DNS

Some basics of DNS & DNS resiliency

Nicolás Antoniello

LACRALO
18 May 2020

Once upon a time…

ICANN

Names and Numbers

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

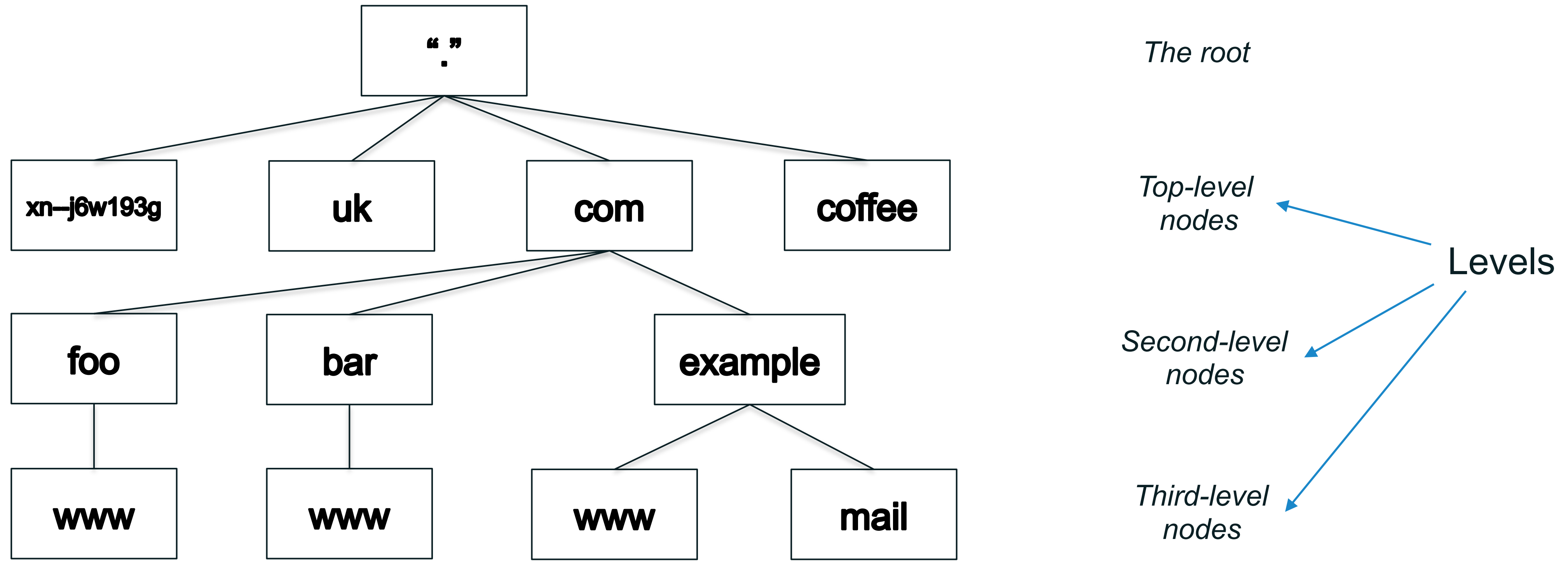
DNS to the Rescue

- 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”

Rise of the DNS !

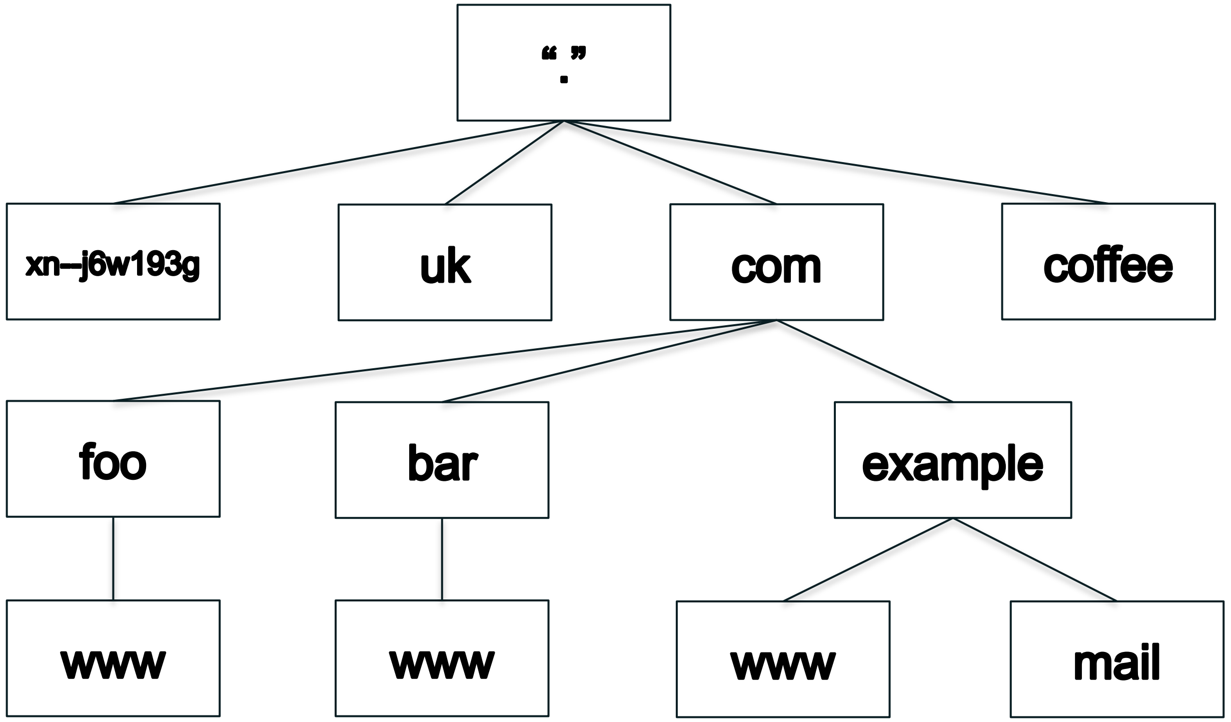
The Name Space

- 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



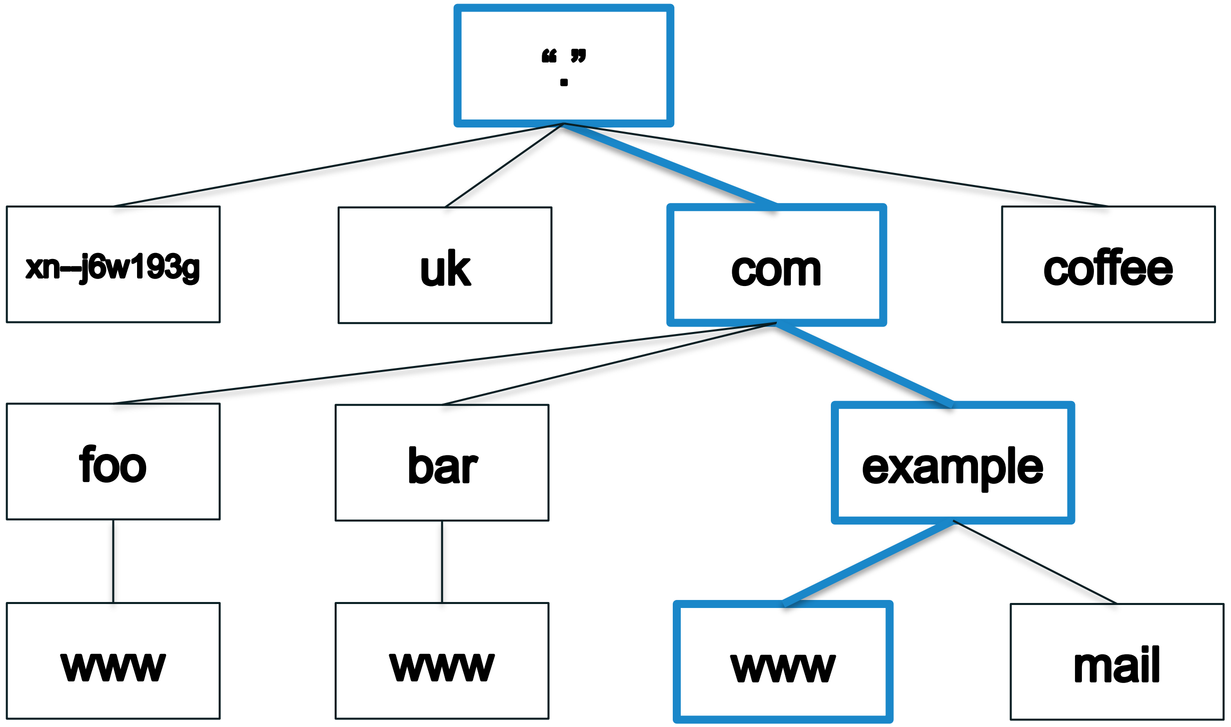
Label Syntax

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



Domain Names

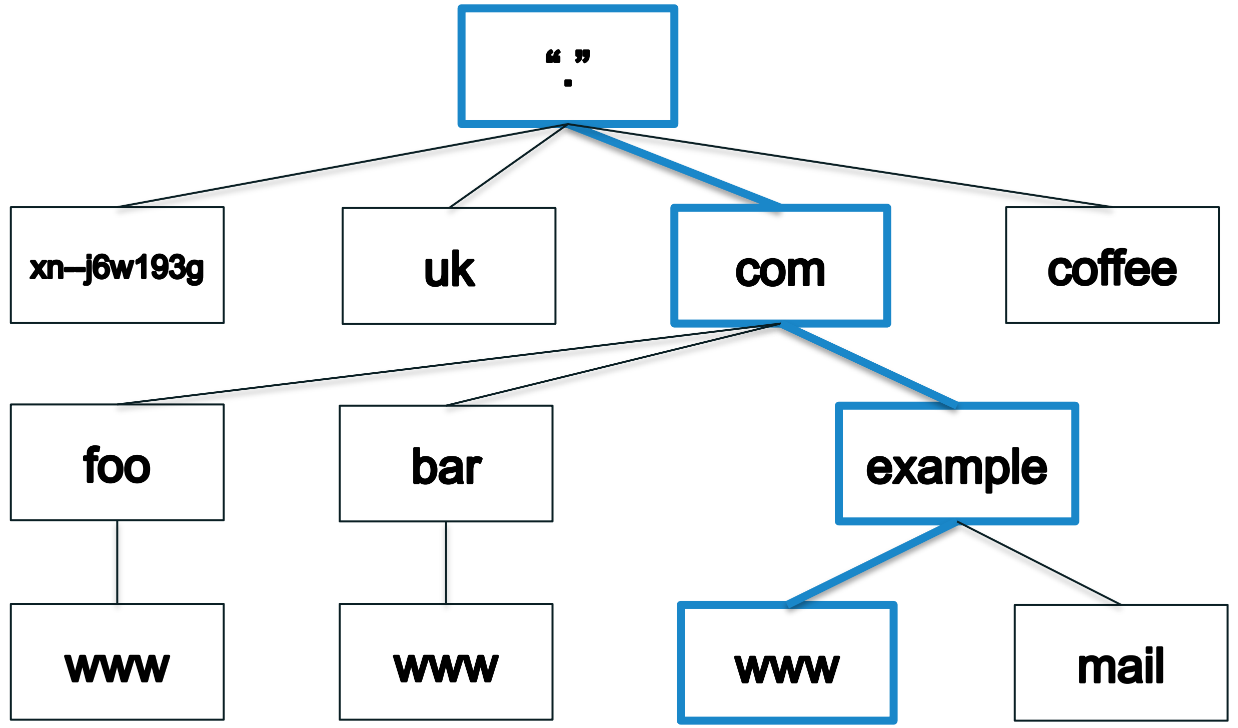
- Every node has a ***domain name***
- That ***domain name*** is built by sequencing node labels from one specified node up to the root, separated by dots
- Highlighted: *www.example.com.*



ICANN

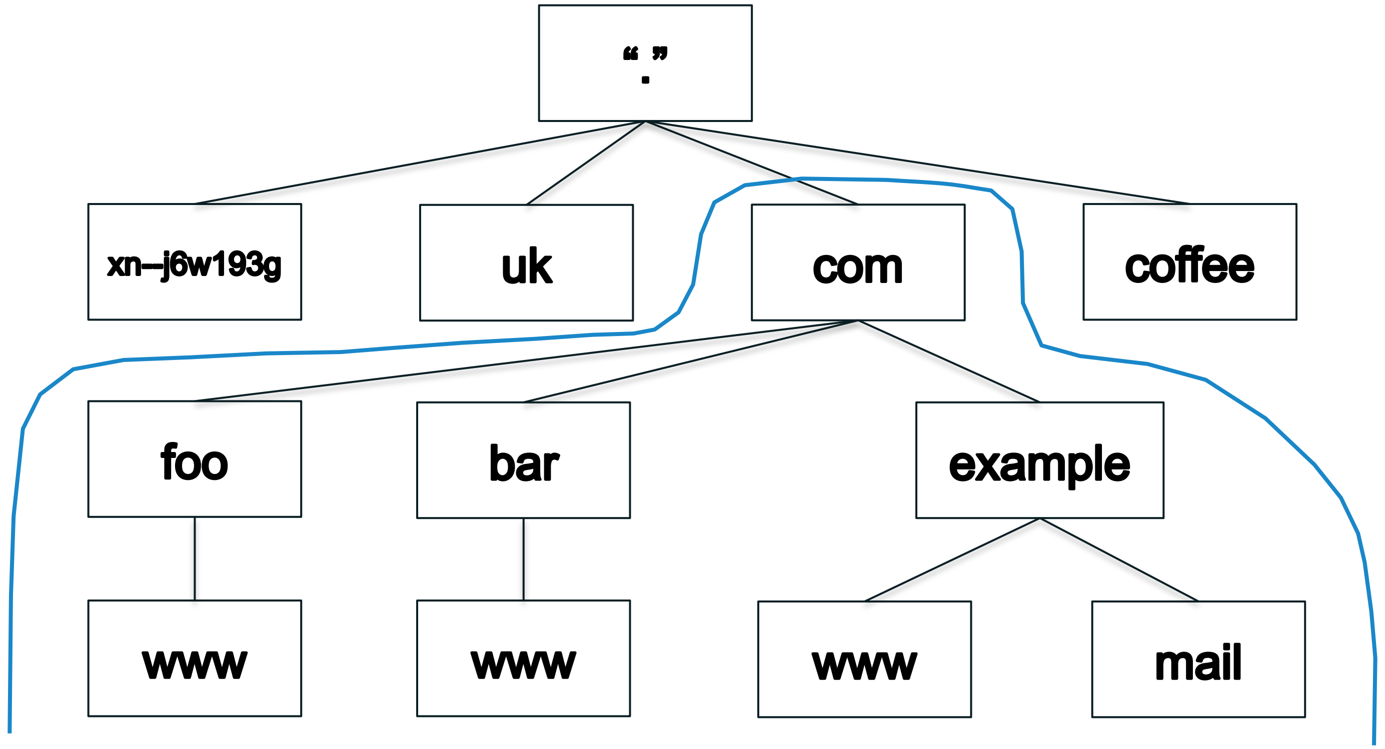
Fully Qualified Domain Names

- 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.*



Domains

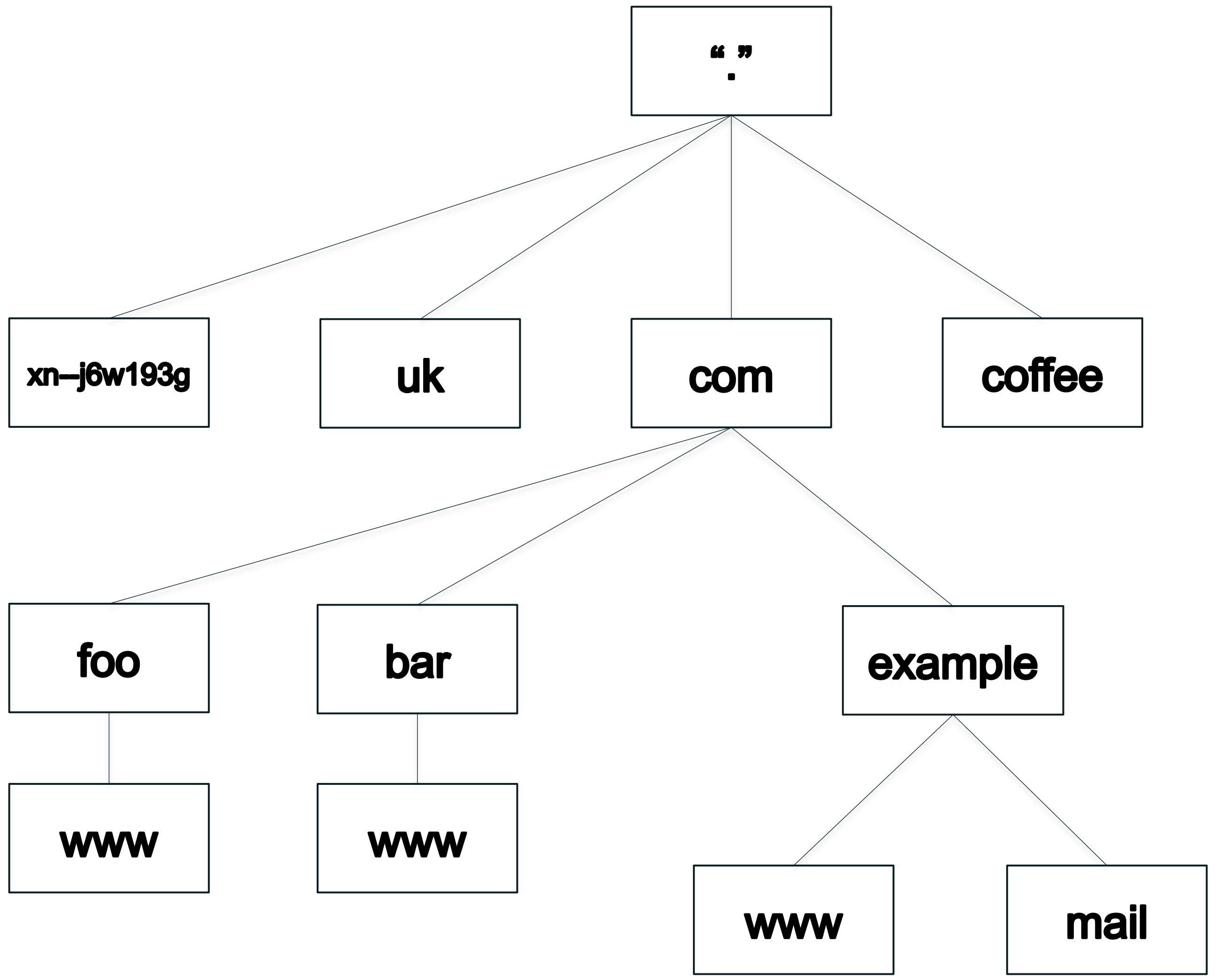
- A ***domain*** is a node and everything below it
- The top node of a domain is the ***apex*** of that domain
- Shown: the *com* domain



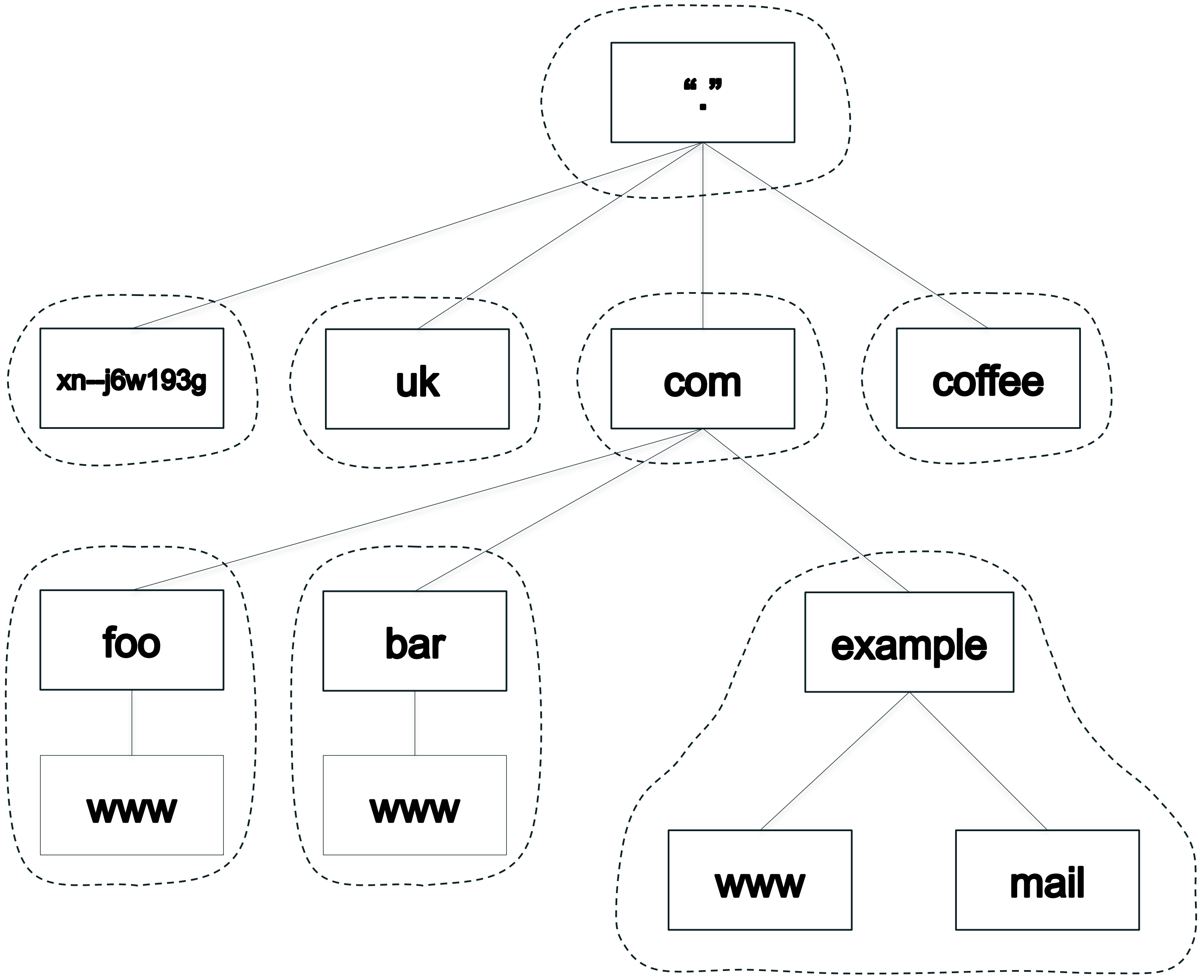
Zones

- 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***

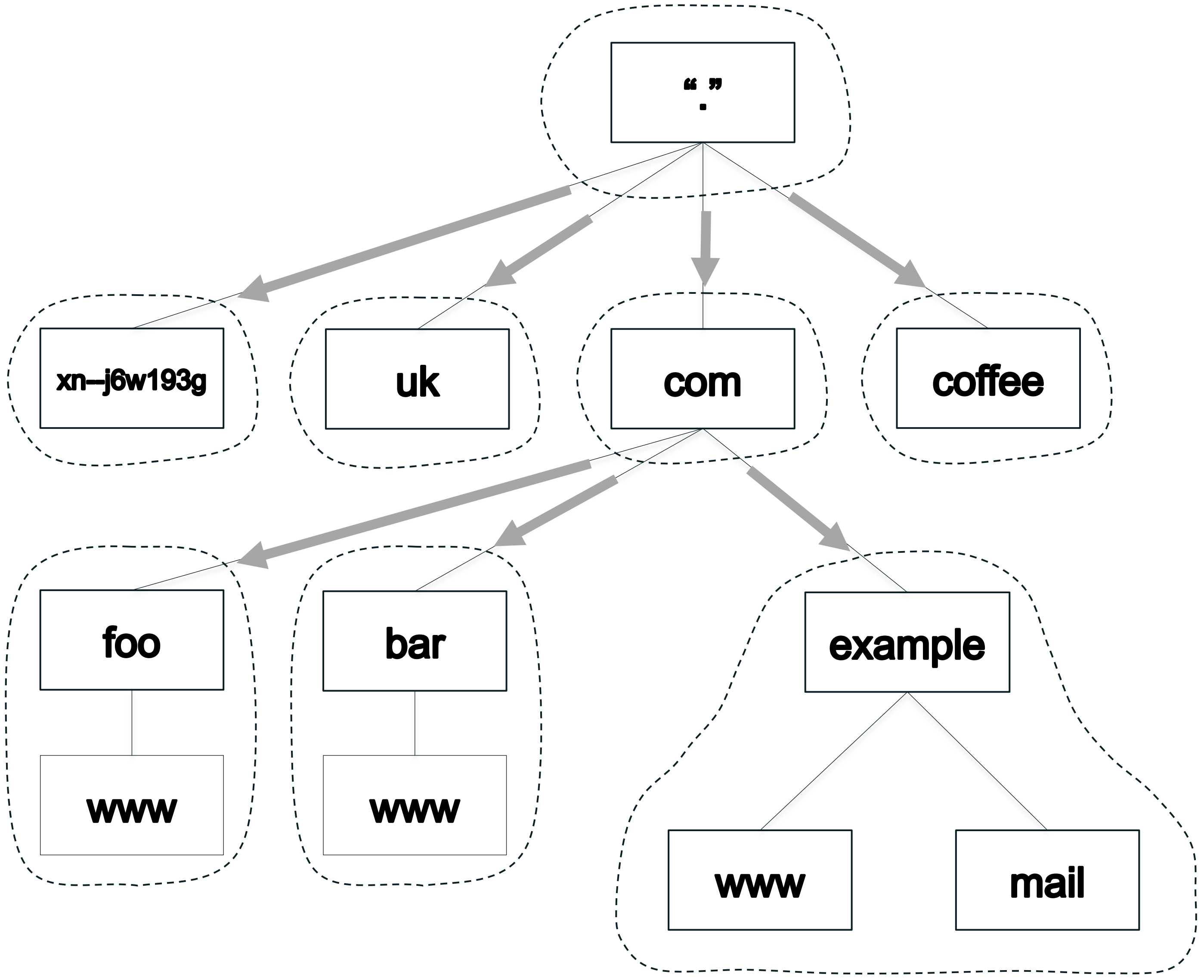
The Name Space



Zones are Administrative Boundaries



Delegation Creates Zones



DNS Database and Data

ICANN

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

Zone Files

- A zone consists of multiple resource records
- All the resource records for a zone are stored in a ***zone file***
- Every zone has (at least) one zone file
- Resource records from multiple zones are never mixed in the same file

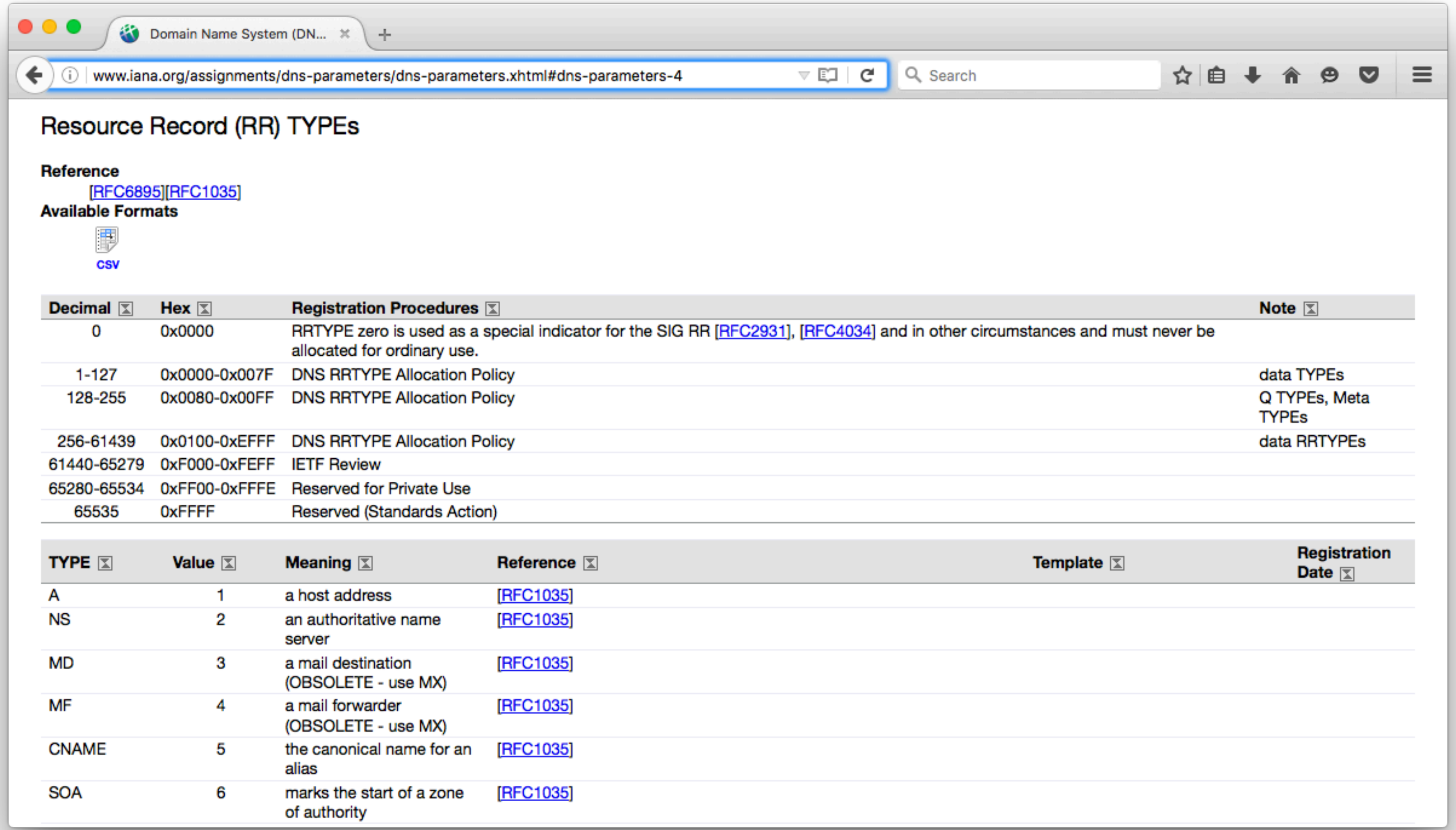
Common Resource Record Types

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)

Lots of Resource Records

- There are many other resource record types
- 87 types allocated
- IANA “DNS Resource Record (RR) TYPE Registry” under “Domain Name System (DNS) Parameters”
    - *http://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-4*

IANA DNS Resource Record (RR) TYPE Registry


Domain Name System (DN...
www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-4
Search

Resource Record (RR) TYPEs

Reference
[RFC6895][RFC1035]
Available Formats
CSV

Decimal	Hex	Registration Procedures	Note
0	0x0000	RRTYPE zero is used as a special indicator for the SIG RR [RFC2931], [RFC4034] and in other circumstances and must never be allocated for ordinary use.	
1-127	0x0000-0x007F	DNS RRTYPE Allocation Policy	data TYPEs
128-255	0x0080-0x00FF	DNS RRTYPE Allocation Policy	Q TYPEs, Meta TYPEs
256-61439	0x0100-0xEFFF	DNS RRTYPE Allocation Policy	data RRTYPEs
61440-65279	0xF000-0xFEFF	IETF Review	
65280-65534	0xFF00-0xFFFE	Reserved for Private Use	
65535	0xFFFF	Reserved (Standards Action)	

TYPE	Value	Meaning	Reference	Template	Registration Date
A	1	a host address	[RFC1035]		
NS	2	an authoritative name server	[RFC1035]		
MD	3	a mail destination (OBSOLETE - use MX)	[RFC1035]		
MF	4	a mail forwarder (OBSOLETE - use MX)	[RFC1035]		
CNAME	5	the canonical name for an alias	[RFC1035]		
SOA	6	marks the start of a zone of authority	[RFC1035]		

Address Records (A & AAAA)

- Most common use of DNS is mapping domain names to IP addresses
- Two most common types of resource records are:
  - Address (A) record stores mapping for a domain name to an IPv4 address

```
example.com.            A           192.0.2.7
```

  - “Quad A” (AAAA) record stores mapping for a domain name to an IPv6 address

```
example.com.            AAAA        2001:db8::7
```

A Sample of More Resource Record Types

- **TXT**
  - Arbitrary text
- **URI**, **NAPTR**
  - Map domain names to URIs
- **TLSA**
  - Used by DANE to associate X.509 certificates with a domain name
- **CDS**, **CDNSKEY**, **CSYNC**
  - Child-parent synchronization
- **X25**, **ISDN**, **ATMA**
  - Addresses for non-IP networking protocols
- **LOC**, **GPOS**
  - Location information
- …and many more.

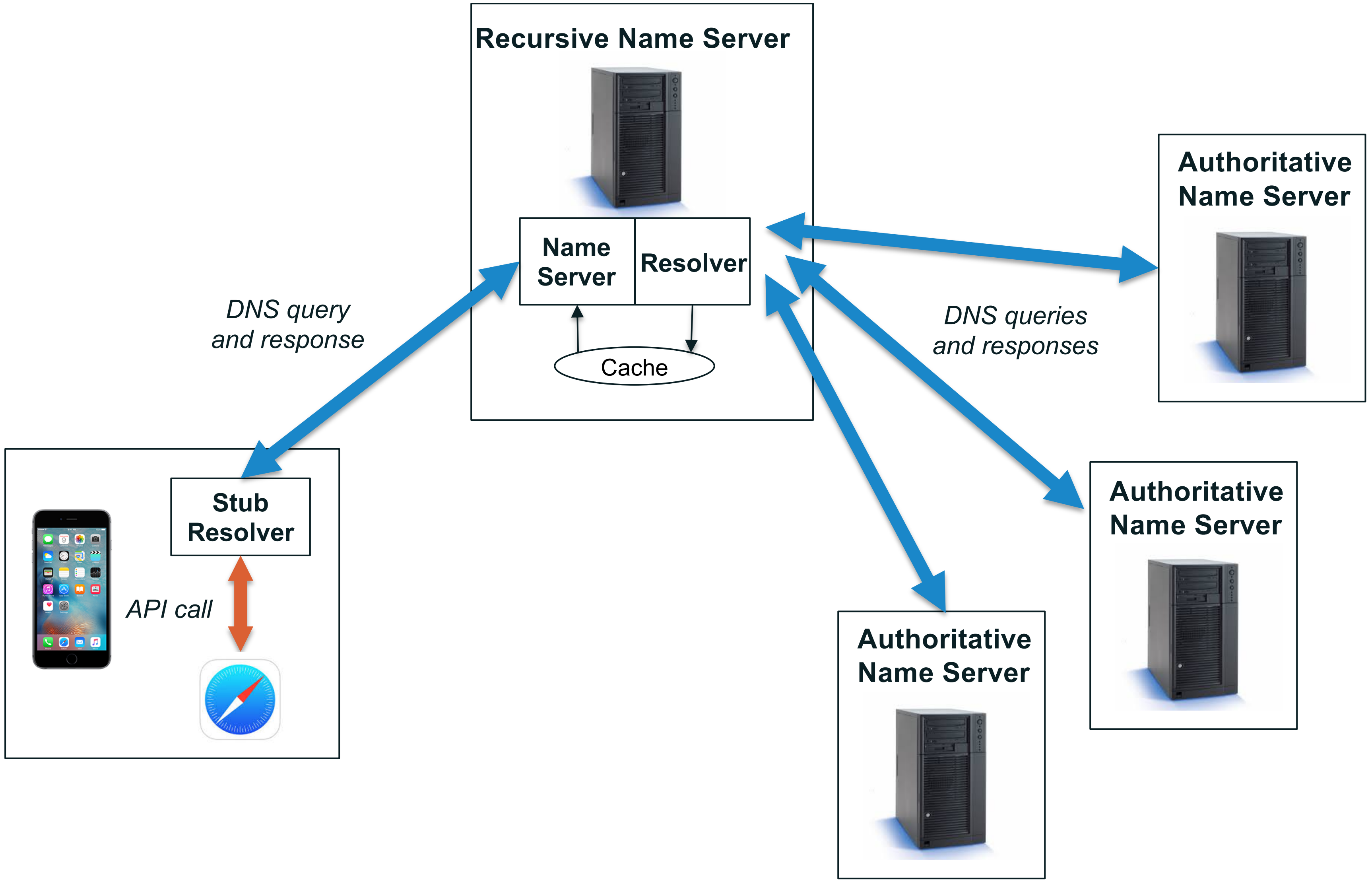
Resolution Process

ICANN

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


Recursive Name Server
Name Server
Resolver
Cache
DNS query and response
DNS queries and responses
Authoritative Name Server
Authoritative Name Server
Authoritative Name Server
Stub Resolver
API call

Name Servers and Zones

- Name servers answer queries
- A name server ***authoritative*** for a zone has complete knowledge of that zone
  - 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

- Stub resolvers, recursive name servers and authoritative name servers cooperate to look up DNS data in the name space

- A DNS query always comprises three parameters:
    - Domain name, class, type
        - E.g., *www.example.com*, IN, A

- Two kinds of queries:
    - Stub resolvers send ***recursive*** queries
        - "I need the complete answer or an error."
    - Recursive name servers send ***non-recursive*** or ***iterative*** queries
        - "I can do some of the lookup work myself and will accept a ***referral***."

The Resolution Process

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

We will go though resolution process step by step…

The Resolution Process

But first…

- How do you start the resolution process if there is no local data (you are a resolver and you have just booted up)?
    - Empty cache, and/or
    - Not authoritative for any zones

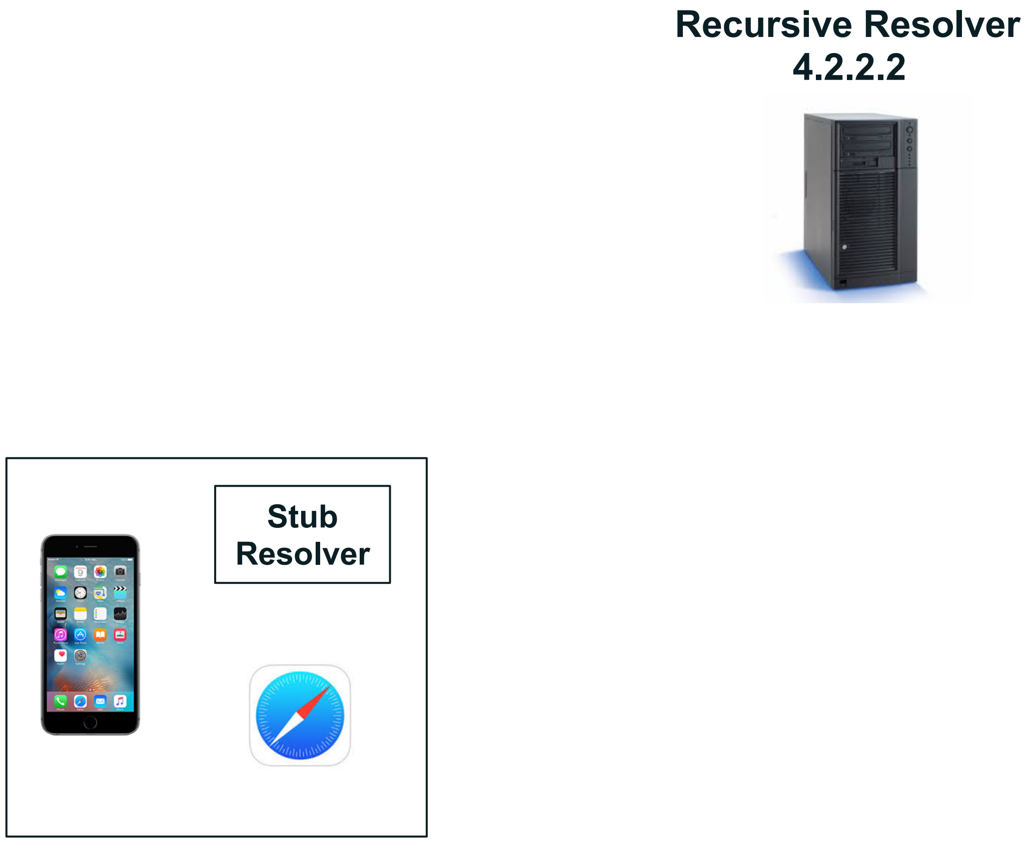
- No choice but to start at the root zone
    - The ***root name servers*** are the servers authoritative for the root zone

- But how does a resolver find the NS, A, and AAAA records for the root name servers?
    - They must be configured (in fact, most of DNS software come preloaded with an up to date version of the file called ***hint file***)
    - No way to discover them

- The ***root hints file*** contains the names and IP addresses of the root name servers
    - https://www.iana.org/domains/root/files

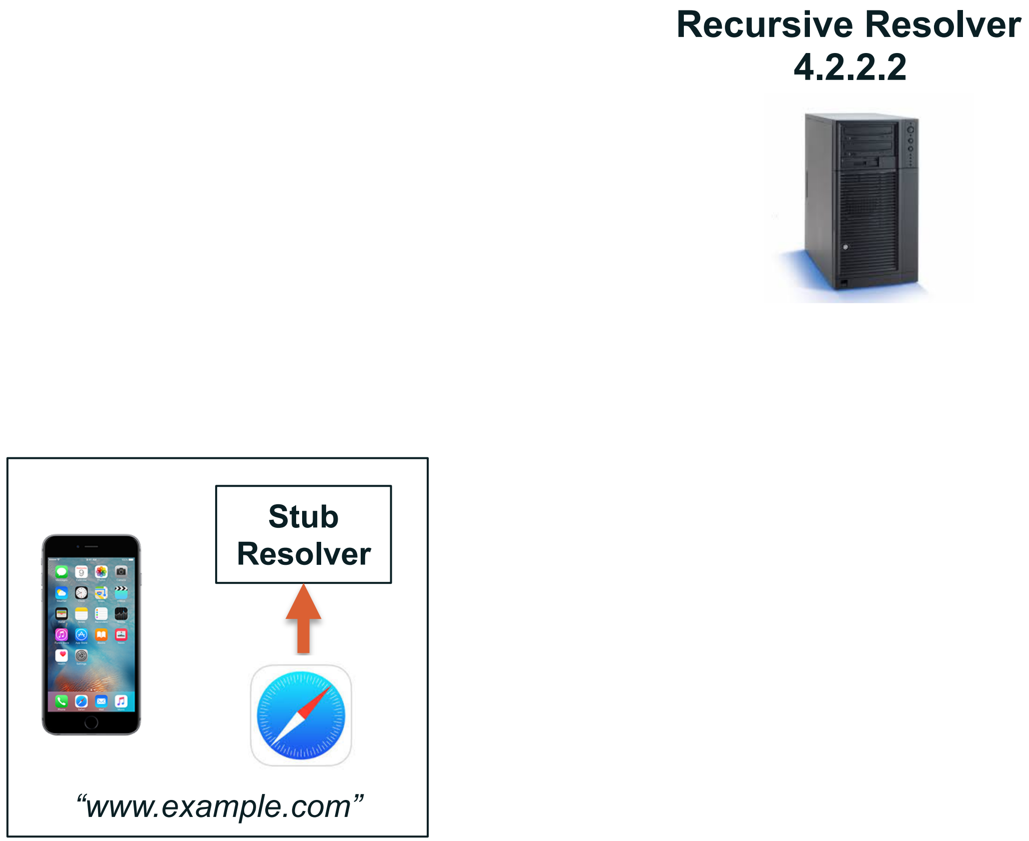
Resolution Process

The phone’s stub resolver is configured to send queries to the recursive resolver with IP address 4.2.2.2



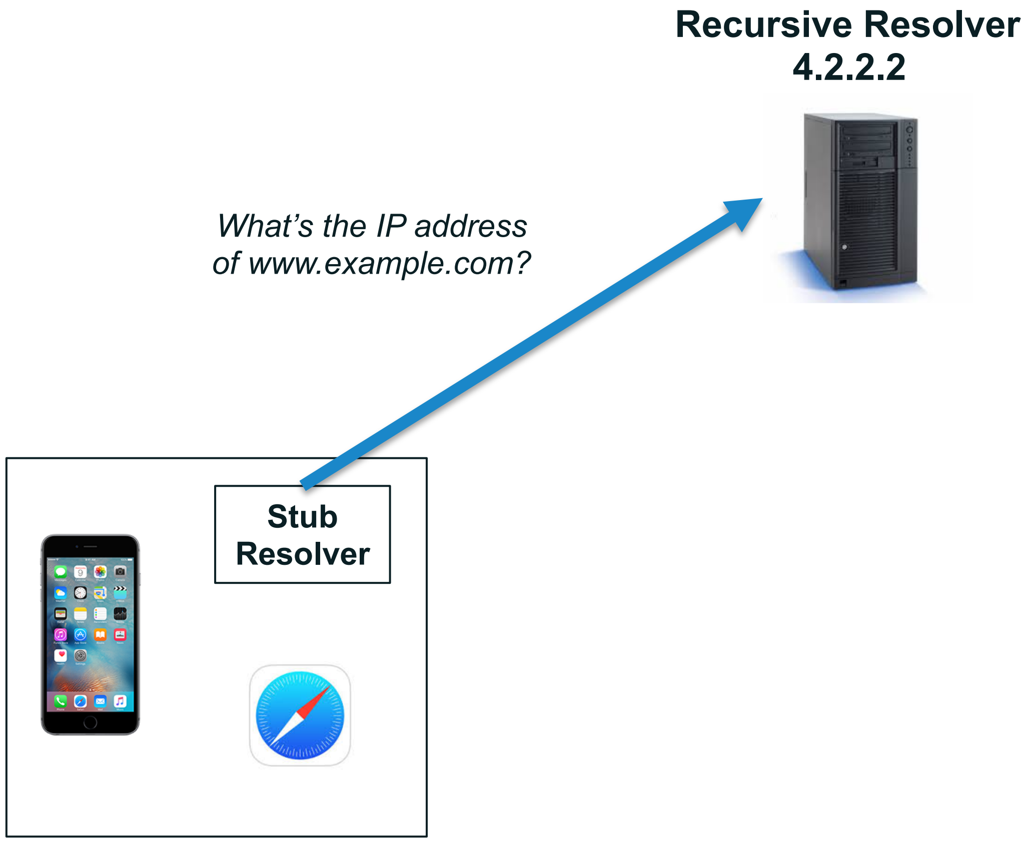
Resolution Process

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



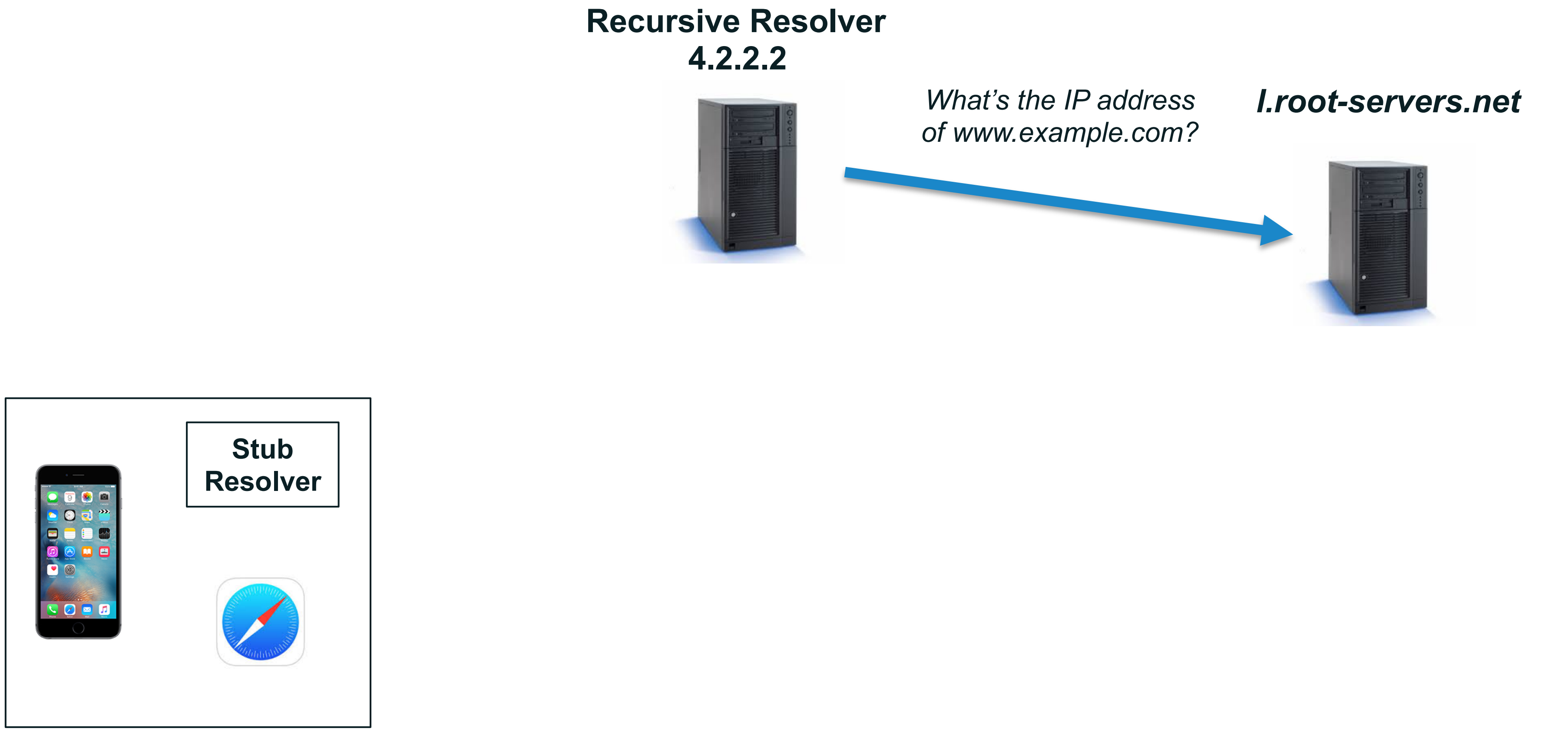
Resolution Process

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



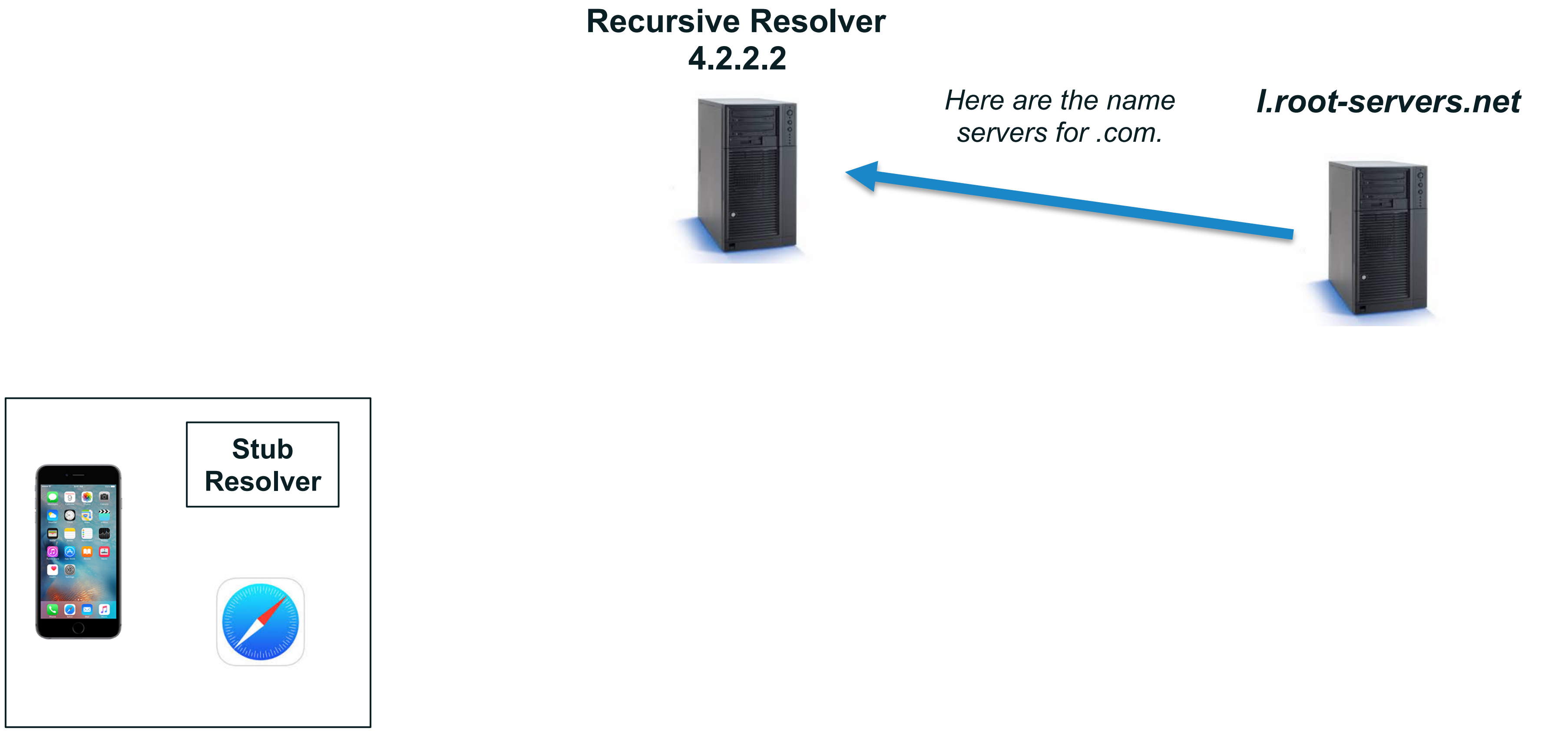
Resolution Process

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
Stub Resolver

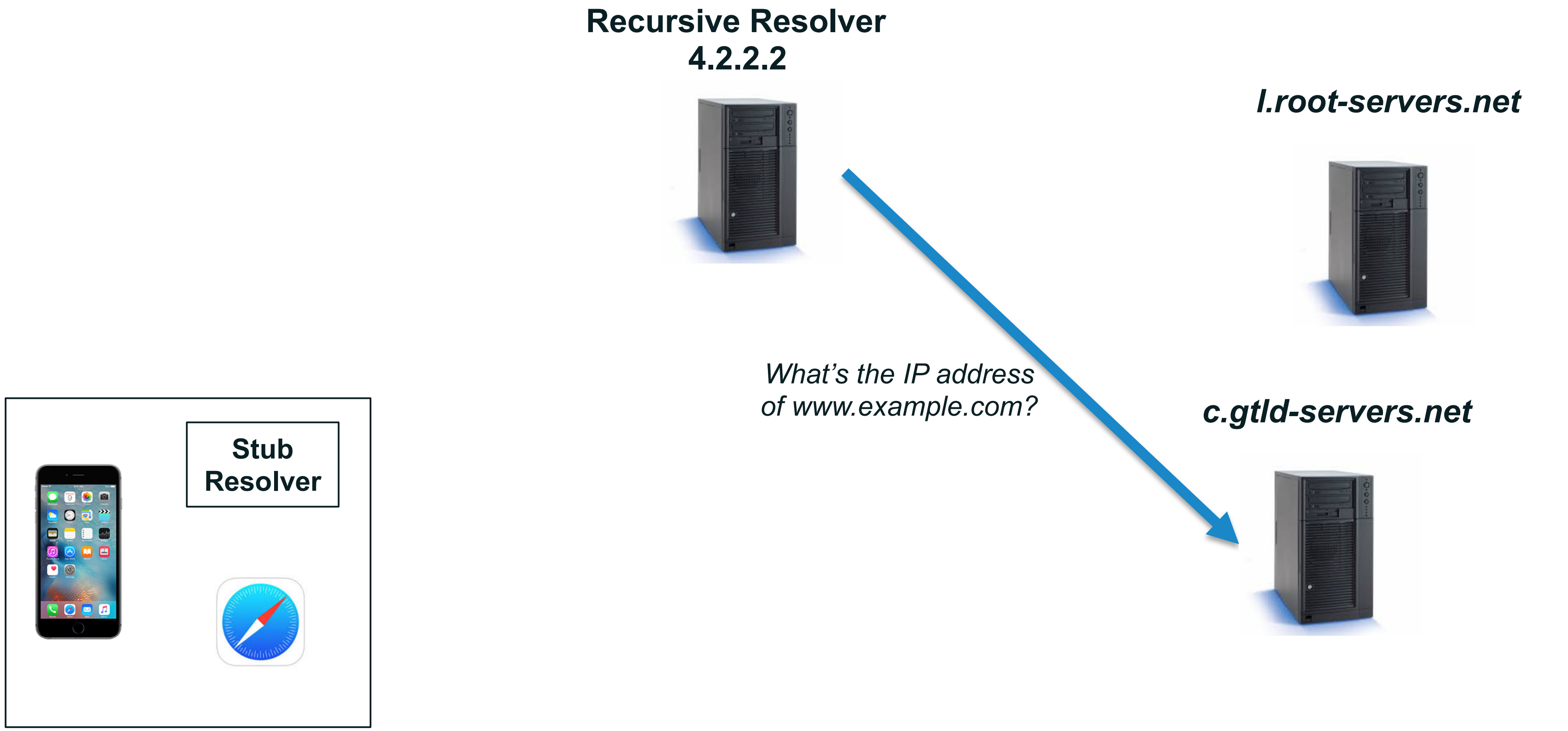
Resolution Process

Root server returns a referral to *.com*



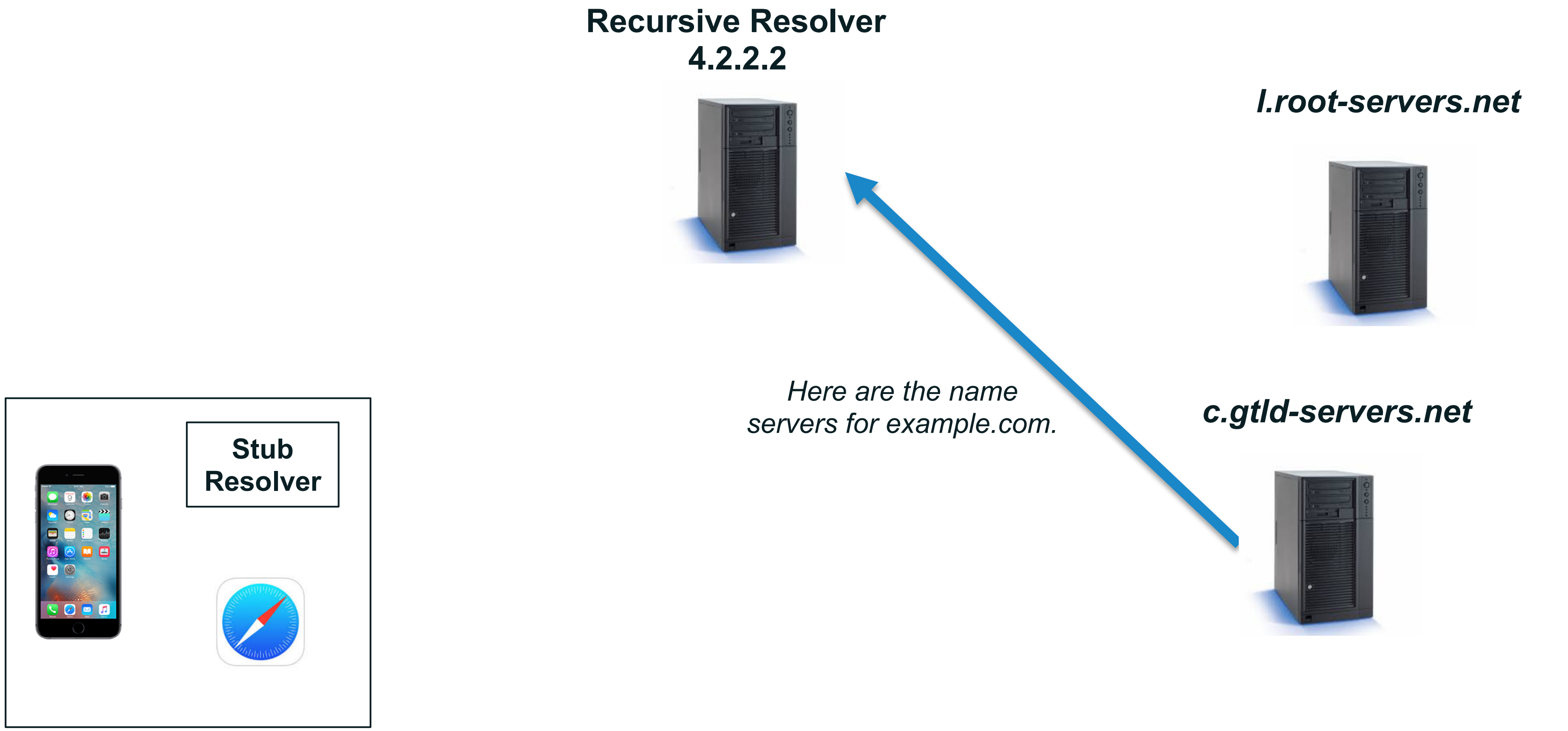
Resolution Process

Recursive resolver queries a *.com* server



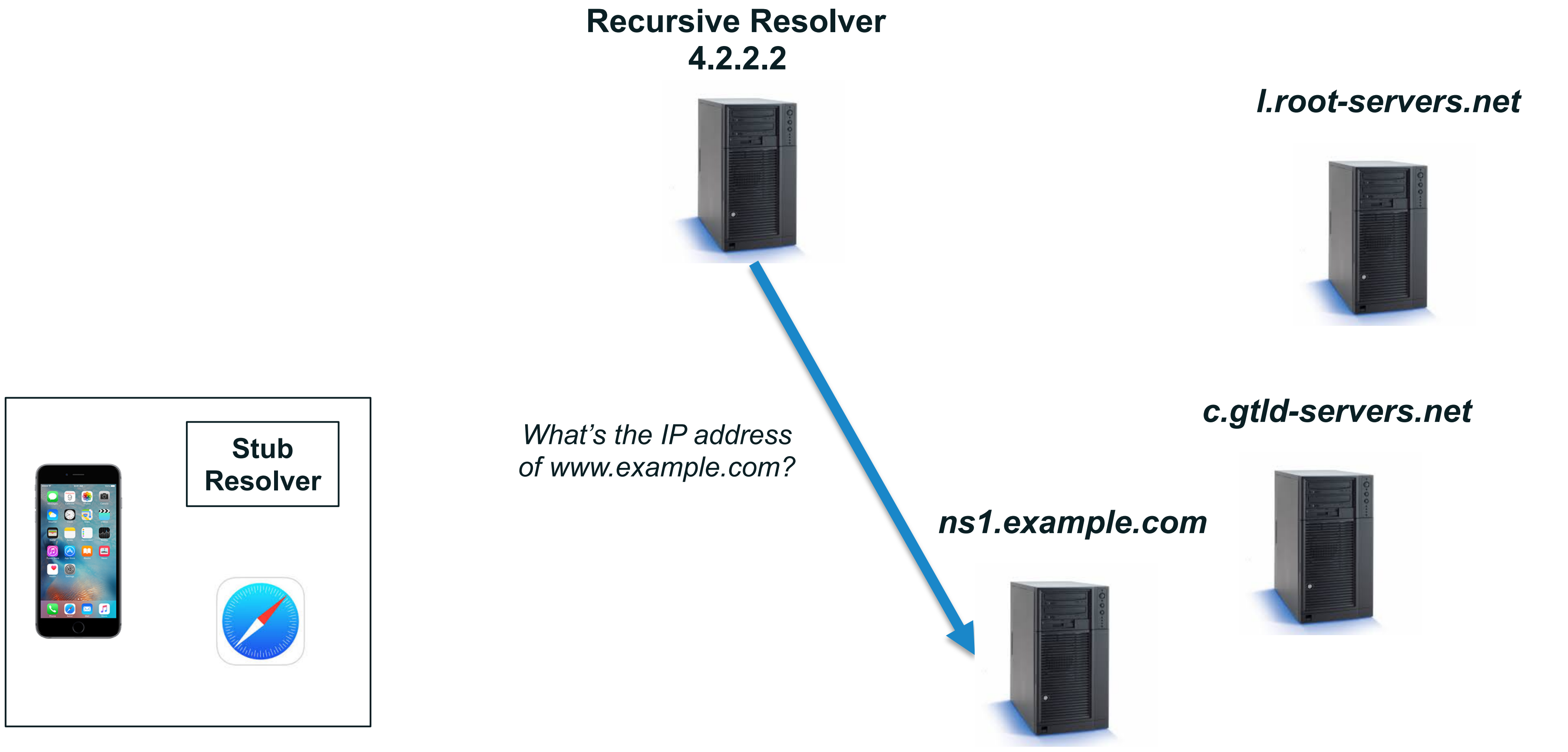
Resolution Process

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



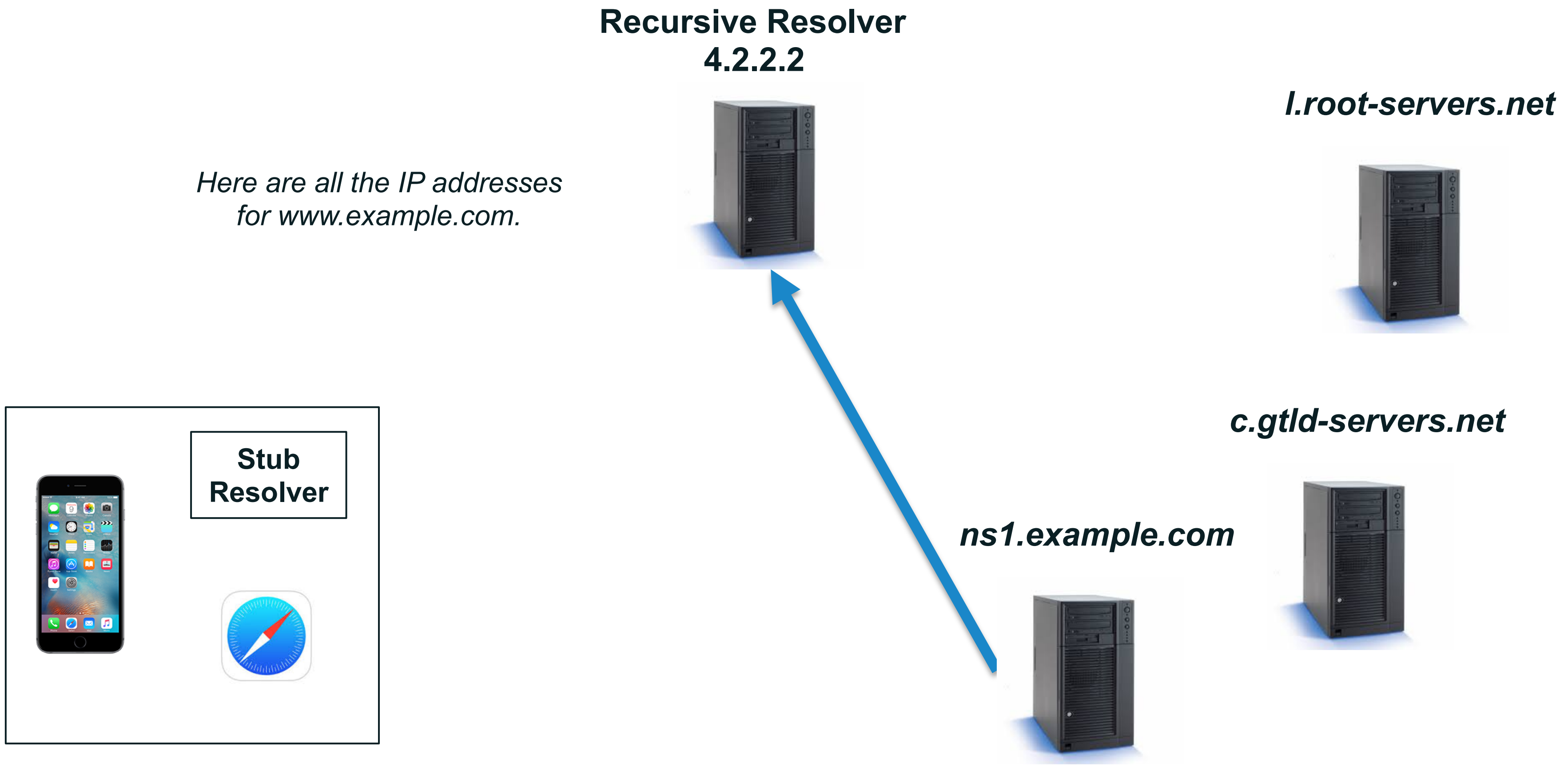
Resolution Process

Recursive resolver queries an *example.com* server



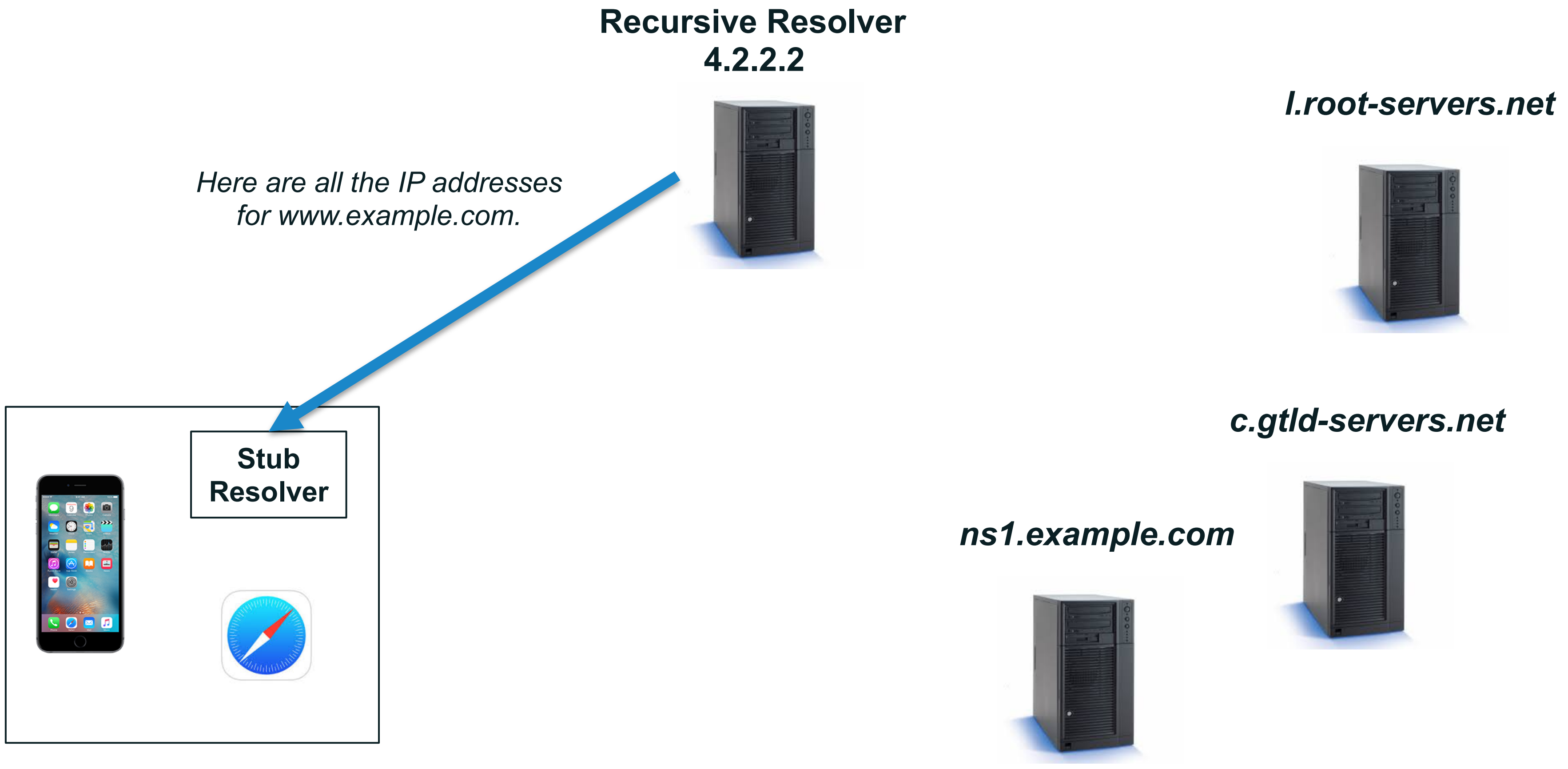
Resolution Process

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



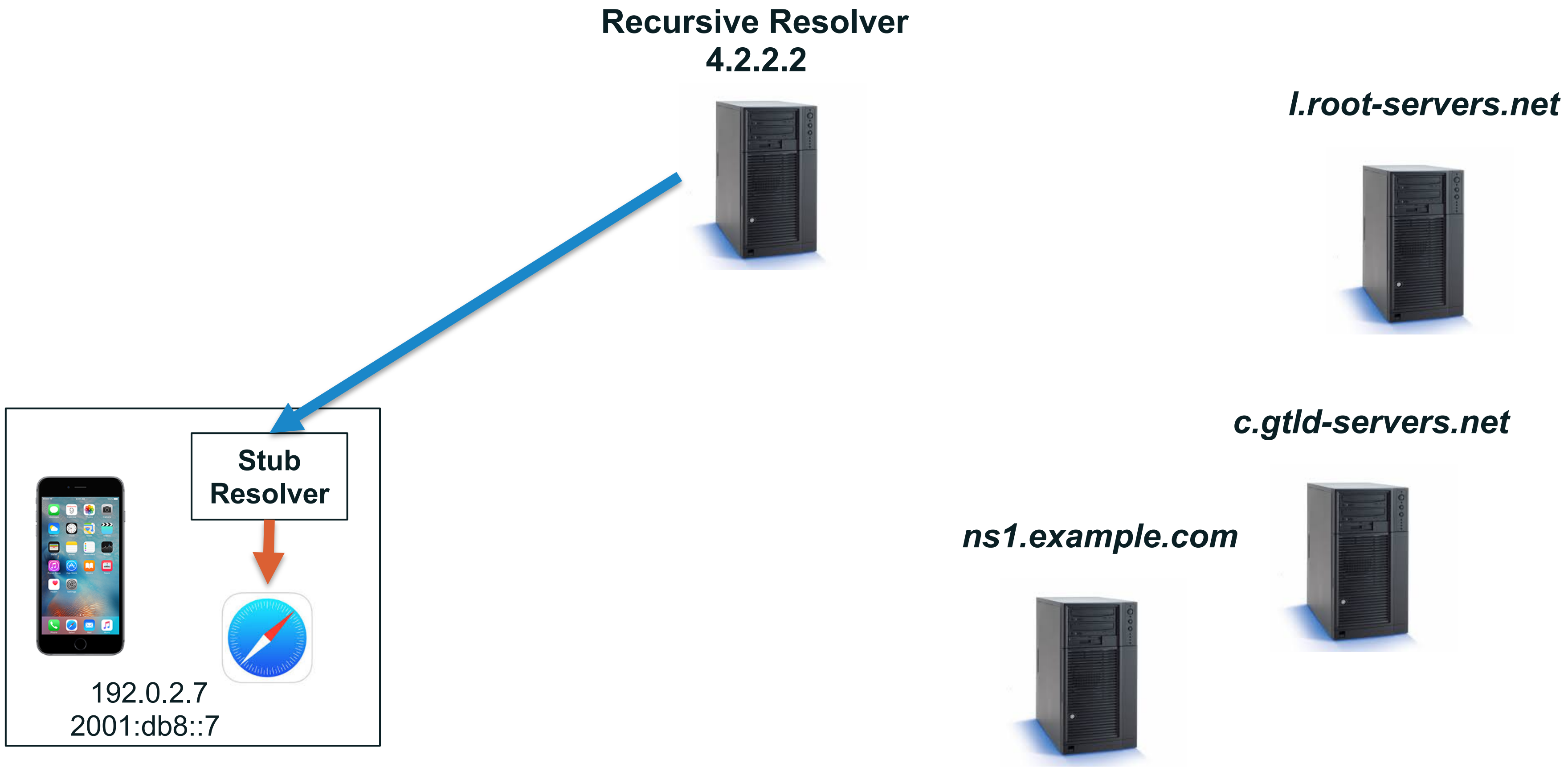
Resolution Process

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



Resolution Process

Stub resolver returns the IP addresses to Safari



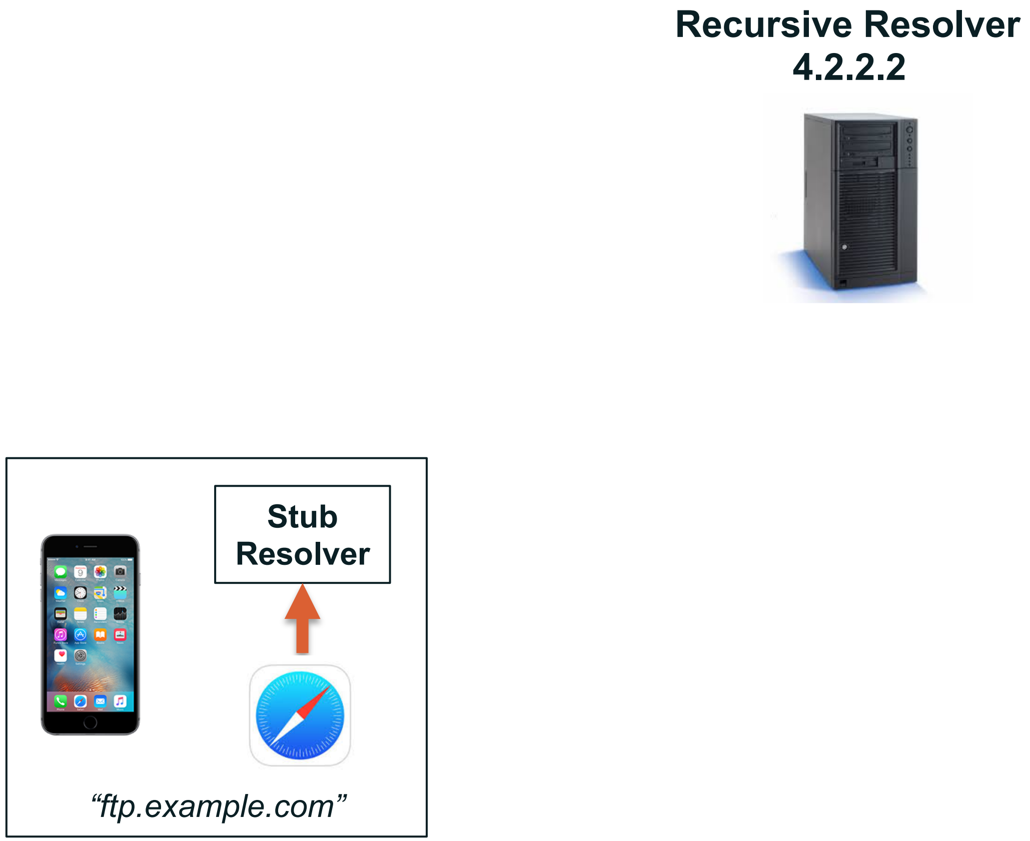
Resolution Process

- 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 . . .

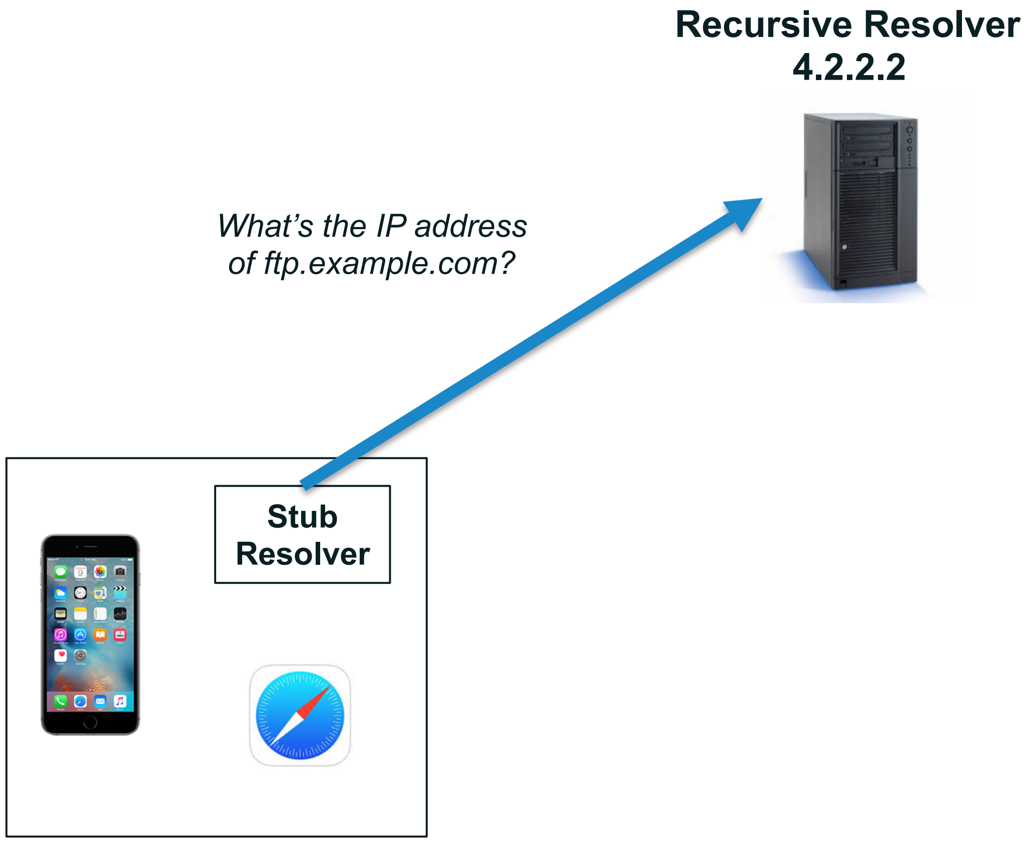
Resolution Process

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



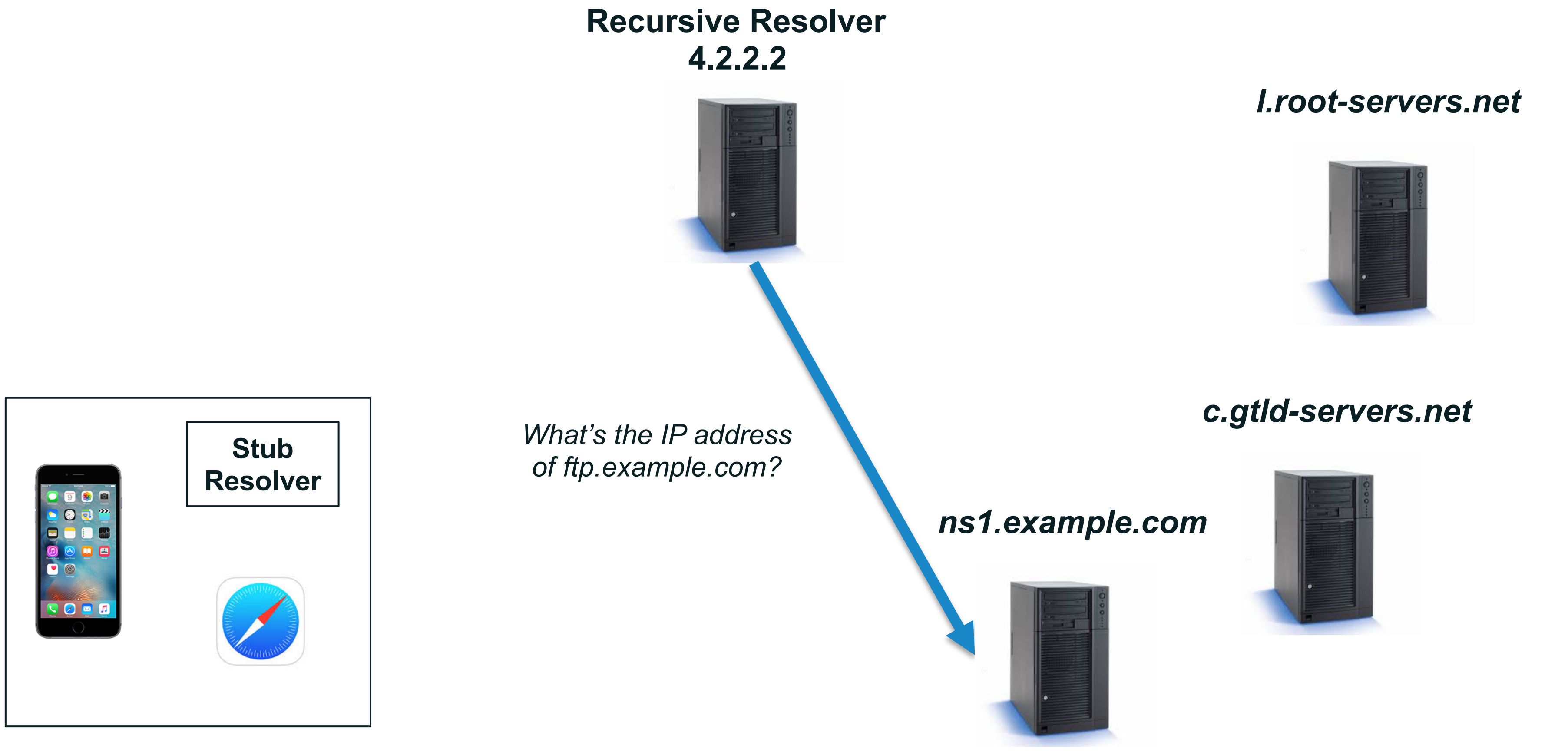
Resolution Process

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



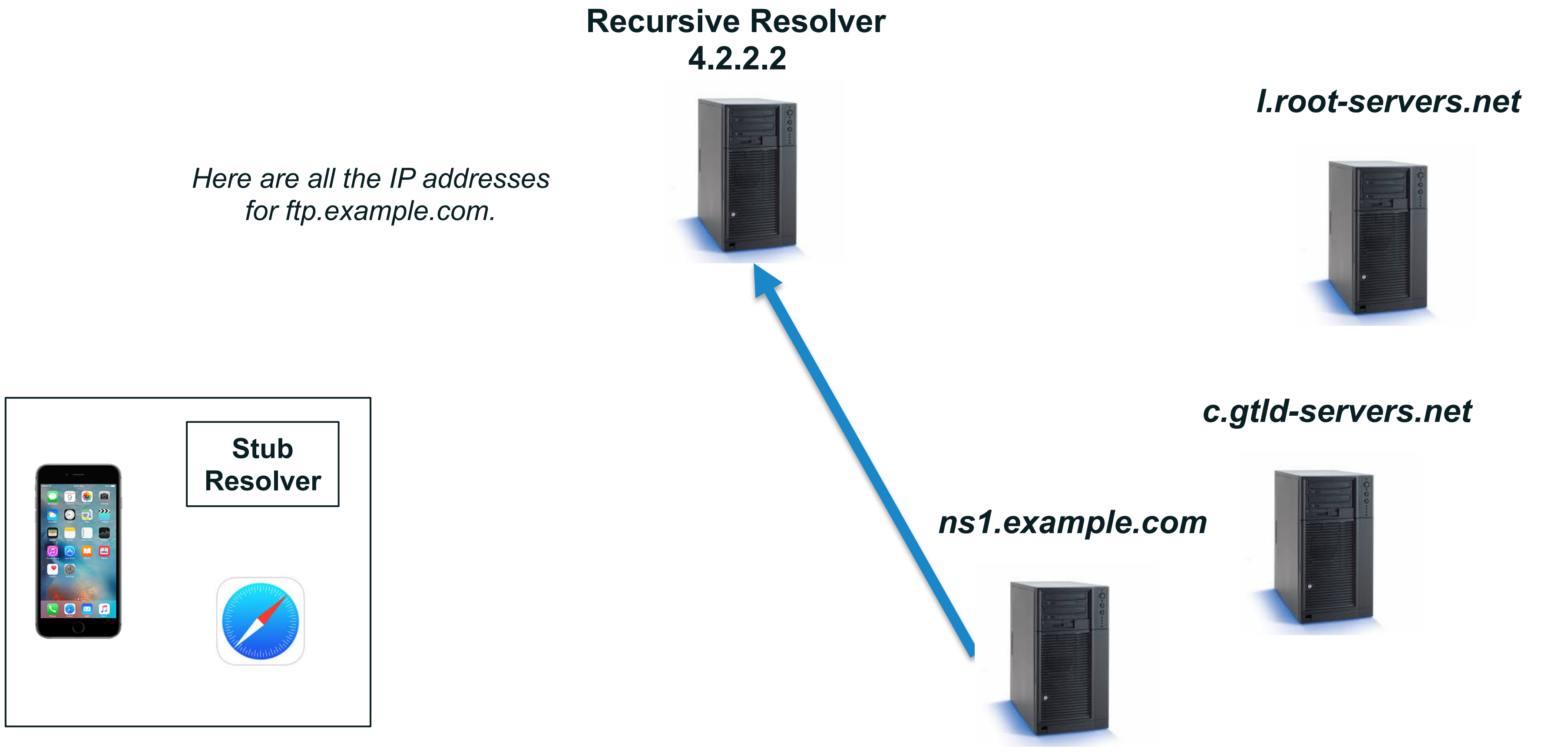
Resolution Process

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



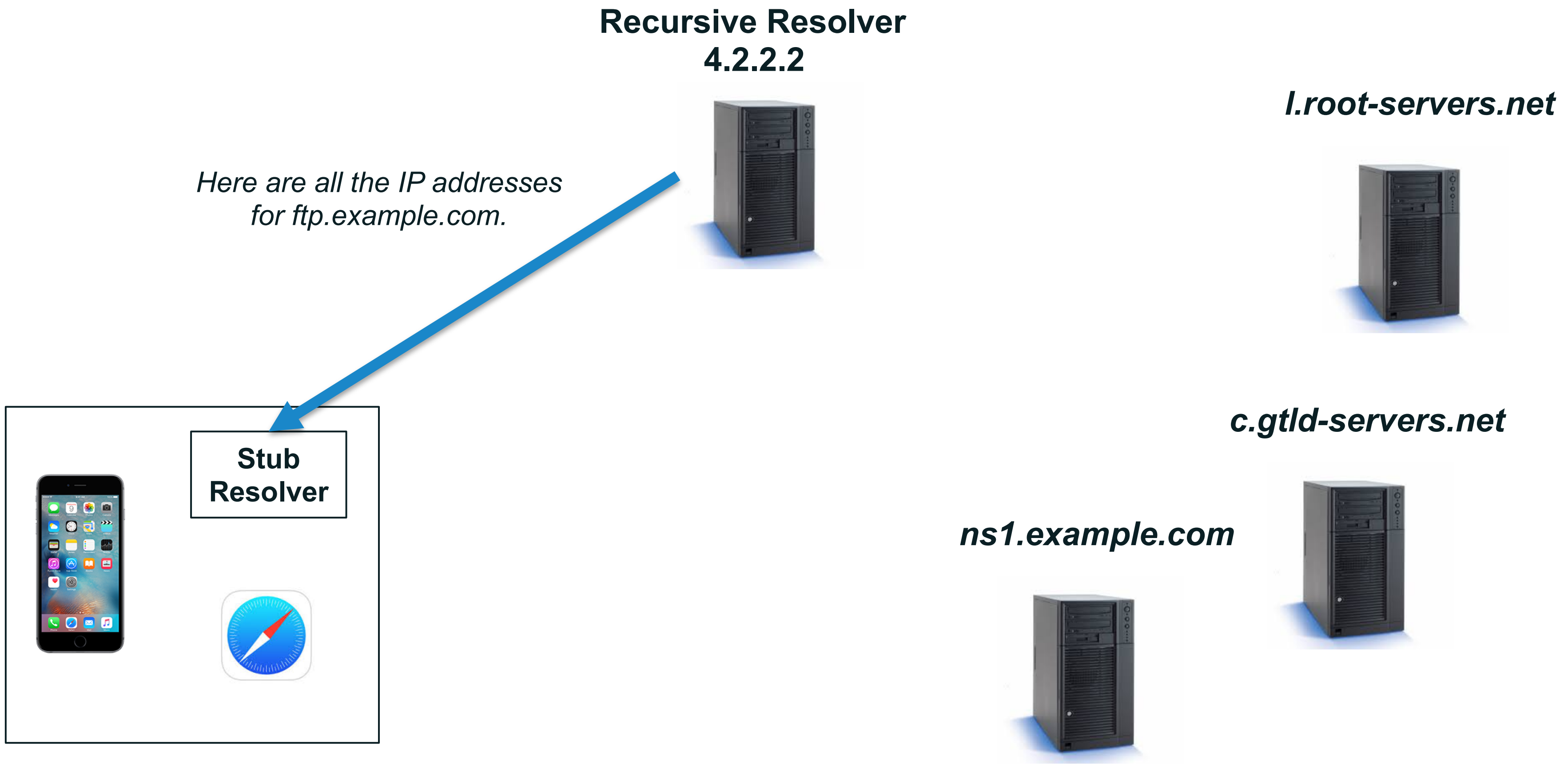
Resolution Process

example.com server returns the answer to the query



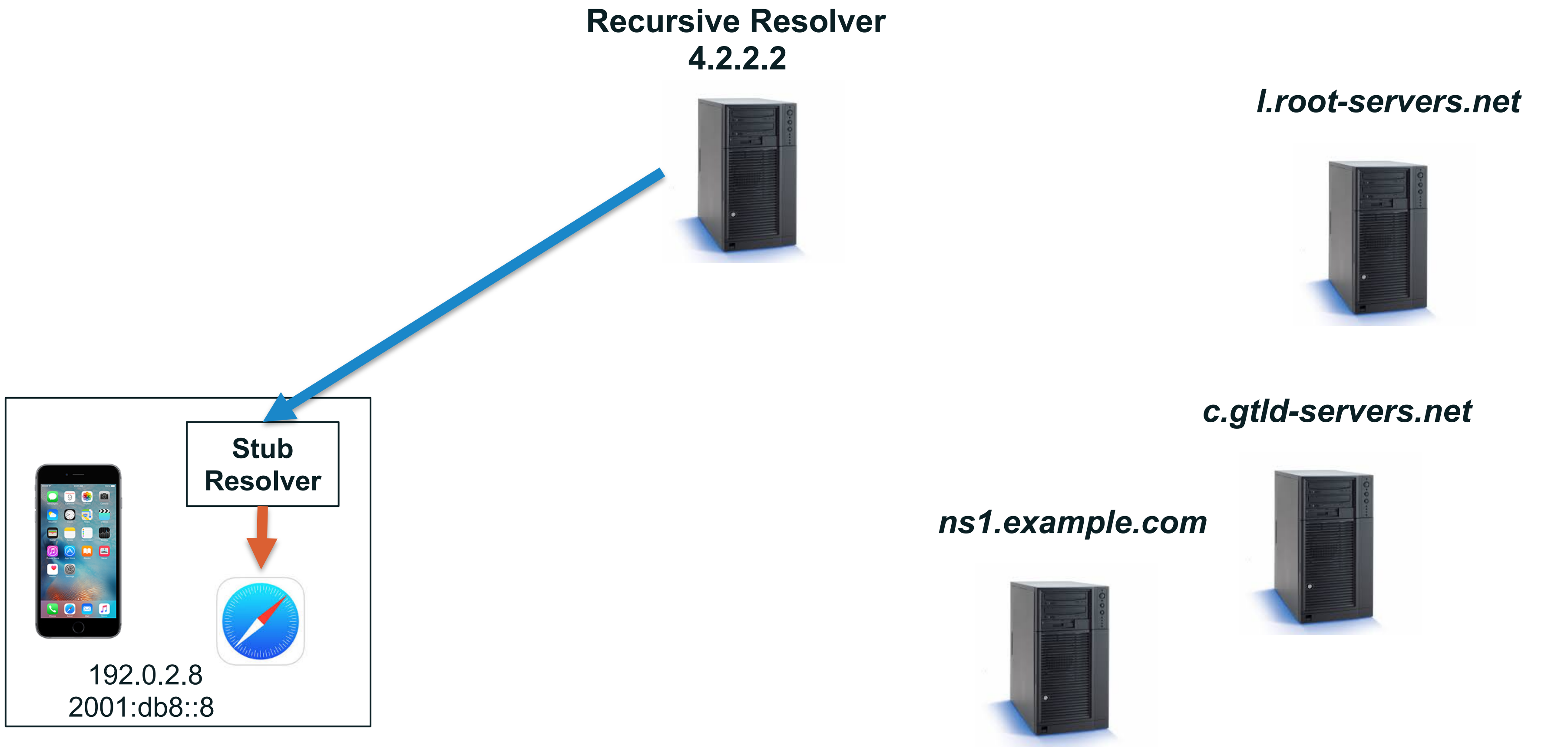
Resolution Process

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



Resolution Process

Stub resolver returns the IP addresses to Safari

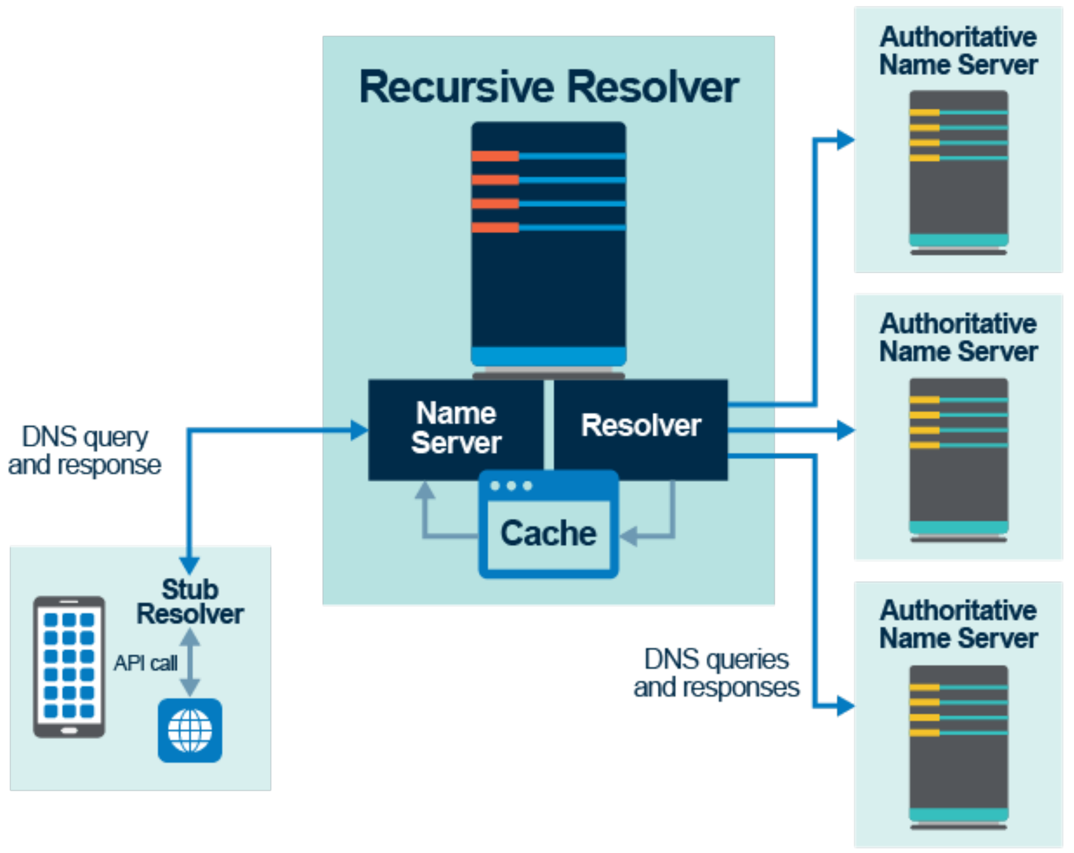


Recalling DNS

ICANN

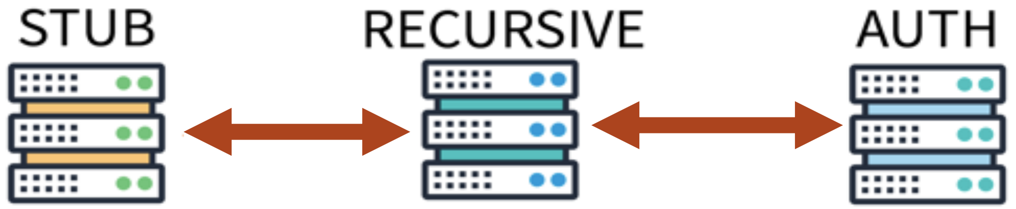
DNS

- 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.
- **Servers**
  - Authoritative servers.
  - Resolvers (Caching, etc).

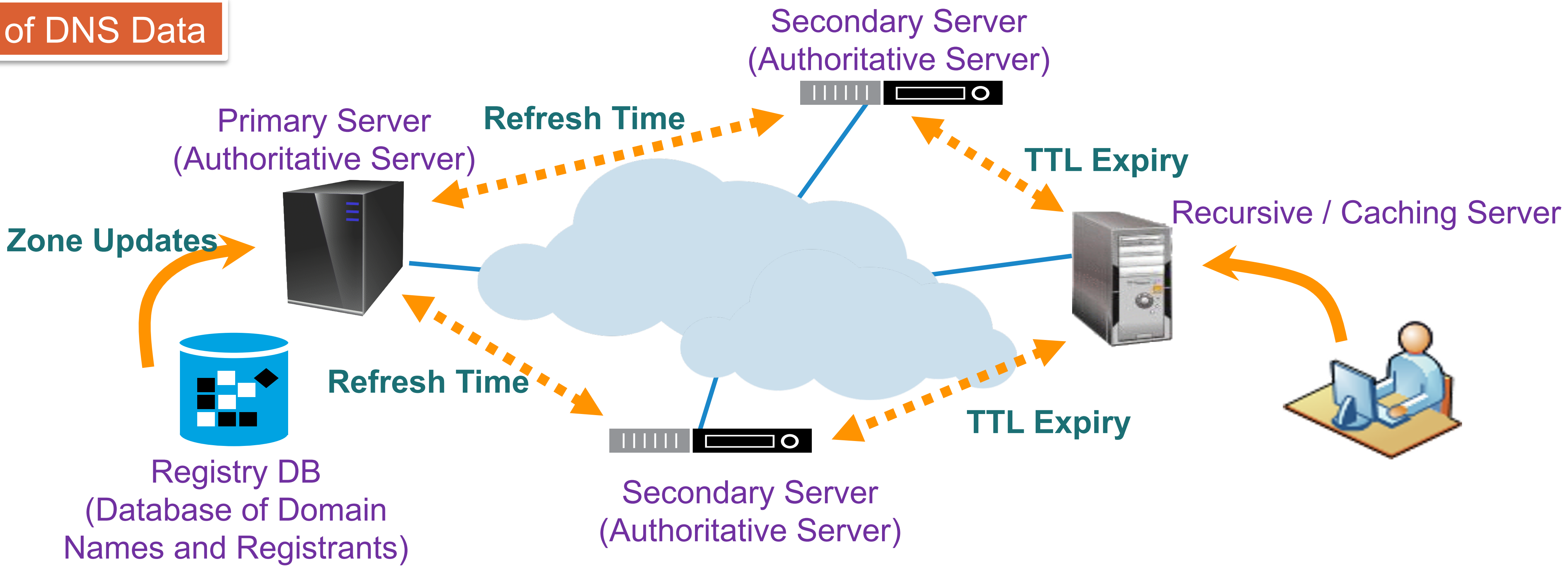

Recursive Resolver
Name Server
Resolver
Cache
DNS query and response
Stub Resolver
API call
Authoritative Name Server
Authoritative Name Server
Authoritative Name Server
DNS queries and responses

DNS Resolution's Traditional Model

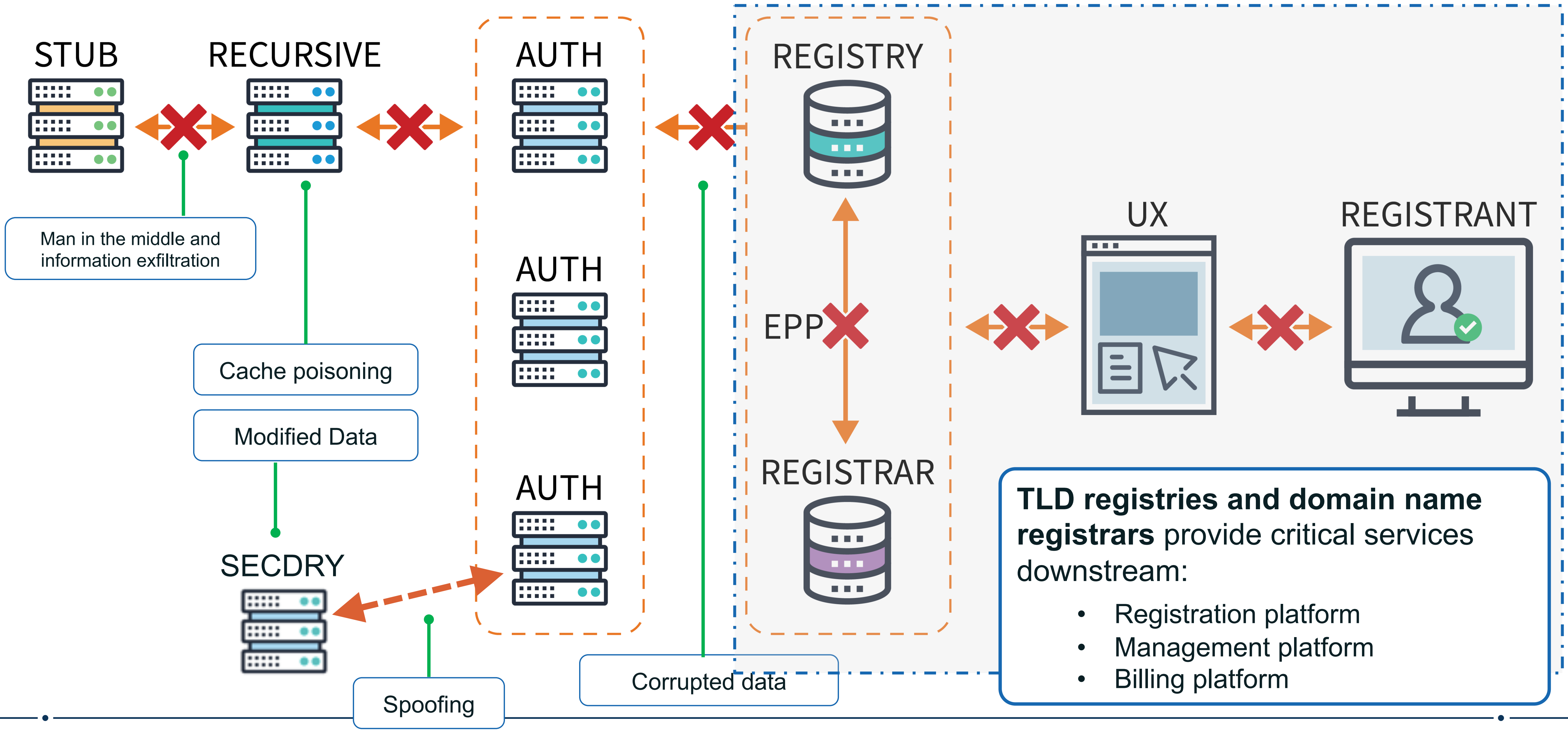
- Stub <-> Recursive <-> Authoritative



Propagation of DNS Data


Secondary Server (Authoritative Server)
Primary Server (Authoritative Server)
Refresh Time
TTL Expiry
Recursive / Caching Server
Zone Updates
Refresh Time
TTL Expiry
Registry DB (Database of Domain Names and Registrants)
Secondary Server (Authoritative Server)

Potential Target Points of the DNS Infrastructure/Ecosystem



TLD registries and domain name registrars provide critical services downstream:

- Registration platform
- Management platform
- Billing platform

DNS Resilience #1

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*** or ***slave*** server retrieves the zone data from another authoritative server via a ***zone transfer***
  - The server it retrieves from is called the ***master server***
- Zone transfer is initiated by the secondary
  - Secondary polls the master periodically to check for changes

Recalling Root Zone Administration

Root Zone Administration Screenshot

- Administration of the root zone is far from a trivial task
- Twelve organizations operate authoritative name servers for the root zone

The Root Servers Operators

A	Verisign
B	University of Southern California Information Sciences Institute
C	Cogent Communications, Inc.
D	University of Maryland
E	United States National Aeronautics and Space Administration (NASA) Ames Research Center
F	Information Systems Consortium (ISC)
G	United States Department of Defense (US DoD) Defense Information Systems Agency (DISA)
H	United States Army (Aberdeen Proving Ground)
I	Netnod Internet Exchange i Sverige
J	Verisign
K	Réseaux IP Européens Network Coordination Centre (RIPE NCC)
L	Internet Corporation For Assigned Names and Numbers (ICANN)
M	WIDE Project (Widely Integrated Distributed Environment)

DNS Resilience #2

About Anycast

Anycast could be defined as a combination of IP addressing and routing scheme, where:

- the same IP address is assigned to many destination devices; and
- the decision of which destination the packet will reach is decided by the network's routing mechanisms and metrics.

Anycast does not require any special configuration at the application level or at any client level. It is a process that is transparent to the client.

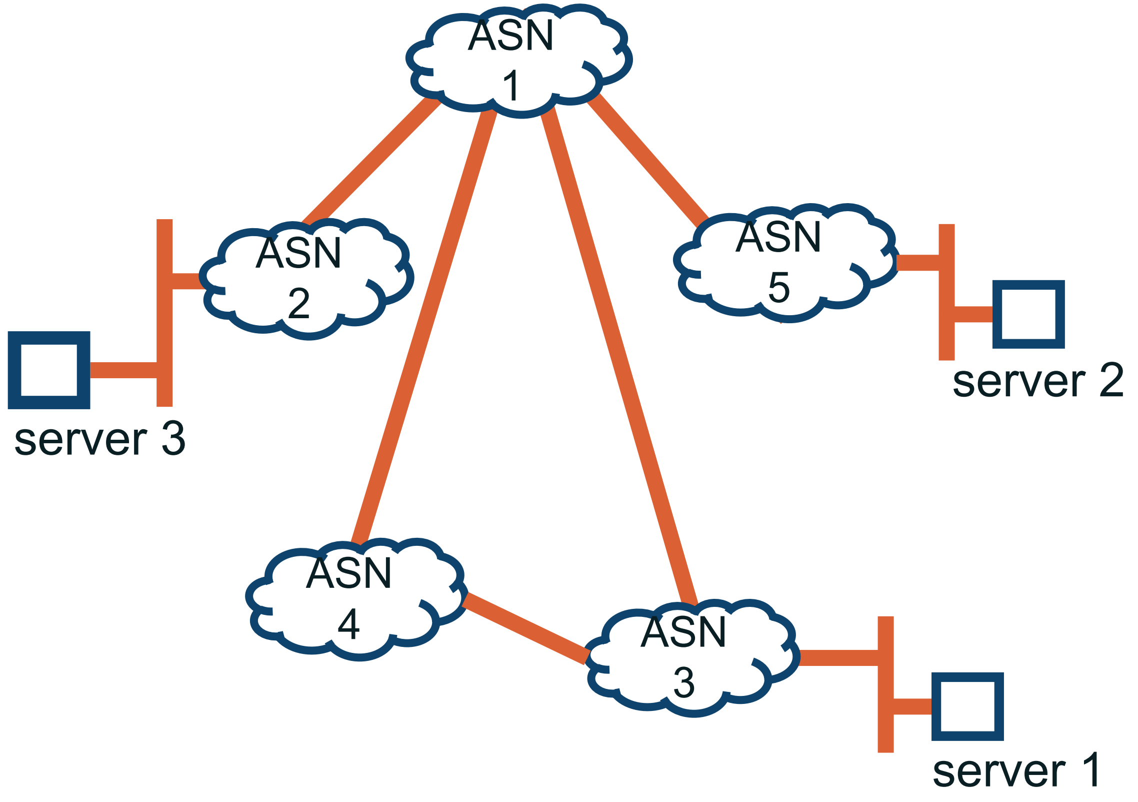
The goal is that packets will reach the *closest* anycast destination according to the routing metrics the network thinks is important (e.g., number of hops).

Anycast Use Cases (Internet)

Implementing anycast at the Internet level

Servers are configured with the same IP address but distributed in different places (different ASes) all over Internet.

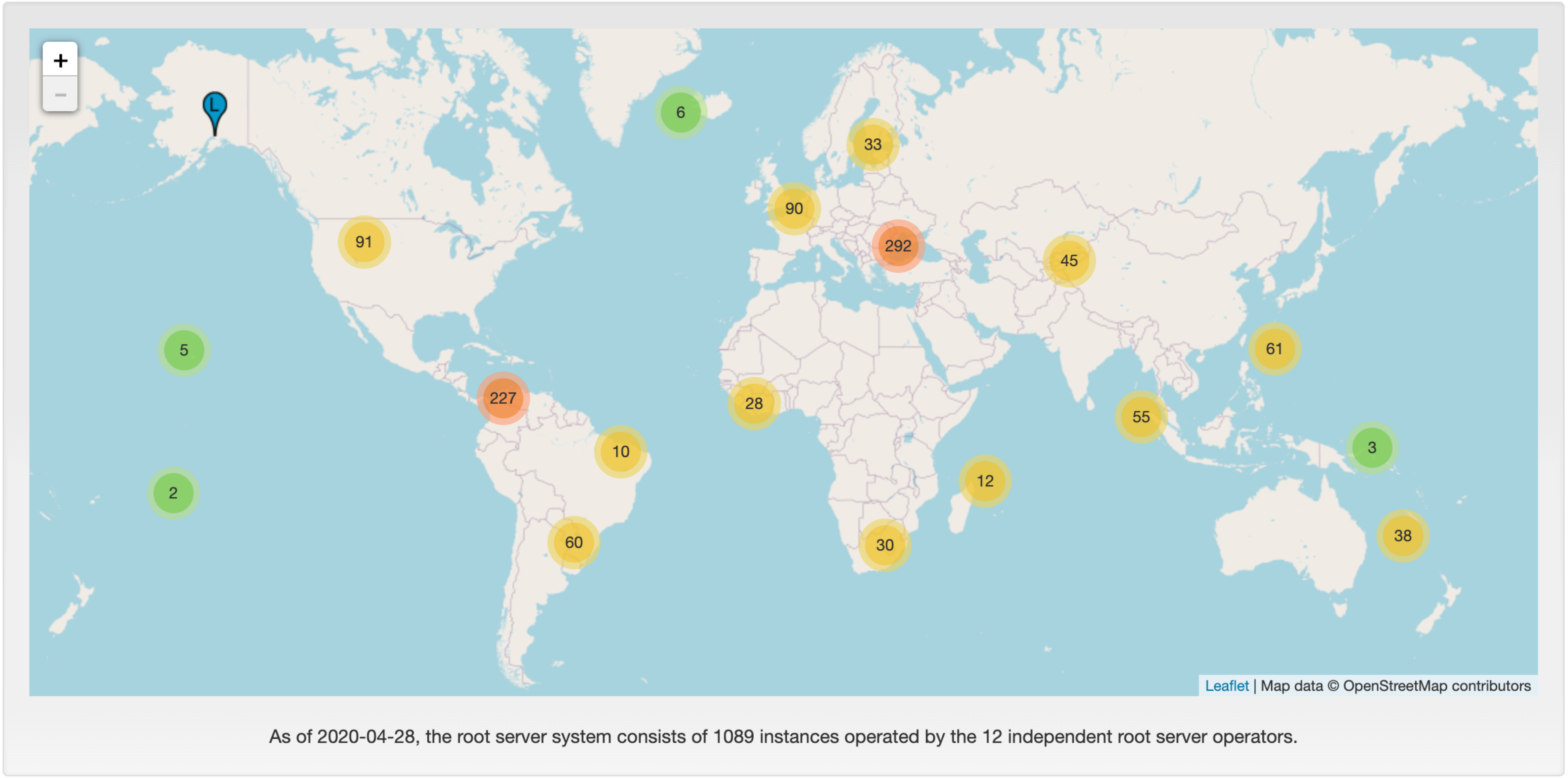
Packets sent by a client will reach one of the servers subject to different AS networking decisions. For instance they'll route packets to the closest server (shortest path to destination).



Anycast for DNS Servers

- Root server operators commonly employ anycast, distributing many **instances** of their root server label to servers all around the world.
- Anycast is also commonly used by recursive resolver operators, distributing many instances of their resolver all around the world.
- Anycast has many benefits for DNS resolvers:
  - Provides redundancy and resiliency to the global DNS infrastructure
  - Spreads the query and response load across many servers
  - Reduces latency by allowing for more instances closer to more clients
  - Provides more robustness, helping to mitigate events like DoS attacks on DNS infrastructure

The *root-servers.org* Web Site



As of 2020-04-28, the root server system consists of 1089 instances operated by the 12 independent root server operators.

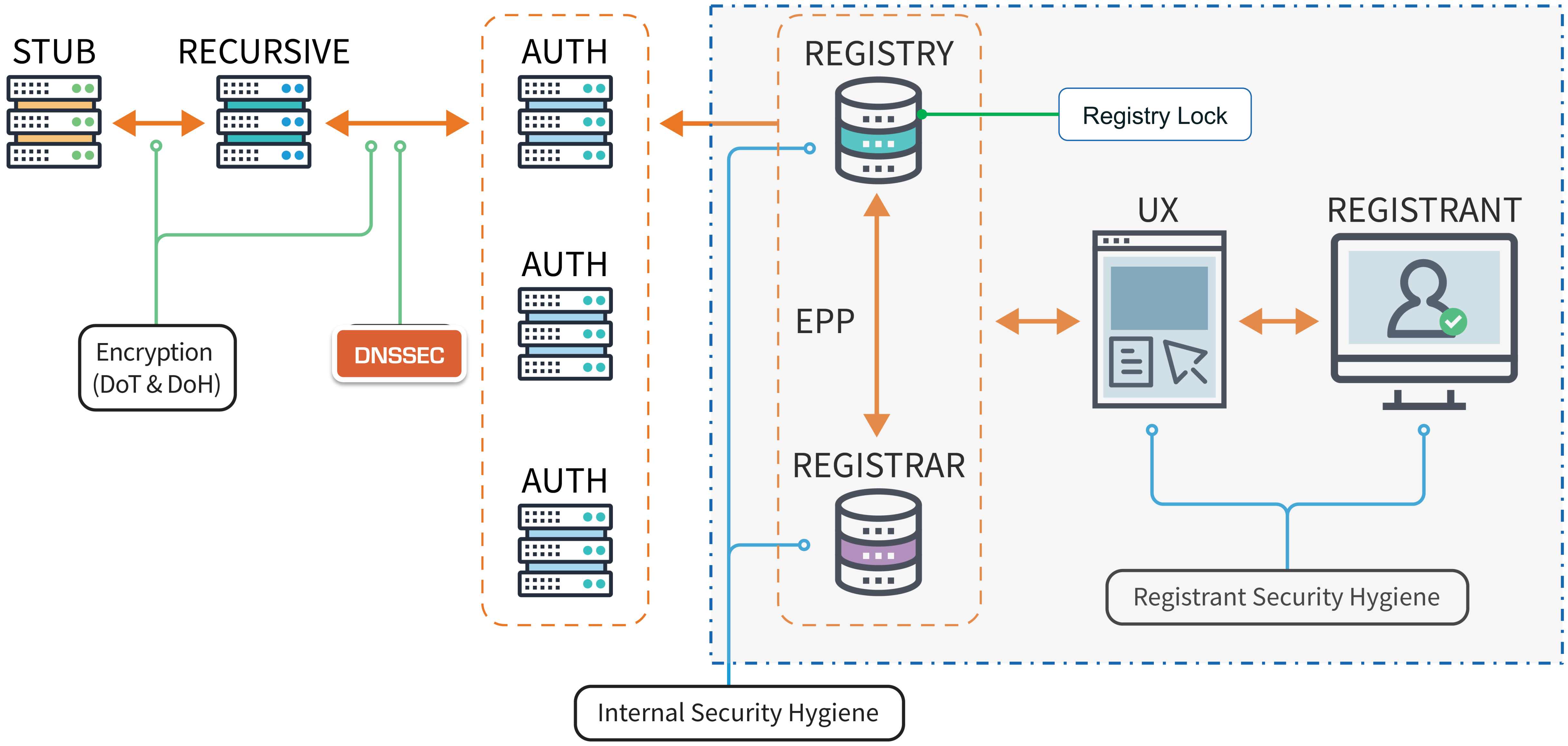
Security and Resiliency of the DNS

Attacks on Services Start by Targeting the DNS Ecosystem

Why attack the DNS?

- Collect data from web traffic to compromised domains
- Chanel to delivery malware
  - DNS is a common method of data exfiltration due to unfiltered port 53
- By meddling with DNS record values, someone can also obtain encryption certificates that is technically “valid” for an organization’s domain names.
  - *Once that is done they can redirected traffic to be decrypted, exposing any user-submitted data. Since the certificate is valid for the domain, end users receive no error warnings.*

Securing the DNS Ecosystem



What is the DNSSEC - Domain Name System Security Extensions

- Helps prevent DNS abuse, DNSSEC introduces cryptography that provides assurances to users that DNS data they are seeing is valid and true
- Allows domain name registrants to **SIGN** their DNS data
- Allows DNS operators **VALIDATE** all DNS data passing through DNS resolvers.

Authenticity: *Are we certain that the entity that publishes the data is authoritative?*
Integrity: *Are the data received the same as what was published?*

DNSSEC does not provide Authorization nor does it provide Confidentiality (privacy)

Benefits of DNSSEC

- **Technical Benefits**
  - Provide Origin authentication/validation
  - Integrity assurance for DNS data
  - *Authenticated denial of existence of DNS data*
- **Impact on players:** *Overall protect the directory lookup*
  - **End User** – Confidence of reaching intended website (complement to https)
  - **Registrant** – Fraud mitigation & greater brand (country code reputation) protection
  - **Registrar** – Comply with industry standards & meet registrant demands for increased security (attract and retain security & reputation-focused registrants)
  - **Registry** – Meet industry best practices & registrar demands for increased domain security

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