NetProbe

Advanced Python Network Sniffer

USER MANUAL

Version 1.0

⚠ **WARNING:** This tool is for AUTHORIZED security testing and educational purposes ONLY. Unauthorized network sniffing is illegal and may result in criminal prosecution. Always obtain written permission before use.

1. Introduction

NetProbe is a pure-Python network packet sniffer designed for cybersecurity professionals, penetration testers, and students. It captures and analyzes live network traffic at the raw socket level, providing deep visibility into Ethernet frames, IP packets, TCP/UDP segments, ICMP messages, ARP packets, and DNS queries.

The tool requires no third-party dependencies — it uses only Python's standard library, making it lightweight and easy to deploy on any Linux system.

2. System Requirements

Requirement	Details
Operating System	Linux (Parrot OS, Kali, Ubuntu, Debian)
Python	3.8 or higher
Privileges	Root / sudo access required
Dependencies	None (standard library only)
Network	Active network interface (Ethernet or Wi-Fi)

3. Installation

3.1 Clone the Repository

```
git clone https://github.com/Michael-Chileshe/cybersec-tools.git
cd cybersec-tools/network-sniffer
```

3.2 Make Executable

```
chmod +x sniffer.py
```

3.3 Verify Python Version

```
python3 --version
```

Ensure the output shows Python 3.8 or higher.

4. Command-Line Options

4.1 General Options

Flag	Argument	Description
-i, --interface	INTERFACE	Network interface to listen on (e.g. eth0, wlan0). Default: all interfaces

-c, --count	NUMBER	Stop after capturing N packets
-v, --verbose	(none)	Increase output detail. Use -v for normal, -vv for full hex dump
-q, --quiet	(none)	Suppress packet output; only show summary at end
-o, --output	FILEPATH	Save packets to file. Format auto-detected: .json, .csv, .txt
--promisc	(none)	Enable promiscuous mode on the interface

4.2 Filter Options

Flag	Argument	Description
-p, --protocol	PROTOCOL	Filter by protocol: TCP, UDP, ICMP, ARP
--src-ip	IP_ADDRESS	Only show packets from this source IP
--dst-ip	IP_ADDRESS	Only show packets to this destination IP
--ip	IP_ADDRESS	Match packets where either source or dest equals this IP
--src-port	PORT	Filter by source port number
--dst-port	PORT	Filter by destination port number
--port	PORT	Match packets on this port (either direction)

5. Usage Examples

5.1 Basic Capture

Capture all traffic on all interfaces:

```
sudo python3 sniffer.py
```

5.2 Interface-Specific Capture

```
sudo python3 sniffer.py -i eth0
sudo python3 sniffer.py -i wlan0
```

5.3 Protocol Filtering

```
sudo python3 sniffer.py -p TCP
sudo python3 sniffer.py -p UDP
sudo python3 sniffer.py -p ICMP
sudo python3 sniffer.py -p ARP
```

5.4 IP and Port Filtering

```
sudo python3 sniffer.py --ip 192.168.1.1
sudo python3 sniffer.py --src-ip 10.0.0.5 --dst-port 80
sudo python3 sniffer.py -p TCP --port 443
```

5.5 Combining Filters

All filter criteria are combined with AND logic:

```
sudo python3 sniffer.py -p TCP --ip 192.168.1.100 --port 80 -v
```

5.6 Verbose and Hex Dump

```
sudo python3 sniffer.py -v          # Multi-line details
sudo python3 sniffer.py -vv         # Full hex dump + payload
```

5.7 Exporting Data

```
sudo python3 sniffer.py -o capture.json   # JSON
sudo python3 sniffer.py -o capture.csv    # CSV
sudo python3 sniffer.py -o capture.txt    # Text
```

5.8 Limited Capture with Export

```
sudo python3 sniffer.py -c 200 -o results.json -q
```

Captures exactly 200 packets, saves to JSON, and only displays the summary at the end.

5.9 Promiscuous Mode

```
sudo python3 sniffer.py -i eth0 --promisc
```

⚠ **WARNING:** Promiscuous mode captures traffic not addressed to your machine. Only use this on networks you own or have written authorization to test.

6. Real-World Scenarios

6.1 DNS Monitoring

Monitor which domains are being resolved on the network:

```
sudo python3 sniffer.py -p UDP --port 53 -v -o dns_log.json
```

6.2 Detecting SSH Brute Force

Watch for repeated SSH connection attempts:

```
sudo python3 sniffer.py -p TCP --dst-port 22 -vv
```

Look for many SYN packets from the same source IP in the session summary.

6.3 HTTP Traffic Inspection

Observe unencrypted HTTP traffic including request URIs:

```
sudo python3 sniffer.py -p TCP --port 80 -vv
```

6.4 ARP Poisoning Detection

Watch for unusual ARP activity that could indicate MITM attacks:

```
sudo python3 sniffer.py -p ARP -v
```

The tool will flag ARP storms in the suspicious activity section of the summary.

6.5 Network Reconnaissance Baseline

Capture a baseline of all network traffic for analysis:

```
sudo python3 sniffer.py -c 5000 -o baseline.json -q
```

7. Understanding the Output

7.1 Compact Mode (Default)

Each packet is shown on a single line with color coding:

```
    1 14:23:01.456 TCP   192.168.1.5:54321 → 93.184.216.34:443 (HTTPS)
[SYN]
```

Green = TCP, Blue = UDP, Magenta = ARP/ICMP, Dim = Other

7.2 Session Summary

When you stop the capture (Ctrl+C), the tool displays a full summary including: total packets and bytes, protocol breakdown with visual bars, top 5 source and destination IPs, top 10 active ports, application protocol counts, and any suspicious activity detected.

7.3 Export File Contents

JSON files contain the full structured data for each packet. CSV files provide a flat table suitable for spreadsheets and data analysis. Text files offer a simple one-line-per-packet log for quick grepping.

8. Anomaly Detection

NetProbe performs lightweight heuristic analysis during capture and flags suspicious patterns:

Detection	What It Means
NULL Scan	TCP packet with no flags set. Used by scanners to evade firewalls.
XMAS Scan	TCP with FIN+PSH+URG flags. Another scanner evasion technique.
ARP Storm	Abnormally high ARP traffic. May indicate ARP spoofing or network loop.
SYN Flood	High rate of SYN-only packets. Signature of denial-of-service attacks.

9. Troubleshooting

"Permission denied" or "Root privileges required" — Run the tool with sudo: sudo python3 sniffer.py

"Socket error: [Errno 19] No such device" — The specified interface does not exist. Check available interfaces with: ip link show

No packets captured — Verify you are on the correct interface. Try without the -i flag to listen on all interfaces. Ensure the network is active.

Promiscuous mode warning — Some virtual interfaces or Wi-Fi drivers may not support promiscuous mode. The tool will warn but continue in normal mode.

Garbled output — Ensure your terminal supports ANSI color codes. Most modern terminals do, but some remote sessions may not.

10. Legal Notice

> ⚠ **WARNING:** Unauthorized interception of network communications is a criminal offense in virtually all jurisdictions. Penalties include imprisonment, fines, and civil liability. ALWAYS obtain explicit written authorization from the network owner before using this tool. You are solely responsible for ensuring legal compliance.

This tool is distributed under the MIT License for authorized security testing and educational use only. The authors accept no liability for misuse.