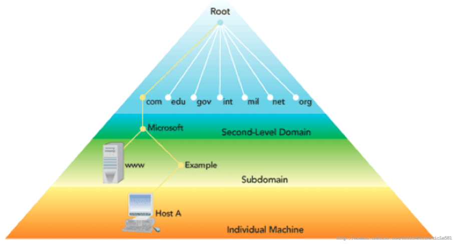
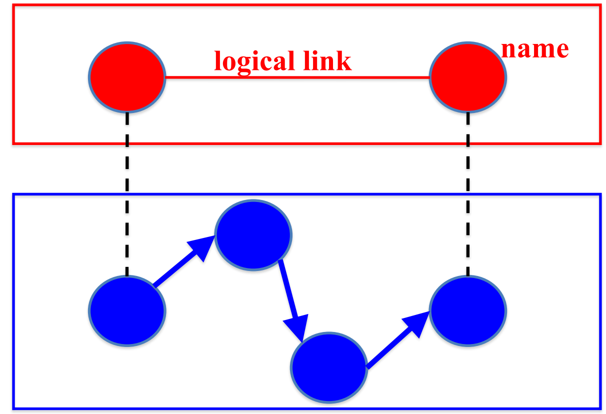


Discovery and DNS

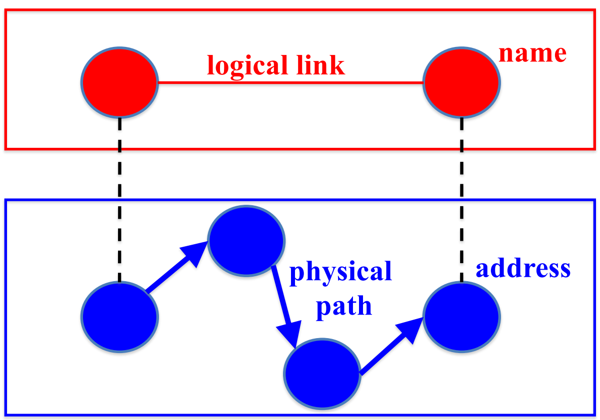
Mike Freedman
COS 461: Computer Networks

http://www.cs.princeton.edu/courses/archive/spr20/cos461/

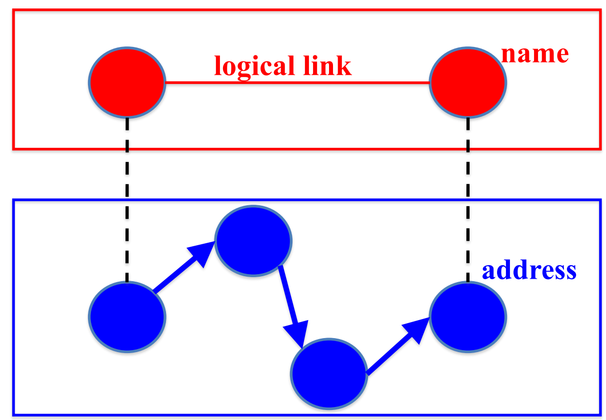
Relationship Between Layers



Routing: Mapping Link to Path



Discovery: Mapping Name to Address



Discovery

Directories

- A key-value store
  - Key: name; value: address(es)
  - Answer queries: given name, return address(es)
- Caching the response
  - Reuse the response, for a period of time
  - Better performance and lower overhead
- Allow entries to change
  - Updating the address(es) associated with a name
  - Invalidating or expiring cached responses

Directory Design: Three Extremes

- Flood the query (e.g., ARP)
  - The named node responds with its address
  - But, high overhead in large networks
- Push data to all clients (/etc/hosts)
  - All nodes store a full copy of the directory
  - But, high overhead for many names and updates
- Central directory server
  - All data and queries handled by one machine
  - But, poor performance, scalability, and reliability

Directory Design: Distributed Solutions

- Hierarchical directory (e.g., DNS)
  - Follow the hierarchy in the name space
  - Distribute the directory, distribute the queries
  - Enable decentralized updates to the directory
- Distributed Hash Table (e.g. P2P applications)
  - Directory as a hash table with flat names
  - Each directory node handles range of hash outputs
  - Use hash to direct query to the directory node

Domain Name System (DNS)

Computer science concepts underlying DNS

- Indirection: names in place of addresses
- Hierarchy: in names, addresses, and servers
- Caching: of mappings from names to/from addresses

9

Strawman Solution #1: Local File

- Original name to address mapping
  - Flat namespace
  - /etc/hosts
  - SRI kept main copy
  - Downloaded regularly
- Count of hosts was increasing: moving from a machine per domain to machine per user
  - Many more downloads
  - Many more updates

10

Strawman Solution #2: Central Server

- Central server
  - One place where all mappings are stored
  - All queries go to the central server
- Many practical problems
  - Single point of failure
  - High traffic volume
  - Distant centralized database
  - Single point of update
  - Does not scale

Need a distributed, hierarchical collection of servers

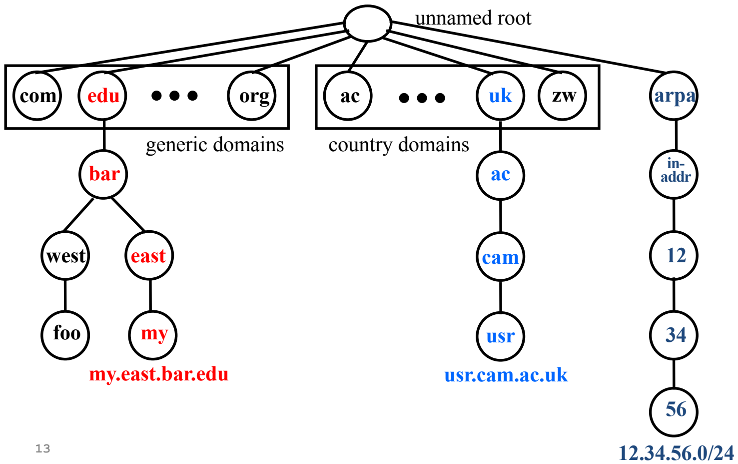
11

Domain Name System (DNS)

- Properties of DNS
  - Hierarchical name space divided into zones
  - Distributed over a collection of DNS servers
- Hierarchy of DNS servers
  - Root servers
  - Top-level domain (TLD) servers
  - Authoritative DNS servers
- Performing the translations
  - Local DNS servers and client resolvers

12

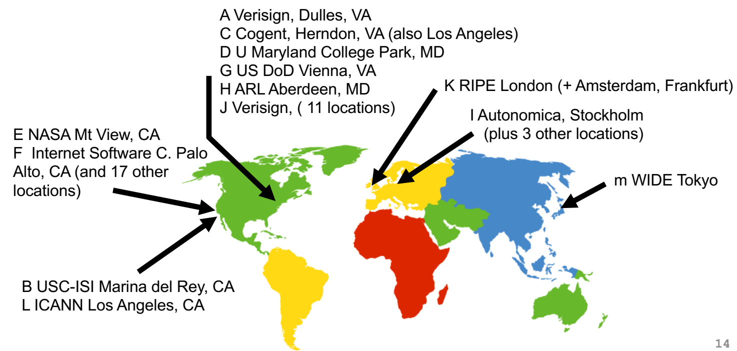
Distributed Hierarchical Database



13

DNS Root Servers

- 13 root servers (see http://www.root-servers.org/)
- Labeled A through M. Most are IP Anycasted.



14

TLD and Authoritative DNS Servers

- Global Top-level domain (gTLD) servers
  - Generic domains (e.g., .com, .org, .edu)
  - Country domains (e.g., .uk, .fr, .ca, .jp)
  - Managed professionally (e.g., Verisign for .com .net)
- Authoritative DNS servers
  - Provide public records for hosts at an organization
  - For the organization's servers (e.g., Web and mail)
  - Can be maintained locally or by a service provider

15

Reliability

- DNS servers are replicated
  - Name service available if at least one replica is up
  - Queries can be load balanced between replicas

```
$ dig NS nytimes.com +norecurse

;; QUESTION SECTION:
;nytimes.com.                    IN      NS

;; AUTHORITY SECTION:
nytimes.com.             349     IN      NS      ns2.p24.dynect.net.
nytimes.com.             349     IN      NS      ns3.p24.dynect.net.
nytimes.com.             349     IN      NS      dns2.p06.nsone.net.
nytimes.com.             349     IN      NS      ns4.p24.dynect.net.
nytimes.com.             349     IN      NS      ns1.p24.dynect.net.
nytimes.com.             349     IN      NS      dns3.p06.nsone.net.
nytimes.com.             349     IN      NS      dns4.p06.nsone.net.
nytimes.com.             349     IN      NS      dns1.p06.nsone.net.
```

16

Reliability

- DNS servers are replicated
  - Name service available if at least one replica is up
  - Queries can be load balanced between replicas
- UDP used for queries
  - Need reliability: must implement this on top of UDP
- Try alternate servers on timeout
  - Exponential backoff when retrying same server
- Same identifier for all queries
  - Don't care which server responds

DNS Queries and Caching

Using DNS

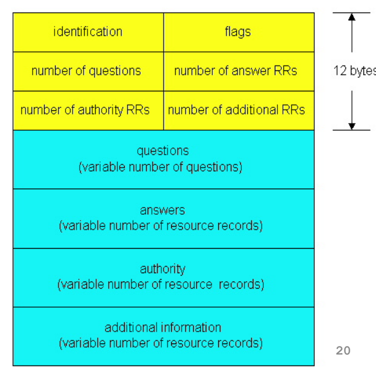
- Local DNS server ("default name server")
  - Usually near the end hosts who use it
  - Local hosts configured with local server (e.g., /etc/resolv.conf) or learn the server via DHCP
- Client application
  - Extract server name (e.g., from the URL)
  - Do *gethostbyname()* or *getaddrinfo()* to get address
- Server application
  - Extract client IP address from socket
  - Optional *gethostbyaddr()* to translate into name

DNS Protocol

DNS protocol : *query* and *reply* msg, both with same *msg format*

Message header

- Identification: 16 bit # for query, reply to query uses same #
- Flags:
  - Query or reply
  - Recursion desired
  - Recursion available
  - Reply is authoritative



DNS Resource Records

RR format: `(name, value, type, ttl)`

- Type=A
  - **Name**: hostname
  - **Value**: IP address
- Type=NS
  - **Name**: domain
  - **Value**: hostname of name server for domain
- Type=CNAME
  - **Name**: alias for some "canonical" (the real) name: www.ibm.com is really srveast.backup2.ibm.com
  - **Value**: canonical name
- Type=MX
  - **Value**: name of mailserver associated with **name**

21

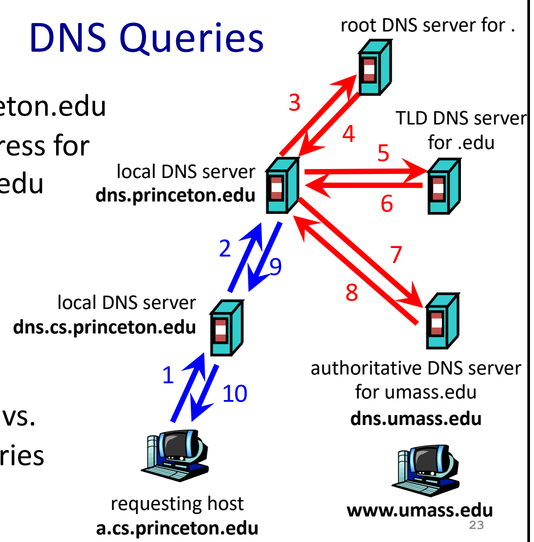
Break for Demo

22

DNS Queries

Host a.cs.princeton.edu wants IP address for www.umass.edu

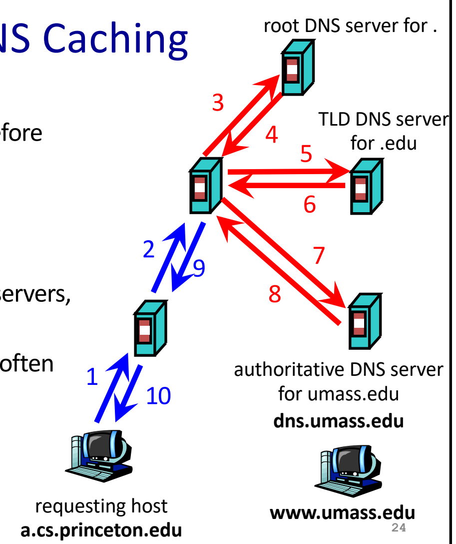
Note Recursive vs. Iterative Queries



23

DNS Caching

- DNS query latency
  - E.g., 1 sec latency before starting a download
- Caching to reduce overhead and delay
  - Small # of top-level servers, that change rarely
  - Popular sites visited often
- Where to cache?
  - Local DNS server
  - Browser



24

```
$ dig nytimes.com +norecurse @a.root-servers.net

;; QUESTION SECTION:
;nytimes.com. IN A

;; AUTHORITY SECTION:
com. 172800 IN NS a.gtld-servers.net.
com. 172800 IN NS b.gtld-servers.net.
com. 172800 IN NS c.gtld-servers.net.
com. 172800 IN NS d.gtld-servers.net.
com. 172800 IN NS e.gtld-servers.net.
com. 172800 IN NS f.gtld-servers.net.
com. 172800 IN NS g.gtld-servers.net.
com. 172800 IN NS h.gtld-servers.net.
com. 172800 IN NS i.gtld-servers.net.
com. 172800 IN NS j.gtld-servers.net.
com. 172800 IN NS k.gtld-servers.net.
com. 172800 IN NS l.gtld-servers.net.
com. 172800 IN NS m.gtld-servers.net.

;; ADDITIONAL SECTION:
a.gtld-servers.net. 172800 IN A 192.5.6.30
b.gtld-servers.net. 172800 IN A 192.33.14.30
c.gtld-servers.net. 172800 IN A 192.26.92.30
d.gtld-servers.net. 172800 IN A 192.31.80.30
e.gtld-servers.net. 172800 IN A 192.12.94.30
f.gtld-servers.net. 172800 IN A 192.35.51.30
```



```
$ dig nytimes.com +norecurse @b.gtld-servers.net

;; QUESTION SECTION:
;nytimes.com. IN A

;; AUTHORITY SECTION:
nytimes.com. 172800 IN NS ns3.p24.dynect.net.
nytimes.com. 172800 IN NS ns1.p24.dynect.net.
nytimes.com. 172800 IN NS ns2.p24.dynect.net.
nytimes.com. 172800 IN NS ns4.p24.dynect.net.
nytimes.com. 172800 IN NS dns1.p06.nsone.net.
nytimes.com. 172800 IN NS dns2.p06.nsone.net.
nytimes.com. 172800 IN NS dns3.p06.nsone.net.
nytimes.com. 172800 IN NS dns4.p06.nsone.net.

;; Query time: 11 msec
;; SERVER: 192.33.14.30#53(192.33.14.30)
;; WHEN: Sat Mar 28 10:56:03 2020
;; MSG SIZE  rcvd: 201
```



```
$ dig nytimes.com +norecurse @ns3.p24.dynect.net

;; QUESTION SECTION:
;nytimes.com. IN A

;; ANSWER SECTION:
nytimes.com. 500 IN A 151.101.193.164
nytimes.com. 500 IN A 151.101.129.164
nytimes.com. 500 IN A 151.101.65.164
nytimes.com. 500 IN A 151.101.1.164

;; AUTHORITY SECTION:
nytimes.com. 300 IN NS ns2.p24.dynect.net.
nytimes.com. 300 IN NS ns4.p24.dynect.net.
nytimes.com. 300 IN NS ns3.p24.dynect.net.
nytimes.com. 300 IN NS ns1.p24.dynect.net.
nytimes.com. 300 IN NS dns3.p06.nsone.net.
nytimes.com. 300 IN NS dns2.p06.nsone.net.
nytimes.com. 300 IN NS dns4.p06.nsone.net.
nytimes.com. 300 IN NS dns1.p06.nsone.net.

;; Query time: 14 msec
;; SERVER: 208.78.71.24#53(208.78.71.24)
;; WHEN: Sat Mar 28 11:23:19 2020
;; MSG SIZE  rcvd: 265
```



```
$ dig ANY nytimes.com +norecurse @ns3.p24.dynect.net
;; Truncated, retrying in TCP mode.

;; QUESTION SECTION:
;nytimes.com. IN ANY

;; ANSWER SECTION:
nytimes.com. 300 IN SOA dns1.p06.nsone.net.
hostmaster.nytimes.com. 2019121930 300 150 1209600 300
nytimes.com. 300 IN NS dns3.p06.nsone.net.
nytimes.com. 300 IN NS dns1.p06.nsone.net.
nytimes.com. 300 IN NS dns4.p06.nsone.net.
nytimes.com. 300 IN NS ns3.p24.dynect.net.
nytimes.com. 300 IN NS ns4.p24.dynect.net.
nytimes.com. 300 IN NS ns2.p24.dynect.net.
nytimes.com. 300 IN NS ns1.p24.dynect.net.
nytimes.com. 300 IN NS dns2.p06.nsone.net.
nytimes.com. 500 IN A 151.101.129.164
nytimes.com. 500 IN A 151.101.193.164
nytimes.com. 500 IN A 151.101.1.164
nytimes.com. 500 IN A 151.101.65.164
nytimes.com. 300 IN MX 10 ASPMX2.GOOGLEMAIL.COM.
nytimes.com. 300 IN MX 10 ASPMX3.GOOGLEMAIL.COM.
nytimes.com. 300 IN MX 1 ASPMX.L.GOOGLE.com.
nytimes.com. 300 IN MX 5 ALT2.ASPMX.L.GOOGLE.com.
nytimes.com. 300 IN MX 5 ALT1.ASPMX.L.GOOGLE.com.
```

```
nytimes.com. 500 IN TXT "google-site-verification=aReMr8hkX3gxeHLKKk4tJ1s970U7QdEqUMIhMmLUfjQ"
nytimes.com. 500 IN TXT "dropbox-domain-verification=41d3jahx0psi"
nytimes.com. 500 IN TXT "google-site-verification=ZTCMdpSKM7HwqTvGUf_00Ef008JhOnbzGgCSUGYfsro"
nytimes.com. 500 IN TXT "docusign=bd506110-db79-430e-b159-cc1d74fe1176"
nytimes.com. 500 IN TXT "google-site-verification=NIqXa_F8IaqdPJhTtexgR0NYbzVLD_-X-uRUvyf4GyQ"
nytimes.com. 500 IN TXT "MS=ms22827202"
nytimes.com. 500 IN TXT "adobe-idp-site-verification=5ce4d99c-af0a-4b76-9217-bd49d3336df0"
nytimes.com. 500 IN TXT "google-site-verification=4qJm5sAZa1_29BTwFjqW09t7_7D4Vee3LBFQqg8xYbs"
nytimes.com. 500 IN TXT "google-site-verification=ZsySMeZ_SRbJZFu-53ptepytP7h5pxHO0qAg8Z2bKug"
nytimes.com. 500 IN TXT "MS=A1BFCA84E21B7011CA98DF9DC251CDDF90E0174B"
nytimes.com. 500 IN TXT "google-site-verification=4TE2ggBoy6PktLjtZ03t32A2oEZ0VD0PY6MnTj8IL_g"
nytimes.com. 500 IN TXT "v=spf1 mx ptr ip4:170.149.160.0/19 ip4:170.149.240.0/19
ip4:209.11.220.51/32 include:alerts.wallst.com include:authsmtp.com include:sendgrid.net
include:_spf.google.com include:inyt.com include:_spf.e.sparkpost.com include:mail.zendesk.com
~all"
nytimes.com. 500 IN TXT "google-site-verification=ic0Ur9LVhZ1hIjTj8LhMOhqx2Z52oS3hU_W2EqBY6UM"
nytimes.com. 500 IN TXT "google-site-verification=jZcmQFxPEP38yqYpmRvo0v_9hQFAdBZPUEBwTNUPUF8"
nytimes.com. 500 IN TXT "253961548-4297453"
nytimes.com. 500 IN TYPE257 \# 22 000569737375656C657473656E63727970742E6F7267
nytimes.com. 500 IN TYPE257 \# 19 0005697373756563 6F6D6F646F63612E636F6D
nytimes.com. 500 IN TYPE257 \# 19 00056973737565617773747275 73742E636F6D
nytimes.com. 500 IN TYPE257 \# 22 0005697373756561 6D617A6F6E74727573742E636F6D
nytimes.com. 500 IN TYPE257 \# 15 000569737375657 06B692E676F6F67
nytimes.com. 500 IN TYPE257 \# 19 00056973737565646967696365 72742E636F6D
nytimes.com. 500 IN TYPE257 \# 17 0005697373756561 6D617A6F6E2E636F6D
nytimes.com. 500 IN TYPE257 \# 19 00056973737565 73796D616E7465632E636F6D
nytimes.com. 500 IN TYPE257 \# 20 0005697373756561 6D617A6F6E6177732E636F6D
```

29

DNS Cache Consistency

- Goal: Ensuring cached data is up to date
- DNS design considerations
  - Cached data is "read only"
  - Explicit invalidation would be expensive
    - Server would need to keep track of all resolvers caching
- Avoiding stale information
  - Responses include a "time to live" (TTL) field
  - Delete the cached entry after TTL expires
- Perform negative caching (for dead links, misspellings)
  - So failures quick and don't overload gTLD servers

30

Setting the Time To Live (TTL)

- TTL trade-offs
  - Small TTL: fast response to change
  - Large TTL: higher cache hit rate
- Following the hierarchy
  - Top of the hierarchy: days or weeks
  - Bottom of the hierarchy: seconds to hours
- Tension in practice
  - CDNs set low TTLs for load balancing and failover
  - Browsers cache for 15-60 seconds

31

Questions

- Tension:
  - DNS operators want high TTL for low load on DNS servers,
  - Domains want low TTL for faster failover b/w IP addrs

  (Y) True (M) False

- By returning IP addresses in "round robin" fashion, DNS operators can ensure equal load better servers

  (Y) True (M) False

- Most applications obey TTLs on DNS records

  (Y) True (M) False

32

Questions

- Tension:
  - DNS operators want high TTL for low load on DNS servers,
  - Domains want low TTL for faster failover b/w IP addrs

  $\boxed{\text{(Y) True}}$ (M) False

- By returning IP addresses in "round robin" fashion, DNS operators can ensure equal load better servers

  (Y) True $\boxed{\text{(M) False}}$

- Most applications obey TTLs on DNS records

  (Y) True $\boxed{\text{(M) False}}$

33

Inserting Resource Records into DNS

- Example: just created startup "FooBar"
- Register foobar.com at namecheap.com
  - Provide registrar with names and IP addresses of authoritative name server (primary and secondary)
  - Registrar inserts two RRs into the com TLD server:
    - (foobar.com, dns1.foobar.com, NS)
    - (dns1.foobar.com, 212.212.212.1, A)
- Put in authoritative server dns1.foobar.com
  - Type A record for www.foobar.com
  - Type MX record for foobar.com

34

DNS attacks (1)

- DNS cache poisoning
  - Client: Ask for www.evil.com
  - Attacker responds with additional section for (www.cnn.com, 1.2.3.4, A)
  - Client/resolver: Thanks! I won't bother check what I asked for.

35

DNS attacks (2)

- DNS hijacking
  - Attacker sends forged DNS reply to client for www.cnn.com, *even when they don't receive the request*
  - How to prevent?
    - Client remembers the 16-bit DNS ID
    - Client only accepts reply if reply ID matches query ID
  - 16 bits: 65K possible IDs
    - What rate for attacker to enumerate all in 1 sec? 64B/packet
    - 64*65536*8 / 1024 / 1024 = 32 Mbps
  - Prevention: Also randomize the DNS source port
    - e.g., Windows DNS alloc's 2500 DNS ports, leads to ~164M possible IDs
    - Would require 80 Gbps
    - Kaminsky attack: this source port…wasn't random after all

36