We conclude our study of common distributed systems with the protocol behind the Domain Name System.

Applications & Application-Layer Protocols: The Domain Name System

Jasleen Kaur
February 24, 2015

Application-Layer Protocols

Outline

- The architecture of distributed systems
  - Client/Server computing
- The programming model used in constructing distributed systems
  - Socket programming
- Example client/server systems and their application-layer protocols
  - The World-Wide Web (HTTP)
  - Reliable file transfer (FTP)
  - E-mail (SMTP & POP)
  - Internet Domain Name System (DNS)
Why don’t hosts/routers use hostnames for routing/addressing?

- Fixed length identifiers are simpler to deal with.

The DNS is a directory service that allows one to lookup the IP addresses of hostnames (and vice versa).

- As we will come to know it, the DNS is also an application-layer protocol running between end-systems (and using lower level transport services).

- But unlike the Web, SMTP, and FTP, it’s not an application that we invoke directly.

- It’s a fundamental service that all distributed applications use.

- And this use is typically transparent to users (and even applications).

- The performance of the DNS determines the performance of many applications such as the web.

- The DNS is also used to implement important system-level services such as load balancing.

**Application-Layer Protocols**

**The Domain Name System (DNS)**

- Computers (hosts, routers) connected to the Internet have two forms of names:
  - IP address — a 32 bit identifier used for addressing hosts and routing data to them
  - Hostname — an ASCII string used by applications

- The DNS is an Internet-wide service that provides mappings between IP addresses and hostnames:
  - The DNS is a distributed database implemented in a hierarchy of name servers
  - The DNS is also an application-layer protocol

- Hosts and routers use name servers to resolve names (address/name translation):
  - Name resolution is an essential Internet function implemented as application-layer protocol

**The Domain Name System**

**Name Hierarchy in DNS**

- Hostname = “dot” separated concatenation of domain names along path toward the root
  - unc.edu
  - cs.unc.edu
Name Hierarchy in the DNS

Top level domains

- Generic domains:
  - `com`, `org`, `net`, `edu`, `gov`, `mil`, `int`
  - `biz`, `info`, `name`, `pro`
- Special sponsored names
  - `.aero`, `.coop`, `.museum`
- Country code domains
  - `.uk`, `.de`, `.jp`, `.us`, etc.

Prices on GreatDomains.com (2/14/10)

- `flirt.com` 1,500,000
- `logo.com` 1,200,000
- `cold.com` 750,000
- `number.com` 750,000
- `compass.com` 700,000
- `jock.com` 500,000
- `jewel.com` 500,000
- `hunk.com` 500,000
- `conceive.com` 375,000
- `rhodes.com` 350,000
- `projects.com` 350,000

- `belly.com` 250,000
- `rumors.com` 250,000
- `biker.com` 250,000
- `sox.com` 150,000
- `uwant.com` 150,000
- `vancouverbc.com` 150,000
- `hangar.com` 125,000
- `shift.com` 100,000
This is a big directory service! And one that is experiencing exponential growth. Started growing with the advent of the world-wide-web. I remember when there were less than 1,000 hosts!

Recall that DNS lookup is done several times in a single web-download — an additional potential lookup for each embedded object in the web-page. Really, your web-proxy-server does the DNS lookup — your browser always goes to the proxy for any web-request. If the proxy doesn’t have the object, it’ll do a DNS lookup and open a socket to the intended web-server.

Politically, the privilege of deciding who gets what domain name can not be with just one political government.

### The Domain Name System

**Designing a distributed service**

- Why not centralize the DNS
  - A server process on a big, well connected supercomputer?
- Centralized systems do not scale!
  - Poor reliability: centralized = single point of failure
  - Poor performance: centralized = “remote access” for most users
  - Difficult to manage: centralized = all customer traffic goes to one location, a large staff has to be present to handle registrations
- A centralized system is not politically feasible in an international network
There are three types of servers:

- **Local name servers** (also called a default name server).
  Every machine knows about its local name server.

- **Authoritative name servers**.
  The authoritative name server can perform name/address translation for that host’s name/address.

- **Local name servers**:
  - Each ISP, university, company, has a local (default) name server authoritative for its own hosts.
  - *Resolvers* always query a name server local to it to resolve any host name.

- A root name server is contacted when a local name server that can’t resolve a name
  - The root server either resolves the name or provides pointers to authoritative servers at lower level of name hierarchy

- There are approx. a dozen root name servers worldwide
This example assumes that the root server knew the identity of the authoritative server for the yahoo.com domain. What if the root server didn’t know this?

(The root server can’t be expected to know the authoritative server for every domain!)
The root name server may not know the authoritative name server for a domain but it may know a server that is close to the requested domain.

Note that this example reverses the data flow.

— Yahoo is now trying to route data back to the cs.unc.edu domain.

What are the trade-offs between using/supporting recursive and iterated queries?

— With recursive queries a contacted server has to maintain state—remember what request it is waiting for and maybe timeout if it doesn’t get a response.

— Also scalability issues with recursive queries.

— Recursive queries place the burden of name resolution (recursively) on the contacted server.

— In an iterated query the contacted server simply replies with the name of the server to contact—“I don’t know; try asking X.”
Note that iterated and recursive queries can be combined.

A query may start as recursive, then be iterated through a few servers, then recursive again.

What are the trade-offs between using/supporting recursive and iterated queries?

With recursive queries you have to maintain state (remember what you are waiting for and maybe timeout if you don’t get a response).

The local DNS server sends the same query to the closest match server.

“What server is the authority for www.yahoo.com?”

The process can be iterated until the local authoritative name server is found and responds.
DNS Name Servers
Caching and updating DNS entries

- Every server caches all the mappings it learns
  - Cache entries are "soft state"
  - They timeout (are deleted) after some time period

- DNS cache update/notify mechanisms under design by the IETF
  - See RFC 2136

DNS Name Servers
DNS resource records

- The DNS is a distributed database storing resource records (RRs)

- Type = A
  - name is a hostname
  - value is hostname’s IP address

- Type = NS
  - name is a domain
  - value is name of authoritative name server for this domain

- Type = CNAME
  - name is an alias name for some "canonical" (the real) name
  - value is canonical name

- Type = MX
  - value is name of mail server host associated with name

Type = A records are the standard hostname-to-IP address mappings.

NS records are used for routing requests towards authoritative name servers.

The idea behind CNAME records is that a hostname may be an alias (e.g., there really is no www.yahoo.com).

The CNAME record gives the real (canonical) name of the aliased host.

MX records are specifically for finding out the name of a domain’s mail server.

- If you need to send mail to bob@hotmail.com you first need to look up the name of hotmail’s mail server.

- The MX record permits a company’s mail server and web server to have identical (aliased) hostnames (e.g., enterprise.com).
The DNS protocol uses UDP on port 53. Identifier needed in DNS messages because DNS runs on UDP, and not TCP.

Why does the DNS use UDP (and not TCP)?

- Reliability, strictly speaking, isn't needed.
- If a request is lost or if there is no reply, you simply try another server. (A form of application-specific error recovery.)
- Also, the overhead of TCP's handshaking isn't worth it for messages that typically can be sent in a single packet.

All header fields are 16-bits wide.

---

### DNS Protocol

#### DNS query and reply messages

- DNS query and reply messages both have the same message format
- Messages have a fixed length message header
  - Identification — 16 bit query/reply identifier used to match replies to queries
  - Flags:
    - Query/Reply bit
    - "Reply is authoritative" bit
    - "Recursion desired" bit
    - ...

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td>2 bytes</td>
</tr>
<tr>
<td>Flags</td>
<td>1 byte</td>
</tr>
<tr>
<td>Number of questions</td>
<td>variable</td>
</tr>
<tr>
<td>Number of answer RRs</td>
<td>variable</td>
</tr>
<tr>
<td>Number of authority RRs</td>
<td>variable</td>
</tr>
<tr>
<td>Number of additional RRs</td>
<td>variable</td>
</tr>
<tr>
<td>Questions (variable number)</td>
<td>variable</td>
</tr>
<tr>
<td>Answers (variable number of records)</td>
<td>variable</td>
</tr>
<tr>
<td>Authority (variable number of records)</td>
<td>variable</td>
</tr>
<tr>
<td>Additional Information (variable number of records)</td>
<td>variable</td>
</tr>
</tbody>
</table>

---

### DNS Protocol

#### DNS query and reply messages

- Messages have a variable-length “question & answer” body
- Questions:
  - The name and type fields (type A or MX) for a query — hotmail.com MX
- Answers:
  - One RR for each IP address answering query
- Authority:
  - Resource records of other authoritative servers

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td>2 bytes</td>
</tr>
<tr>
<td>Flags</td>
<td>1 byte</td>
</tr>
<tr>
<td>Number of questions</td>
<td>variable</td>
</tr>
<tr>
<td>Number of answer RRs</td>
<td>variable</td>
</tr>
<tr>
<td>Number of authority RRs</td>
<td>variable</td>
</tr>
<tr>
<td>Number of additional RRs</td>
<td>variable</td>
</tr>
<tr>
<td>Questions (variable number)</td>
<td>variable</td>
</tr>
<tr>
<td>Answers (variable number of records)</td>
<td>variable</td>
</tr>
<tr>
<td>Authority (variable number of records)</td>
<td>variable</td>
</tr>
<tr>
<td>Additional Information (variable number of records)</td>
<td>variable</td>
</tr>
</tbody>
</table>
The domain name system (DNS) is a distributed database that maps domain names to IP addresses. Each DNS server maintains a cache of recently resolved name-to-address mappings, which it uses to answer subsequent requests for the same names.

The class on the question specifies that how one is to limit the search according to the protocol group (classname) for which lookup information is desired.

- **IN** == Internet group (the default).

A RR is given for each (of the 9) DNS servers answering the query.

- Yahoo has a large number of authoritative name servers.

- DNS can be used for load-balancing.

- DNS can also be used for redirecting to a content-distribution provider. For example, Akamai inserts DNS records that says www.cnn.com is an alias for a xxx.akamai.com server.

- Location-specific alias resolution (different alias in different DNS servers).

The ttl is given in seconds.

- 160,393 seconds = 44.5 hours.
**DNS Resource Records**

*nslookup query/reply message example*

```
ADDITIONAL RECORDS:
- > za.akadns.org
  internet address = 195.219.3.169
- > zb.akadns.org
  internet address = 206.132.100.105
- > zc.akadns.org
  internet address = 124.211.40.4
- > zd.akadns.org
  internet address = 63.209.3.132
- > eur1.akadns.net
  internet address = 213.254.204.197
- > use3.akadns.net
  internet address = 204.2.178.133
- > use4.akadns.net
  internet address = 208.44.108.137
- > use2.akadns.net
  internet address = 63.209.3.132
- > asia3.akadns.net
  internet address = 220.73.220.4
```

Non-authoritative answer:

```
Name:  www.yahoo-bt3.akadns.net
Address:  69.147.314.210
```
**DNS Example**

**DNS processing for an iterated query**

- Resolve the hostname `www2.cnn.com` with a warm cache

<table>
<thead>
<tr>
<th>com.</th>
<th>NS</th>
<th>A.ROOT-SERVERS.NET</th>
<th>198.41.0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>cnn.com</td>
<td>NS</td>
<td>DNS.CNN.COM</td>
<td>128.2.232.1</td>
</tr>
<tr>
<td><a href="http://www.cnn.com">www.cnn.com</a></td>
<td>A</td>
<td>207.25.71.28</td>
<td></td>
</tr>
</tbody>
</table>

**The Domain Name System**

**Summary**

- F gets 270,000,000+ hits per day
  - Other servers have comparable load
- The Verisign TLD servers answer 5,000,000,000 queries per day
- Clearly the DNS would collapse without:
  - Hierarchy
  - Distributed processing
  - Caching

- If DNS fails, Internet services stop working!