Packet sniffing project

Python code for this mto develop this one

**# packet-sniffer/app.py**

from scapy.all import sniff, IP, TCP, UDP, ICMP

from datetime import datetime

import argparse

def proto\_of(pkt):

if TCP in pkt: return "TCP"

if UDP in pkt: return "UDP"

if ICMP in pkt: return "ICMP"

if IP in pkt: return f"IP proto {pkt[IP].proto}"

return pkt.name

def handle(pkt):

ts = datetime.now().strftime("%H:%M:%S")

if IP in pkt:

print(f"[{ts}] {pkt[IP].src} -> {pkt[IP].dst} | {proto\_of(pkt)} | len={len(pkt)}")

else:

print(f"[{ts}] {pkt.summary()}")

parser = argparse.ArgumentParser(description="Simple packet sniffer (Stage A)")

parser.add\_argument("-i","--iface", default=None, help="Interface, e.g., eth0/wlan0/ens33")

parser.add\_argument("-f","--filter", default="ip", help="BPF filter (e.g., 'tcp', 'udp', 'port 53')")

parser.add\_argument("-c","--count", type=int, default=0, help="Packets to capture (0=infinite)")

args = parser.parse\_args()

sniff(prn=handle, store=False, iface=args.iface, filter=args.filter, count=args.count)

**Advance**

# packet-sniffer/app.py (Stage B: CSV + PCAP)

from scapy.all import sniff, IP, TCP, UDP, ICMP

from scapy.utils import PcapWriter

from datetime import datetime

from pathlib import Path

import argparse, csv

def proto\_of(pkt):

if TCP in pkt: return "TCP"

if UDP in pkt: return "UDP"

if ICMP in pkt: return "ICMP"

if IP in pkt: return f"IP proto {pkt[IP].proto}"

return pkt.name

def main():

parser = argparse.ArgumentParser(description="Packet sniffer with CSV/PCAP logging")

parser.add\_argument("-i","--iface", default=None, help="Interface (e.g., eth0/wlan0/ens33)")

parser.add\_argument("-f","--filter", default="ip", help="BPF filter (e.g., 'tcp', 'udp', 'port 53')")

parser.add\_argument("-c","--count", type=int, default=0, help="Packets to capture (0=infinite)")

args = parser.parse\_args()

# prepare paths

base\_dir = Path(\_\_file\_\_).resolve().parent

log\_dir = base\_dir / "logs"

log\_dir.mkdir(parents=True, exist\_ok=True)

stamp = datetime.now().strftime("%Y%m%d\_%H%M%S")

csv\_path = log\_dir / f"packets\_{stamp}.csv"

pcap\_path = log\_dir / f"packets\_{stamp}.pcap"

# writers

csv\_file = open(csv\_path, "w", newline="")

csv\_writer = csv.writer(csv\_file)

csv\_writer.writerow(["time","src","dst","proto","length"])

pcap = PcapWriter(str(pcap\_path), append=True, sync=True)

print(f"[+] Writing CSV -> {csv\_path}")

print(f"[+] Writing PCAP -> {pcap\_path}")

print(f"[+] Sniffing on iface={args.iface or 'DEFAULT'} filter='{args.filter}' count={args.count or '∞'}")

print("[+] Ctrl+C to stop")

def handle(pkt):

# write PCAP first (every packet)

pcap.write(pkt)

# pretty console line + CSV row if IP present

ts = datetime.now().strftime("%H:%M:%S")

if IP in pkt:

src, dst = pkt[IP].src, pkt[IP].dst

proto = proto\_of(pkt)

length = len(pkt)

print(f"[{ts}] {src} -> {dst} | {proto} | len={length}")

csv\_writer.writerow([ts, src, dst, proto, length])

else:

print(f"[{ts}] {pkt.summary()}")

try:

sniff(prn=handle, store=False, iface=args.iface, filter=args.filter, count=args.count)

except KeyboardInterrupt:

pass

finally:

csv\_file.flush(); csv\_file.close()

print("[+] Stopped. CSV/PCAP saved.")

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

main()

**# day 5 code**

from scapy.all import sniff, IP

import csv

import argparse

import os

# Ensure logs folder exists

os.makedirs("logs", exist\_ok=True)

def packet\_callback(packet):

if IP in packet:

with open("logs/day5\_sniffer.csv", "a", newline="") as f:

writer = csv.writer(f)

writer.writerow([

packet[IP].src,

packet[IP].dst,

packet[IP].proto,

len(packet)

])

print(f"[+] {packet[IP].src} -> {packet[IP].dst} | Proto: {packet[IP].proto} | Len: {len(packet)}")

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

parser = argparse.ArgumentParser()

parser.add\_argument("-i", "--iface", required=True, help="Network interface (e.g., eth0, wlan0, lo)")

parser.add\_argument("-c", "--count", type=int, default=20, help="Number of packets to capture")

args = parser.parse\_args()

# Initialize CSV file with headers

with open("logs/day5\_sniffer.csv", "w", newline="") as f:

writer = csv.writer(f)

writer.writerow(["Source IP", "Destination IP", "Protocol", "Length"])

print(f"[!] Starting sniffer on {args.iface}, capturing {args.count} packets...")

sniff(iface=args.iface, prn=packet\_callback, count=args.count)

print("[✔] Capture finished. Saved to logs/day5\_sniffer.csv")

Day 6 code

# packet-sniffer/app.py (Stage B: CSV + PCAP)

from scapy.all import sniff, IP, TCP, UDP, ICMP

from scapy.utils import PcapWriter

from datetime import datetime

from pathlib import Path

import argparse, csv

def proto\_of(pkt):

if TCP in pkt: return "TCP"

if UDP in pkt: return "UDP"

if ICMP in pkt: return "ICMP"

if IP in pkt: return f"IP proto {pkt[IP].proto}"

return pkt.name

def main():

parser = argparse.ArgumentParser(description="Packet sniffer with CSV/PCAP logging")

parser.add\_argument("-i","--iface", default=None, help="Interface (e.g., eth0/wlan0/ens33/lo)")

parser.add\_argument("-f","--filter", default="ip", help="BPF filter (e.g., 'tcp', 'udp', 'port 53')")

parser.add\_argument("-c","--count", type=int, default=0, help="Packets to capture (0=infinite)")

args = parser.parse\_args()

# prepare paths

base\_dir = Path(\_\_file\_\_).resolve().parent

log\_dir = base\_dir / "logs"

log\_dir.mkdir(parents=True, exist\_ok=True)

stamp = datetime.now().strftime("%Y%m%d\_%H%M%S")

csv\_path = log\_dir / f"packets\_{stamp}.csv"

pcap\_path = log\_dir / f"packets\_{stamp}.pcap"

# writers

csv\_file = open(csv\_path, "w", newline="")

csv\_writer = csv.writer(csv\_file)

csv\_writer.writerow(["time","src","dst","proto","length"])

pcap = PcapWriter(str(pcap\_path), append=True, sync=True)

print(f"[+] Writing CSV -> {csv\_path}")

print(f"[+] Writing PCAP -> {pcap\_path}")

print(f"[+] Sniffing on iface={args.iface or 'DEFAULT'} filter='{args.filter}' count={args.count or '∞'}")

print("[+] Ctrl+C to stop")

def handle(pkt):

# write PCAP first (every packet)

pcap.write(pkt)

# pretty console line + CSV row if IP present

ts = datetime.now().strftime("%H:%M:%S")

if IP in pkt:

src, dst = pkt[IP].src, pkt[IP].dst

proto = proto\_of(pkt)

length = len(pkt)

print(f"[{ts}] {src} -> {dst} | {proto} | len={length}")

csv\_writer.writerow([ts, src, dst, proto, length])

else:

print(f"[{ts}] {pkt.summary()}")

try:

sniff(prn=handle, store=False, iface=args.iface, filter=args.filter, count=args.count)

except KeyboardInterrupt:

pass

finally:

csv\_file.flush(); csv\_file.close()

print("[+] Stopped. CSV/PCAP saved.")

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

main()

**day 7 code**

**# packet-sniffer/app.py (Stage C: Filtering + Summary)**

**from scapy.all import sniff, IP, TCP, UDP, ICMP**

**from scapy.utils import PcapWriter**

**from datetime import datetime**

**from pathlib import Path**

**import argparse, csv**

**def proto\_of(pkt):**

**if TCP in pkt: return "TCP"**

**if UDP in pkt: return "UDP"**

**if ICMP in pkt: return "ICMP"**

**if IP in pkt: return f"IP proto {pkt[IP].proto}"**

**return pkt.name**

**def main():**

**parser = argparse.ArgumentParser(description="Packet sniffer with filters + summary")**

**parser.add\_argument("-i","--iface", default=None, help="Interface (e.g., eth0/wlan0/ens33)")**

**parser.add\_argument("-f","--filter", default="ip", help="BPF filter (e.g., 'tcp', 'udp', 'port 53')")**

**parser.add\_argument("-c","--count", type=int, default=0, help="Packets to capture (0=infinite)")**

**args = parser.parse\_args()**

**# paths**

**base\_dir = Path(\_\_file\_\_).resolve().parent**

**log\_dir = base\_dir / "logs"**

**log\_dir.mkdir(parents=True, exist\_ok=True)**

**stamp = datetime.now().strftime("%Y%m%d\_%H%M%S")**

**csv\_path = log\_dir / f"packets\_{stamp}.csv"**

**pcap\_path = log\_dir / f"packets\_{stamp}.pcap"**

**# writers**

**csv\_file = open(csv\_path, "w", newline="")**

**csv\_writer = csv.writer(csv\_file)**

**csv\_writer.writerow(["time","src","dst","proto","length"])**

**pcap = PcapWriter(str(pcap\_path), append=True, sync=True)**

**# counters**

**stats = {"TCP":0, "UDP":0, "ICMP":0, "OTHER":0}**

**print(f"[+] Writing CSV -> {csv\_path}")**

**print(f"[+] Writing PCAP -> {pcap\_path}")**

**print(f"[+] Sniffing on iface={args.iface or 'DEFAULT'} filter='{args.filter}' count={args.count or '∞'}")**

**print("[+] Ctrl+C to stop")**

**def handle(pkt):**

**pcap.write(pkt)**

**ts = datetime.now().strftime("%H:%M:%S")**

**if IP in pkt:**

**src, dst = pkt[IP].src, pkt[IP].dst**

**proto = proto\_of(pkt)**

**length = len(pkt)**

**print(f"[{ts}] {src} -> {dst} | {proto} | len={length}")**

**csv\_writer.writerow([ts, src, dst, proto, length])**

**if proto in stats:**

**stats[proto] += 1**

**else:**

**stats["OTHER"] += 1**

**try:**

**sniff(prn=handle, store=False, iface=args.iface, filter=args.filter, count=args.count)**

**except KeyboardInterrupt:**

**pass**

**finally:**

**csv\_file.flush(); csv\_file.close()**

**print("\n[+] Capture finished. Summary:")**

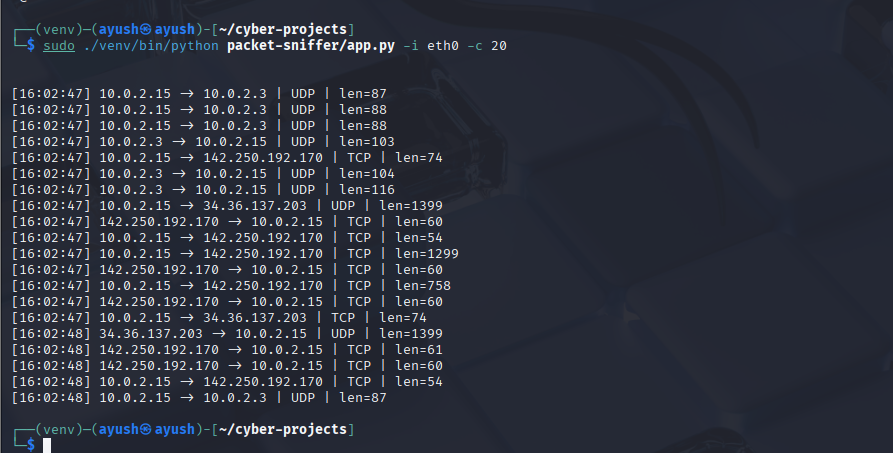
**for k,v in stats.items():**

**print(f" {k}: {v} packets")**

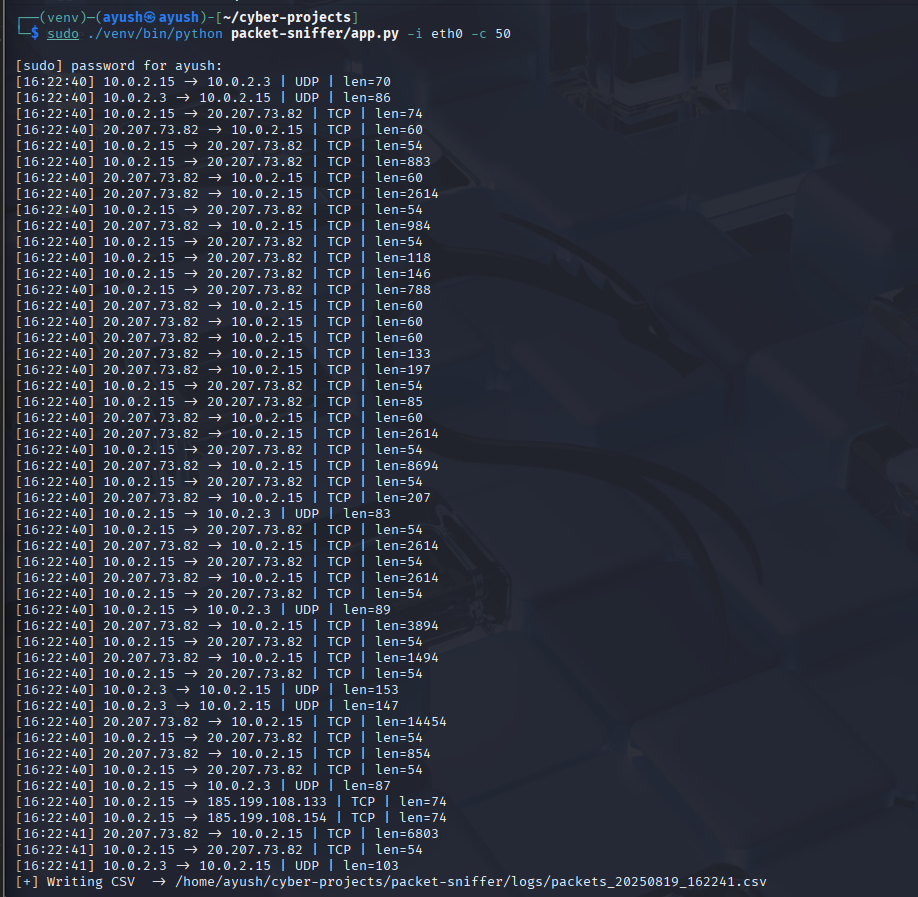
**print("[+] CSV/PCAP saved.")**

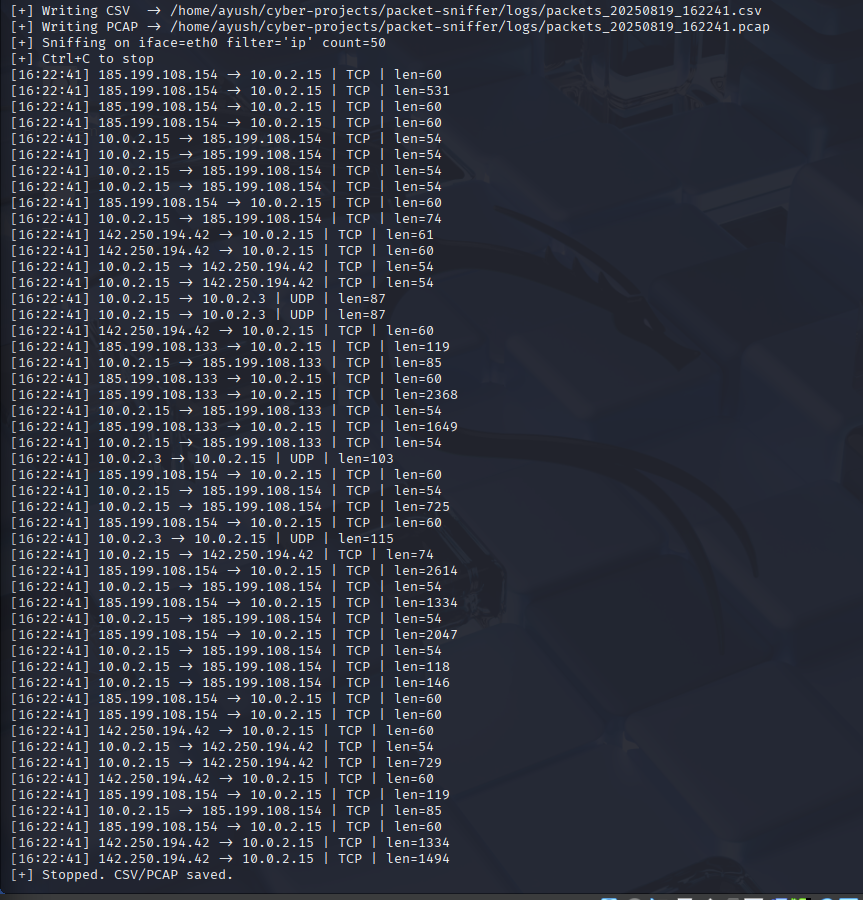
**if \_\_name\_\_ == "\_\_main\_\_":**

**main()**

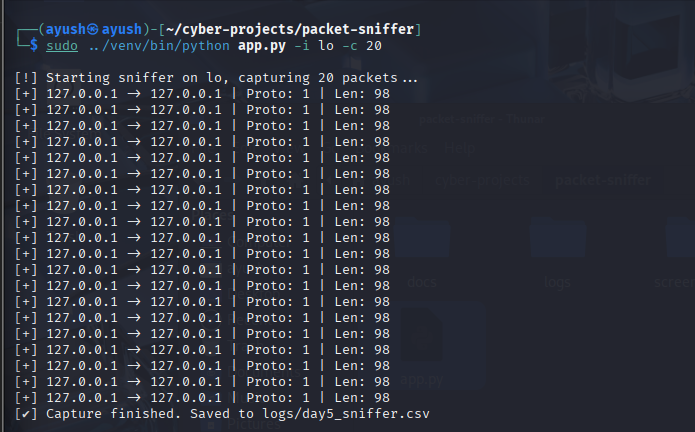


Adv

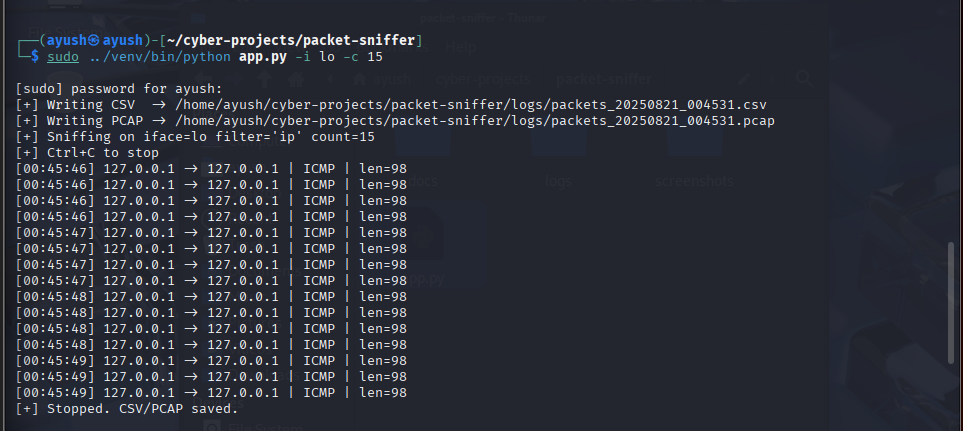


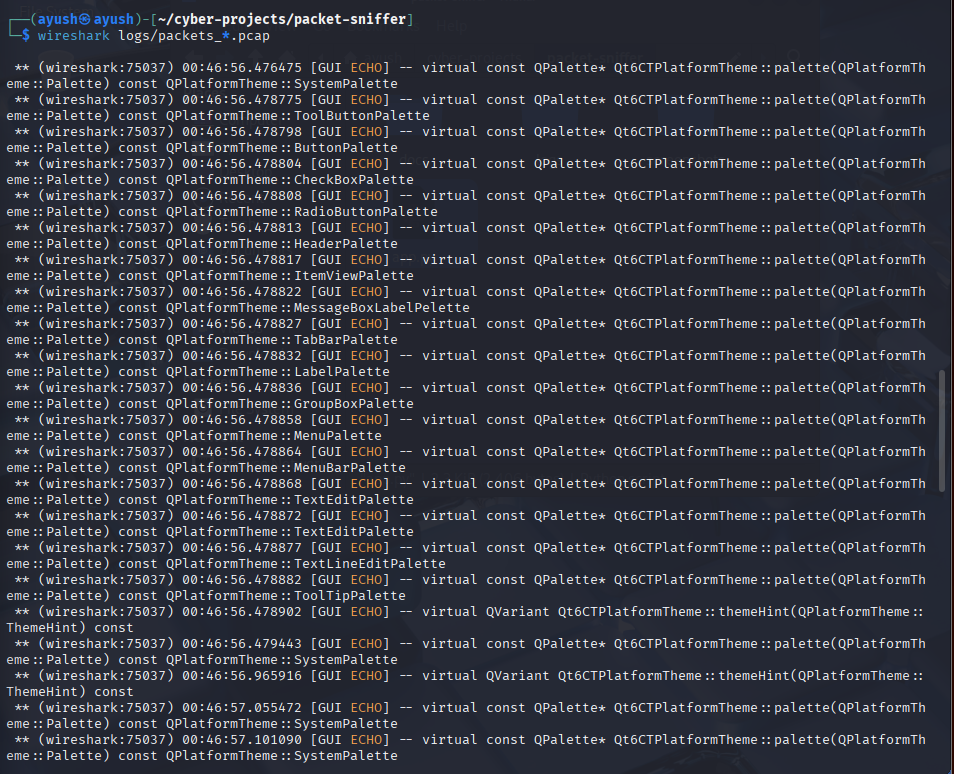


Day 5



Day 6





Day 7 output

