Chapter 5

Structured Naming

Internet Naming Service: DNS*

*referred to slides by David Conrad at nominum.com

Overview

- Introduction to the DNS
- DNS Components
- DNS Structure and Hierarchy
- The DNS in Context

DNS History (1)

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

DNS History (2)

- 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- In 1984, Paul Mockapetris released the first version (RFCs 882 and 883, superseded by 1034 and 1035 …)

The DNS is…

- 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

The DNS is also…

- 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 as a Lookup Mechanism

- Users generally prefer names to numbers
- Computers prefer numbers to names
- DNS provides the mapping between the two
  - I have “x”, give me “y”

DNS as a Database

- Keys to the database are “domain names”
  - www.foo.com, 18.in-addr.arpa, 6.4.e164.arpa
- Over 200,000,000 domain names stored
- Each domain name contains one or more attributes
  - Known as “resource records”
- Each attribute individually retrievable

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 cachable to 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 are propagated to replicas according to timing set by the zone administrator
- Cached data expires according to timeout set by zone administrator

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
  - If UDP, DNS protocol handles retransmission, sequencing, etc.

Dynamicity

- Database can be updated dynamically
  - Add/delete/modify of any record
  - Only master can be dynamically updated

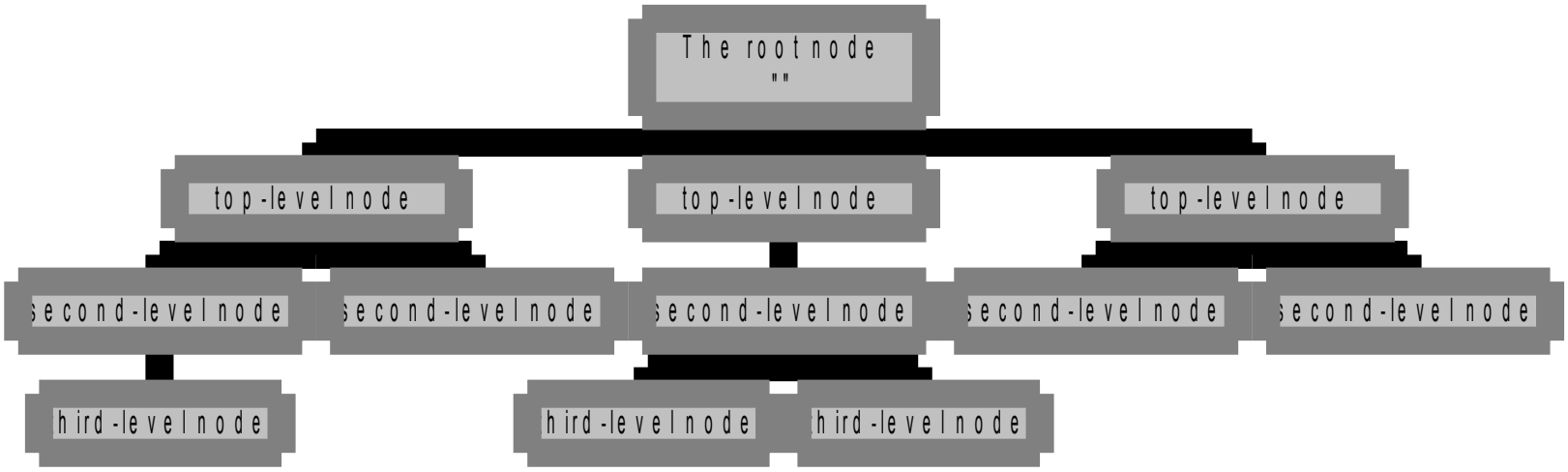
- Modification of the master database triggers replication

Overview

- Introduction to the DNS
- DNS Components
  - The name space
  - The servers
  - The resolvers
- DNS Structure and Hierarchy
- The DNS in Context

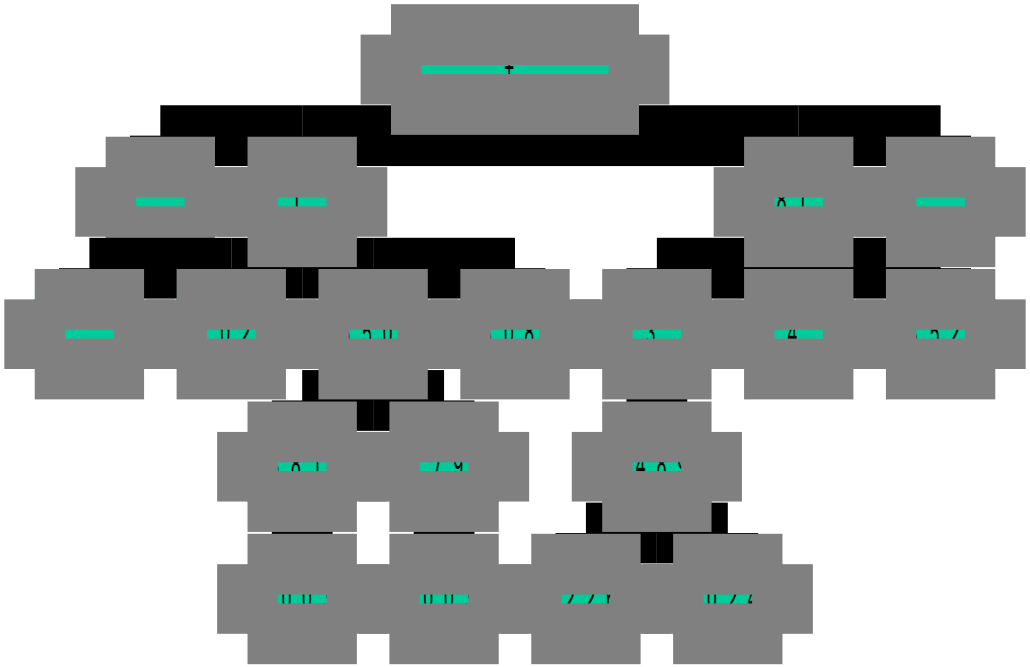
The Name Space

- 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 “”



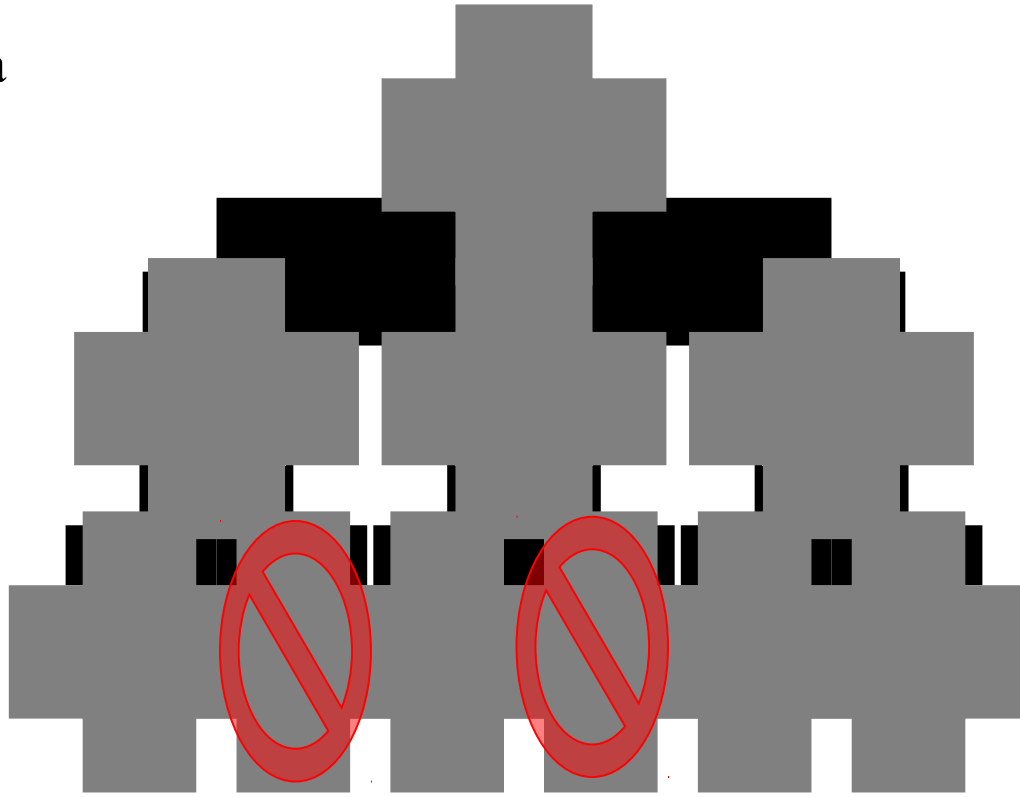
An Analogy – E.164

- Root node maintained by the ITU (call it “+”)
- Top level nodes = country codes (1, 81, etc)
- Second level nodes = regional codes (1-402, 81-3, etc.)



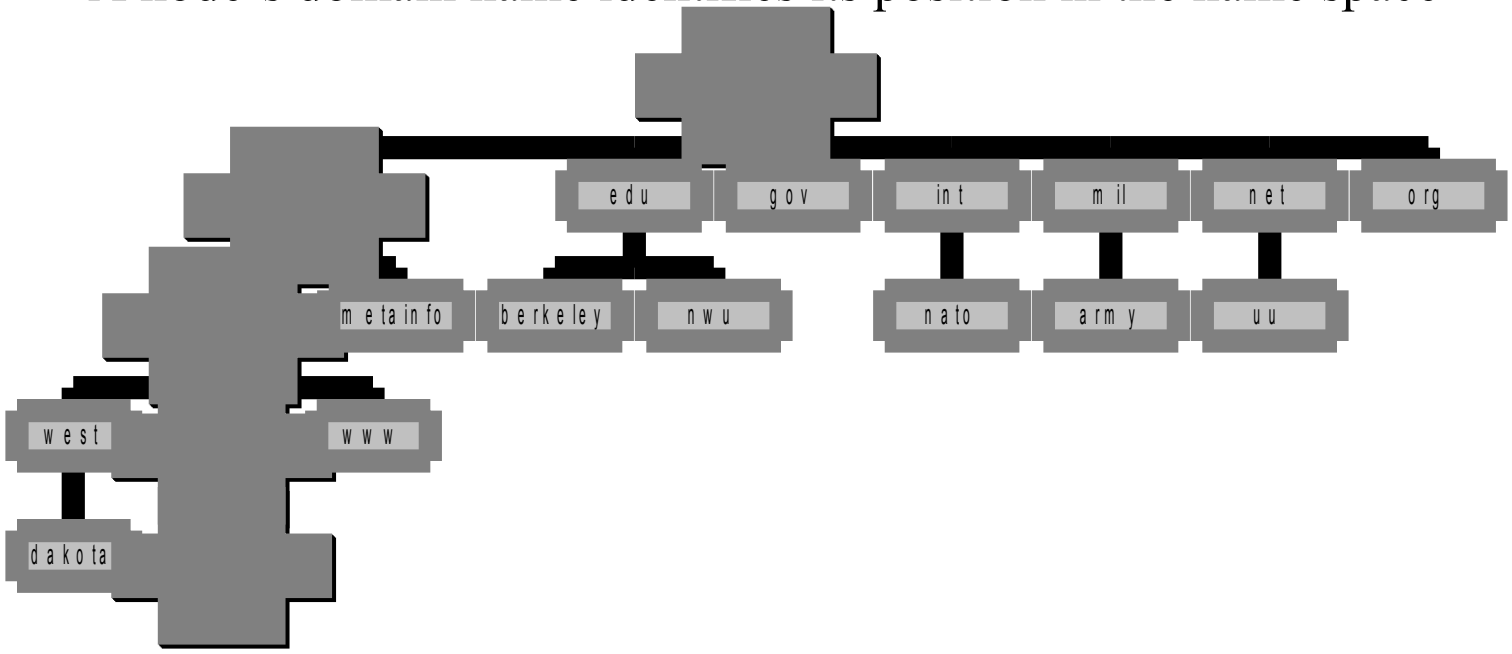
Labels

- Each node in the tree must have a label
  - A string of up to 63 bytes
  - RFCs 852 and 1123 define legal characters for “hostnames”
    - A-Z, 0-9, and “-” only with a-z and A-Z treated as the same
- Sibling nodes must have unique labels
- The null label is reserved for the root node



Domain Names

- 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



Subdomains

- One domain is a subdomain of another if its domain name ends in the other's domain name
  - So *sales.nominum.com* is a subdomain of
    - *nominum.com & com*
  - *nominum.com* is a subdomain of *com*

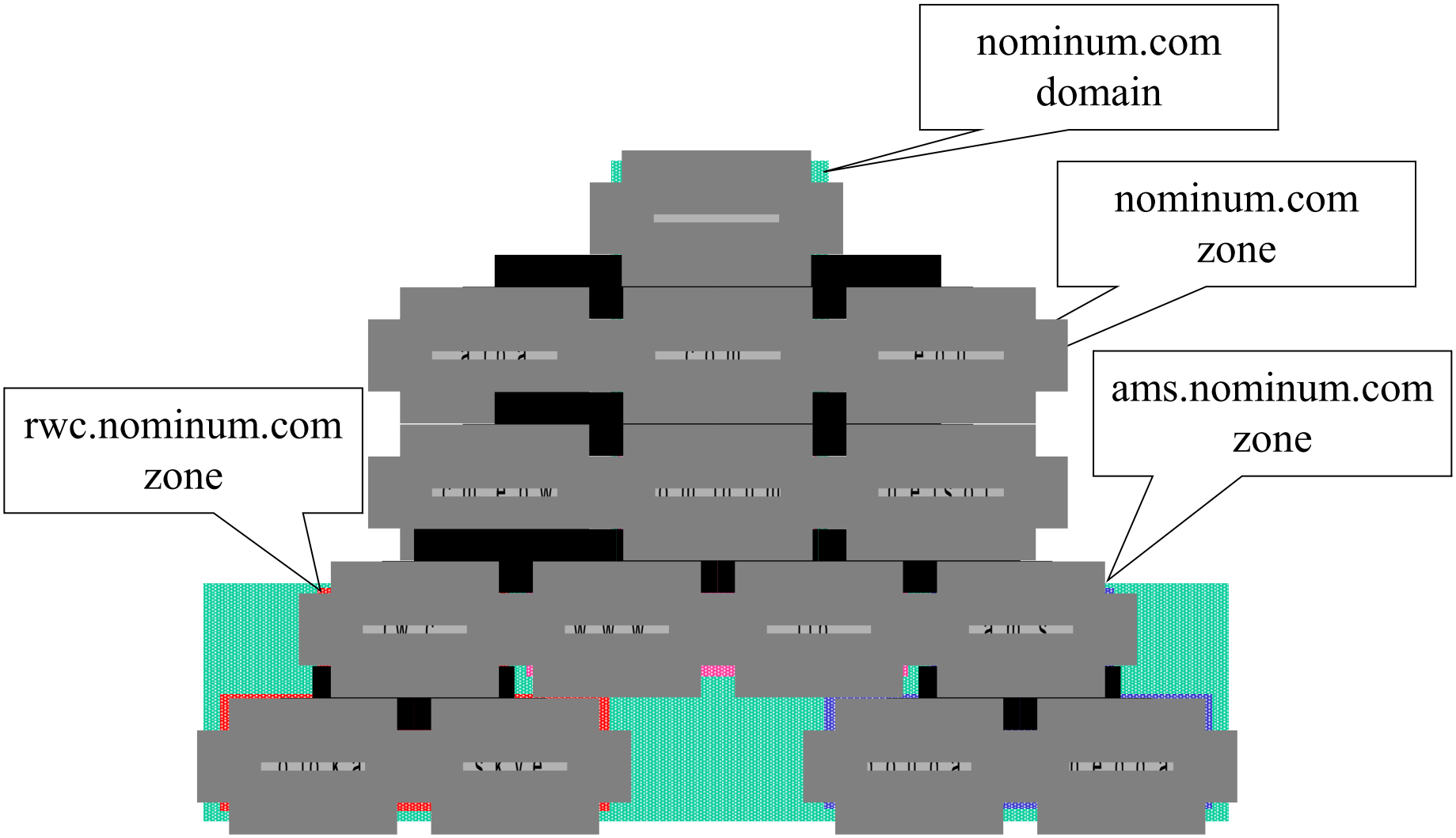
Delegation

- 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

Delegation Creates 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

Dividing a Domain into Zones



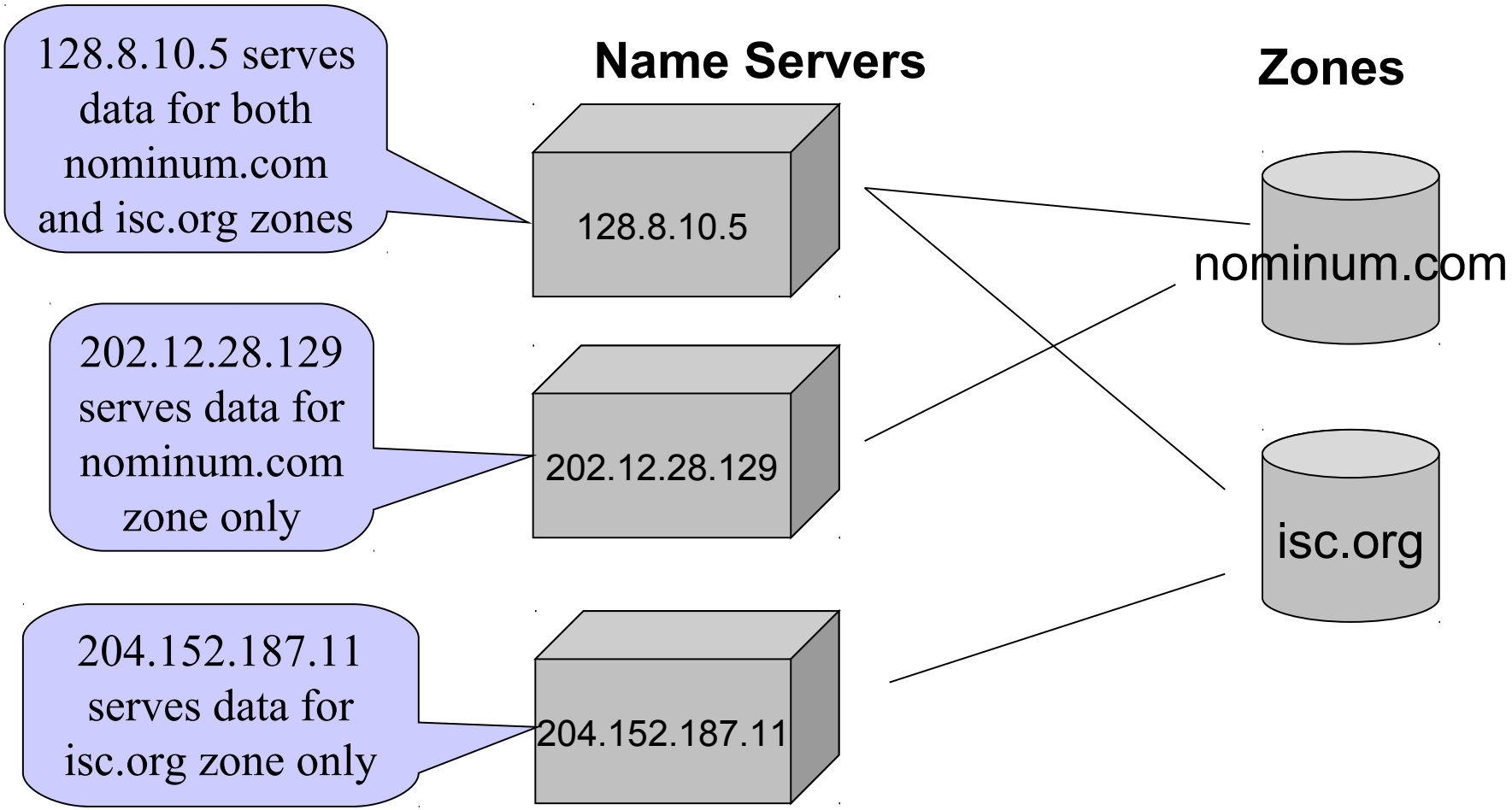
Overview

- Introduction to the DNS
- DNS Components
  - The name space
  - The servers
  - The resolvers
- DNS Structure and Hierarchy
- The DNS in Context

Name Servers

- Name servers store information about the name space in units called “zones”
  - The name servers that load a complete zone are said to “have authority for” or “be authoritative for” the zone
- Usually, more than one name server are authoritative for the same zone
  - This ensures redundancy and spreads the load
- Also, a single name server may be authoritative for many zones

Name Servers and Zones



Types of Name Servers

- Two main types of servers
  - Authoritative – maintains the data
    - Master – where the data is edited
    - Slave – where data is replicated to
  - Caching – stores data obtained from an authoritative server
- No special hardware necessary

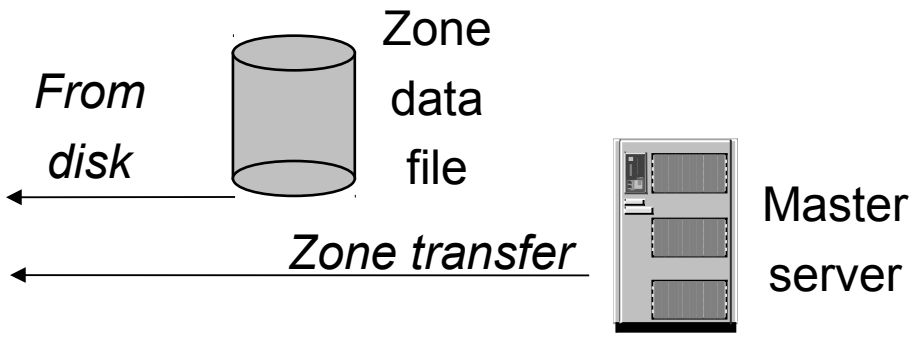
Name Server Architecture

- You can think of a name server as part of:
  - *database server,* answering queries about the parts of the name space it knows about (i.e., is authoritative for),
  - *cache,* temporarily storing data it learns from other name servers, and
  - *agent,* helping resolvers and other name servers find data

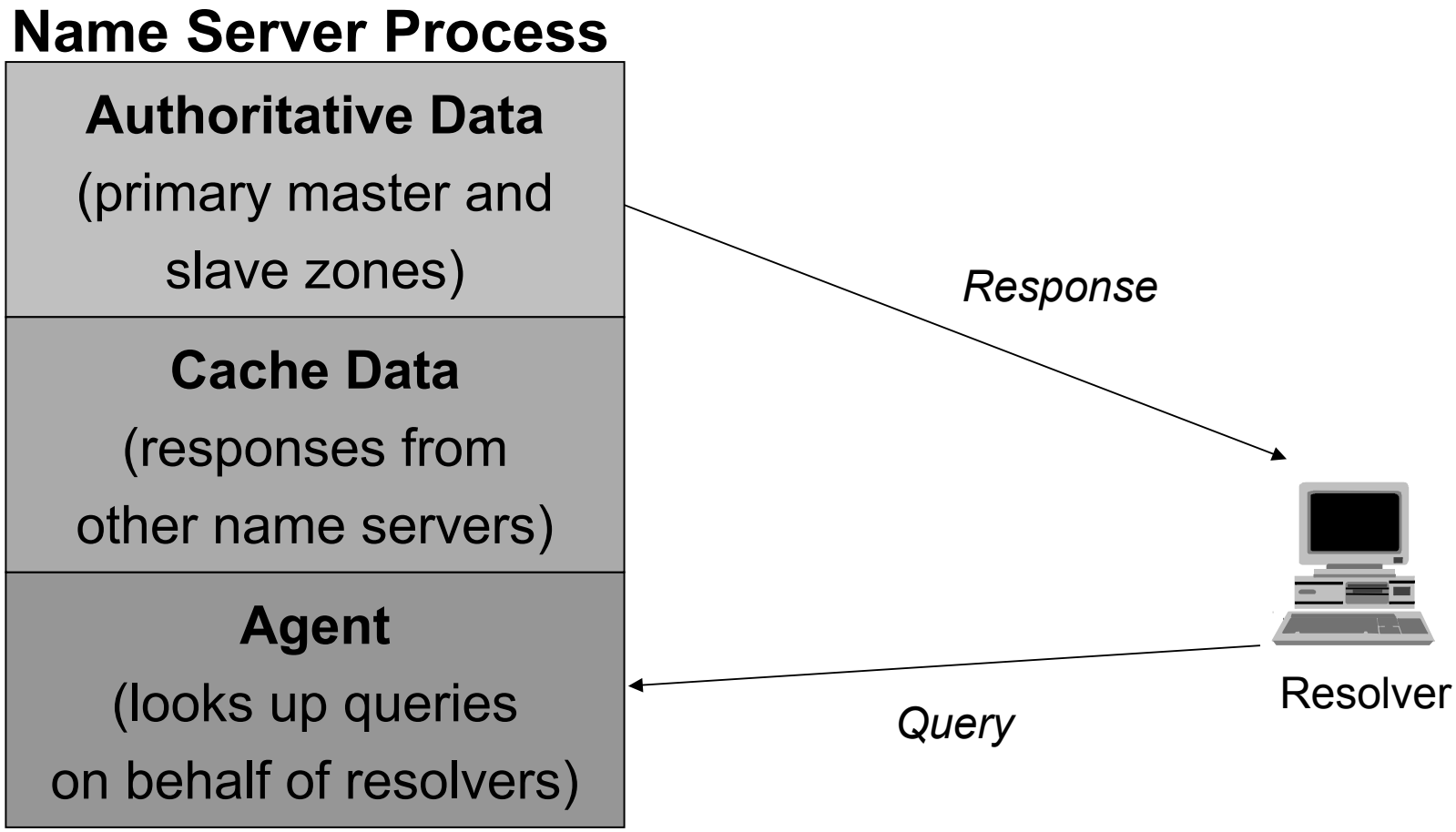
Name Server Architecture

Name Server Process

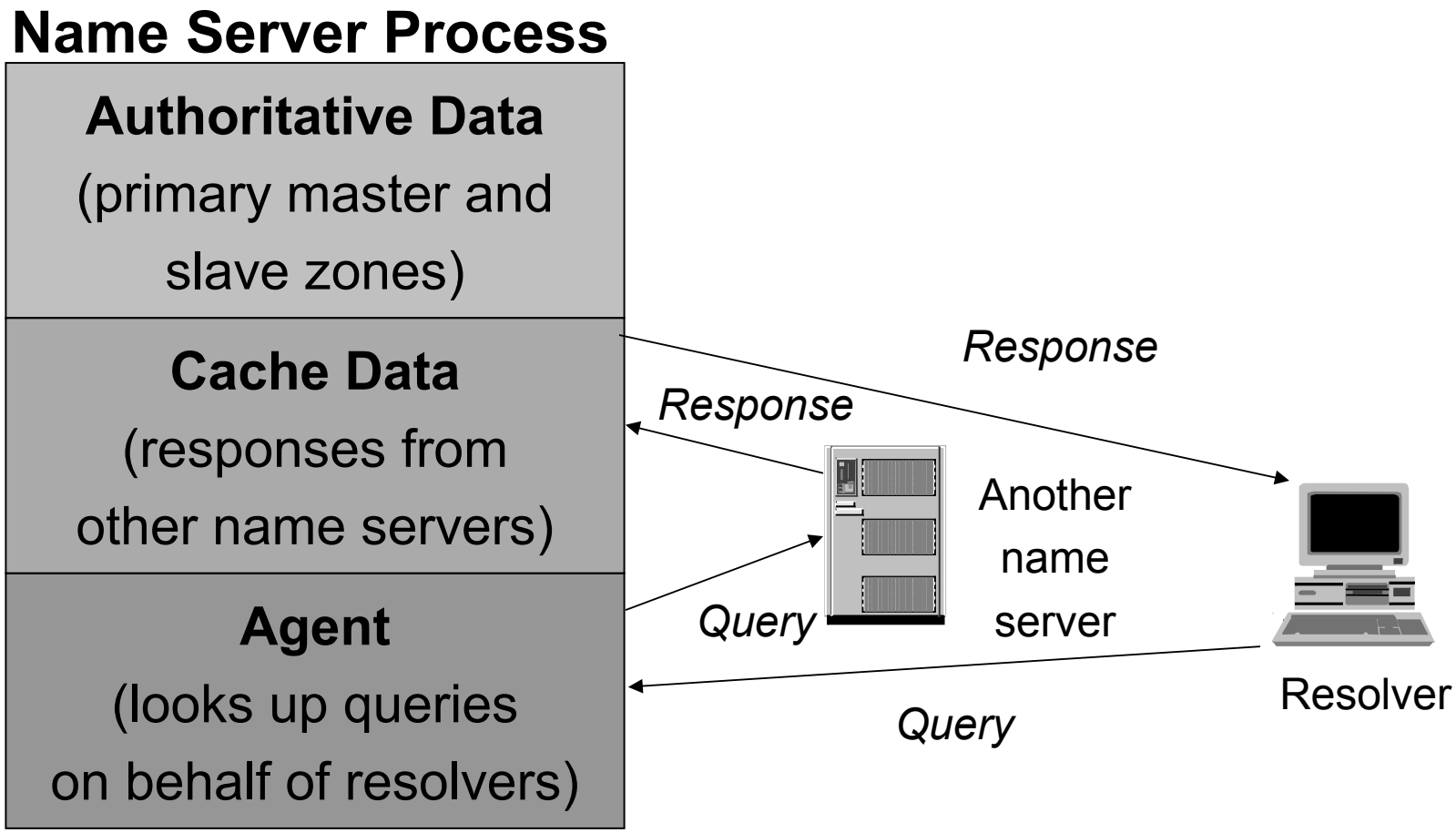
Authoritative Data (primary master and slave zones)
Cache Data (responses from other name servers)
Agent (looks up queries on behalf of resolvers)



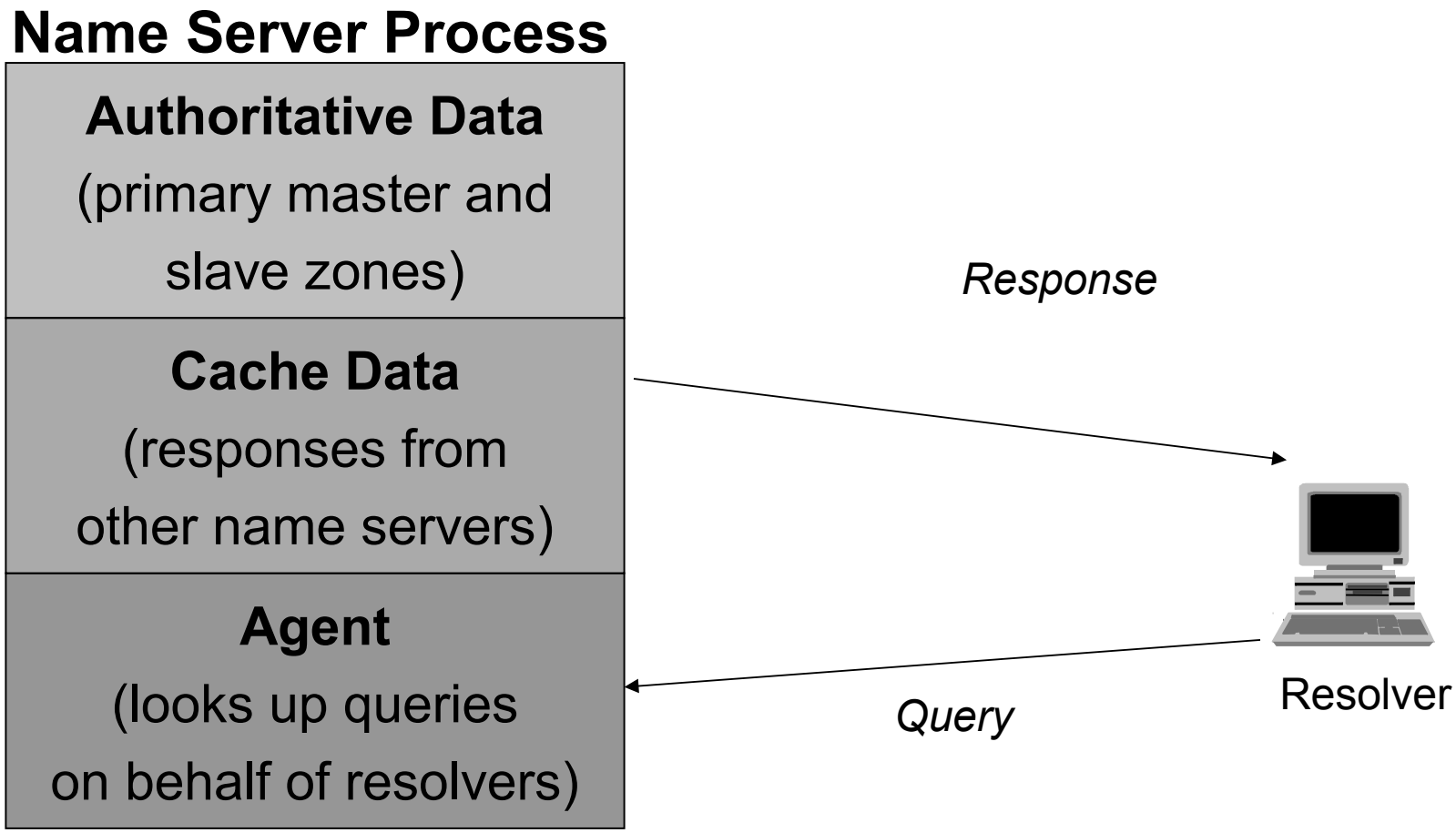
Authoritative Data



Using Other Name Servers



Cached Data


Name Server Process
Authoritative Data
(primary master and
slave zones)
Cache Data
(responses from
other name servers)
Agent
(looks up queries
on behalf of resolvers)
Response
Query
Resolver

Overview

- Introduction to the DNS
- DNS Components
  - The name space
  - The servers
  - The resolvers
- DNS Structure and Hierarchy
- The DNS in Context

Name Resolution

- *Name resolution* is the process by which resolvers and name servers cooperate to find data in the name space
- Closure mechanism for DNS?
  - Starting point: the names and IP addresses of the name servers for the root zone (the "root name servers")
  - The root name servers know about the top-level zones and can tell name servers whom to contact for all TLDs

Name Resolution

- A DNS query has three parameters:
  - A domain name (e.g., *www.nominum.com*),
    - Remember, every node has a domain name!
  - A class (e.g., *IN*), and
  - A type (e.g., *A*)
  - http://network-tools.com/nslook/
- Upon receiving a query from a resolver, a name server
  - 1) looks for the answer in its authoritative data and its cache
  - 2) If step 1 fails, the answer must be looked up

The Resolution Process

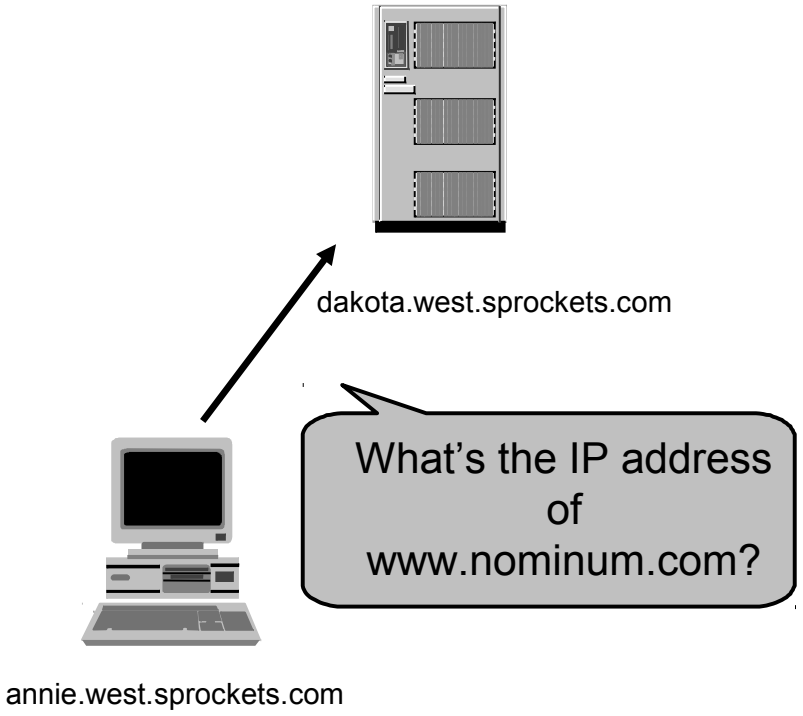
- Let's look at the resolution process step-by-step:

annie.west.sprockets.com

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

The Resolution Process

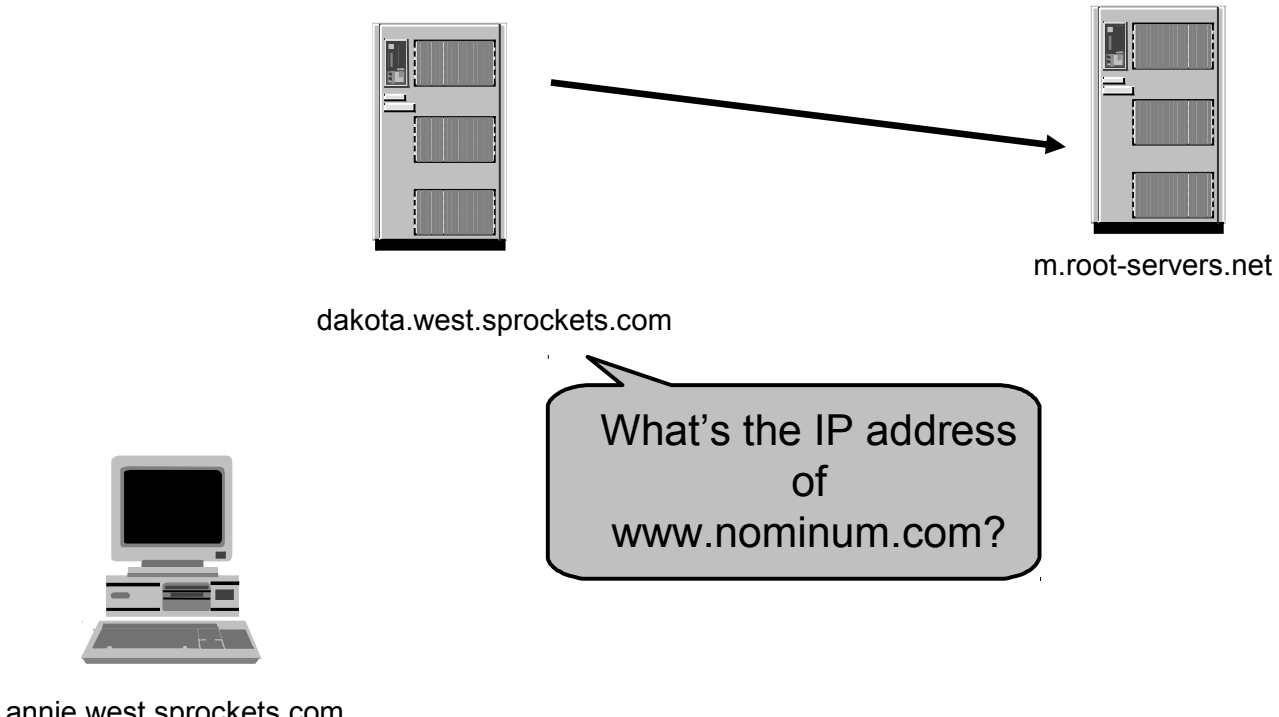
- The workstation *annie* asks its configured name server, *dakota,* for *www.nominum.com's* address




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

The Resolution Process

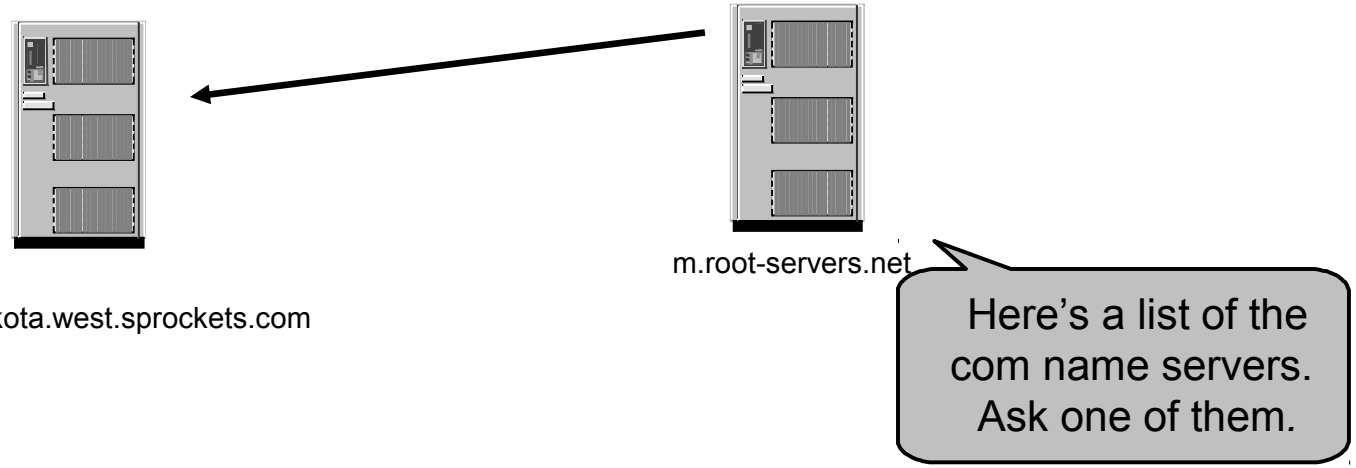
- The name server *dakota* asks a root name server, *m*, for *www.nominum.com's* address




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

The Resolution Process

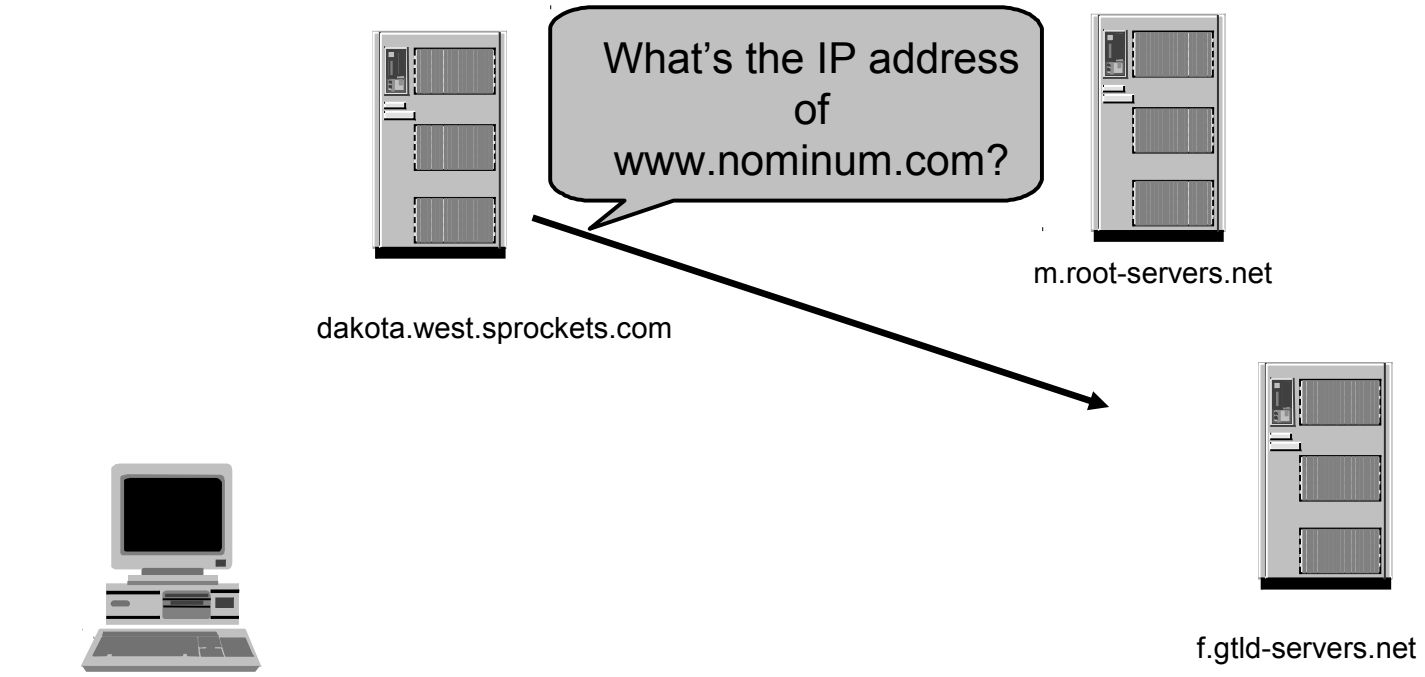
- The root server *m* refers *dakota* to the *com* name servers
- This type of response is called a “referral”




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

The Resolution Process

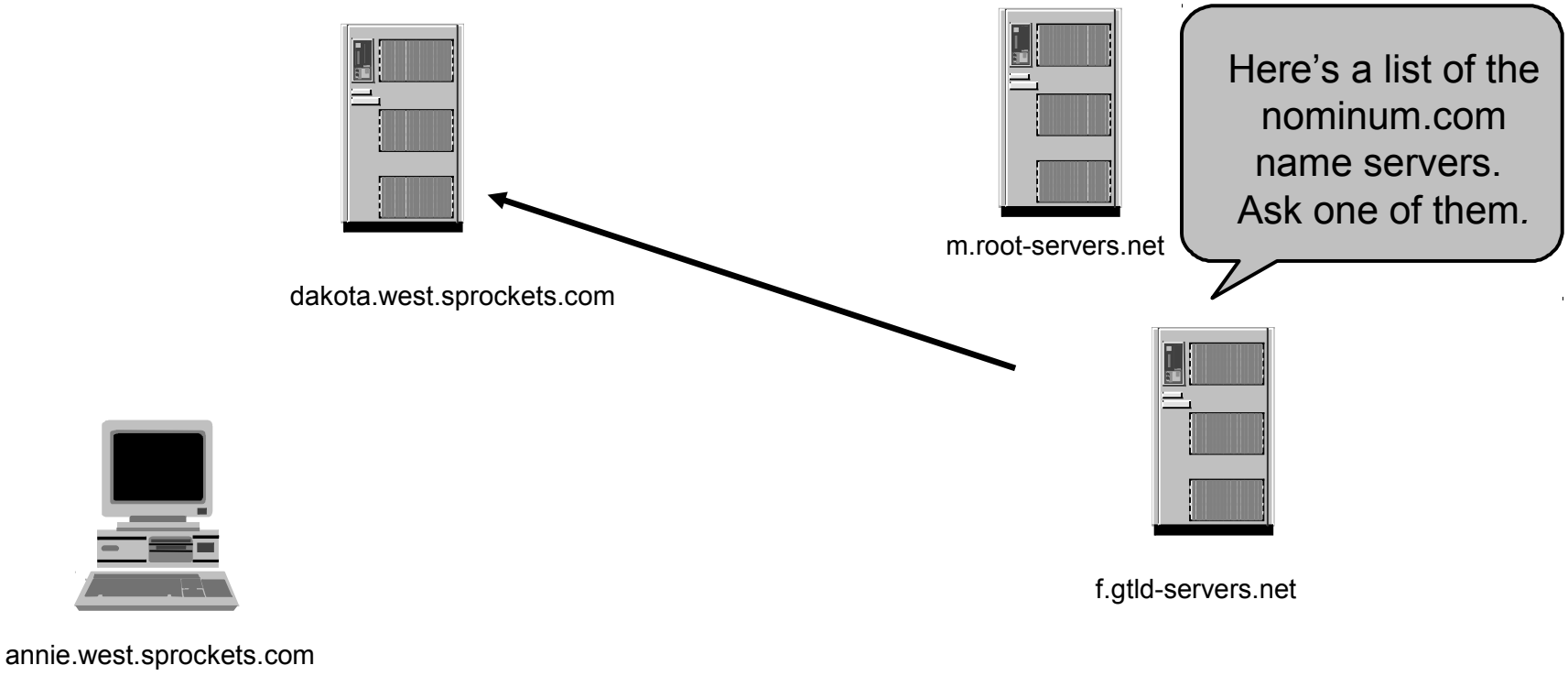
- The name server *dakota* asks a *com* name server, *f*, for *www.nominum.com's* address




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

The Resolution Process

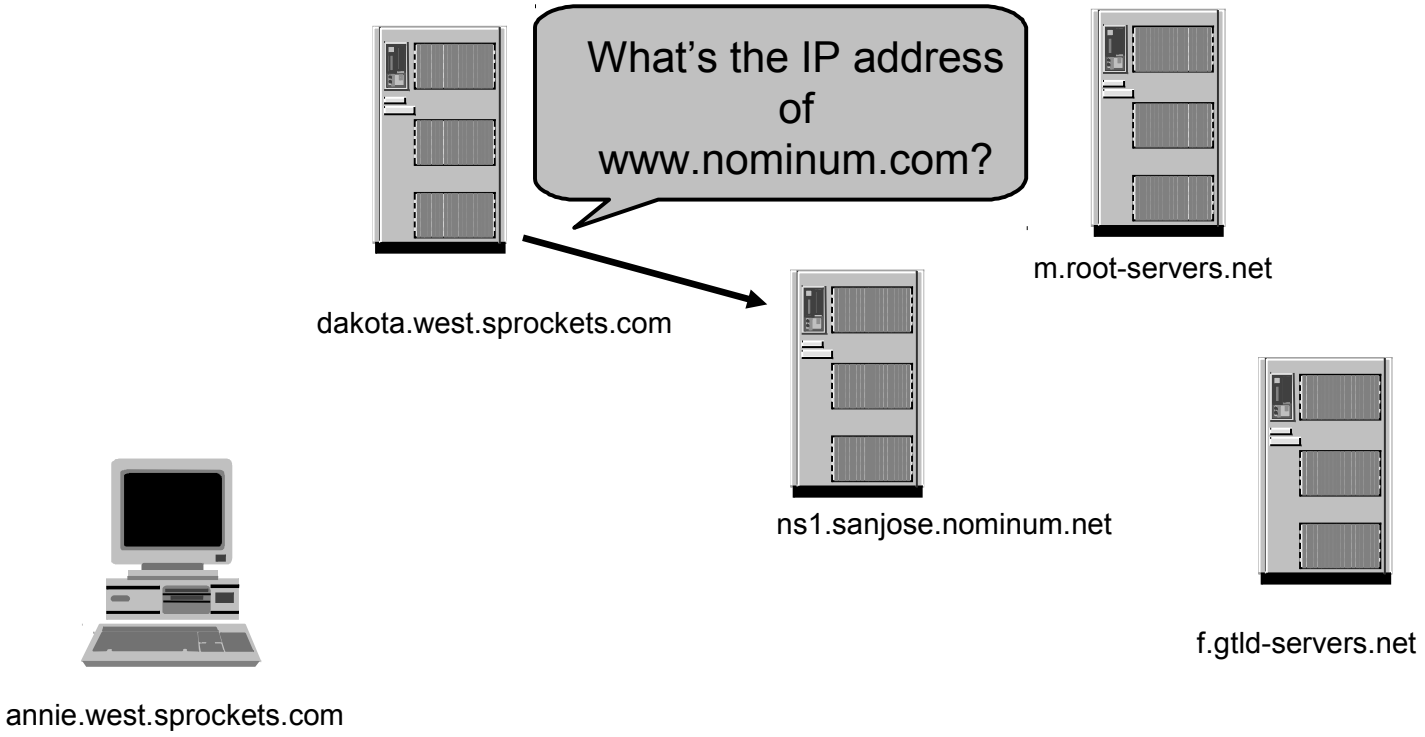
- The *com* name server *f* refers *dakota* to the *nominum.com* name servers




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

The Resolution Process

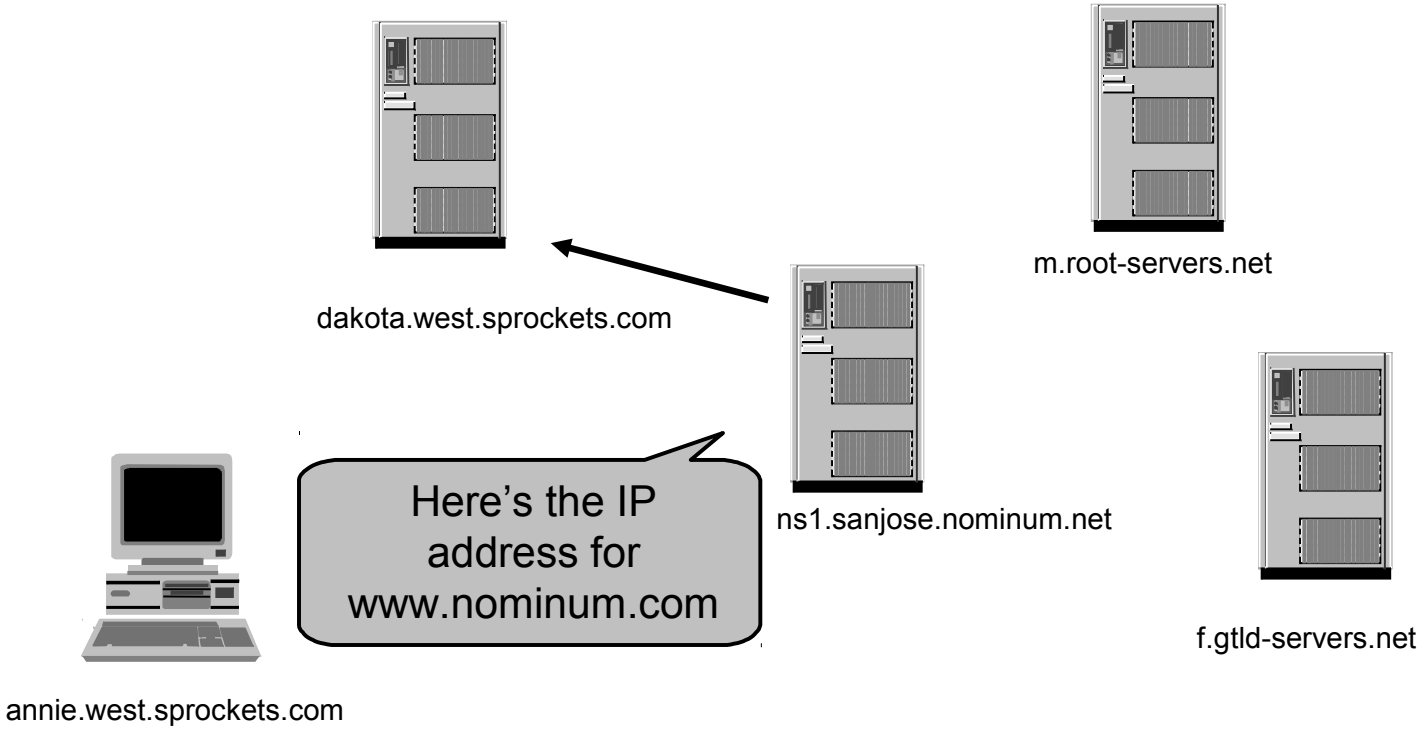
- The name server *dakota* asks a *nominum.com* name server, *ns1.sanjose*, for *www.nominum.com's* address




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

The Resolution Process

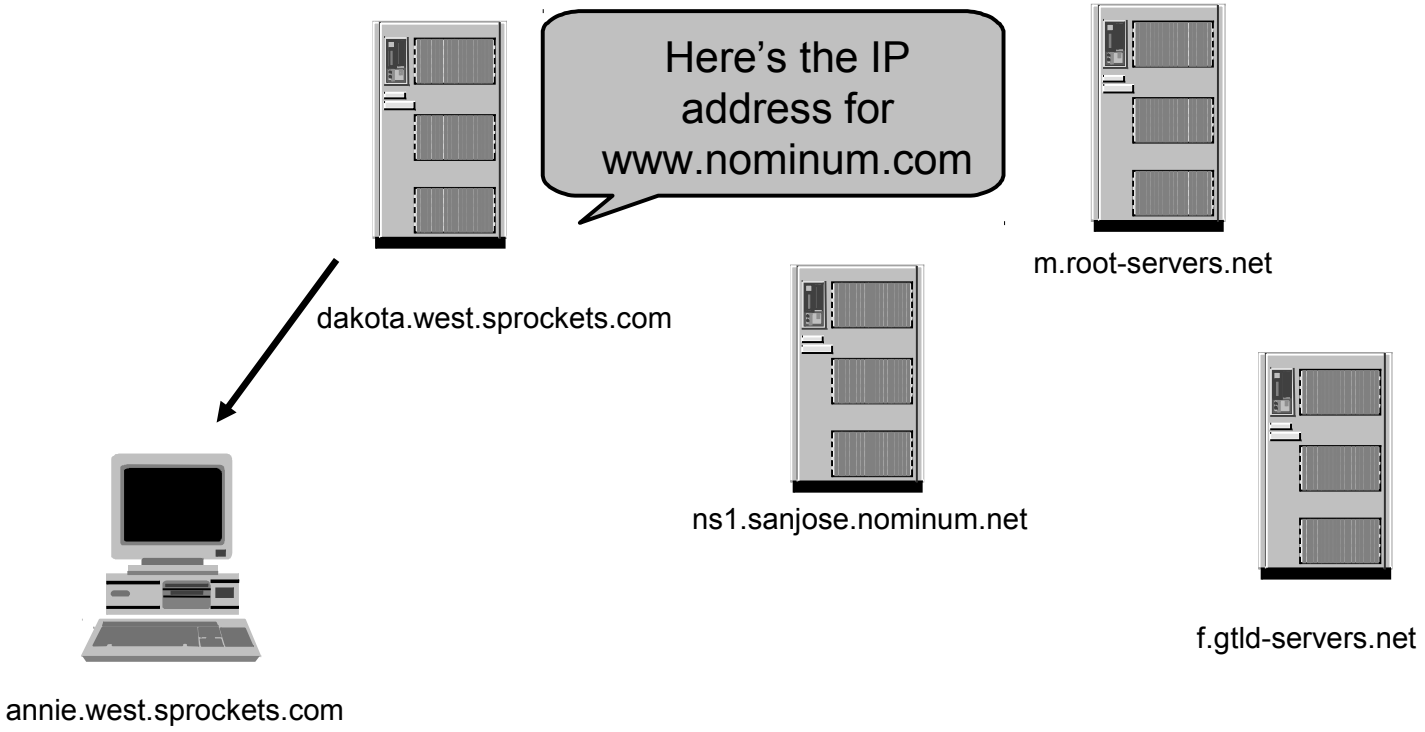
- The *nominum.com* name server *ns1.sanjose* responds with *www.nominum.com's* address




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

The Resolution Process

- The name server *dakota* responds to *annie* with *www.nominum.com's* address




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

Resolution Process (Caching)

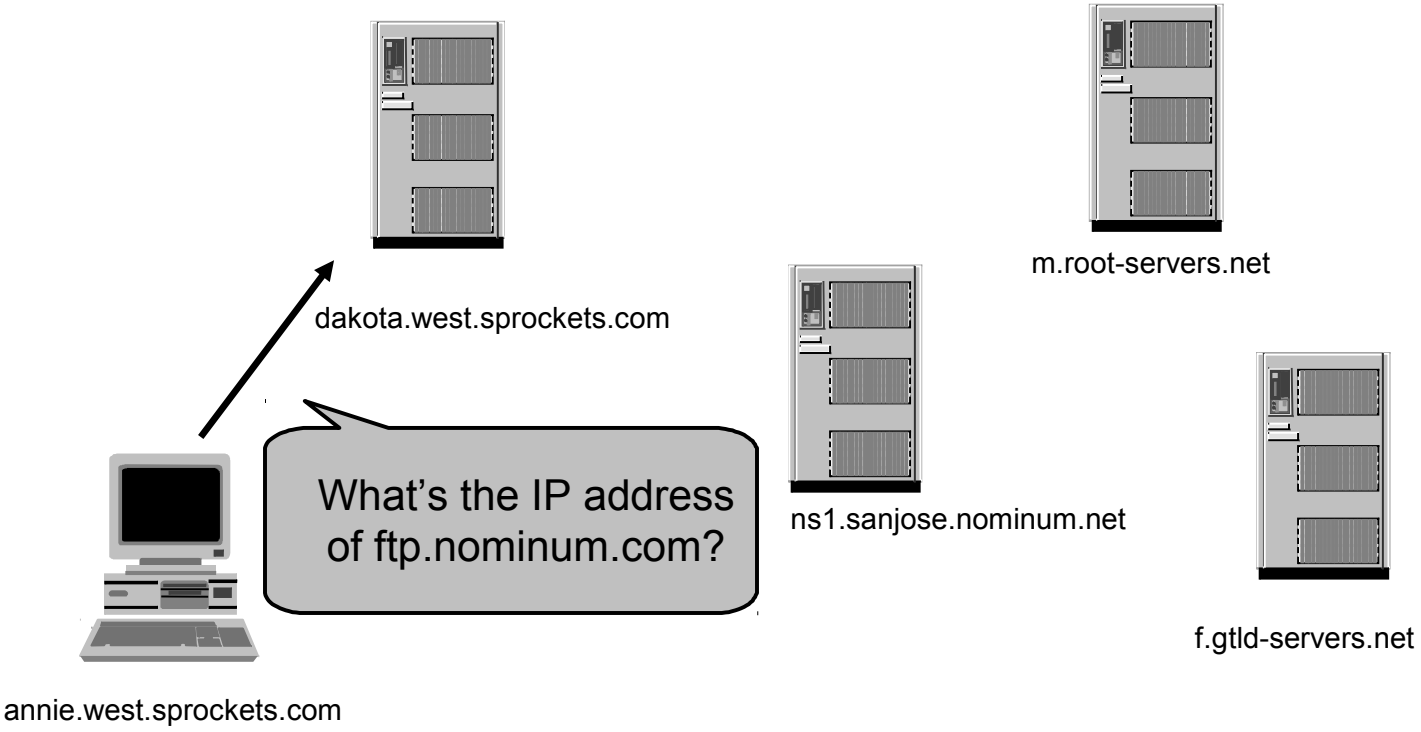
- After the previous query, the name server *dakota* now knows:
  - The names and IP addresses of the *com* name servers
  - The names and IP addresses of the *nominum.com* name servers
  - The IP address of *www.nominum.com*
- Let's look at the resolution process again

annie.west.sprockets.com

```
ping ftp.nominum.com.
```

Resolution Process (Caching)

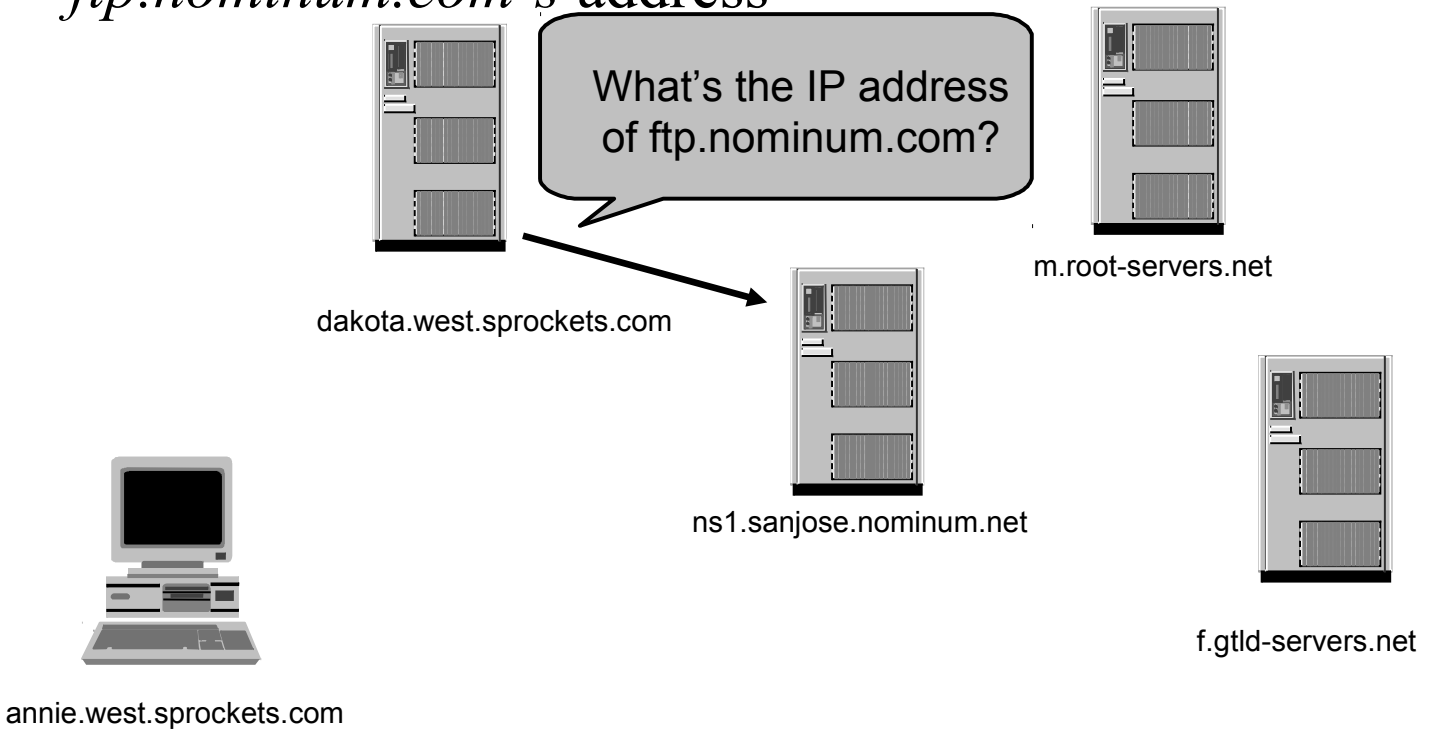
- The workstation *annie* asks its configured name server, *dakota,* for *ftp.nominum.com's* address




```
ping ftp.nominum.com.
```

Resolution Process (Caching)

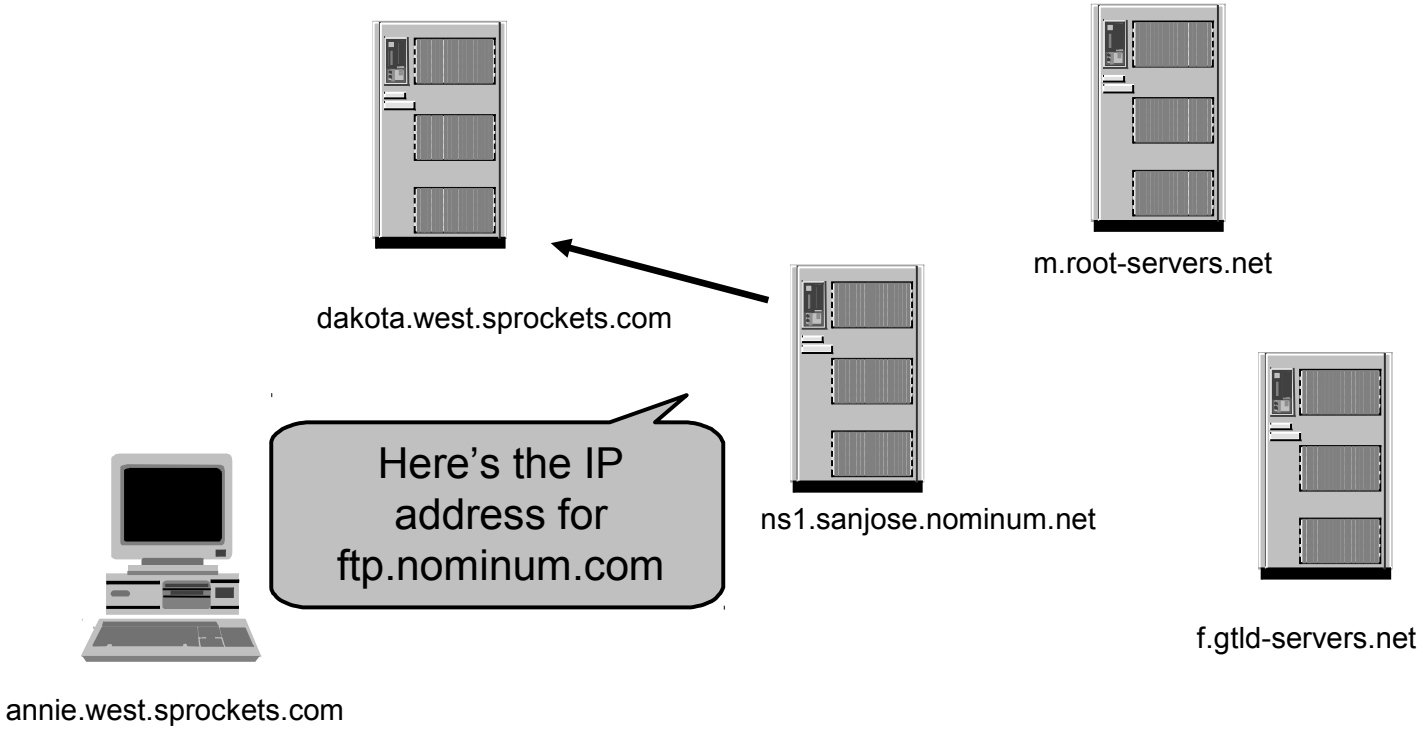
- *dakota* has cached a NS record indicating *ns1.sanjose* is an *nominum.com* name server, so it asks it for *ftp.nominum.com's* address




```
ping ftp.nominum.com.
```

Resolution Process (Caching)

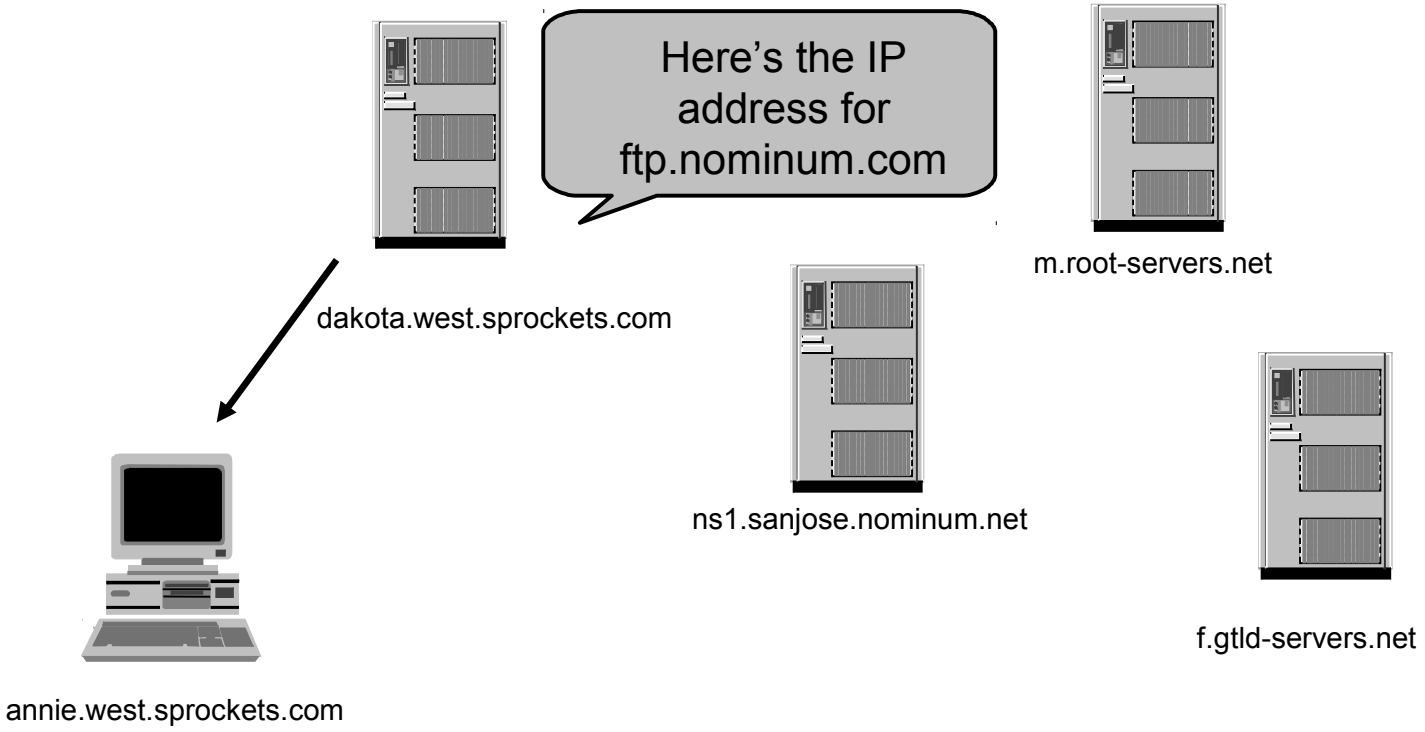
- The *nominum.com* name server *ns1.sanjose* responds with *ftp.nominum.com's* address




```
ping ftp.nominum.com.
```

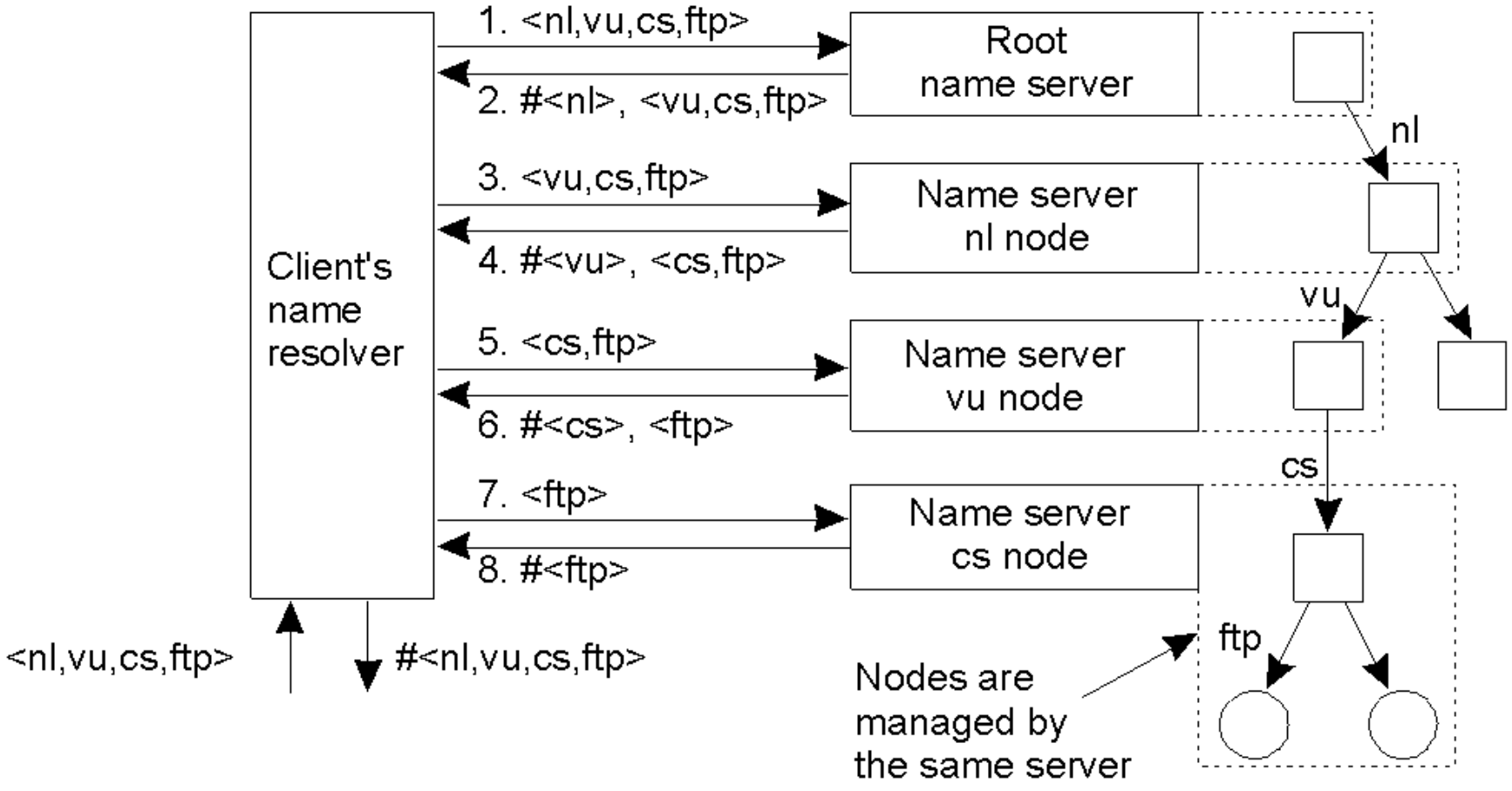
Resolution Process (Caching)

- The name server *dakota* responds to *annie* with *ftp.nominum.com's* address

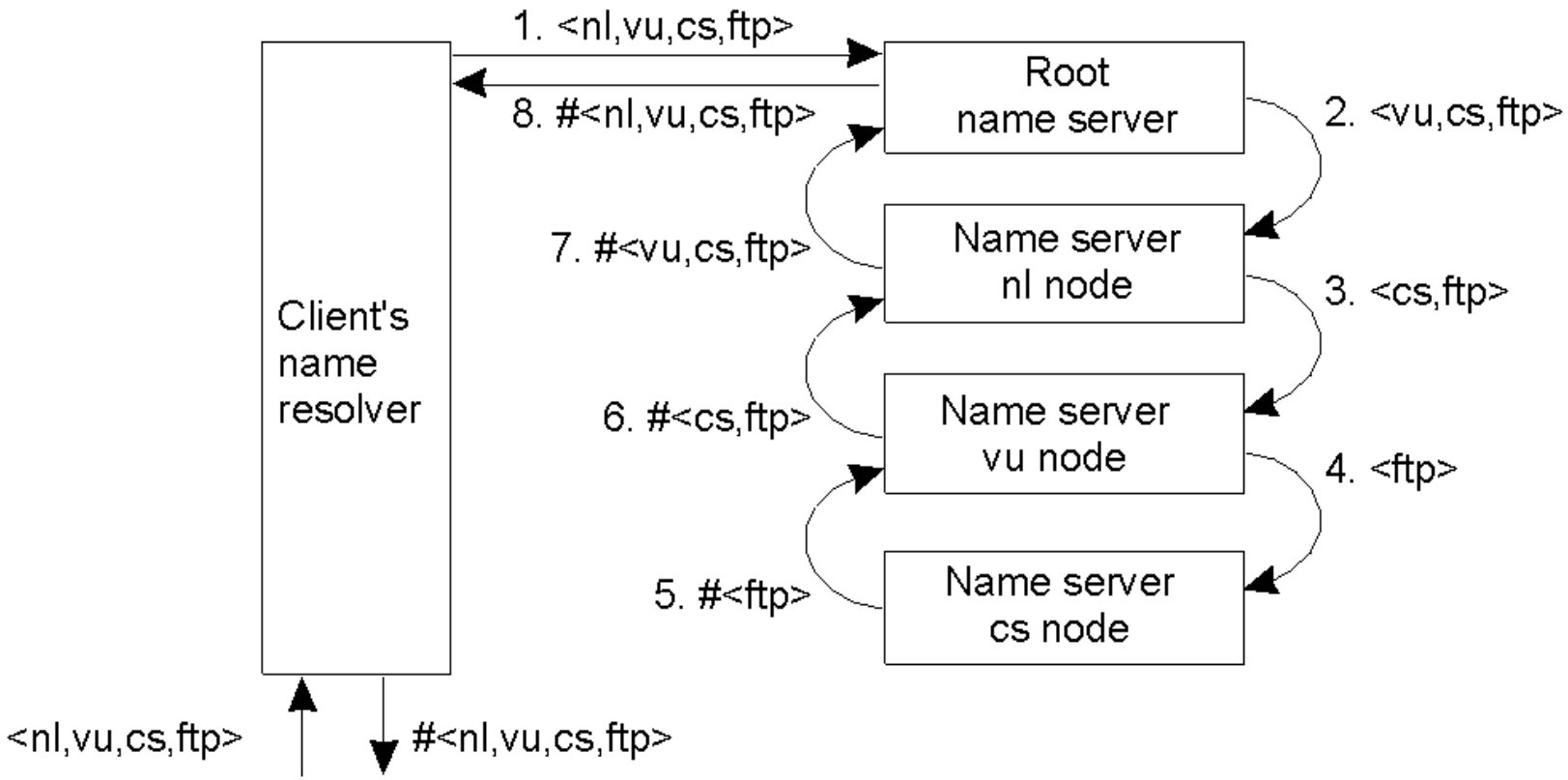



```
ping ftp.nominum.com.
```

Iterative Name Resolution



Recursive Name Resolution (1)

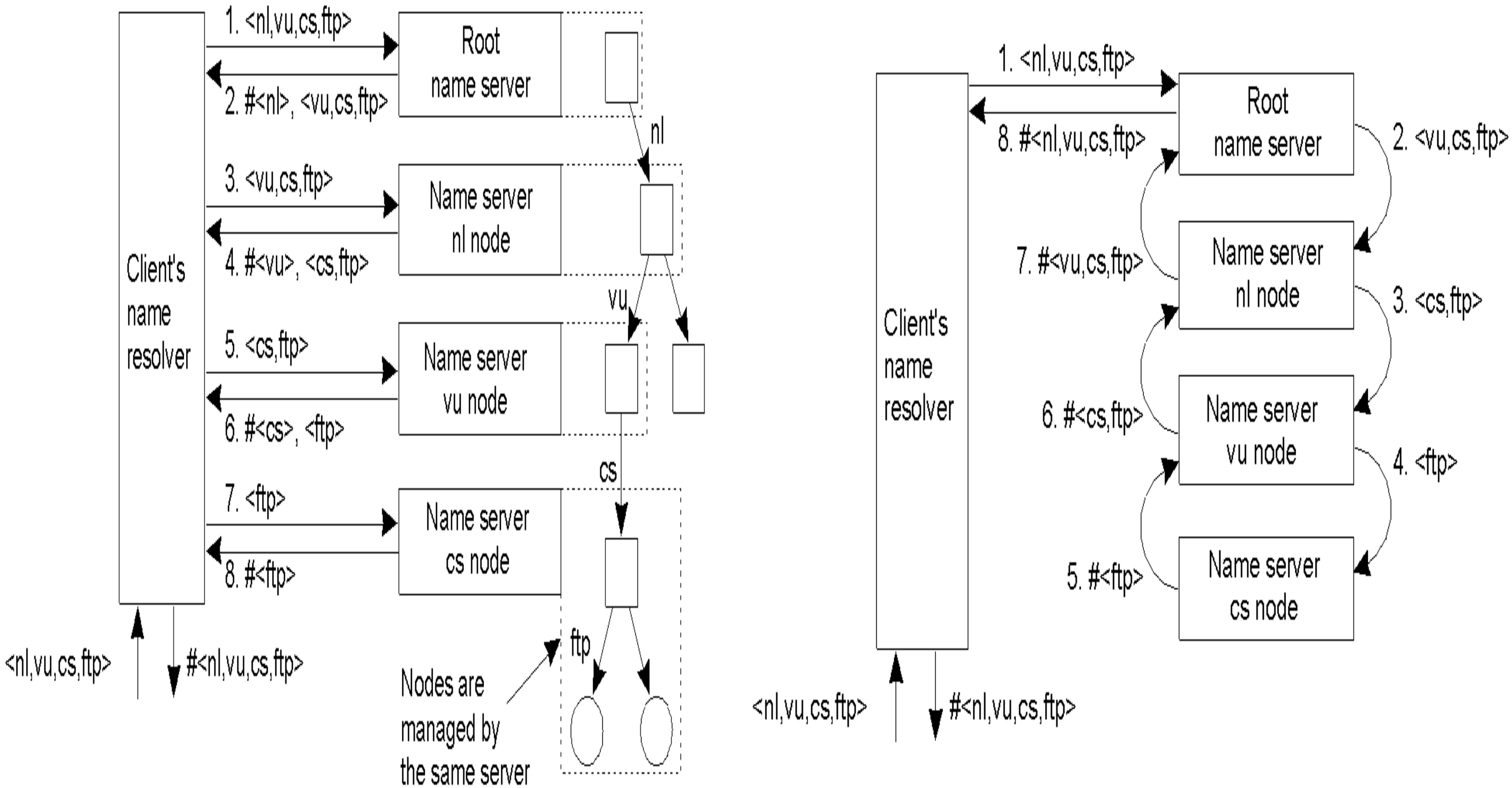


Recursive Name Resolution (2)

Server for node	Should resolve	Looks up	Passes to child	Receives and caches	Returns to requester
cs	<ftp>	#<ftp>	--	--	#<ftp>
vu	<cs,ftp>	#<cs>	<ftp>	#<ftp>	#<cs> #<cs, ftp>
ni	<vu,cs,ftp>	#<vu>	<cs,ftp>	#<cs> #<cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>
root	<ni,vu,cs,ftp>	#<nl>	<vu,cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>	#<nl> #<nl,vu> #<nl,vu,cs> #<nl,vu,cs,ftp>

- Recursive name resolution of <*nl, vu, cs, ftp*>. Name servers cache intermediate results for subsequent lookups.

Iterative versus Recursive Resolution (1)

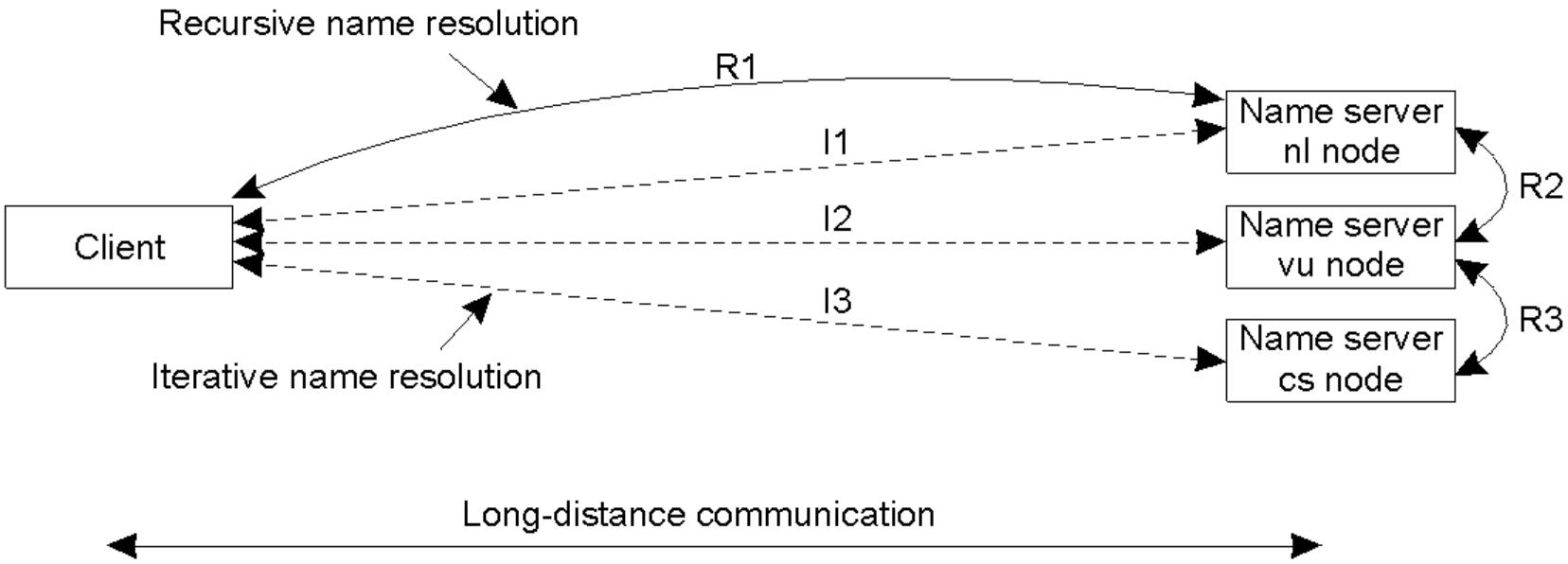


Performance-wise, which is better? / Which one costs more? / How about with caching?

Iterative versus Recursive Resolution (2)

- 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

Iterative versus Recursive Resolution (3)



- The comparison between recursive and iterative name resolution with respect to communication costs.

Overview

- Introduction to the DNS
- DNS Components
- DNS Structure and Hierarchy
- The DNS in Context

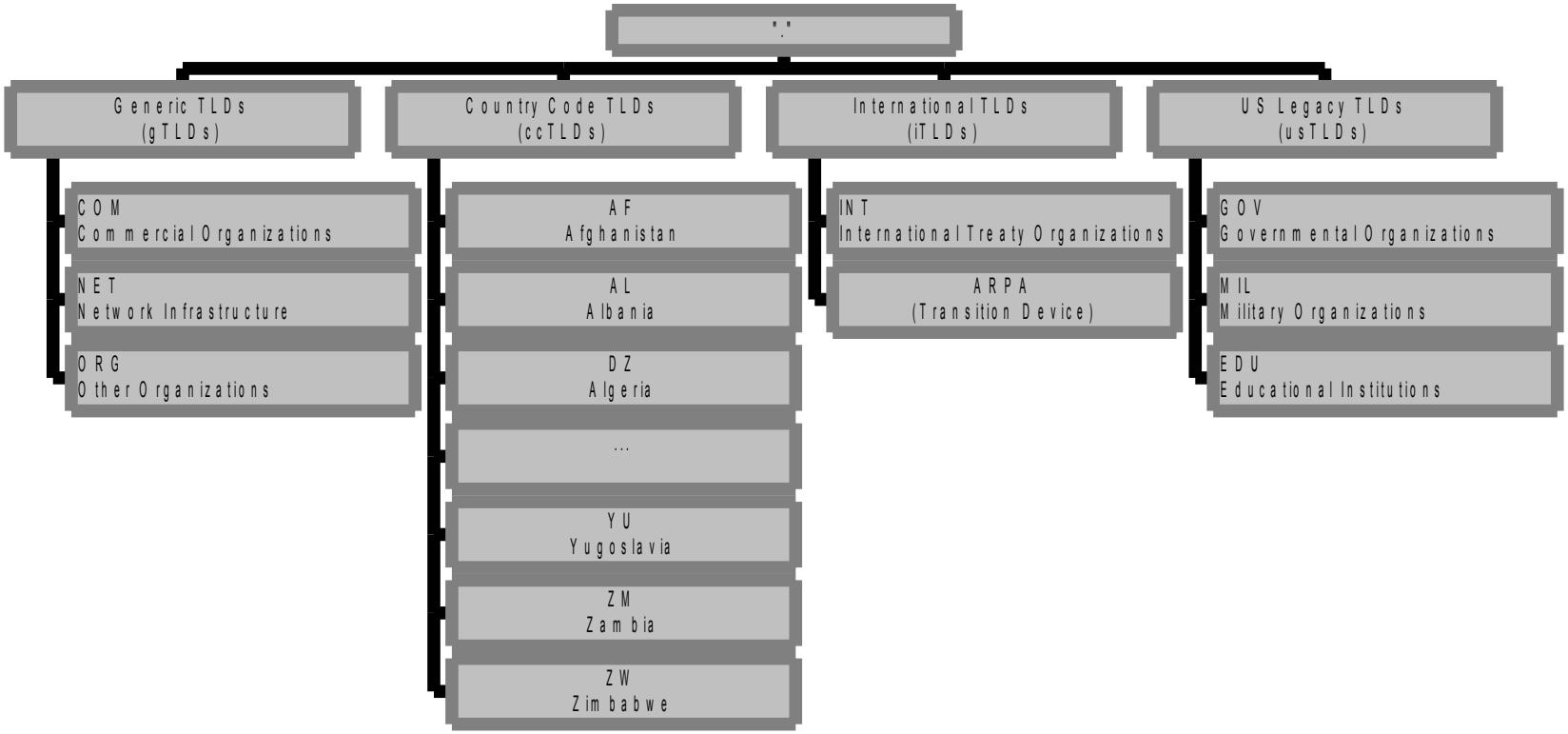
DNS Structure and Hierarchy

- The DNS imposes no constraints on how the DNS hierarchy is implemented except:
  - A single root
  - The label restrictions
  - So, can we create a host with a name *a.wonderful.world*?
- If a site is not connected to the Internet, it can use any domain hierarchy it chooses
  - Can make up whatever TLDs (top level domains) you want
- Connecting to the Internet implies use of the existing DNS hierarchy

Top-level Domain (TLD) Structure

- In 1983 (RFC 881), the idea was to have TLDs correspond to network service providers
  - e.g., ARPA, DDN, CSNET, etc.
    - Bad idea: if your network changes, your name changes
- By 1984 (RFC 920), functional domains was established
  - e.g., GOV for Government, COM for commercial, EDU for education, etc.
- RFC 920 also
  - Provided country domains
  - Provided “Multiorganizations”
    - Large, composed of other (particularly international) organizations
  - Provided a stable TLD structure until 1996 or so

The Current TLDs



Internet Corporation for Assigned Names and Numbers (ICANN)

- ICANN's role: to oversee the management of Internet resources including
  - Addresses
    - Delegating blocks of addresses to the regional registries
  - Protocol identifiers and parameters
    - Allocating port numbers, etc.
  - Names
    - Administration of the root zone file
    - Oversee the operation of the root name servers

The Root Nameservers

- The root zone file lists the names and IP addresses of the authoritative DNS servers for all top-level domains (TLDs)
- The root zone file is published on 13 servers, “A” through “M”, around the Internet
- Root name server operations currently provided by volunteer efforts by a very diverse set of organizations

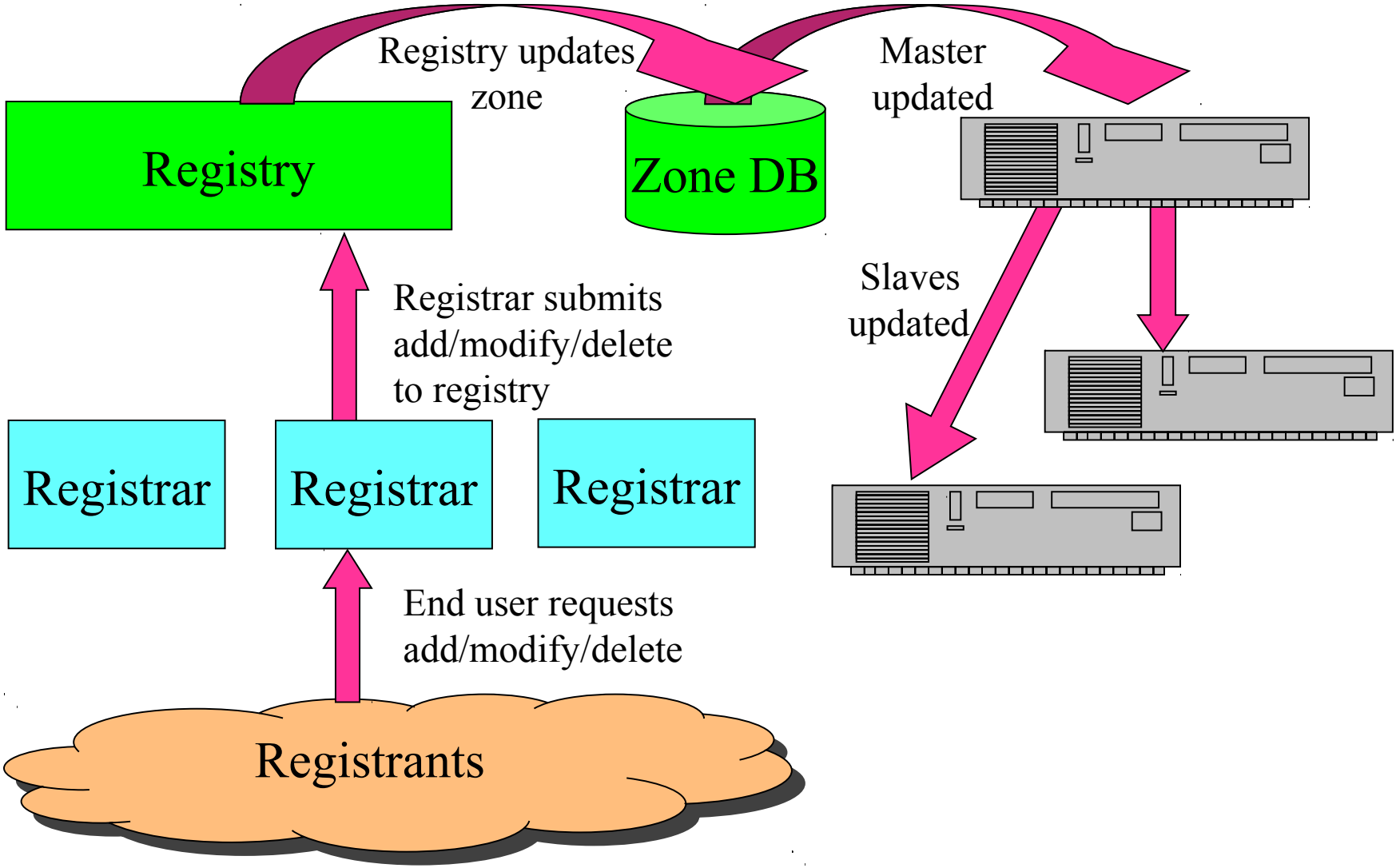
Root Name Server Operators

Nameserver	Operated by:
A	Verisign (US East Coast)
B	University of S. California –Information Sciences Institute (US West Coast)
C	Cogent Communications (US East Coast)
D	University of Maryland (US East Coast)
E	NASA (Ames) (US West Coast)
F	Internet Software Consortium (US West Coast)
G	U. S. Dept. of Defense (ARL) (US East Coast)
H	U. S. Dept. of Defense (DISA) (US East Coast)
I	Autonomica (SE)
J	Verisign (US East Coast)
K	RIPE-NCC (UK)
L	ICANN (US West Coast)
M	WIDE (JP)

Registries, Registrars, and Registrants

- A classification of roles in the operation of a domain name space
- Registry
  - the name space's database
  - the organization which has edit control of that database
  - the organization which runs the authoritative name servers for that name space
- Registrar
  - the agent which submits change requests to the registry on behalf of the registrant
- Registrant
  - the entity which makes use of the domain name

Registries, Registrars, and Registrants



Verisign: the registry and registrar for gTLDs

- .COM, .NET, and .ORG
  - By far the largest top level domains on the Internet today

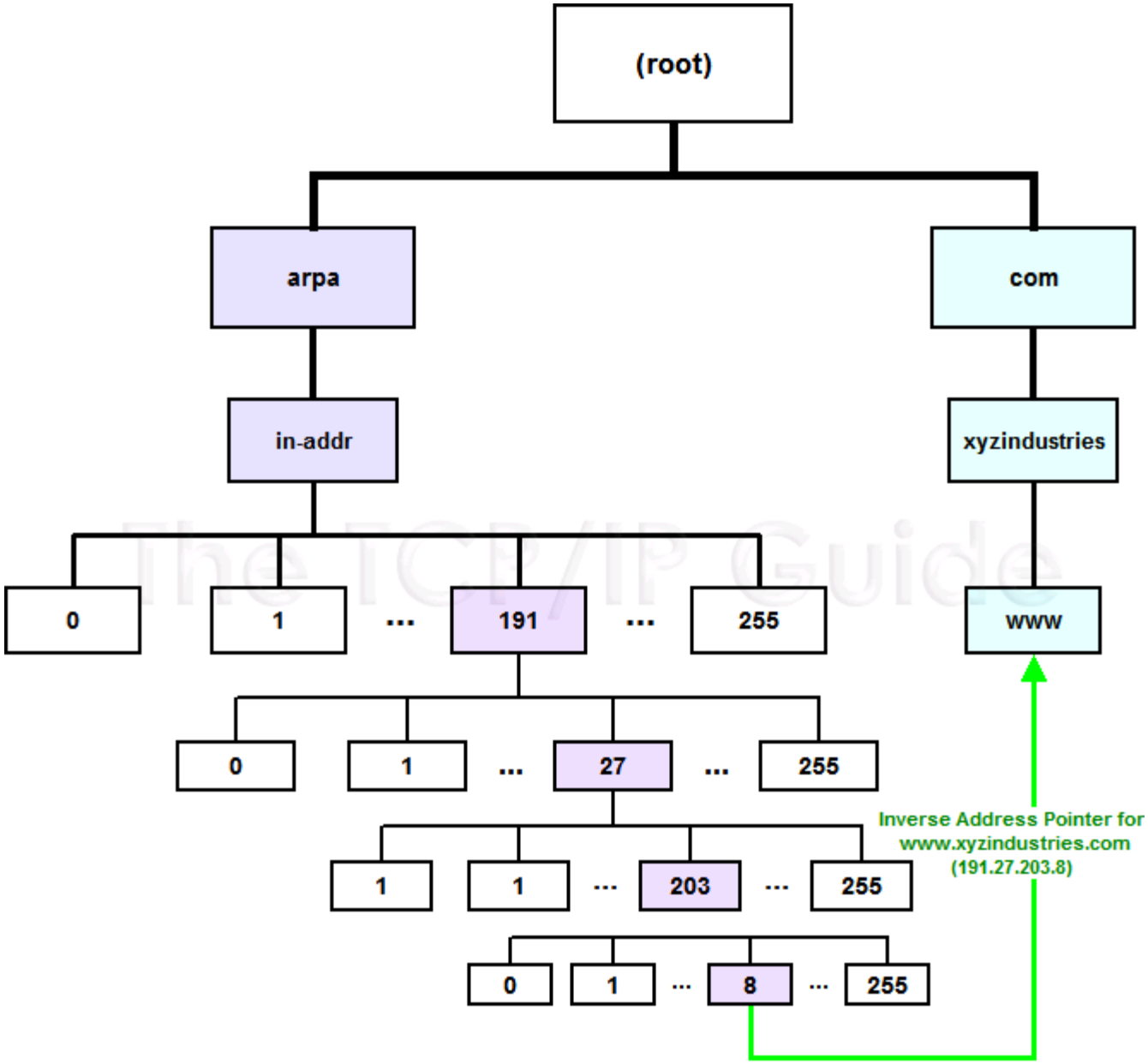
- Verisign received the contract for the registry for .COM, .NET, and .ORG
  - also a registrar for these TLDs

Overview

- Introduction to the DNS
- DNS Components
- DNS Hierarchy
- The DNS in Context

Load Concerns

- 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



Performance Concerns

- DNS is a very lightweight protocol
  - Simple query – response
- Any performance limitations are the result of network limitations
  - Speed of light
  - Network congestion
  - Switching/forwarding latencies

Security Concerns

- Base DNS protocol (RFC 1034, 1035) is insecure
  - DNS spoofing (cache poisoning) attacks are possible
- DNS Security Enhancements (DNSSEC, RFC 2565) remedies this flaw
  - But creates new ones
    - DoS attacks
    - Amplification attacks
- DNSSEC strongly discourages large flat zones
  - Hierarchy (delegation) is good

Questions?