DNS,   CDNs   Lecture 14

Sam Madden

DNS

What is the relationship between a domain name (e.g., youtube.com) and an IP address?

DNS is the system that determines this mapping.

Basic idea:

You contact a DNS server, give it a query

It responds with the answer, or forwards your answer on to some other server

Example:

Whenever a server answers a query, it adds it to a cache

When you get a response, you add it to a cache

Each cached value has a duration after which you are supposed to look up the name again

At the top of the DNS hierarchy are root servers

Then servers for .com, .edu, etc ...

Then servers for MIT, youtube, etc....

Show cached

```
dig +norec csail.mit.edu
```

A record vs NS Record
TTL

Observe that A records expire much sooner than NS Records

Example:

csail.mit.edu

```
dig
dig edu@E.ROOT-SERVERS.NET.
dig mit.edu @G.GTLD-SERVERS.NET.
dig csail.mit.edu @bitsy.mit.edu.
```

How do you get your DNS server initially?

- Your admin tells it to you
- DHCP (Domain Host Configuration Protocol) -- you send out a broadcast on the local link -- DHCP server responds (via another broadcast message) with:
    - An IP address for you to use
    - The IP address of a "gateway" router for you to use
    - The IP address of a DNS server to use
    - ....

DNS has some fancy features:

- One physical machine can have multiple names

- One name can correspond to multiple IP addresses
    DNS load balancing
    DNS server picks which name to return
        most DNS servers cycle through these in "round robin" fashion

Example:

```
dig google.com
```

```
;; ANSWER SECTION:
google.com.          282    IN      A       209.85.171.100
google.com.          282    IN     A      74.125.45.100
google.com.          282    IN     A      74.125.67.100
```
Content delivery networks (CDNs -- e.g., Akamai)
Use DNS to provide scalability and adapt to load

Suppose I have some content that is accessed a lot -- e.g., a video on youtube

I can balance load amongst my local servers using DNS load balancing, but I still have to own the servers.

Puts a huge load on my servers to deliver it; can't adapt to load spikes (e..g, the "slashdot" effect.)

Also, for users that are far away (e.g., in Asia or Australia), they have to download that content over long distance and thin pipes, and ISPs in Australia have to pay a lot for than bandwidth.

For content -- like this video -- that is accessed repeatedly, would be be better to not have to go all the way to San Bruno CA everytime.

Idea: create a local cache

Solution 1: Proxy cache. For every URL, look up in a local cache (perhaps run by your ISP) to see if the content is there. If it is, fetch it. Otherwise, get original data. Just like DNS requests, web pages can have cache lifetimes associated with them which proxy caches respect.

Problems:

- Helps ISP but not necessarily content provider
- requires clients to configure their browsers to use proxy caches
- Doesn't address slashdot effect

Exist so called "transparent proxies" that can do this filtering automatically, but this may be distasteful to users, especially if their ISP is doing it (companies do this all the time.)

Solution 2: Content Distribution Networks (e.g., Akamai)

Show diagram:

Give example:

    nytimes.com
    dig graphics8.nytimes.com

From both MIT network and cellular network

Observe that TTL for answer is very short -- Why? -- Handle slashdot effect -- can dynamically start using more and more akamai servers for a particular request

Akamai -- company

Gives a way for content provider to offload load from server

Also helps ISP if server is inside ISP (creates an incentive for ISPs to participate!)

Akamai has thousands of caches all over the world

When a request for content -- like from images8.nytimes.com arrives -- it uses dns to forward user to the nearest server.

Determining the "nearest" server and how many servers to allocate is their secret sauce.

For dynamic content, Akamai also works.

Example -- show attempt to select best route back to Akamai, maintenance of reachability info, etc.

Akamai is a more general example of something called an **overlay network**.

An overlay is a way to create a network with new features or a different structure by building on top of an existing network.

Akamai creates an overlay on top of the IP network that chooses the best route from amongst a collection of IP servers.

Overlays are used widely to extend the network with new features, for example :

Provide a different topology (e.g., direct connection of clients)
- E.g., for administrative reasons (VPNs)
- Or for performance reasons (Akamai)

Provide a different addressing mechanism (e.g., content addressability, P2P)

VPN example:

Companies want to allow remote users or sites to have access to their corporate Internet sites

Good old days might have done this with a modem, but that's slow, and a pain to run.

VPN provides a way for remote users to appear to be on internal network while actually being external.

Idea is to "tunnel" traffic over public internet using an overlay (client and server inside of network):

Diagram:

This is a simple example, but in principle can create complex multi-hop VPNs.