**Scanning Local Network for Open Ports**

The objective of this task was to perform network enumeration on a local machine using Nmap, one of the most widely used network scanning tools. The scan was aimed at identifying open ports and detecting the services running on those ports.

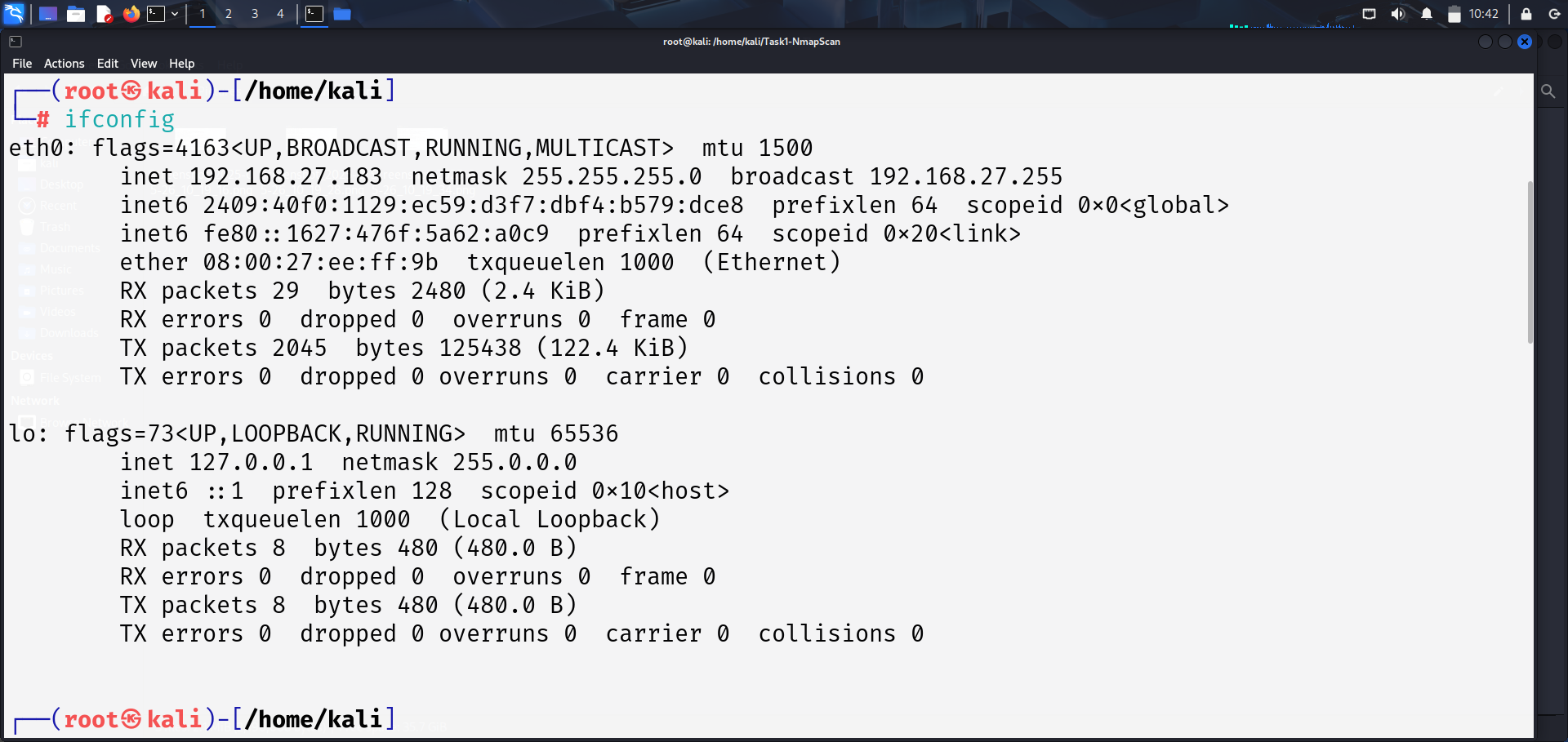
**Steps Followed**

1. **Identified My Local IP Address**

Before running the scan, I needed to know the IP address of a machine in my local network. To find my own local IP, I used:

ifconfig

This showed me my network interface details. I noted down the IP in the same subnet to target during the scan.



1. **Performed Nmap Stealth & Version Scan**

I used the following command to run a **stealth scan (-sS)** and gather **service version information (-sV)**, and saved the output to a file using -oN:

nmap -sS -sV 192.168.27.183/24 -oN nmap\_scan\_result.txt

**Explanation of the command:**

* -sS: Performs a **stealth (SYN) scan**, which is less likely to be detected by firewalls and intrusion detection systems.
* -sV: Enables version detection to identify which services and versions are running on open ports.
* -oN: Saves the output of the scan into a text file called nmap\_scan\_result.txt for documentation and further analysis.

1. **Step 3: Output and Results**

After running the above command, Nmap scanned the target system and produced a list of **open ports**, along with the services running on those ports and their version information.

The results were saved in a file:

nmap\_scan\_result.txt

Example output includes:

PORT STATE SERVICE VERSION

8008/tcp open http?

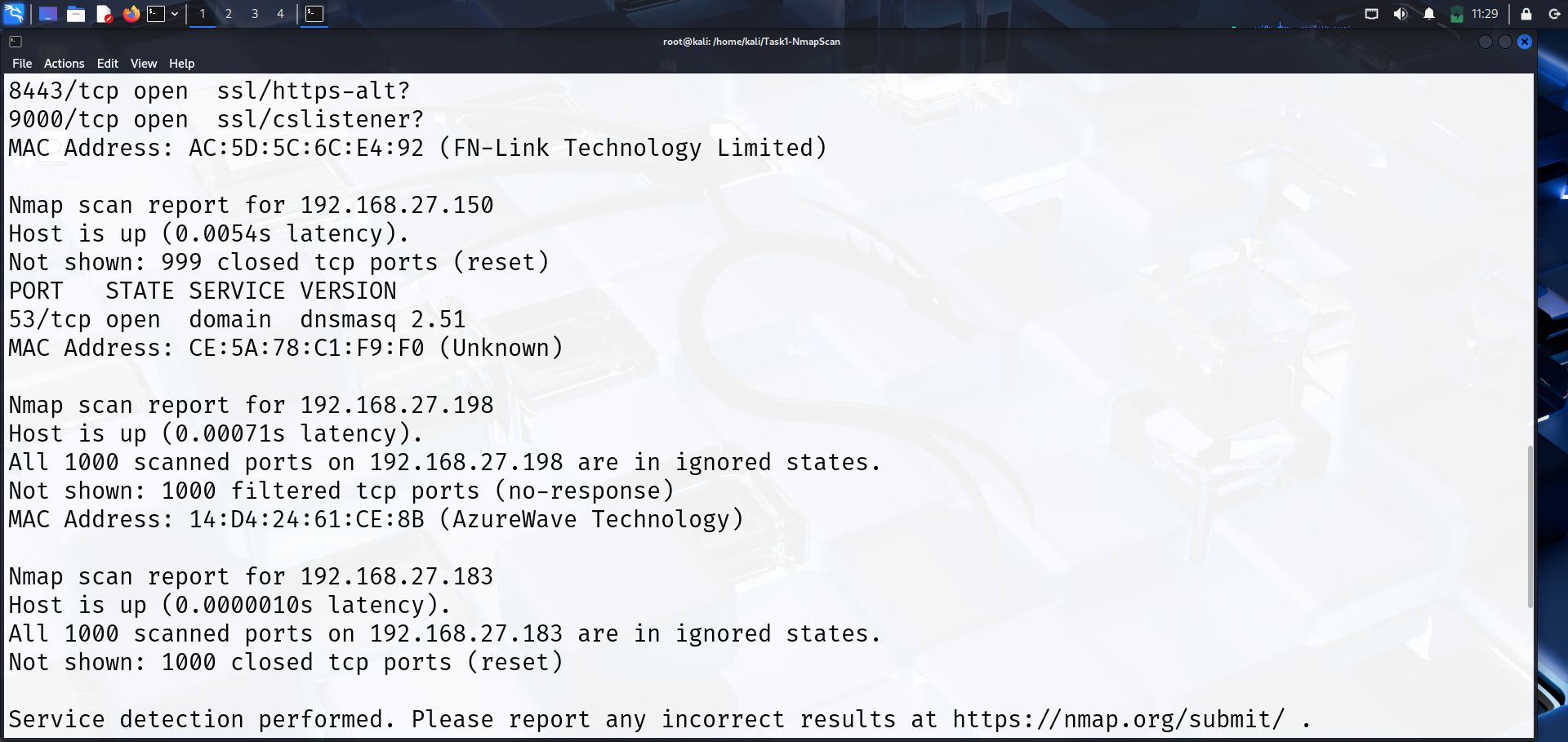
8009/tcp open ssl/castv2 Ninja Sphere Chromecast driver

8443/tcp open ssl/https-alt?

9000/tcp open ssl/cslistener?

53/tcp open domain dnsmasq 2.51





**Common services running on those ports and Potential security risks from open ports.**

**Port 8008/tcp — http?**

* **Common Use**: Web interface for **Google Chromecast**, smart devices, or IoT apps.
* **Service**: Unknown (http? with a question mark means Nmap thinks it might be HTTP but isn't sure).
* **Explanation**: Port 8008 is often used by **Chromecast or other streaming devices** to serve a local web UI.
* **Risk:** If this web interface is exposed without proper authentication, attackers could access or control the device.
* **Possible Attack:** Unauthorized access to smart devices, data leaks, or device hijacking.

**Port 8009/tcp — ssl/castv2**

* **Common Use**: Chromecast communication using **CastV2 protocol** over SSL.
* **Service**: Ninja Sphere Chromecast driver (detected by Nmap).
* **Explanation**: This port is used for **secure communication** between the Chromecast device and apps like YouTube or Netflix.
* **Risk:** Even though it uses SSL, vulnerabilities in the Chromecast or its drivers could allow attackers to intercept or manipulate communication.
* **Possible Attack:** Man-in-the-middle (MITM) attacks or exploiting device-specific flaws.

**Port 8443/tcp — ssl/https-alt?**

* **Common Use**: Alternative HTTPS web service port.
* **Service**: Likely HTTPS, but Nmap is unsure (https-alt?).
* **Explanation**: Web servers sometimes run their **secure login/admin panels** on port 8443 instead of the default 443 to reduce risk.
* **Risk:** If the web interface on this port has weak authentication or unpatched vulnerabilities, attackers can exploit it to gain admin access.
* **Possible Attack:** Remote code execution, data theft, or configuration changes.

**Port 9000/tcp — ssl/cslistener?**

* **Common Use**: Used by various apps for **remote command/control** or custom applications (like SonarQube, Metasploit listeners, etc.).
* **Service**: cslistener? — possibly a Command & Control (C2) listener.
* **Explanation**: This port might be used by a **listener waiting for connections**, often in malware analysis or pen-testing tools like **Cobalt Strike**.
* **Risk:** This port might be used by software that listens for remote commands. If improperly secured, it can be exploited to execute malicious commands.
* **Possible Attack:** Remote code execution, backdoor installation, or unauthorized system control.

**Port 53/tcp — domain (dnsmasq 2.51)**

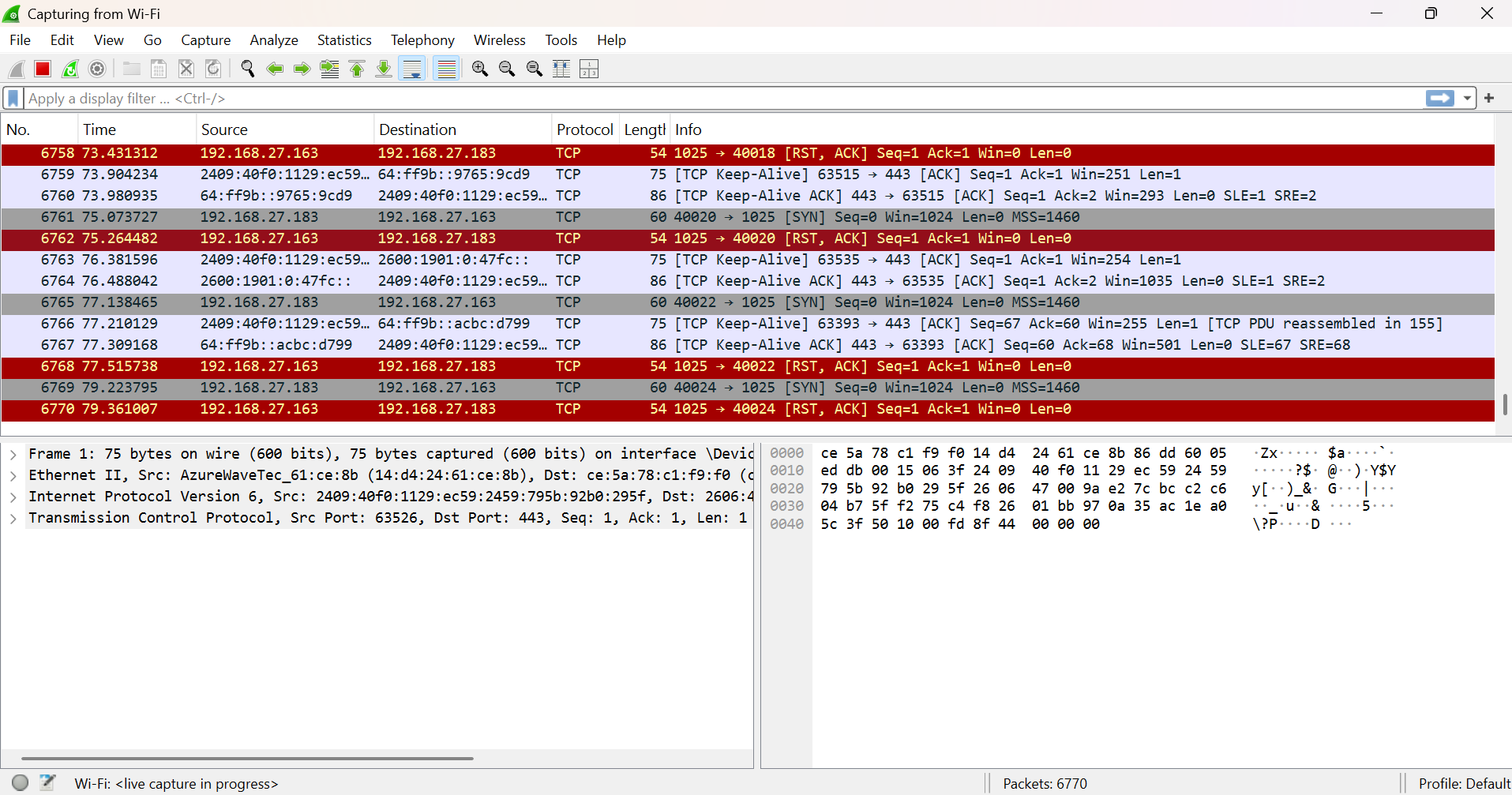
* **Common Use**: **DNS (Domain Name System)** service.
* **Service**: dnsmasq 2.51
* **Explanation**: This is a **DNS resolver and DHCP server**, often used in small routers and devices to handle internet name resolution.
* **Risk:** DNS servers are critical infrastructure. If dnsmasq is misconfigured or outdated, it can be vulnerable to DNS spoofing, cache poisoning, or denial of service.
* **Possible Attack:** Redirecting users to malicious sites, intercepting traffic, or disrupting network services.

**Using Wireshark for Packet Capture**

Wireshark is a powerful tool that captures and analyzes network traffic in real-time. In this task, I used Wireshark to capture **TCP packets** between my machine and the target IP during the Nmap scan.

**Why Wireshark is Useful Here**

* **Monitor Traffic:** It shows the details of every packet sent and received, helping to understand how the scan probes the target.
* **Analyze Protocols:** It breaks down TCP packets to show flags, sequence numbers, and data, which helps in understanding the handshake and scanning methods.
* **Verify Scan Behavior:** By seeing the actual packets, I could confirm that the SYN scan (-sS) sent SYN packets without completing the full connection, making it a stealthy scan.
* **Detect Anomalies:** If any suspicious or unexpected traffic appears, Wireshark helps spot potential security issues or active defenses on the target.



Wireshark complements Nmap by providing a low-level view of network communication. It is an essential tool for network troubleshooting, security analysis, and learning how scanning techniques work.