5.5

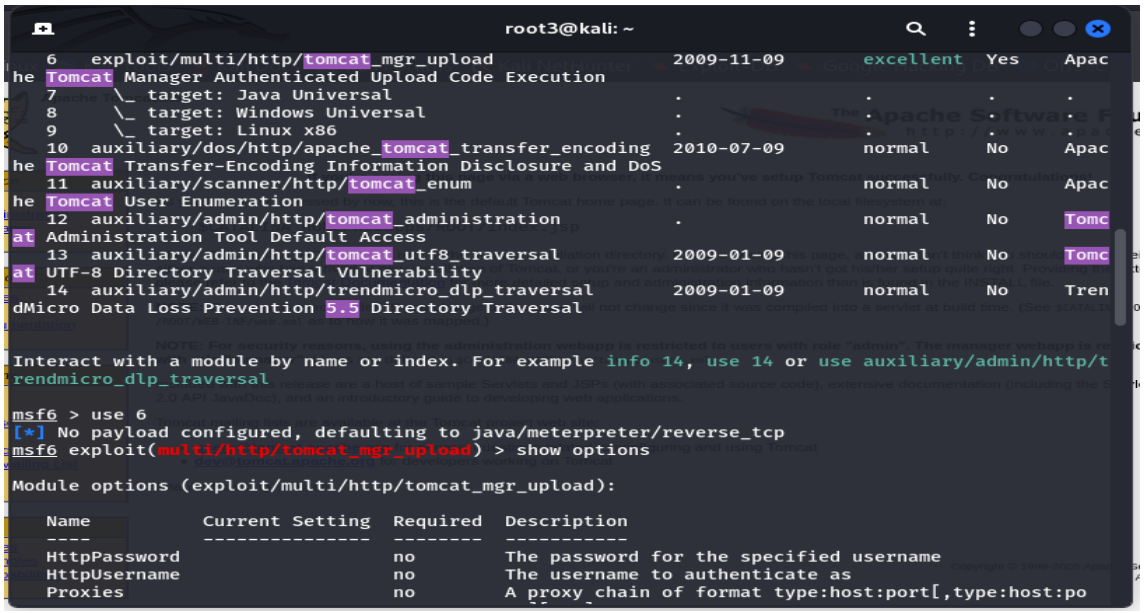


Figure 4.17: Use tomcat_mgr_upload

(b) **Set Exploit Parameters:**

I configure the following options:

- RHOSTS: 192.168.93.129 (the target Tomcat host)
- RPORT: 8180 (the port on which Tomcat is listening)
- HttpUsername: tomcat (default Tomcat manager username)
- HttpPassword: tomcat (default Tomcat manager password)

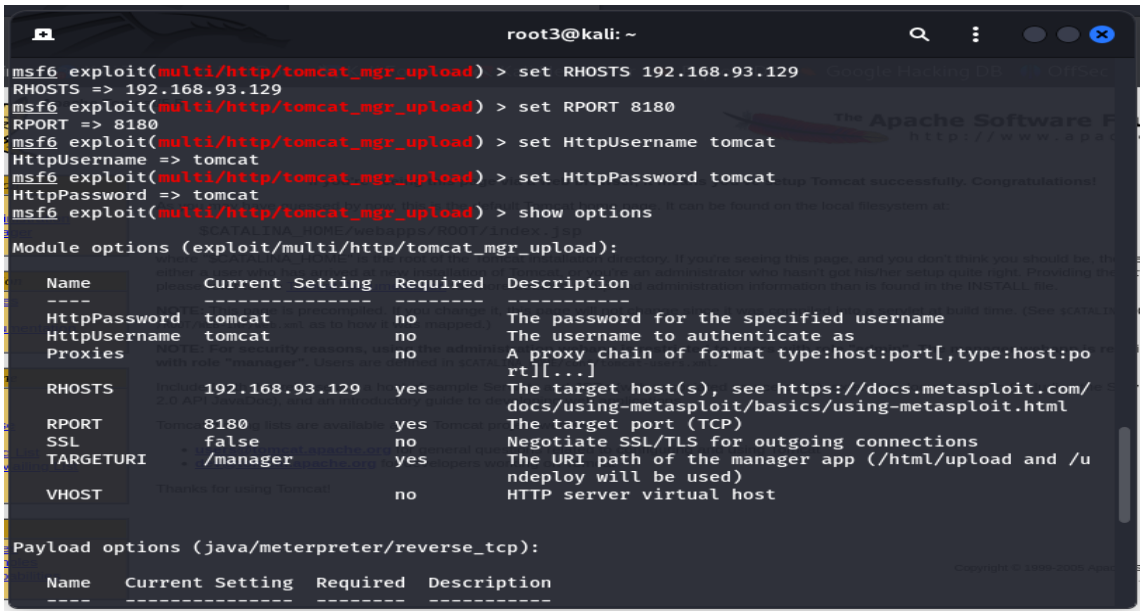


Figure 4.18: Set RHOSTS RPORT .. Tomcat

(c) **Exploit Execution:** When the exploit is run, Metasploit performs the following actions:

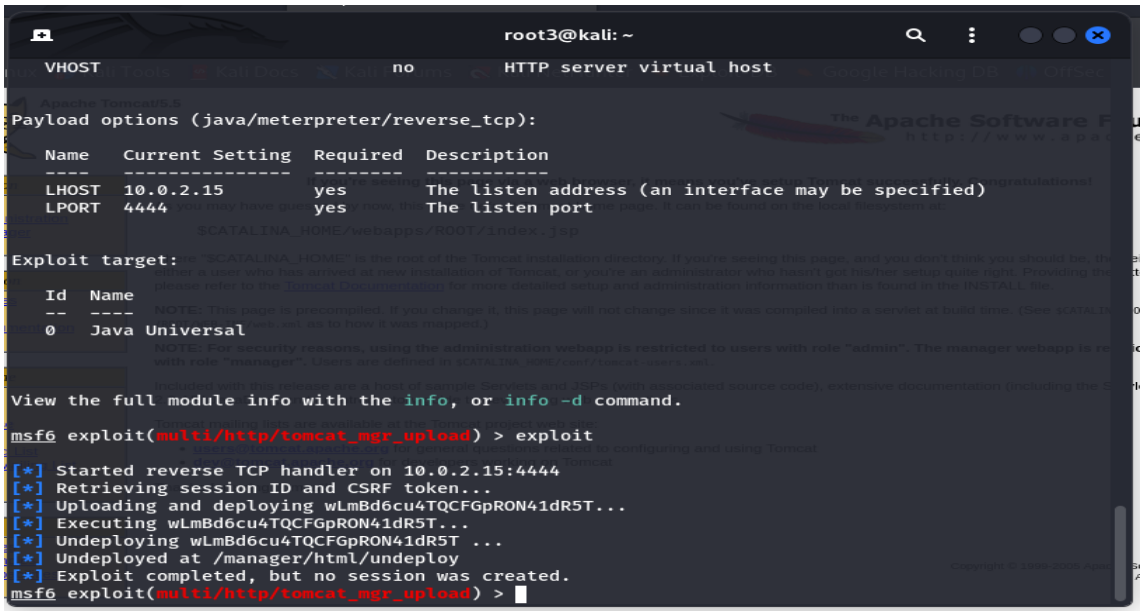


Figure 4.19: Exploiting Tomcat on Port 8180

- Reverse TCP Handler: A listener is started on the attacker's machine at 192.168.93.129:4444 to handle the incoming reverse shell connection.
- Session ID and CSRF Token: The exploit retrieves necessary session tokens to bypass CSRF protection.
- Upload Malicious Application: The attacker uploads a .war file (Java Web Application Archive), which contains the payload (in this case, a Meterpreter reverse shell).
- Deploy and Execute: After uploading, the malicious application is deployed, and Metasploit attempts to execute it.
- Undeploy: After execution, the malicious .war file is undeployed to remove traces of the exploit.

(d) **Outcome:**

- Exploit completed: While the exploit runs successfully, no reverse shell session is created, and the attacker does not gain access to the target system.

3. **Results Analysis and Interpretation:**

(a) Exploit Execution:

- The exploit successfully uploaded the malicious payload, but no reverse shell was established. Possible causes include network/firewall issues or Tomcat security settings blocking code execution.

(b) Potential Issues Identified:

- Weak authentication due to default credentials (tomcat:tomcat), which should be changed in production.

(c) Mitigation Failures:

- The lack of a reverse shell suggests misconfiguration or defensive measures preventing payload execution, though basic security measures (like strong credentials) were not implemented.

4. **Security Measures for Protecting Apache Tomcat:**

- o Always change the default Tomcat Manager username (tomcat) and password (tomcat) to a strong, unique set of credentials. Use password managers to store and manage these credentials securely.
- IP whitelisting: Limit access to the Tomcat Manager by only allowing specific trusted IP addresses or internal networks to access the http://<target>:8180/manager.
- Firewall Configuration: Use firewalls to block all external access to the Tomcat Manager and other administrative interfaces (e.g., /manager, /host-manager), except from authorized administrators.
- Implement strong authentication methods for Tomcat, such as multi-factor authentication (MFA) or LDAP integration for managing user credentials.
- Configure Tomcat to require strong passwords and user roles to limit access based on permissions.

4.2.2 High Vulnerabilities

4.2.2.1 Vulnerabilities in the MySQL Service (Port 3306)

MySQL is an open-source database management system that allows you to store, manage, and retrieve data in relational databases. MySQL is used for applications ranging from websites to enterprise systems, due to its speed, flexibility, and easy integration with programming languages such as PHP, Python, and Java.

Main features of MySQL:

- **Open source**: MySQL is free software under the GPL license.
- **Performance**: It is designed to be fast and efficient in processing SQL queries.

- **Secure**: It offers security features such as authentication and connection encryption.

MySQL Port 3306 Port 3306 is the default port used by MySQL to establish communication between the MySQL server and its clients. When applications or users want to connect to a MySQL database, they connect to this port to exchange information.

Role of Port 3306 Port 3306 allows the transmission of SQL queries from clients to the server, and results from the server to clients. This port is used for any interaction between applications and the MySQL database server, whether it is a SELECT, INSERT, UPDATE, or DELETE query.

1. **Context: Common MySQL Vulnerabilities**

   In this scenario, the objective is to analyze and understand the security posture of a MySQL server running on the target machine `192.168.93.129`. Various connection attempts are made to assess potential vulnerabilities, focusing on port 3306, which is the default MySQL port.

2. **Attack Scenario: Exploiting MySQL Vulnerabilities**

   (a) **Nmap Scan (MySQL Empty Password Script):** The Nmap scan using the `mysql-empty-password` script was unsuccessful. The script failed to execute properly, but the fact that port 3306 is open indicates that MySQL is running and listening for connections on that port. This could be a potential vulnerability if the root user has an empty password.

```
nmap –script=mysql-empty-password 192.168.93.129
```

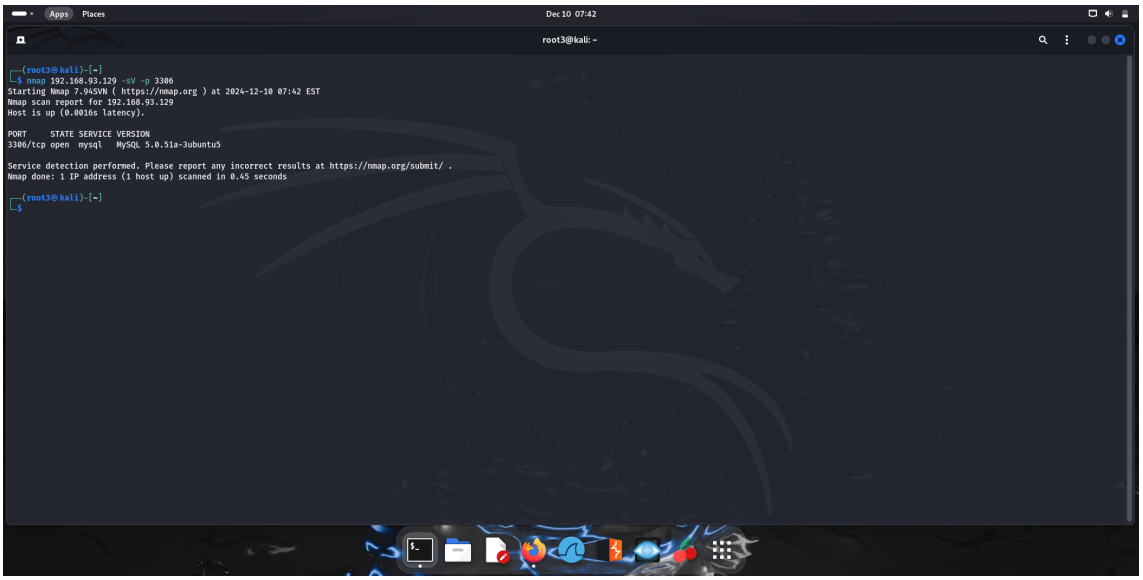


Figure 4.20: Scan opened port 3306

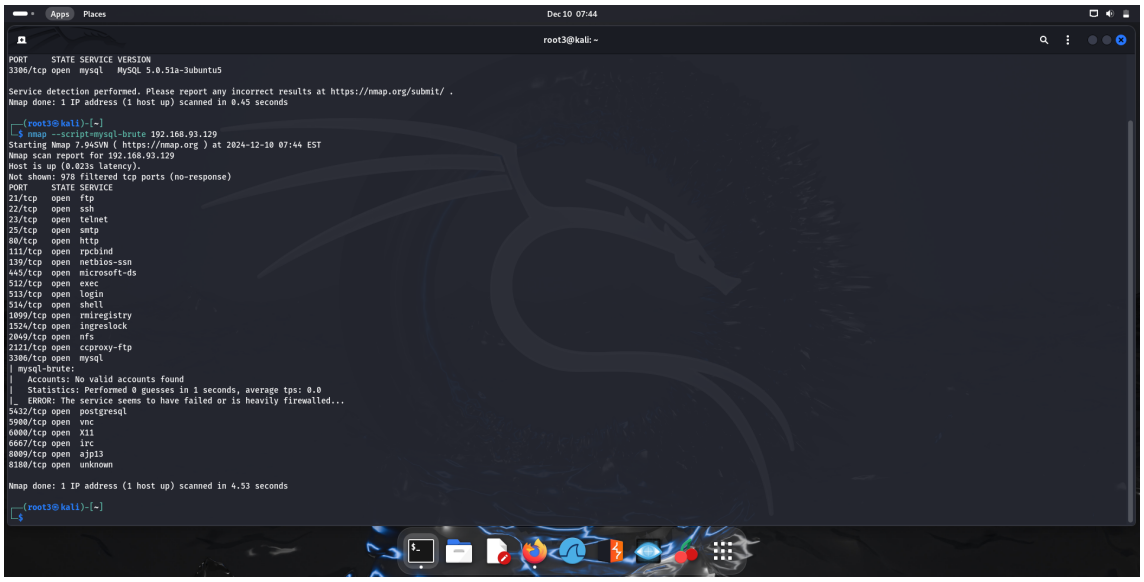


Figure 4.21: Scan script mysql brute

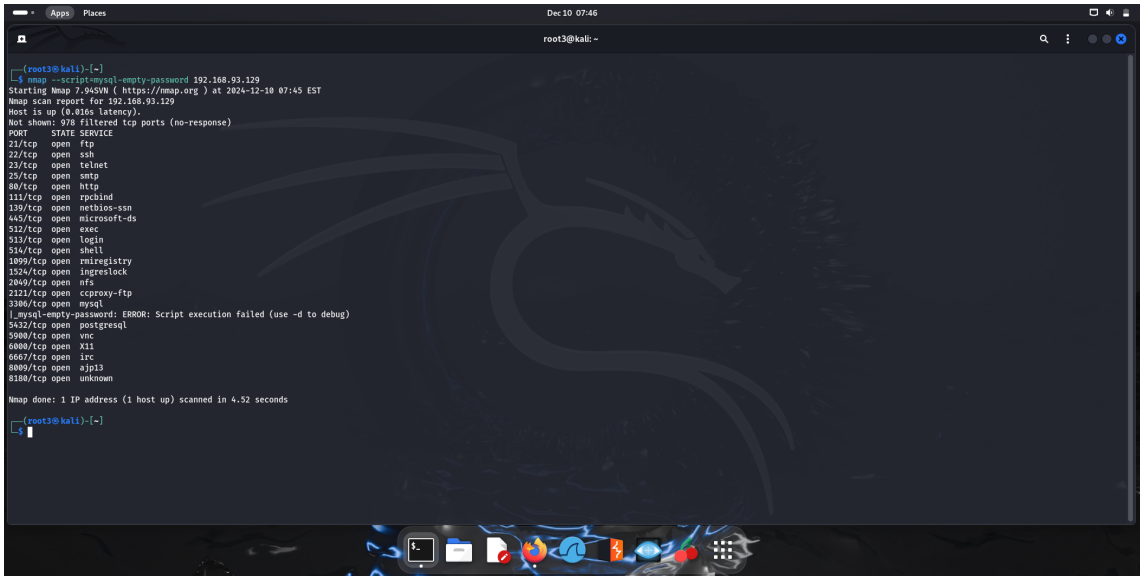


Figure 4.22: MySQL Empty Password Script

(b) **SSL/TLS Connection Attempt:** The attempt to connect using SSL failed due to a version mismatch. This may indicate that the MySQL server is using an outdated or insecure SSL/TLS configuration, which can be exploited by attackers to intercept or tamper with sensitive data.

```
mysql -u root -h 192.168.93.129 -p
```

The error message suggests that the MySQL client and server were unable to negotiate a secure SSL/TLS connection. This could lead to information leakage or man-in-the-middle (MITM) attacks if not addressed.

(c) **Access Denied for Root User:** The connection attempts with the `-skip-ssl` flag failed, as expected. The MySQL server rejected access to the root user because authentication failed due to incorrect credentials.

3. **Recommendations for Protecting Port 3306**

   (a) Secure the MySQL root user account by setting a strong password.

   (b) Disable remote root login to reduce exposure.

   (c) Use SSL/TLS encryption to protect communication between MySQL clients and the server.

   (d) Restrict access to port 3306 using a firewall or VPN, allowing only trusted clients to connect.

   (e) Regularly update MySQL to address known vulnerabilities.

4.2.2.2 Vulnerabilities in HTTP Services (Port 80)

The **HTTP protocol** is a stateless communication protocol used for transferring hypertext on the web. It allows a client (usually a web browser) to request resources from a web server, which responds with the requested information.

1. **Context: Common HTTP Vulnerabilities**

   The server listens on port 80 for HTTP traffic. It is important to verify whether any known vulnerabilities affect this version of Apache and PHP, particularly those related to remote code execution.

2. **Attack Scenario: Exploiting HTTP Vulnerabilities**

   (a) **Service Detection (Nmap)**: The `nmap` tool was used to perform a service scan on port 80. The service detection result is as follows:

```
PORT STATE SERVICE VERSION
80/tcp open http Apache httpd 2.2.8 ((Ubuntu) DAV/2)
```

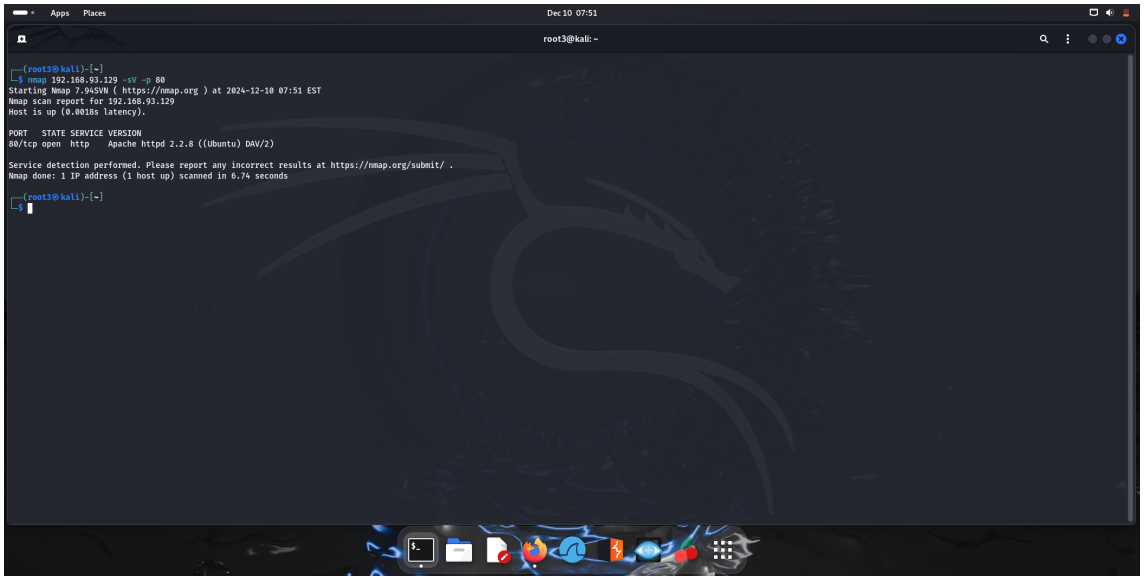


Figure 4.23: Nmap scan showing HTTP on port 80.

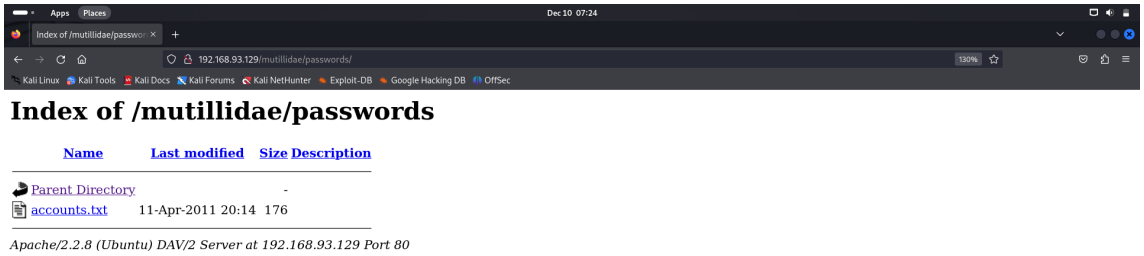


Figure 4.24: Accessing the server via the IP address.

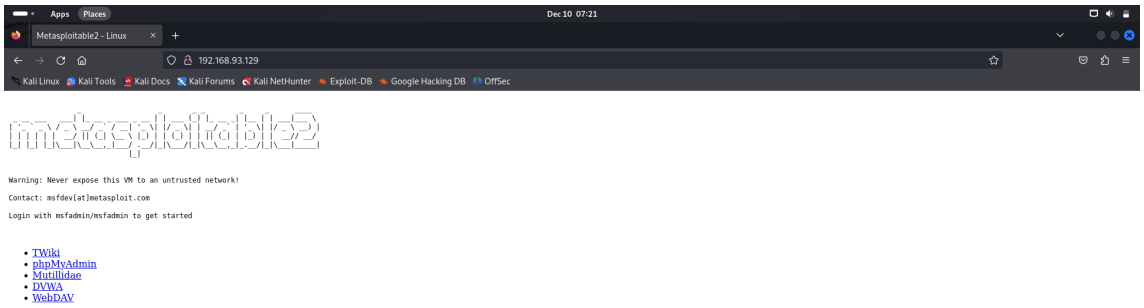


Figure 4.25: Mutillidae password page.

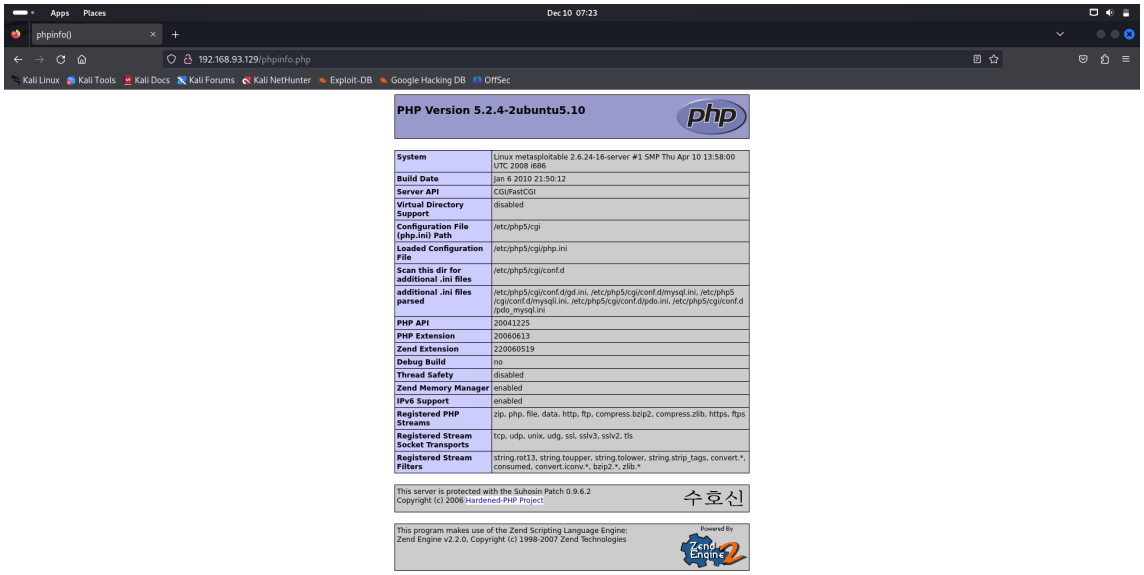


Figure 4.26: PHP information page on the server.

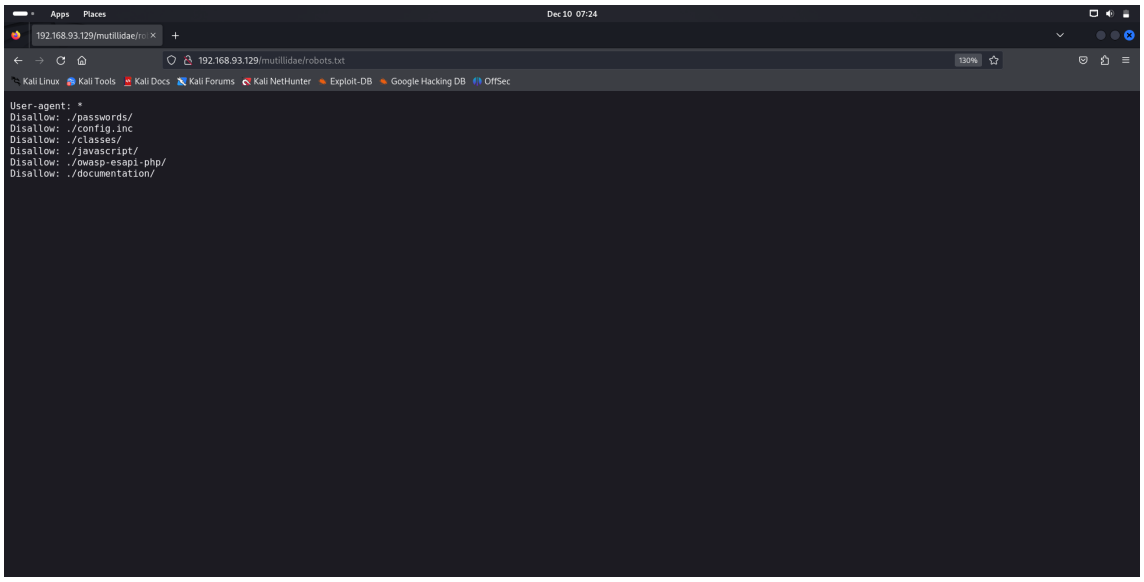


Figure 4.27: Mutillidae robots.txt file.

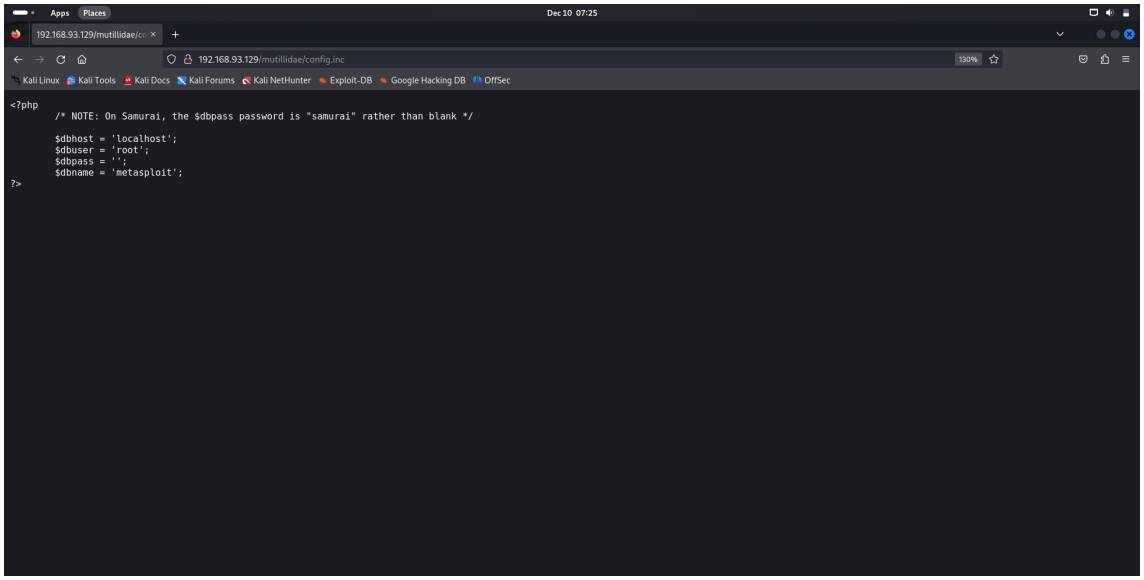


Figure 4.28: Mutillidae configuration page.

(b) **HTTP Version Detection with Metasploit**:

The `auxiliary/scanner/http/http_version` module in Metasploit was used to detect the exact version of Apache and PHP running on the server. The following information was collected:

```
Apache/2.2.8 (Ubuntu) DAV/2 (Powered by
PHP/5.2.4-2ubuntu5.10)
```

(c) **Exploit Search with Searchsploit**: Using `searchsploit`, a search was performed to identify potential exploits for Apache 2.2.8 with PHP. The following

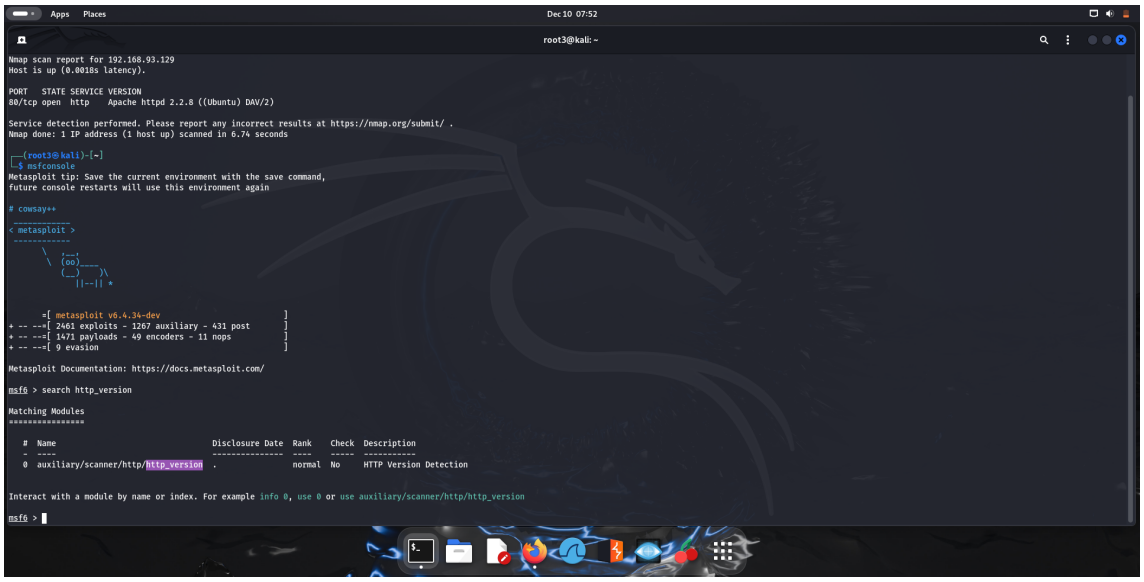


Figure 4.29: HTTP version detection using Metasploit.

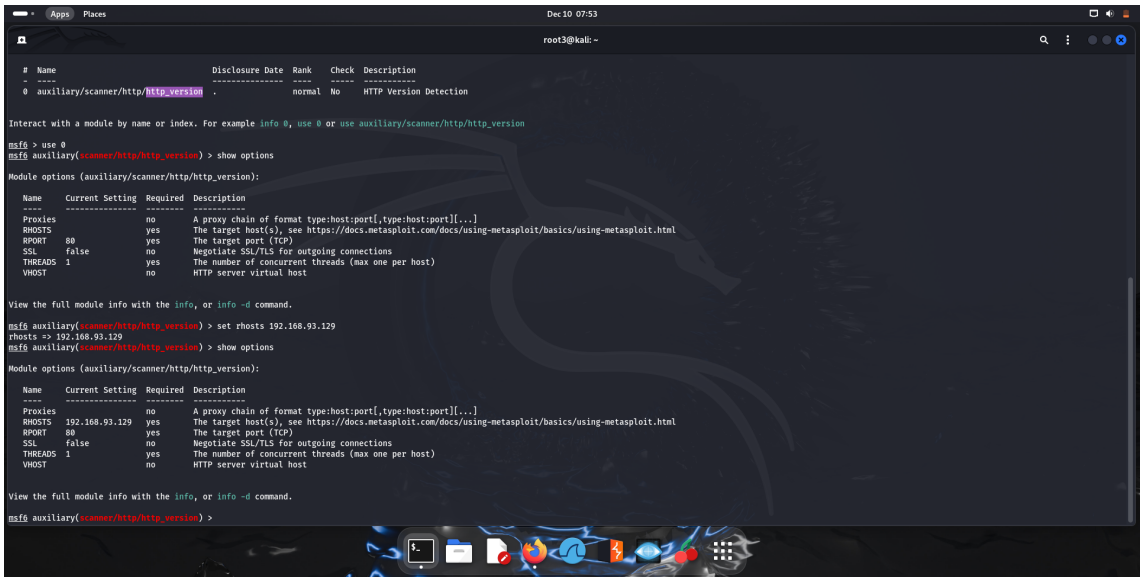


Figure 4.30: Setting RHOSTS for the Metasploit scanner.

results were found:

- `Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution`
- `Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner`

(d) **Exploitation with Metasploit**: The `php_cgi_arg_injection` exploit was attempted due to the vulnerability in PHP versions below 5.3.12. However, the exploit did not succeed in creating a session. The server failed to execute the malicious code, despite initiating the reverse TCP handler.

```
Exploit completed, but no session was created.
```

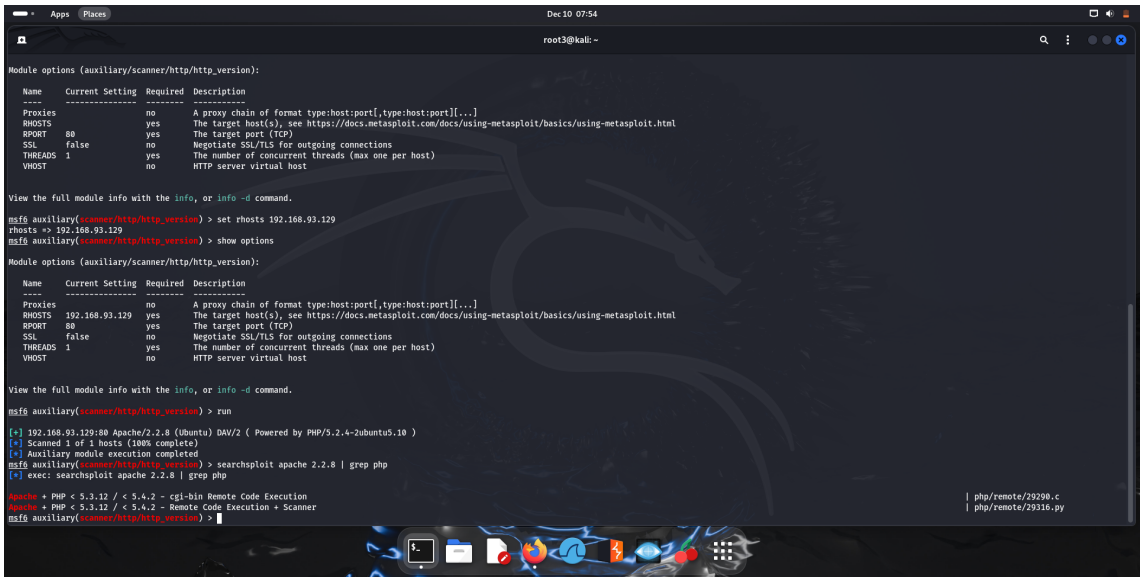


Figure 4.31: Searchsploit results for Apache 2.2.8 and PHP vulnerabilities.

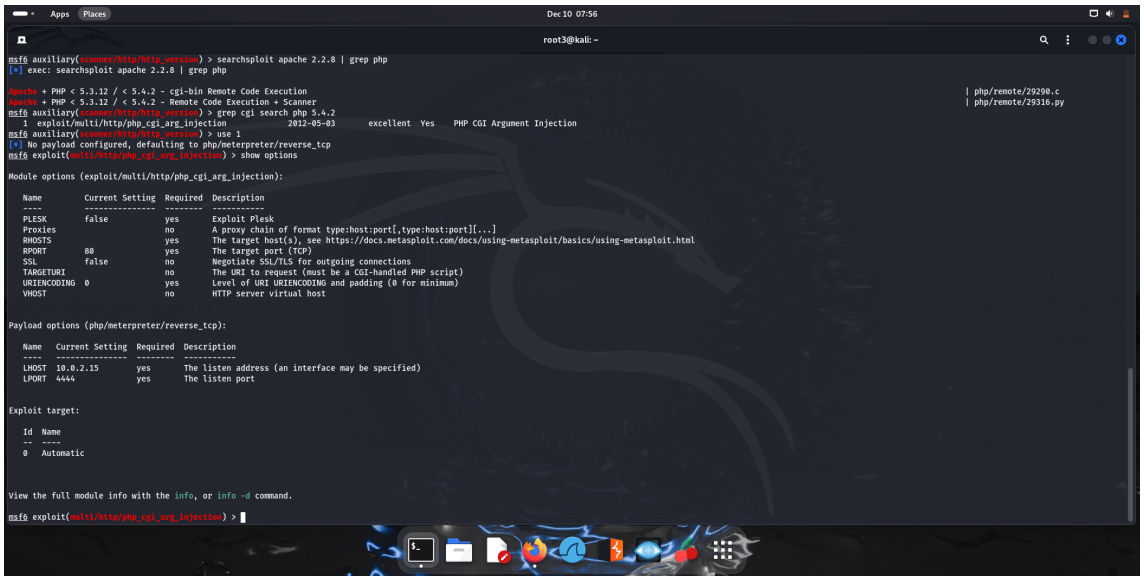


Figure 4.32: Attempted exploitation of the HTTP server.

3. Results Analysis and Interpretation

- **Apache Version:** Apache 2.2.8 is outdated and contains several known vulnerabilities, especially in relation to the integration with PHP and handling of CGI scripts.
- **PHP Version:** PHP 5.2.4 is also outdated and vulnerable to various attacks, including remote code execution (RCE) and CGI argument injection in PHP scripts.

4. Security Measures for HTTP Services

To secure the Apache and PHP server:

- **Update Apache:** Ensure the latest stable version of Apache is used to mitigate known vulnerabilities in older versions.
- **Update PHP:** Upgrade to a supported version of PHP (at least 7.x or higher) to close the known vulnerabilities associated with PHP 5.2.4.

4.2.3 Medium Vulnerabilities

4.2.3.1 Vulnerabilities in PostgreSQL (Port 5432)

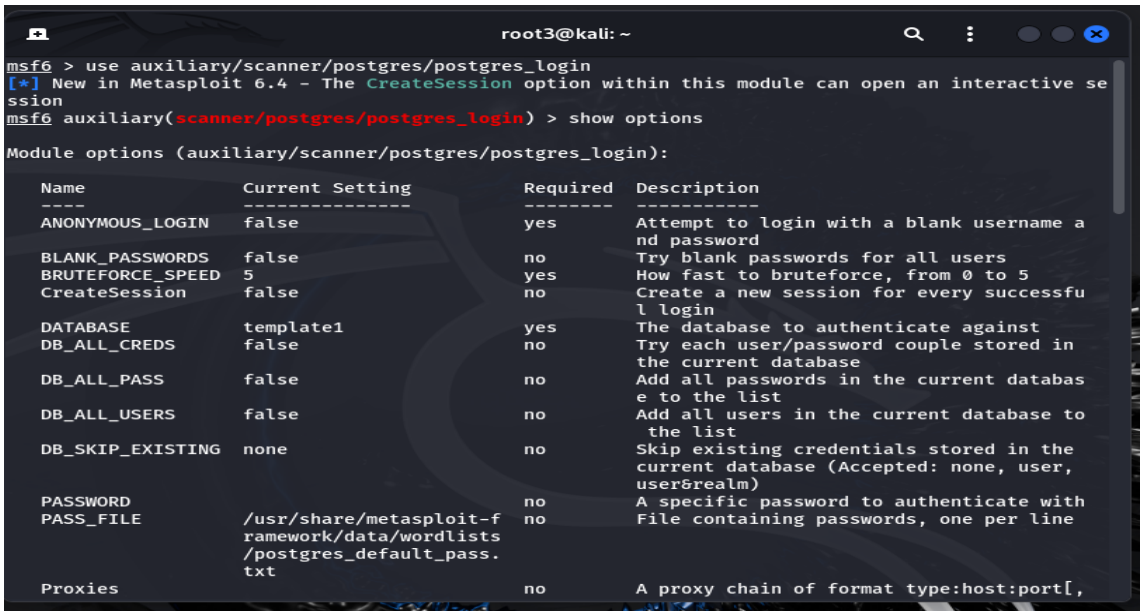
1. **Context: Attack Scenario**

   In this attack scenario, the objective is to exploit a PostgreSQL server running on port 5432. The attack leverages a brute-force login attack using the Metasploit Framework's postgres_login auxiliary module to gain unauthorized access to the PostgreSQL service.

2. **Attack Steps:**

   (a) Setting Module Options:

   - USERNAME: I set the username to postgres, a common default PostgreSQL username.
   - USER_AS_PASS: This option is set to true, meaning the module will attempt to use the username as the password (e.g., postgres:postgres).
   - RHOSTS: I specefie the target host IP (192.168.93.129) as the target for the attack.
   - RPORT: The PostgreSQL service is typically running on port 5432, which is the default port for PostgreSQL.

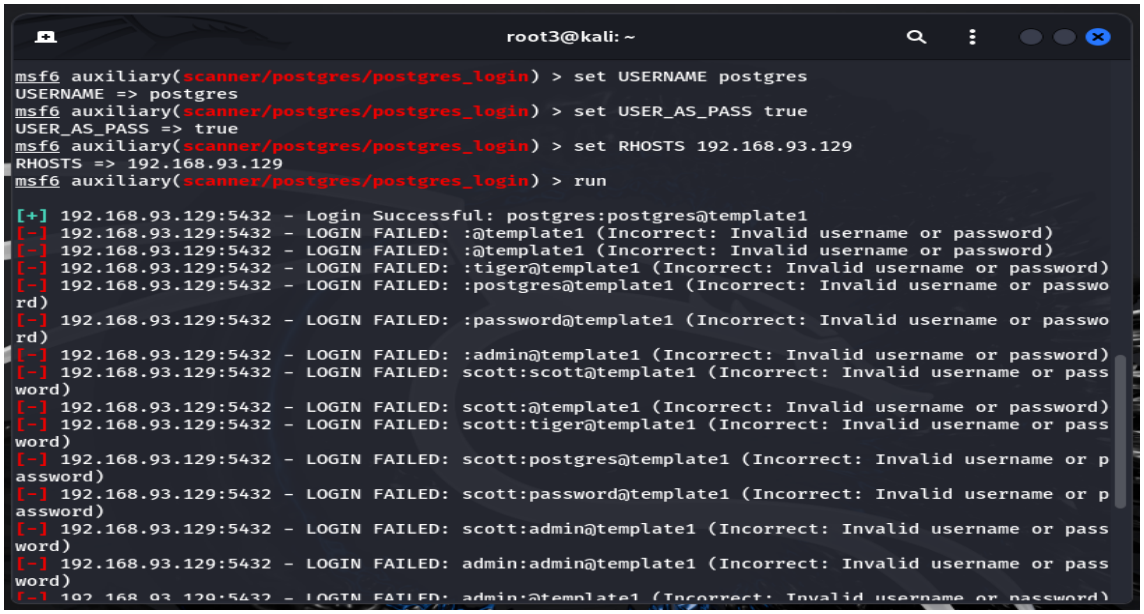


```
root3@kali: ~
msf6 > use auxiliary/scanner/postgres/postgres_login
[*] New in Metasploit 6.4 - The CreateSession option within this module can open an interactive se
ssion
msf6 auxiliary(scanner/postgres/postgres_login) > show options

Module options (auxiliary/scanner/postgres/postgres_login):

   Name               Current Setting           Required  Description
   ----               ---------------           --------  -----------
   ANONYMOUS_LOGIN    false                     yes       Attempt to login with a blank username a
                                                          nd password
   BLANK_PASSWORDS    false                     no        Try blank passwords for all users
   BRUTEFORCE_SPEED   5                         yes       How fast to bruteforce, from 0 to 5
   CreateSession      false                     no        Create a new session for every successfu
                                                          l login
   DATABASE           template1                 yes       The database to authenticate against
   DB_ALL_CREDS       false                     no        Try each user/password couple stored in
                                                          the current database
   DB_ALL_PASS        false                     no        Add all passwords in the current databas
                                                          e to the list
   DB_ALL_USERS       false                     no        Add all users in the current database to
                                                           the list
   DB_SKIP_EXISTING   none                      no        Skip existing credentials stored in the
                                                          current database (Accepted: none, user,
                                                          user&realm)
   PASSWORD                                     no        A specific password to authenticate with
   PASS_FILE          /usr/share/metasploit-f   no        File containing passwords, one per line
                      ramework/data/wordlists
                      /postgres_default_pass.
                      txt
   Proxies                                      no        A proxy chain of format type:host:port[,
```

Figure 4.33: Use Postgress Login

(b) Running the Brute-Force Attack:

- I initiate the run command, and the module begins brute-forcing login attempts using various combinations of usernames and passwords (in this case, attempting the username as the password as well).
- I continue until it successfully logs in with a valid credential.



```
root3@kali: ~
msf6 auxiliary(scanner/postgres/postgres_login) > set USERNAME postgres
USERNAME => postgres
msf6 auxiliary(scanner/postgres/postgres_login) > set USER_AS_PASS true
USER_AS_PASS => true
msf6 auxiliary(scanner/postgres/postgres_login) > set RHOSTS 192.168.93.129
RHOSTS => 192.168.93.129
msf6 auxiliary(scanner/postgres/postgres_login) > run

[+] 192.168.93.129:5432 - Login Successful: postgres:postgres@template1
[-] 192.168.93.129:5432 - LOGIN FAILED: :@template1 (Incorrect: Invalid username or password)
[-] 192.168.93.129:5432 - LOGIN FAILED: :@template1 (Incorrect: Invalid username or password)
[-] 192.168.93.129:5432 - LOGIN FAILED: :tiger@template1 (Incorrect: Invalid username or password)
[-] 192.168.93.129:5432 - LOGIN FAILED: :postgres@template1 (Incorrect: Invalid username or passwo
rd)
[-] 192.168.93.129:5432 - LOGIN FAILED: :password@template1 (Incorrect: Invalid username or passwo
rd)
[-] 192.168.93.129:5432 - LOGIN FAILED: :admin@template1 (Incorrect: Invalid username or password)
[-] 192.168.93.129:5432 - LOGIN FAILED: scott:scott@template1 (Incorrect: Invalid username or pass
word)
[-] 192.168.93.129:5432 - LOGIN FAILED: scott:@template1 (Incorrect: Invalid username or password)
[-] 192.168.93.129:5432 - LOGIN FAILED: scott:tiger@template1 (Incorrect: Invalid username or pass
word)
[-] 192.168.93.129:5432 - LOGIN FAILED: scott:postgres@template1 (Incorrect: Invalid username or p
assword)
[-] 192.168.93.129:5432 - LOGIN FAILED: scott:password@template1 (Incorrect: Invalid username or p
assword)
[-] 192.168.93.129:5432 - LOGIN FAILED: scott:admin@template1 (Incorrect: Invalid username or pass
word)
[-] 192.168.93.129:5432 - LOGIN FAILED: admin:admin@template1 (Incorrect: Invalid username or pass
word)
[-] 192.168.93.129:5432 - LOGIN FAILED: admin:@template1 (Incorrect: Invalid username or password)
```

Figure 4.34: Set USERNAME USER_AS_PASS RHOSTS Postgress

(c) Success

- I have successfully authenticated to the PostgreSQL server with the postgres:postgres credentials.

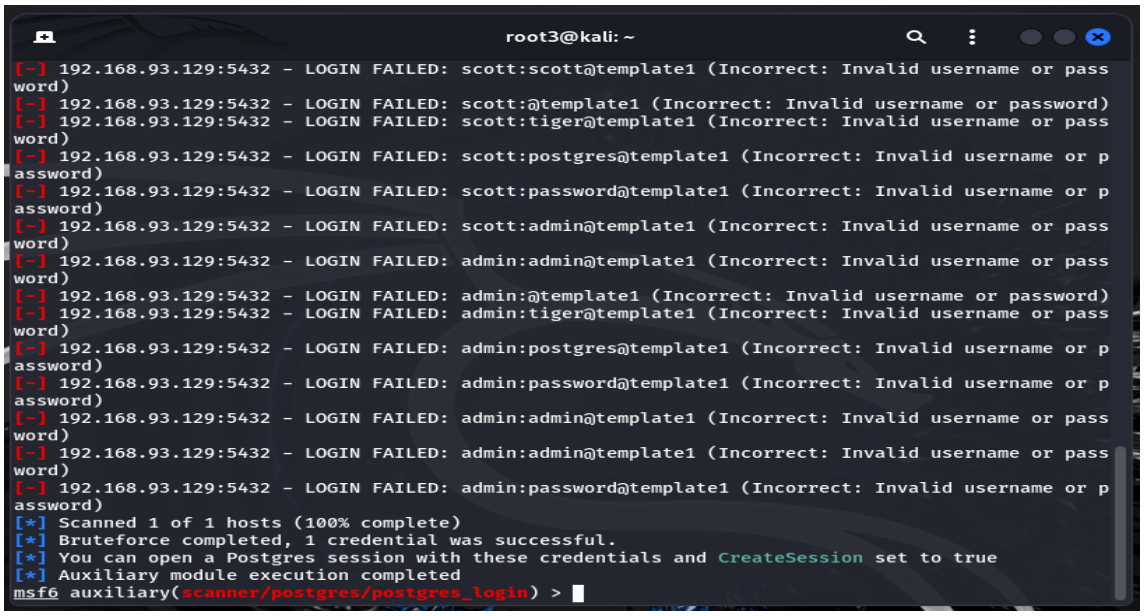


Figure 4.35: Exploiting Postgress

3. **Results Analysis and Interpretation**

   (a) Strong credential management:

   - Always avoid default credentials, enforce strong passwords, use role-based access control (RBAC), and rotate credentials regularly.

   (b) Network security:

   - Secure databases behind firewalls, use VPNs or SSH tunnels for remote access, implement IP allowlisting, and enable SSL/TLS encryption.

   (c) Monitoring and detection:

   - Log login attempts, configure alerts for suspicious activities, enforce account lockout policies, and conduct regular security audits.

4. **Security Measures for Protecting PostgreSQL**

   - Start with PostgreSQL's built-in security features:
     - pg_hba.conf for managing authentication, implement SSL/TLS for secure connections, and configure role-based access control (RBAC).
   - For access control:

  - Create application-specific users, grant the minimum necessary privileges, audit permissions and access logs regularly, and remove unused accounts promptly.

- Secure remote access through:
  - Use SSH tunneling for encrypted connections, require VPN for external access, implement network segmentation and firewalls, and utilize connection pooling tools like PgBouncer.

4.2.4 Low Vulnerabilities

4.2.4.1 Vulnerabilities in Java Rmi (Ports 1099 49104)

1. **Context: Attack Scenario** The scenario involves exploiting a Java RMI (Remote Method Invocation) service running on port 1099 of a target machine (192.168.93.129). The attacker is using the Metasploit Framework (msf6) to exploit an insecure default configuration in the Java RMI server to gain remote code execution (RCE).

2. **Attack Scenario :**

   (a) Attacker Preparation: I configure the exploit to use a reverse TCP payload with the attacker's machine (192.168.93.129) as the listener on port 4444.

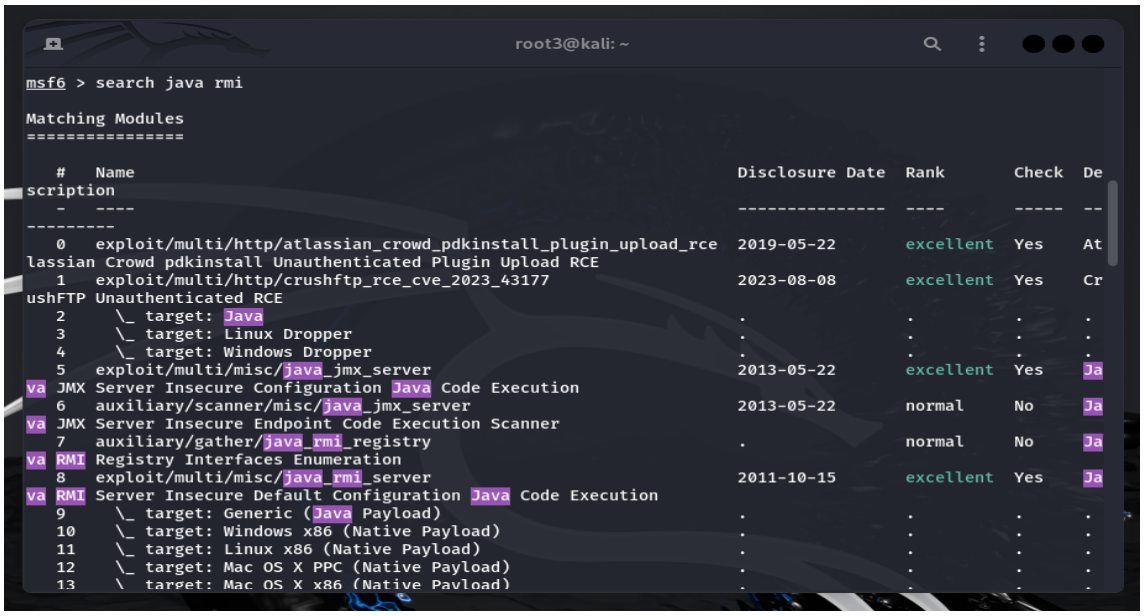


Figure 4.36: Search JAVA RMI

   (b) Payload Setup: The exploit specifies that the victim machine will contact the attacker's HTTP server (running on port 8080).

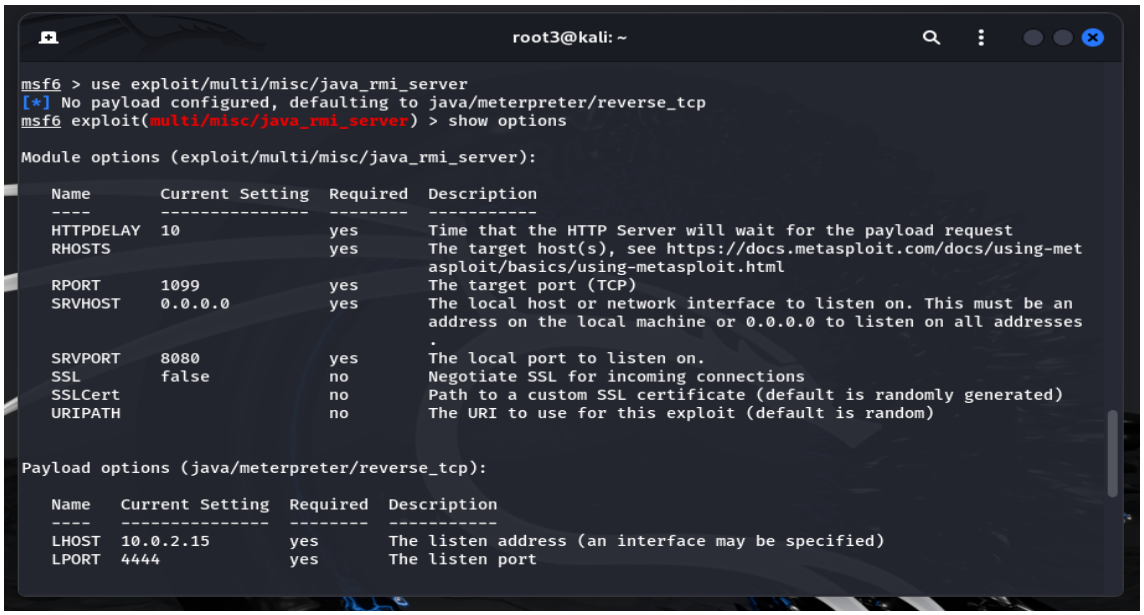


Figure 4.37: Using java_rmi_server

(c) Exploit Execution: I send RMI headers and calls to the target machine, which is expected to trigger the payload.

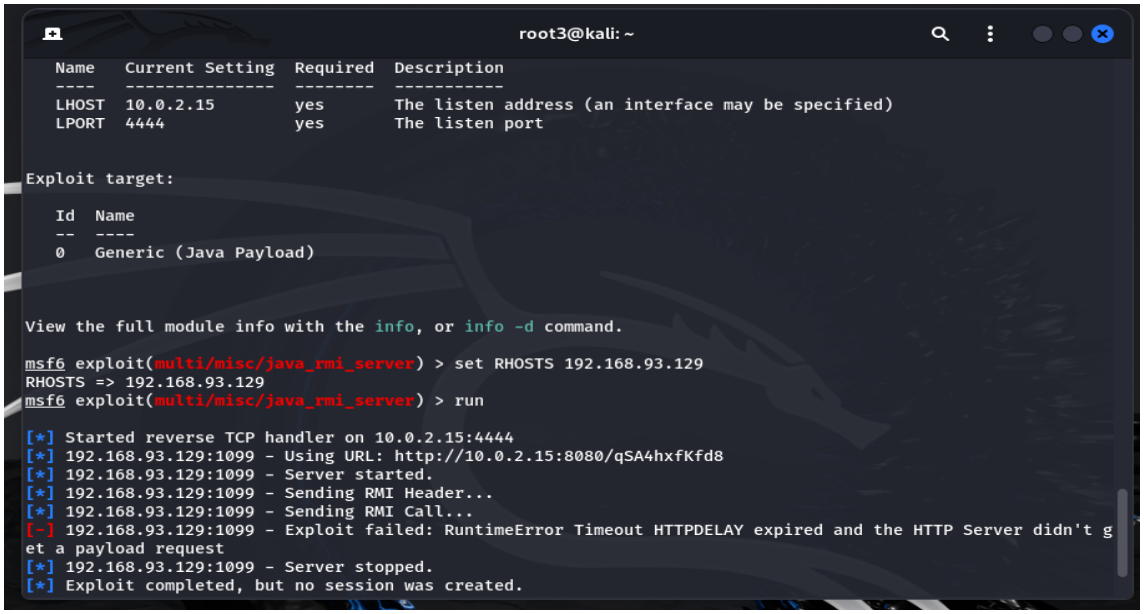


Figure 4.38: Exploiting Java RMI

(d) Payload Delivery: The victim machine should initiate a connection to the attacker's HTTP server to retrieve the payload and establish a reverse connection.

3. **Results Analysis and Interpretation** From the output of the Metasploit attack (msf6), we observe the following:

(a) Exploit Details:

- Connection Attempt: The exploit starts successfully by setting up the reverse TCP handler and HTTP server.
- RMI Call Sent: The exploit attempts to send the RMI header and calls to the target on port 1099.

(b) Error: Timeout:

- The attack fails because the HTTP server, which is supposed to deliver the payload to the victim, times out. The message Timeout HTTPDELAY expired and the HTTP Server didn't get a payload request indicates that the target machine didn't make an HTTP request to the attacker's HTTP server (which was supposed to fetch the payload).

4. **Security Measures for Protecting**

- RMI Security: Ensure that RMI services are configured to only accept requests from trusted sources. Use authentication mechanisms such as SSL/TLS and set up proper access control policies for RMI services.
- Limit RMI Access: Restrict the ability to access the RMI server through firewalls or by binding the service to trusted interfaces only.
- Patch Vulnerabilities: Keep Java environments up-to-date by applying the latest patches for known vulnerabilities. Many of these vulnerabilities can be mitigated by simply keeping Java versions current.
- Block unnecessary ports (like 1099 for RMI) from the external network, especially if RMI is not required to be exposed to the internet.
- Implement network-level protections (like intrusion prevention systems and firewalls) to prevent unauthorized access to critical services.

Conclusion

The exploration of vulnerabilities within the Metasploitable 2 environment highlights the critical importance of proactive vulnerability management in cybersecurity. By employing Nessus and Metasploit, this study showcases the pivotal steps involved in identifying, analyzing, and addressing security weaknesses. Nessus proved instrumental in uncovering vulnerabilities stemming from outdated software, misconfigurations, and exposed services, offering a clear picture of potential attack vectors. Metasploit's practical application demonstrated how these weaknesses can be exploited, emphasizing the consequences of neglecting proper security practices.

The findings underline that a comprehensive and proactive approach is necessary for effective defense. This includes implementing regular vulnerability scans, applying timely patches, securing system configurations, and monitoring network activities for unusual behavior. Moreover, the results stress the importance of integrating advanced tools, continuous education, and hands-on training into cybersecurity strategies to counteract the evolving tactics of malicious actors.

Ultimately, this report reaffirms that maintaining a resilient cybersecurity posture requires not just technical measures but also fostering a culture of vigilance and preparedness. As cyber threats grow in complexity, the integration of proactive risk management and a commitment to staying informed will be key to safeguarding digital infrastructures effectively.

Webography & Bibliography

[1] Claranet. *Scans de vulnérabilités — Claranet*. https://www.claranet.com/fr/expertises/cyber-securite/consulting/scans-de-vulnerabilites. [Consulté le 11 novembre 2024]. 2024.

[2] ManageEngine. *Scanneur de vulnérabilités — ManageEngine*. https://www.manageengine.fr/produits/gestion-des-vulnerabilites/fonctionnalites/scanneur-de-vulnerabilites.html. [Consulté le 11 novembre 2024]. 2024.

[3] *Metasploit Help*. https://help.metasploit.com/Content/home.html. Accessed: 2024-12-10. 2020.

[4] *Metasploitable 2 Documentation*. https://docs.rapid7.com/metasploit/metasploitable-2/. Accessed: 2024-12-10. 2020.

[5] Mandeep Singh et al. "Penetration Testing on Metasploitable 2". In: *International Journal of Engineering and Computer Science* 9 (May 2020), pp. 25014–25022. DOI: 10.18535/ijecs/v9i05.4476.