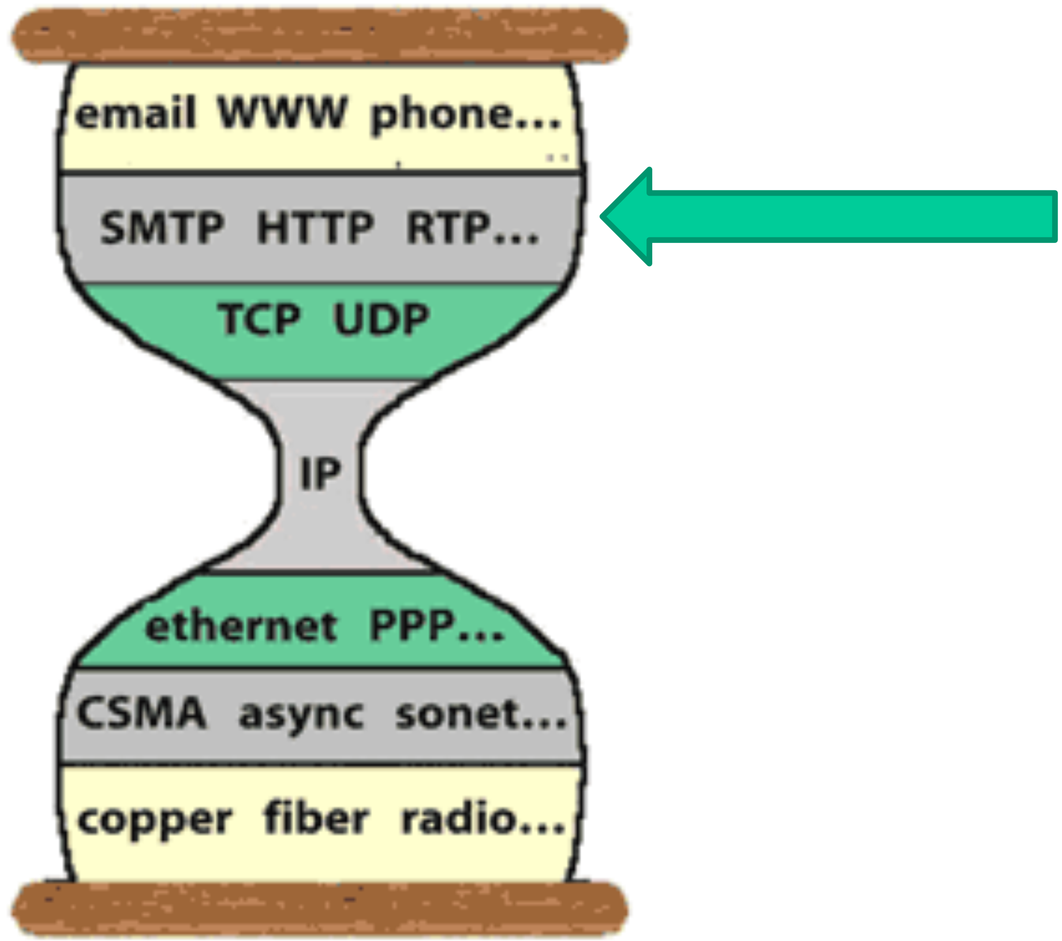
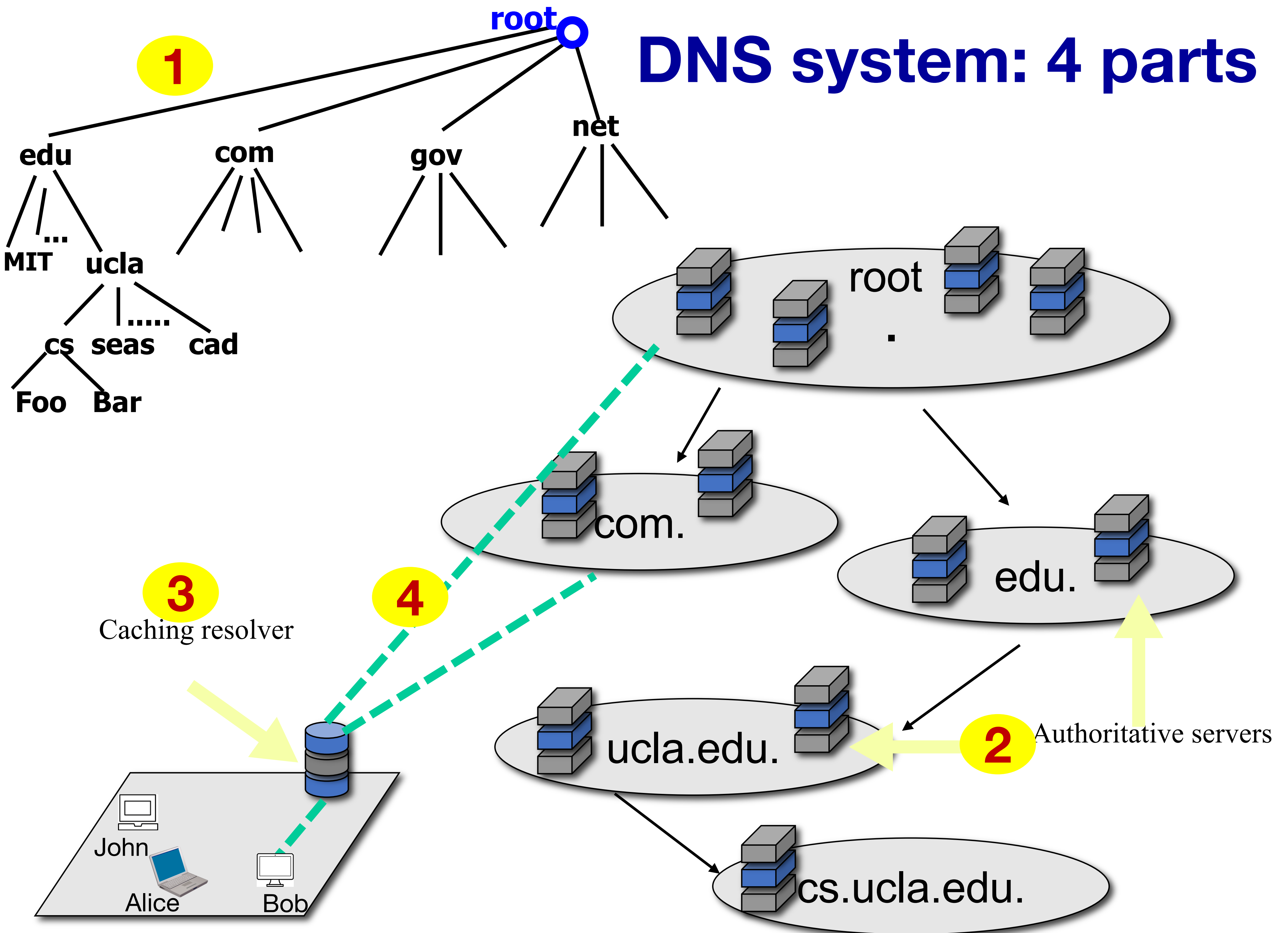
CS118 Lecture-5: Continue with DNS

- Brief intro to DNS protocol
- DNS performance and resiliency
- DNS and CDN
- DNS and network security



DNS system: 4 parts



DNS Namespace Governance

- Internet Corporation for Assigned Names and Numbers (ICANN, https://www.icann.org/) oversees the management of
  - Assignment of Top Level Domains (TLDs)
  - Delegation of TLD managements
  - Operation of the root *name servers*
- TLD operators
  - Running TLD name servers
  - allocate 2nd level domain names
    - e.g.: `edu` allocates the name `ucla.edu` to UCLA
- 2nd level domain owners assign 3rd level names
  - `ucla.edu` allocates `cs.ucla.edu` to the CS dept

FYI

Commercial example: Verisign

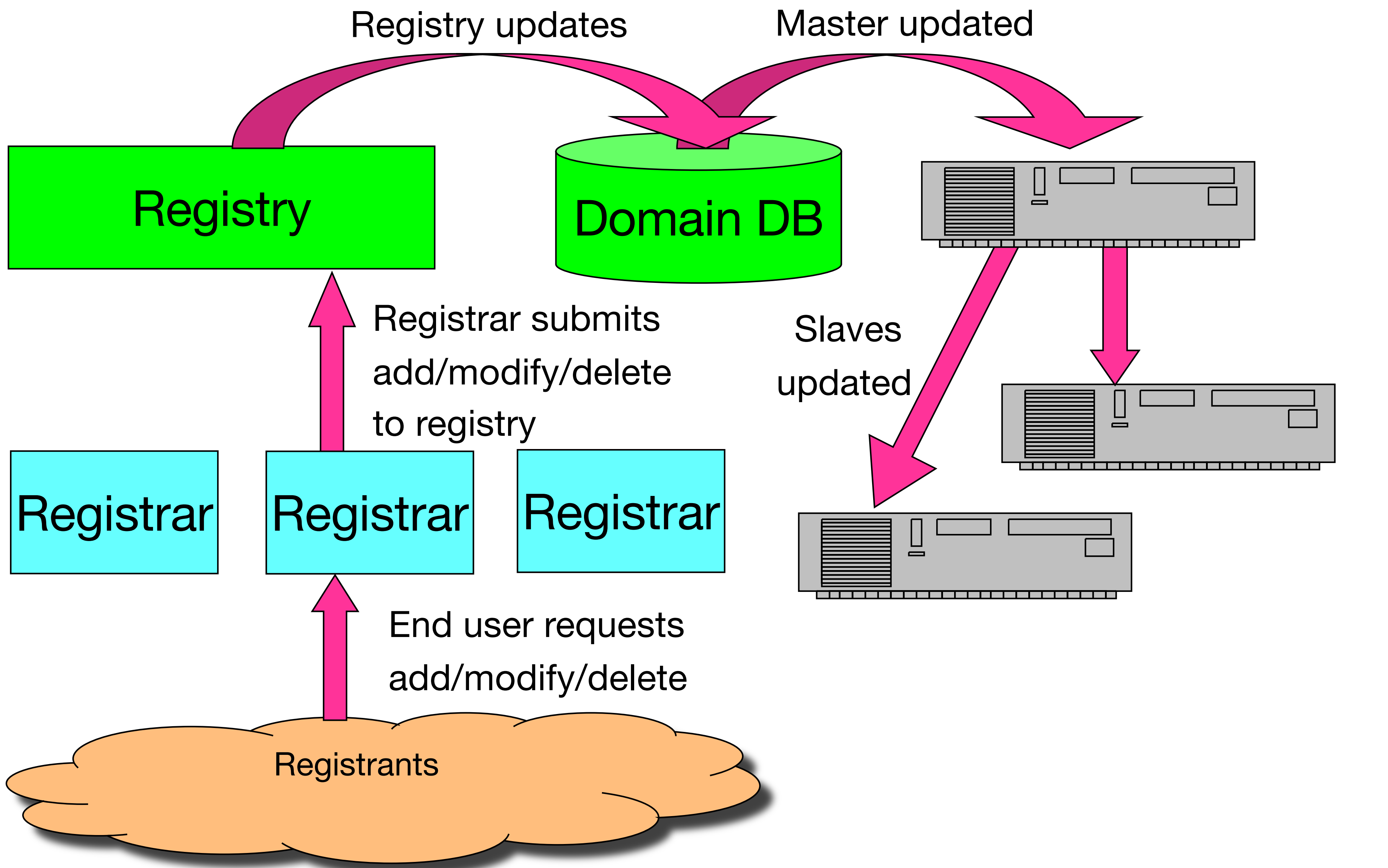
- ICANN delegates the management of .com to Verisign
- Verisign operates *authoritative name servers* for .com domain
- Verisign contracts registrars to sell domain names to public
  - Example registrars
    - GoDaddy (US)
    - CoolOcean (India)

Nowadays cloud providers also join the market…

Amazon, Cloudflare…

- There exist a *very* large number of registrars

Registries, registrars, registrants



Registries, registrars, registrants

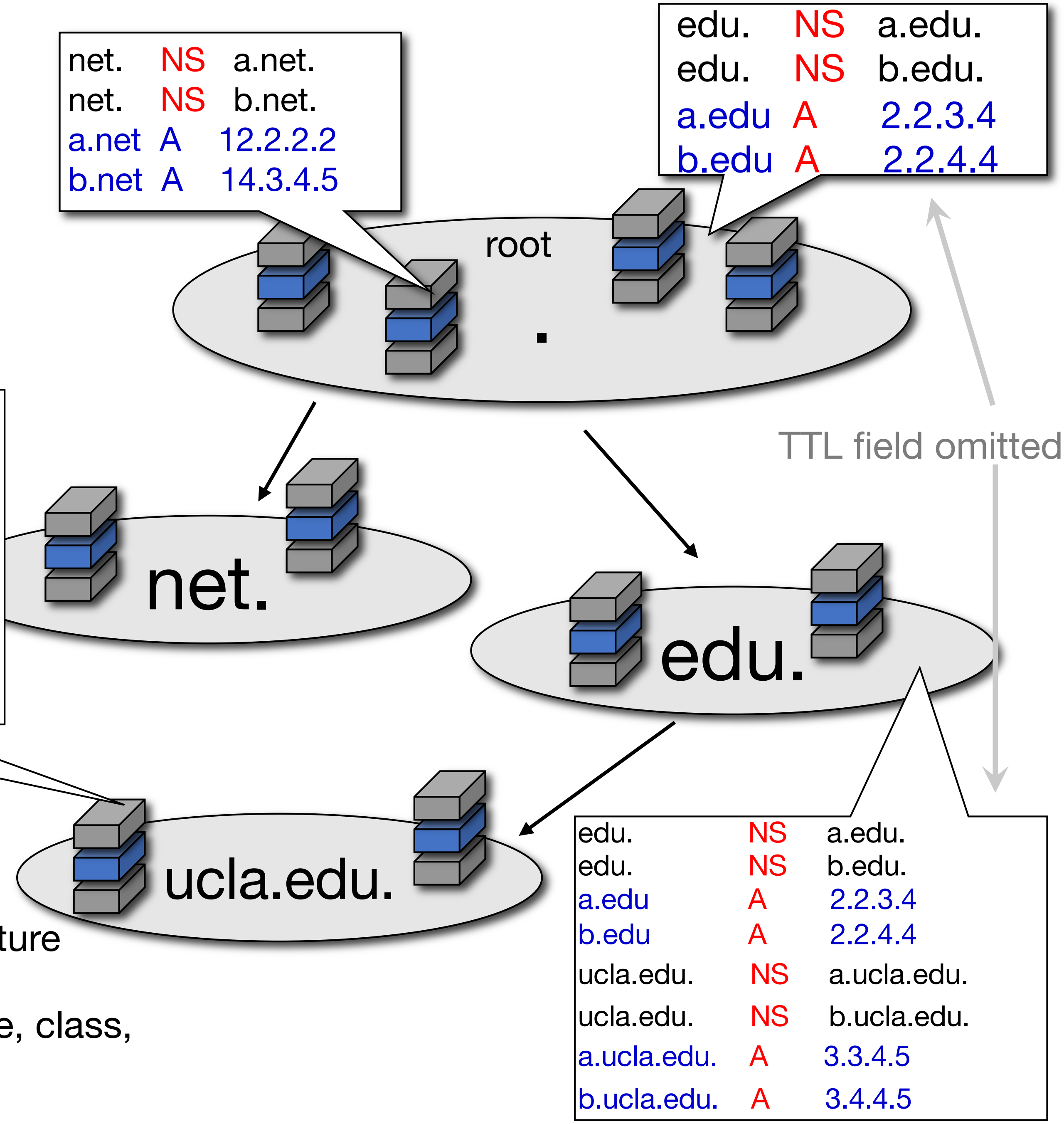
FYI

- Registry (e.g., Public Interest Registry)
  - An organization that manages a DNS namespace
    - Allocate names, or work with a registrar for name allocations
    - Run TLD name servers
- Registrar (e.g., Cloudflare)
  - An organization that sells domain names to the public
  - Submits change requests to the registry on behalf of the registrant
- Registrant (e.g., cs118.org -- us!)
  - Person or company who registers a domain name
  - A registrant can manage its domain name's settings through its own registrar.

Glue together DNS authoritative servers

Each NS RR of zone Z and the corresponding *glue RR* is stored in both Z's own and its parent's zone files

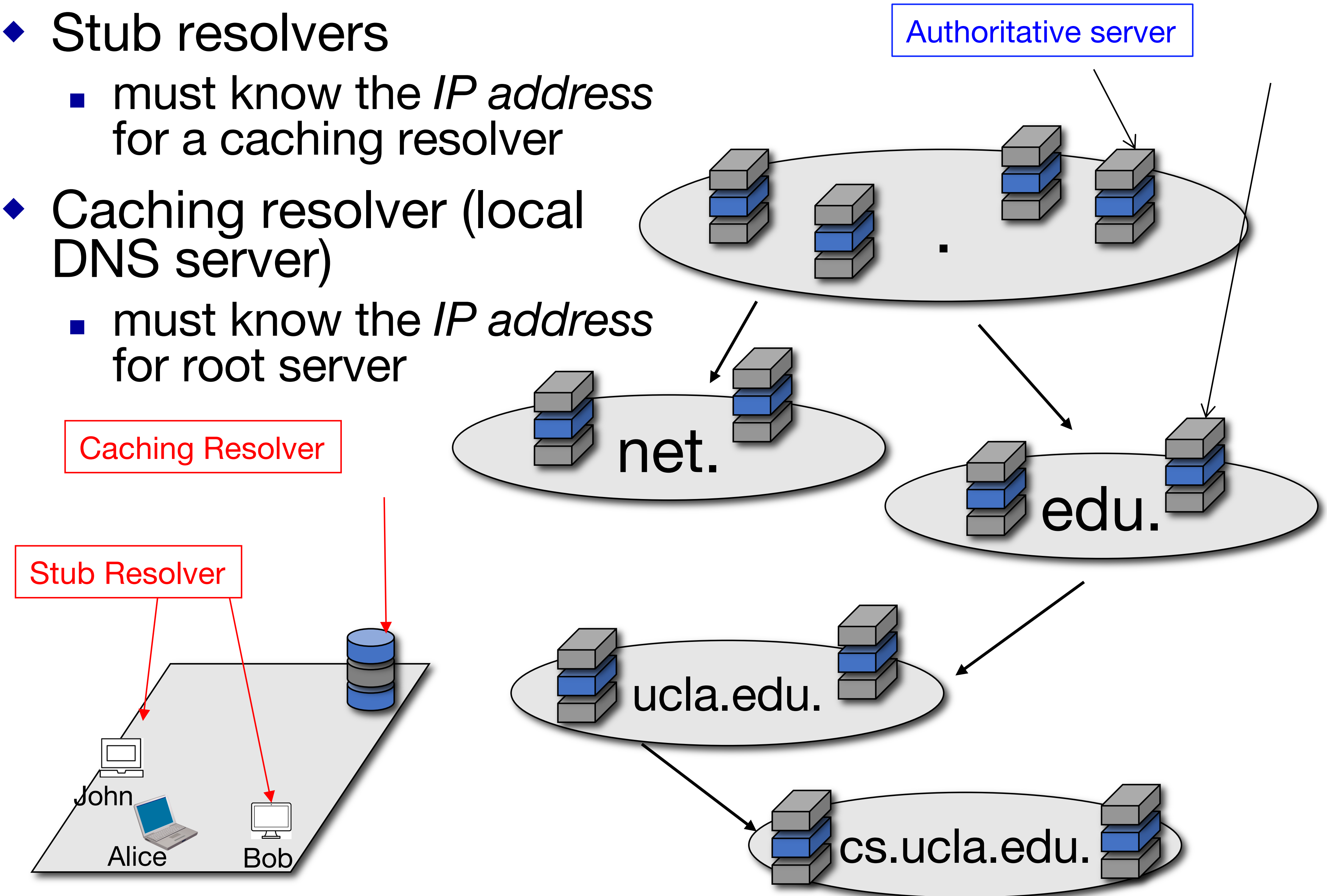
NAME	TYPE	TTL	VALUE
ucla.edu	NS	824	a.ucla.edu
ucla.edu	NS	824	b.ucla.edu
a.ucla.edu	A	600	3.3.4.5
b.ucla.edu	A	900	3.4.4.5
www.ucla.edu	A	1700	3.2.2.2
mail.ucla.edu	A	3100	3.3.3.3
….			



- All DNS data stored in a data structure called "*resource record*" (RR)
- An RR contains 5 fields: name, type, class, TTL, value

Bootstrapping DNS lookup

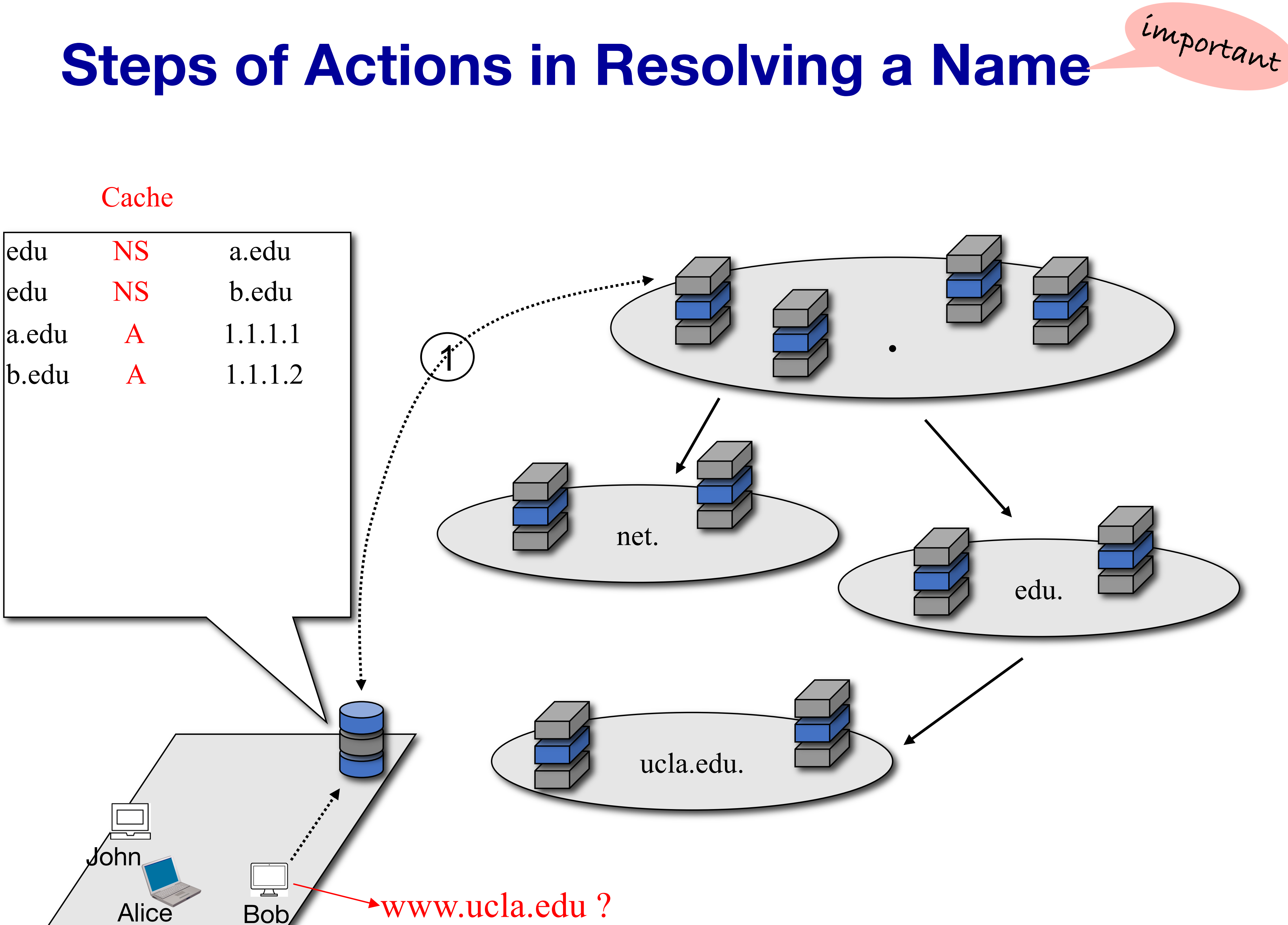
- Stub resolvers
  - must know the *IP address* for a caching resolver
- Caching resolver (local DNS server)
  - must know the *IP address* for root server



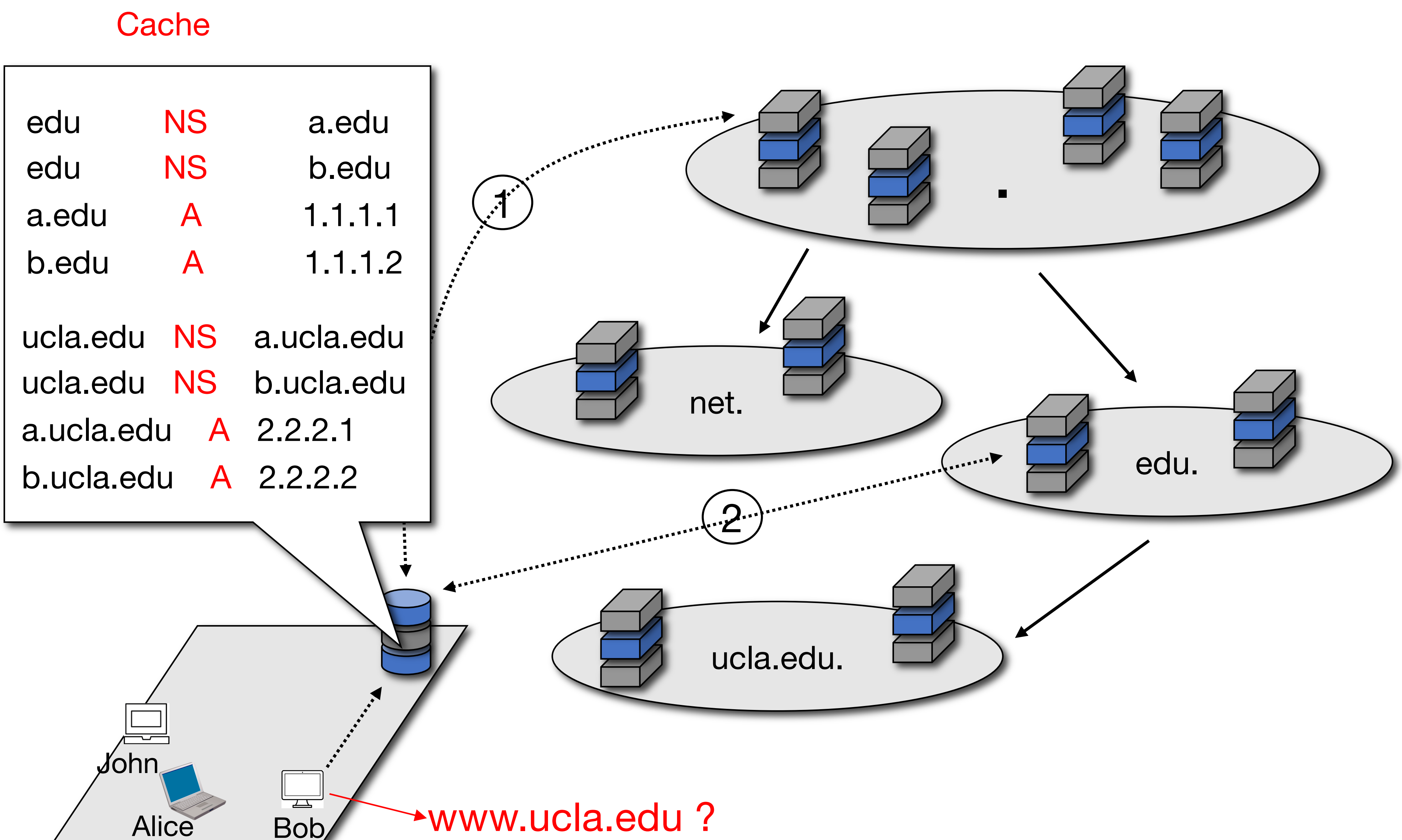
DNS Resolution

- Whenever an app needs to communicates: first call DNS to translate the name to IP address, then open socket with the destination address
    - System call `getaddrinfo(), gethostbyname()`
- Stub resolver
    - Configured with the IP address of the caching resolver(s)
    - Send DNS queries to local caching resolvers
- Caching resolver (local DNS server)
    - Has the *IP address* of root servers, hard-coded in
    - Query authoritative servers, cache the data from replies

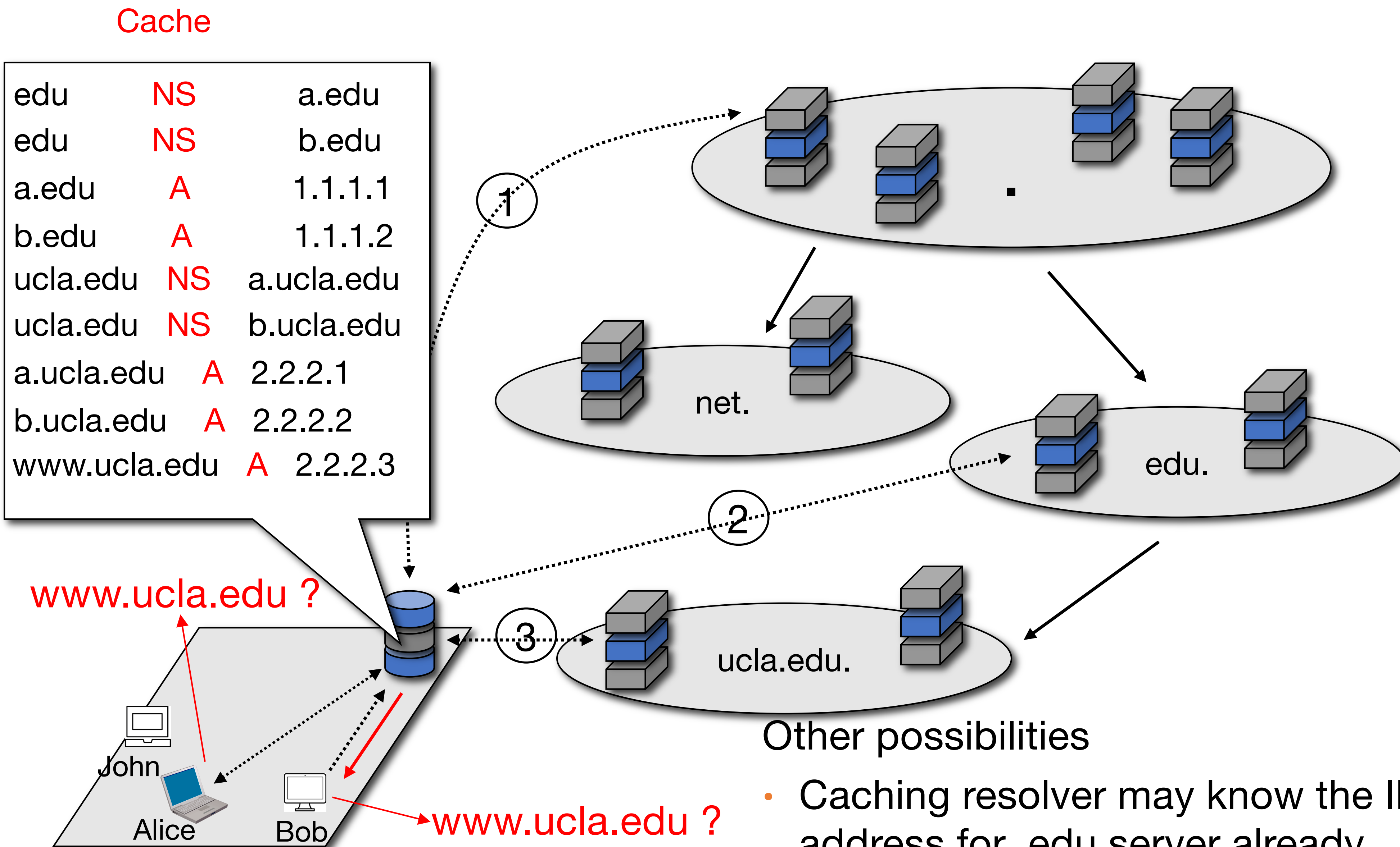
Steps of Actions in Resolving a Name



Steps of Actions in Resolving a Name



Steps of Actions in Resolving a Name



Other possibilities

- Caching resolver may know the IP address for .edu server already

Summary: How a DNS name gets resolved?

1. A user host sends a query for `www.ucla.edu` (asking for its IP address) to a local DNS caching resolver
   - provided by your ISP
     - In recent years: provided by Google (8.8.8.8), CloudFlare (1.1.1.1), etc
2. The caching resolver either finds a *relevant* answer in its cache,
   - any of the following are relevant to `www.ucla.edu`
     - An exact match: `www.ucla.edu`'s IP address
     - ucla.edu DNS server IP address: go to step-5
     - .edu DNS server IP address: go to step-4

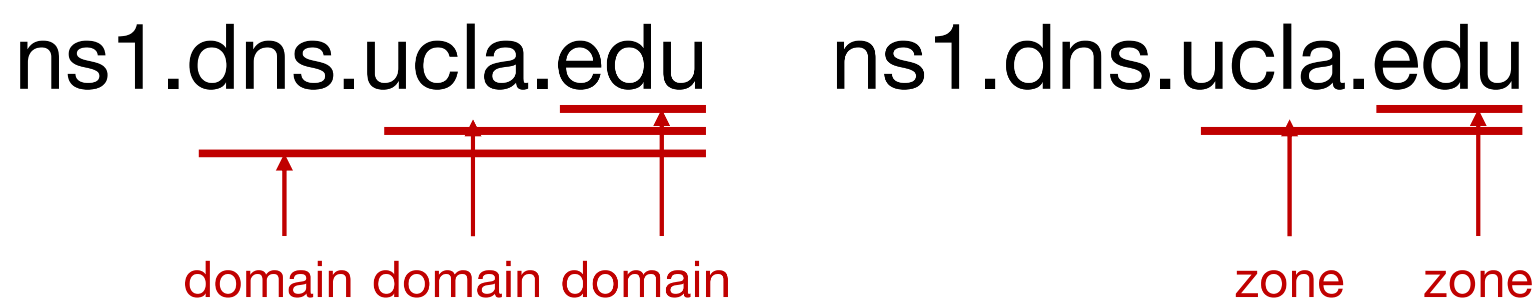
   otherwise sends the query to one of the root servers
3. The root server replies with pointers to `.edu` servers
4. The caching resolver queries `.edu` DNS server, which replies with pointers to `ucla.edu` DNS servers
5. The caching resolver queries `ucla.edu` DNS server to get the IP address for `www.ucla.edu`, and sends the answer back to user host

Namespace Allocation vs. Delegation

- Administrator of a domain can *allocate names* under its own **namespace** to other organizations or individuals, creating a subdomain
- The administrator can *delegate* the **management** responsibility of a subdomain, creating an administration unit called *zone*
  - The parent and subdomain zones can now be administered independently
- Delegation: *the key to DNS system's scalability*

Domain vs. Zone

- Domain (from allocation)
  - Determined by the namespace structure
- Zone (from delegation)
  - Determined by administration



Namespace hierarchy !=
Operation/Administration hierarchy

Exploring DNS

- dig
  - Should be available by default on macOS
  - Part of "bind" package on Linux (and if brave enough, on Windows)

https://www.digwebinterface.com/

```
tianyuan% dig . NS

; <<>> DiG 9.10.6 <<>> . NS
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 38900
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;. IN NS

;; ANSWER SECTION:
. 16232 IN NS e.root-servers.net.
. 16232 IN NS h.root-servers.net.
. 16232 IN NS l.root-servers.net.
. 16232 IN NS i.root-servers.net.
. 16232 IN NS a.root-servers.net.
. 16232 IN NS d.root-servers.net.
. 16232 IN NS c.root-servers.net.
. 16232 IN NS b.root-servers.net.
. 16232 IN NS j.root-servers.net.
. 16232 IN NS k.root-servers.net.
. 16232 IN NS g.root-servers.net.
. 16232 IN NS m.root-servers.net.
. 16232 IN NS f.root-servers.net.
```

```
tianyuan% dig a.root-servers.net (A)

; <<>> DiG 9.10.6 <<>> a.root-servers.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR,
id: 30471
;; flags: qr rd ra; QUERY: 1, ANSWER: 1,
AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;a.root-servers.net. IN A

;; ANSWER SECTION:
a.root-servers.net. 604800 IN A 198.41.0.4
```

```
tianyuan% dig a.root-servers.net aaaa
......
;; QUESTION SECTION:
;a.root-servers.net. IN AAAA

;; ANSWER SECTION:
a.root-servers.net. 604800 IN AAAA
2001:503:ba3e::2:30
```

- 2nd level domains:
  - UCLA runs its own DNS servers
- 3rd level domains: CS dept runs its own DNS servers

```
tianyuan% dig ucla.edu ns
.......
;; QUESTION SECTION:
;ucla.edu. IN NS

;; ANSWER SECTION:
ucla.edu. 917 IN NS ns2.dns.ucla.edu.
ucla.edu. 917 IN NS ns3.dns.ucla.edu.
ucla.edu. 917 IN NS ns4.dns.ucla.edu.
ucla.edu. 917 IN NS ns1.dns.ucla.edu.

;; ADDITIONAL SECTION:
ns1.dns.ucla.edu. 10093
ns2.dns.ucla.edu. 17620
ns2.dns.ucla.edu. 19766
ns3.dns.ucla.edu. 11775
ns4.dns.ucla.edu. 21258
```

```
tianyuan% dig cs.ucla.edu ns

;; QUESTION SECTION:
;cs.ucla.edu.                    IN       NS

;; ANSWER SECTION:
cs.ucla.edu.          14400      IN       NS        NS0.cs.ucla.edu.
cs.ucla.edu.          14400      IN       NS        NS3.cs.ucla.edu.
cs.ucla.edu.          14400      IN       NS        NS2.DNS.ucla.edu.
cs.ucla.edu.          14400      IN       NS        NS2.cs.ucla.edu.
cs.ucla.edu.          14400      IN       NS        NS3.DNS.ucla.edu.
cs.ucla.edu.          14400      IN       NS        NS1.cs.ucla.edu.
cs.ucla.edu.          14400      IN       NS        NS1.DNS.ucla.edu.
```

DNS data is coded in Resource Record (RR)

RR format

```
+--------------+-----+-----+---------+-----+---------------+
+     name     |type |class|   TTL   |  RL |     RDATA     |
+--------------+-----+-----+---------+-----+---------------+
```

# of bytes	<variable length>	2	2	4	2	<variable length>

- Name: a list of labels
  - label: 1-byte length value, followed by n char's
- Type: A, AAAA, NS, CNAME, MX, TXT, ...
  - a number of new types added in recent years
- Class: protocol family (IN = Internet)
- TTL: cache lifetime measured in second
  - DNS operators set the TTL value for data in master file
- RDLENGTH: Resource Data length

RDATA: A function of RR type

Some commonly used DNS RR types

- A: IPv4 address; AAAA: IPv6 address
  - Name = a DNS name,
  - RDATA = IP address
- NS: an authoritative DNS name server for the named domain
  - Name = a domain name,
  - RDATA = DNS server's name
- CNAME: *canonical name*
  - Name = *a* canonical name
  - RDATA = the real DNS name
- MX: mail server
  - Name = a domain name
  - RDATA = 2-byte preference value + DNS name for mail server
- TXT: any value in text format

DNS protocol

FYI

Client-server based: DNS *query* and *reply* over UDP/TCP

Message header:

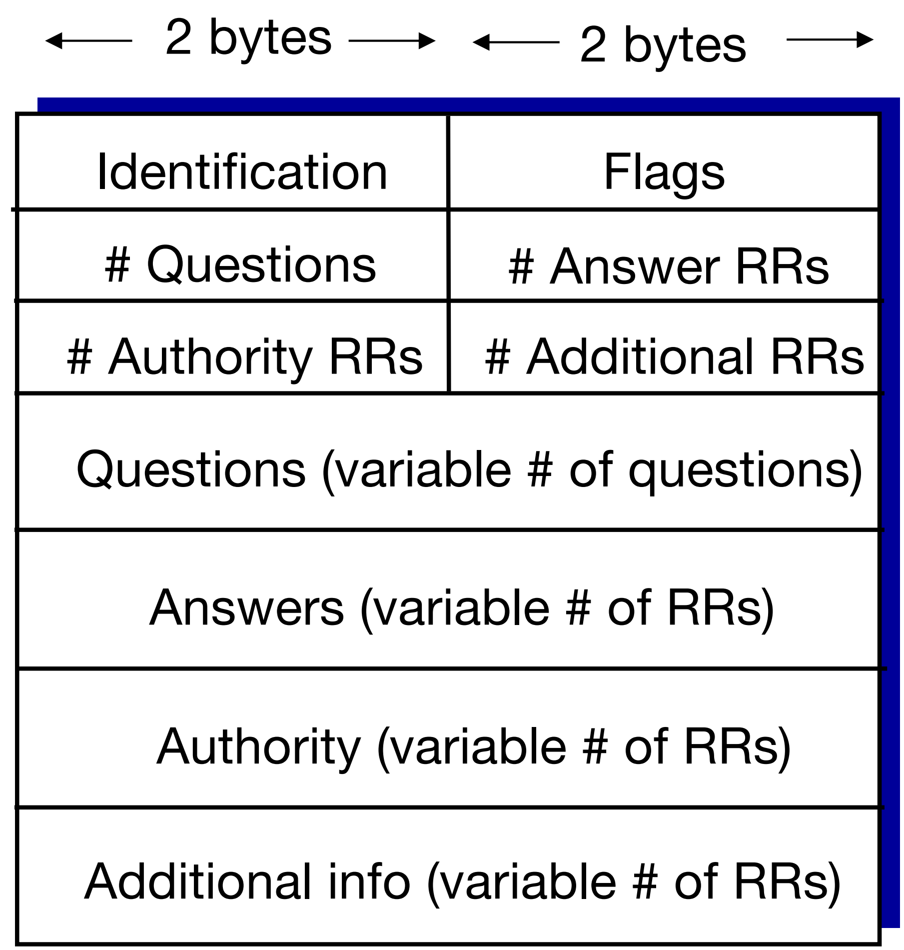
- **Identification:** 16 bit # for query, reply to query uses same #
- **Flags:**
  - Query or reply
  - Recursion desired
  - Recursion available
  - Reply is authoritative

← 2 bytes →	← 2 bytes →
Identification	Flags
# Questions	# Answer RRs
# Authority RRs	# Additional RRs
Questions (variable # of questions)	
Answers (variable # of RRs)	
Authority (variable # of RRs)	
Additional info (variable # of RRs)	

DNS protocol (contd.)

FYI

Client-server based: DNS *query* and *reply* over UDP/TCP



Name, type fields for a query

RRs in response to query

Records for authoritative servers

Additional “ helpful” info that may be used

RRset: a set of resource recodes

- DNS stores *multiple* values of the same name, class, & type in *multiple* RRs

```
tianyuan% dig ucla.edu ns

; <<>> DiG 9.10.6 <<>> ucla.edu ns
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
13378
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY:
0, ADDITIONAL: 0

;; QUESTION SECTION:
;ucla.edu. IN NS

;; ANSWER SECTION:
ucla.edu. 1077 IN NS ns3.dns.ucla.edu.
ucla.edu. 1077 IN NS ns4.dns.ucla.edu.
ucla.edu. 1077 IN NS ns1.dns.ucla.edu.
ucla.edu. 1077 IN NS ns2.dns.ucla.edu.
```

an RRset

- An RRset: made of all the RRs with the same name, class, and type
  - the basic DNS response unit

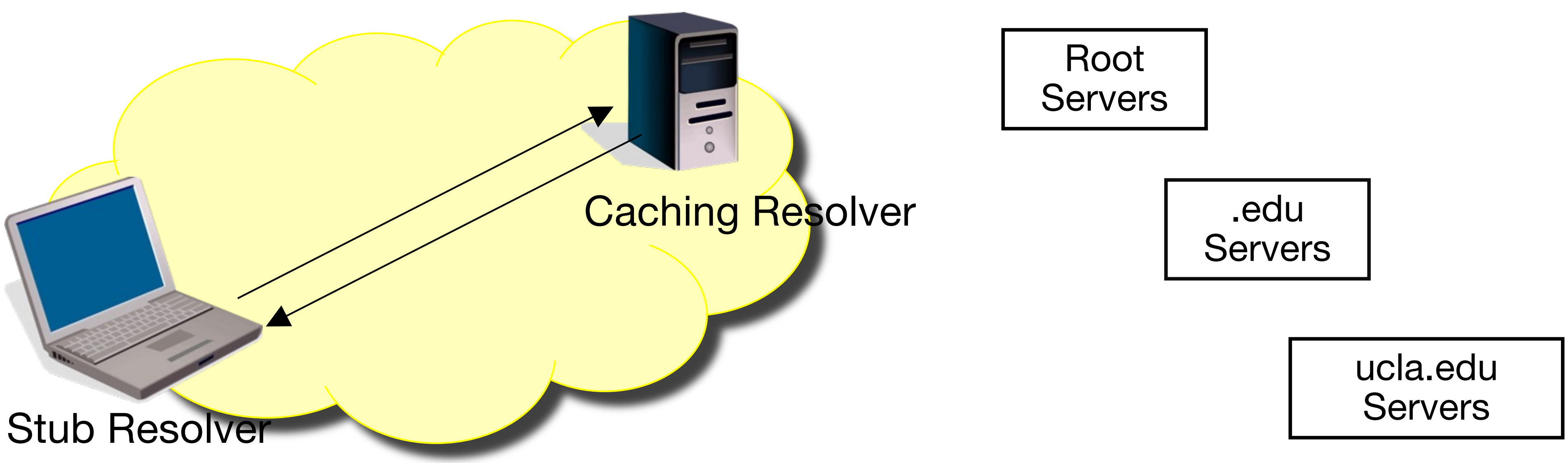
Performance of the DNS System

DNS caching

- Each resolver saves a copy of all query replies
  - Both caching resolvers and stub resolvers keep a cache
  - Stale cache entries removed from cache when their TTL expires
- A caching resolver is most likely to have in its cache
  - the DNS server info (*both* names & IP addresses) for popular TLDs (e.g. .com, .edu)
  - the DNS server information for popular sites (e.g. google, apple, amazon, cnn, etc.)
- 2 major advantages from caching
  - reduce authoritative server load and network traffic
  - shorten response delay

How a caching resolver makes query decisions: an example

- Your browser needs IP address for www.ucla.edu: the caching resolver ***CR*** doesn't have it
- Where will ***CR*** send its first query to?
  - Depending on what other info ***CR*** may have in its cache
    - ucla.edu authoritative server info?
    - .edu authoritative server info?
    - If none of the above: go to one of the root servers



Scale the DNS system

- Scale the namespace e.g. how many names the Internet can have
  - Hierarchical namespace, with variable name length
- Scale the management by delegation

172800 sec = 2 days

```
tianyuan% dig edu. ns
......
;; QUESTION SECTION:
;edu.                            IN        NS

;; ANSWER SECTION:
edu.                   172800    IN        NS         j.edu-servers.net.
edu.                   172800    IN        NS         m.edu-servers.net.
edu.                   172800    IN        NS         g.edu-servers.net.
edu.                   172800    IN        NS         l.edu-servers.net.
edu.                   172800    IN        NS         k.edu-servers.net.
edu.                   172800    IN        NS         i.edu-servers.net.
edu.                   172800    IN        NS         b.edu-servers.net.
edu.                   172800    IN        NS         c.edu-servers.net.
edu.                   172800    IN        NS         d.edu-servers.net.
edu.                   172800    IN        NS         a.edu-servers.net.
......
```

Providing Resilient DNS Service

important

- Resilient service = high availability in face of network and/or server failures
- basic means for resiliency: redundancy
  - Redundancy: replicating authoritative servers
  - Caching: as opportunistic replication
- Root domain has 13 replicate authoritative server names and corresponding IP addresses

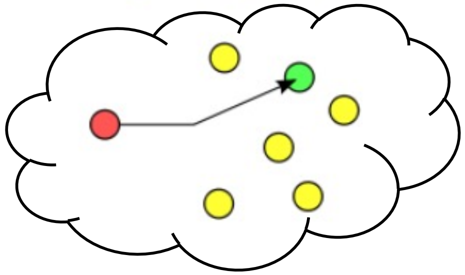
  https://root-servers.org/: “As of 2025-01-15T04:40:06Z, the root server system consists of 1921 instances operated by the 12 independent root server operators.”

Anycast delivery

FYI

https://en.wikipedia.org/wiki/Anycast

Unicast



Unicast: A given IP address block ***A*** is announced from a single location

Broadcast

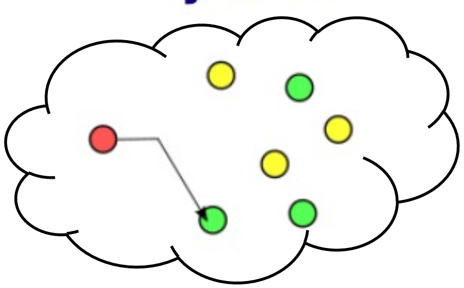
Broadcast: if a packet's destination IP is broadcast, send it everywhere

Multicast

Multicast: an IP multicast address represents a group of recipients

- need fundamental changes to IP forwarding
- need multicast routing protocol support

Anycast



Anycast: A given IP address block ***A*** is announced from multiple locations

- a route receives reachability to ***A*** from multiple neighbors, pick the shortest path to forward packets

(ab)Using DNS for Content Distribution Networks (CDN)

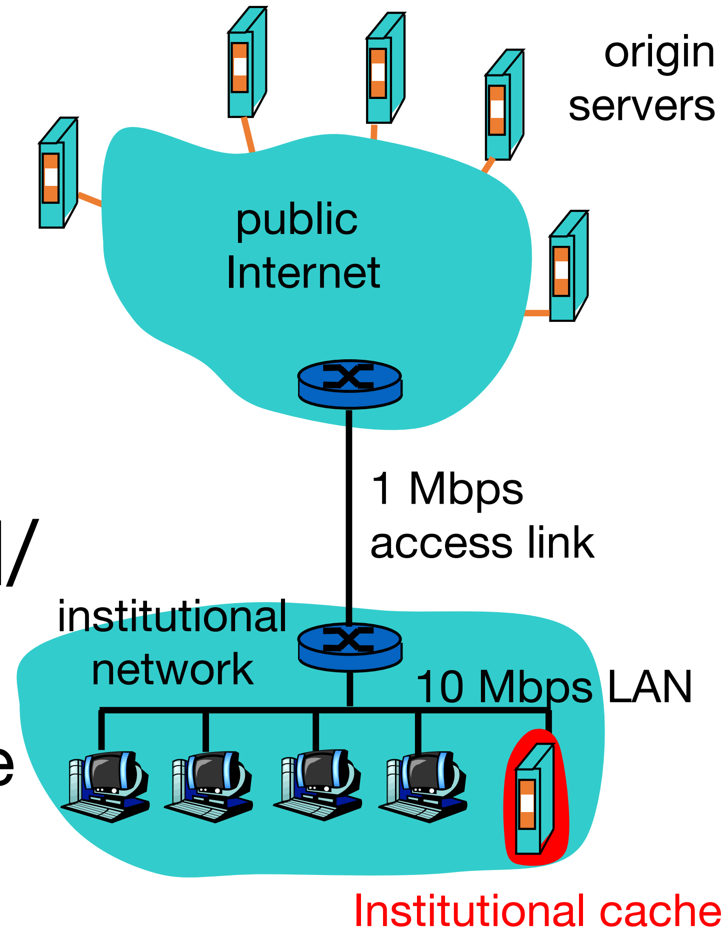
```
tianyuan% dig @ns1.dns.ucla.edu www.ucla.edu
...
;; QUESTION SECTION:
;www.ucla.edu. IN A


;; ANSWER SECTION:
www.ucla.edu. 60 IN CNAME d1zev4mn1zpfbc.cloudfront.net.
d1zev4mn1zpfbc.cloudfront.net. 56 IN A 18.154.132.29
d1zev4mn1zpfbc.cloudfront.net. 56 IN A 18.154.132.63
d1zev4mn1zpfbc.cloudfront.net. 56 IN A 18.154.132.13
d1zev4mn1zpfbc.cloudfront.net. 56 IN A 18.154.132.92
```

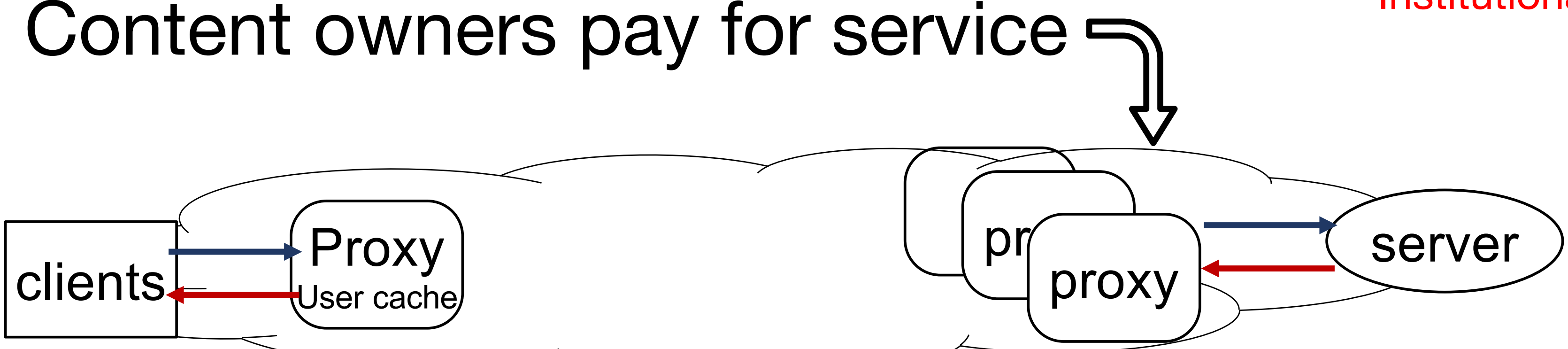
???

Where contents can be cached

and who provide the caches

- Lecture-2 on HTTP caching:
  - Caches provided by end user sites, located near users
- Caches can also be provided/ arranged by content owners
  - CDN providers offer caching service
    - *with caches widely distributed*
  - Content owners pay for service





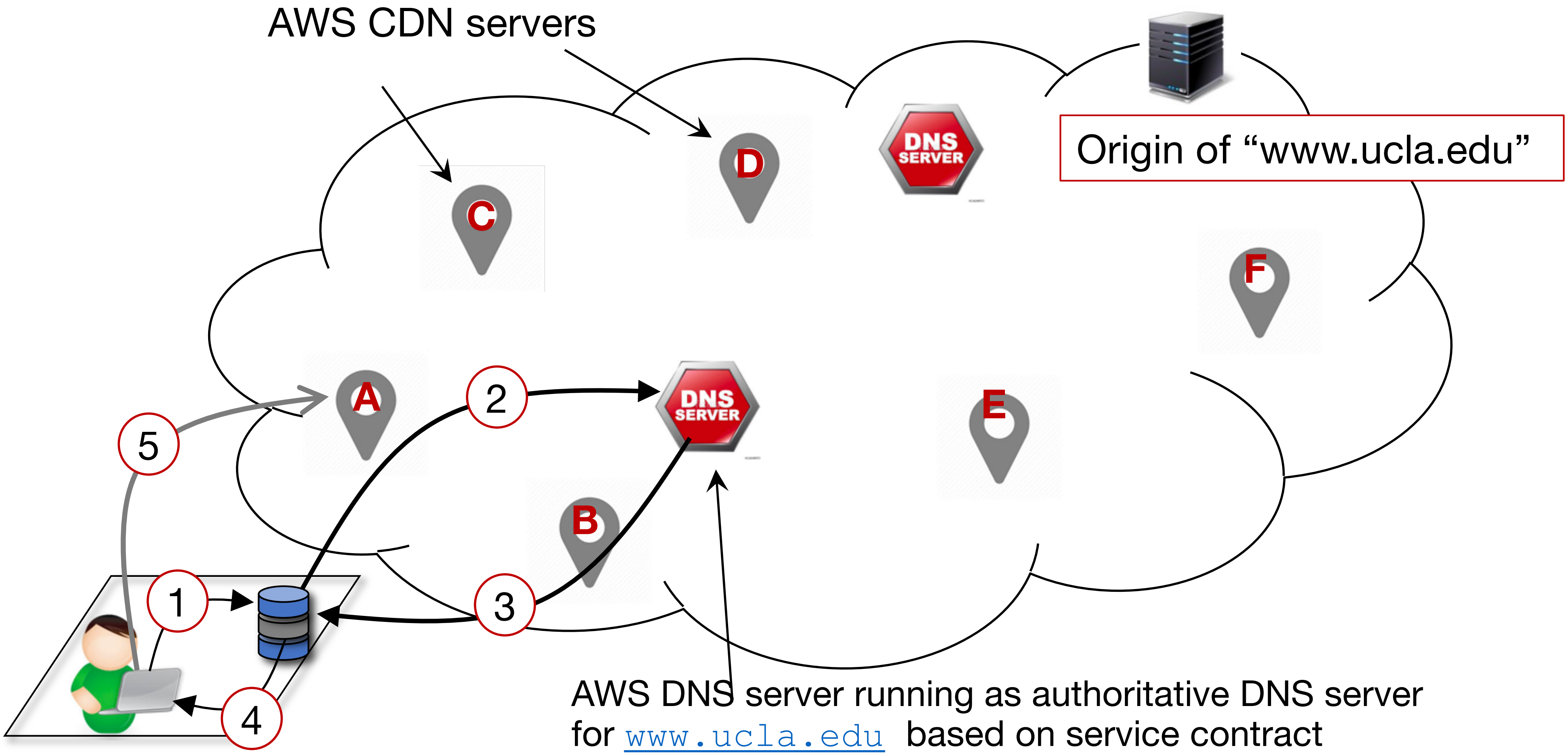
How to get user HTTP requests to nearby CDN server

without changing user host or application:

- A user queries DNS to get web server www.ucla.edu IP address, then sets up TCP connection to it
  - One can make the DNS server return the IP address of nearest CDN server
- When UCLA pays for CDN service: outsource www.ucla.edu hosting to the CDN provider
  - `www.ucla.edu. 60 IN CNAME d1zev4mn1zpfbc.cloudfront.net`

↑ AWS CDN Product

Getting user's HTTP requests to a nearby CDN server



1-4: user host queries for www.ucla.edu's IP address, gets back IP address for AWS server-A

5: user host connects to AWS server-A to fetch the page

Another example: cs118.org

- cs118.org buys Cloudflare service (domain and CDN)
  - Cloudflare provides authoritative DNS servers for cs118.org
- When Cloudflare DNS server receives a DNS query:
  - Get the source IP address from the query message
  - Use the address to estimate the user location, then return the IP address of a nearby CDN box
- cs118.org turns on HTTPS?
  - Share the crypto key with Cloudflare to make a browser believe it's connected to cs118.org
- What if some CDN box is overloaded? Or crashed?

Using DNS for load balancing

- Common practice by CDN servers
- Assign a very short TTL for the final (DNS lookup) result, to avoid it being cached for long

```
tianyuan% dig cs118.org a
;; QUESTION SECTION:
;cs118.org. IN A


;; ANSWER SECTION:
cs118.org. 75 IN A 104.21.48.1
cs118.org. 75 IN A 104.21.64.1
cs118.org. 75 IN A 104.21.96.1
cs118.org. 75 IN A 104.21.112.1
cs118.org. 75 IN A 104.21.16.1
cs118.org. 75 IN A 104.21.80.1
cs118.org. 75 IN A 104.21.32.1
```

DNS and security

Attacks to DNS

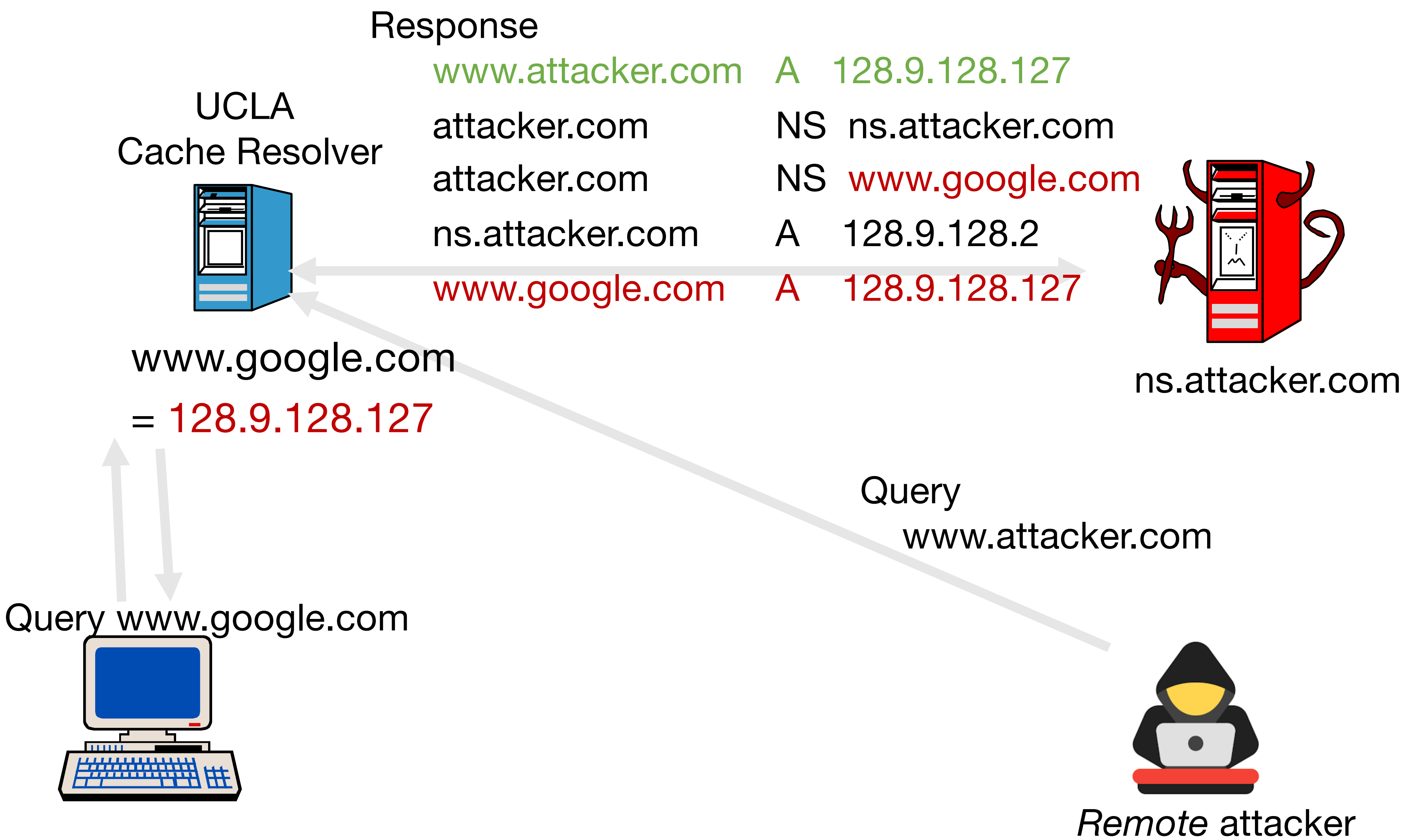
Brute force DDoS attacks

- flood root servers with queries (so far: not successful)
- flood specific lower level DNS servers
  - some of them not well provisioned

Man-in-middle attacks

- Intercept queries, then send bogus relies to caching resolvers
  - resulting in *cache poisoning*
- Other means to achieve *cache poisoning*

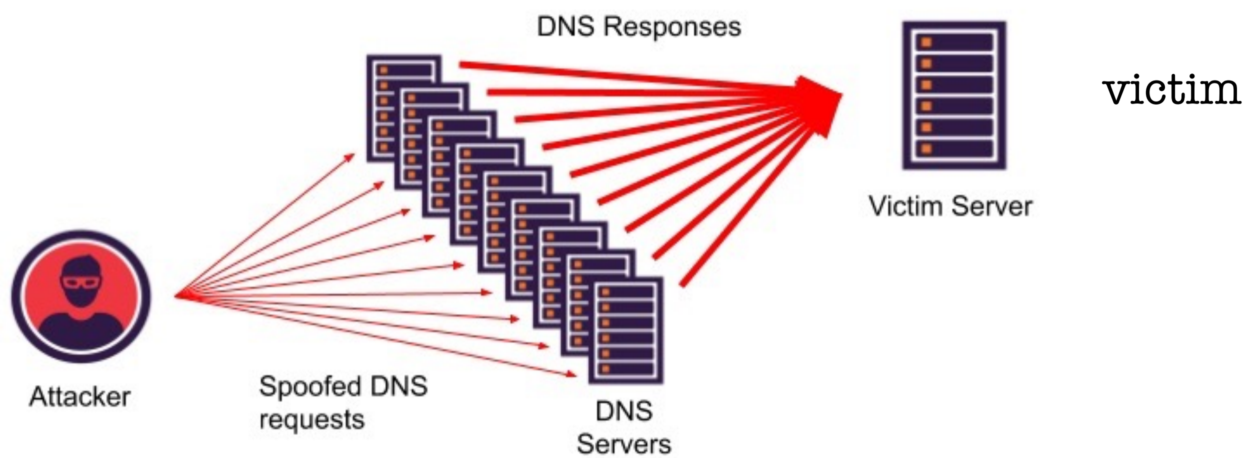
DNS cache poisoning



Abusing DNS as attack tool

Exploiting DNS for DDoS

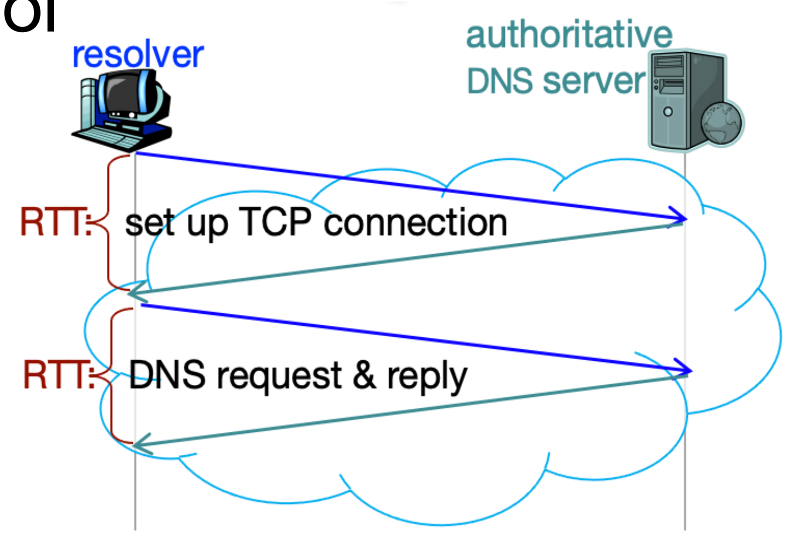
- Send queries with *spoofed* source address = victim IP
- Using large number of compromised devices to amplify attack



- One way to mitigate: DNS over TCP instead of UDP

Running DNS over TCP versus UDP

- DNS protocol: an application protocol
- Running over TCP
  - Take minimum 2 RTTs to get reply

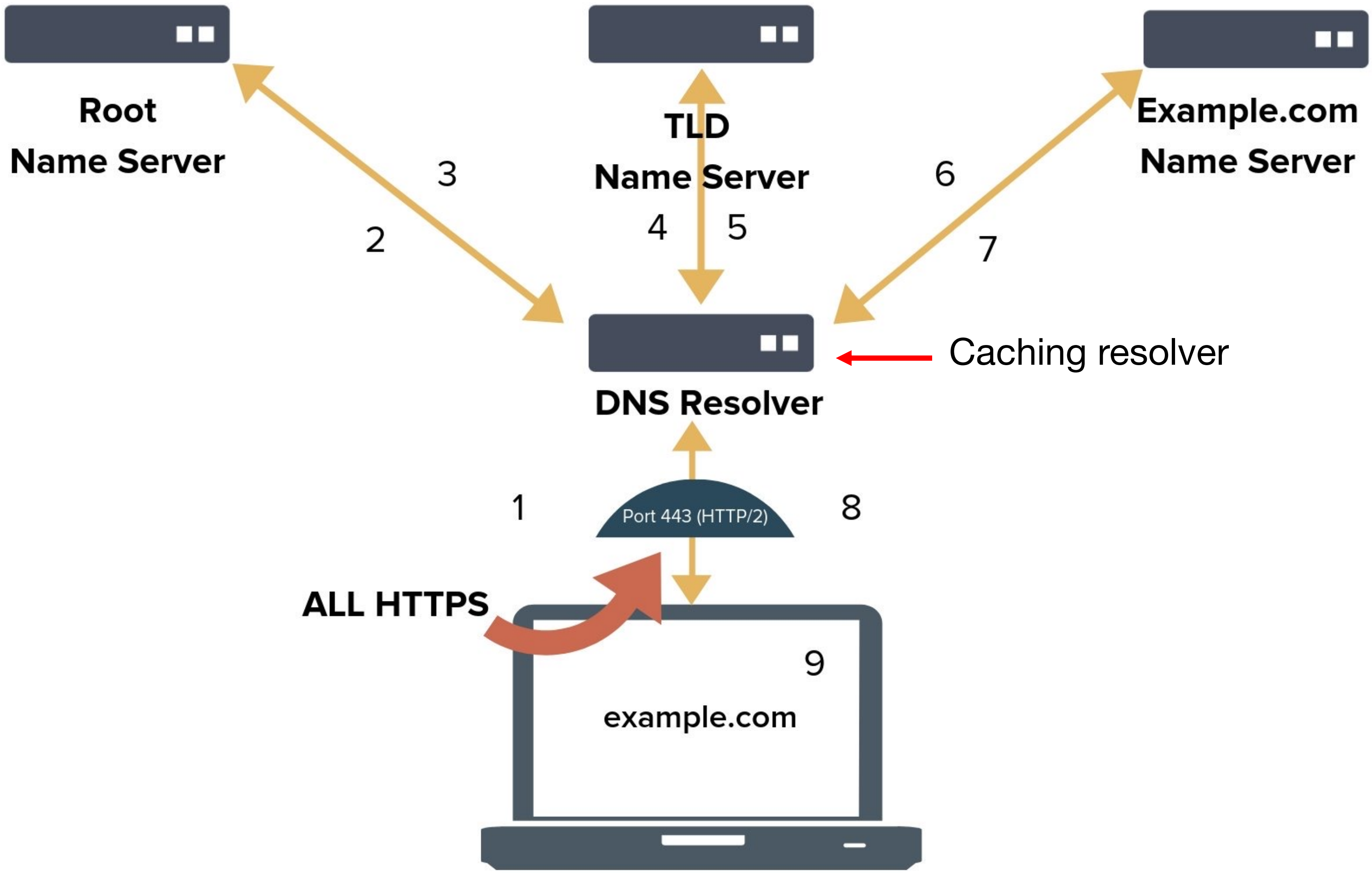


- Running over UDP: the resolver detects packet loss and retries
  - when sending a query, resolver sets a retransmission timer
    - If no loss: receives reply in one RTT
  - If no reply received when the timer expires: retry with another authoritative server in the same RRset

packet losses happen from time to time, independent from which transport protocol being used. TCP simply hides losses by doing loss detection and retry (but it can't retry a different server).

Latest change: DNS over HTTPS (DoH)

FYI



DNS: Not a fully automated system

- DNS defines the following
  - standard formats for storing DNS data (RR)
  - standard protocol for querying the database
  - Tuning knobs of DNS service configurations
- Operators do the following
  - define domain boundaries and child-domain delegations
  - define desired operation policies
    - Cache validation period (TTL)
    - Master → secondary server synchronization period
  - manually update the master file of each domain
    - For ns RR and glue RR updates: contact the parent zone's operator

Inserting records into DNS

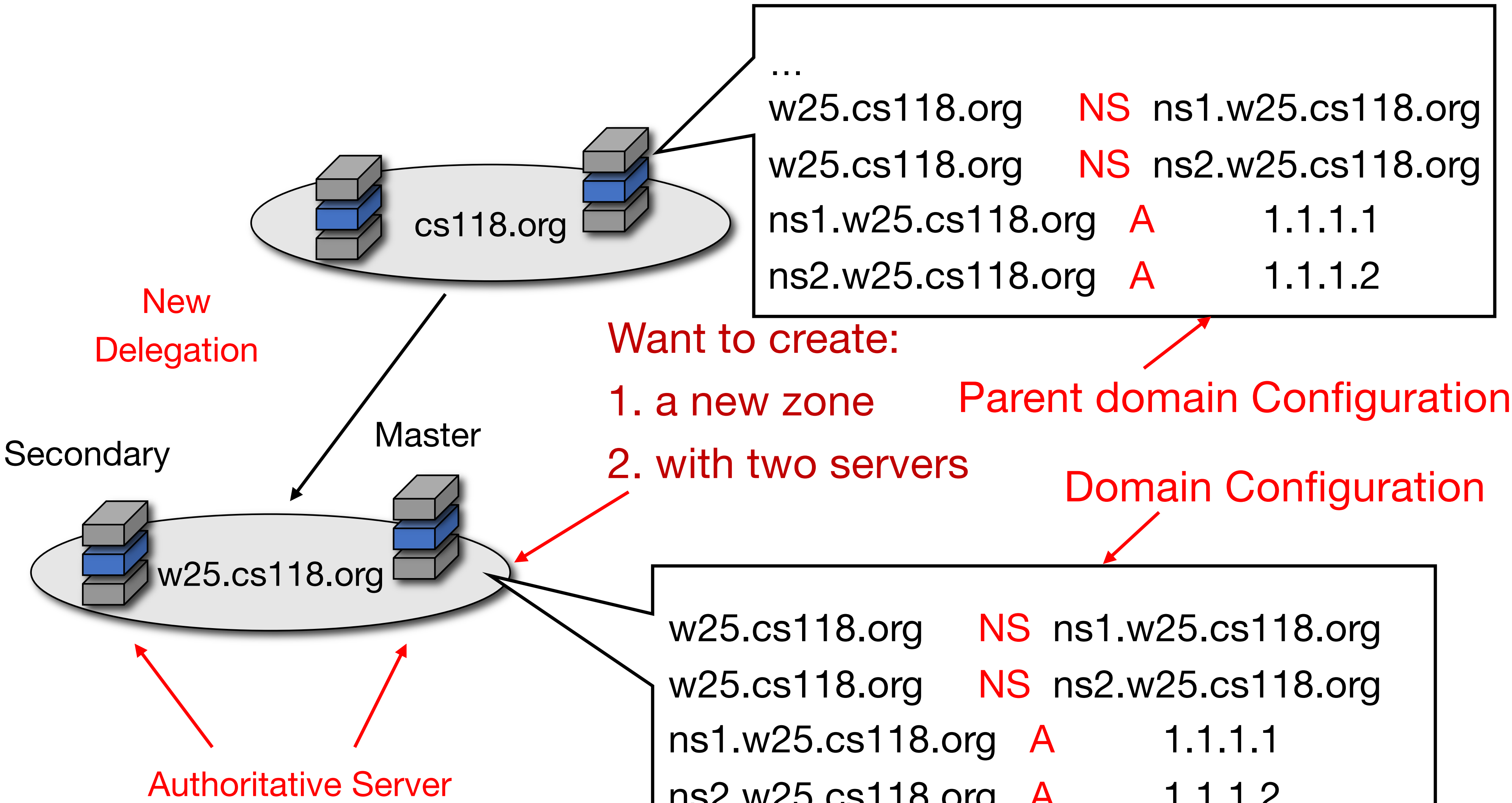
- Example: assume creating a "W25 CS118" zone
- Register name w25 at cs118.org
  - Need to provide with names and IP addresses of your authoritative name servers (primary and secondary)
  - Registrar inserts two RRs into the cs118.org name server:

```
w25.cs118.org,      NS,     ns1.w25.cs118.org
ns1.w25.cs118.org, A,       1.1.1.1
```

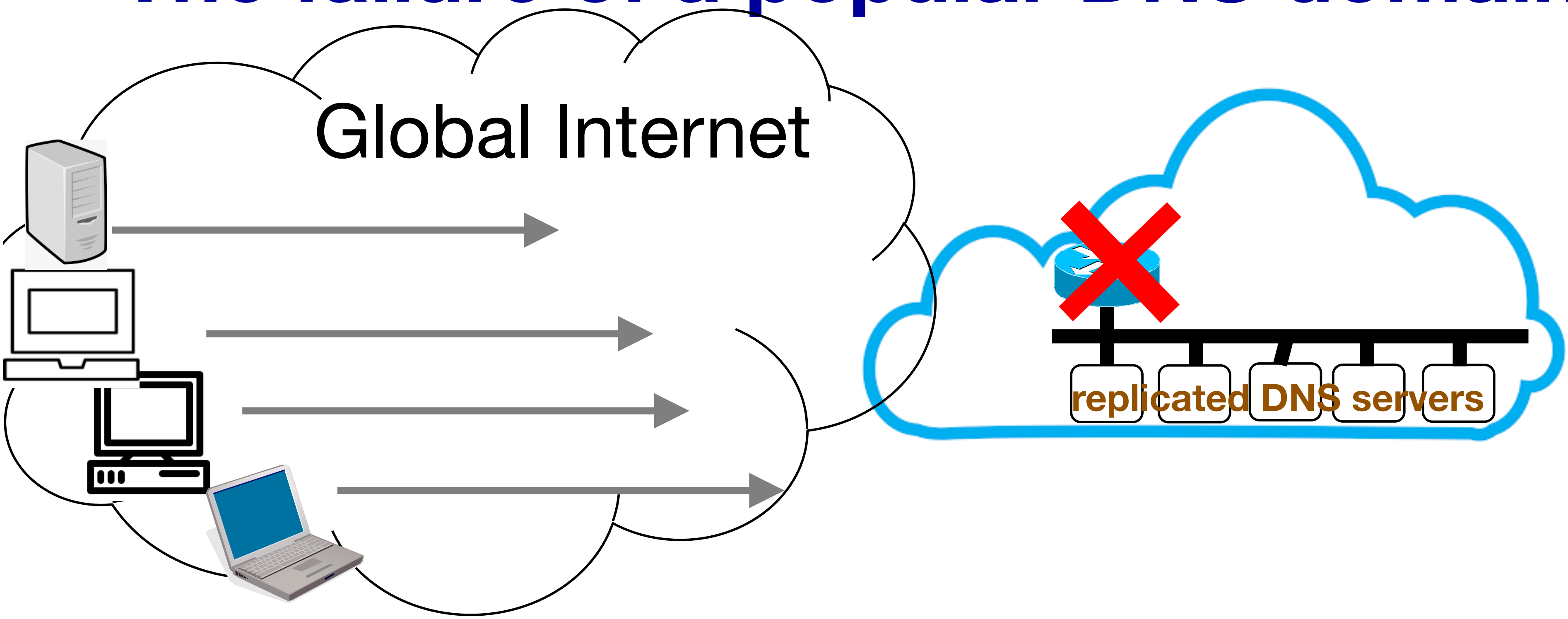
- Put in authoritative server Type A record for `www.w25.cs118.org`

How do people get the IP address of **`www.w25.cs118.org`**?

Example Configuration



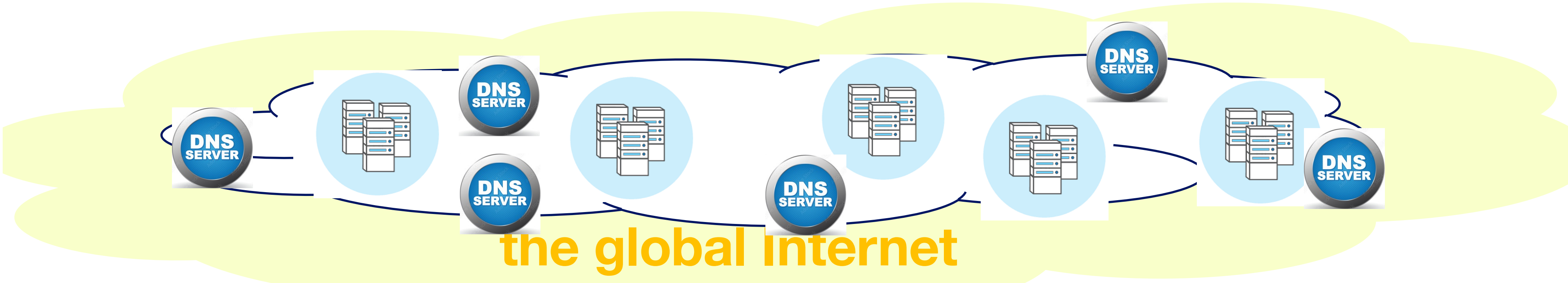
The failure of a popular DNS domain



- A popular domain → lots queries for DNS names under this domain
- If the domain's authoritative servers no longer reachable: all cached entries time-out eventually
- When caching resolvers tried to look up a name: what happens when all the authoritative servers for that domain become unreachable?

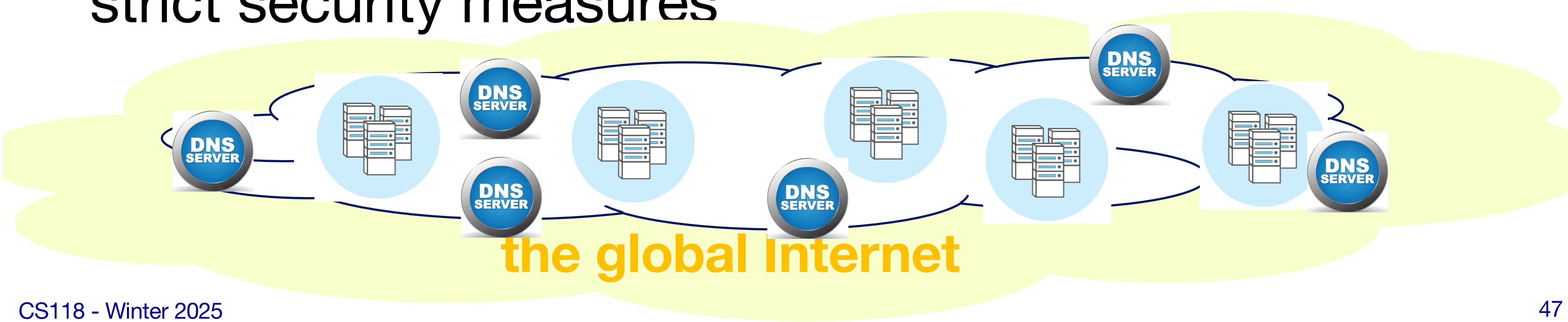
Meta network connectivity

- Meta has an internal network
  - connecting its data centers and DNS servers world-wide
    - They reach each other through the internal routing
    - All Meta authoritative DNS servers hosted internally
- Meta data centers and DNS servers make BGP routing announcements to the global internet
- Meta DNS servers keep track data-center availability, reply queries with nearest datacenter address and short TTLs
  - If a DNS server can't reach any data centers, it stops making BGP announcements



Understanding the Meta Failure

- During a regular internet maintenance, an error caused all the datacenters unreachable
  - Audit tool designed to prevent such mistakes had a bug
- Meta DNS servers withdrew their BGP announcements
  - Facebook, WhatsApp, Instagram names became unresolvable
- The loss of DNS broke many internal tools normally used to investigate and resolve outages
- Engineers sent to datacenters were delayed access by strict security measures



The Meta failures impacted others

- SERVFAIL: caching resolver times out, cache negative results
- A tsunami of additional DNS traffic follows
  - apps won't accept an error for an answer and start retrying
  - end-users also won't take an error for an answer and start reloading the pages, or killing and relaunching their apps, sometimes aggressively.
- 30x increase of DNS traffic caused latency and timeout issues to other platforms