MIRACLE UKWA

🔐 Simulated Threat & Response: Port Scanning Detection

This document details a simulated port scanning attack and the defensive measures implemented, providing a practical understanding of network security principles.

1. Network Design

For this simulation, a basic virtual network environment was established:

🖥️ **Victim System:** An Ubuntu virtual machine, acting as the target for the simulated attack.

🦹 **Attacker System:** A Kali Linux virtual machine, used to launch the port scan.

🌐 **Connection:** Both virtual machines were configured to reside on the same virtual network, facilitating direct communication and attack simulation.

🛡️ Defense Tools Installed:

Snort: An open-source Intrusion Detection System (IDS) used for real-time traffic analysis and alert generation.

UFW (Uncomplicated Firewall): A user-friendly interface for managing Netfilter firewall rules on Linux, employed to control network access.

This isolated setup allowed for safe and controlled simulation of a cyber threat and the testing of defensive responses without impacting a live production environment.

2. The Threat: Port Scanning

🔍 What Happened

A port scanning attack was simulated using Nmap, a powerful network discovery and security auditing tool. The objective of this scan was to identify open ports on the victim system.

💥 Why It’s Dangerous

Open ports signify services listening for incoming connections. If these services are vulnerable or misconfigured, they can be exploited by attackers to gain unauthorized access, exfiltrate data, or disrupt operations. Port scanning is a fundamental reconnaissance step in most cyber attacks, providing attackers with crucial information about potential entry points.

🛠️ Attack Command

The following Nmap command was executed from the Kali Linux attacker system:

nmap -sS 10.0.2.15

This command performed a SYN scan (-sS), also known as a "half-open" scan. This technique is often preferred by attackers due to its stealthy nature, as it does not complete the full TCP three-way handshake, making it less likely to be logged by the target system.

3. Tools Used and Their Functions

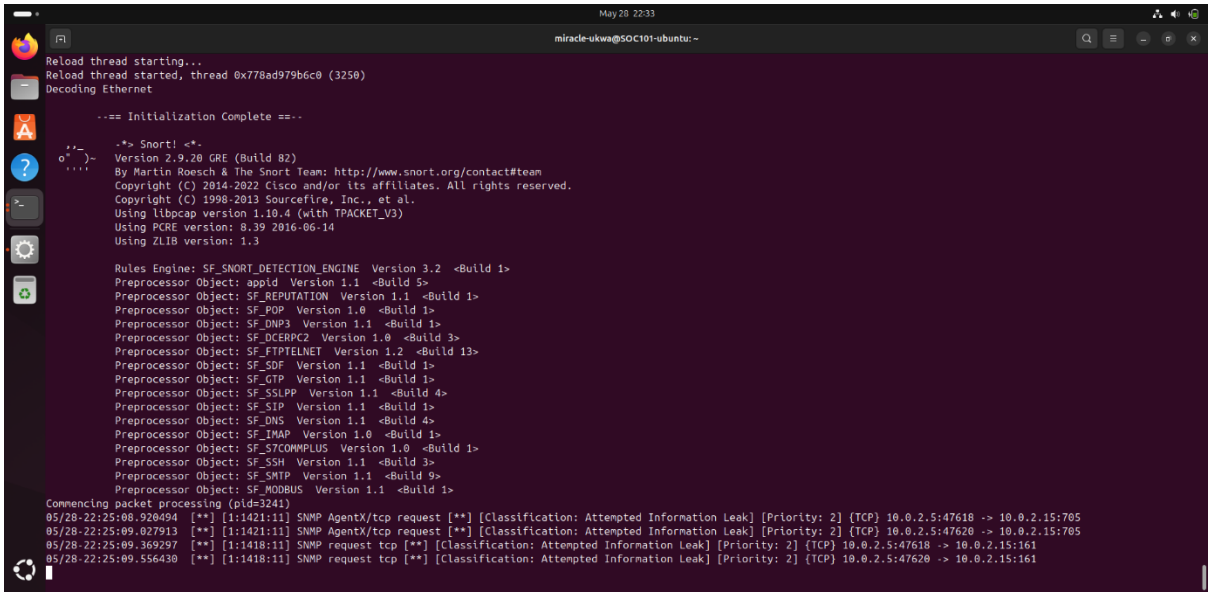
Tool	Purpose
Nmap	Simulated the port scanning attack.
Snort	Detected suspicious scanning behavior.
UFW	Configured firewall rules to block/limit ports.
Wireshark	Captured network packets for detailed analysis of the scan.

4. Controls Used to Detect/Stop the Attack

☑ Detection

Snort Configuration: Snort was configured with specific rules designed to identify patterns indicative of a port scan. This involved monitoring for a high number of connection attempts from a single source within a short time frame, which is characteristic of Nmap's scanning methodology.

Alert Generation: Upon execution of the Nmap scan, Snort successfully detected the suspicious activity and generated an alert, indicating a potential port scan.



☑ Prevention

UFW Configuration: UFW (Uncomplicated Firewall) was enabled and configured to implement a strict "deny by default" policy for incoming connections. This means that unless a specific port is explicitly allowed, all incoming traffic to that port will be blocked.

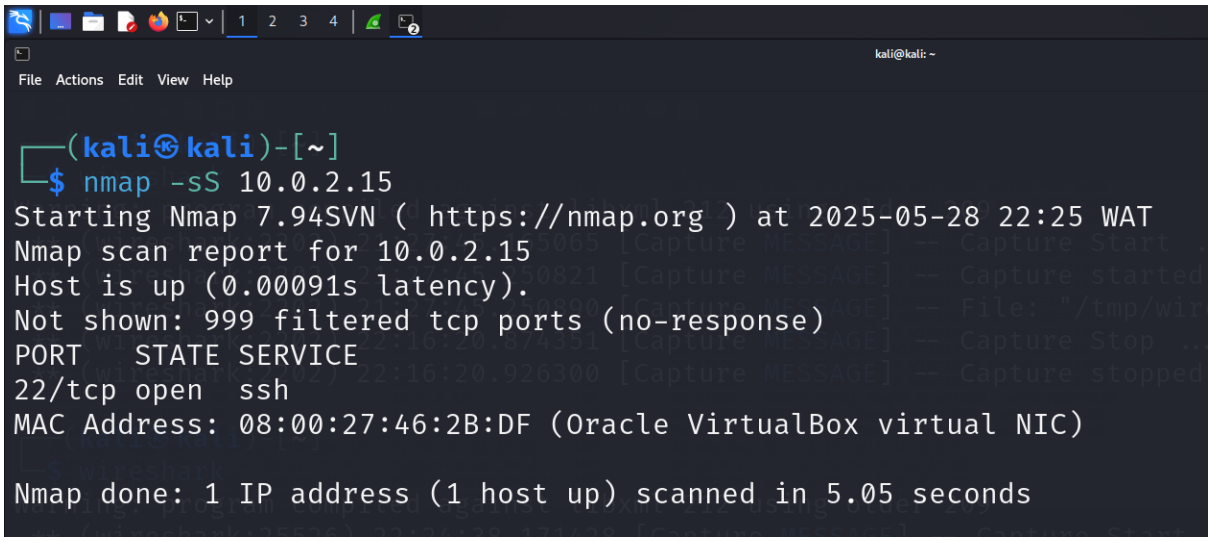
sudo ufw default deny incoming

sudo ufw allow 22/tcp

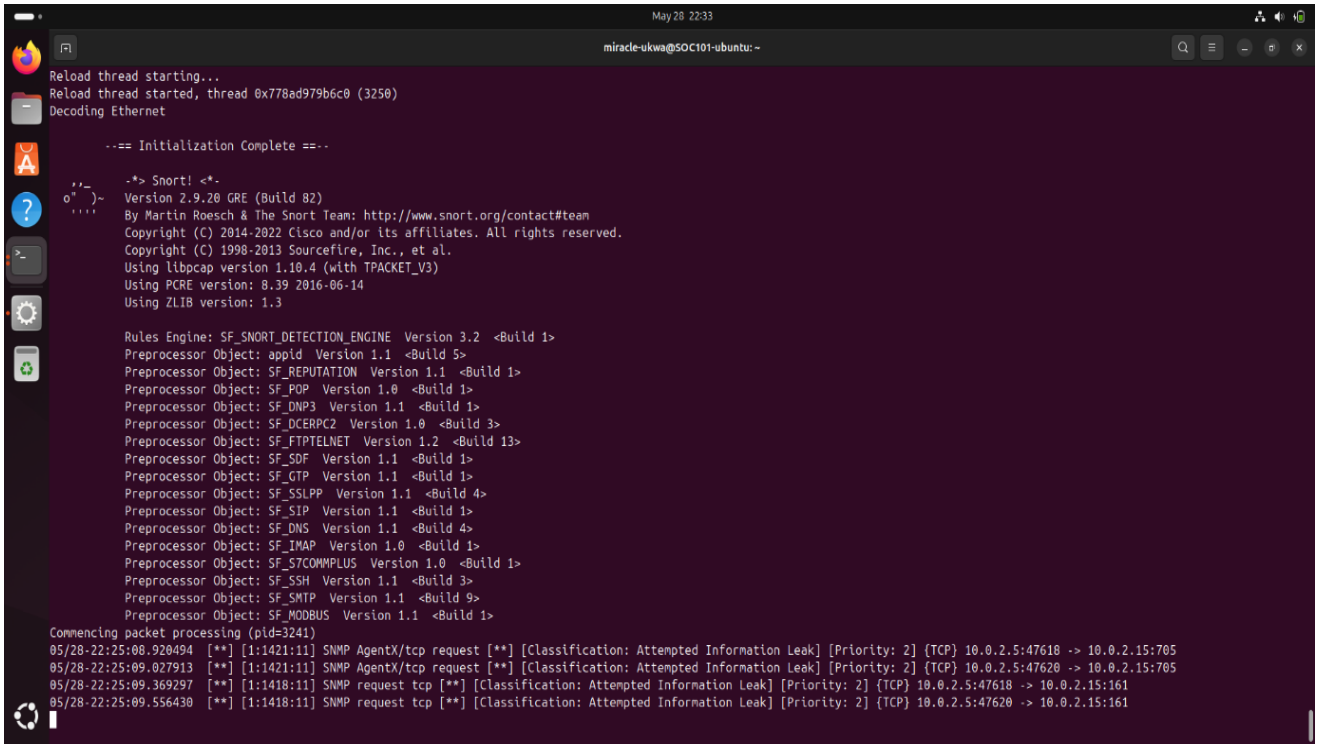
Port Restriction: By allowing only SSH (port 22/tcp) and denying all other incoming connections, the attack surface of the victim system was significantly reduced. This ensures that only essential services, explicitly permitted, are reachable from the network.

5. Screenshots & Logs

Nmap:



Snort Alert Logs:



```
miracle-ukwa@SOC101-ubuntu:~$ sudo ufw status
Status: inactive
miracle-ukwa@SOC101-ubuntu:~$ sudo apt install ufw
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ufw is already the newest version (0.36.2-6).
ufw set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 101 not upgraded.
miracle-ukwa@SOC101-ubuntu:~$ sudo ufw default deny incoming
[sudo] password for miracle-ukwa:
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
miracle-ukwa@SOC101-ubuntu:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
miracle-ukwa@SOC101-ubuntu:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
miracle-ukwa@SOC101-ubuntu:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

6. Reflection: What I Learned

This simulation provided valuable insights into the attacker's perspective and the importance of layered security. I gained a practical understanding of how attackers leverage tools like Nmap for reconnaissance and how detection systems such as Snort can effectively identify these activities. Furthermore, the exercise reinforced the critical role of firewalls in minimizing the attack surface by controlling network access to system services.

Moving forward, to enhance my understanding of network security, I would explore more advanced attack vectors, such as exploiting known vulnerabilities or performing denial-of-service attacks. Additionally, I plan to delve into Suricata, another powerful open-source IDS/IPS, to compare its capabilities with Snort and explore its deeper analysis features.

Network Setup Checklist

Here’s a quick guide to protect a small office/home system:

Disable unused ports/services: Minimize the attack surface by shutting down any network services or applications that are not essential.

Use a firewall (e.g., UFW or Windows Defender Firewall): Implement a firewall to filter incoming and outgoing network traffic, allowing only necessary connections.

Monitor with an IDS like Snort or Suricata: Deploy an Intrusion Detection System to actively monitor network traffic for suspicious patterns and alert on potential threats.

Log and review suspicious activity: Maintain comprehensive logs of network events and regularly review them for any indicators of compromise or unusual behavior.

Keep your OS and tools updated: Regularly apply security patches and updates to your operating systems and security tools to protect against known vulnerabilities.