04 – Identifiers

- UUID
- URI
- Format
- Characteristics

- Coulouris, Ch 9
- rfc3986
- Ahmed, 2005
- Subharthi, 2009

Resource identification

- All resources should have a name
- Resource without Identifier
  - Can not be shared
  - Can not be accessed
- UUUI
- URI

UUID

- Universally Unique Identifier
- Globally Unique Identifier (GUID)
- 128 bit identifier
- No central authority 9administration/registration)
  - Generated locaaly
-

UUID generation

- Combination of components
- V1
  - MAC Address + date (with 0.1 milisec precision)
- V2
  - User Id + MAC Address + date
- V3/V5
  - Digest(adress+name+identifier)
- V4
  - Random values

UUID

- “Unique”across space and time
  - Will roll over on 3400 DC
  - Extremely likely to be different
- Tags short lived or persistent objects
- Does not specify name resolution architecture
  - Application level

URI

- Universal resource identifier
- URN
  - Universal resource name
- URL
  - Universal resource locator

URI

- Identify resources in coherent way
- Processed by common SW (e.g. browsers)
- Uniform syntax
  - Incorporates indefinitely type of identifiers
  - Schemes
- Global namescpaces (for each scheeme)
- Exetnsible
  - New schemes <=> new identifiers

URI = scheme ":" hier-part ["?" query] ["#" fragment]

hier-part = "//" authority path-abempty

/ path-absolute

/ path-rootless

/ path-empty

- Examples
  -
```
foo://example.com:8042/over/there?name=ferret#nose
\_/   \______________/\_________/ \_________/ \__/
 |           |            |            |        |
scheme     authority       path        query   fragment
 |   _____________________|__
/ \ /                     \
urn:example:animal:ferret:nose
```

URL

- Provides formalized resource information for
  - Location
  - Access
- May specify operations to perform
- Contains Internet application protocol required
  - Http mailto, ftp

URN

- Resource identifiers
- Location independet
- Longevity of reference
- Syntax
  - Scheme : NID : namespace specifif string
- NID
  - Namespace identifier
  - Registers and informal
- Urn:isbn:0-395-36641-1

AHMED classification

- Readability
  - Are ID human readable
- Extensibility
  - changes namespace/scope
- Size
  - How many unique ID
- Authority
  - Who assigns names
    - Centralized/distributed
- Name persistency
  - Static
    - one name → one entity
  - Dynamic
    - Over time
      - One name → multiple enities

- Name resolution architecture
  - System architecture
  - Scalability
  - Efficiency
  - Fault tolerance
  - Consistency
- Standardization

AHMED classification

- Readability
  - UUID – non-human
  - URI
    - URL – yes
    - URN – not necessarly
  - DNS
- Extensibility
  - UUID
    - NO
    - 128 bits (impossible more than 2^{128} names)
    - Incompatibility between current future implement.
  - IRI
    - Yes
    - New scheems
    - URN – new NID
  - DNS
- Size
  - UUID 2^{128} name
  - URI – infinite
  - DNS - inficnite
- Authority
  - UUID
    - No central authority
  - URL
    - Scheme: centralized IANA
    - Name: centralized ICANN
    - Remaining: hieraquically
  - URN
    - NID – centralized IANA
    - Remainig: hiraquically
  - DNS

AHMED classification

- Name persistency
  - UUID – theoretical
    - UUIDs are unique across time and space
  - URI – depends
    - Location information decreases persistency
- Standardization
  - UUID yes
  - URI yes
  - DNS
- Name resolution architecture
  - UUID - Not specified
    - Application level
  - URL - Depends on DNS
  - URN
    - Resolver translates URN to URL
    - Heuristic resolution
      - Meta-information hints
    - New DNS structure
      - Point to resolvers
    - Z3950, THTTP, RCDS, HDL

DNS History

- ARPANET utilized a central file HOSTS.TXT
  - Contains names to addresses mapping
  - Maintained by SRI's NIC
    - Stanford-Research-Institute: Network-Information-Center
- Management
  - Administrators email changes to NIC
  - NIC updates HOSTS.TXT periodically
  - Administrators FTP (download) HOSTS.TXT

DNS History

- As the system grew, HOSTS.TXT had problems with:
  - Scalability (traffic and load)
  - Name collisions
  - Consistency
- In 1984, Paul Mockapetris released the first version
  - RFCs 882 and 883
  - superseded by 1034 and 1035 …

Name service evolution

- ARPANET
  - Centralized management
  - Manuel replication
- central service?
  - Single point of failure
  - Traffic volume
  - Distant centralized database
  - Single point of update
  - Doesn’t scale!

DNS

- The “Domain Name System”
- What Internet users use to reference anything by name on the Internet
- The mechanism by which Internet software translates names to attributes such as addresses
- A globally distributed, scalable, reliable database
- Comprised of three components
  - A “name space”
  - Servers making that name space available
  - Resolvers (clients) which query the servers about the name space

DNS roles

- Lookup mechanism
  - Users generally prefer names to numbers
  - Computers prefer numbers to names
  - DNS provides the mapping between the two
- Database
  - Keys to the database are “domain names”
  - Over 200,000,000 domain names stored
  - Each domain name contains one or more attributes
    - Known as “resource records”
  - Each attribute individually retrievable

DNS characteristics

- Global Distribution
  - Data is maintained locally, but retrievable globally
    - No single computer has all DNS data
  - DNS lookups can be performed by any device
  - Remote DNS data is locally chached
    - improve performance
- Loose Coherency
  - Each version of a subset of the database (a zone) has a serial number
    - The serial number is incremented on each database change
  - Changes to the master copy of the database
    - propagated to replicas according to timing set by the zone administrator
  - Cached data expires according to timeout set by zone administrator

DNS characteristics

- Scalability
  - No limit to the size of the database
  - No limit to the number of queries
  - Tens of thousands of queries handled easily every second
  - Queries distributed among masters, slaves, and caches
- Reliability
  - Data is replicated
    - Data from master is copied to multiple slaves
  - Clients can query
    - Master server
    - Any of the copies at slave servers
  - Clients will typically query local caches
  - DNS protocols can use either UDP or TCP

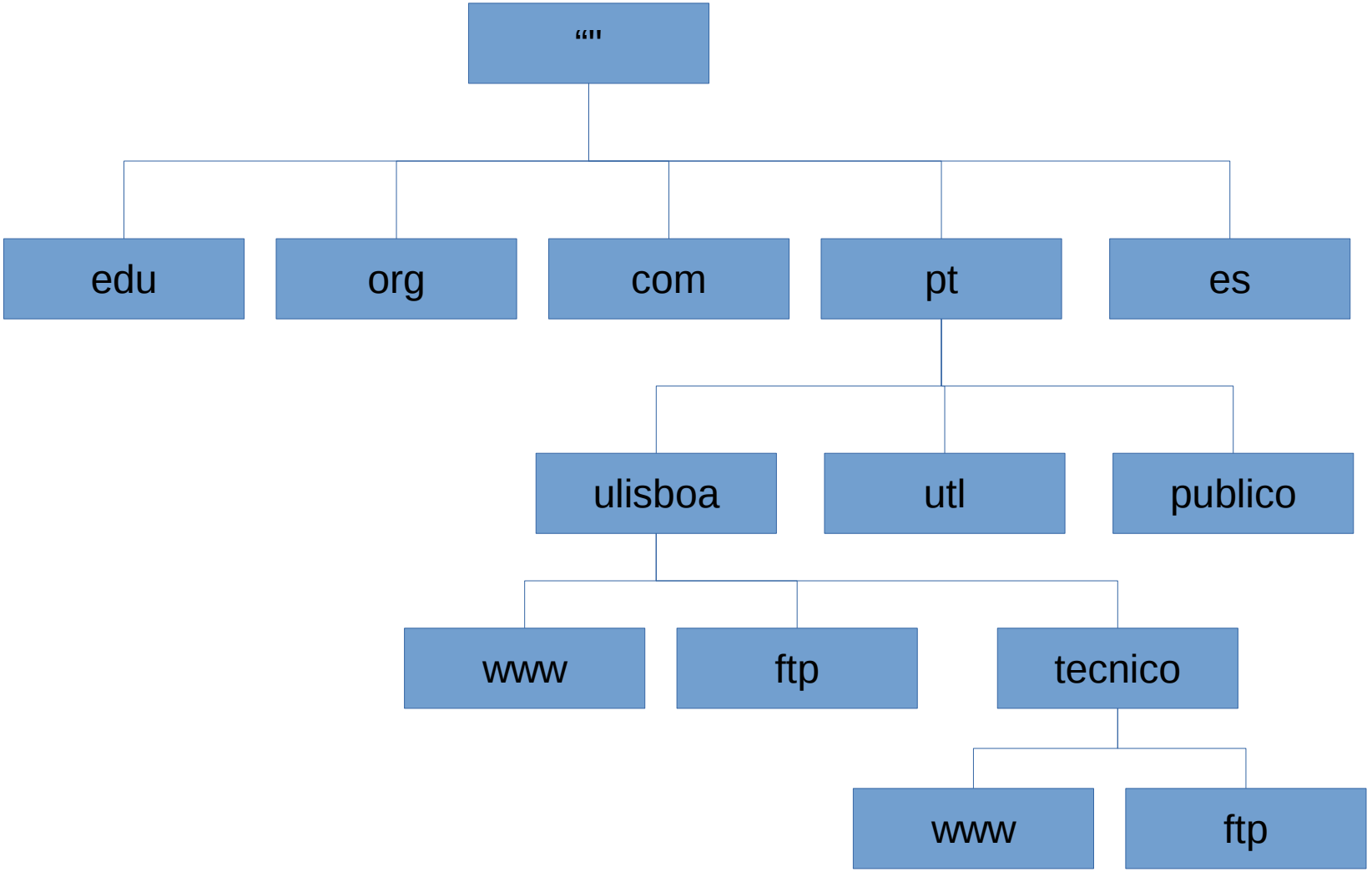
DNS Dynamicity

- Database can be updated dynamically
  - Add/delete/modify of any record
  - Only master can be dynamically updated
- Modification of the master database triggers replication

Name space

- The name space is the structure of the DNS database
  - An inverted tree with the root node at the top
- Each node has a label
  - The root node has a null label, written as “”
- A domain name is the sequence of labels from a node to the root, separated by dots (“.”s), read left to right
  - The name space has a maximum depth of 127 levels
  - Domain names are limited to 255 characters in length
- A node’s domain name identifies its position in the name space

Name space



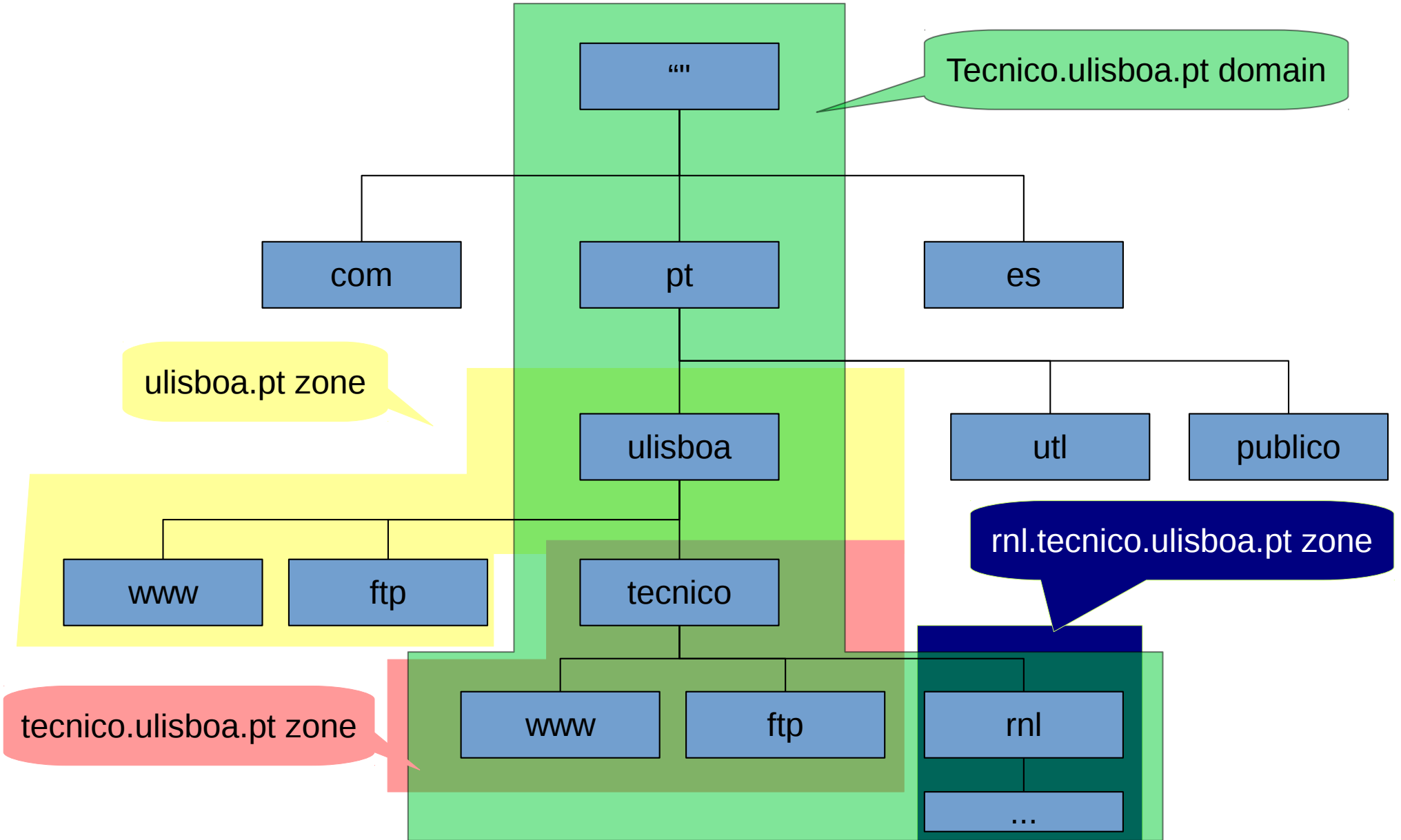
Subdomains

- One domain is a subdomain of another if its domain name ends in the other's domain name
  - **tecnico.ulisboa.pt** is a subdomain of
    - **ulisboa.pt** and **pt**
  - **ulisboa.pt** is a subdomain of **pt**

Delegation

- Administrators can create subdomains to group hosts
  - According to geography, organizational affiliation etc.
- An administrator of a domain can delegate responsibility for managing a subdomain to someone else
- The parent domain retains links to the delegated subdomains
- Zones
  - Each time an administrator delegates a subdomain,
    - a new unit of administration is created
    - The subdomain and its parent domain can now be administered independently
    - These units are called zones
    - The boundary between zones is a point of delegation in the name space
- Delegation is good: it is the key to scalability

Name space


""
Tecnico.ulisboa.pt domain
com
pt
es
ulisboa.pt zone
ulisboa
utl
publico
rnl.tecnico.ulisboa.pt zone
www
ftp
tecnico
tecnico.ulisboa.pt zone
www
ftp
rnl
...

Servers

- One zone multiple servers
- One server multiple zones
- Authoritative
  - maintains the data
- Master
  - where the data is edited
- Slave
  - where data is replicated to
- Caching
  - stores data obtained from an authoritative server

The Resolution Process

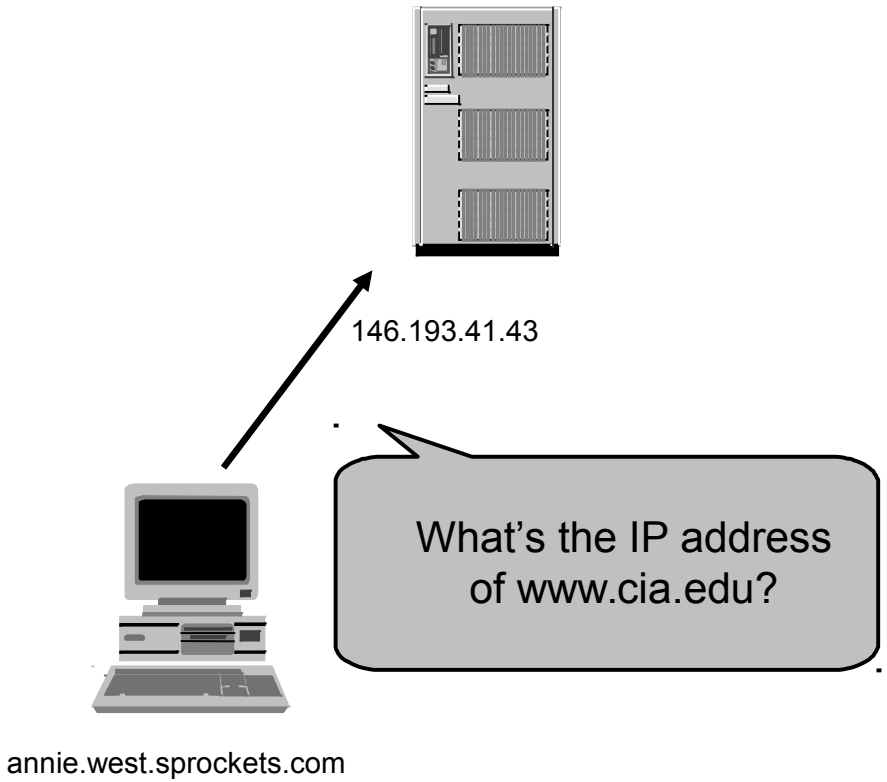
- Let’s look at the resolution process step-by-step:

Anilina.gsd.inesc-id.pt

```
ping www.cia.edu
```

The Resolution Process

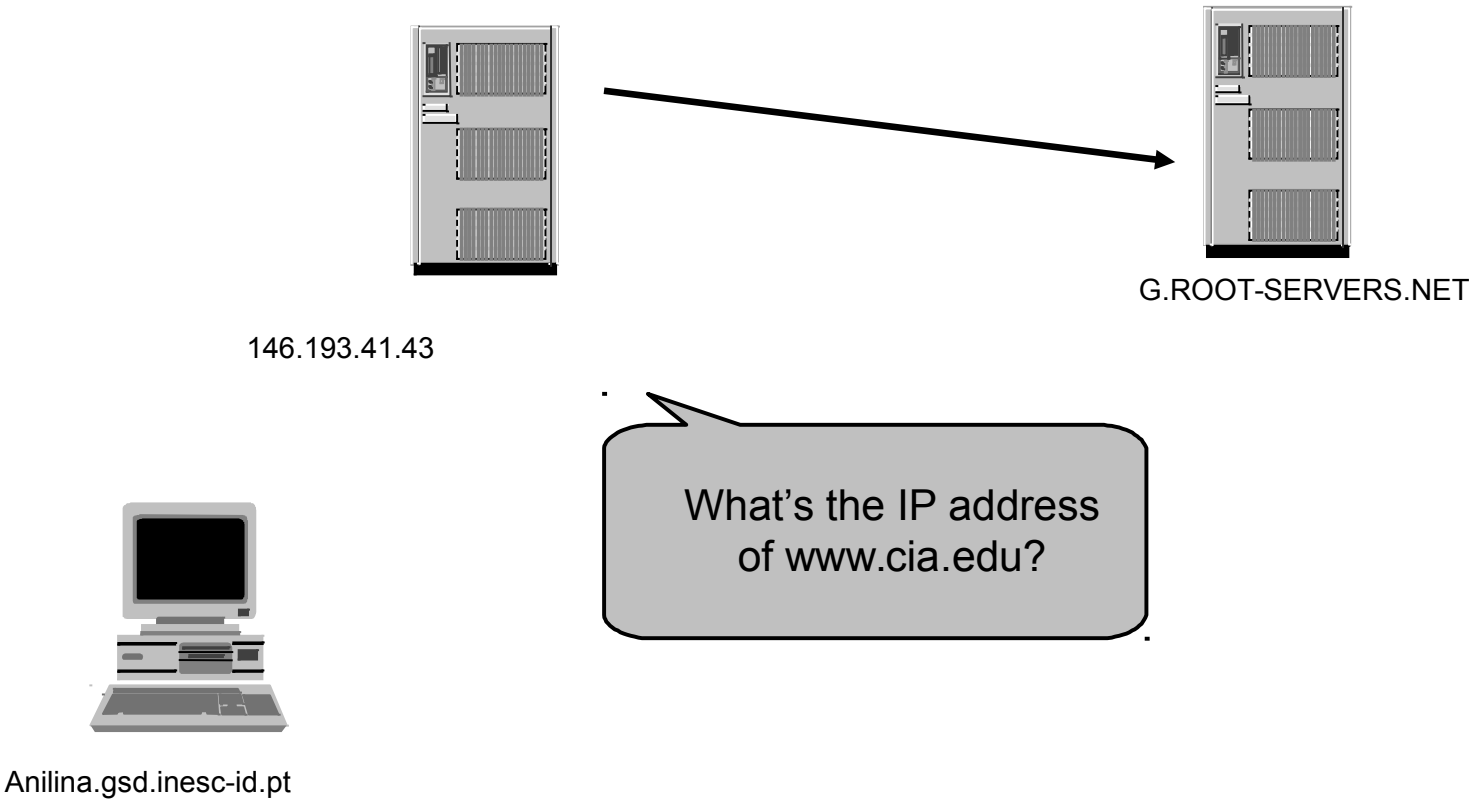
The workstation *anilina* asks its configured name server,146.193.41.43, for *www.cia.edu's* address




```
ping www.nominum.com.
```

The Resolution Process

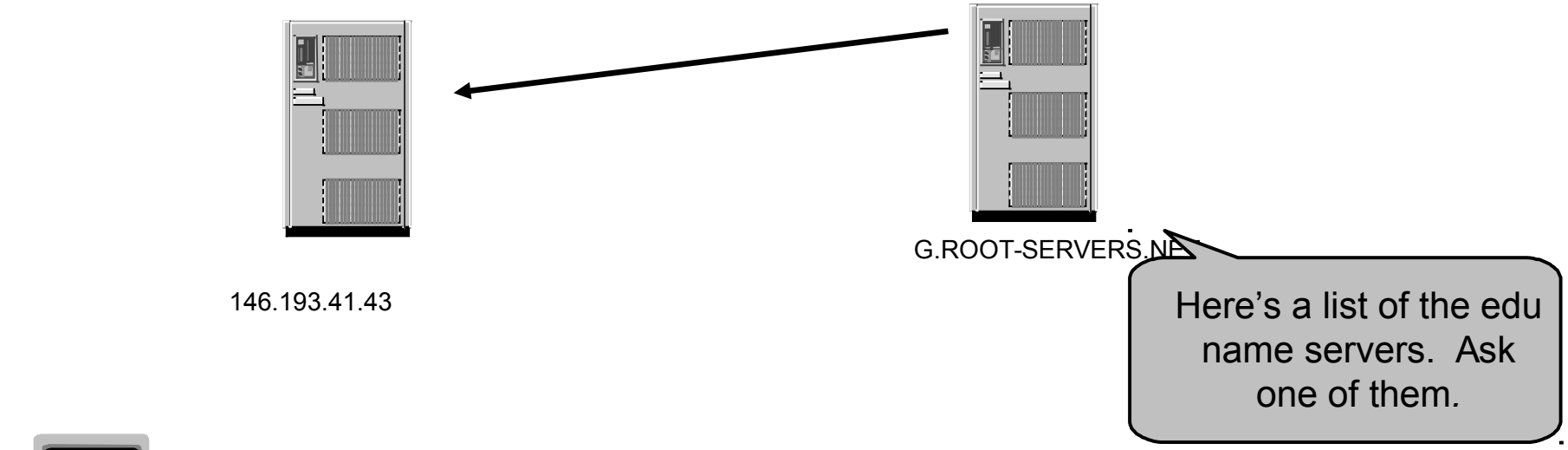
The name server 146.193.41.43 asks a root name server, *g*, for *www.cia.edu's* address




```
ping www.nominum.com.
```

The Resolution Process

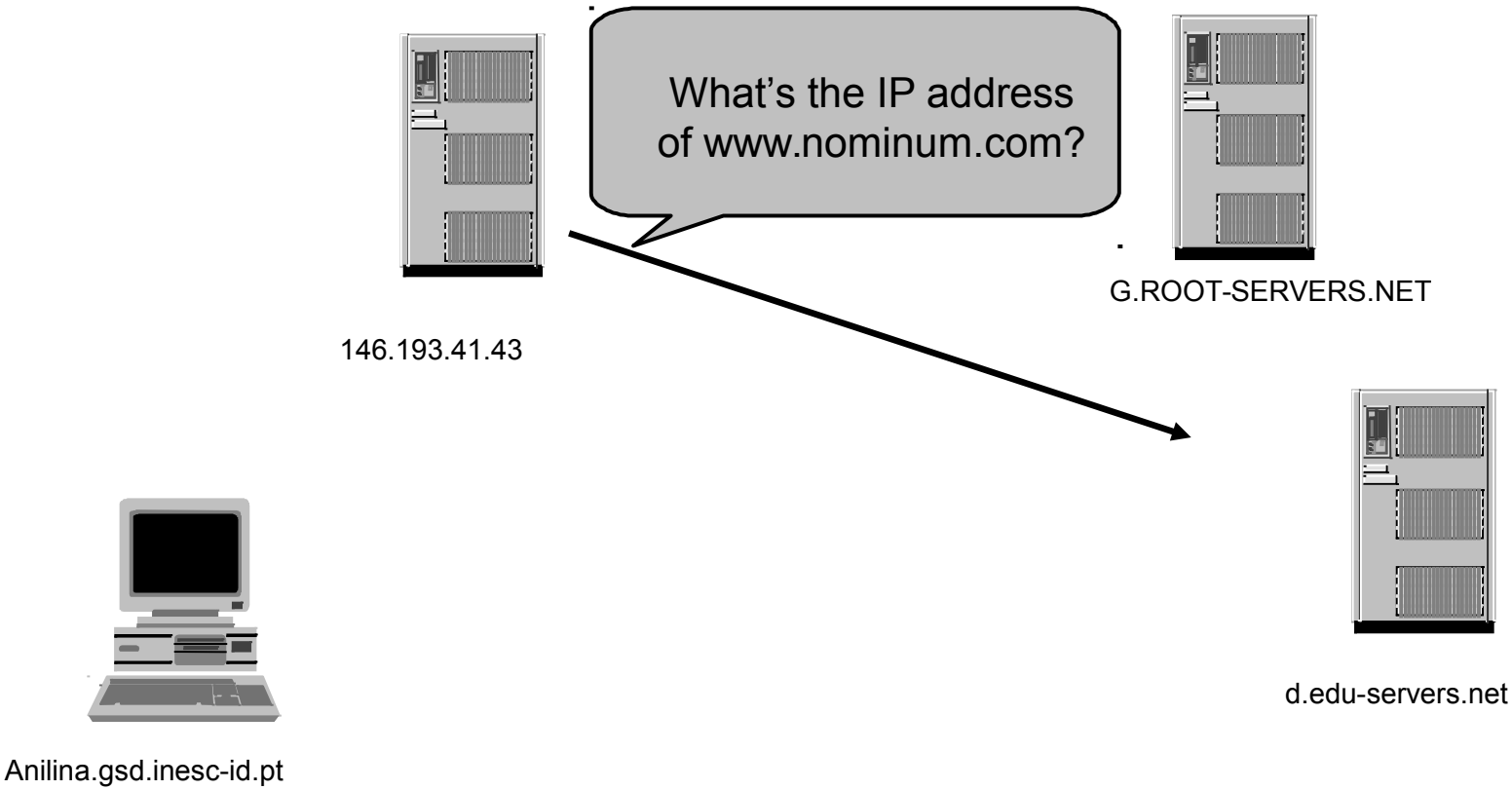
- The root server *g* refers *146.193.41.43* to the *edu* name servers
- This type of response is called a “referral”




```
ping www.nominum.com.
```

The Resolution Process

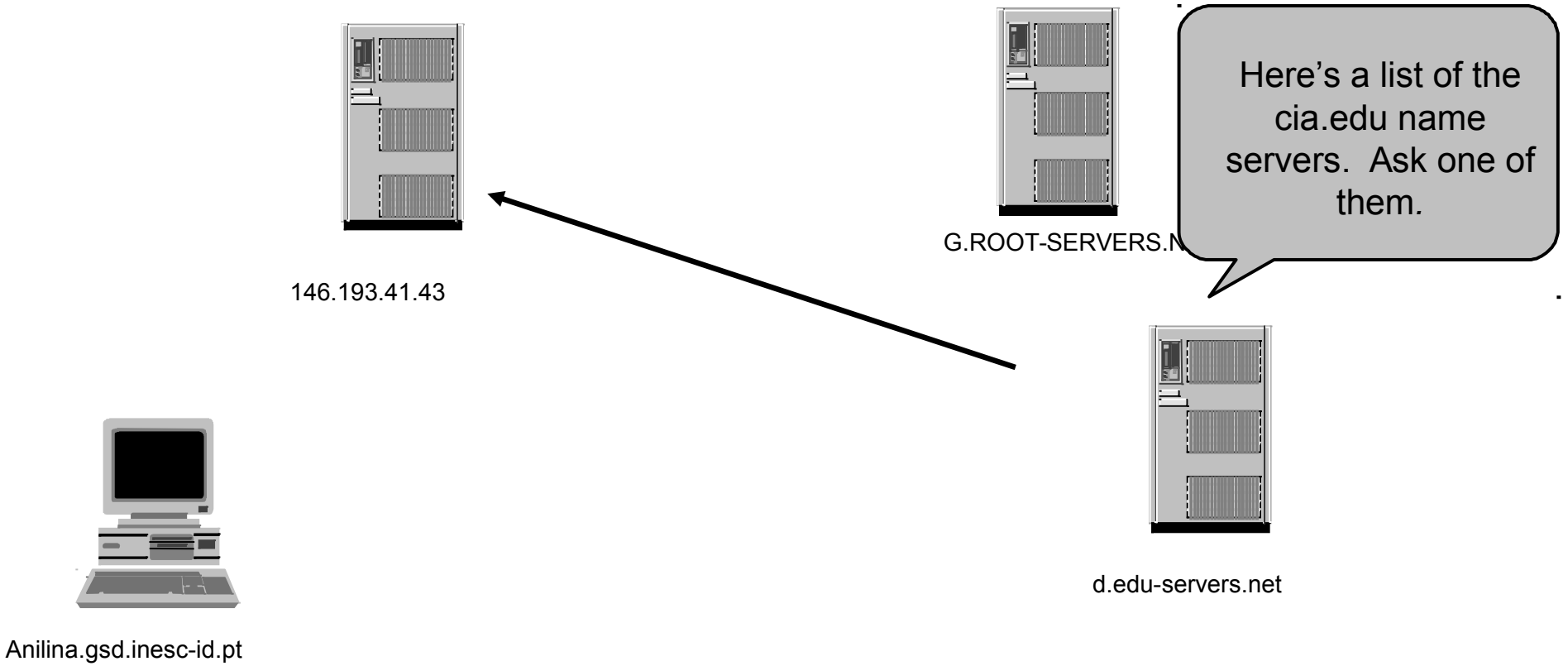
- The name server *dakota* asks a *edu* name server, *d*, for www.cia.edu*'s* address




```
ping www.nominum.com.
```

The Resolution Process

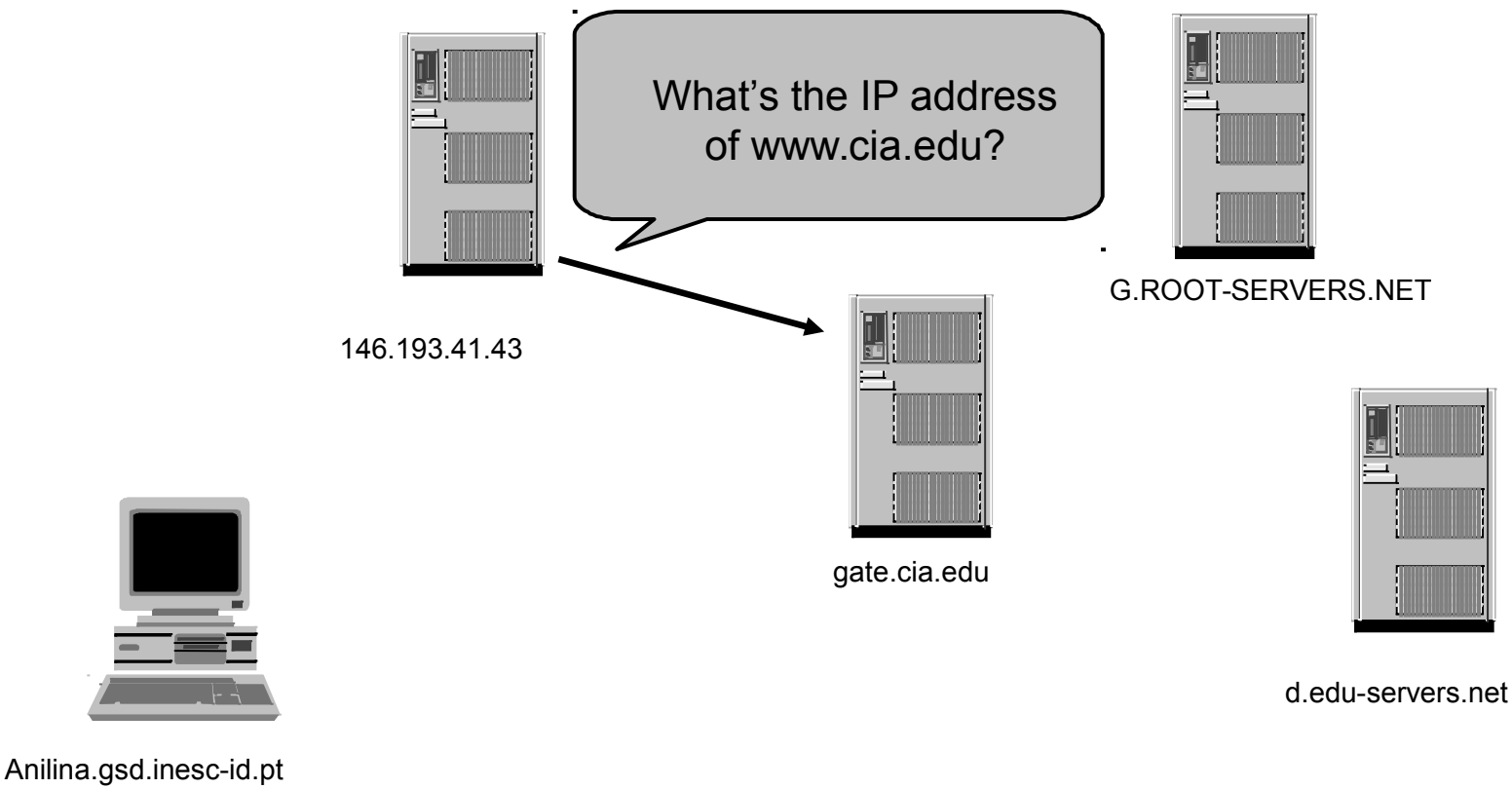
- The *edu* name server *d* refers 146.193.41.43 to the *cia.edu* name servers




```
ping www.nominum.com.
```

The Resolution Process

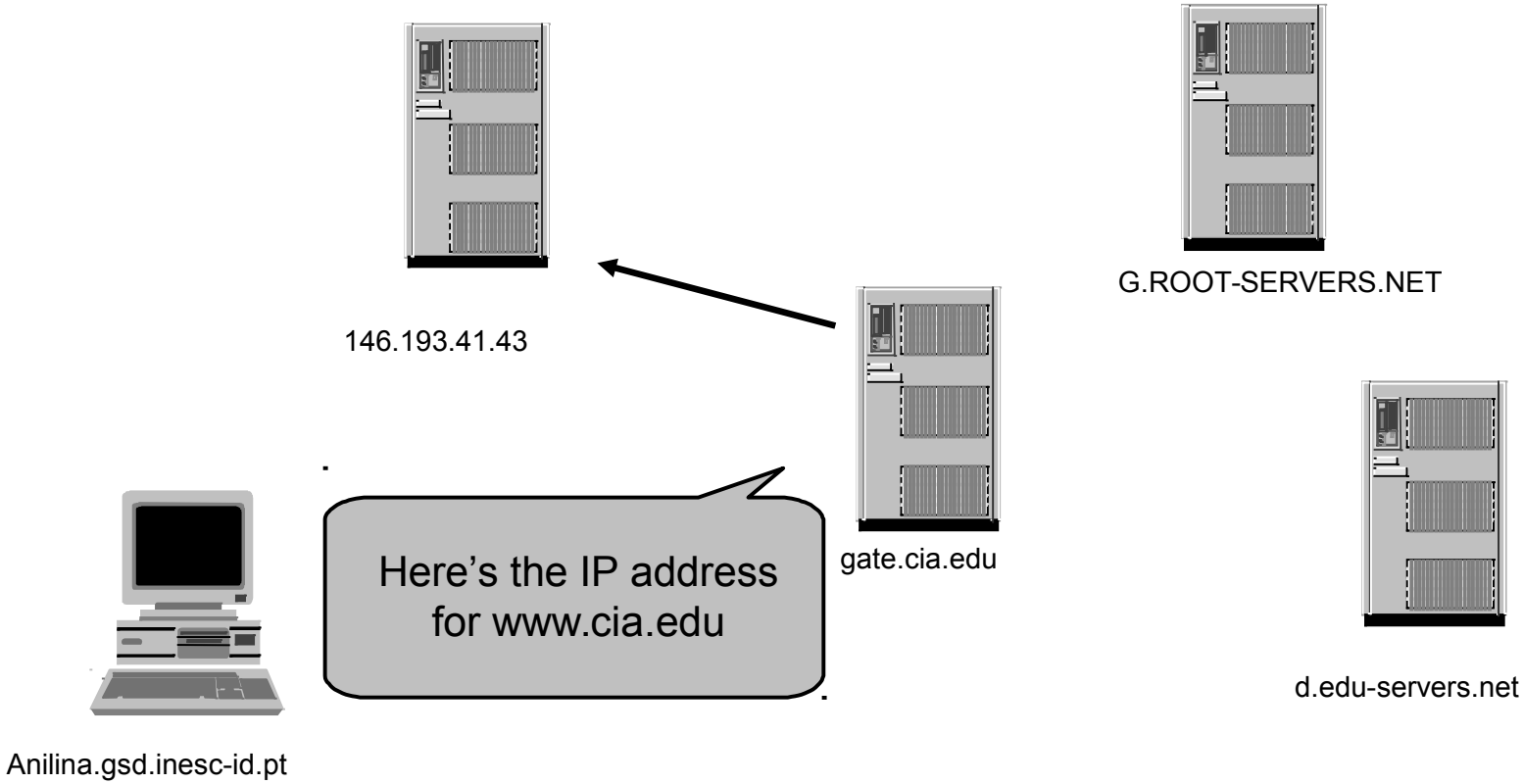
- The name server *146.193.41.43* asks a *cia.edu* name server, *gate.cia.edu*, for www.cia.edu *'s* address




```
ping www.cia.edu.
```

The Resolution Process

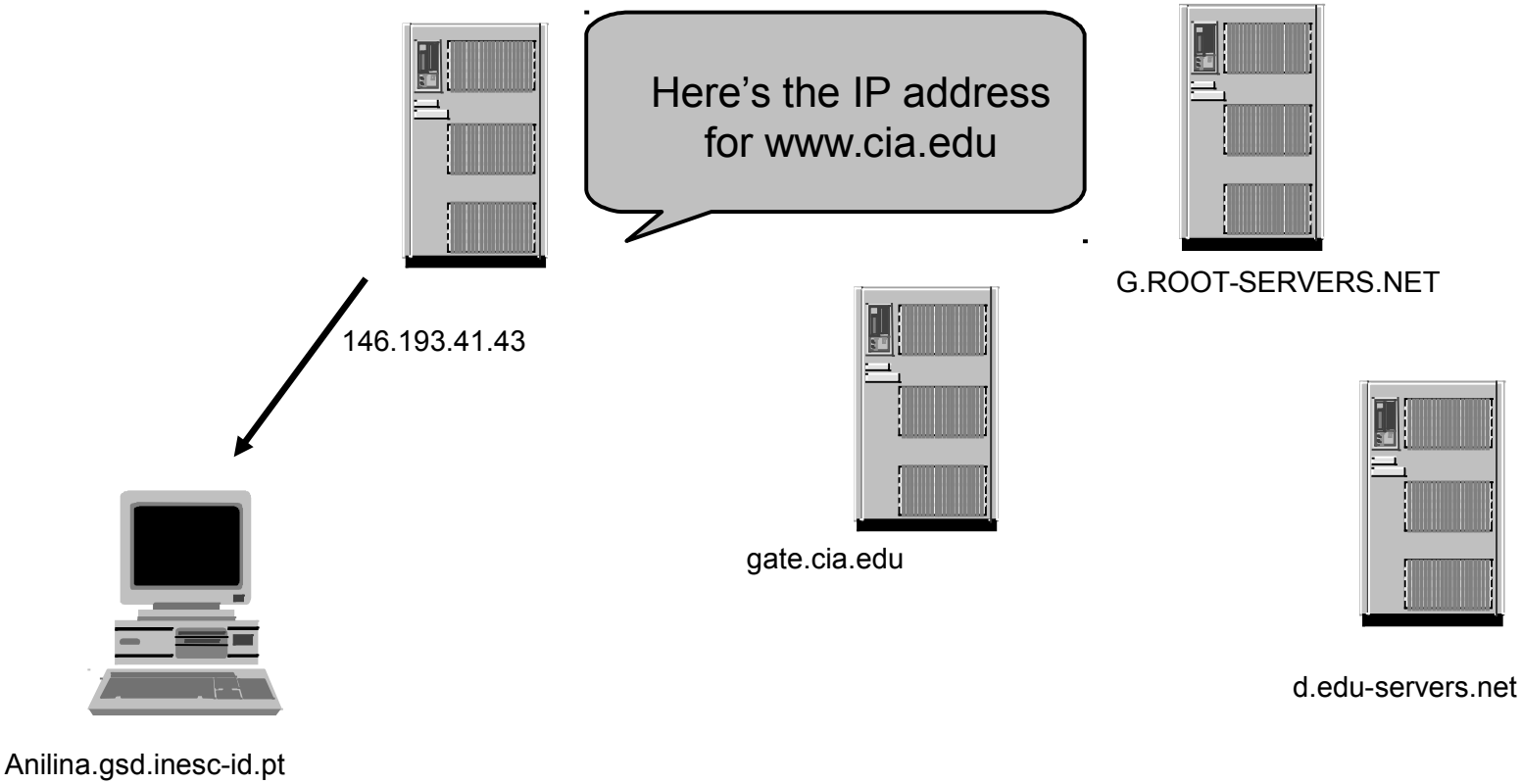
- The cia.edu name server *gate.cia.edu* responds with *www.cia.edu's* address




```
ping www.cia.edu.
```

The Resolution Process

- The name server *146.193.41.43* responds to *anilina* with *www.cia.edu's* address




```
Ping www.cia.edu.
```

Resolution Process (Caching)

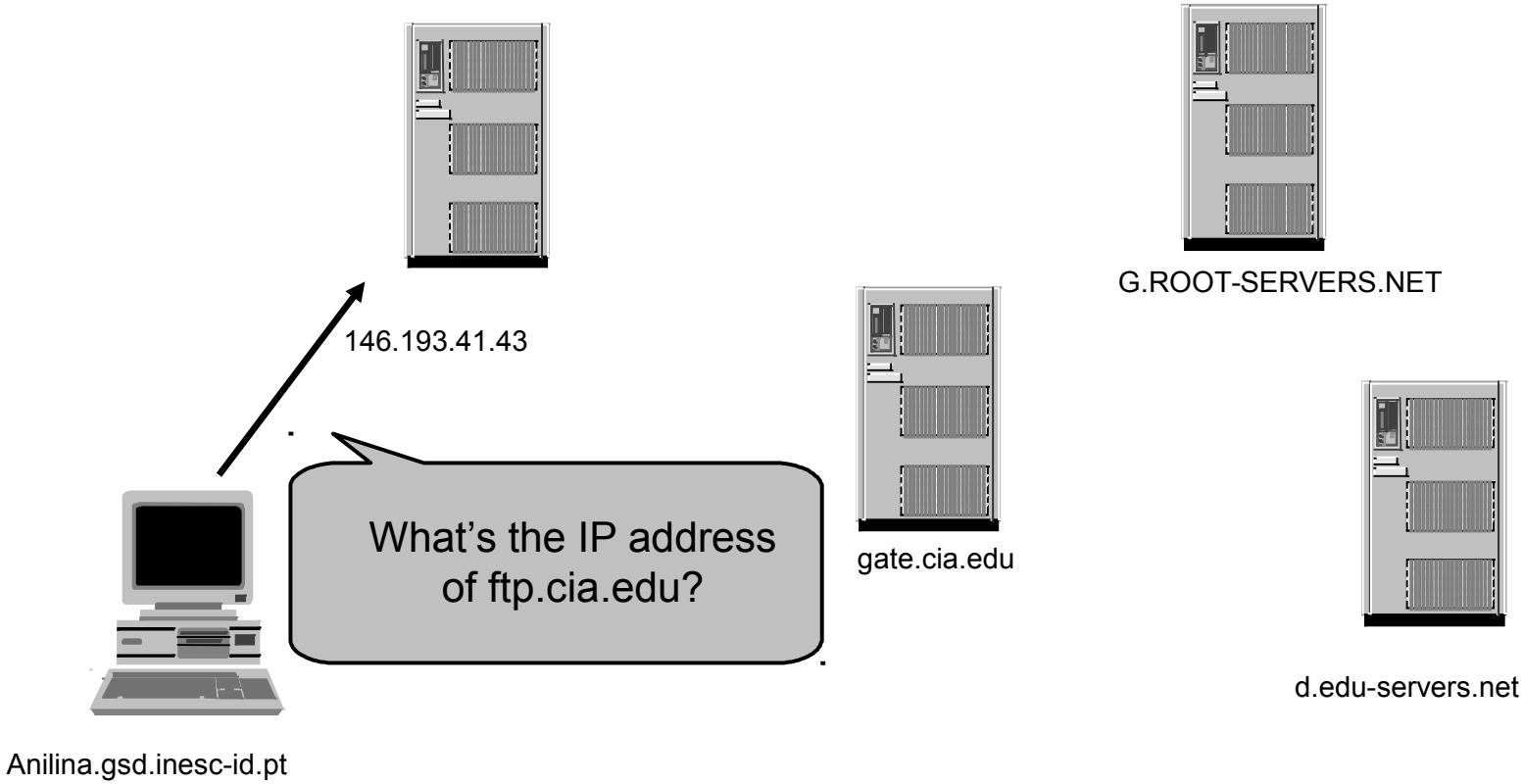
- After the previous query, the name server *146.193.41.43* knows:
  - The names and IP addresses of the *edu* name servers
  - The names and IP addresses of the *cia.edu* name servers
  - The IP address of *www.cia.edu*
- Let’s look at the resolution process again

Anilina.gsd.inesc-id.pt

```
ping ftp.cia.edu.
```

Resolution Process (Caching)

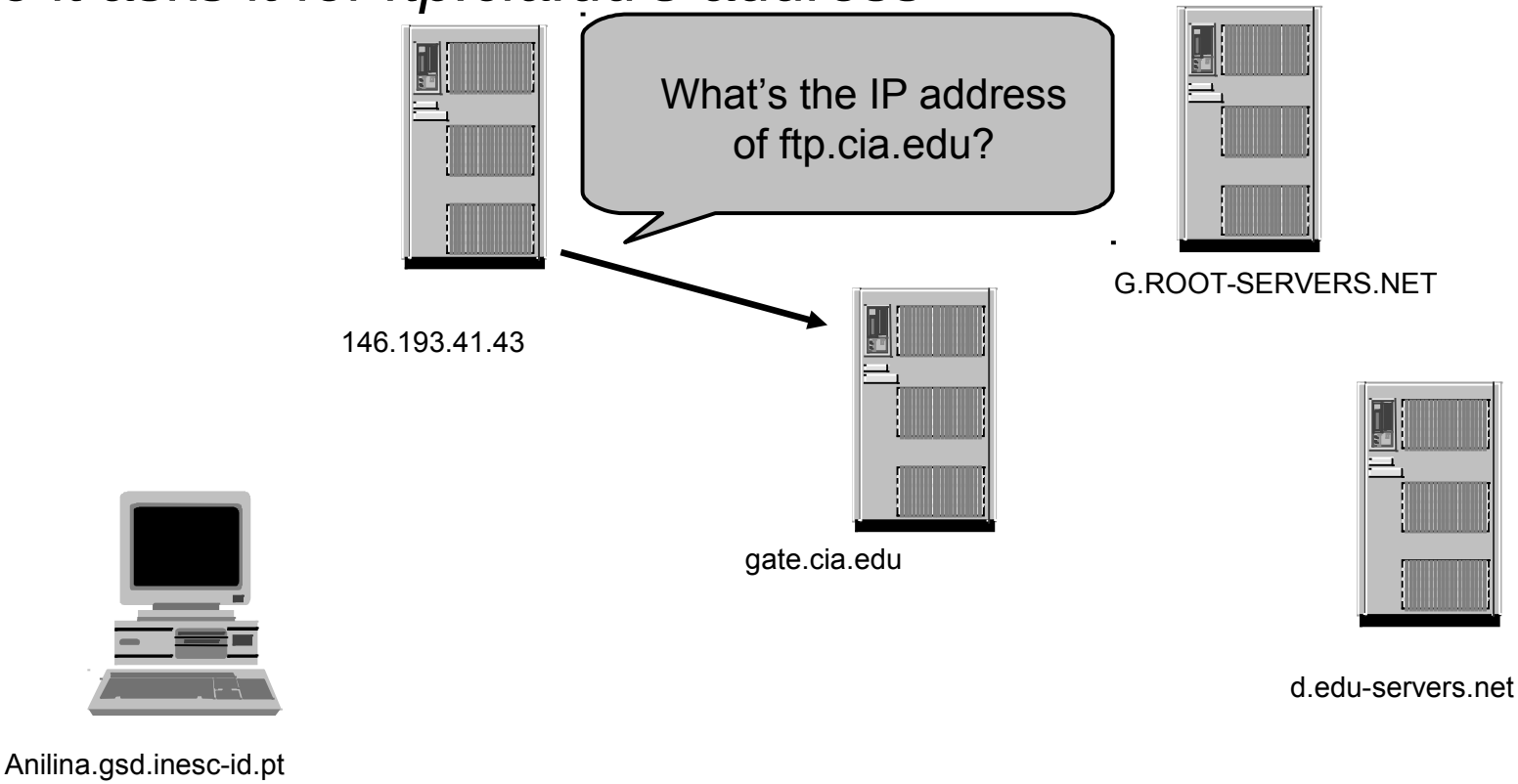
The workstation *anilina* asks its configured name server, 146.193.41.43, for *ftp.cia.edu's* address




```
ping ftp.cia.edu.
```

Resolution Process (Caching)

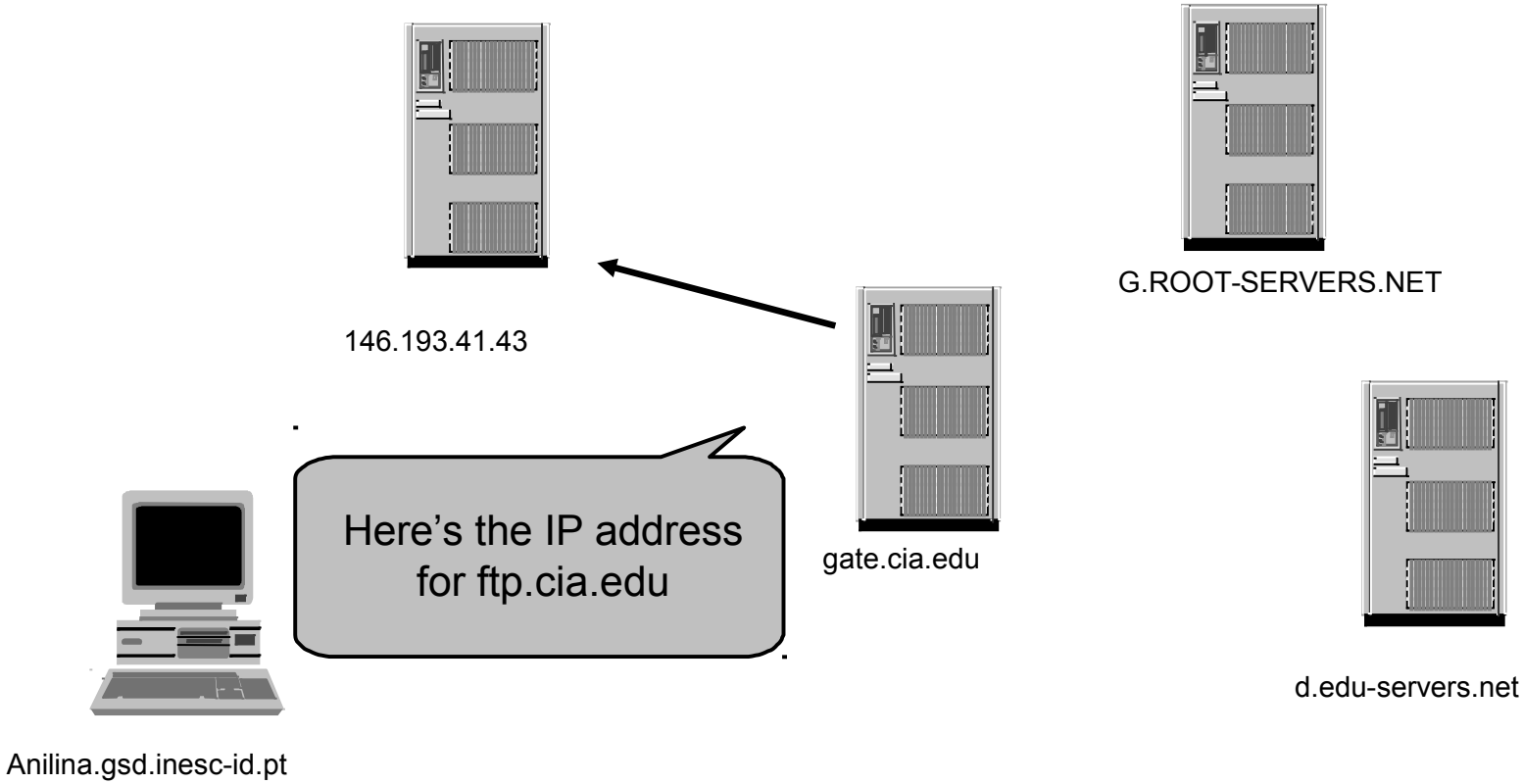
- *146.193.41.43* has cached a NS record indicating *gate.cia.edu* is an *cia.edu* name server,
- so it asks it for *ftp.cia.du's* address




```
ping ftp.cia.edu.
```

Resolution Process (Caching)

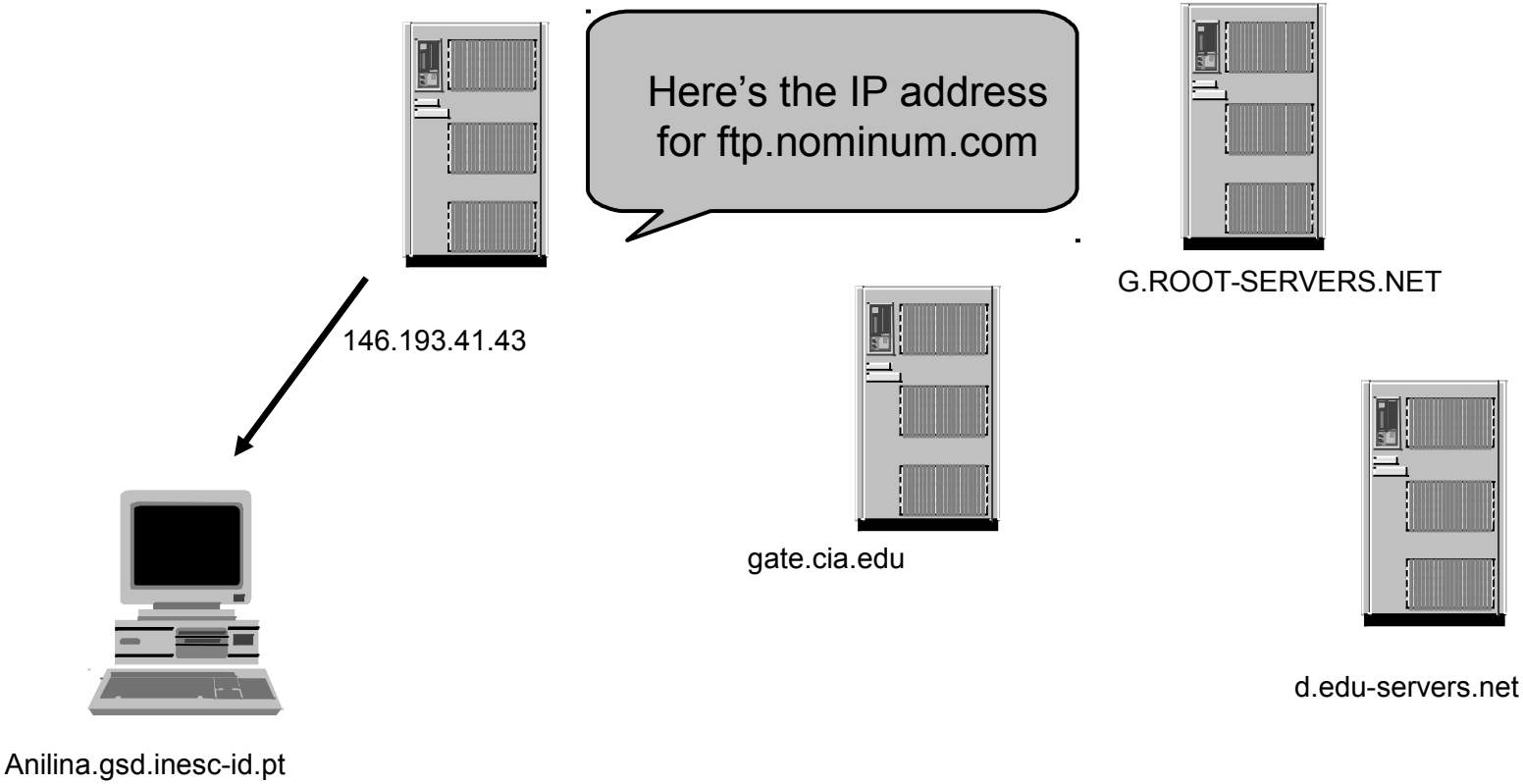
- The cia.*com* name server *gate.cia.edu* responds with *ftp.cia.edu's* address




```
ping ftp.cia.edu.
```

Resolution Process (Caching)

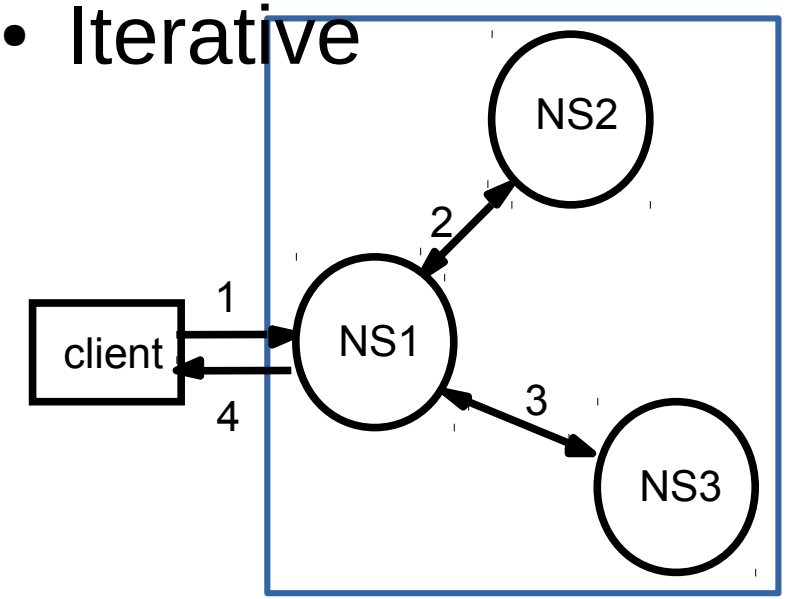
- The name server *146.193.41.43* responds to *anilina* with *ftp.cia.edu's* address



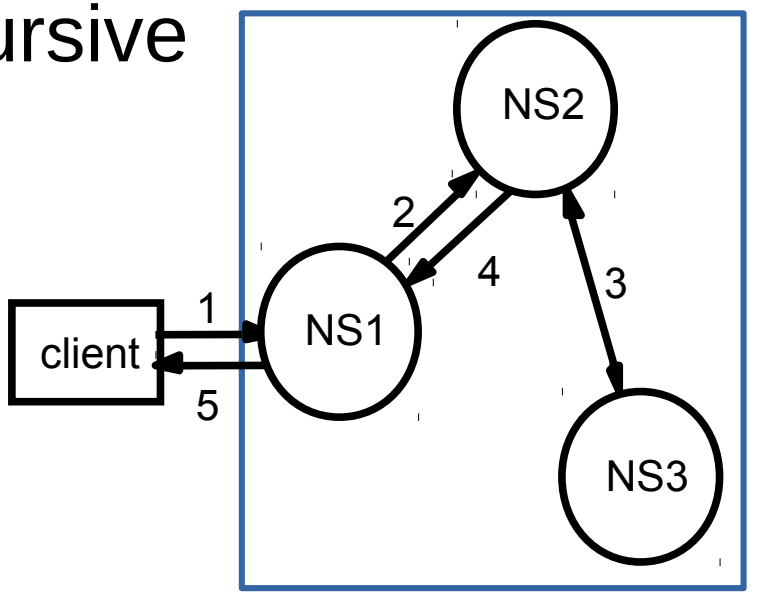

```
ping ftp.cia.edu.
```

Iterative vs recursive

- Iterative



- Recursive



- DNS server never maintains state
- Caching is local
  - others cannot benefit from it
- resource intensive
  - DNS server maintains state
- Cache namespace
  - helpful for other queries

Performance

- DNS can handle the load
- DNS root servers get approximately 3000 queries per second
  - Empirical proofs (DDoS attacks) show root name servers can handle 50,000 queries per second
  - Limitation is network bandwidth, not the DNS protocol
- in-addr.arpa zone, which translates numbers to names, gets about 2000 queries per second
- DNS is a very lightweight protocol
  - Simple query – response
- Any performance limitations are the result of network limitations

Iterative vs recursive

- Performance-wise, which is better?
  - Recursive method puts higher performance demand on each name server
- Which works better with caching?
  - Recursive method works better with caching
- How about communication cost?
  - Recursive method can reduce communication cost

Reliability

- DNS servers are replicated
  - Name service available if ≥ one replica is up
  - Queries can be load balanced between replicas
- UDP used for queries
  - Need reliability ✉ must implement this on top of UDP!
  - Why not just use TCP?
- Try alternate servers on timeout
  - Exponential backoff when retrying same server
- Same identifier for all queries
  - Don’t care which server responds