DNS101

What’s the DNS and How It Works

Yazid Akanho

AFRALO Capacity Building Webinar
May 2021

ICANN

Agenda

- Once upon a time
- Rise of the DNS
- DNS Database and Data
- Resolution process
- Caching
- DNS Resilience

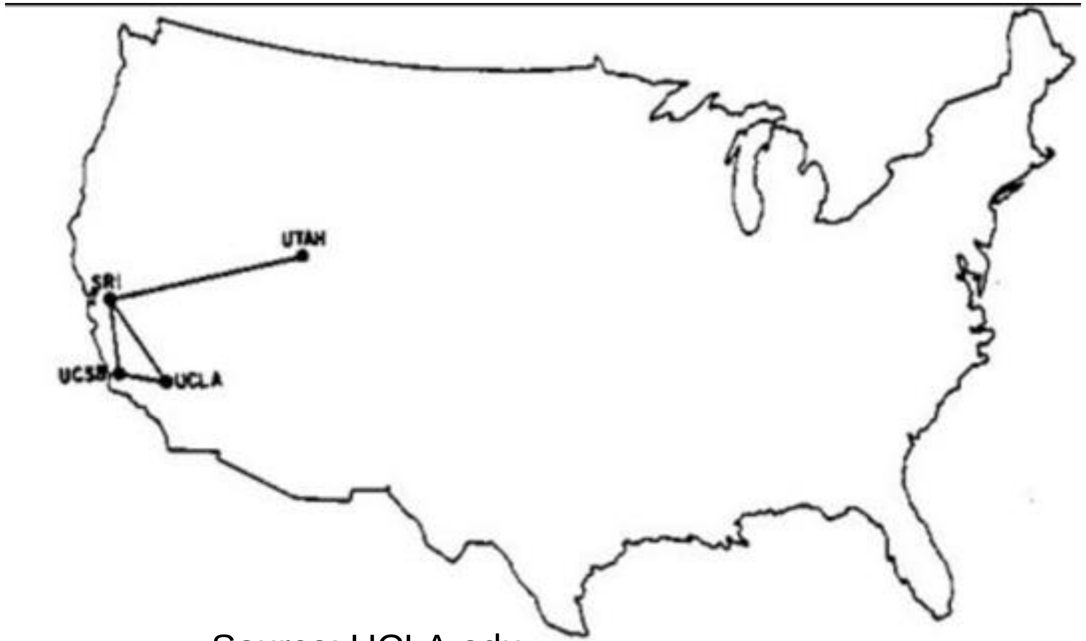
Once upon a time…

Discussion: how did Internet started and what is it today ?

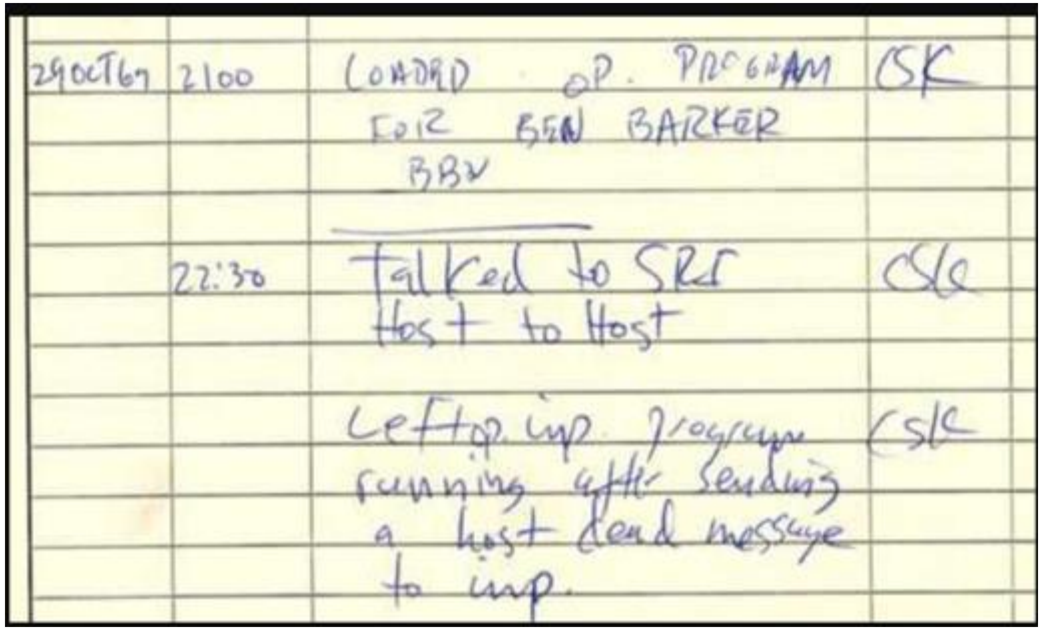
Questions &
Feedback

The Network of Networks

- 1969 - ARPANET is Born on October 29th – 4 Participating Institutions:
  - University of California, Los Angeles (UCLA)
  - Stanford Research Institute (SRI)
  - University of California, Santa Barbara
  - University of Utah



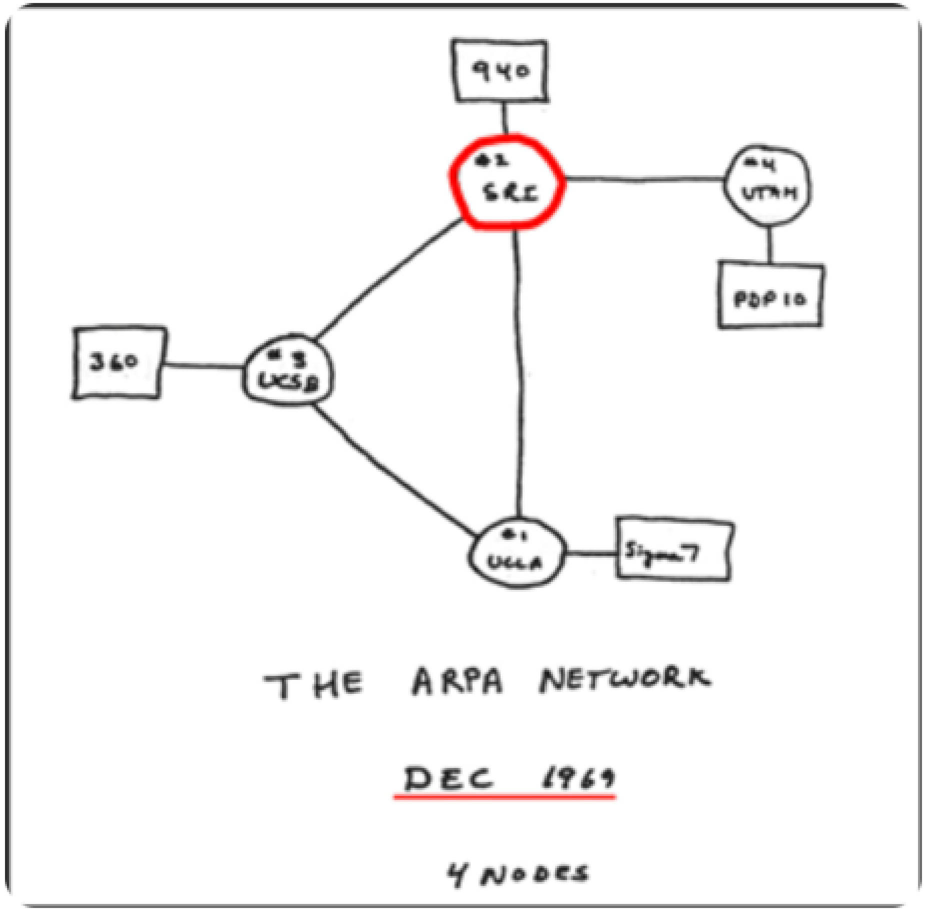
Source: UCLA.edu

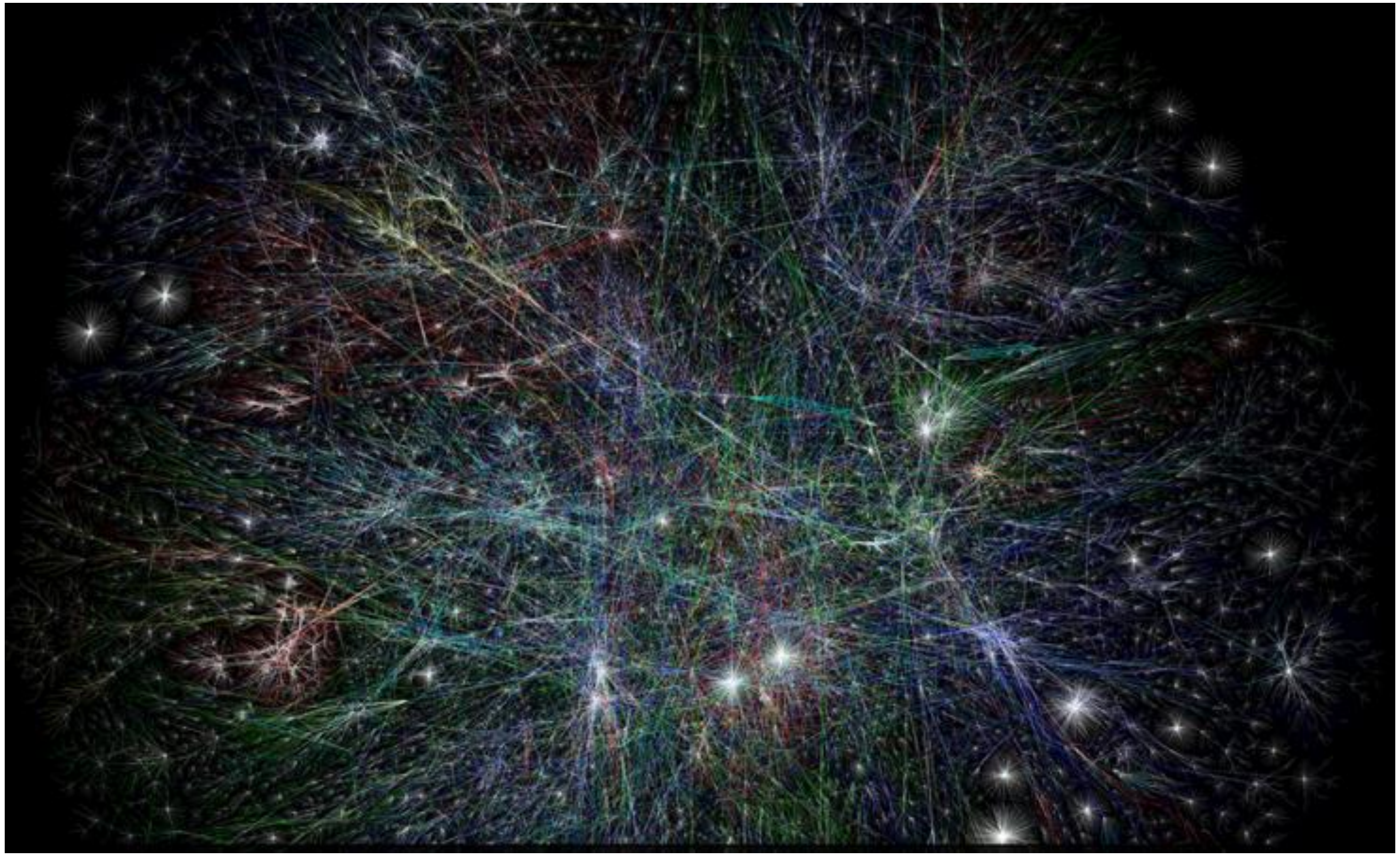


29OCT69	2100	LOADED OP. PROGRAM FOR BEN BARKER BBN	CSK
	22:30	Talked to SRI Host to Host	CSK
		Left op. imp. program running after sending a host dead message to imp.	CSK

Source: edn.com

The Network of Networks





Source: sri.com

Source: Kaspersky.com

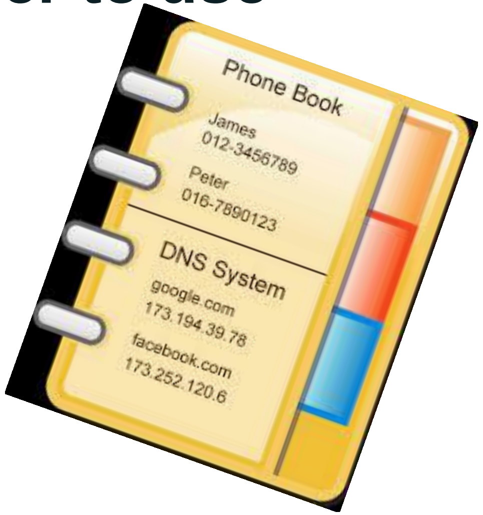
The Network of Networks: +100.000 networks; plenty of services

Names and Numbers

- Devices are identified over the Internet using IP addresses.
  - IPv4: 192.0.2.7
  - IPv6: 2001:db8::7

- While **IP addresses are easy for machines** to use, **people prefer to use names**.

- In the early days of the Internet, names were simple
  - No domain names yet
  - “Single-label names”, 24 characters maximum
  - Referred to as ***host names***



Name Resolution

- Mapping names to IP addresses (and IP addresses to names) is ***name resolution***
- Name resolution on the early Internet used a plain text ***file*** named HOSTS.TXT
  - Same function but slightly different format than the former */etc/hosts*
  - Centrally maintained by the NIC (Network Information Center) at the Stanford Research Institute (SRI)
  - Network administrators sent updates via email
- Ideally everyone had the latest version of the file
  - Released once per week
  - Downloadable via FTP

Problems with HOSTS.TXT

- Naming contention
  - Edits made by hand to a text file (no database)
  - No good method to prevent duplicates

- Synchronization
  - No one ever had the same version of the file

- Traffic and load
  - Significant bandwidth required then just to download the file

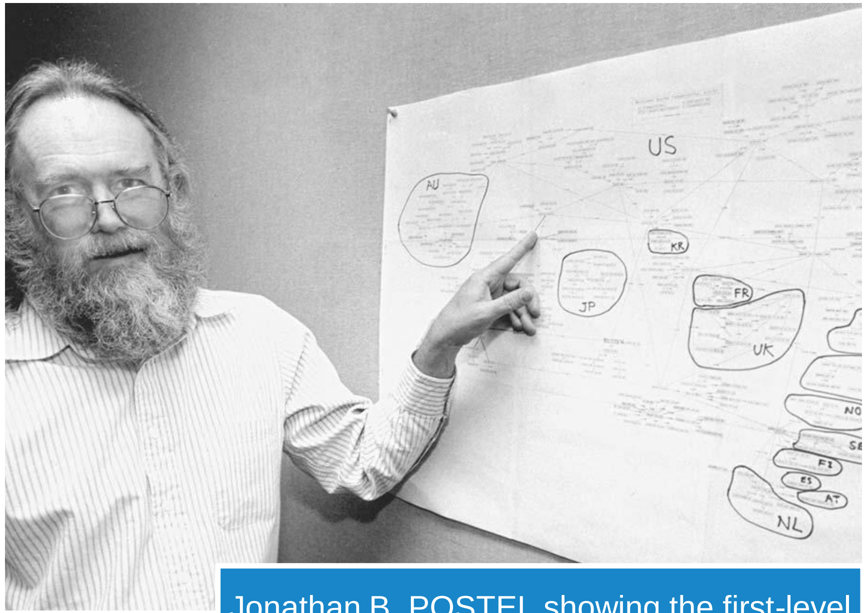
- **A centrally maintained host file just didn’t scale**

DNS to the Rescue

- Discussion started in the early 1980s on a replacement
- Goals:
  - Address HOST.TXT scaling issues
  - Simplify email routing
- Result was the ***Domain Name System***
- Requirements in multiple documents:
  - RFC 799, “Internet Name Domains”
  - RFC 819, “The Domain Naming Convention for Internet User Applications”
  - Most referred to: RFC 1034 and RFC 1035

Paul MOKAPETRIS & John POSTEL: inventors of DNS

Paul MOKAPETRIS

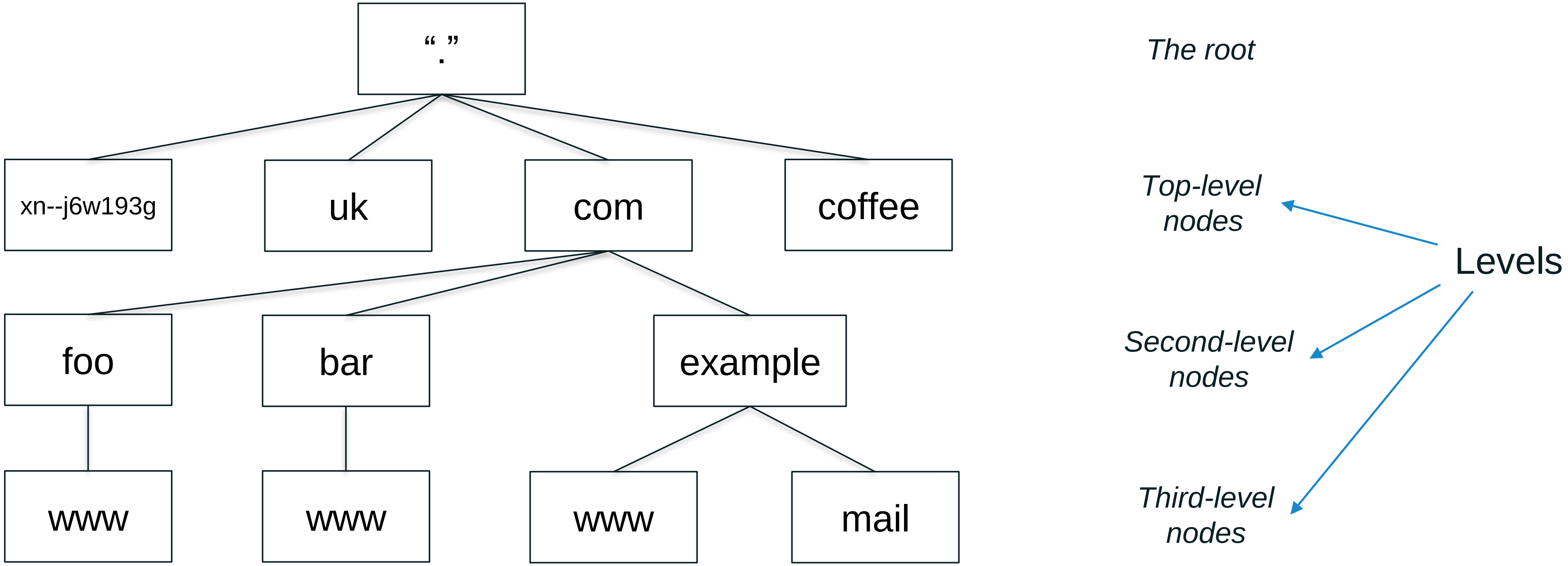


Jonathan B. POSTEL showing the first-level domains on a map in 1994

Rise of the DNS !

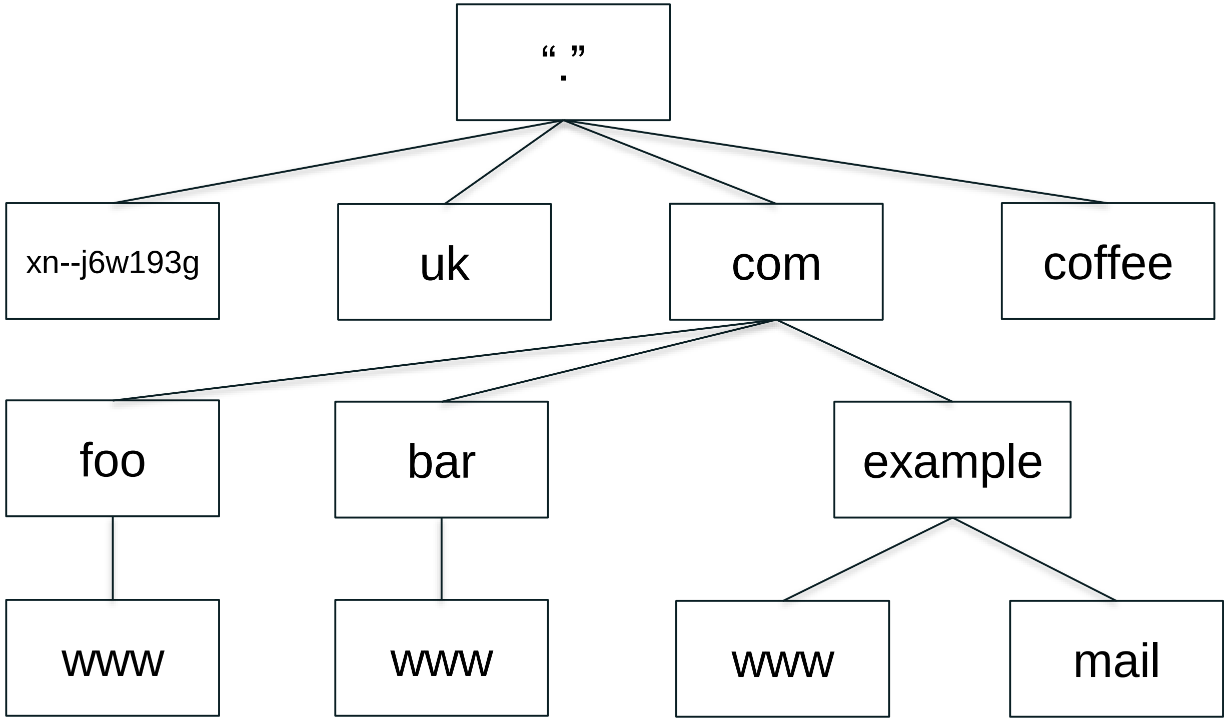
The Name Space

- DNS database structure is an **inverted tree** called the ***name space***
- Each node has a label
- The root node (and only the root node) has a null label



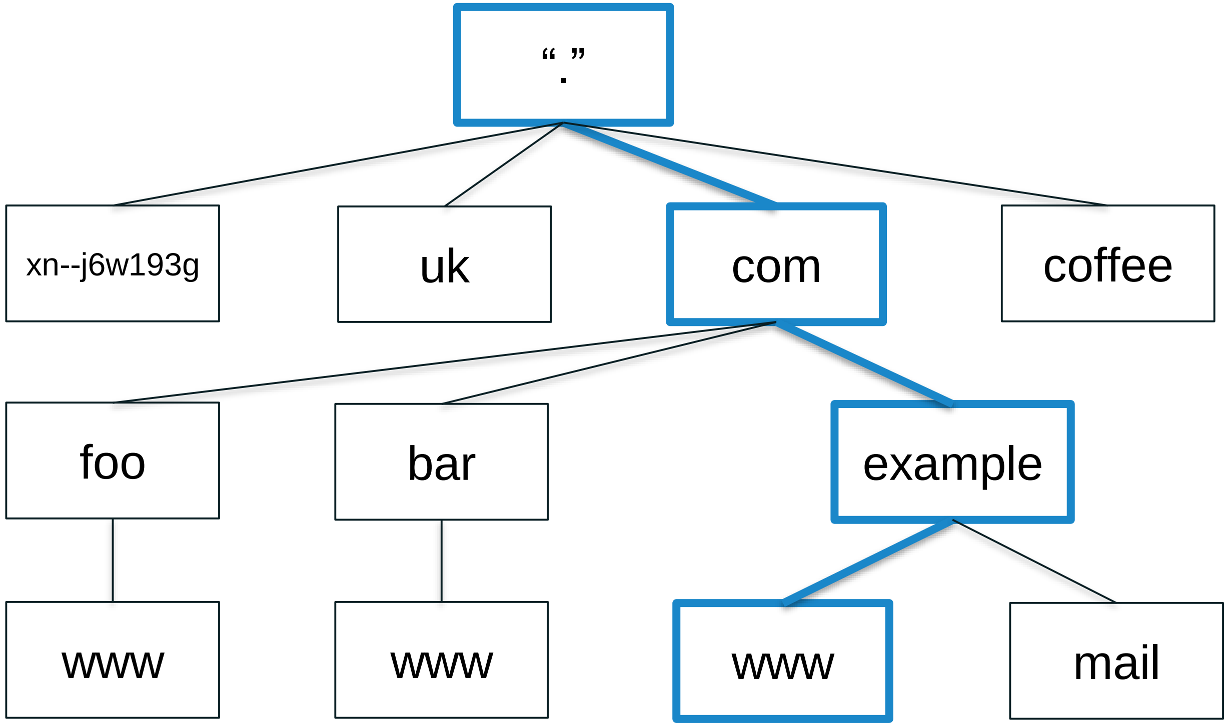
Label Syntax (before IDN)

- **Legal characters** for labels are “LDH” (letters, digits, hyphen)
- Maximum length 63 characters
- Comparisons of label names are not case sensitive



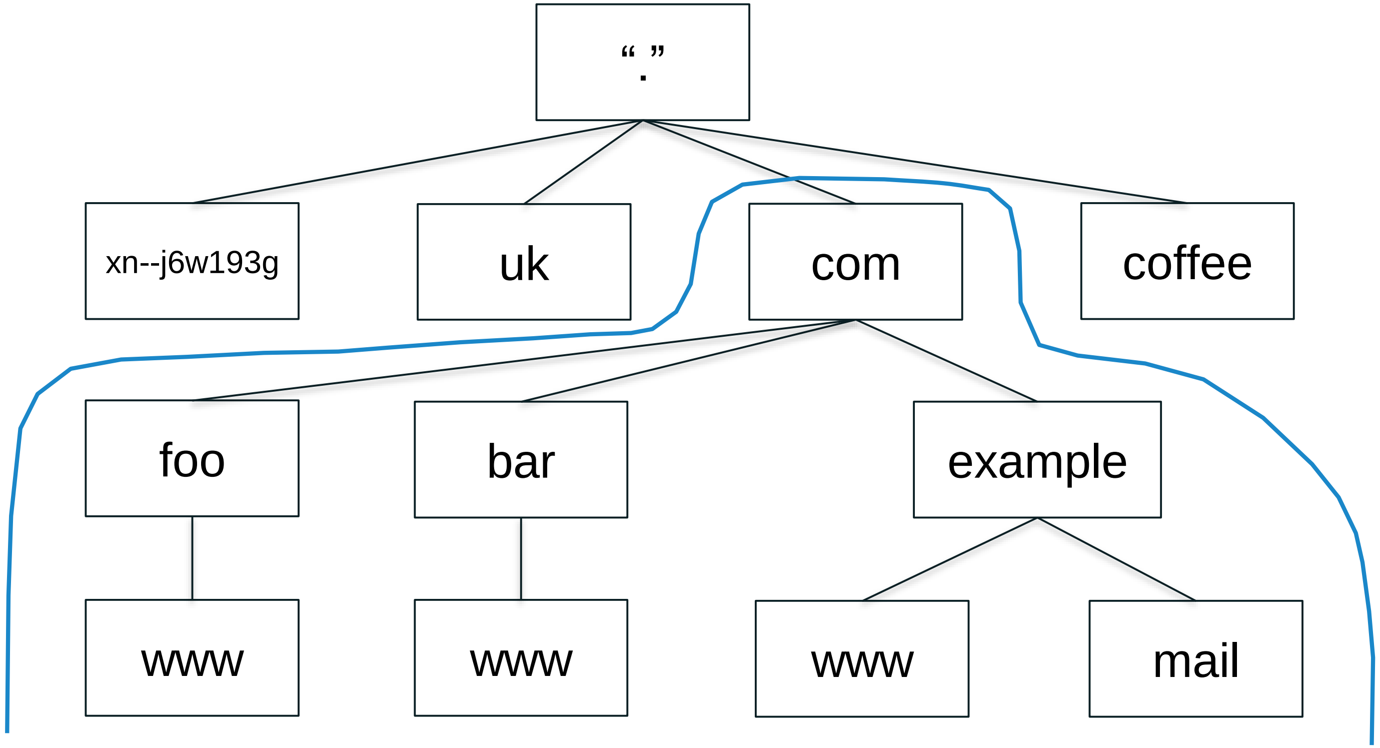
Domain Names

- Every node has a ***domain name***
- That ***domain name*** is built by sequencing node labels from one specified node up to the root, separated by dots.
- Highlighted: *www.example.com**.***



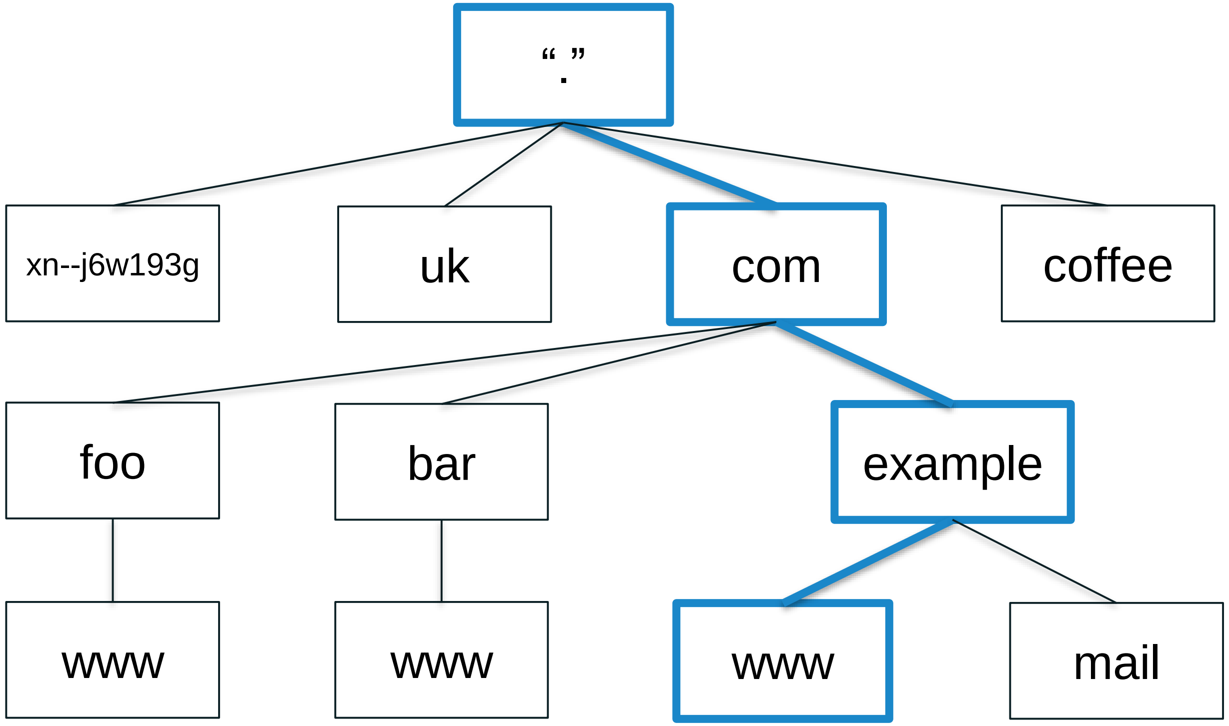
Domains

- A ***domain*** is a node and everything below it.
- The top node of a domain is the ***apex*** of that domain.
- Shown: the *com* domain.



Fully Qualified Domain Names

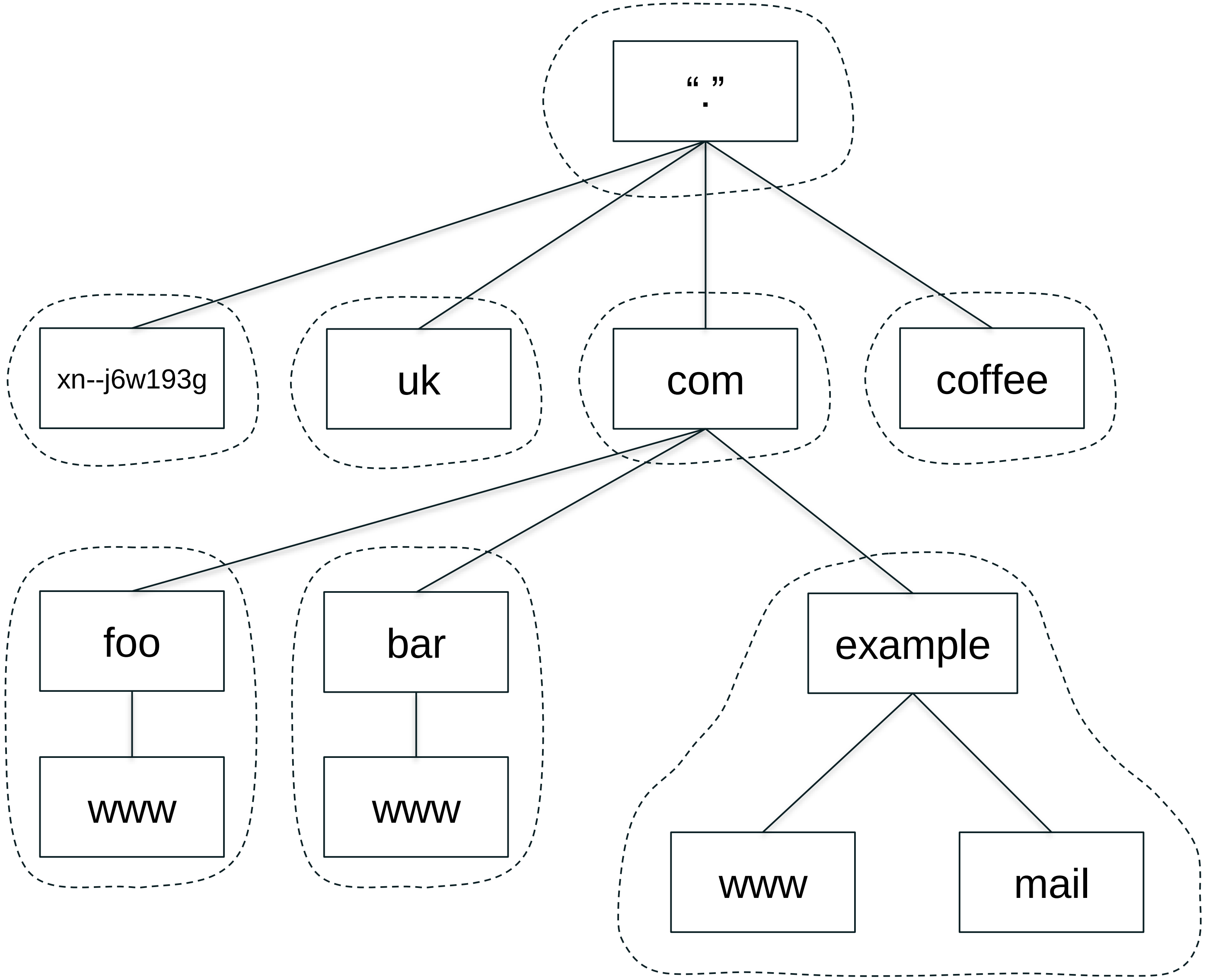
- A ***fully qualified domain name (FQDN)*** unambiguously identifies a node
  - Not relative to any other domain name
- An FQDN **ends in a dot**
- Example FQDN: *www.example.com.*



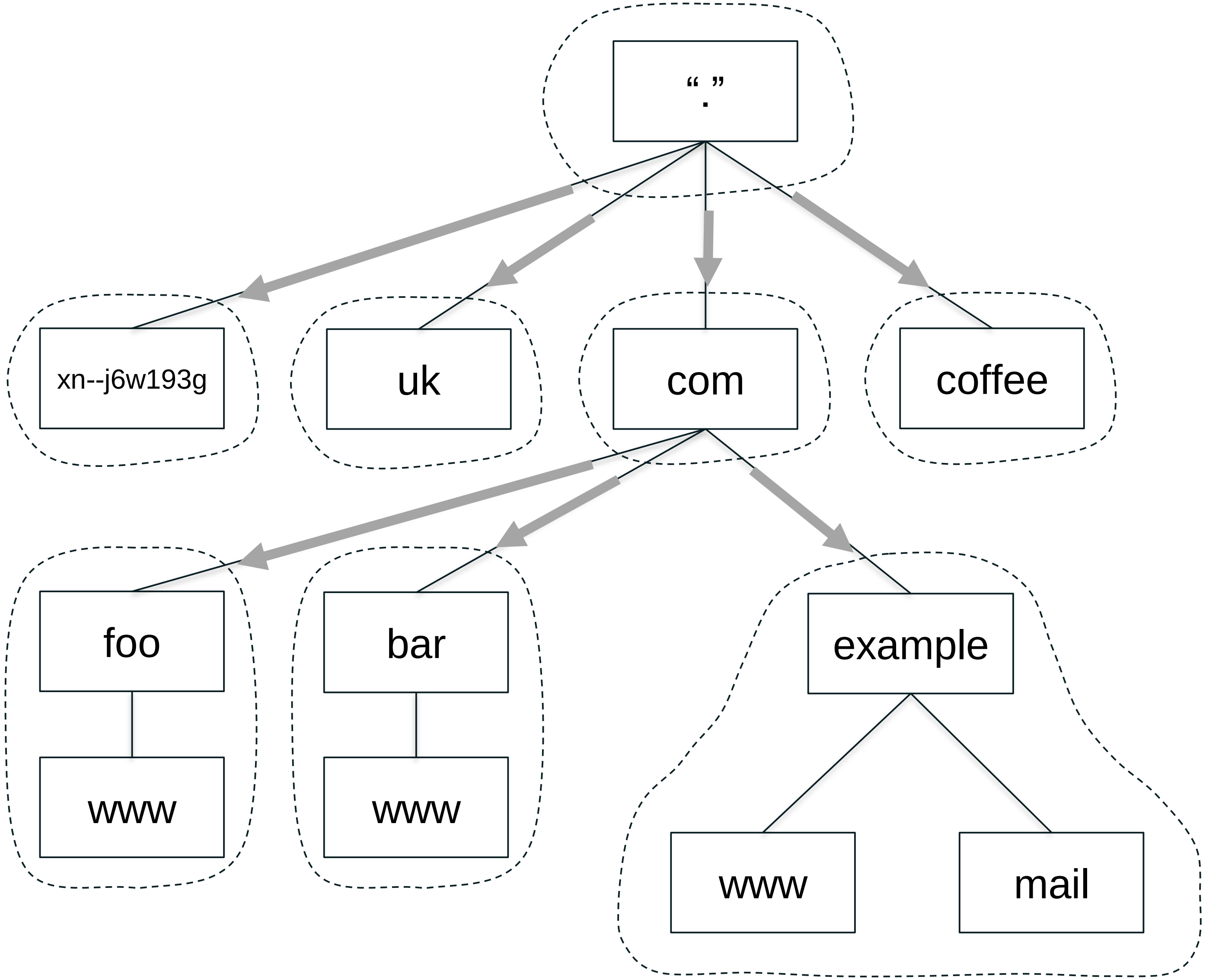
Zones

- The name space is divided up to allow distributed administration
- Administrative divisions are called ***zones***
- An administrator of any zone may delegate the administration of a subtree of its zone, thus creating a new zone
- ***Delegation*** creates zones
  - Delegating zone is the ***parent***
  - Created zone is the ***child***

Zones are Administrative Boundaries

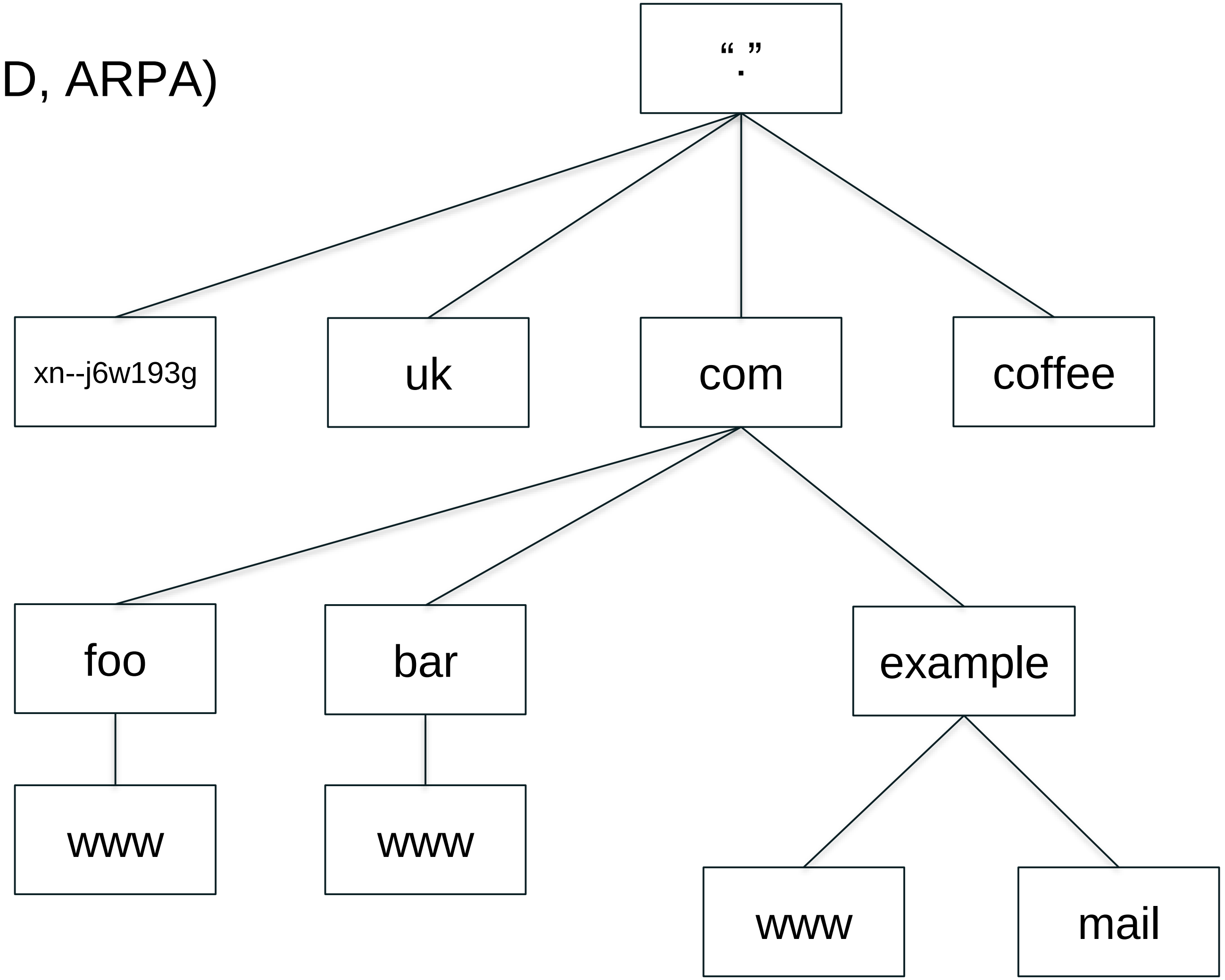


Delegation Creates Zones



The Name Space today

- +1500 TLD (gTLD, ccTLD, IDN TLD, ARPA)
- The ”root” in its own is a “system”.



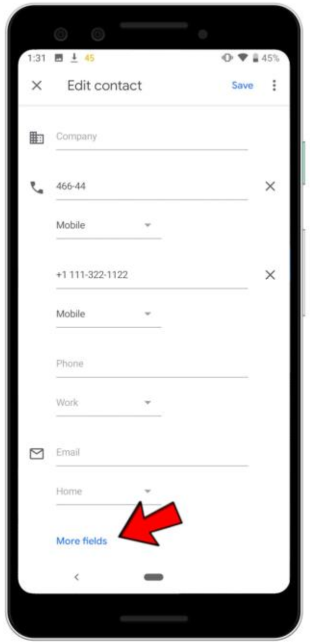
DNS Database and Data

DNS Data

- The DNS standard specifies the format of DNS data sent over the network
    - Informally called “wire format”
- The standard also specifies a text-based representation for DNS data called ***master file format,*** used for storing the data (much like tables in a database)
- A ***zone file*** contains all the data for a zone in master file format

DNS Resource Records

- Recall every node has a domain name
- A domain name can have different kinds of data associated with it
- That data is stored in ***resource records*** (this are the records in DNS database)
  - Sometimes abbreviated as ***RRs***
- Different record types for different kinds of data



1:31 45 45%

Edit contact Save

Company

466-44

Mobile

+1 111-322-1122

Mobile

Phone

Work

Email

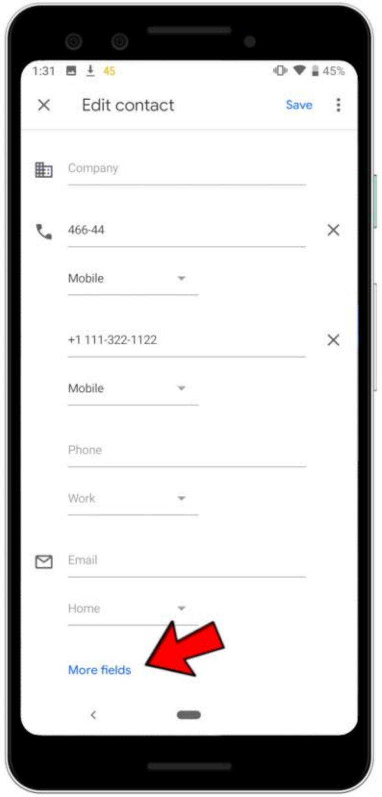
Home

More fields

Discussion: What type of Resource Record do you know ?

Common Resource Record Types

A	IPv4 address
AAAA	IPv6 address
NS	Name of an authoritative name server
SOA	“Start of authority”, appears at zone apex
CNAME	Name of an alias to another domain name
MX	Name of a “mail exchange server”
PTR	IP address encoded as a domain name (for reverse mapping)


Edit contact

Company

466-44

Mobile

+1 111-322-1122

Mobile

Phone

Work

Email

Home

Sample Zone File: *example.com*

```
example.com.      SOA   ns1.example.com. hostmaster.example.com. (
                         20200316155500 ; serial
                         86400          ; refresh (1 hour)
                         7200           ; retry (2 hour)
                         2592000        ; expire (4 weeks 2 days)
                         172800 )       ; minimum (2 days)
example.com.      NS    ns1.example.com.
example.com.      NS    ns2.example.com.
example.com.      NS    ns1.p41.dynect.net.
example.com.      NS    ns1.p41.dynect.net.
example.com.      NS    ns1.p41.dynect.net.
example.com.      NS    ns1.p41.dynect.net.
example.com.      NS    a1.verisigndns.com.
example.com.      NS    a2.verisigndns.com.
example.com.      NS    a3.verisigndns.com.
example.com.      A     192.0.2.7
example.com.      AAAA  2001:db8::7
example.com.      MX    10 mail.example.com.
example.com.      MX    20 mail-backup.example.com.
www.example.com.  CNAME example.com.
ns1.example.com.  A     192.0.2.1
ns2.example.com.  A     192.0.2.2
```

Discussion: Let’s play and retrieve RRs for some domains!

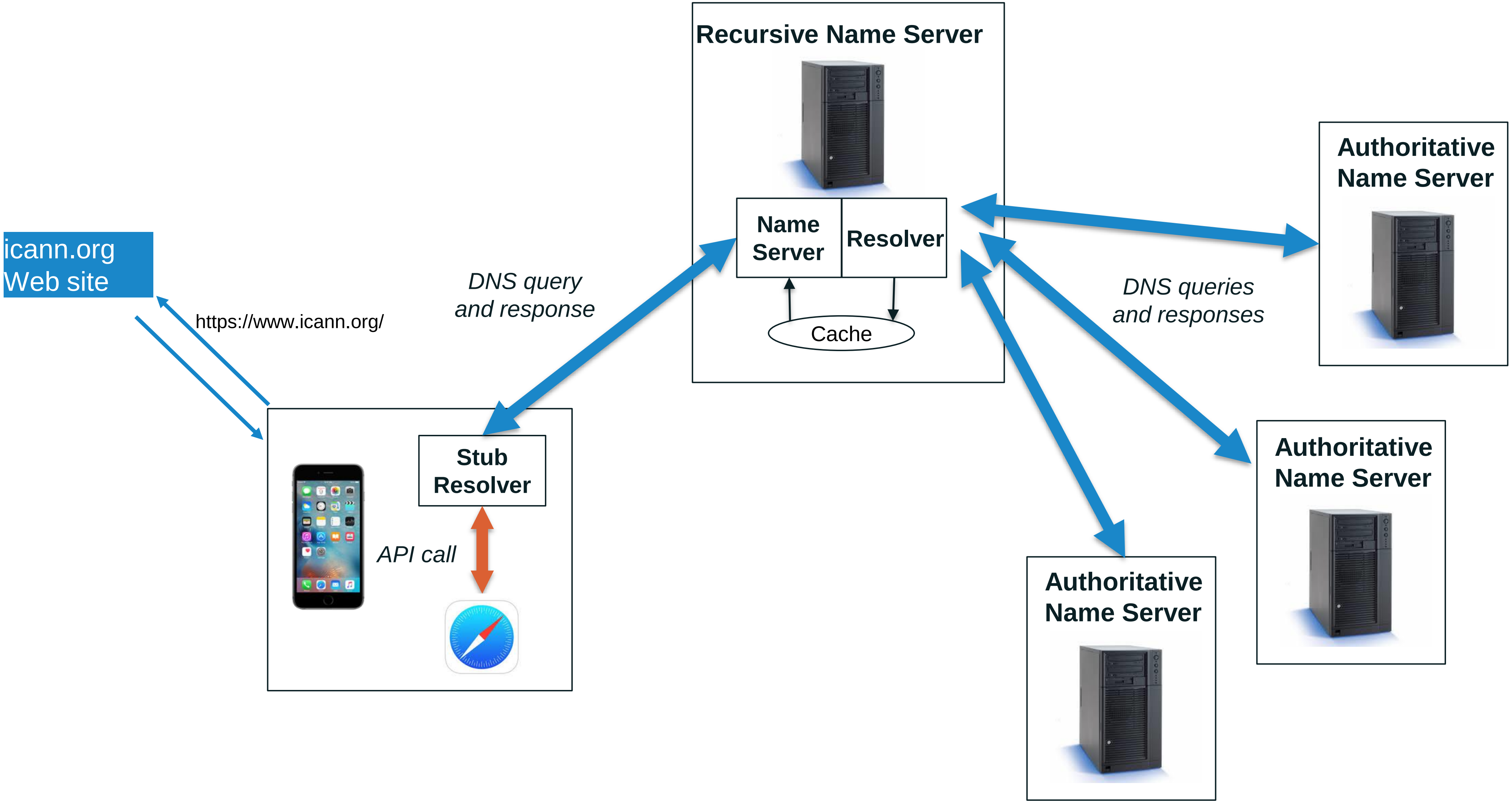
- CLI : dig or nslookup
- Web : https://www.digwebinterface.com/

Resolution Process

DNS in a nutshell

- DNS is a **distributed database**
  - Data is maintained locally but available globally
- ***Resolvers*** send queries
- ***Name servers*** answer queries
- Optimizations:
  - Caching to improve performance
  - Replication to provide redundancy and load distribution

DNS Components at a Glance



Name Servers and Zones

- Name servers answer queries.
- A name server ***authoritative*** for a zone has **complete knowledge of that zone** (remember the zone file!).
  - Can provide a definitive answer to queries about the zone.
- Zones should have multiple authoritative servers.
  - Provides redundancy.
  - Spreads the query load.

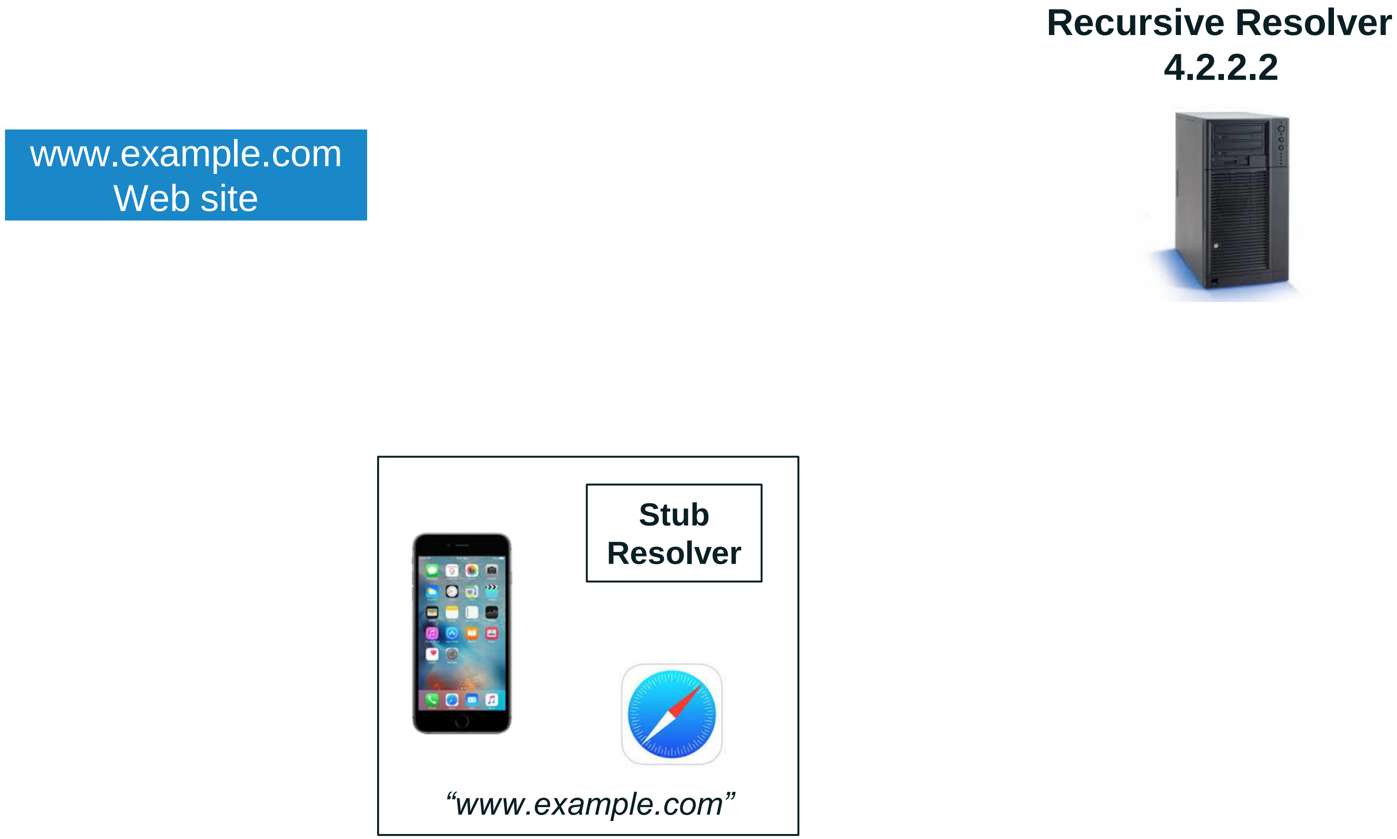
The Resolution Process

- The resolution process is the implementation of translating from a domain name to an IP address , or more general getting the answer for a specific query.

We will go though resolution process step by step…

Resolution Process

A user types *www.example.com* into Safari, which then calls the stub resolver function to resolve the name



Recursive Resolver
4.2.2.2

www.example.com
Web site

Stub Resolver

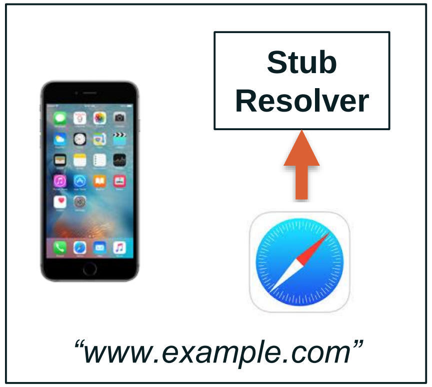
“www.example.com”

Resolution Process

A user types *www.example.com* into Safari, which then calls the stub resolver function to resolve the name

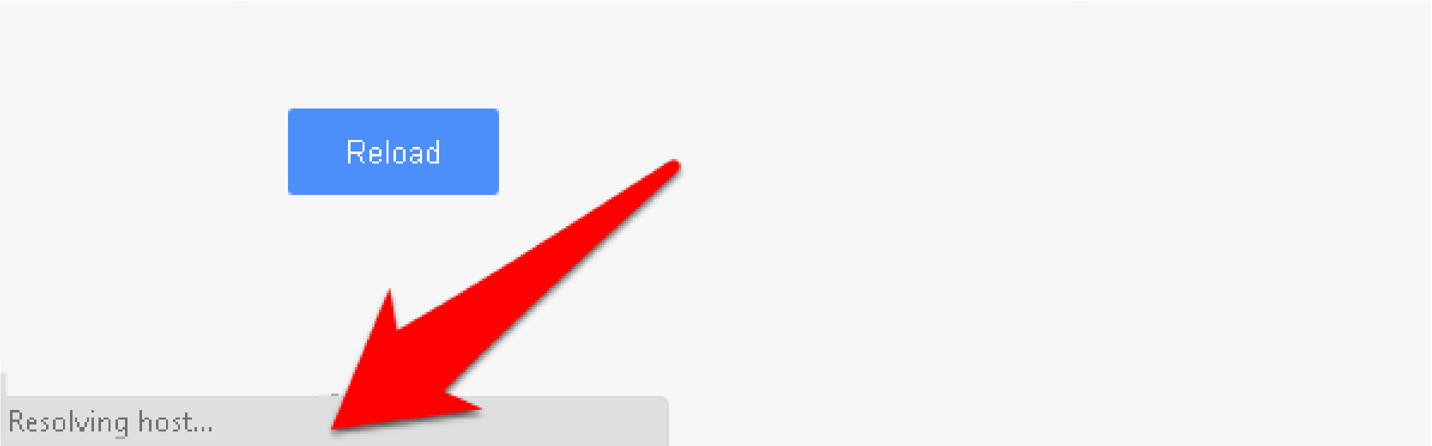
Recursive Resolver
4.2.2.2

www.example.com
Web site



Stub Resolver

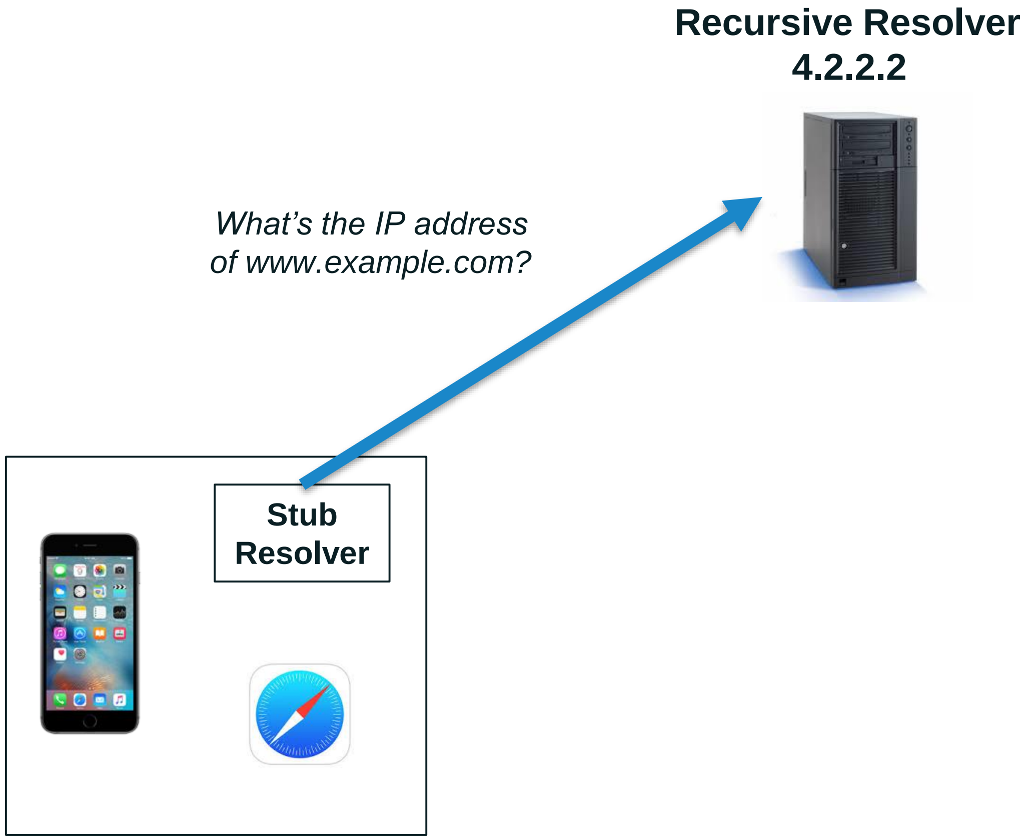
“www.example.com”



Resolving host...

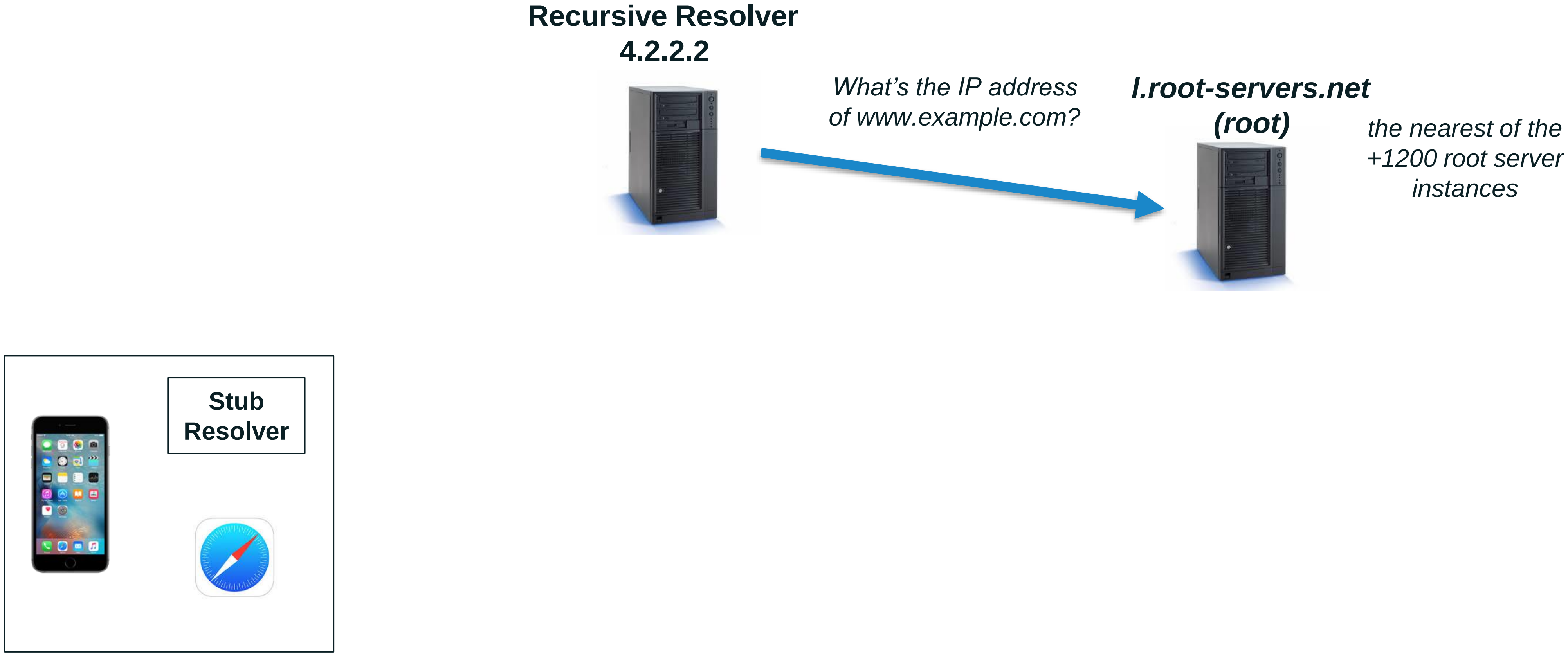
Resolution Process

The phone’s stub resolver sends a query for *www.example.com*, IN, A to 4.2.2.2



Resolution Process

Recursive resolver 4.2.2.2 has no data cached for *www.example.com*, so it queries a root server



Recursive Resolver
4.2.2.2

What's the IP address of www.example.com?

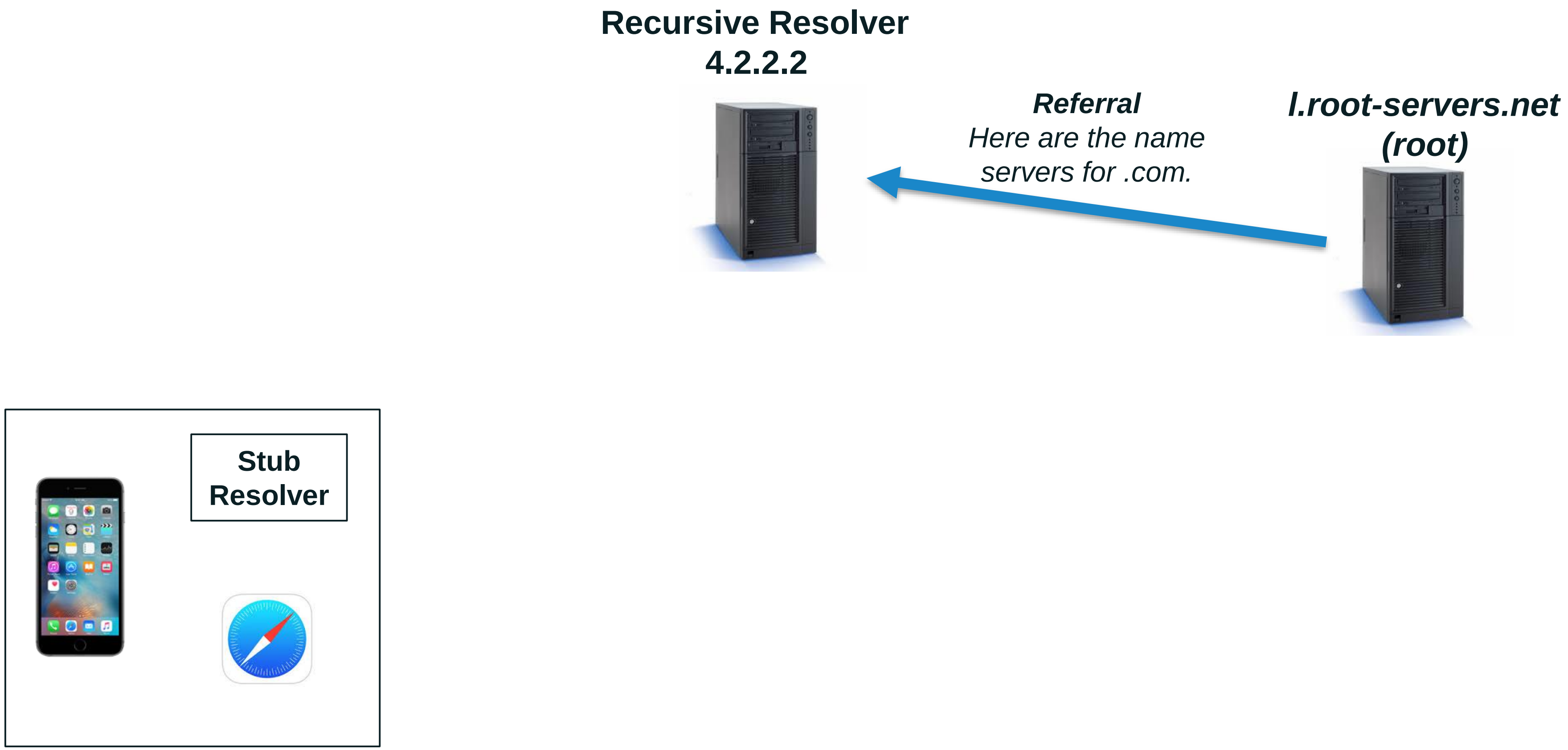
l.root-servers.net (root)

the nearest of the +1200 root server instances

Stub Resolver

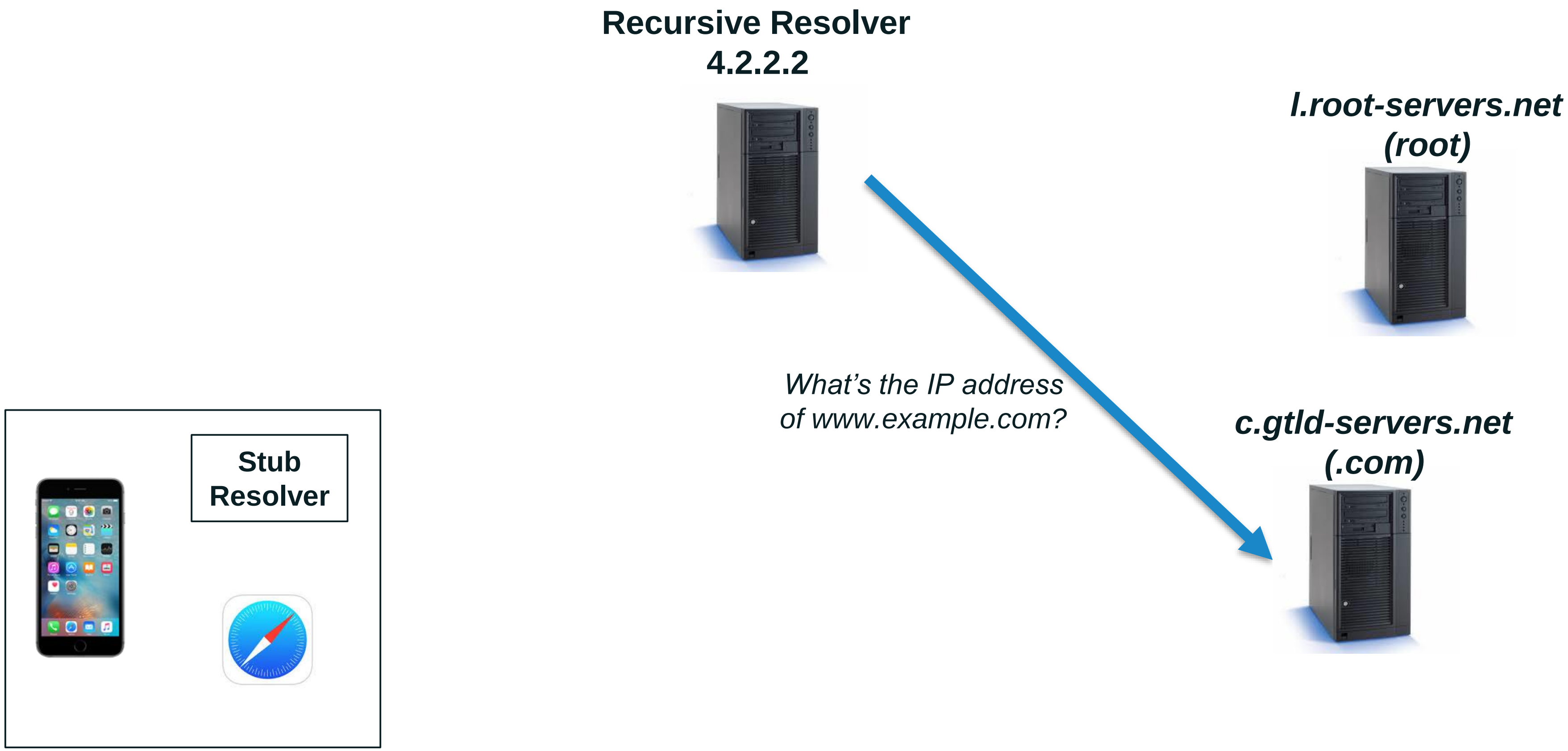
Resolution Process

Root server returns a referral to *.com*



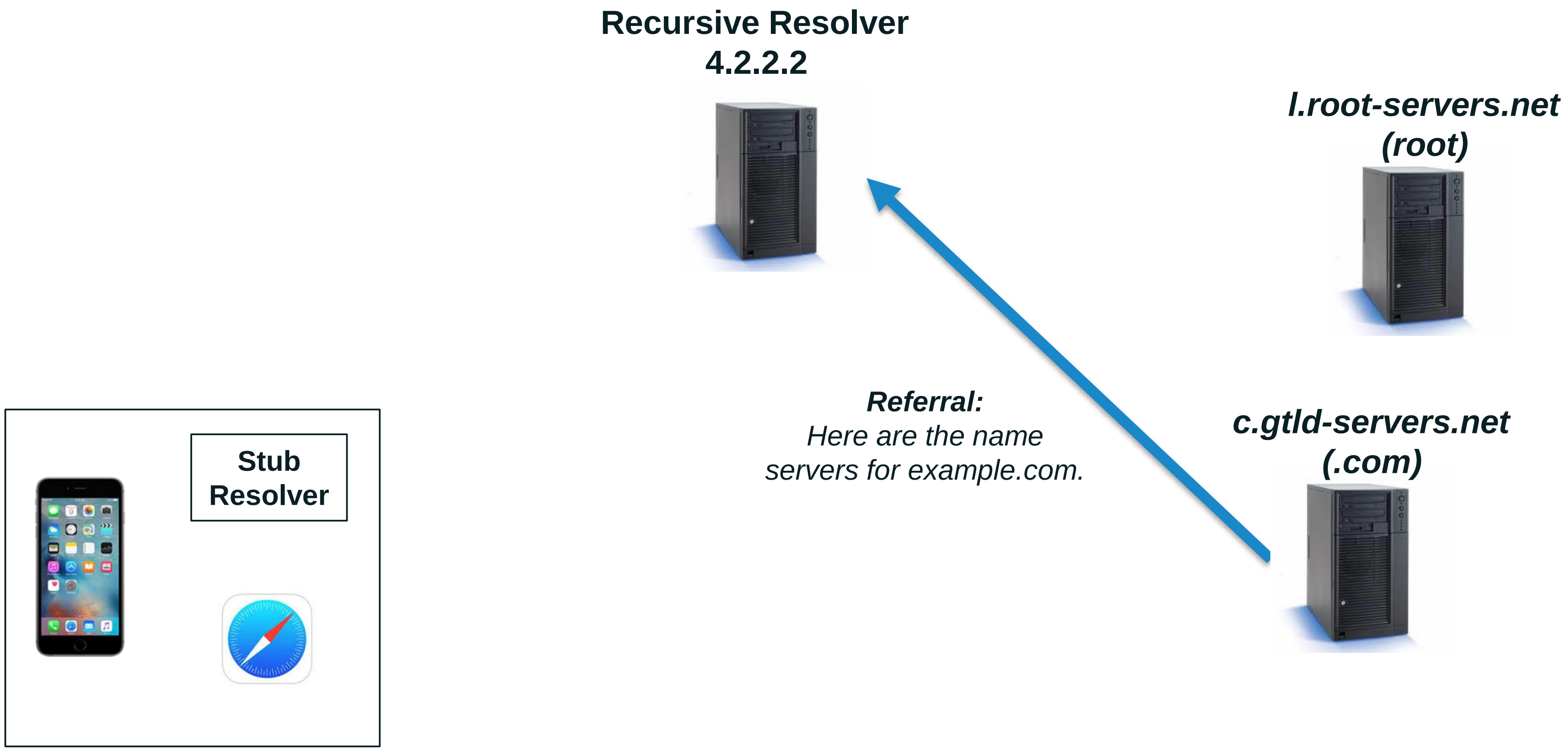
Resolution Process

Recursive resolver queries a *.com* server



Resolution Process

.com server returns a referral to *example.com*



Recursive Resolver
4.2.2.2

l.root-servers.net
(root)

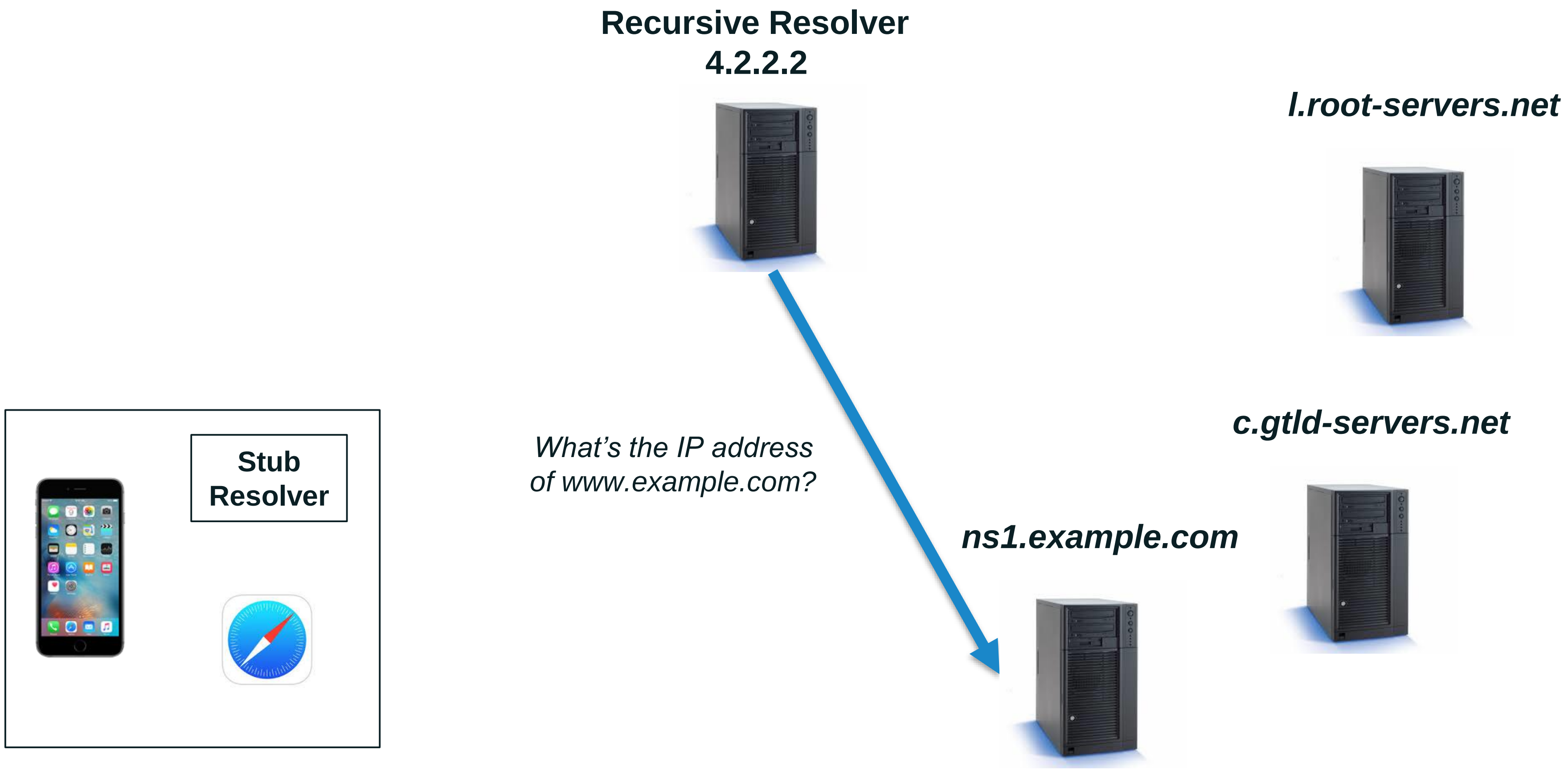
Referral:
Here are the name
servers for example.com.

c.gtld-servers.net
(.com)

Stub
Resolver

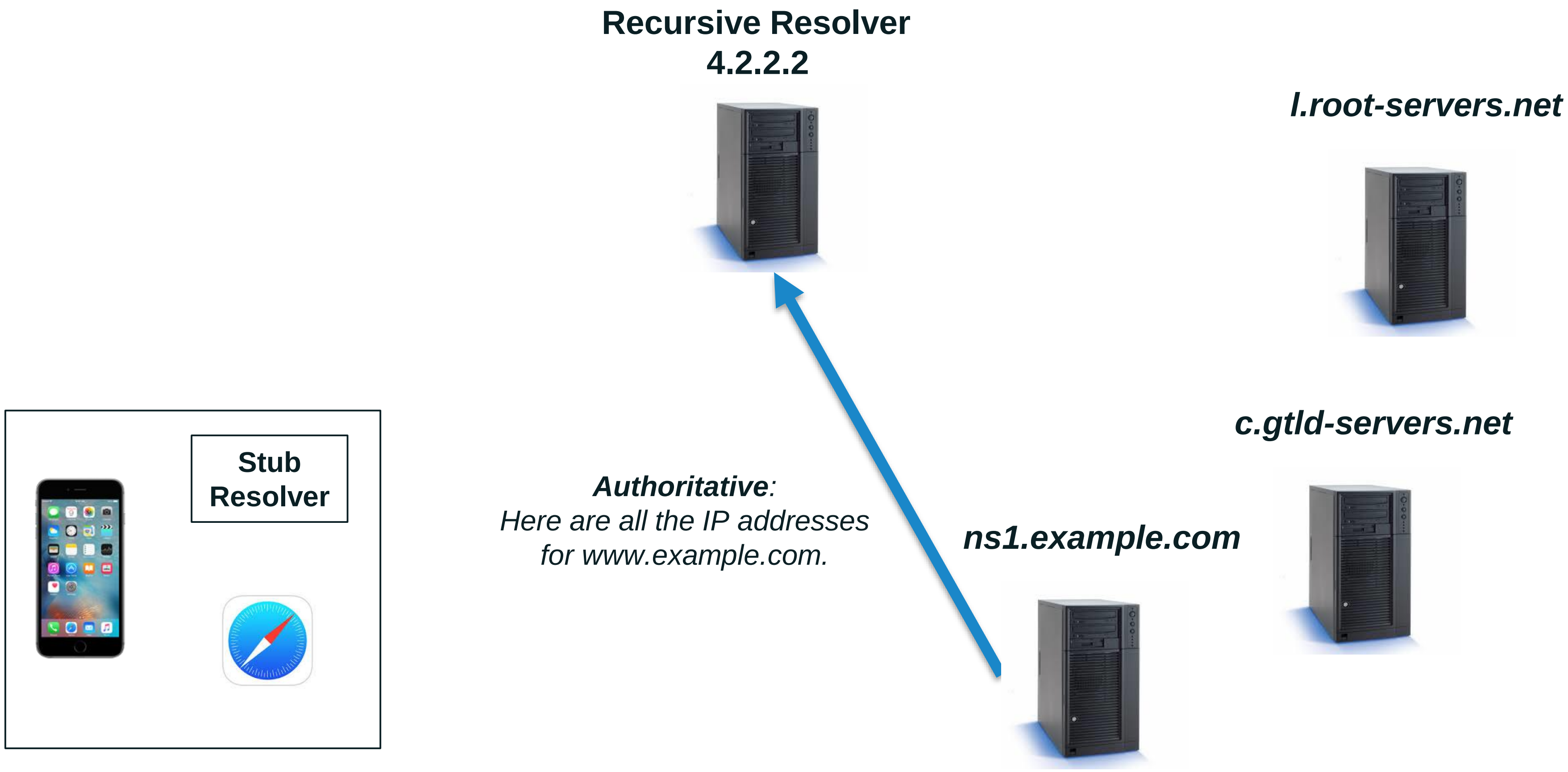
Resolution Process

Recursive resolver queries an *example.com* server



Resolution Process

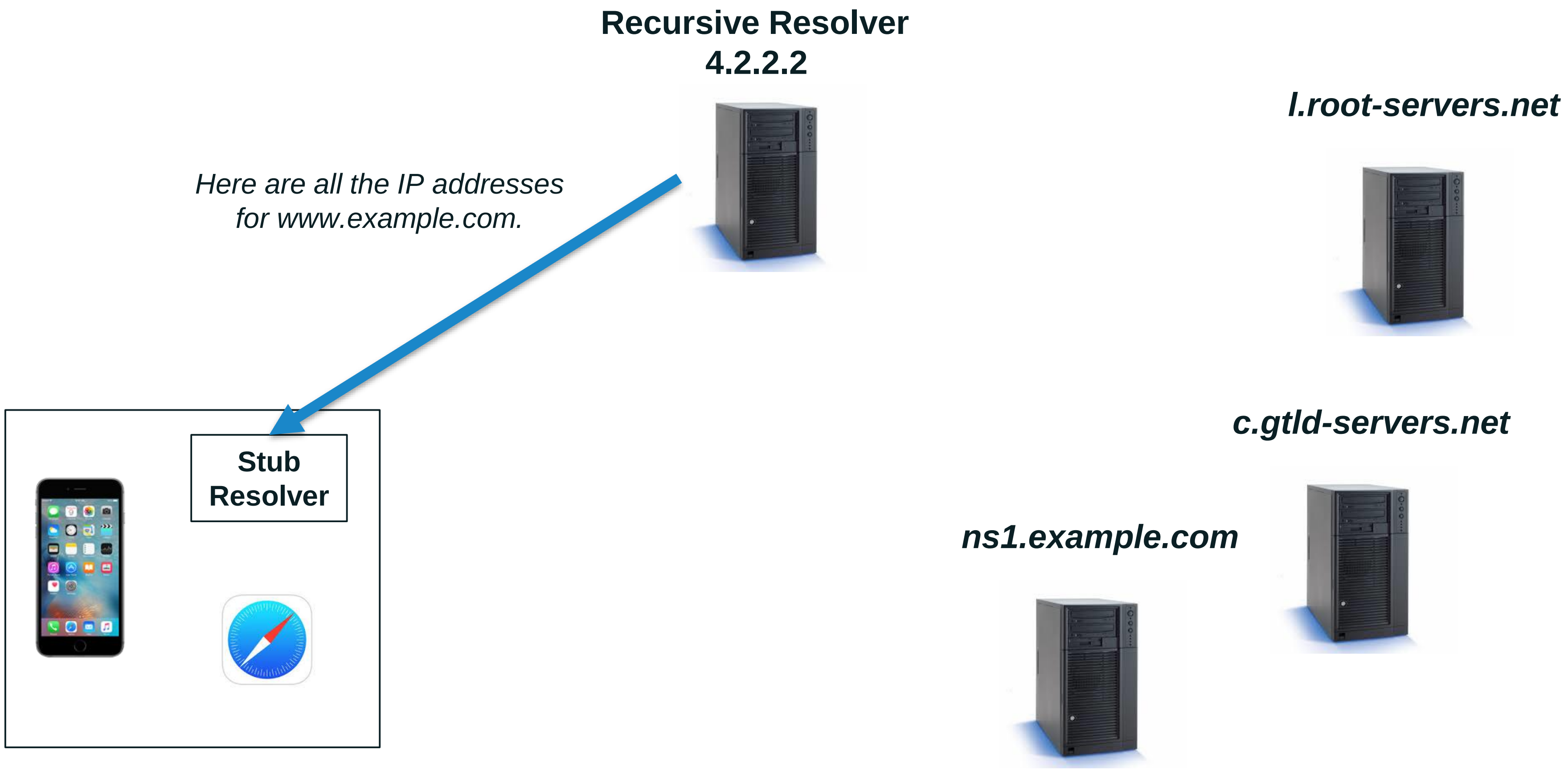
example.com server returns the answer to the query because it is the authoritative for example.com



***Authoritative**:*
Here are all the IP addresses for www.example.com.

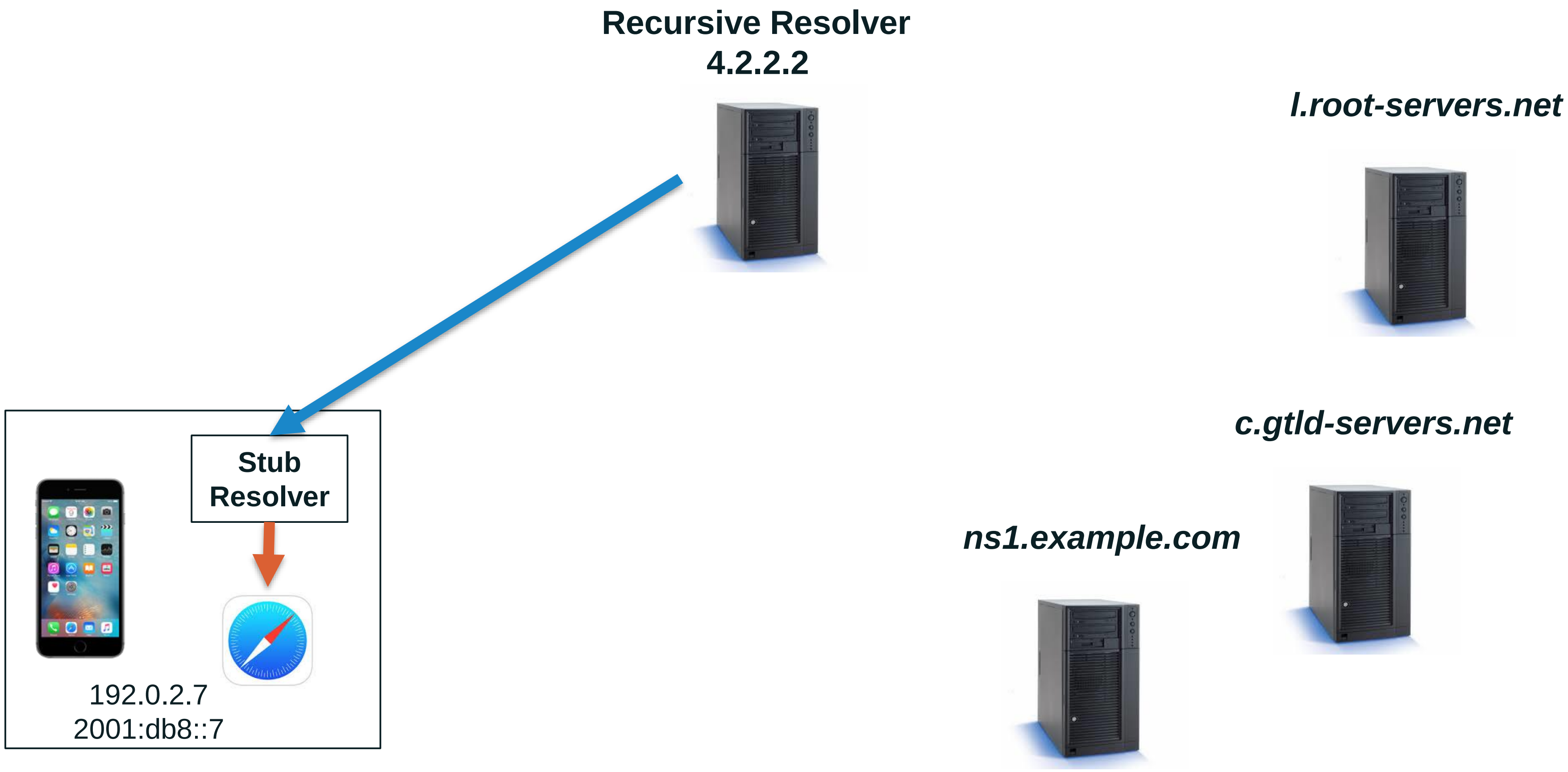
Resolution Process

Recursive resolver returns the answer to the query to the stub resolver



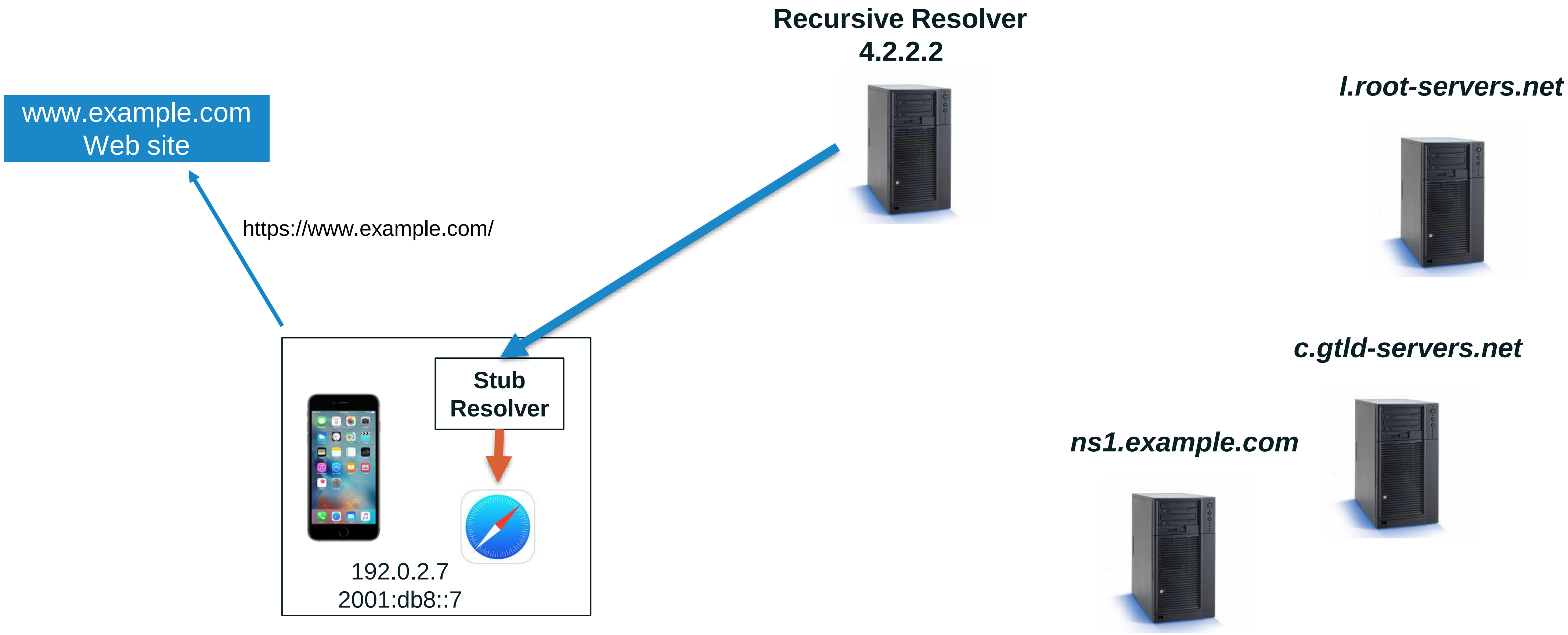
Resolution Process

Stub resolver returns the IP addresses to Safari



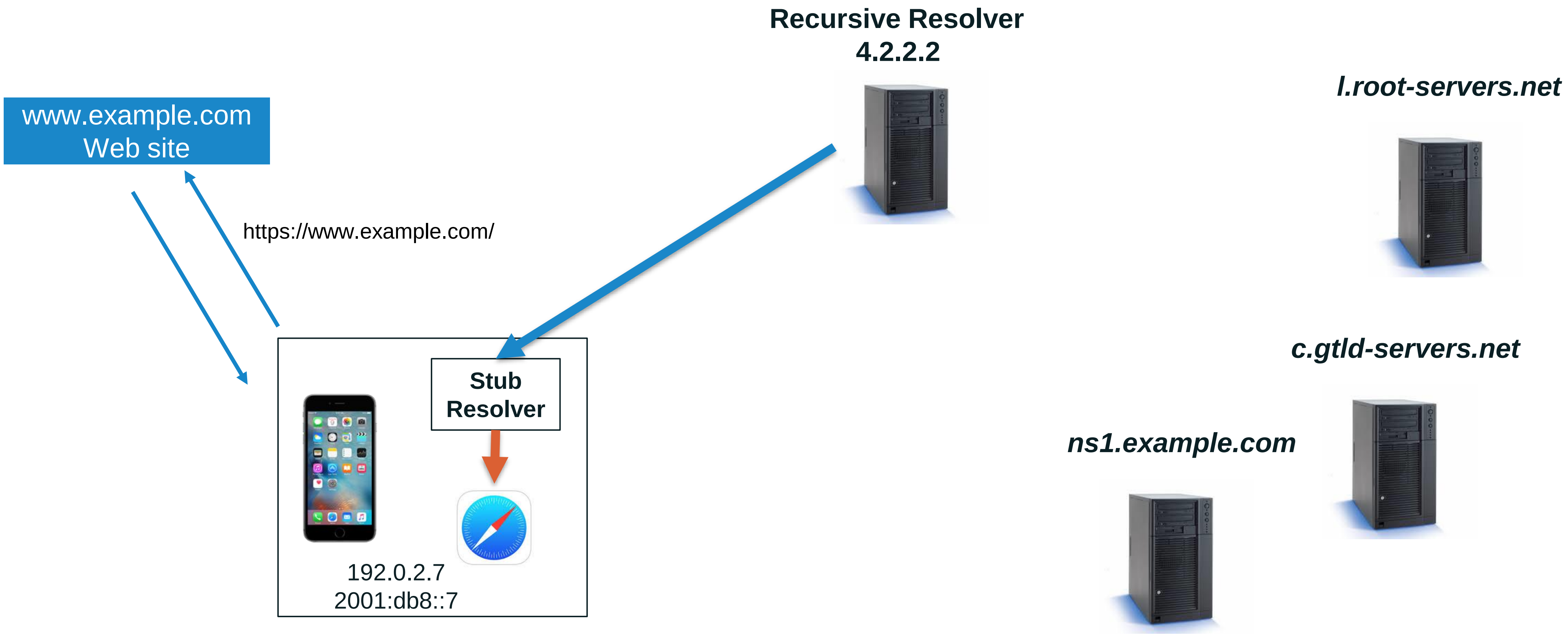
Post Resolution Process

Safari can now open the session with example.com web site.



Post Resolution Process

Example.com web site should reply to the user



Caching

Understanding Caching

- When a recursive resolver boots up, it has no DNS data for specific domain names (except the root name servers, which are in its configuration files).
- Each time the recursive resolver learns the answer for a query, it *caches* the data to re-use for any future identical queries.
- It only caches the answer for a limited time: the TTL of the RR.
- When the TTL expires, the resolver clears that data from its cache. Any future query results in a fresh lookup.
- Caching **speeds up the resolution process** and lowers potential load throughout the DNS.

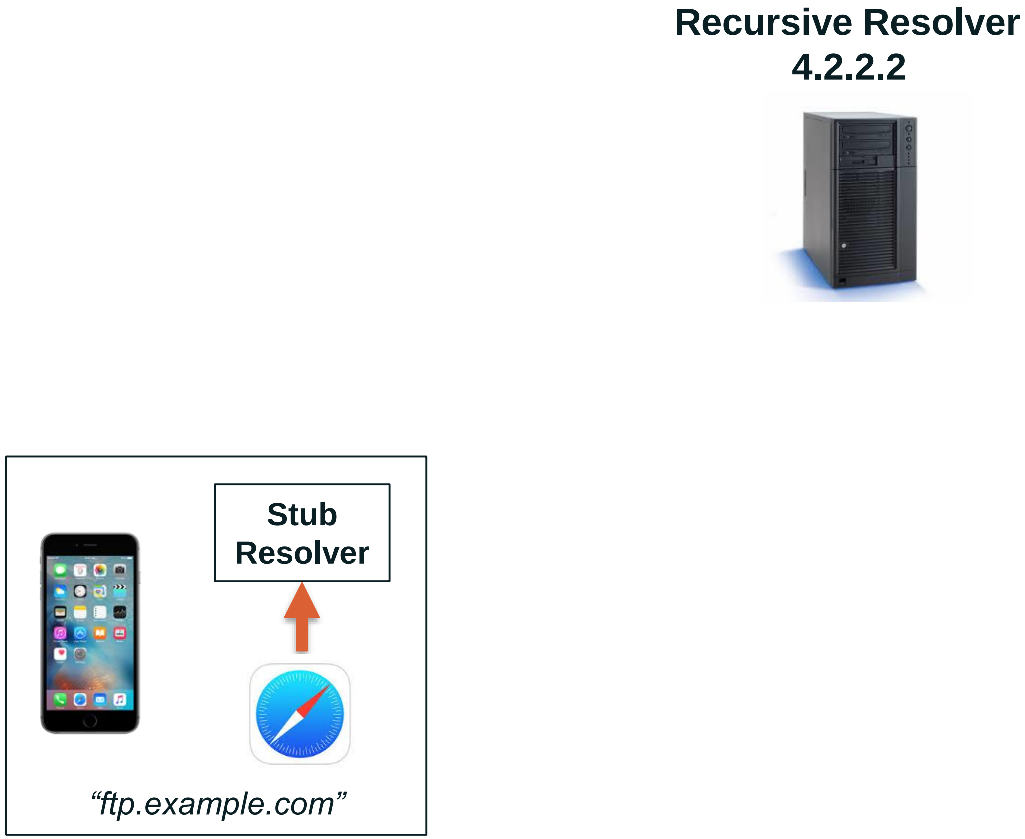
Resolution Process (caching)

- After the previous query, the recursive resolver at 4.2.2.2 now knows:
    - Names and IP addresses of the .com servers
    - Names and IP addresses of the example.com servers
    - IP addresses for www.example.com
- It caches all that data so that it can answer future queries quickly, without repeating the entire resolution process.

Let’s look at another query immediately following the first query . . .

Resolution Process (caching)

A user types *ftp.example.com* into Safari, and it calls the stub resolver function to resolve the name



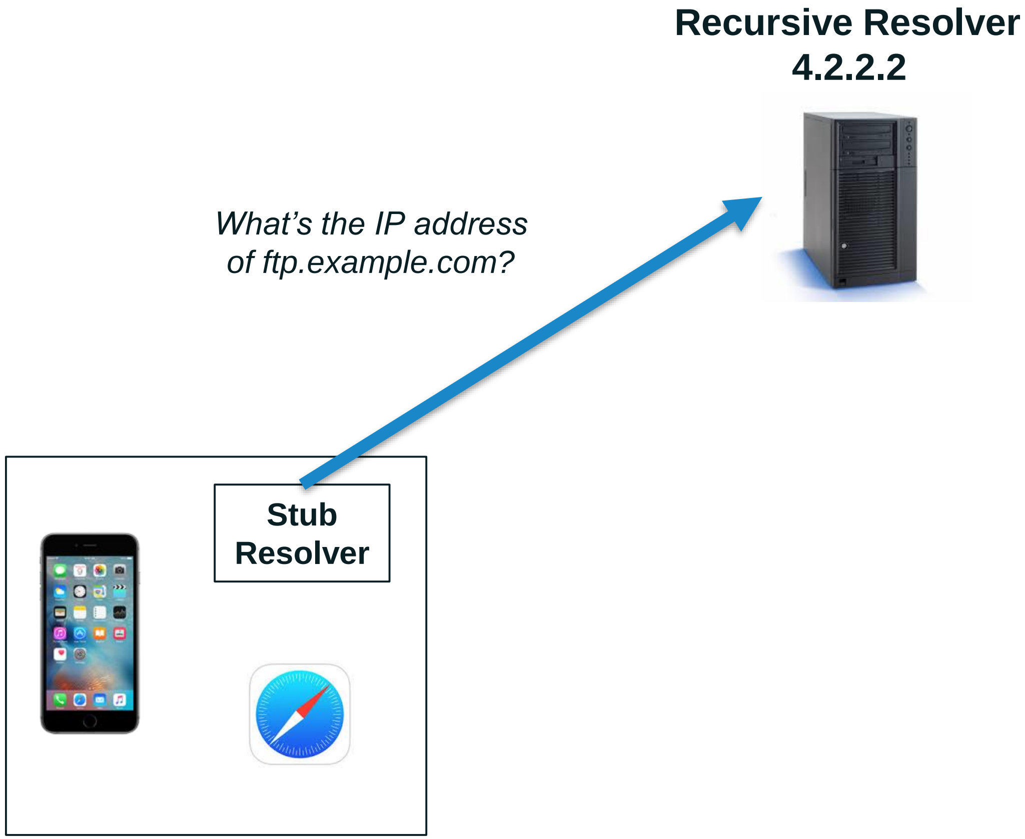
Recursive Resolver
4.2.2.2

Stub Resolver

"ftp.example.com"

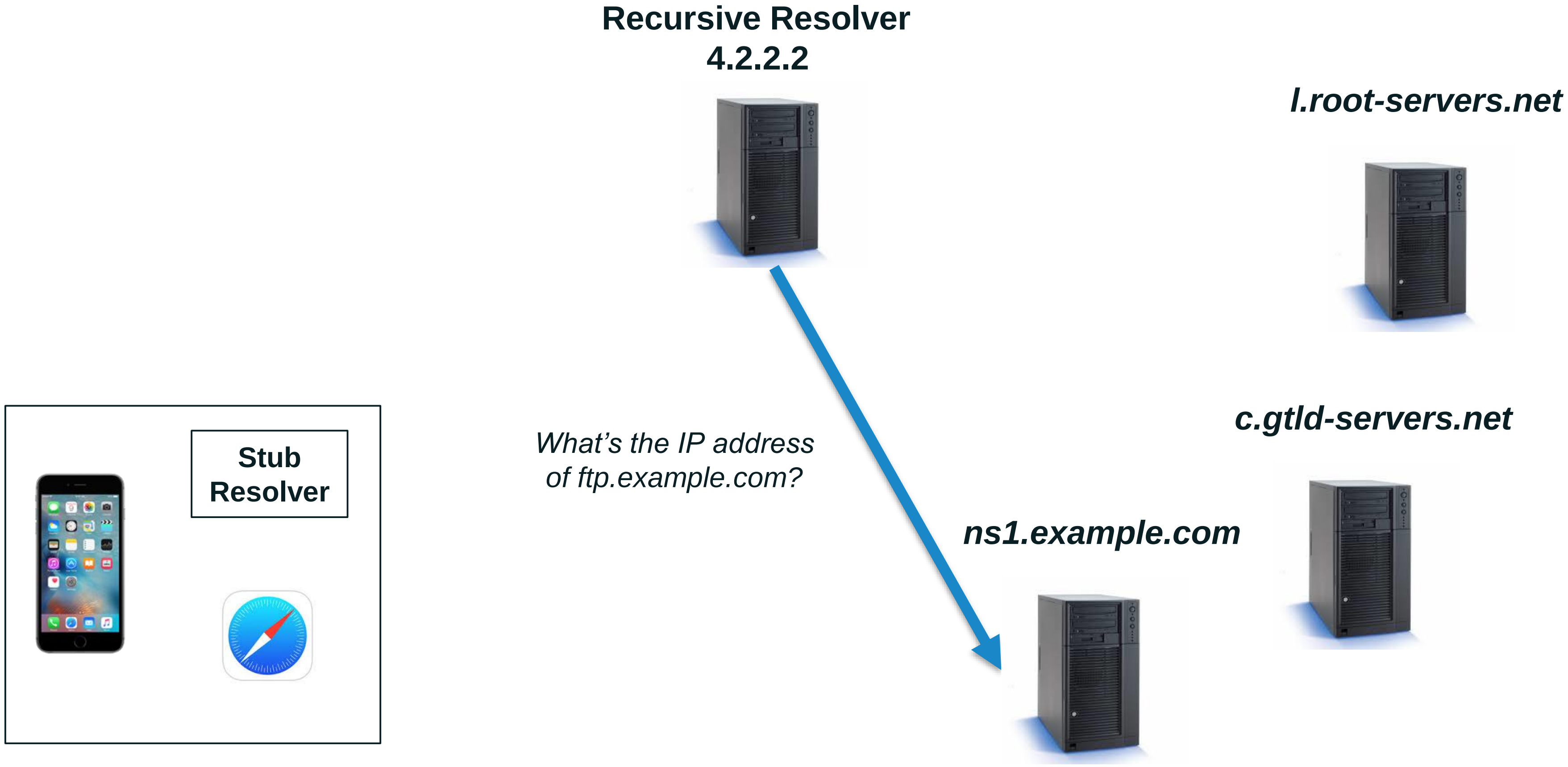
Resolution Process (caching)

The phone’s stub resolver sends a query for *ftp.example.com*/IN/A to 4.2.2.2



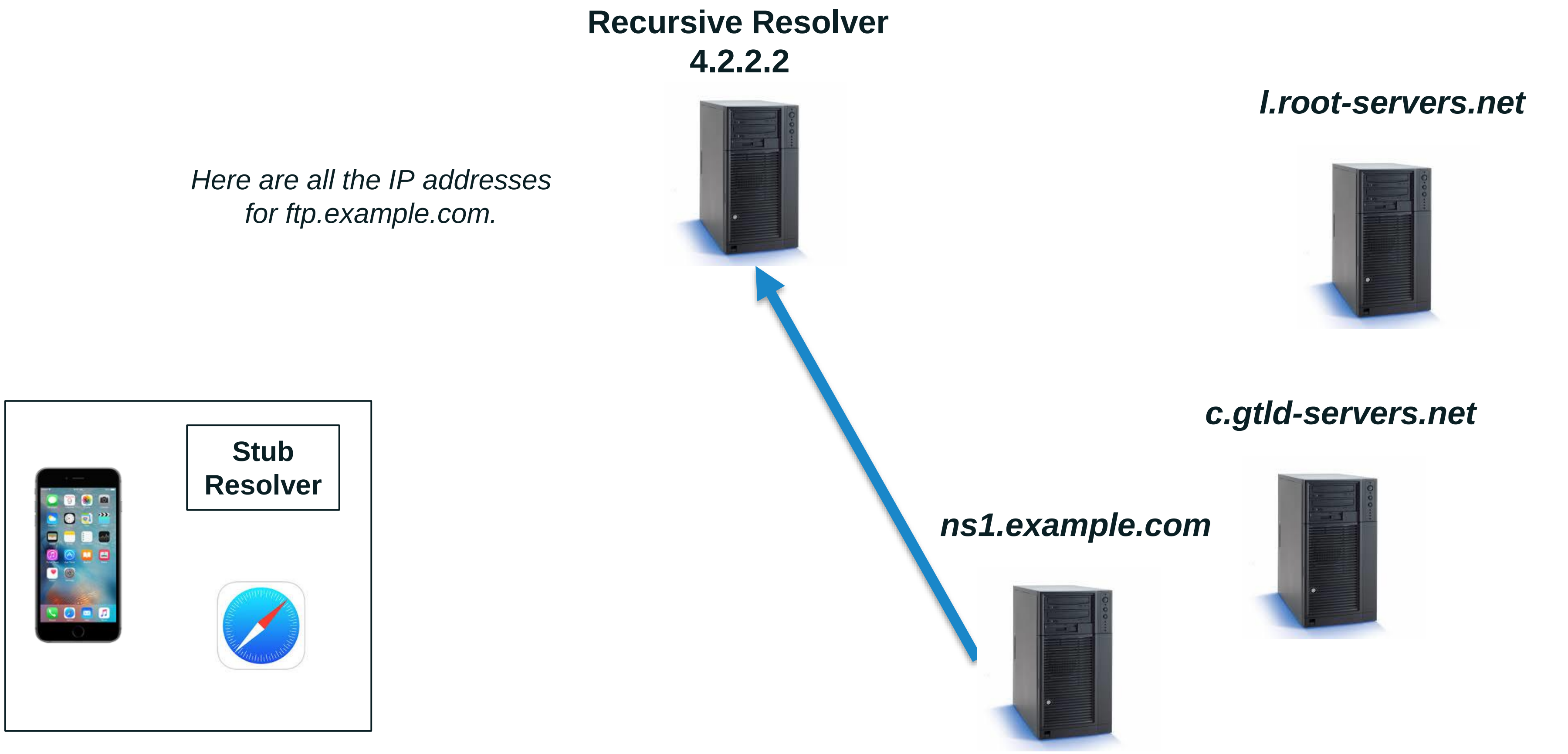
Resolution Process (caching)

Recursive resolver goes directly to example.com servers because it has that data in its cache



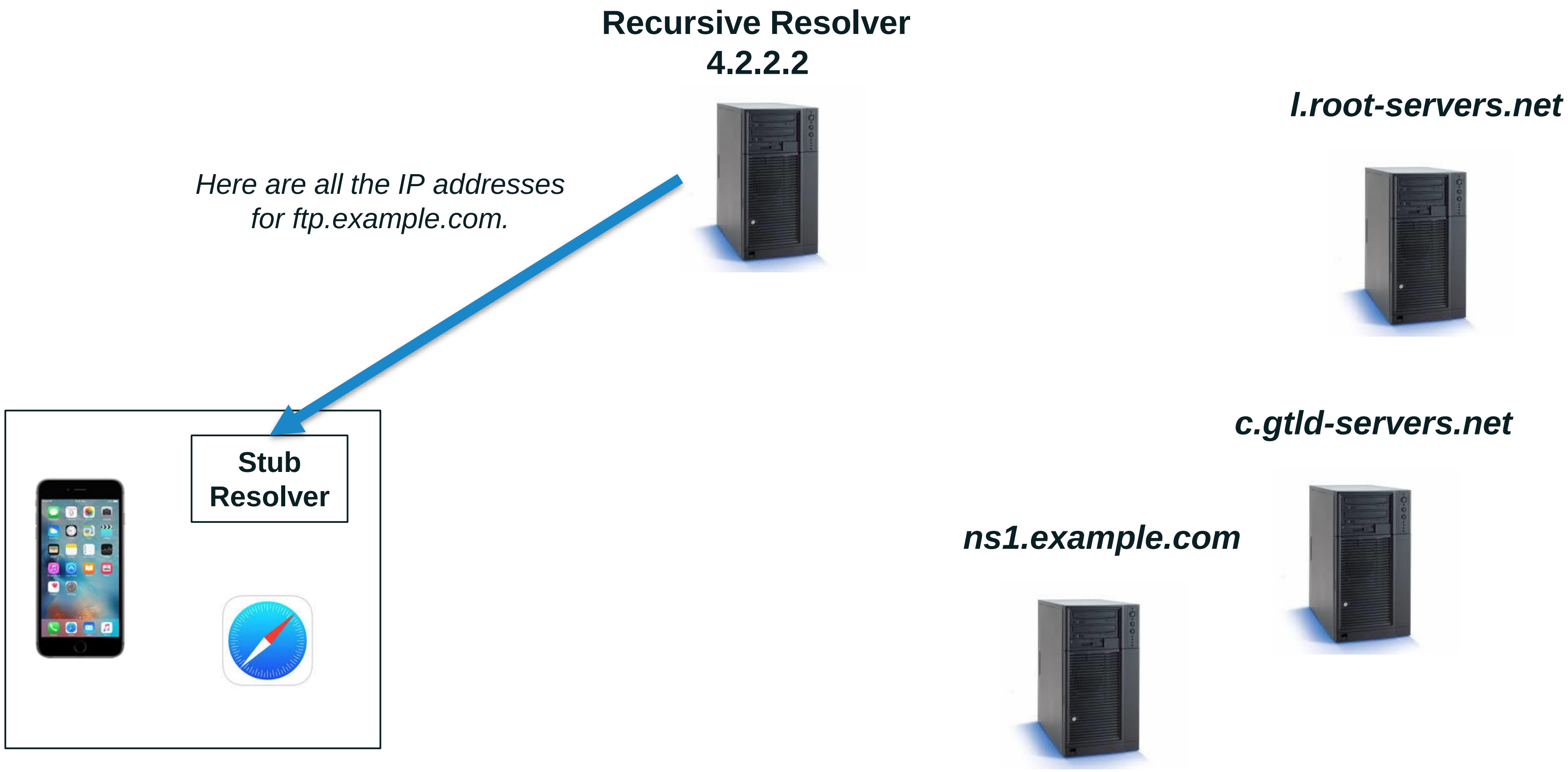
Resolution Process (caching)

example.com server returns the answer to the query



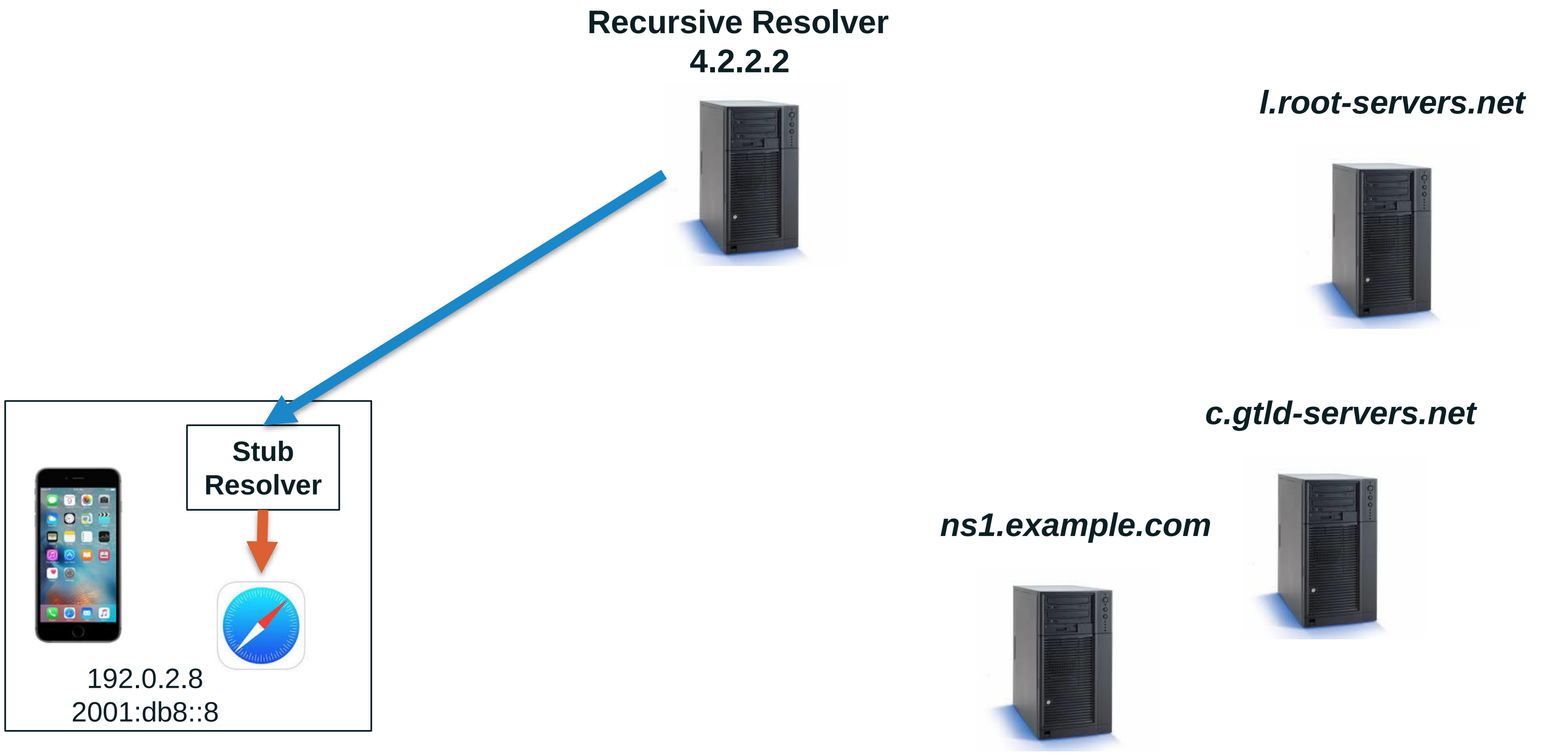
Resolution Process (caching)

Recursive resolver returns the answer to the query to the stub resolver



Resolution Process (caching)

Stub resolver returns the IP addresses to Safari



DNS Resilience

DNS Resilience #1

- Zones may and **should have multiple authoritative** servers
  - Provides redundancy
  - Spreads the query load

Authoritative Server Synchronization

- How do you keep a zone’s data in sync across multiple authoritative servers?
- Fortunately, zone replication is built into the DNS protocol
- A zone’s ***primary*** name server has the definitive zone data
  - Changes to the zone are made on the primary
- A zone’s ***secondary*** server retrieves the zone data from another authoritative server via a ***zone transfer***
  - The server it retrieves from is called the ***primary server***
- Zone transfer is initiated by the secondary
  - Secondary polls the primary periodically to check for changes

DNS Resilience #2 – (Root Server System's Resiliency)

- A root server operator may deploy copies of the root server it operates anywhere in the world using a technique called ***anycast***
  - Provides **redundancy and resiliency** to global DNS infrastructure
  - Spreads the load on its root server
- Each of those copies are called ***instances*** of the root server
- All instances should have identical DNS data to ensure they all give the same answers

Interested in looking at the content of your stub resolver ???

- Windows users:
  - Ctrl + R , then type ***cmd*** and press Enter
  - In the new window, type ***ipconfig /displaydns:*** this will show the content of your local cache (stub resolver).
  - To remove everything from this cache, type ***ipconfig /flushdns*** and press **Enter**

Additional resources

- TE Course Catalogue - https://www.icann.org/resources/pages/tech-engagement-training-course-catalogue-2021-04-22-en
- OCTO publications - https://www.icann.org/resources/pages/octo-publications-2019-05-24-en
- Recent Publication - A Primer in Registration Data Access Protocol (RDAP) Performance - https://www.icann.org/en/system/files/files/octo-024-17may21-en.pdf
- Recent Blog: “How ICANN Strengthened its Technical Engagement Around the World”: https://www.icann.org/en/blogs/details/how-icann-strengthened-its-technical-engagement-around-the-world-23-4-2021-en
- Domain Abuse Activity Reporting - https://www.icann.org/octo-ssr/daar
- ITHI - https://ithi.research.icann.org/
- KINDNS - https://community.icann.org/display/KINDNS

ICANN

One World, One Internet

Visit us at **icann.org**

- @icann
- facebook.com/icannorg
- youtube.com/icannnews
- flickr.com/icann
- linkedin/company/icann
- slideshare/icannpresentations
- soundcloud/icann