# 📄 Network Packet Sniffer & Analyzer – Project Report

## 🧠 Objective

To build a Python-based tool that captures network packets from a specified interface in real time, analyzes the packet data (IP, TCP, UDP), and detects potential suspicious behavior such as port scans. The tool saves captured traffic to a `.pcap` file for further offline analysis.

## 🔧 Tools & Technologies Used

- **Programming Language**: Python
- **Libraries**: Scapy (packet sniffing and analysis), CSV
- **OS**: Windows 11
- **Environment**: Anaconda (custom virtual environment)
- **Packet Capture Interface**: Npcap (WinPcap-compatible mode)

## 📦 Project Components

### 1. `sniffer.py` – Packet Capture Script

- Captures packets from a user-selected network interface using Scapy.
- Filters to only capture **IP packets**.
- Saves captured data to a `.pcap` file (`packets.pcap`) using `wrpcap()`.

### 2. `analyzer.py` – Packet Analysis Script

- Reads the saved `packets.pcap` file using Scapy's `rdpcap()`.
- Displays a summary for each IP packet including:
    - Source and Destination IP
    - Protocol (TCP or UDP)
    - Source and Destination Ports
- Detects **potential port scans** by identifying IPs that send packets to multiple ports.

## 📊 Sample Output

```
[+] IP Packet: 10.14.146.90 -> 20.195.84.16 | Protocol: 6
    TCP Port: 51939 -> 443
[+] IP Packet: 10.14.146.215 -> 224.0.0.251 | Protocol: 17
    UDP Port: 5353 -> 5353
[!] Possible Port Scan Detected from 192.168.1.100 on ports: [22, 23, 80, 443, 8080, ...]
```