

Name: Gul Jain

Report: 3331 assignment

Point 1: Choice of programming language: python

1. Python is versatile and easy to use and understand
2. Python has a lot of helpful libraries
3. I have been using python since 2019 and feel confident with the language.

Point 2: What files have I created.

1. Client.py
2. Server.py
 - (coded in VScode)
 - Python3 used

Point 3: important point from the spec

- Master File: <domain-name> <type> <data>
- You may assume that the master file contains at least one NS record for the root zone, and its corresponding A record
- Client Query: header, question
- Server response: header, question, answer, authority additional
- Possible record types: A, NS, CNAME

Point 4: Libraries used

1. struct
2. random
3. sys
4. time
5. datetime
6. socket
7. threading

Client.py

1. constants
 - a. A = 1
 - b. NS = 2
 - c. CNAME = 5
 - d. BYTE_4 = 4
 - e. DIST = 35

1. Takes in the following arguments: server_port:, qname, qtype, timeout
2. Checks if the arguments are valid
3. Calls client(server_port, qName, qType, timeoutTime) function
4. In client function, firstly a qId is generated
5. Then another function encode_client_query(qId, qName, qType) is called that is responsible for encoding the client query.
6. In encode_client_query, two other functions encode_question(qType, qName) and encode_header(question, qId) are called to encode the question and header respectively.
7. In encode_question function, the qname is encoded using .encode() function of python and a mapping value is generated for qtype using the encodeQtype function. Then the length of the qName is also encoded which is helpful when we decode this query. Lastly, a summation of all the encoded bytes is returned
8. Similarly, the length of the question and the qId are encoded in encode_header function
9. A summation of encoded header and question is returned in encode_client_query which is sent to the server.
10. After that a response is fetched from the server and decoded through decode_response function which is then printed.

Drawbacks

1. Could have handled error-checking better
2. The alignment of domain name, query type and data is not perfect
3. Use of global variables is not a good practise.
4. A bit of redundancy code is present.

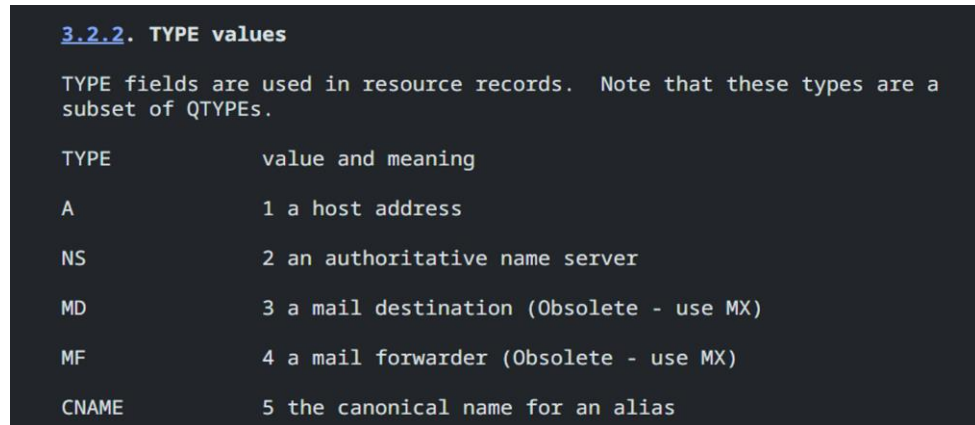
Server.py

2. Global variables
 - a. addr_dict = {}
 - b. cname_dict = {}
 - c. ns_dict = {}
3. constants
 - a. A = 1
 - b. NS = 2
 - c. CNAME = 5
4. Takes in the following arguments: serverport
5. First, it calls the load_rrs("master.txt" function to load all the resource records.

6. In load_rrs function, a loop exists to iterate over each record to fetch the required domain name, record type and corresponding data.
7. After that, these arguments are passed onto the add_record function. In add_record function, records are into into addr_dict, ns_dict, cname_dict based on their record type as values where keys are the domain names.
8. Once all the records are rightfully registered, the server(server_port) function is called in main.
9. In server, a server socket is created and a print statement is passed indicating that the server is active. After that, the server loops till forever to listen and answer to any queries.
10. Then we receive data and client address using sock.recvfrom(2048). This data contains the client's question and header so we call functions decode_question and decode_header to decode them
11. Once we have the header, question and client address, we pass it in as argument along with sock to call generate_request_response to run on a new thread.
12. Then we fetch values of client_port, query_id, qname and qtype and use it to print server logs.
13. Once that is done, we call get_resource_records function to fetch resource records.
14. Get_resource_records takes in arguments qtype, qname, ans_list=None, auths_list=None and adds_list=None and appends a string containing a record in the respective list if the required conditions are satisfied. It also recursively calls the function again if the query type is cname and the qname already exists in cname_dict to find alias.
15. If none of the conditions are met, we try to find NS record for the root zone and its corresponding A record using the function fetch_ns_a_record(qname, auths_list, adds_list)
16. In the function fetch_ns_a_record(qname, auths_list, adds_list), the domain is split into parts on the basis of '.' And a subdomain is created without the first index of the domain name and finds the ns record and corresponding a record.
17. After that, a response is generated by calling the generate_response function which encodes various sections of dns response. It creates a response_size variable to keep track of the bytes and encodes the question, answer records, authority records and additional records. At the end, it sums it all up and returns the response back which is then sent to the client
18. Some more helper functions used
 - a. encodeRtype(rtype)
 - b. decodeRtype(rtype)
 - c. encodeQtype(qtype)
 - d. decodeQtype(qtype)

- e. `encode_header(size, qid)`
- f. `decode_header(data)`
- g. `encode_res_ques(qtype, qname)`
- h. `encode_rr(rtype, ans)`
- i. `decode_question(data)`

References taken:



The screenshot shows a section titled "3.2.2. TYPE values" with a note that TYPE fields are used in resource records and are a subset of QTYPES. Below this is a table with two columns: "TYPE" and "value and meaning". The table lists five types: A (1, a host address), NS (2, an authoritative name server), MD (3, a mail destination, obsolete - use MX), MF (4, a mail forwarder, obsolete - use MX), and CNAME (5, the canonical name for an alias).

TYPE	value and meaning
A	1 a host address
NS	2 an authoritative name server
MD	3 a mail destination (Obsolete - use MX)
MF	4 a mail forwarder (Obsolete - use MX)
CNAME	5 the canonical name for an alias

references: Computer Networking_ A Top-Down Approach, 7th Edition - Computer Networking_ A Top Down Approach, 7th, converted

section: 2.7.1 Socket Programming with UDP

struct module reference:

<https://www.geeksforgeeks.org/struct-module-python/>

<https://docs.python.org/3/library/struct.html>

I also referenced a few lines from a couple of stackOverflow posts and I was unable to find afterwards.