Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

1. How can DNS be used to load balance services? Give a concrete explanation for google.com.

For popular websites e.g., google.com, there are multiple replicated servers. They all have a unique IP address that is associated with one canonical hostname. The DNS databases contain this set of IP addresses. So, when a client makes a DNS query, the DNS server replies with this set of addresses. The load balancing comes in with how the set is ordered. Every time the DNS sends the set of IP address it reorders them. Usually, the reordering is just rotating the order by 1 rotation.

With the command “host google.com” we see an interesting output. We are given 1 IPv4 and IPv6 address. But then we are given multiple lines regarding google.com mail. We see that “google.com mail” is handled by different servers. If we do “host X” where x is any one of those servers, we get different IP addresses. So, we can see that the DNS database has a set of IPs that handle the mail servers of google.

With the command “nslookup google.com” we don’t really see any information that can allude to the fact that a DNS server is load balancing.

With the commands “dig google.com” and “drill google.com” we can see information that proves load balancing is happening. In both commands in the response, we can look to the “Authority Section” to gain insights into load balancing. We see that google.com. is paired with 4 other servers. These records are type NS records which means the domain google.com (aka the Name) is associated with the hostname of an authoritative DNS server (aka the Value). In our case these are ns1.google.com, …, ns4.google.com. Then if we look further down the output to the “Additional Section” we can see that each authoritative DNS server has its own IP address (4 and 6).

Lastly, if we call the dig or drill command again, we see the order of the Authoritative DNS servers rotate. Thus, we can see the DNS database is performing load balancing.

Below is the relevant terminal output. This is not all encompassing of what I did. If you would like to see the full output, I direct you to the file: “Question1TerminalOutput.txt.” The output in that file is not in the exact order in which I discuss my answer.

Host Commands:

```
mumma@Ethans-MacBook-Air ~ % host google.com
google.com has address 142.250.190.110
google.com has IPv6 address 2607:f8b0:4009:80b::200e
google.com mail is handled by 50 alt4.aspmx.l.google.com.
google.com mail is handled by 40 alt3.aspmx.l.google.com.
google.com mail is handled by 10 aspmx.l.google.com.
google.com mail is handled by 30 alt2.aspmx.l.google.com.
google.com mail is handled by 20 alt1.aspmx.l.google.com.
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

```
mumma@Ethans-MacBook-Air ~ % host alt1.aspmx.l.google.com.
alt1.aspmx.l.google.com has address 173.194.77.27
alt1.aspmx.l.google.com has IPv6 address 2607:f8b0:4023:401::1a
mumma@Ethans-MacBook-Air ~ % alt1.aspmx.2.google.com.
zsh: command not found: alt1.aspmx.2.google.com.
mumma@Ethans-MacBook-Air ~ % host alt2.aspmx.l.google.com.
alt2.aspmx.l.google.com has address 142.250.112.26
alt2.aspmx.l.google.com has IPv6 address 2607:f8b0:4023:1402::1b
```

Dig Commands:

```
mumma@Ethans-MacBook-Air ~ % dig google.com

; <<>> DiG 9.16.20 <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 13253
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 27fdd991bf5eb082ade94b906140ad71600d55c58cd7d5cd (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.         300    IN      A       142.250.190.110

;; AUTHORITY SECTION:
google.com.         11948IN        NS      ns2.google.com.
google.com.         11948IN        NS      ns3.google.com.
google.com.         11948IN        NS      ns4.google.com.
google.com.         11948IN        NS      ns1.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.             26668IN       A       216.239.32.10
ns2.google.com.             26668IN       A       216.239.34.10
ns3.google.com.             26668IN       A       216.239.36.10
ns4.google.com.             26668IN       A       216.239.38.10
ns1.google.com.             21143IN       AAAA 2001:4860:4802:32::a
ns2.google.com.             21143IN       AAAA 2001:4860:4802:34::a
ns3.google.com.             21143IN       AAAA 2001:4860:4802:36::a
ns4.google.com.             21143IN       AAAA 2001:4860:4802:38::a

;; Query time: 31 msec
;; SERVER: 129.79.1.1#53(129.79.1.1)
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

```
;; WHEN: Tue Sep 14 10:10:57 EDT 2021
;; MSG SIZE  rcvd: 331

mumma@Ethans-MacBook-Air ~ % dig google.com

; <<>> DiG 9.16.20 <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53244
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 97e1078223b0ea39a5e607b36140ad741f451f3969c34d83 (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.         40      IN      A       142.251.32.14

;; AUTHORITY SECTION:
google.com.         11605IN         NS      ns1.google.com.
google.com.         11605IN         NS      ns4.google.com.
google.com.         11605IN         NS      ns2.google.com.
google.com.         11605IN         NS      ns3.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.             96771IN         A       216.239.32.10
ns2.google.com.             96771IN         A       216.239.34.10
ns3.google.com.             96771IN         A       216.239.36.10
ns4.google.com.             96771IN         A       216.239.38.10
ns1.google.com.             116107          IN      AAAA 2001:4860:4802:32::a
ns2.google.com.             179073          IN      AAAA 2001:4860:4802:34::a
ns3.google.com.             179073          IN      AAAA 2001:4860:4802:36::a
ns4.google.com.             179073          IN      AAAA 2001:4860:4802:38::a

;; Query time: 16 msec
;; SERVER: 129.79.1.1#53(129.79.1.1)
;; WHEN: Tue Sep 14 10:11:00 EDT 2021
;; MSG SIZE  rcvd: 331
```

2. What is the inherent weakness of traditional DNS? Give an example of how an attack might utilize it.

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

I believe by traditional DNS we are talking about a single centralized DNS server. So, to start, this is a single point of failure. If the DNS server crashes nothing can be resolved easily and we must start memorizing IP address (YUCK). Second, traffic volume. This single DNS server will have to handle all the traffic of the internet. This will reach to the billions or even trillions or higher within one day. And we know it takes time to process and respond to one query so the time to wait for a response will grow as more requests come in. Third, a centralized DNS server cannot be everywhere geographically. So, that means users farther from the server will have a longer wait time to receive a response. Lastly, maintenance will be an issue. This database will be huge! It would contain every single record of all internet hosts. Not to mention how frequently it would have to update to keep on top of the changing environment. And with each update, presumably, the server wouldn't be able to respond to queries.

One attack Eve could do is a DDoS attack. This would flood the DNS server with an enormous number of bogus queries which will slow down the response time for everyone else. Potentially even crashing the DNS server. Another type of attack that might be possible is spoofing the DNS server's IP. This could reroute a user's query to Eve's system. To be honest, I'm not exactly sure what Eve might be able to do actively with this information. But certainly, she would know what website you are trying to access. Now that I think a little more about this. I suppose she could (if sophisticated enough) send back a webpage of the website you are trying to access. Then a user might be fooled into entering login credentials or other sensitive information and sending it back to Eve.

3. Perform a manual iterative DNS query for mail-relay.iu.edu with dig starting from the root servers. List all commands and their outputs and explain why you issued every command. Do not use tracing features (dig +trace) for your final write-down.

I used this reference to give me an idea of how to get started here: https://serverfault.com/questions/755355/performing-dns-queries-starting-with-one-of-the-root-servers-using-dig-command - I confused the dig @server and name arguments but once I read Hakan Lindqvist's answer I figured out what I needed to do. Okay now getting into the answer.

The first command used was "dig". This gave me a list of root servers to work with and drill down from. I just picked the first one and hoped it had a record for mail-relay.iu.edu.

```
mumma@Ethans-MacBook-Air ~ % dig

; <<>> DiG 9.16.20 <<>>
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37013
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

```
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 2ddd7640e6a9084fd7cf99b46140cbe15b6770bdc2dfac5e (good)
;; QUESTION SECTION:
;.                             IN      NS

;; ANSWER SECTION:
.                     159114   IN      NS     b.root-servers.net.
.                     159114   IN      NS     e.root-servers.net.
.                     159114   IN      NS     g.root-servers.net.
.                     159114   IN      NS     d.root-servers.net.
.                     159114   IN      NS     k.root-servers.net.
.                     159114   IN      NS     a.root-servers.net.
.                     159114   IN      NS     h.root-servers.net.
.                     159114   IN      NS     i.root-servers.net.
.                     159114   IN      NS     l.root-servers.net.
.                     159114   IN      NS     j.root-servers.net.
.                     159114   IN      NS     c.root-servers.net.
.                     159114   IN      NS     m.root-servers.net.
.                     159114   IN      NS     f.root-servers.net.

;; ADDITIONAL SECTION:
a.root-servers.net.   245132   IN      A      198.41.0.4
b.root-servers.net.   177131   IN      A      199.9.14.201
c.root-servers.net.   276244   IN      A      192.33.4.12
d.root-servers.net.   438601   IN      A      199.7.91.13
e.root-servers.net.   177131   IN      A      192.203.230.10
f.root-servers.net.   139184   IN      A      192.5.5.241
g.root-servers.net.   177167   IN      A      192.112.36.4
h.root-servers.net.   177167   IN      A      198.97.190.53
i.root-servers.net.   177131   IN      A      192.36.148.17
j.root-servers.net.   177131   IN      A      192.58.128.30
k.root-servers.net.   276244   IN      A      193.0.14.129
l.root-servers.net.   264669   IN      A      199.7.83.42
m.root-servers.net.   352195   IN      A      202.12.27.33
a.root-servers.net.   263315   IN      AAAA   2001:503:ba3e::2:30
b.root-servers.net.   177131   IN      AAAA   2001:500:200::b
c.root-servers.net.   276244   IN      AAAA   2001:500:2::c
d.root-servers.net.   438601   IN      AAAA   2001:500:2d::d
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

```
e.root-servers.net.     177131          IN      AAAA   2001:500:a8::e
f.root-servers.net.     438601          IN      AAAA   2001:500:2f::f
g.root-servers.net.     177167          IN      AAAA   2001:500:12::d0d
h.root-servers.net.     177167          IN      AAAA   2001:500:1::53
i.root-servers.net.     177131          IN      AAAA   2001:7fe::53
j.root-servers.net.     177131          IN      AAAA   2001:503:c27::2:30
k.root-servers.net.     267127          IN      AAAA   2001:7fd::1
l.root-servers.net.     264669          IN      AAAA   2001:500:9f::42
m.root-servers.net.     438601          IN      AAAA   2001:dc3::35

;; Query time: 2 msec
;; SERVER: 10.79.1.1#53(10.79.1.1)
;; WHEN: Tue Sep 14 12:20:49 EDT 2021
;; MSG SIZE  rcvd: 839
```

The second command used was “dig @a.root-servers.net. mail-relay.iu.edu.” This command gives us TLD DNS servers. Luckily this server (a.roo-servers.net.) gave me a hit on mail-relay.iu.edu. If no record was found on this server, I would’ve checked the next server in the list from the first command. With the response here we received a bunch of TLD edu servers (e.g., b.edu-servers.net.). Again, I just picked one from the returned list. At this point since I specified mail-relay.iu.edu in the command I know these are TLD servers that know where to get the record for mail-relay.iu.edu.

```
mumma@Ethans-MacBook-Air ~ % dig @a.root-servers.net. mail-relay.iu.edu

; <<>> DiG 9.16.20 <<>> @a.root-servers.net. mail-relay.iu.edu
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32258
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 13, ADDITIONAL: 27
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;mail-relay.iu.edu.             IN      A

;; AUTHORITY SECTION:
edu.                    172800          IN      NS      b.edu-servers.net.
edu.                    172800          IN      NS      f.edu-servers.net.
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

```
edu.                172800  IN  NS    i.edu-servers.net.
edu.                172800  IN  NS    a.edu-servers.net.
edu.                172800  IN  NS    g.edu-servers.net.
edu.                172800  IN  NS    j.edu-servers.net.
edu.                172800  IN  NS    k.edu-servers.net.
edu.                172800  IN  NS    m.edu-servers.net.
edu.                172800  IN  NS    l.edu-servers.net.
edu.                172800  IN  NS    h.edu-servers.net.
edu.                172800  IN  NS    c.edu-servers.net.
edu.                172800  IN  NS    e.edu-servers.net.
edu.                172800  IN  NS    d.edu-servers.net.

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

;; Query time: 15 msec
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

```
;; SERVER: 2001:503:ba3e::2:30#53(2001:503:ba3e::2:30)
;; WHEN: Tue Sep 14 12:21:05 EDT 2021
;; MSG SIZE  rcvd: 841
```

The third command used was "dig @b.edu-servers.net. mail-relay.iu.edu." Now this command returned the list of authoritative DNS servers. IU maintains their own and there are three of them: dns1.iu.edu., dns2.iu.edu, dns3.iu.edu.

```
mumma@Ethans-MacBook-Air ~ % dig @b.edu-servers.net. mail-relay.iu.edu

; <<>> DiG 9.16.20 <<>> @b.edu-servers.net. mail-relay.iu.edu
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57679
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 3, ADDITIONAL: 6
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;mail-relay.iu.edu.             IN      A

;; AUTHORITY SECTION:
iu.edu.                 172800  IN      NS      dns1.iu.edu.
iu.edu.                 172800  IN      NS      dns2.iu.edu.
iu.edu.                 172800  IN      NS      dns3.iu.edu.

;; ADDITIONAL SECTION:
dns1.iu.edu.            172800  IN      A       134.68.220.8
dns1.iu.edu.            172800  IN      AAAA    2001:18e8:3:220::10
dns2.iu.edu.            172800  IN      A       129.79.1.8
dns2.iu.edu.            172800  IN      AAAA    2001:18e8:2:8::10
dns3.iu.edu.            172800  IN      A       52.23.85.80

;; Query time: 6 msec
;; SERVER: 2001:503:231d::2:30#53(2001:503:231d::2:30)
;; WHEN: Tue Sep 14 12:21:42 EDT 2021
;; MSG SIZE  rcvd: 207
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

Our last command is "dig @dns1.iu.edu. mail-relay.iu.edu." Using an authoritative server from the last command we can finally receive the A record of mail-relay.iu.edu. Apparently, there are two servers running mail-relay.iu.edu. One at 134.68.220.21 and the other at 129.79.1.63.

```
mumma@Ethans-MacBook-Air ~ % dig @dns1.iu.edu. mail-relay.iu.edu

; <<>> DiG 9.16.20 <<>> @dns1.iu.edu. mail-relay.iu.edu
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5897
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 6
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d1c48d2c30b14b993972c0286140cc2efd1f68c64db0887e (good)
;; QUESTION SECTION:
;mail-relay.iu.edu.             IN      A

;; ANSWER SECTION:
mail-relay.iu.edu.      300     IN      A       134.68.220.21
mail-relay.iu.edu.      300     IN      A       129.79.1.63

;; AUTHORITY SECTION:
iu.edu.                 3600    IN      NS      dns3.iu.edu.
iu.edu.                 3600    IN      NS      dns2.iu.edu.
iu.edu.                 3600    IN      NS      dns1.iu.edu.

;; ADDITIONAL SECTION:
dns1.iu.edu.            3600    IN      A       134.68.220.8
dns2.iu.edu.            3600    IN      A       129.79.1.8
dns3.iu.edu.            3600    IN      A       52.23.85.80
dns1.iu.edu.            3600    IN      AAAA    2001:18e8:3:220::10
dns2.iu.edu.            3600    IN      AAAA    2001:18e8:2:8::10

;; Query time: 5 msec
;; SERVER: 2001:18e8:3:220::10#53(2001:18e8:3:220::10)
;; WHEN: Tue Sep 14 12:22:05 EDT 2021
;; MSG SIZE  rcvd: 267
```

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

4. You are sitting in a coffee shop and are connected to a public WLAN. You fire up wireshark and start sniffing the traffic of other customers. You notice that all of their traffic is over https so you cannot simply read it. You also notice something else, what is it and what are the implications?

If we filter Wireshark using "dns". This allows us to know what the other coffee shop patrons are browsing on the web. This is because DNS uses UDP and plaintext. So unlike HTTPS this is completely visible. This information us insight into what websites are being visited by the coffee patrons. If we see a patron with a laptop decorated with music related stickers and dns calls to Spotify or other music related websites, we can infer that patron is visiting these sites.

Additionally, we can see DHCP requests. The WLAN receives this request and issues the device an IP address to use. This gives us an IP address to device mapping. This could allow Eve to infer even further information or even attack a patron.

References:
https://en.wikipedia.org/wiki/Domain_Name_System
https://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol

5. Suppose that IU has an internal DNS cache. You are an ordinary user (no network admin). Can you determine (and if yes, how) if a given external website was recently accessed?

The short answer is no. Originally, I thought maybe I could use the TTL value of a dns record to figure this out. Ultimately that didn't work because my understanding of the TTL value was incorrect. I thought it was an active count value when in fact it is a value the cache should use to store it. After that amount of time has passed since caching the record, the system should delete it. So, this idea was useless. In my lab class someone discussed the possibility of using the query time to figure out if an external website was recently accessed. This is an interesting thought I didn't think to explore. At first glance, this doesn't seem viable because the query times aren't consistent. They jump around the initial query time, and you would think if a website was cached the query time would consistently be below the initial query time. Another interesting observation is that sometimes the authority section and additional sections are blank for subsequent dig commands. I don't think this is an indication of a website being cached. In my mind, the absence of those sections would need to be more consistent. Finally, I arrive at the answer no, I don't think its possible. Obviously the local dns

Ethan Behar
9/14/2021
Professor Swany
Assignment: 02_DNS

cache would require permissions to access the files/data of the records. If these records could be accessed without admin permissions Eve could do a lot of naughty stuff like redirect a website’s traffic to her own server.