CSCI-1680
DNS

Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, Scott Shenker and John Jannotti
• TCP milestone II: sign up for a meeting this week (announcement soon)

• TCP gearup III: tentative, but probably this Thursday 5-7pm

• HW3: due tonight—it’s short!

We’re working through our grading backlog, should have progress soon
Connecting to a server: the story so far

POV: You want to connect to some website

You

Some site 5.6.7.8
Connecting to a server: the story so far

POV: You want to connect to some website

You

Some site 5.6.7.8

connect(5.6.7.8, 80)

Is this how users interact with the network? No!
Why not? Why is this bad?

connect(5.6.7.8, 80)
Why not? Why is this bad?

- Need to know IP addresses!
  - Users won’t know now
  - Hosts don’t know—can’t remember every single one!

- Some host \( \neq \) its IP address? No!
  - A large website may be run by many servers
  - Devices may move between networks
What we have so far

IP addresses

• Used by routers to forward packets
• Fixed length, binary numbers
• Assigned based on where host is on the network
• Usually refers to one host
What we have so far

IP addresses
- Used by routers to forward packets
- Fixed length, binary numbers
- Assigned based on where host is on the network
- Usually refers to one host

Examples
- 5.6.7.8
- 212.58.224.138
- 2620:6e:6000:900:c1d:c9f7:8a1c:2f48
What we have

IP addresses
• Used by routers to forward packets
• Fixed length, binary numbers
• Assigned based on where host is on the network
• Usually refers to one host

Examples
• 5.6.7.8
• 212.58.224.138
• 2620:6e:6000:900:c1d:c9f7:8a1c:2f48

Efficient forwarding: ✅
Human readable: ❌
Scalable for distributed services: ❌
What we have

IP addresses
- Used by routers to forward packets
- Fixed length, binary numbers
- Assigned based on where host is on the network
- Usually refers to one host

Examples
- 5.6.7.8
- 212.58.224.138
- 2620:6e:6000:900:c1d:c9f7:8a1c:2f48

=> Need a new abstraction for "stuff" we are trying to access
What we want: a new abstraction for names

connect("website.com", 80)

connect("5.6.7.8", 80)
What we want: a new abstraction for names

- Human-readable
- Variable length
- Don't need to care about where destination is/what server it is

=> Can refer to a service, not just a host

connect("website.com", 80)
connect(5.6.7.8, 80)
What does this mean?

cs.brown.edu => 128.148.32.110

Why?
• Names are easier to remember
• Addresses can change underneath
• Useful Multiplexing/sharing
What does this mean?

```
cs.brown.edu => 128.148.32.110
```

Why?

- Names are easier to remember
- Addresses can change underneath
  - e.g., renumbering when changing providers
- Useful Multiplexing/sharing
  - One name -> multiple addresses
  - Multiple names -> one address
Another Change in Layers…

• Remember ARP
  – ARP: maps IP addresses to MAC addresses
The original way: one file: `hosts.txt`

- Flat namespace
- Central administrator kept master copy (for the Internet)
- To add a host, emailed admin
- Downloaded file regularly
<table>
<thead>
<tr>
<th>HOST NAME</th>
<th>HOST ADDRESS</th>
<th>SPONSOR</th>
<th>LIAISON</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACC</td>
<td>10.2.0.54</td>
<td>VDM</td>
<td>ARPA</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Lockwood, Gregory (LOCKWOOD@BNOC)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Associated Computer Consultants</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>414 East Cota Street</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Santa Barbara, California 93101</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(805) 965-1023</td>
</tr>
<tr>
<td>CPUtype</td>
<td>PDP-11/70(UNIX)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACCAT-TIP</td>
<td>10.2.0.35</td>
<td>ARPA</td>
<td>McBride, William T. (MCBRIDE@USC-ISIC)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Naval Ocean Systems Center</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Code 8321</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>271 Catalina Boulevard</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>San Diego, California 92152</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(714) 225-2083 (AV) 953-2083</td>
</tr>
<tr>
<td>CPUtype</td>
<td>H-316</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AEROSPACE</td>
<td>10.2.0.65</td>
<td>AFSC</td>
<td>Nelson, Louis C. (LOU@AEROSPACE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Aerospace Corporation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A2/1013</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>P.O. Box 92957</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Los Angeles, California 90009</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(213) 615-4424</td>
</tr>
<tr>
<td>CPUtype</td>
<td>VAX-11/780(UNIX)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AFGL</td>
<td>10.1.0.66</td>
<td>AFSC</td>
<td>Cosentino, Antonio (CSENTINOAFC-HQ)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Air Force Geophysics Laboratory</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SUNA</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mail Stop 30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Hanscom Air Force Base</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Massachusetts 01731</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(617) 861-4161 (AV) 478-4161</td>
</tr>
<tr>
<td>CPUtype</td>
<td>PDP-11/50(RSX11M)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AFGL-TAC</td>
<td>10.2.0.66</td>
<td>AFSC</td>
<td>Cosentino, Antonio (CSENTINOAFC-HQ)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Air Force Geophysics Laboratory</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SUNA</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mail Stop 30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Hanscom Air Force Base</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Massachusetts 01731</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(617) 861-4161 (AV) 478-4161</td>
</tr>
<tr>
<td>CPUtype</td>
<td>C/30</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The original way: one file: `hosts.txt`

- Flat namespace
- Central administrator kept master copy (for the Internet)
- To add a host, emailed admin
- Downloaded file regularly

Does it scale?
The original way: one file: hosts.txt

• Flat namespace
• Central administrator kept master copy (for the Internet)
• To add a host, emailed admin
• Downloaded file regularly

Does it scale?

Lol no.
Scalable (Address <-> Name) Mappings

Original way: one file: hosts.txt
- Flat namespace
- Central administrator kept master copy (for the Internet)
- To add a host, emailed admin
- Downloaded file regularly

Is this feasible today? Lol no.
Domain Name System (DNS)

- Originally proposed by RFC882, RFC883 (1983)

- Distributed protocol to translate hostnames -> IP addresses
  - Human-readable names
  - Delegated control
  - Load-balancing/content delivery
  - So much more...

=> Distributed key-value store, before it was cool…
Goals for DNS

- Scalability
- Distributed Control
- Fault Tolerance
Goals for DNS

• Scalability
  – Must handle a huge number of records
    • With some software synthesizing names on the fly
  – Must sustain update and lookup load

• Distributed Control
  – Let people control their own names

• Fault Tolerance
  – Minimize lookup failures in face of other network problems
The good news

Compared to other distributed systems, some properties that make these goals easier to achieve…

1. Read-mostly database
   Lookups MUCH more frequent than updates
2. Loose consistency
   When adding a machine, not end of the world if it takes minutes or hours to propagate

Can use lots and lots of caching
   - Once you’ve lookup up a hostname, remember
   - Don’t have to look again in the near future
The good news

Compared to other distributed systems, some properties that make these goals easier to achieve...
The good news

Compared to other distributed systems, some properties that make these goals easier to achieve...

1. Read-mostly database
   Lookups MUCH more frequent than updates

2. Loose consistency
   When adding a machine, not end of the world if it takes minutes or hours to propagate
How it works

Hierarchical namespace broken into zones

cslab1a.cs.brown.edu
How it works

• Hierarchical namespace broken into zones
  – root (.), edu., brown.edu., cs.brown.edu.,
  – Zones separately administered => delegation
  – Parent zone tells you how to find servers for subdomains

• Each zone served from multiple replicated servers
• Lots and lots of caching
Types of DNS servers
“Types” of DNS servers

- **Top Level Domain (TLD) servers**
  - Generic domains (e.g., com, org, edu)
  - Country domains (e.g., uk, br, tv, in, ly)
  - Special domains (e.g., arpa)
  - Corporate domains (…)

- **Authoritative DNS servers**
  - Provides public records for hosts at an organization
  - Can be maintained locally or by a service provider

- **Recursive resolvers**
  - Big public servers, or local to a network
  - Lots of caching
How a resolver works
Resolver operation

- Apps make **recursive** queries to local DNS server (1)
  - Ask server to get answer for you
- Server makes **iterative** queries to remote servers (2,4,6)
  - Ask servers who to ask next
  - Cache results aggressively

```
Resolver operation

• Apps make **recursive** queries to local DNS server (1)
  – Ask server to get answer for you
• Server makes **iterative** queries to remote servers (2,4,6)
  – Ask servers who to ask next
  – Cache results aggressively
```
Where is the root server?

- Located in New York
- How do we make the root scale?

Verisign, New York, NY
DNS Root Servers

• 13 Root Servers (www.root-servers.org)
  – Labeled A through M (e.g, A.ROOT-SERVERS.NET)
• Does this scale?
DNS Root Servers

• 13 Root Servers (www.root-servers.org)
  – Labeled A through M (e.g, A.ROOT-SERVERS.NET)
• Remember anycast?

A Verisign, New York, NY (also Frankfurt, HK, London, LA)
C Cogent, Herndon, VA (also Los Angeles, NY, Chicago, Frankfurt and 3+)
D U Maryland College Park, MD (also in 106 other locations)
G US DoD Columbus, OH (+5)  K RIPE London (plus 41 other locations)
H ARL Aberdeen, MD (also San Diego)
J Verisign (118 locations)

E NASA Mt View, CA (+70)
F Internet Software Consortium, Palo Alto, CA (and 57 other locations)

B USC-ISI Marina del Rey, CA
L ICANN Los Angeles, CA (plus 157 other locations)

I Netnod, Stockholm (plus 49 other locations)

M WIDE Tokyo
  plus Seoul, Paris, San Francisco, Osaka

DNS Root Servers

- 13 Root Servers (www.root-servers.org)
  - Labeled A through M (e.g, A.ROOT-SERVERS.NET)
- Remember anycast?

A Verisign, New York, NY (also Frankfurt, HK, London, LA)
C Cogent, Herndon, VA (also Los Angeles, NY, Chicago, Frankfurt and 3+)
D U Maryland College Park, MD (also in 106 other locations)
G US DoD Columbus, OH (+5)  K RIPE London (plus 41 other locations)
H ARL Aberdeen, MD (also San Diego)
J Verisign (118 locations)

E NASA Mt View, CA (+70)
F Internet Software Consortium, Palo Alto, CA (and 57 other locations)

B USC-ISI Marina del Rey, CA
L ICANN Los Angeles, CA (plus 157 other locations)

I Netnod, Stockholm (plus 49 other locations)

M WIDE Tokyo
  plus Seoul, Paris, San Francisco, Osaka
DNS Root Servers: Today

From: www.root-servers.org
DNS Caching

• Recursive queries are expensive
• Caching greatly reduces overhead
  – Top level servers very rarely change
  – Popular sites visited often
  – Local DNS server caches information from many users
• How long do you store a cached response?
  – Original server tells you: TTL entry
  – Server deletes entry after TTL expires
Reverse DNS

How do we get the other direction, IP address to name?

• Addresses have a natural hierarchy:
  – 128.148.32.12

• Idea: reverse the numbers: 12.32.148.128 …
  – and look that up in DNS

• Under what TLD?
  – Convention: in-addr.arpa
  – Lookup 12.32.148.128.in-addr.arpa
  – in6.arpa for IPv6
DNS Protocol

• TCP/UDP port 53
• Most traffic uses UDP
  – Lightweight protocol has 512 byte message limit
  – Retry using TCP if UDP fails (e.g., reply truncated)
• Bit in query determines if query is recursive
$ dig cs.brown.edu @10.1.1.10
; <<>> DiG 9.10.6 <<>> cs.brown.edu @10.1.1.10
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 8536
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1220
;; QUESTION SECTION:
;cs.brown.edu. IN A

;; ANSWER SECTION:
cs.brown.edu. 1800 IN A 128.148.32.12

;; Query time: 69 msec
;; SERVER: 10.1.1.10#53(10.1.1.10)
;; WHEN: Tue Apr 19 09:03:39 EDT 2022
;; MSG SIZE  rcvd: 57
% dig +norec cs.brown.edu @j.root-servers.net

; <<>> DiG 9.10.6 <<>> +norec cs.brown.edu @j.root-servers.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61618
;; flags: qr QUERY: 1, ANSWER: 0, AUTHORITY: 13, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;cs.brown.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 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
b.edu-servers.net. 172800 IN A 192.33.14.30
c.edu-servers.net. 172800 IN A 192.26.92.30
d.edu-servers.net. 172800 IN A 192.31.80.30
e.edu-servers.net. 172800 IN A 192.12.94.30

When server doesn’t know all info...
Example

dig . ns

dig +norec www.cs.brown.edu @a.root-servers.net

dig +norec www.cs.brown.edu @a.edu-servers.net

dig +norec www.cs.brown.edu @bru-ns1.brown.edu

www.cs.brown.edu. 86400 IN A 128.148.32.110
Resource Records

All DNS info represented as resource records (RR)

name [ttl] [class] type rdata

- name: domain name
- TTL: time to live in seconds
- class: for extensibility, normally IN (1) “Internet”
- type: type of the record
- rdata: resource data dependent on the type

• Example RRs

  www.cs.brown.edu.  86400 IN A  128.148.32.110
  cs.brown.edu.    86400 IN NS  dns.cs.brown.edu.
  cs.brown.edu.    86400 IN NS  ns1.ucsb.edu.
## DNS record types

<table>
<thead>
<tr>
<th>RR Type</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>IPv4 Address</td>
<td>128.148.56.2</td>
</tr>
<tr>
<td>AAAA</td>
<td>IPv6 Address</td>
<td>2001:470:8956:20::1</td>
</tr>
<tr>
<td>CNAME</td>
<td>Specifies an alias (&quot;Canonical name&quot;)</td>
<td>systems.cs.brown.edu. 86400 IN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>systems-v3.cs.brown.edu. 86400 IN A 128.148.36.51</td>
</tr>
<tr>
<td>MX</td>
<td>Mail servers</td>
<td>MX &lt;priority&gt; &lt;ip&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>eg. MX 10 1.2.3.4</td>
</tr>
<tr>
<td>SOA</td>
<td>Start of authority</td>
<td>Information about who owns a zone</td>
</tr>
<tr>
<td>PTR</td>
<td>Reverse IP lookup</td>
<td>7.34.148.128.in-addr.arpa. 86400 IN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PTR quanto.cs.brown.edu.</td>
</tr>
<tr>
<td>SRV</td>
<td>How to reach specific services (eg. host, port)</td>
<td>_minecraft._tcp.example.net 3600</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SRV &lt;priority&gt; &lt;weight&gt; &lt;port&gt; &lt;server IP&gt;</td>
</tr>
</tbody>
</table>

Inserting a Record in DNS

Your new startup helpme.com
Some important details

• How do local servers find root servers?
  – DNS lookup on a.root-servers.net?
  – Servers configured with root cache file
  – Contains root name servers and their addresses
    . 3600000 IN NS A.ROOT-SERVERS.NET.
    A.ROOT-SERVERS.NET. 3600000 A 198.41.0.4
    ...

• How do you get addresses of other name servers?
  – To obtain the address of www.cs.brown.edu, ask a.edu-servers.net, says a.root-servers.net
  – How do you find a.edu-servers.net?
  – Glue records: A records in parent zone
Other uses of DNS

• Local multicast DNS
  – Used for service discovery
  – Made popular by Apple
  – This is how you learn of different Apple TVs in the building

• Load balancing

• CDNs (more on this later)
Reliability

• Answers may contain several alternate servers
• Try alternate servers on timeout
  – Exponential backoff when retrying same server
• Use same identifier for all queries
  – Don’t care which server responds, take first answer
Inserting a Record in DNS

• Your new startup helpme.com
• Get a block of addresses from ISP
  – Say 212.44.9.0/24
• Register helpme.com at namecheap.com (for ex.)
  – Provide name and address of your authoritative name server (primary and secondary)
  – Registrar inserts RR pair into the .com TLD server:
    • helpme.com NS dns1.helpme.com
    • dns1.helpme.com A 212.44.9.120
• Configure your authoritative server (dns1.helpme.com)
  – Type A record for www.helpme.com
  – Type MX record for helpme.com
Inserting a Record in DNS, cont(51,43),(870,835)

• Need to provide reverse PTR bindings
  – E.g., 212.44.9.120 -> dns1.helpme.com
• Configure your dns server to serve the 9.44.212.in-addr.arpa zone
  – Need to add a record of this NS into the parent zone (44.212.in-addr.arpa)
• Insert the bindings into the 9.44.212.in-addr.arpa zone
DNS Security

- You go to Starbucks, how does your browser find www.google.com?
  - Ask local name server, obtained from DHCP

- Can you trust this DNS server?
Great Firewall of CIT

If attacker is on the path (say, it is the ISP, or a malicious version of TStaff), what could they do?

- Can sniff all DNS queries
- Send fake responses back first
- Could do this selectively, to direct facebook.com to cs.brown.edu, for example…
Great Firewall of CIT

If attacker is on the path (say, it is the ISP, or a malicious version of TStaff), what could they do?
Public DNS

Public DNS resolvers provided by cloud companies and ISPs
• 8.8.8.8 (Google)
• 1.1.1.1 (Cloudflare)
• ... and others

Why do this?
DNS: 8.8.8.8  
Alternatif: 8.8.4.4
“Helpful” ISPs

• Many ISPs hijack NXDOMAIN responses to “help” by offering search and advertisement related to the domain
• E.g., www.bicycleisntadomain.com doesn’t (currently) exist
  – Could return a page with search and ads on bicycles (or domain registrations?)
What can be done?

Some defenses against DNS spoofing/hijacking
What can be done?

Some defenses against DNS spoofing/hijacking

- **DNSSEC**: protocol to sign/verify hierarchy of DNS lookups
  - Expensive to deploy, hierarchy must support at all levels
  - APNIC DNSSEC monitor: [https://stats.labs.apnic.net/dnssec](https://stats.labs.apnic.net/dnssec)

- **Tunneling DNS**: client uses DNS via more secure protocol
  - DNS over HTTPS
  - DNS over TLS
More on DNS
Structure of a DNS Message

- Same format for queries and replies
  - Query has 0 RRs in Answer/Authority/Additional
  - Reply includes question, plus has RRs
- Authority allows for delegation
- Additional for glue, other RRs client might need
## Header format

- **Id**: match response to query; **QR**: 0 query/1 response
- **RCODE**: error code.
- **AA**: authoritative answer, **TC**: truncated,
- **RD**: recursion desired, **RA**: recursion available

```
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
| ID |                  |                  |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|QR| Opcode | AA| TC| RD| RA| Z | RCODE |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|QDCOUNT |                  |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|ANCOUNT |                  |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|NSCOUNT |                  |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|ARCOUNT |                  |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```
Other RR Types

• **CNAME (canonical name):** specifies an alias
  
  - www.l.google.com. 300 IN A 72.14.204.147

• **MX record:** specifies servers to handle mail for a domain (the part after the @ in email addr)
  - Different for historical reasons

• **SOA (start of authority)**
  - Information about a DNS zone and the server responsible for the zone

• **PTR (reverse lookup)**
  
  - 7.34.148.128.in-addr.arpa. 86400 IN PTR quanto.cs.brown.edu.