Billy Ouattara (920603707)
Jose Gavidia (921477055)
dns.py, utils.py, gpt_version_dns.py, tmz_response.html, tmz_gpt_page.html

In this part of the project, we constructed messages to be sent to different types of DNS resolvers, so we could extract the IP address of tmz.com. To do so, we had to go through a series of parsing of the DNS resolvers responses(from root, TLD, and Authoritative). The process involved the following steps:

- Construct DNS message
- Sending message to root DNS resolver
- Parsing the response from the root DNS to extract tld resolver IP
- Sending our message to the TLD resolver
- Parsing our response from the TLD to extract Authoritative DNS resolver IP
- Sending our message to the Authoritative DNS
- Finally, parsing our response from Authoritative DNS to obtain tmz.com IP address

To construct the message, we looked at the DNS message format from the ietf website: ietf.org/rfc/rfc1035.txt. Message was composed of 7 fields. Six fields composing the header section with a size of 12 bytes and 1 section representing the DNS question. The size of the DNS question section is 4 bytes plus the size of the question name(Qname). For all subsection of the header and question, we formatted them as bytes. Below is a description of how we formatted the DNS message:

1. The Transaction ID is b“\x04\xd2” representing the the decimal number 39158.
2. All the sections of the flags were set to 0 in bytes. Since were are making a query, QR is set to 0 in the flags.
3. Since our query contains only 1 question, the question section of the header is set to the decimal value 1 in bytes.
4. For the last three sections: answer, authority and additional, we set to 0 since we don’t need to provide them for our query.
5. Lastly, for our question section, we looked at the Qname format of the tmz.com dns request using wireshark which was found to be: b“\x03\x74\x6d\x7a\x03\x63\x6f\x6d”. We have stored in our code as tmz.com bytes format for our query. To form our question

section, we added the Qname as well as Qtype and Qclass like the following: b“\x03\x74\x6d\x7a\x03\x63\x6f\x6d” + b'\x00' + b'\x00\x01' + b'\x00\x01'

   a. b'\x00' represents the end of the Qname.
   b. Each b'\x00\x01' sets Qtype amd Qclass to 1 respectively.

6. The combination of all those bytes representing each sections forms our DNS query message.

Concerning parsing the responses, the resolver IP address from our Root and TLD responses resided in the additional section of the DNS message. To extract the different IPs from Root and TLD, we extracted the last four bytes of the DNS message response we basically represents the last IP address in the additional section. We convert those four bytes into decimal and in the IPv4 format. That resolver IP is then used to make a DNS query to the right resolver. For our response from the authoritative, the tmz.com IP is located in the answer section of the message. The answer section like the additional section contains a list of IP addresses. To obtain tmz.com IP, we extracted the first IP address from the answer section from the Authoritative response message. To get that IP address we extracted the portion of the message response that contains the address.

We know that the header section of a DNS message is 12 bytes long. When adding the Question section of the message, it becomes 25 since our Question section is 13 bytes long. In the item of the answer section, the first 12 bytes represent name, class, type, TTL, and Rdlength. Therefore, by summing all those bytes, we get 37 which is the starting point of our IP address. The IP address is four bytes long. When doing the following:

```
tmz_response[37:41]
```

we are extracting the first IP address of tmz.com from the list of IPs in the answer section of the message we received from the Authoritative DNS.

Note: Wireshark was helpful in helping us understand further the structure of the different DNS messages when looking at them at bytes level.