DNS

Karst Koymans & Niels Sijm

Informatics Institute

University of Amsterdam

Tuesday, September 7, 2012

Outline

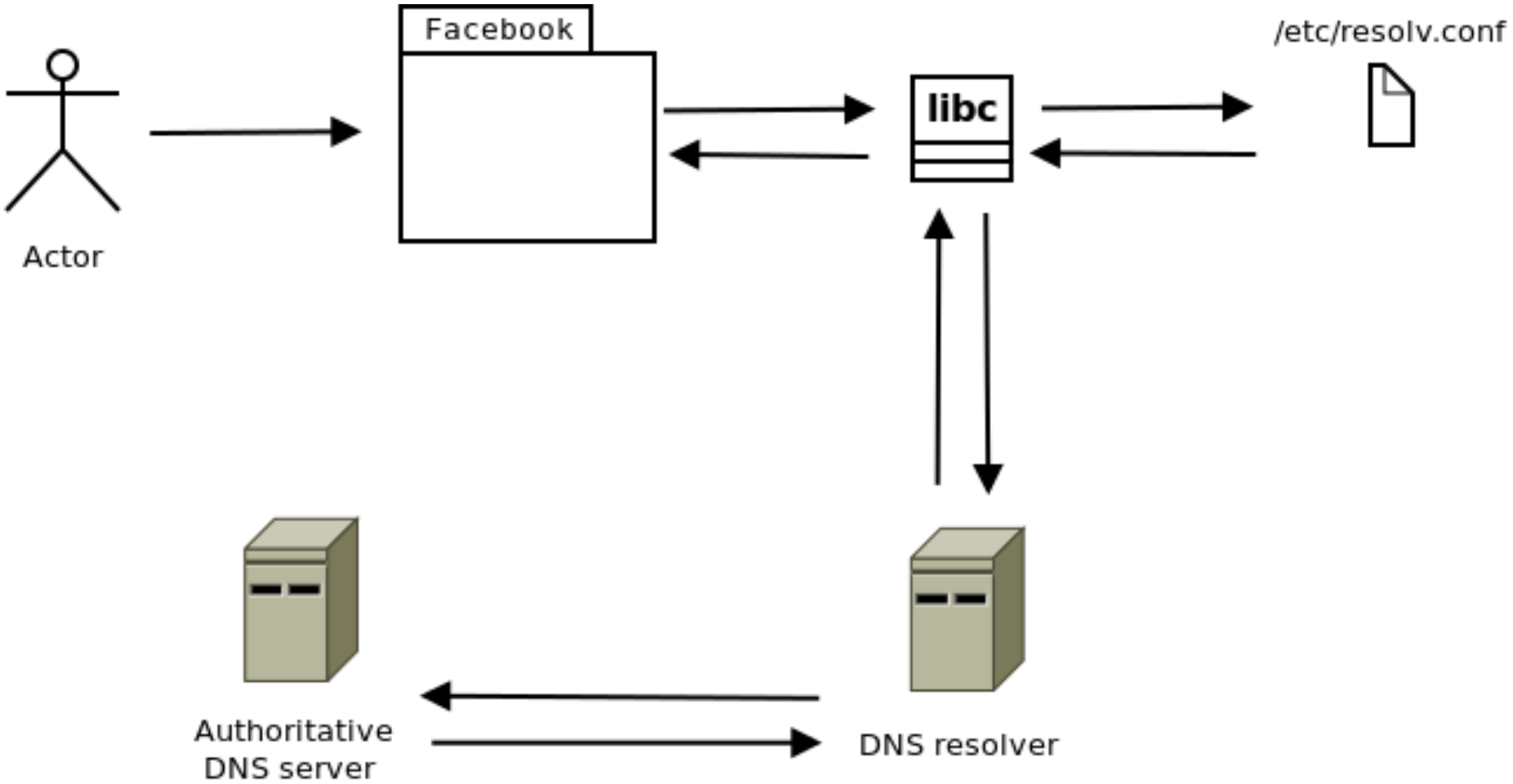
DNS: primary usecase

- DNS = Domain Name System
  - Something to do with domain names
  - Something to do with IP addresses
- Question: what is the IP address of www.google.com?
  - Client asks server
  - Server responds with answer
  - . . . case closed?

DNS: secundary usecases

- E-mail routing
  - Where to deliver e-mail for Niels.Sijm@os3.nl?
  - Question: what is the IP address of os3.nl?
    - What about mail for subdomains?
    - What about mail for *other* domains?
- IPv6 addresses
  - Question: what is the IP address of www.google.com?
  - Answer: *1.2.3.4* or *1:2:3:4:5:6:7:8*?

Important entities in DNS



1st Architectural option: centralized

- One single DNS server: 1.1.1.1
  - Simple: one place for all your questions!
  - SPoF (Single Point of Failure)
- Multiple DNS servers: 1.1.1.1, 2.2.2.2, 3.3.3.3, ...
  - Simple: multiple pre-defined places for all your questions!
  - Easy to remember, easy to use, resilient to network failures.
- Scaling issues
  - Peanutes for 10 hosts; undoable for 1,000,000,000 hosts
  - Network traffic does not scale
  - Administration of database becomes infeasable too

2nd Architectural option: decentralized

- Use a hierarchy instead of one big flat master file
  - Solves all of your scaling issues
  - Need to tweak protocol to redirect questions
  - Seems simple, introduces quite some issues
- How to split up the database?
  - Use subdomain to split up database
  - Use the first letter of a domain name
  - Create a cryptographical hash and use fist octet
  - Use /dev/random and write down the outcome
  - ...

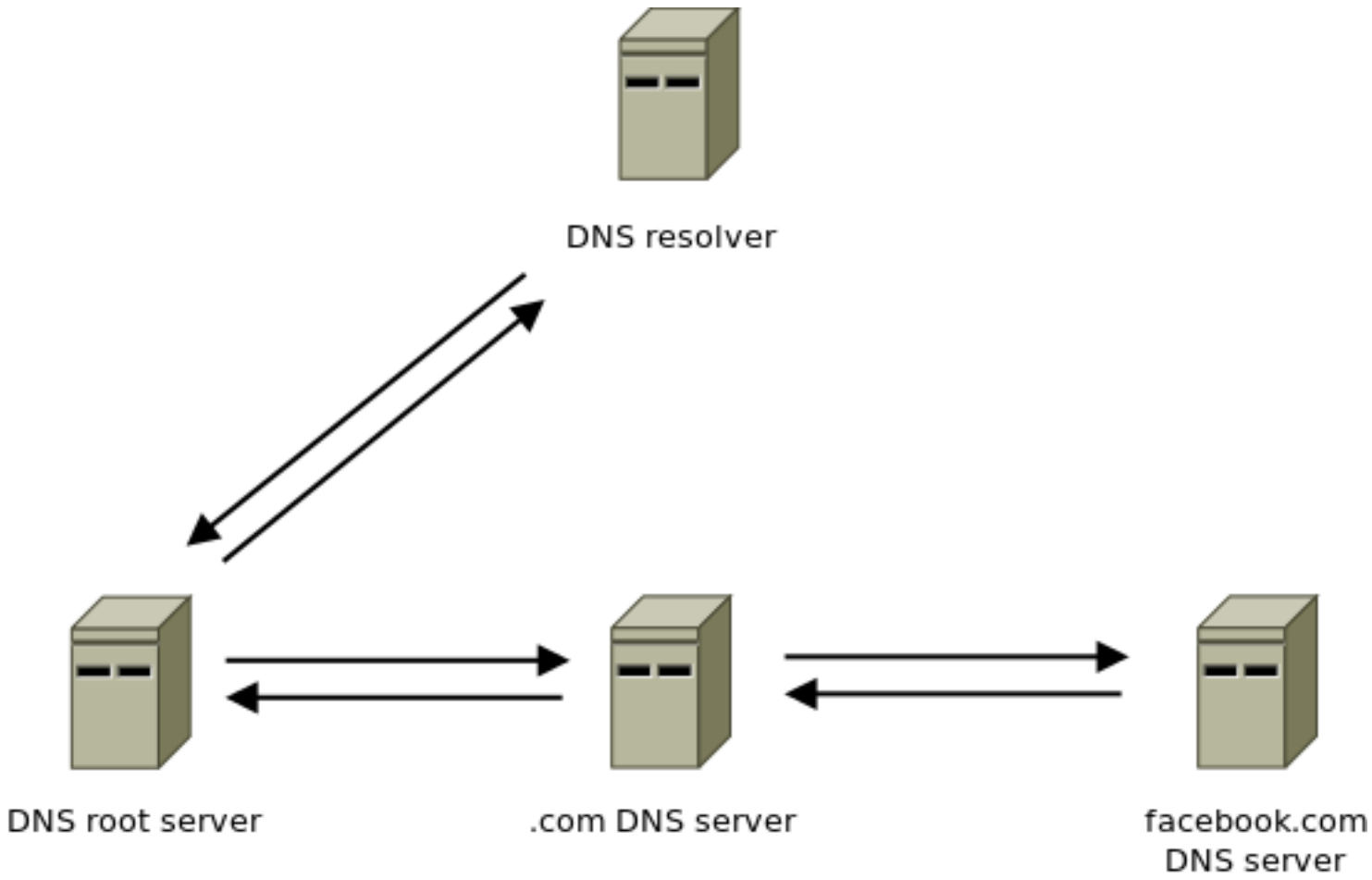
3rd Architectural option: distributed

- Not a hierarchy but an "unmananged" network
  - Who own what part of the database?
    - Distributed Hash Table (DHT) works well in practise
  - Works for P2P networks and BitCoins and CDNs and Skype and...
- Authority problems
  - Can you hijack a part of the database as in a DHT?
  - What if a nodes goes down?
    - How to duplicate information?
    - How to redirect questions?
    - How to keep information up-to-date?

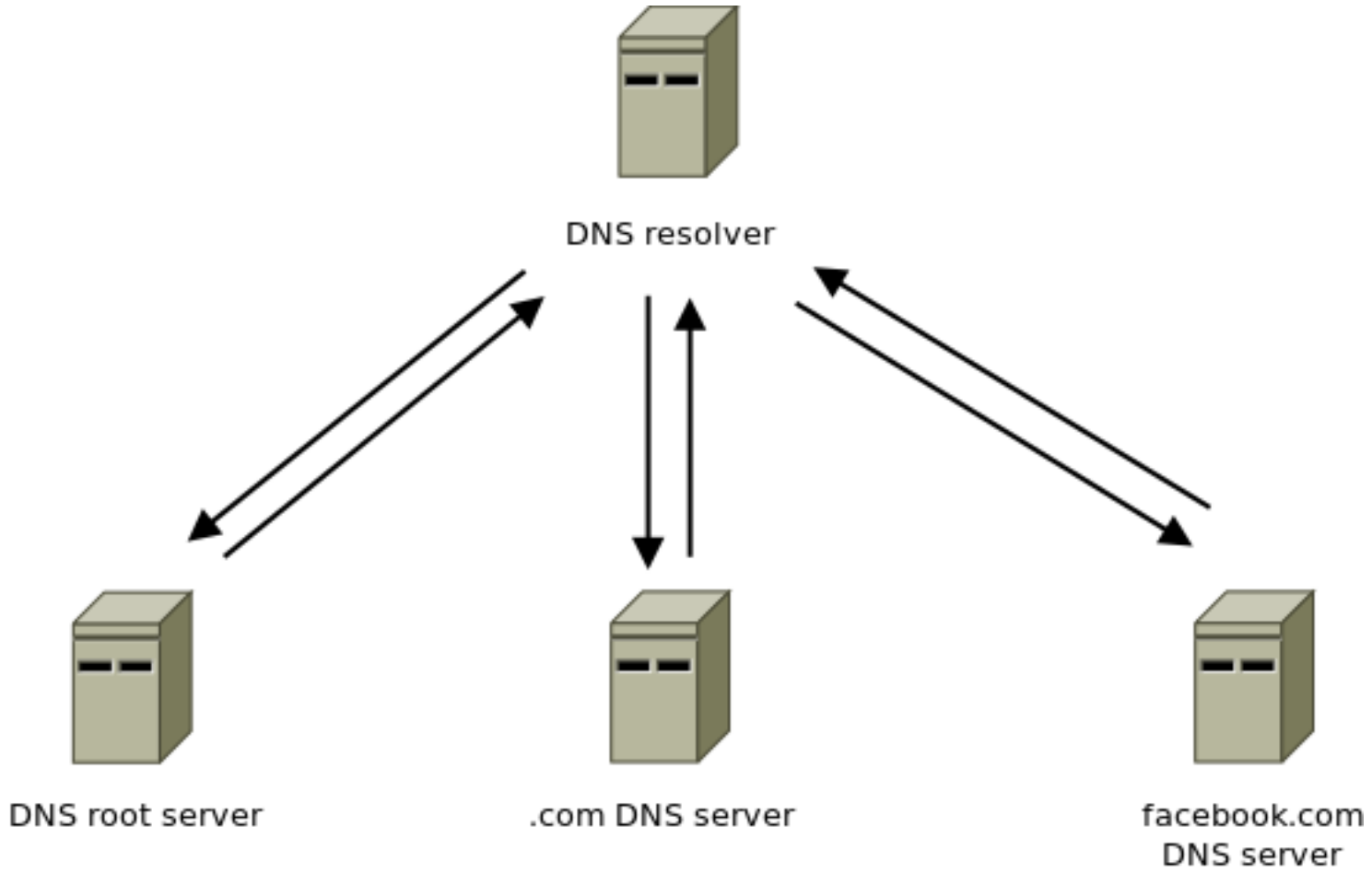
How DNS works

- Decentralized architecture
- Subdomains are used as delimiter

1st way of resolving in a decentralized network



2nd way of resolving in a decentralized network



Outline

1973—1985

- December 1973
  - HOSTS.TXT (RFC 606)
- November 1983
  - DNS invented (RFC 882)
- October 1984
  - TLDs defined (RFC 920)

RFC 920, section Initial Set of Top Level Domains

- .ARPA
- Categories
  - .GOV
  - .EDU
  - .COM
  - .MIL
  - .ORG
- Countries
- Multiorganizations

RFC 920, section Initial Set of Top Level Domains

Countries

The English two letter code (alpha-2) identifying a country according the the ISO Standard for "Codes for the Representation of Names of Countries" [5].

RFC 920, section Initial Set of Top Level Domains

Multiorganizations

A multiorganization may be a top level domain if it is large, and is composed of other organizations; particularly if the multiorganization can not be easily classified into one of the categories and is international in scope.

Winter of 1985

January 1985:

- SRI runs DNS service
  - Nonprofit Research Institute
  - SRI-NIC, in cooperation with IANA
- .NET (forgotten in RFC 920)

Summer of 1985

July 1985

- ccTLDs established
  - .US (February 15, 1985)
  - .UK, .GB (July 24, 1985)
  - .AU (March 5, 1986)
  - .NL (April 25, 1986)
  - .JP (August 5, 1986)

1987—1994

- November 1987
  - DNS Specification
  - STD 13 (IETF standard), RFC 1034, RFC 1035
- April 1993
  - InterNIC starts, initiated by NSF (National Science Foundation) and operated by NSI (Network Solutions Inc.) and AT&T
  - InterNIC managed the allocations of addresses

1994—1996

- June 1994
  - Commercial use becomes dominant
- September 1995
  - Charging for domain name registration starts

Summer of 1997

1997 — Start planning for competition

On July 1, 1997, as part of the Administration's Framework for Global Electronic Commerce, the President directed the Secretary of Commerce to privatize the management of the domain name system (DNS) in a manner that increases competition and facilitates international participation in its management.

Source: MoU (Memorandum of Understanding; November 1998)[1]

[1] Also see RFC 2860 (June 2000)

November 1998

- November 1998
  - Start of ICANN
    - Internet Corporation for Assigned Numbers and Names
- Responsibilities
  - IP address assignment, via ASO
    - Address Supporting Organization
  - Internet domain names, via GNSO and ccNSO
    - Generic Names Supporting Organization
    - Country Code Names Supporting Organization
  - Protocol parameters and port numbers, supported by IANA
    - Internet Assigned Numbers Authority

Outline

DNS concepts

- Domain Name Space (Domain Name Tree)
- Resource Records
- Name Servers
- Resolvers

Domain names

- Nodes (internal and leaf) have a label
  - root label is empty: “” (not “ ”)
  - non-root labels must be non-empty
  - labels are 0-63 octets long
  - the root label is the only one with length 0
- A domain name is a sequence of labels separated by “.” (dot)
  - specifying the complete path to the root
  - and ending in the (empty) root label
- A domain is a domain name
  - together with all domain names below it

To slash or not to slash

- Compare domain names to pathnames in a filesystem
  - Labels (filenames)
    - separated by "/" (slash).
  - Absolute versus relative pathnames

To dot or not to dot

- Absolute domain (FQDN)
  - mail.serv.os3.nl.
- Relative domain
  - mail
  - mail.serv
- machine.cs can (could?) give problems
  - Why?

Resource Records (RR's)

- owner (domain name)
- ttl (time to live (in cache))
- class (IN, CH, HS)
  - Only IN actively used
- type (A, AAAA, CNAME, DNAME, MX, NS, PTR, SOA, SRV, ...)
- resource data (depends on type)
- BIND syntax
  - `owner [ttl] [class] type data`
  - ttl and class are optional and default to $TTL and IN

"A" record

- An A record (address record) translates

  a domain name to an IPv4 address

  - mail.serv.os3.nl. $\longrightarrow$ 145.100.96.25
- Multihomed hosts have several A records
  - Routers may have multiple A records
- Example (assuming the $ORIGIN is `os3.nl.`)
  - `mail.serv IN A 145.100.96.25`

Example of multiple A records

router.studlab.os3.nl.	A	145.100.104.1
router.studlab.os3.nl.	A	145.100.104.33
router.studlab.os3.nl.	A	145.100.104.65
router.studlab.os3.nl.	A	145.100.104.97
router.studlab.os3.nl.	A	145.100.104.129
router.studlab.os3.nl.	A	145.100.104.145
router.studlab.os3.nl.	A	145.100.104.161
router.studlab.os3.nl.	A	145.100.104.193
129.104.100.145.in-addr.arpa.	PTR	router.studlab.os3.nl.

"AAAA" record

- AAAA are sometimes called quad-A records
- A quad-A record translates a domain name to an IPv6 address
  - mail.serv.os3.nl. $\longrightarrow$ 2001:610:158:960::25
- Many hosts have multiple AAAA records
  - It is quite normal in IPv6 to belong to multiple subnets
- Example (assuming the $ORIGIN is `os3.nl.`)
  - `mail.serv IN AAAA 2001:610:158:960::25`

"CNAME" record

- A CNAME (canonical name) record defines an alias
  - www.os3.nl. $\longrightarrow$ info4u.os3.nl.
  - No other RR's are allowed
  - Does not work for subdomains
    - DNAME record proposed for that
- Example (assuming the $ORIGIN is `os3.nl.`)
  - `www IN CNAME info4u`

"DNAME" record

- A DNAME is used for non-terminal DNS Name Redirection
- Allows other RR types at the same owner

  except CNAME, DNAME
- Synthesizes CNAME records for clients who do not ascertain

  their understanding of DNAME semantics
- Processed before wildcards
- Also called "Delegation Name" because of its use

  instead of NS records in certain cases
- Example (assuming the $ORIGIN is `nl.`)
  - `ruu IN DNAME uu`

"MX" record

- MX (Mail eXchanger) record defines for a domain
  - mail servers for that domain
  - and the order of their preference
  - lower precedence is preferred
- MX must not point to a CNAME
- Example (assuming the $ORIGIN is `os3.nl.`)
  - `@ IN MX 10 smtp1`
  - `@ IN MX 10 smtp2`
  - `@ IN MX 20 backup.example.nl.`
  - `@ IN MX 30 backup.example.be.`

"NS" record

- NS (Name Server) record defines a cut (zone)
  - Must list at least two name servers
  - Makes DNS decentralized
  - Delegates responsibility
- NS record must not point to a CNAME
- Example (assuming the $ORIGIN is `os3.nl.`)
  - `@ IN NS ns1`

"PTR" record

- A PTR (pointer) record literally points

  to an arbitrary point in the DNS tree
- Mostly used for "reverse" lookup
  - 145.100.96.25 $\longrightarrow$ mail.serv.os3.nl.
  - But lookup works via in-addr.arpa
    - 25.96.100.145.in-addr.arpa.
- Example (assuming the $ORIGIN is `os3.nl.`)
  - `25.96.100.145.in-addr.arpa.  IN PTR mail.serv`

"SOA" record

- An SOA (Start Of Authority) record administrates important zone parameters
  - hostname of master server
    - ns1.os3.nl.
  - email address (in "dot" form) of responsible person
    - hostmaster.os3.nl. $\longrightarrow$ hostmaster@os3.nl
  - numerical parameters

Numerical SOA params (recommended values)

Time values are given in seconds

The SOA record itself can have a low TTL (for instance 3600 = 1 hour)

- Serial (YYYYMMDDnn)
- Refresh (86400 = 1 day)
- Retry (7200 = 2 hours)
- Expire (3600000 = 1000 hours $\sim$ 40 days)
- "Minimum" (172800 = 2 days, but...)

Numerical SOA params (OS3 example)

These values are quite low (during IP migration)

- Serial (2007110900)
- Refresh (3600 = 1 hour)
- Retry (1800 = 30 minutes)
- Expire (21600 = 6 hours)
- "Minimum" (3600 = 1 hour, but...)

SOA example

```
os3.nl.    IN    SOA    ns1.os3.nl.    hostmaster.os3.nl. (
           2010071401 ; serial (version)
                3600 ; refresh period (1 hour)
                1800 ; retry interval (30 minutes)
               21600 ; expire time (6 hours)
                3600 ; "minimum" (1 hour)
           )
```

“Minimum”

- Different interpretations
  - Minimal TTL allowed (never used this way)
  - Default TTL, if TTL not specified (BIND 8)
  - TTL for caching negative replies (BIND 9)
- BIND 9 uses global $TTL for default TTL

"SRV" record

- A SRV (service) record specifies the location

  of the services that a domain supports
- The format for the information about a certain domain "Name" uses
  - "_Service._Proto.Name" as the owner domain name
  - "Priority Weight Port Target" as its resource data
- It is a typical generator of empty non-terminals
  - Like "_Proto.Name" in the above case
- Example
  - _sip._tcp.example.com. IN SRV 10 20 5060 sip.example.com.

Resource Record sets (RRsets)

- An RRset is a grouping of a set of RR's

  with the same owner, class and type
- All RR's in an RRset must have the same TTL
- DNSSEC signs complete RRsets with RRSIG RR's
  - Which might make the RRSIG RR an exception to the TTL rule :)
  - But in fact the DNSSEC specification tells us they do not form a resource record set at all (RFC 4035, section 2.2)

Outline

Name servers and zones

- Zones are created by cuts (delegations)
- Cuts are defined by NS records
  - "inside" parent zone
  - non-authoritative by definition
- Glue A records sometimes needed
  - When name servers for the delegation are "in bailiwick"
  - Or in the more general case when name servers have circular dependencies and create "bailiwick loops"

Bootstrap issues

- Hint file for root RR's
- Glue for child zones
  - Glue NS records (stub server)
  - Glue A records (for servers inside the child zone)
- Glue data is not authoritative unless the parent is also a slave server
- Non-authoritative data should be replaced by authoritative data as soon as that becomes available

Name server types

- Master (primary)
- Slave (secondary)
- Stub (limited secondary)
- Stealth (secondary that is not listed)
- Caching-only (never authoritative)
- Forward-only (using “forwarders”)

Outline

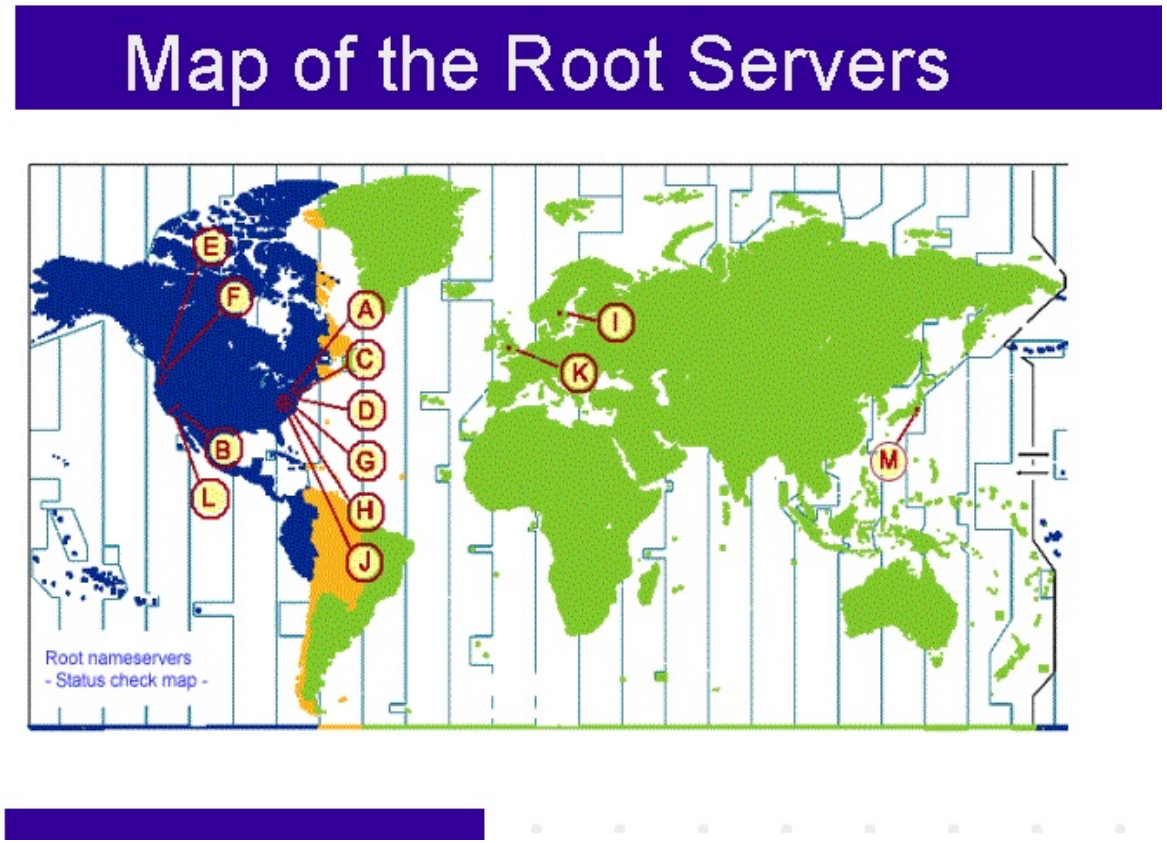
DNS structure

- Hierarchical tree
  - its root is “unnamed” (“unlabeled”)
    - in fact the root uses the “empty label”: “”
  - Top Level Domains (TLDs)
    - generic TLDs (gTLDs)
    - country code TLDs (ccTLDs)
- Decentralized database

Root servers

- Status in 2001, according to ICANN official Michael Roberts
  - 13 root servers
  - Most of them located in the US (10)
- Nowadays there is a complete infrastructure with both global and local servers

Root servers map



Map provided by ICANN

Root server list (part 1)

Name	Org	Where	Globals	Locals
A	Verisign	Los Angeles, CA, US	6	0
B	USC-ISI	Marina del Rey, CA, US	0	1
C	Cogent Communications	Herndon, VA, US	6	0
D	University of Maryland	College Park, MD, US	1	0
E	NASA (Ames)	Mountain View, CA, US	1	0
F	ISC (Internet Software Consortium)	Palo Alto, CA, US	2	47
G	US DOD NIC	Columbus, OH, US	6	0

Root server list (part 2)

Name	Org	Where	Globals	Locals
H	US Army Research Lab (ARL)	Aberdeen, MD, US	2	0
I	Autonomica (NORDUnet)	Stockholm, SE	38	0
J	Verisign	Dulles, VA, US	64	6
K	RIPE NCC	London, UK	5	13
L	ICANN	Los Angeles, CA, US	50	0
M	WIDE	Tokyo, JP	5	1

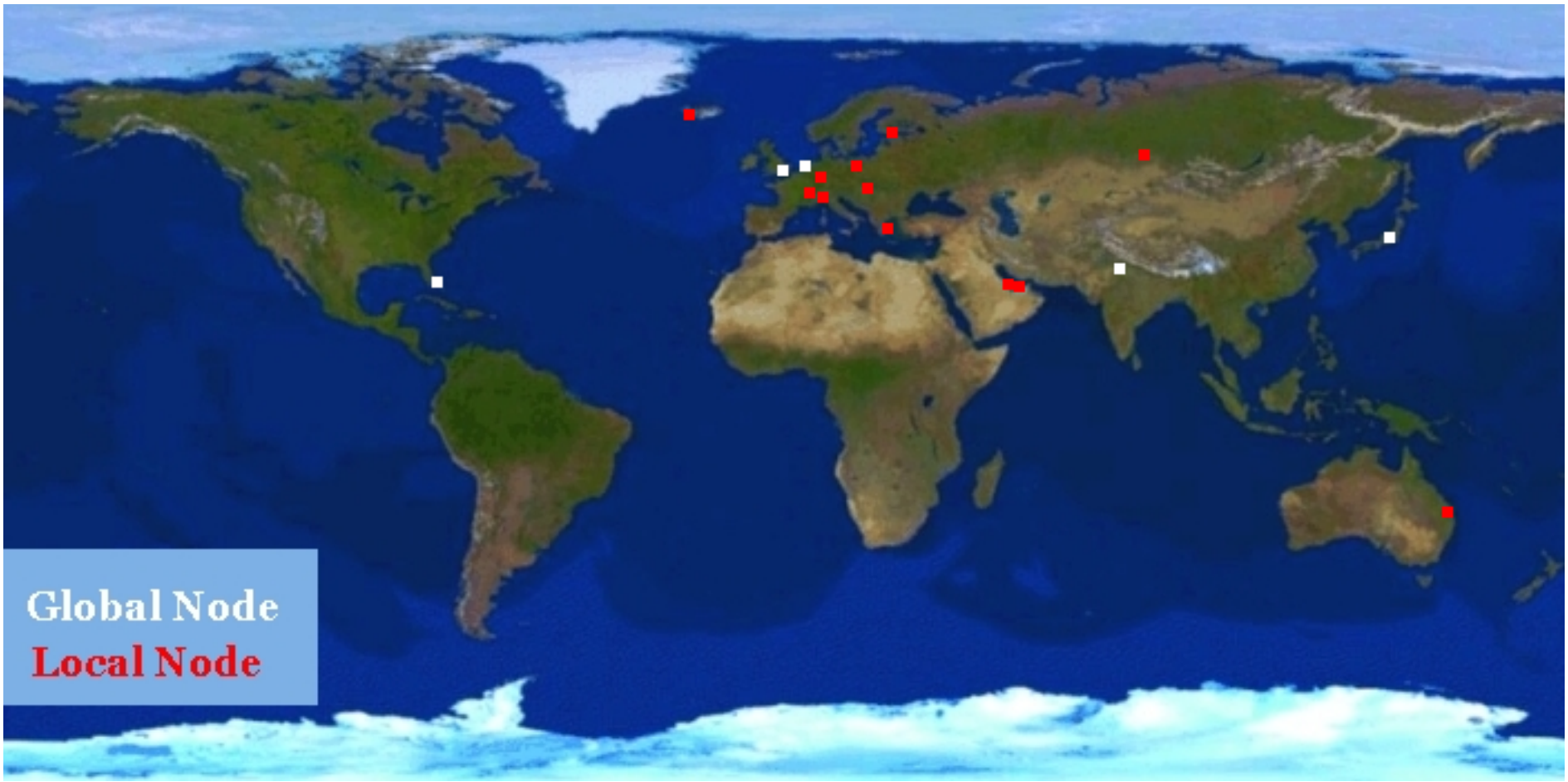
Source: `http://www.root-servers.org/` (retrieved 20110912)

Anycast

- Overloading of IP address
- Route to nearest instance (BGP metric)
- Global or local significance
- Live data for k root can be found at

  `http://k.root-servers.org/`

k root server presence (2012 snapshot)



Source: `http://k.root-servers.org/pics/map.png` RIPE NCC

Anycasted root servers map (2012 snapshot)

Source: `http://www.root-servers.org/`

Outline

Recursion and iteration

- Recursive behaviour
  - Server follows referrals itself and
  - Often doesn't have authoritative data at all (almost)
  - Usually builds up a cache
- Iterative behaviour
  - Server answers with authoritative data or
  - Server passes referrals back to clients
  - Often only has authoritative data and no cache

Resolver

- Library doing domain name lookup
  - Uses `/etc/resolv.conf`
  - Contacts a recursive nameserver
  - Does not follow referrals itself

Caching

- Necessary for performance
- Negative caching adds more functionality
  - See RFC 2308
  - Lots of subtleties

Common mistakes

- See RFC 1912 and also RFCs 2181 and 4697
  - Using CNAME's in MX and NS records
  - Forgetting the final "."
  - Lame delegation
  - Lack of human coordination