Exercises for Hands-on 1: DNS

You should submit answers *only* to the questions asked. In particular, please do not include pages of output from `dig` unless specifically requested.

Getting Started

1. Using `dig`, find the IP address for `thyme.lcs.mit.edu`. What is the IP address?
2. The `dig` answer for `thyme` includes a record of type `CNAME`. In the terminology of chapter 4, what does `CNAME` mean?
3. What is the expiration time for the `thyme CNAME` record?

---

1. thyme.lcs.mit.edu ⇒ mercury.lcs.mit.edu ⇒ 18.26.0.122
2. Literally, a Canonical Name record, but in the book they use the term DNS ‘synonym’ as it binds an indirect name (see (1)) and in general terms, it’s an alias.
3. 1800 seconds

4. Run these commands to find what the computer you're using gets when it looks up "`data`" and "`data.`".

   ```
   dig +domain=mit.edu data
   dig +domain=mit.edu data.
   ```

   What are the two resulting IP addresses?
5. Why are the results different? Look at the man page for `dig` to see what the `+domain=` parameter does. Based on the output of the two commands, what is the difference between the DNS searches being performed for `data` and `data.`?

---

4. (dig +domain=mit.edu data) ⇒ 18.17.196.100
   (dig +domain=mit.edu data.) ⇒ 127.0.53.53

5. Data. is doing an internal query, as the IP returned is from the localhost. Data is doing an external search, thus getting MIT's 18.*.*.* address. In other words, the 'data' call is calling `dig data.mit.edu`, while the 'data.' call is calling `dig +domain=localhost data` or even `dig data.` because the trailing dot signifies an absolute path. And, if no usable server addresses are found, dig will send the query to the localhost.

Understanding hierarchy

For this problem, you will go through the steps of resolving a particular hostname, mimicking a standard recursive query. Assuming it knows nothing else about a name, a DNS resolver will ask a well-known root server. The root servers on the Internet are in the domain `root-servers.net`. One way to get a list of them is with the command:

```
athena% dig . ns
```

6. Use `dig` to ask *one* of the root servers the address of `lirone.csail.mit.edu`, *without* recursion. What command do you use to do this?
7. It is unlikely that these servers actually know the answer so they will *refer* you to a server (or list of servers) that might know more. Go through the hierarchy from the root without recursion, following the referrals manually, until you have found the address of `lirone.csail.mit.edu`. What commands did you use to do this? What IP address did you find for `lirone`?

---

6. `$ dig +norecurse lirone.csail.mit.edu `

7.

- $ dig @a.ROOT-SERVERS.NET lirone.csail.mit.edu +norecurs
- $ dig @a.edu-servers.net. lirone.csail.mit.edu +norecurs
- $ dig @usw2.akam.net. lirone.csail.mit.edu +norecurs
- $ dig @auth-ns3.csail.mit.edu. lirone.csail.mit.edu +norecurs

Finally, (lirone.csail.mit.edu) ⇒ 128.52.129.186

Understanding Caching

These queries will show you how your local machine's DNS cache works.

8. Ask your default server for information, without recursion, about the host `www.dmoz.org`.
    a. What command did you use?
    b. Did your default server have the answer in its cache? How do you know?
    c. How long did this query take? If this information was cached, please find some other host name that is not cached and do this section with that other host.
9. Now, ask your default server this same query but *with* recursion. It should return an answer for you. How long did this take?
10. Finally, ask your default server again without recursion. How long does this request take? Has the cache served its purpose?

---

8 a) $ dig +norecurs www.dmoz.org.

b) Yes. It didn't return an 'Authority' section, so it didnt ask another authority what the name was, plus it has an answer, so it must've already been in the cache.

a) dig +norecurs foxnews.com.

b) No. It asked a number of authorities, took a non-zero amount of time, and I would never have used dig on or visited foxnews.com before.

c) ;; Query time: 8 msec

9. ;; Query time: 36 msec

10. An answer was returned in 0msec. The cache served its purpose!