🔒 Packet Sniffer & Analyzer - Final Year Project

A powerful, real-time network packet capturing and analysis tool built with Python and web technologies.

📋 Table of Contents

🎯 Overview

This Packet Sniffer & Analyzer captures and analyzes network packets in real-time, providing detailed insights into network traffic. It's designed as a lightweight alternative to professional tools like Wireshark, with an intuitive web interface.

Key Capabilities

- Real-time packet capture from network interfaces
- Protocol identification (TCP, UDP, HTTP, DNS, ICMP)
- Live statistics and visualization
- Packet filtering and searching
- Export functionality (CSV format)

✨ Features

Core Features

1. **Live Packet Capture**
   - Capture packets from any network interface (Ethernet, WiFi, Loopback)
   - Real-time display of packet information
   - Support for multiple protocols
2. **Detailed Packet Information**
   - Source and Destination IP addresses
   - Protocol type (TCP, UDP, HTTP, DNS, ICMP)
   - Port numbers
   - Packet size
   - Timestamp
3. **Advanced Filtering**
   - Filter by protocol type
   - Search by IP address
   - Search by port number
   - Real-time filtering without stopping capture
4. **Statistics Dashboard**

    - Total packets captured
    - Data transferred (KB)
    - Capture duration
    - Packets per second rate
    - Protocol distribution with visual bars
5. **Data Export**
    - Export captured packets to CSV format
    - Timestamped filenames
    - All packet details included
6. **User-Friendly Interface**
    - Modern, responsive web UI
    - Real-time updates
    - Color-coded protocols
    - Smooth animations and transitions

🛠️ Tech Stack

Backend

- **Python 3.7+**
- **Scapy** - Packet manipulation and capture
- **Flask** - Web framework
- **Flask-CORS** - Cross-origin resource sharing

Frontend

- **HTML5** - Structure
- **CSS3** - Styling with modern gradients and animations
- **JavaScript (Vanilla)** - Dynamic functionality and API integration

📦 Installation

Prerequisites

- Python 3.7 or higher
- Administrator/Root privileges (required for packet capture)
- pip (Python package manager)

Step 1: Install Python Dependencies

bash

```
pip install scapy flask flask-cors
```

Or using requirements.txt:

bash

```
pip install -r requirements.txt
```

Step 2: Download Project Files

Ensure you have the following files in your project directory:

- `packet_sniffer.py` - Standalone CLI version
- `app.py` - Flask web server
- `packet_sniffer_ui_connected.html` - Web interface

Step 3: Verify Installation

bash

```
python3 -c "import scapy; import flask; print('All dependencies installed!')"
```

🚀 Usage

Method 1: Web Interface (Recommended)

1. **Start the Flask server** (with admin privileges): **Linux/Mac:**

bash

```
sudo python3 app.py
```

Windows (Run as Administrator):

cmd

```
python app.py
```

2. **Access the web interface:**
   - Open your browser
   - Navigate to: `http://localhost:5000`
3. **Start Capturing:**
   - Select network interface (or use default)
   - Click "Start Capture"

- Monitor real-time packets
- Apply filters as needed
- Export data when done

Method 2: Command Line Interface

1. **Run the CLI version** (with admin privileges): **Linux/Mac:**

bash

```
sudo python3 packet_sniffer.py
```

Windows (Run as Administrator):

cmd

```
python packet_sniffer.py
```

2. **Available Commands:**
   - `start` - Start packet capture
   - `stop` - Stop packet capture
   - `show` - Display captured packets
   - `stats` - Show statistics
   - `save` - Save to CSV/JSON
   - `clear` - Clear captured packets
   - `exit` - Exit program

📁 Project Structure

```
packet-sniffer-project/
│
├── packet_sniffer.py            # Standalone CLI version
├── app.py                       # Flask web server
├── packet_sniffer_ui_connected.html # Web UI (backend-connected)
├── packet_sniffer_ui.html       # Web UI (demo version)
├── requirements.txt             # Python dependencies
├── README.md                    # This file
│
└── exports/                     # (Created automatically)
    ├── packet_capture_*.csv     # Exported packet data
    └── packet_capture_*.json    # Exported packet data
```

🖼️ Screenshots

Main Dashboard

- Live packet stream table
- Real-time statistics cards
- Protocol distribution charts

Features in Action

- Color-coded protocol badges
- Interactive filtering
- Search functionality
- Export capabilities

🔧 Troubleshooting

Common Issues

1. Permission Denied Error

Problem: `PermissionError: Operation not permitted`

Solution:

- Linux/Mac: Run with `sudo`
- Windows: Run terminal as Administrator

2. Module Not Found

Problem: `ModuleNotFoundError: No module named 'scapy'`

Solution:

bash

```
pip install scapy flask flask-cors
```

3. No Packets Captured

Problem: Capture starts but no packets appear

Solution:

- Verify you selected the correct network interface
- Check if interface has active traffic
- Try using default interface
- Ensure firewall isn't blocking

4. Port 5000 Already in Use

Problem: `Address already in use`

Solution:

- Change port in `app.py`: `app.run(port=5001)`
- Or kill existing process:
  - Linux/Mac: `sudo lsof -ti:5000 | xargs kill -9`
  - Windows: `netstat -ano | findstr :5000` then `taskkill /PID <PID> /F`

5. Cannot Capture HTTPS Content

Problem: HTTPS packets show encrypted data

Solution:

- This is expected behavior - HTTPS is encrypted
- You can see IP addresses, ports, and packet sizes
- Cannot decrypt content without SSL/TLS keys

🎓 Educational Value

Learning Outcomes

1. **Networking Fundamentals**
   - TCP/IP protocol stack
   - Network layers (OSI model)
   - Packet structure and headers
2. **Cybersecurity Concepts**
   - Network monitoring
   - Traffic analysis
   - Protocol identification
   - Security implications
3. **Programming Skills**

- Python network programming
- Web development (Flask, HTML/CSS/JS)
- API design
- Real-time data handling

4. **Tools & Libraries**
   - Scapy for packet manipulation
   - Flask for web services
   - JavaScript for dynamic UIs

Future Enhancements

Planned Features

1. **Advanced Analytics**
   - Geographic IP visualization
   - Traffic graphs and charts
   - Bandwidth monitoring
   - Anomaly detection
2. **Security Features**
   - Intrusion Detection System (IDS)
   - Suspicious pattern recognition
   - Alert notifications
   - Blacklist/Whitelist management
3. **Enhanced Filtering**
   - Custom filter rules
   - Regular expression support
   - Save filter presets
4. **Database Integration**
   - Store packet history
   - Query historical data
   - Generate reports
5. **Additional Export Formats**
   - JSON export
   - PCAP file format
   - PDF reports
6. **Performance Improvements**
   - Multi-threaded capture
   - Packet buffering
   - Memory optimization

Requirements File

Create `requirements.txt`:

```
scapy>=2.4.5
flask>=2.0.0
flask-cors>=3.0.10
```

⚖️ Legal & Ethical Considerations

Important Notes

- **Only use on networks you own or have permission to monitor**
- Unauthorized packet sniffing may be illegal in your jurisdiction
- This tool is for educational purposes only
- Respect privacy and data protection laws
- Use responsibly and ethically

Ethical Guidelines

1. Always obtain proper authorization
2. Don't capture sensitive personal information
3. Secure exported data properly
4. Follow organizational policies
5. Use for learning and legitimate network troubleshooting only

👥 Project Information

Author

- **Project Type:** Final Year Project
- **Domain:** Cybersecurity & Networking
- **Level:** Undergraduate

Supervisor Guidelines

This project demonstrates:

- Strong understanding of network protocols
- Full-stack development capabilities
- Security awareness
- Professional documentation
- Real-world application potential

📞 Support

For issues or questions:

1. Check the Troubleshooting section
2. Review Scapy documentation: https://scapy.readthedocs.io/
3. Review Flask documentation: https://flask.palletsprojects.com/

License

This project is created for educational purposes. Please respect network privacy and legal requirements in your jurisdiction.

Happy Packet Sniffing! 🔍📡

Remember: With great power comes great responsibility. Use this tool ethically and legally!