ECS 152A: HW2 Part 1a

Nghi Dao (921147615), Bian Lee (920763430)

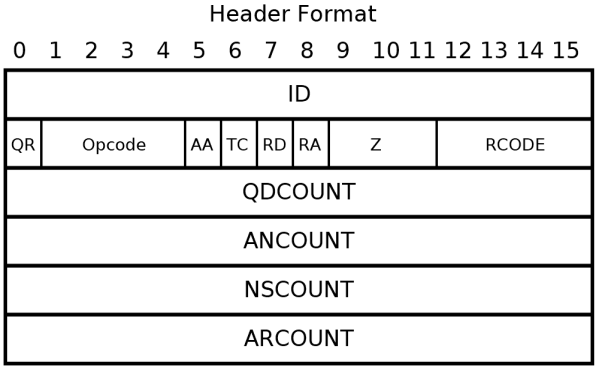
November 2024

Final submission: DNSClient.py
ChatGPT response: DNSClientGPT.py, link here

Creating the DNS Packet

First, we created the header as follow:

- **Identification:** A random 16-bit value.
- **Flags:** Set to 0x0000, which disables recursion.
- **Question count:** A 16-bit value to the number of questions, which in this case is 1.
- **Answer count:** A 16-bit value set to 0, since we are not answering any questions.
- **Authority record count:** A 16-bit value set to 0.
- **Additional record count:** A 16-bit value also set to 0.



To create the questions, we first encode the each domain name by splitting it up into multiple subdomains. Then, assuming each domain is at most 63 characters long (since that is the maximum size supported for a subdomain), we append 1 byte in front of each subdomain to indicate its length. Since the length is at most 63, the byte that encodes the length will use at most 6 of the 8 bits, with the last 2 bits reserved for pointers. For simplicity, we will not make use of pointers for encoding the packets, but we will have to take it into account when decoding. After doing so, we will add a null character to the end (or a length of 0). Afterwards, we will also add a 16-bit QTYPE value of 0x0001 or 0x0001 to indicate either an NS record or an A record respectively. When requesting to the root DNS, we will look for an NS record, and when requesting to the TLD or Authoritative server, we will look for an A record. We will also then add a 16-byte QCLASS value of 0x0001 to indicate a standard DNS query. An example of the encoded tmz.com question to the root server is shown as follows:

Length	Name	Length	Name	Length	QTYPE	QCLASS
0x03	**tmz**	**0x03**	**com**	**0x00**	**0x0002**	**0x0001**

The question is appended immediately after the header and all numbers are encoded in big endian.

Parsing the DNS Response

To decode the header of the DNS, we can simply read decode 2 bytes at a time to retrieve the ID, Flags, QDCount, ANCOUNT, NSCOUNT, and ARCOUNT, as shown in the diagram of the header earlier. Since the question of the request is echoed back, we will also need to read the question before we can read the rest of the response.

To read a question, we first need to parse the domain by first parsing each subdomain. Each subdomain begins with a byte that indicates either a length of a pointer. If the most significant is 11 then it is a pointer, if it is 00 then it is a length. When it is a length, we simply read next same amount bytes to get the subdomain and continue until we reach a null byte. If it is a pointer, then the address that the pointer is referring to is the remaining 6 bits of the bytes and the next byte, which in total gives a 14 bit-value for the pointer. The pointer points to another byte in the response, and we simply read domain again starting at the location the pointer points to and continue until we reach a null byte before we continue reading from the original location.

After reading the domain, we can then read the next 2 bytes to get the QTYPE and then another 2 to get the QCLASS. We can repeat this process depending on the number of questions (which should be 1).

For the other records in the answer, authorative records, and additional records, we can parse each record by reading the domain, similar to the question. However, each record will have contain a ttl and resource data length. We are not too concerned with the ttl, but the format of the resource data depends on the type of record:

- **Type = 1:** IPv4
- **Type = 28:** IPv6
- **Type = 2:** NS Record

These are the 3 types that we need to handle. When we request from the root DNS or TLD, we will get back no answer records, but instead type 2 records in the authoritative record section and the resource data is the name of the authoritative server, in which the ip can be found in the additional records section of the DNS response. When we request from the authoritative server, the answer will be either type 1 or type 28, which contains the IP that we are interested in.

An example of the format of each record is as follows:

Length	Name	Length	Name	Length	QTYPE	QCLASS	TTL	RD LENGTH	RD DATA
0x03	**tmz**	**0x03**	**com**	**0x00**	**0x0001**	**0x0001**	**100**	**0x0004**	**8 8 8 8**