Ayaan Puri: 920893614
Vikram Penumarti: 920928592

Part 1:

File location:

- part1_dns/dnsclientAyaan_Puri_920893614_Vikram_Penumarti_920928592.py

In this part, we implemented a DNS client using the socket API. We constructed a DNS query manually, sent it to a public DNS resolver, parsed the reply, extracted A records for tmz.com, and then measured RTT values for both the DNS lookup and an HTTP connection to the IP address.

Constructing the DNS Request

We first built the DNS request packet. This included:

- A 12-byte DNS header (transaction ID, Standard query flags)
- The Question section (encoded domain name)

The request was made using basic Python operations and struct.pack().

Sending the DNS Query

We used a UDP socket to communicate with the public DNS resolver 1.1.1.1 and recorded the start time before sending the packet. Once the response came, we recorded the end time to calculate the DNS RTT.

Parsing the DNS Response

To parse the DNS reply:

- Unpacked the 12-byte response header and checked the RCODE.
- Skipped the Question section by reading the encoded domain name.
- Processed each resource record by reading its TYPE, CLASS, TTL, and RDLENGTH.
- For A records, we converted the RDATA field into an IPv4 address.

The resolver returned two valid A records for tmz.com.

HTTP Connection RTT

We used a TCP socket to connect to port 80 on the server, after getting the first IP address. The time taken for the TCP connect() call was measured as the RTT between our machine and the tmz.com web server.

Results

- DNS resolver RTT: ~26–29 ms
- Returned A records:
    - 13.248.160.137
    - 76.223.34.124
- TCP connect RTT to tmz.com server: ~23 ms