Yejin Shin

111850347

CSE 310

Homework 1

1. a. False. There are total 4 requests that client will send. First request is to set up connection with server, and three requests for 3 images.

   b. True. The reason is because the two URLs https://www.cs.stonybrook.edu/about-us.html and https://www.cs.stonybrook.edu/admissions.html have same domain name of cs.stonybrook.edu.

   c. True. The connection is Peer-to-Peer, which uses HTTP protocol.

   d. True. Application layer protocols ensure connection is set up properly as security issue.

   e. False. The root name server in DNS shows top level domain name server as its query. TLD server returns authoritative name servers.

2. HTTP 1.0 had non-persistent connection, so 1 connection per 1 request was necessary. HTTP 1.1 is better version based on HTTP 1.0, which supports 3P connections: persistent, parallelization, and pipelining. HTTP 1.1's persistent connection occurs only when the client and server is opening connection first time. Once the connection is successful, then no additional connection is needed for requests until the connection is over. HTTP 1.1 supports up to 6 parallel connections simultaneously. This method is usually fast, but may be affected from the bandwidth of network it is connected to. If so, dealing with requests in order will be faster. Also, parallelization takes up large portion of memory thus internet browsers typically limit the number of parallel connections to 4. Pipelining is method of sending multiple requests consecutively and reduce delay. However, HTTP takes requests based on the order they arrived so pipelining could result more delay if a request is takes long time to handle.

3. If IP address of server is unknown and only URL is given, then DNS and HTTP will be used to connect to the server. DNS provides IP address of the URL's domain name, so client can make connection with the domain name's server.

4.

The iterative query of DNS takes following steps:

1. Client asks local DNS server for IP address of the URL.
2. Local DNS server asks the root name server, and get response of which TLD server to ask.
3. Local DNS server asks the TLD server and get response of which authoritative server to ask.
4. Local DNS server gets the IP address from the authoritative server and takes it to client.

The recursive query of DNS takes following steps:

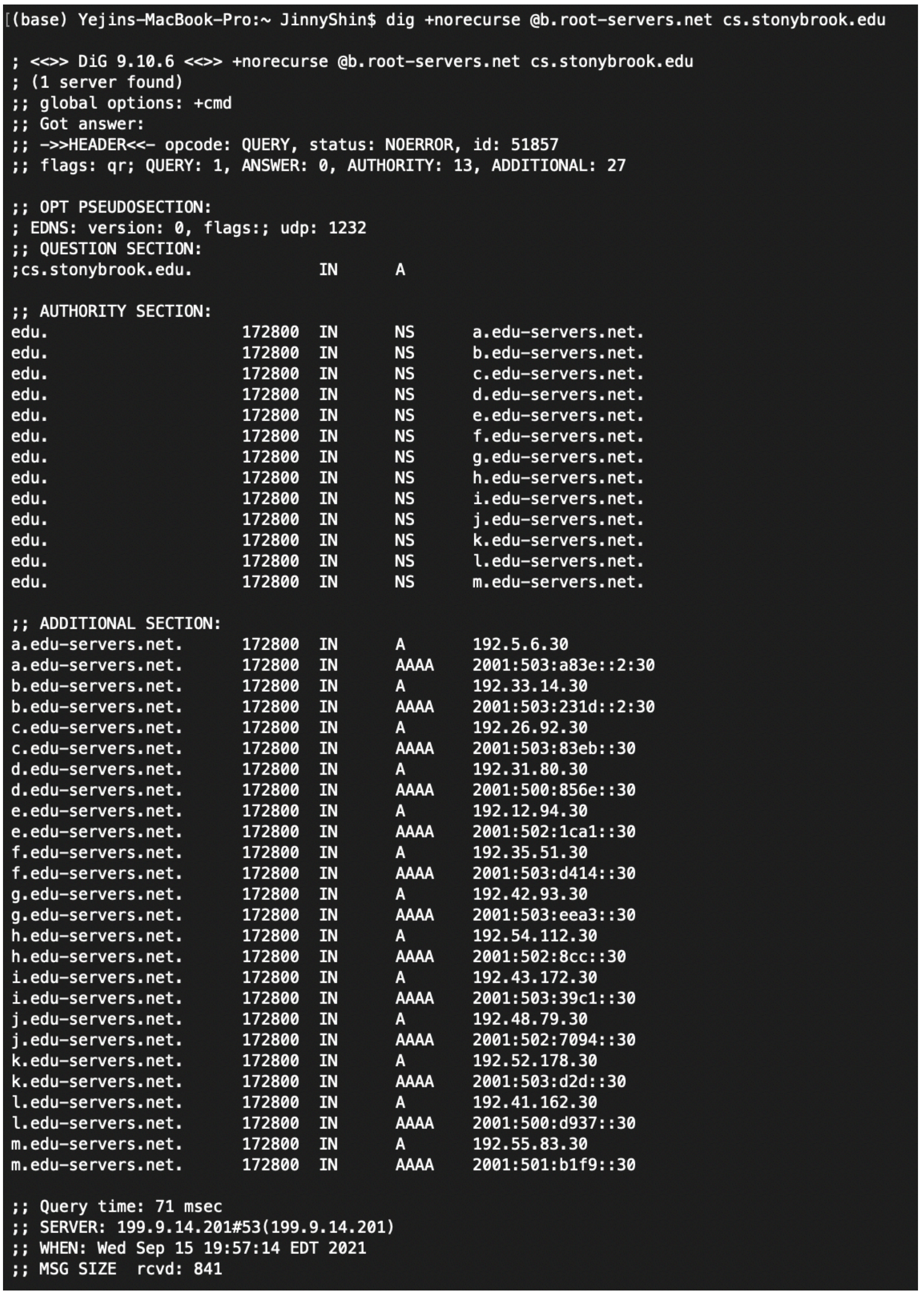
1. Client asks local DNS server for IP address of the URL.
2. Local DNS server asks the root name server.

3. The root name server asks to appropriate TLD server.
4. TLD server asks to the authoritative server and gets response.
5. TLD server sends the response back to the root name server.
6. The root name server sends the response back to the local DNS server
7. The local DNS server sends the response to the client.

The key difference between iterative query and recursive query is that local DNS handles the responses between DNS servers during iterative query. On the other hand, there is no one specific server that watches over the process in recursive query.

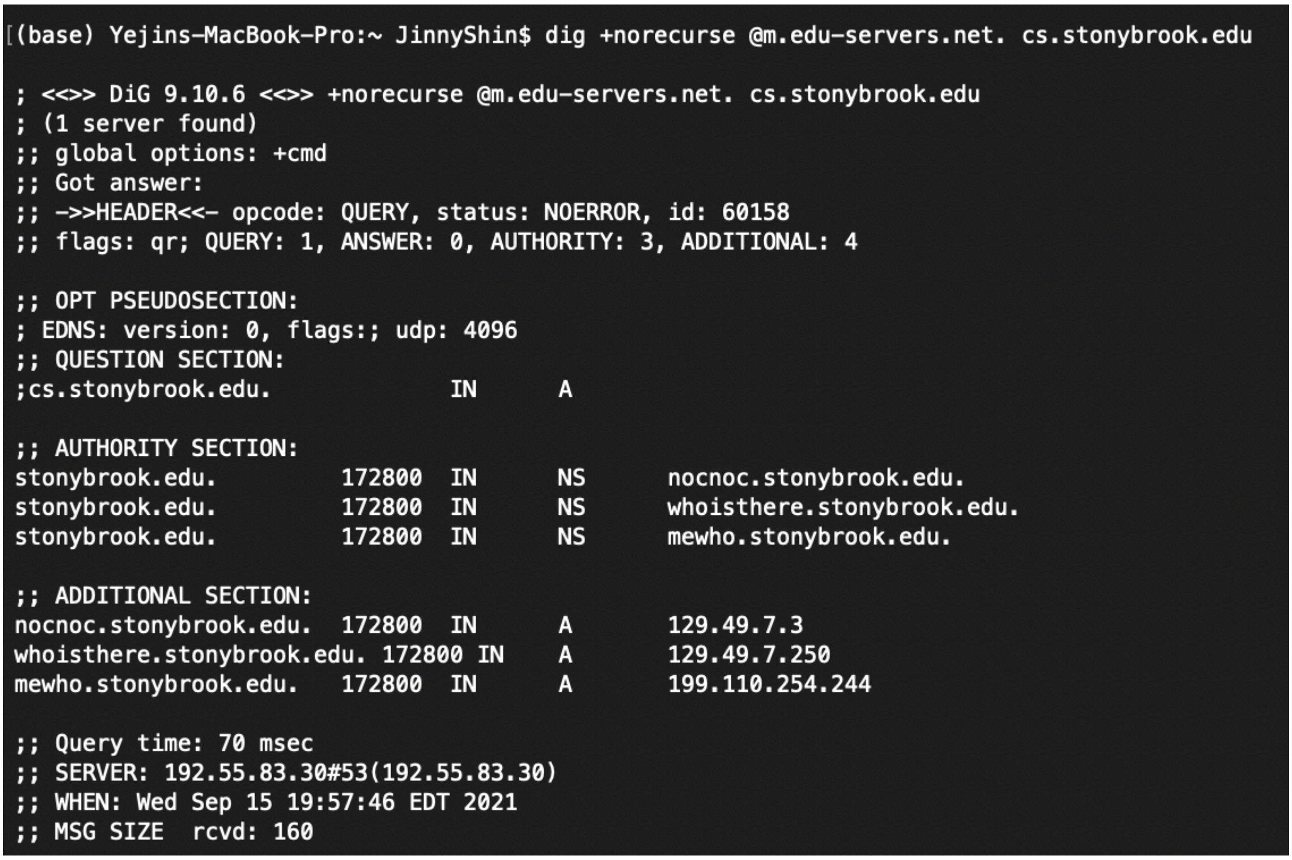
5.

dig from @b.root-servers.net to TLD server



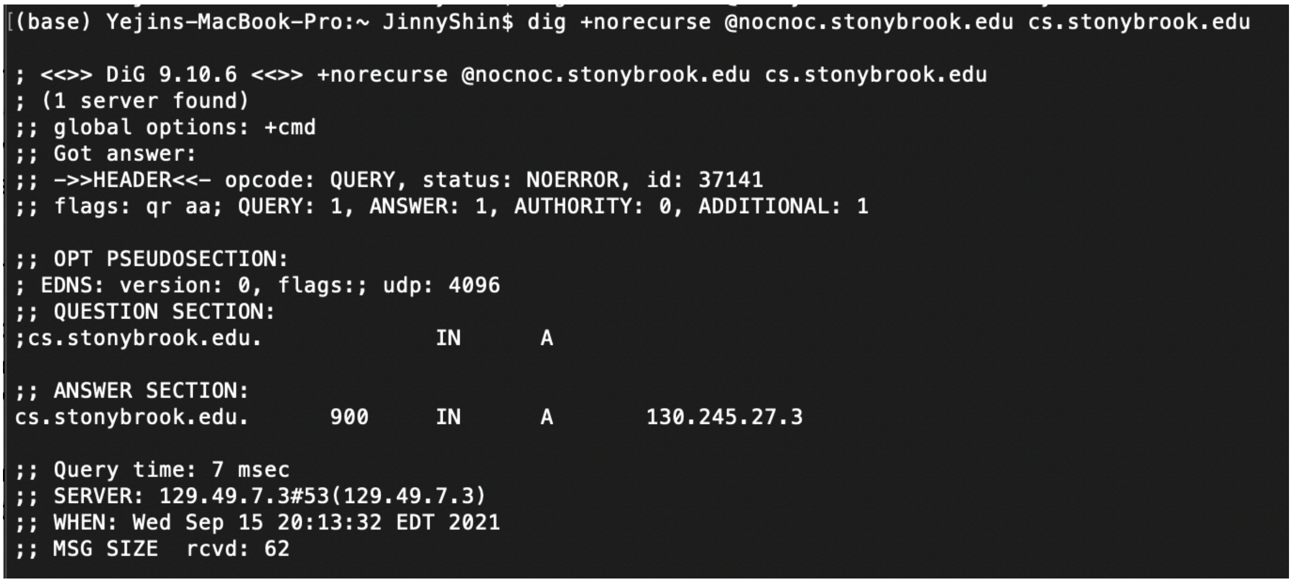
```
[(base) Yejins-MacBook-Pro:~ JinnyShin$ dig +norecurse @b.root-servers.net cs.stonybrook.edu

; <<>> DiG 9.10.6 <<>> +norecurse @b.root-servers.net cs.stonybrook.edu
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51857
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 13, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;cs.stonybrook.edu.             IN      A

;; AUTHORITY SECTION:
edu.                    172800  IN      NS      a.edu-servers.net.
edu.                    172800  IN      NS      b.edu-servers.net.
edu.                    172800  IN      NS      c.edu-servers.net.
edu.                    172800  IN      NS      d.edu-servers.net.
edu.                    172800  IN      NS      e.edu-servers.net.
edu.                    172800  IN      NS      f.edu-servers.net.
edu.                    172800  IN      NS      g.edu-servers.net.
edu.                    172800  IN      NS      h.edu-servers.net.
edu.                    172800  IN      NS      i.edu-servers.net.
edu.                    172800  IN      NS      j.edu-servers.net.
edu.                    172800  IN      NS      k.edu-servers.net.
edu.                    172800  IN      NS      l.edu-servers.net.
edu.                    172800  IN      NS      m.edu-servers.net.

;; ADDITIONAL SECTION:
a.edu-servers.net.      172800  IN      A       192.5.6.30
a.edu-servers.net.      172800  IN      AAAA    2001:503:a83e::2:30
b.edu-servers.net.      172800  IN      A       192.33.14.30
b.edu-servers.net.      172800  IN      AAAA    2001:503:231d::2:30
c.edu-servers.net.      172800  IN      A       192.26.92.30
c.edu-servers.net.      172800  IN      AAAA    2001:503:83eb::30
d.edu-servers.net.      172800  IN      A       192.31.80.30
d.edu-servers.net.      172800  IN      AAAA    2001:500:856e::30
e.edu-servers.net.      172800  IN      A       192.12.94.30
e.edu-servers.net.      172800  IN      AAAA    2001:502:1ca1::30
f.edu-servers.net.      172800  IN      A       192.35.51.30
f.edu-servers.net.      172800  IN      AAAA    2001:503:d414::30
g.edu-servers.net.      172800  IN      A       192.42.93.30
g.edu-servers.net.      172800  IN      AAAA    2001:503:eea3::30
h.edu-servers.net.      172800  IN      A       192.54.112.30
h.edu-servers.net.      172800  IN      AAAA    2001:502:8cc::30
i.edu-servers.net.      172800  IN      A       192.43.172.30
i.edu-servers.net.      172800  IN      AAAA    2001:503:39c1::30
j.edu-servers.net.      172800  IN      A       192.48.79.30
j.edu-servers.net.      172800  IN      AAAA    2001:502:7094::30
k.edu-servers.net.      172800  IN      A       192.52.178.30
k.edu-servers.net.      172800  IN      AAAA    2001:503:d2d::30
l.edu-servers.net.      172800  IN      A       192.41.162.30
l.edu-servers.net.      172800  IN      AAAA    2001:500:d937::30
m.edu-servers.net.      172800  IN      A       192.55.83.30
m.edu-servers.net.      172800  IN      AAAA    2001:501:b1f9::30

;; Query time: 71 msec
;; SERVER: 199.9.14.201#53(199.9.14.201)
;; WHEN: Wed Sep 15 19:57:14 EDT 2021
;; MSG SIZE  rcvd: 841
```

dig from TLD server to Authoritative server



```
[(base) Yejins-MacBook-Pro:~ JinnyShin$ dig +norecurse @m.edu-servers.net. cs.stonybrook.edu

; <<>> DiG 9.10.6 <<>> +norecurse @m.edu-servers.net. cs.stonybrook.edu
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60158
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 3, ADDITIONAL: 4

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;cs.stonybrook.edu.             IN      A

;; AUTHORITY SECTION:
stonybrook.edu.         172800  IN      NS      nocnoc.stonybrook.edu.
stonybrook.edu.         172800  IN      NS      whoisthere.stonybrook.edu.
stonybrook.edu.         172800  IN      NS      mewho.stonybrook.edu.

;; ADDITIONAL SECTION:
nocnoc.stonybrook.edu.  172800  IN      A       129.49.7.3
whoisthere.stonybrook.edu. 172800 IN    A       129.49.7.250
mewho.stonybrook.edu.   172800  IN      A       199.110.254.244

;; Query time: 70 msec
;; SERVER: 192.55.83.30#53(192.55.83.30)
;; WHEN: Wed Sep 15 19:57:46 EDT 2021
;; MSG SIZE  rcvd: 160
```

dig from Authoritative server to IP address



```
[(base) Yejins-MacBook-Pro:~ JinnyShin$ dig +norecurse @nocnoc.stonybrook.edu cs.stonybrook.edu

; <<>> DiG 9.10.6 <<>> +norecurse @nocnoc.stonybrook.edu cs.stonybrook.edu
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37141
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;cs.stonybrook.edu.             IN      A

;; ANSWER SECTION:
cs.stonybrook.edu.      900     IN      A       130.245.27.3

;; Query time: 7 msec
;; SERVER: 129.49.7.3#53(129.49.7.3)
;; WHEN: Wed Sep 15 20:13:32 EDT 2021
;; MSG SIZE  rcvd: 62
```

6.

(i) With the CNAME that foo.com received when it registered the image on it, client will be redirected to the CNAME. CNAME is another name for a domain, which serves as a bridge between domain name and IP address. When the client enters URL for the image, then CNAME of the image will be found and the IP address connected that CNAME will be returned. The CNAME has many IP addresses which are different by the location where client is accessing. Therefore, DNS redirection to close CDN that has the image, using the CNAME.

(ii) CDN has choice of routing and application based redirect without DNS redirect. One example of them is HTTP redirect and this method can be implemented with many applications. All communication between server and clients will be passed through encrypted path. The redirection is simply adding another routing to the redirecting URL.