```python
#!/usr/bin/env python

# Suppress warnings about missing IPv6 route and tcpdump bin
import logging
logging.getLogger("scapy.loading").setLevel(logging.ERROR)
logging.getLogger("scapy.runtime").setLevel(logging.ERROR)

from scapy.all import *

# TCP flags
def decode_tcp_flag(f):
    if ((f & 0x01) or (f & 0x04) or (f & 0x08) or (f & 0x20) or (f & 0x40) or (f & 0x80)):
        return 0
    elif ((f & 0x02) and (f & 0x10)):
        return 2
    elif (f & 0x02):
        return 1
    else:
        return 0

# counter of total packets
total_packets = 0
total_bytes = 0

# read input pcap file
pcap = rdpcap(sys.argv[1])

# read sinkholes.txt
sinkholes = [line.rstrip('\n') for line in open("sinkholes.txt", "r")]

# arp table
arp_table = {}

# IIS unicode
iis_unicodes = ['%255c', '%25%35%63', '%252f', '%%35c', '%%35%63', '%C1%1C', '%C1%9C', '%C0%AF', '%c1%1c', '%c0%af', '%c1%9c', '%c0%af', '%e0%80%af', '%f0%80%80%af', '%f8%80

for pkt in pcap:

    if pkt.haslayer(IP):
        # Q2 Spoofed packets
        if (not ((pkt[IP].src[0:3] == '10.') or (pkt[IP].dst[0:3] == '10.'))) :
            print("[Spoofed IP address]: src:" + pkt[IP].src + ", dst:" + pkt[IP].dst)

        # TCP packet
        if pkt.haslayer(TCP):
            # Q3 Unauthorized servers
            tcp_flags = decode_tcp_flag(pkt[TCP].flags)
            if ( not(pkt[IP].src[0:3] == '10.') and (pkt[IP].dst[0:3] == '10.') and (tcp_flags == 1)):
                print ("[Attempted server connection]: rem:" + pkt[IP].src + ", srv:" + pkt[IP].dst + ", port:" + str(pkt[TCP].dport))
            if ( (pkt[IP].src[0:3] == '10.') and not(pkt[IP].dst[0:3] == '10.') and (tcp_flags == 2)):
                print ("[Accepted server connection]: rem:" + pkt[IP].dst + ", srv:" + pkt[IP].src + ", port:" + str(pkt[TCP].sport))

            # Q6 IIS worms
            #if( (pkt[TCP].dport == 80) and not(tcp_flags == 1)):
            #    http_host = pkt[HTTPRequest].Host
            #    for iis_unicode in iis_unicodes:
            #        print ("[Unicode IIS exploit]: src:" + pkt[IP].src + ", dst:" + pkt[IP].dst)

        # UDP packet
        if pkt.haslayer(UDP):
            # Q4 Sinkholes
            if pkt.haslayer(DNSRR):
                if ((pkt[DNSRR].type == 1) and (pkt[DNSRR].rdata in sinkholes) ):
                    print ("[Sinkhole lookup]: src:"+ pkt[IP].dst + ", host:" + pkt[DNSRR].rrname[:-1] + ", ip:" + pkt[DNSRR].rdata)
            # Q7 NTP
            if ((pkt[UDP].dport == 123) and pkt.haslayer(Raw) and (ord(pkt[Raw].load[3]) == 42) ):
                print ("[NTP DDoS]: vic:" + pkt[IP].src + ", srv:" + pkt[IP].dst)

    if pkt.haslayer(ARP):
        # Q5 ARP
        if (pkt[ARP].op == 2):
            psrc = pkt[ARP].psrc
            hwsrc = pkt[ARP].hwsrc
            if arp_table.has_key(psrc):
                if not (arp_table[psrc] == hwsrc) :
                    print ("[Potential ARP spoofing]: ip:" + psrc + ", old:" + arp_table[psrc].upper() + ", new:" + hwsrc.upper())
            arp_table[psrc] = hwsrc



for p, (sec, usec, wirelen) in RawPcapReader(sys.argv[1]):
    # Q1 Anomaly detection
    total_packets += 1
    total_bytes += wirelen
print("Analyzed " + str(total_packets) + " packets, size " + str(total_bytes))
```