Domain Name System

DNS *Domain Name System*

The *domain name system* is usually used to translate a host name into an IP address

Domain names comprise a hierarchy so that names are unique, yet easy to remember.

DNS introduction

- People: many identifiers:
  - SSN, name, passport #
- Internet hosts, routers:
  - IP address (32 bit) - used for addressing datagrams
- “name”, e.g., cs.pub.ro by humans

Map between IP addresses and name ?

- Domain Name System:
  - distributed database
    - implemented in hierarchy of many name servers
  - application-layer protocol
    - host, routers, name servers to communicate to resolve names (address/name translation)
- Note: core Internet function, implemented as application-layer protocol
- Complexity at network’s “edge”

In The Beginning…

- Early host database was a ***text file***
- Operational nightmare!
  - Rate of change in file was growing exponentially
    - NIC staff overwhelmed
  - Everyone copied it (nightly!) to get current version
    - Network overwhelmed
  - Many opportunities for errors
    - And we experienced many of them
  - Fate sharing model wrong
    - Affected parties couldn’t change database

DNS begining

…. there was the HOSTS.TXT file

- Before DNS (until 1985), the name-to-IP address was done by downloading a single file (hosts.txt) from a central server with FTP
  - Names in hosts.txt are not structured.
  - The hosts.txt file still works on most operating systems. It can be used to define local names.
    - `/etc/hosts` [Unix]
    - `c:\windows\system32\drivers\etc\hosts` [Windows]
- A centrally-maintained file, distributed to all hosts on the Internet

hosts.txt doesn't scale

- ✗ Huge file
- ✗ Needs frequent copying to ALL hosts
- ✗ Consistency
- ✗ Always out-of-date
- ✗ Name uniqueness
- ✗ Single point of administration

Naming

- How do we efficiently locate resources?
  - DNS: name -> IP address
- • Challenge
  - How do we scale these to the wide area?

Obvious Solutions

Why not centralize DNS?

  - Single point of failure
  - Traffic volume
  - Distant centralized database
  - Single point of update

Why not use /etc/hosts?

- Original Name to Address Mapping
  - Flat namespace
  - /etc/hosts
  - SRI kept main copy
  - Downloaded regularly
- Count of hosts was increasing: machine per domain -> machine per user
  - Many more downloads
  - Many more updates

Domain Name System Goals

- Basically a wide-area distributed database
- Scalability
- Decentralized maintenance
- Robustness
- Global scope
  - Names mean the same thing everywhere
- Don't need
  - Atomicity
  - Strong consistency

The domain name system was born (Paul Mockapetris, RFC1034/1035)

- DNS is a Distributed Database for holding name to IP address (and other) information
- Distributed:
  - Shares the administration
  - Shares the load
- Robustness and performance through:
  - Replication
  - Caching
- A *critical piece of Internet infrastructure*

- Conceptually, programmers can view the DNS
- database as a collection of millions of *host entry structures*:

```
/* DNS host entry structure */
struct hostent {
        char        *h_name;             /* official domain name of host */
        char        **h_aliases;         /* null-terminated array of domain names */
        int         h_addrtype;          /* host address type (AF_INET) */
        Int          h_length;           /* length of an address, in bytes */
        char        **h_addr_list;        /* null-terminated array of in_addr structs */
};
```

- **in_addr** is a struct consisting of 4-byte IP address

Functions for retrieving host entries from DNS:

- **gethostbyname:** query key is a DNS host name.
- **gethostbyaddr:** query key is an IP address.

Host name structure

- Each host name is made up of a sequence of *labels* separated by periods.
  - Each label can be up to 63 characters
  - The total name can be at most 255 characters.
- Alphanumeric characters a-z, A-Z, 0-9 and -
- Examples:
  - pub.ro
  - cs.pub.ro

The most important element of address is on the right (TLD – top level domain)

Domain Name Space

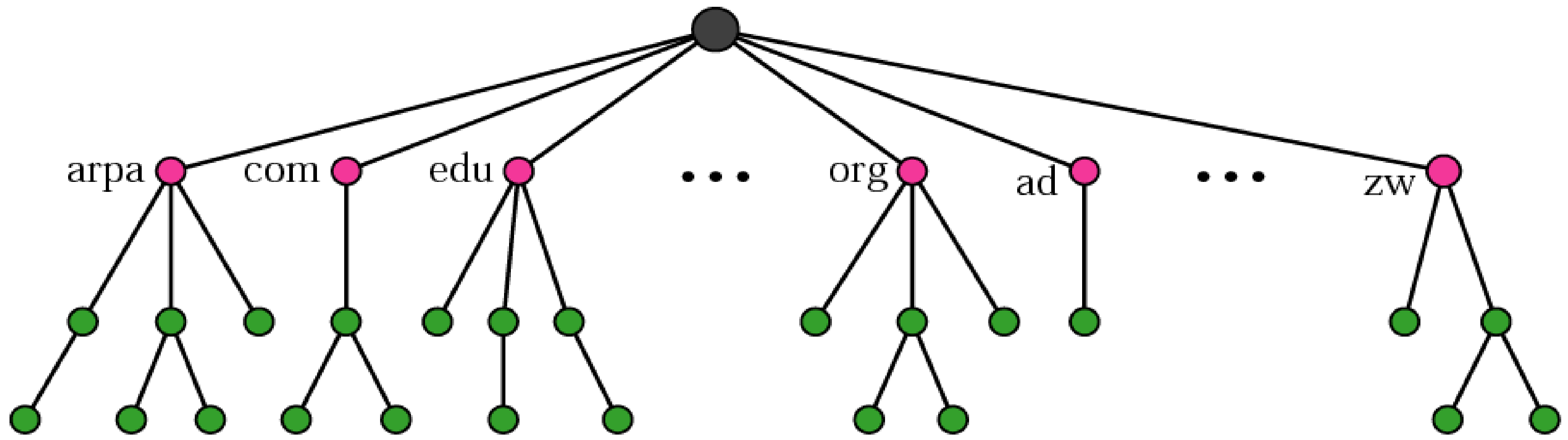
Label

Each label can be up to 63 characters

Domain Name

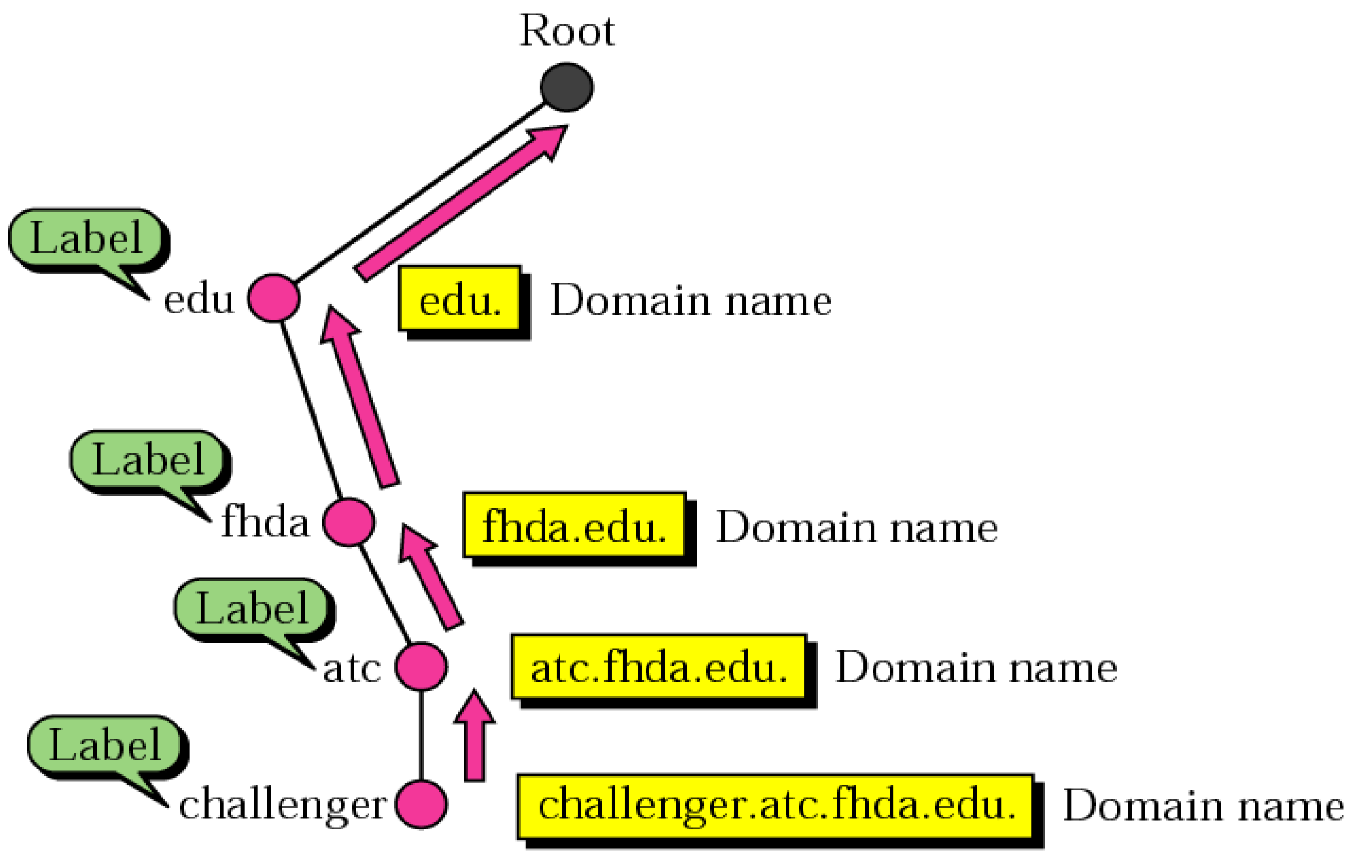
Sequence of the labels separated by dots

Domain name space



Children of the node have different labels

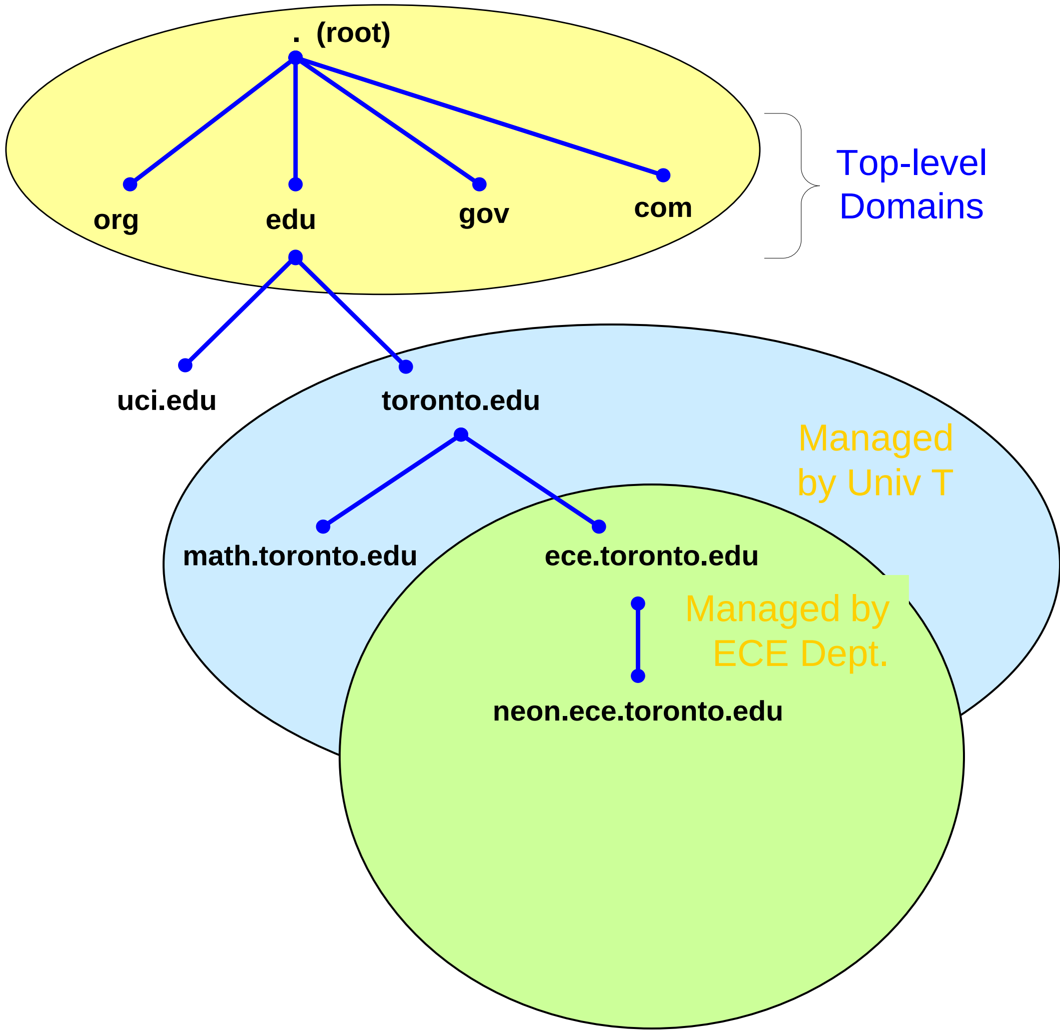
Domain names and labels



DNS Name hierarchy

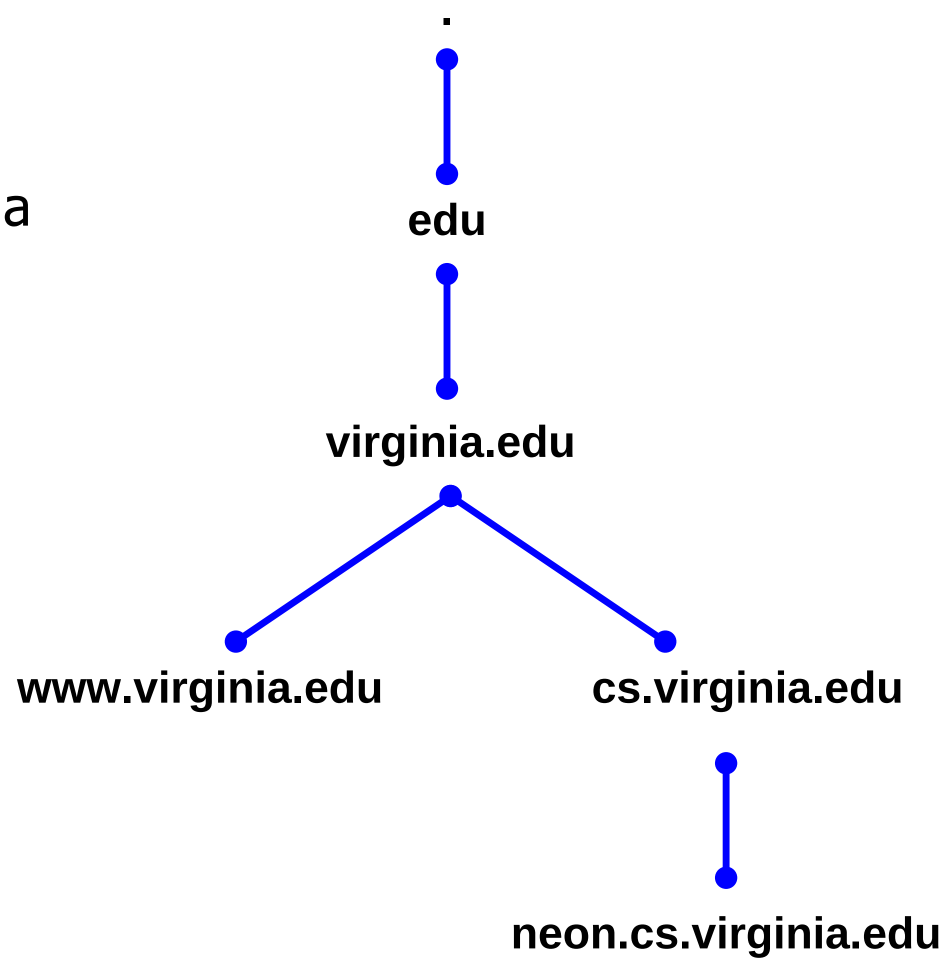
- DNS hierarchy can be represented by a tree
- Root and top-level domains are administered by an Internet central name registration authority (ICANN)

- Below top-level domain, administration of name space is delegated to organizations
- Each organization can delegate further



Domain name system

- Each node in the DNS tree represents a DNS name
- Each branch below a node is a DNS domain.
  - DNS domain can contain hosts or other domains (subdomains)
- Example:
  DNS domains are
  ., edu, virginia.edu, cs.virginia.edu



Domain names

- Hosts and DNS domains are named based on their position in the domain tree
- Every node in the DNS domain tree can be identified by a unique
  Fully Qualified Domain Name (FQDN).
  The FQDN gives the position in the DNS tree.

cs.virginia.edu or cs.virginia.edu.

- A FQDN consists of labels (“cs”,“virginia”,”edu”) separated by a period (“.”)
- There can be a period (“.”) at the end.
- Each label can be up to 63 characters long
- FQDN contains characters, numerals, and dash character (“-”)
- FQDNs are not case-sensitive

DNS Messages

Header

Question Section

Answer Section

Authoritative Section

Additional Information Section

FQDN and PQDN

FQDN	PQDN
challenger.atc.fhda.edu.	challenger.atc.fhda.edu
cs.hmme.com.	cs.hmme
www.funny.int.	www

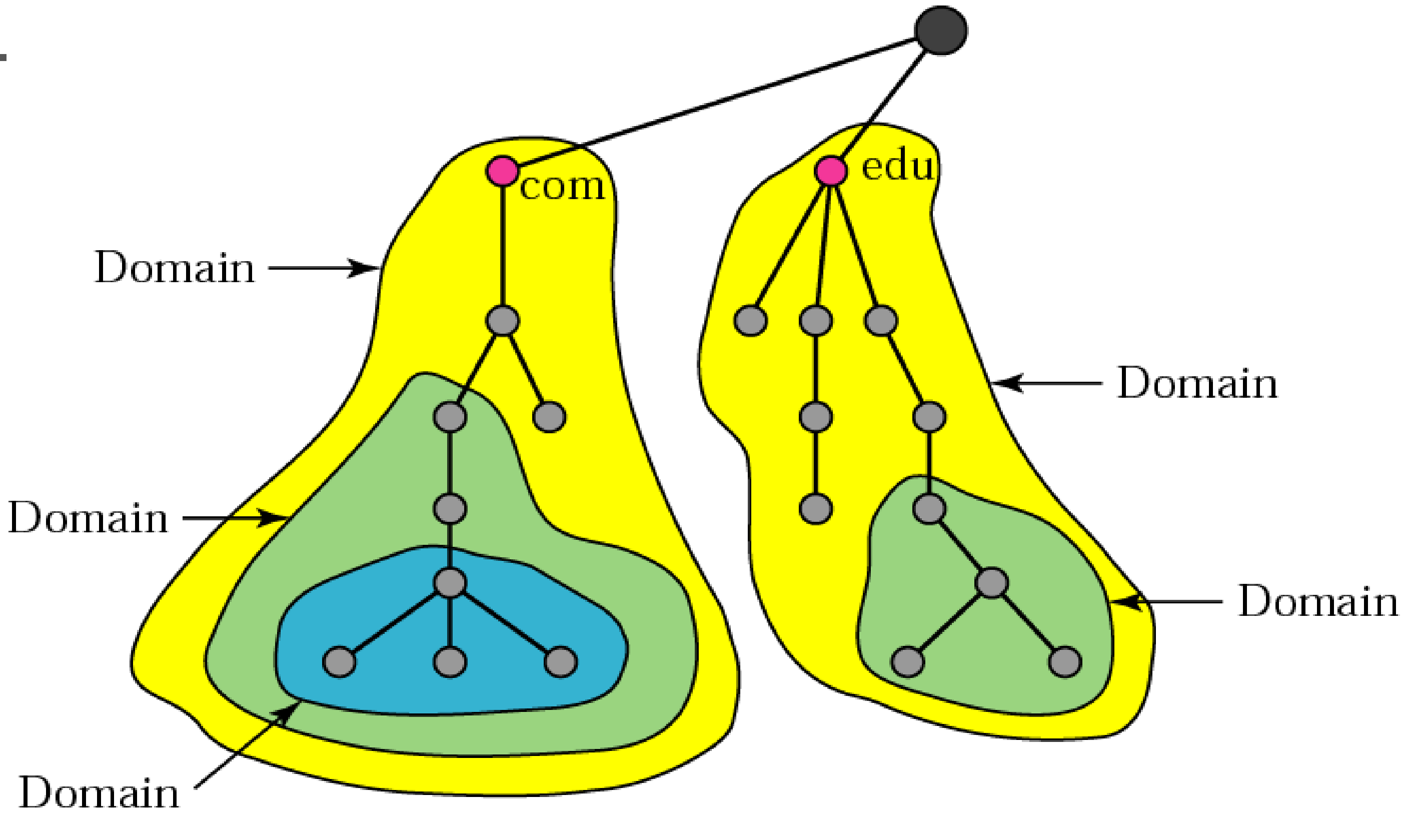
FQDN – Full qualified domain name (the name must end with a null label – the label ends with dot (.)

PQDN – Partially qualified domain name

FULLY QUALIFIED DOMAIN NAME (FQDN)
If a label is terminated by a null string, it is called a fully qualified domain name. An FQDN is a domain name that contains the full name of a host.

PARTIALLY QUALIFIED DOMAIN NAME (PDQN)
If a label is not terminated by a null string, it is called partially qualified domain name. A PQDN starts from a node, but it does not reach the root.

Domains



The domain name for a host is the sequence of labels that lead from the host (leaf node in the naming tree) to the top of the worldwide naming tree.

Contains resource records and sub-domains

Some resource records point to authoritative server for sub-domains / zones

A domain is a subtree of the worldwide naming tree.

Distribution of Name Spaces

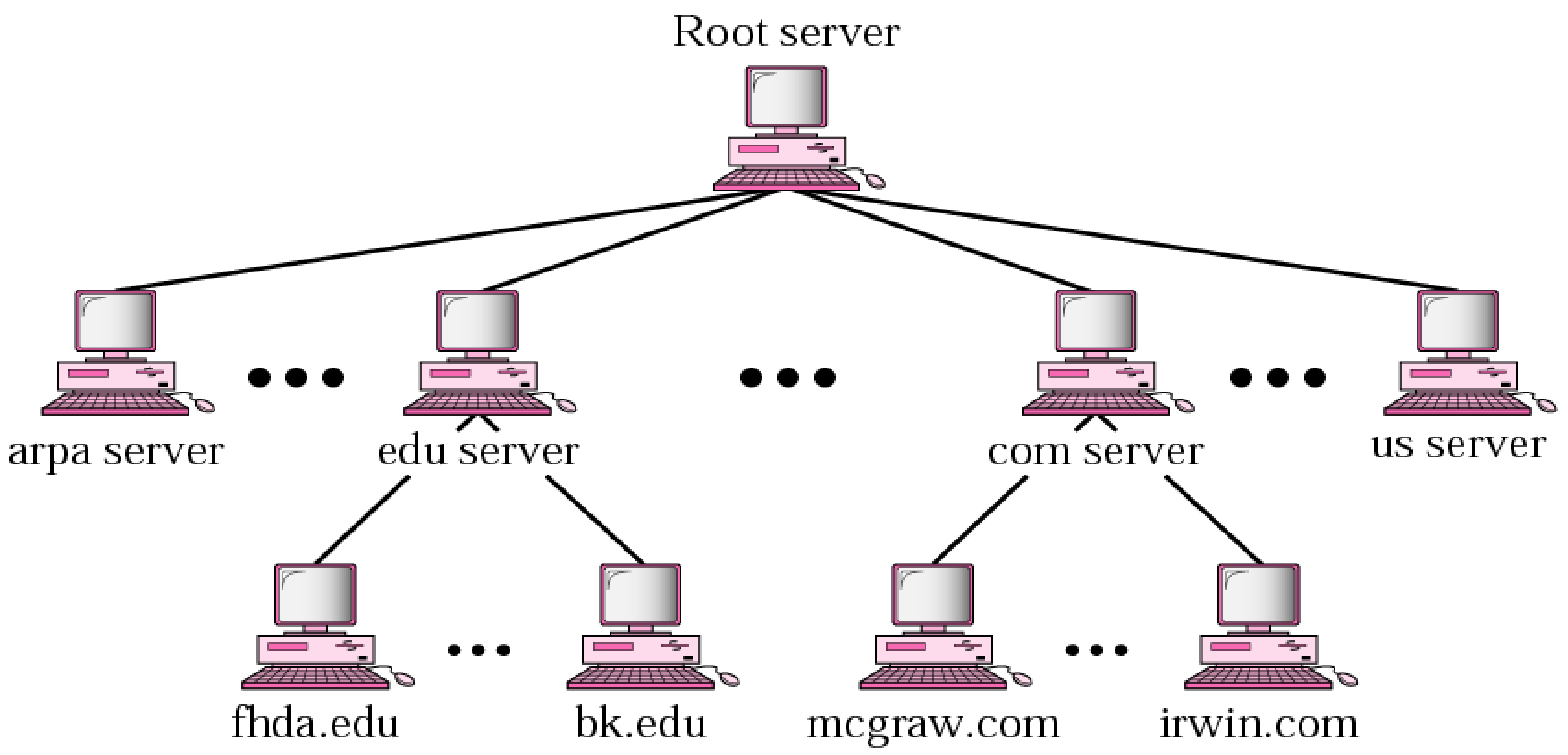
Hierarchy of Name Servers

Zone

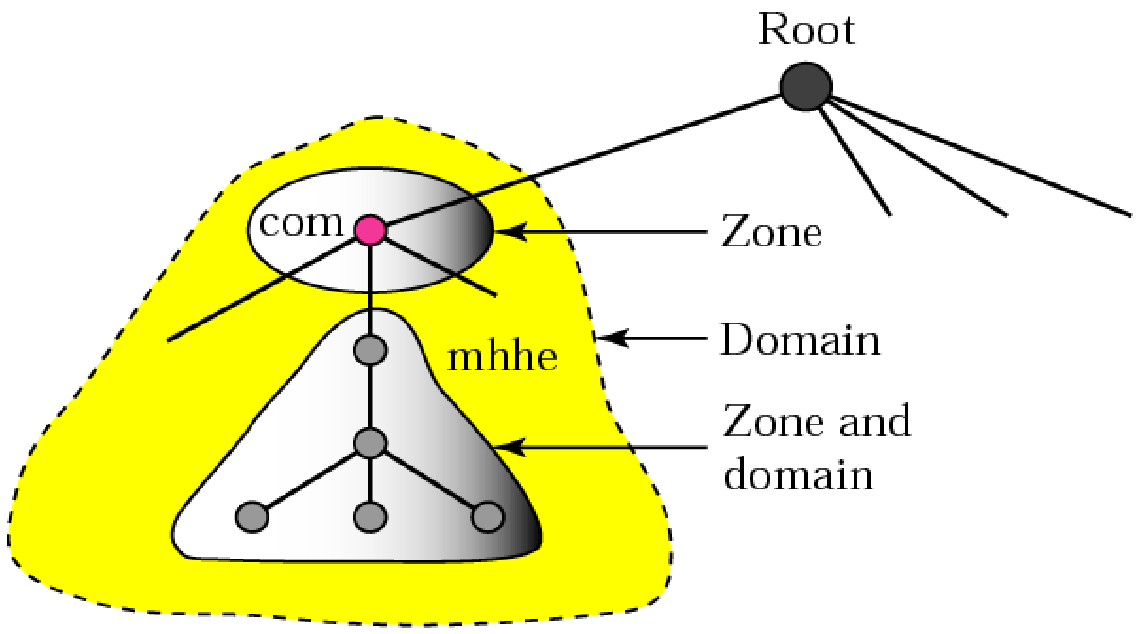
Root Server

Primary and Secondary Servers

Hierarchy of name servers



Zones and domains



Its records are held in a database ("zonefile") and served from an authoritative name server

Zone refers to all the resource records in a domain but not its sub domains

A primary server loads all information from the disk file;

The secondary server loads all information from the primary server.

DNS In The Internet

Generic Domain

Country Domain

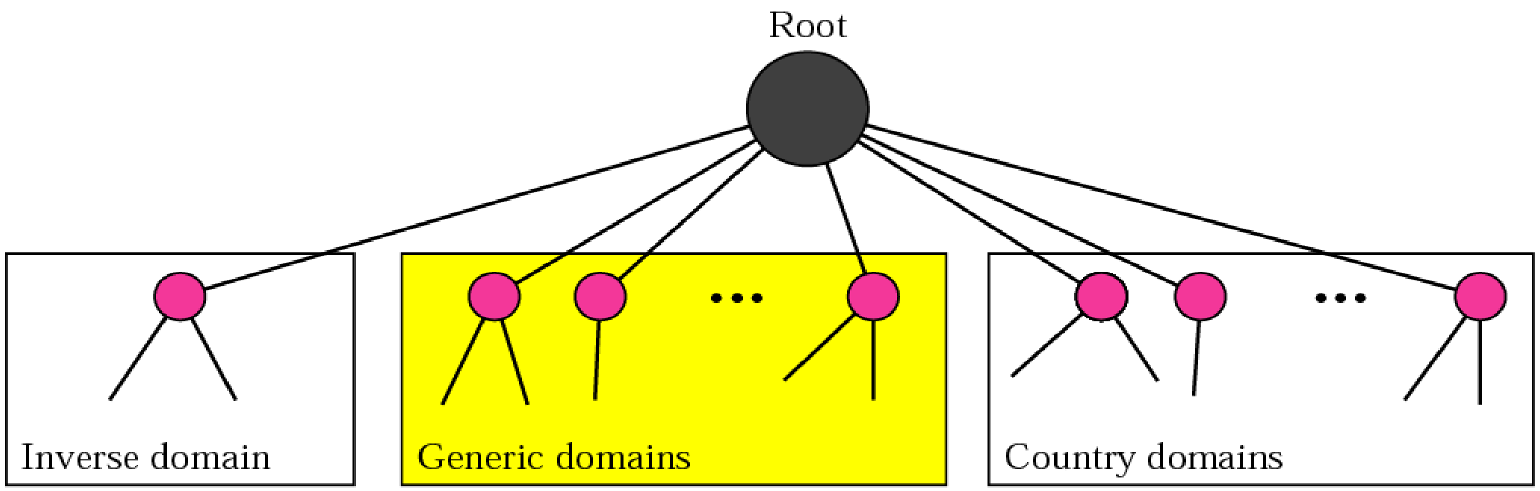
Inverse Domain

Top-level domains

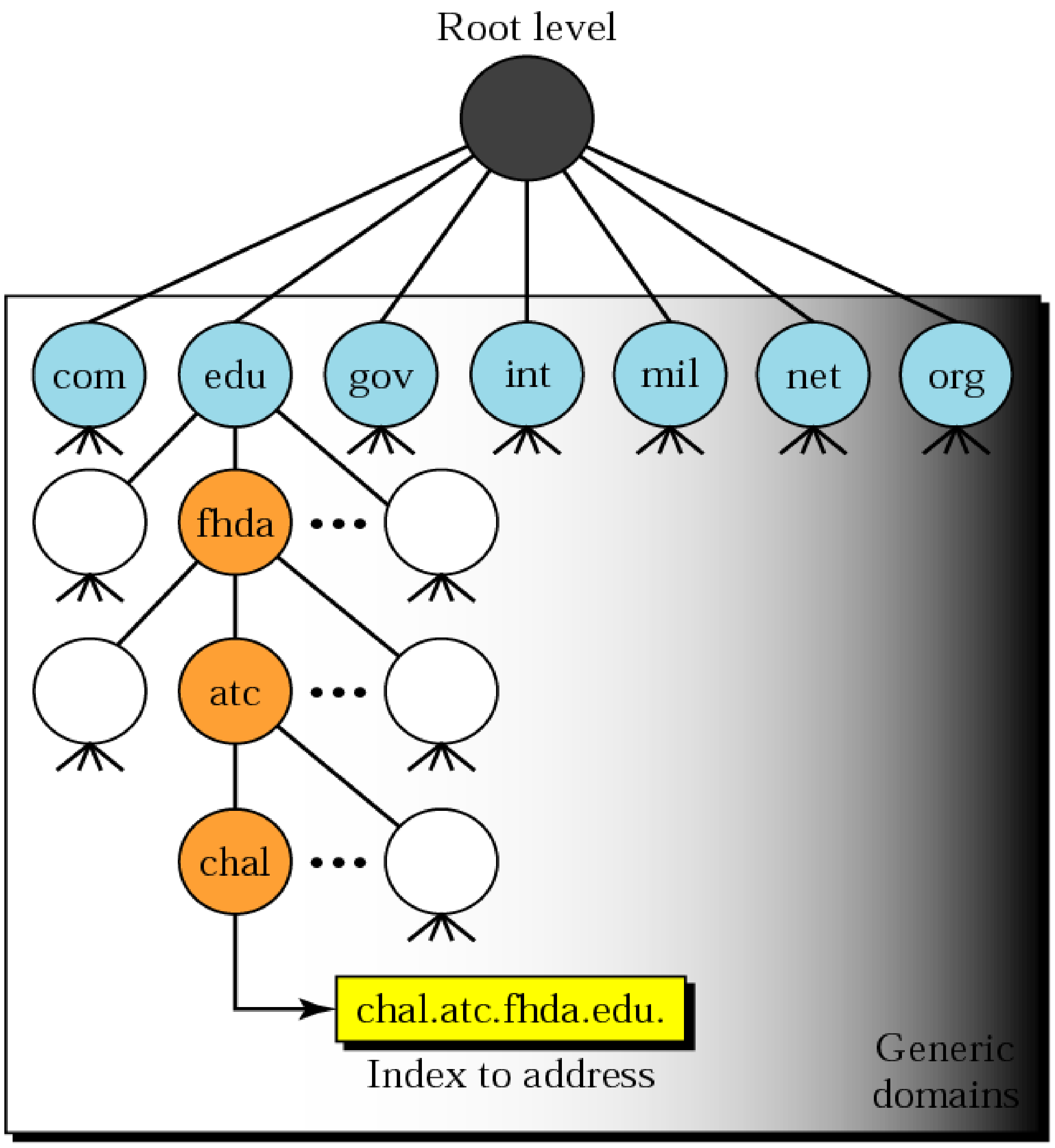
- Three types of top-level domains:
  - Organizational: 3-character code indicates the function of the organization
    - Used primarily within the US
    - Examples: gov, mil, edu, org, com, net
  - Geographical: 2-character country or region code
    - Examples: ro, us, va, jp, de
  - Reverse domains: A special domain (in-addr.arpa) used for IP address-to-name mapping

There are more than 200 top-level domains.

DNS in the Internet



Generic domains



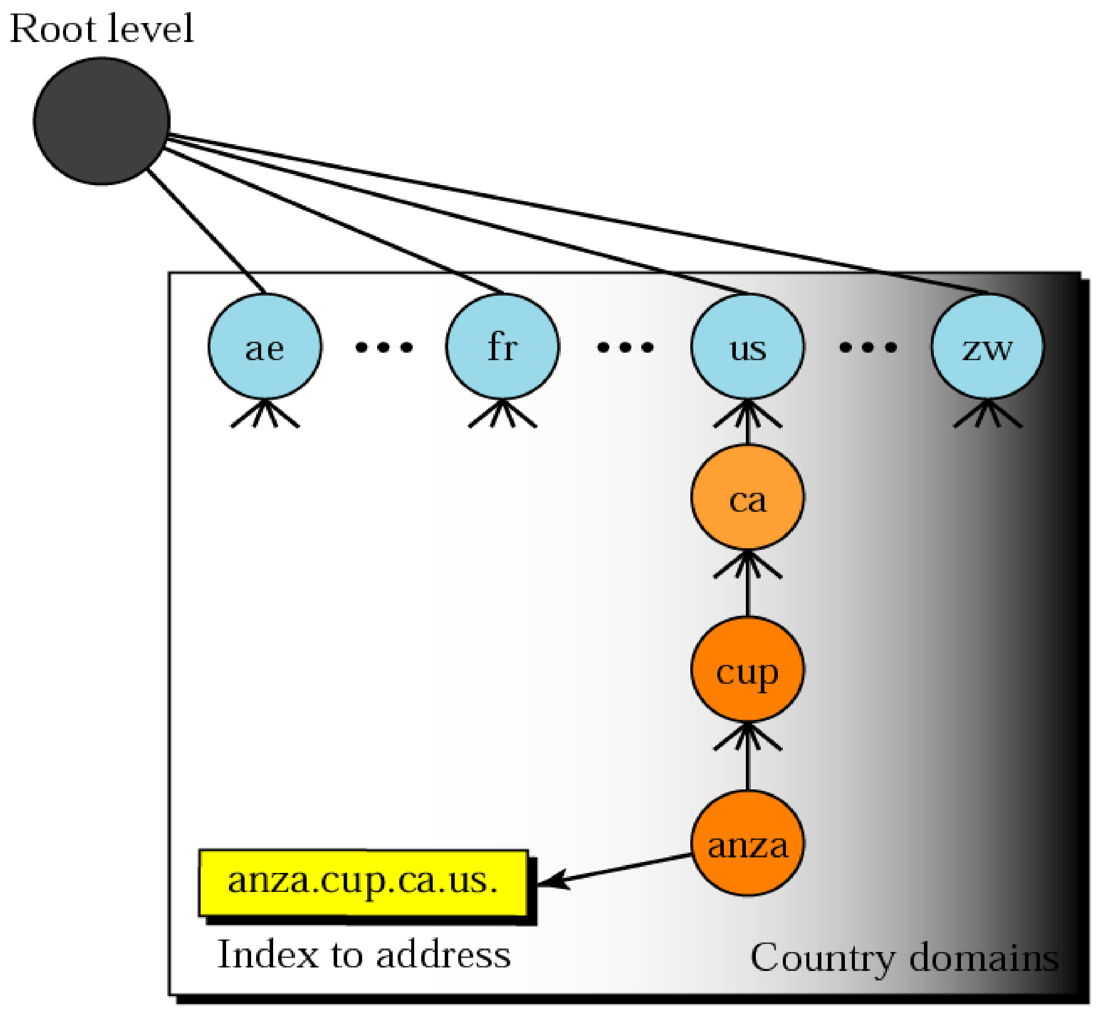
Generic domain labels

Label	Description
com	Commercial organizations
edu	Educational institutions
gov	Government institutions
int	International organizations
mil	Military groups
net	Network support centers
org	Nonprofit organizations

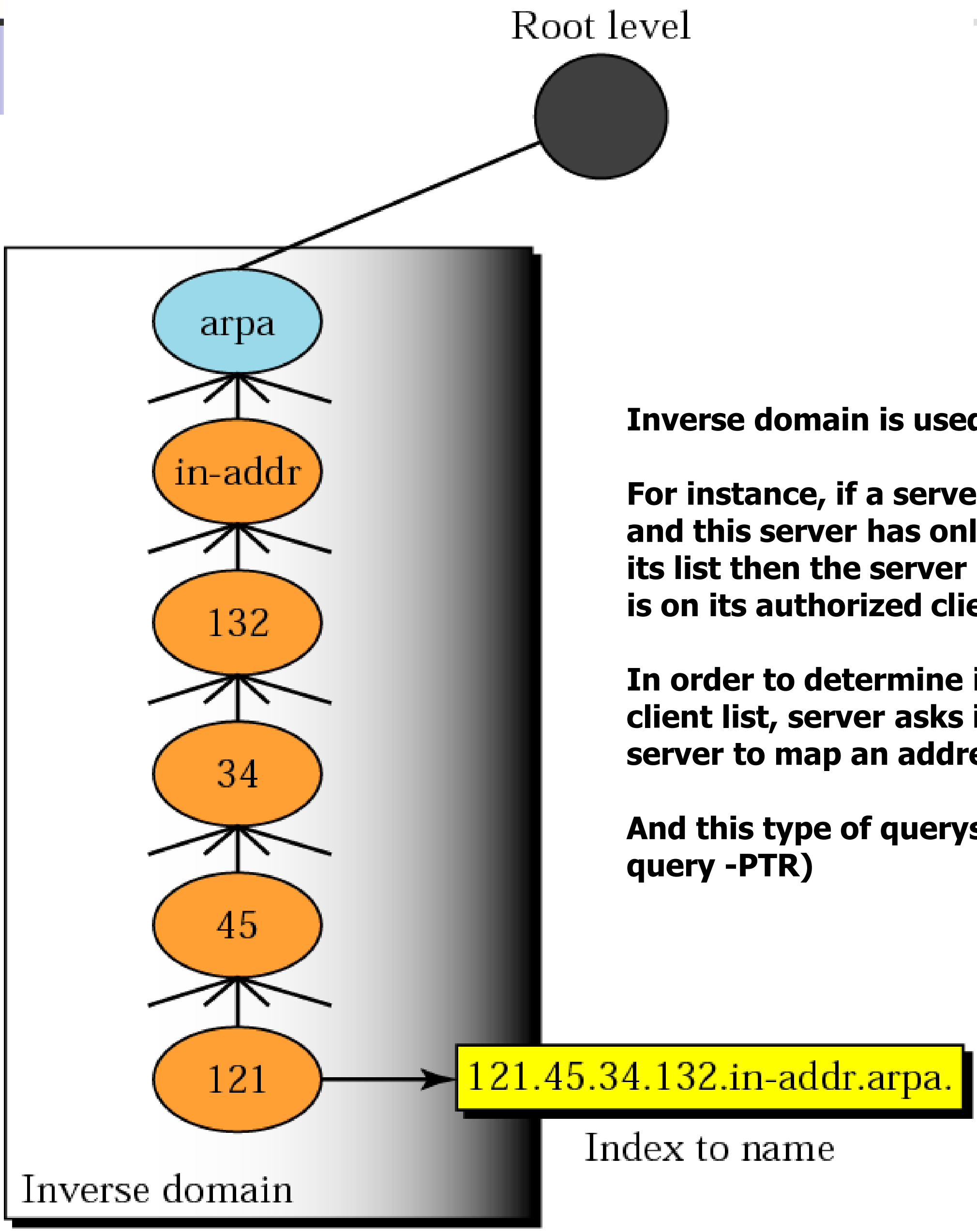
New generic domain labels

Label	Description
aero	Airlines and aerospace companies
biz	Businesses or firms (similar to com)
coop	Cooperative business organizations
info	Information service providers
museum	Museums and other nonprofit organizations
name	Personal names (individuals)
pro	Professional individual organizations

Country domains



Inverse domain



Inverse domain is used to map an address to a name.

For instance, if a server receives a request from a client and this server has only the ip addresses of the clients in its list then the server needs to find out if this client is on its authorized client list.

In order to determine if the client is on the authorized client list, server asks its resolver to query to the DNS server to map an address to name.

And this type of querys are called inverse query(pointer query -PTR)

Using the DNS

- A Domain Name (like www.tiscali.co.uk) is the *KEY* to look up information
- The result is one or more *RESOURCE RECORDS* (RRs)
- There are different RRs for different types of information
- You can ask for the specific type you want, or ask for "any" RRs associated with the domain name

What is a resource record?

- *A domain contains resource records*
- *Resource records are analogous to files*
- *Classified into types*
- *Some of the important types are SOA, NS, A, CNAME and MX*
- *Normally defines in “zone files”*
- *https://docs.nexcess.net/article/how-to-find-the-ip-address-of-a-website-or-server.html*

Commonly seen RRs
(Resource Record)

- A (address): map hostname to IP address
- PTR (pointer): map IP address to name
- MX (mail exchanger): where to deliver mail for user@*domain*
- CNAME (canonical name): map alternative hostname to real hostname
- TXT (text): any descriptive text
- NS (name server),
- SOA (start of authority): used for delegation and management of the DNS itself

The “A” Record

- **The “Address” record**
- **One or more normally defines a host**
- **Contains an IPv4 Address (*the address computers use to uniquely identify each other on the internet*)**

Eg. The record:

```
    www                    A          203.18.56.31
```

In the ***ausregistry.com.au*** *domain, defines the host uniquely identifiable as*

“www.ausregistry.com.au”

to be reachable at the IPv4 Address

203.18.56.31

The “CNAME” Record

- **A CNAME defines an alias**
- *The alias will then be resolved, if another CNAME is encountered then the process continues until an A record is found*

Eg. The record:

```
search              CNAME       www.google.com.
```

*In the **ausregistry.com.au** domain**,** defines the name uniquely identifiable as*

“search.ausregistry.com.au”

to be and alias to

“www.google.com”

The "MX" Record

- **An MX record defines the mail servers for a particular domain**
- *Mail eXchange records hold the name of hosts, and their priorities, able to deliver mail for the domain.*

Eg. The record:

```
ausregistry.com.au          MX      10       mail
```

*In the **ausregistry.com.au** domain**,***

*defines the host **mail** to be the priority **10 mail server***

*for the **"ausregistry.com.au"** domain*

The “NS” Record

- **An NS record defines the authoritative Name servers for the domain.**
- *The “Name Server” records also define the name servers of children domains*

Eg. The record:

```
internal        NS      ns1.hosting.com.au.
```

*In the **ausregistry.com.au** domain,*

*defines the host **“ns1.hosting.com.au”***

*to be a **name sever** for the **“internal.ausregistry.com.au”** sub-domain*

What is a Delegation?

- ***Delegation refers to the act of putting***

  NS (name server)

  records in a domain name “delegating” control of a sub-domain to another entity

- ***This entity then has the ability to control the resource records in this sub-domain and delegate further children domains to other entities*.**

Eg. IANA (**Internet Assigned Numbers Authority)** delegating control of a country code domain to the country.

DNS is a Client-Server application

- Requests and responses are normally sent in UDP packets, port 53
- Occasionally uses TCP, port 53
  - for very large requests, e.g. zone transfer from master to slave
- TCP/53 must **NOT be filtered.**

Resolution

Resolver

Mapping Names to Addresses

Mapping Addresses to Names

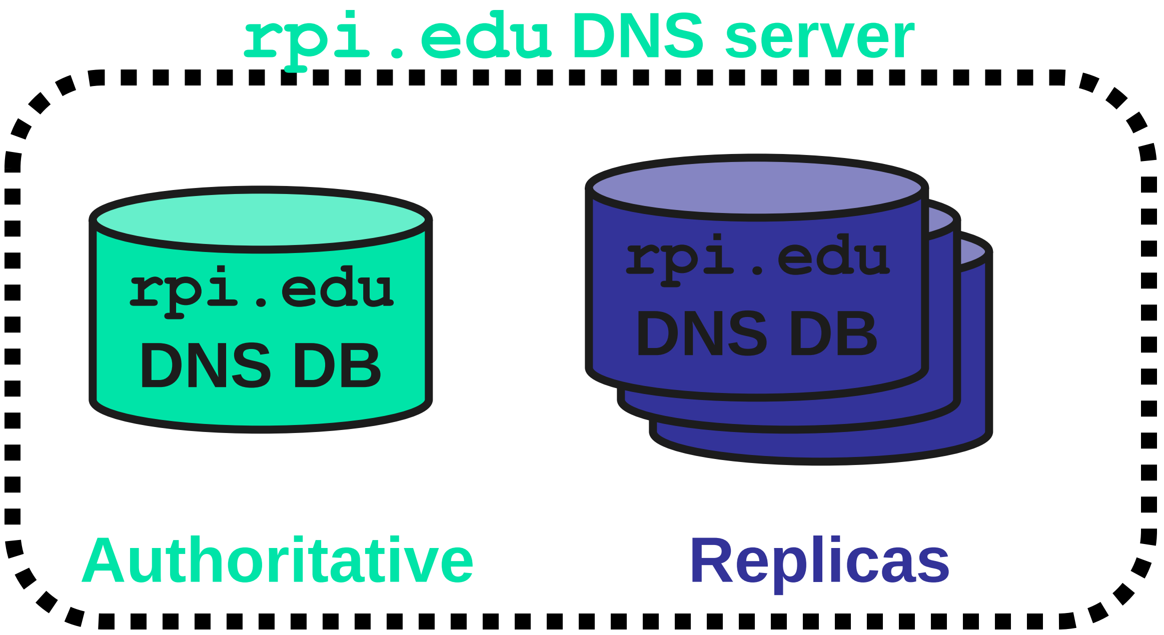
Recursive Resolution

Iterative Resolution

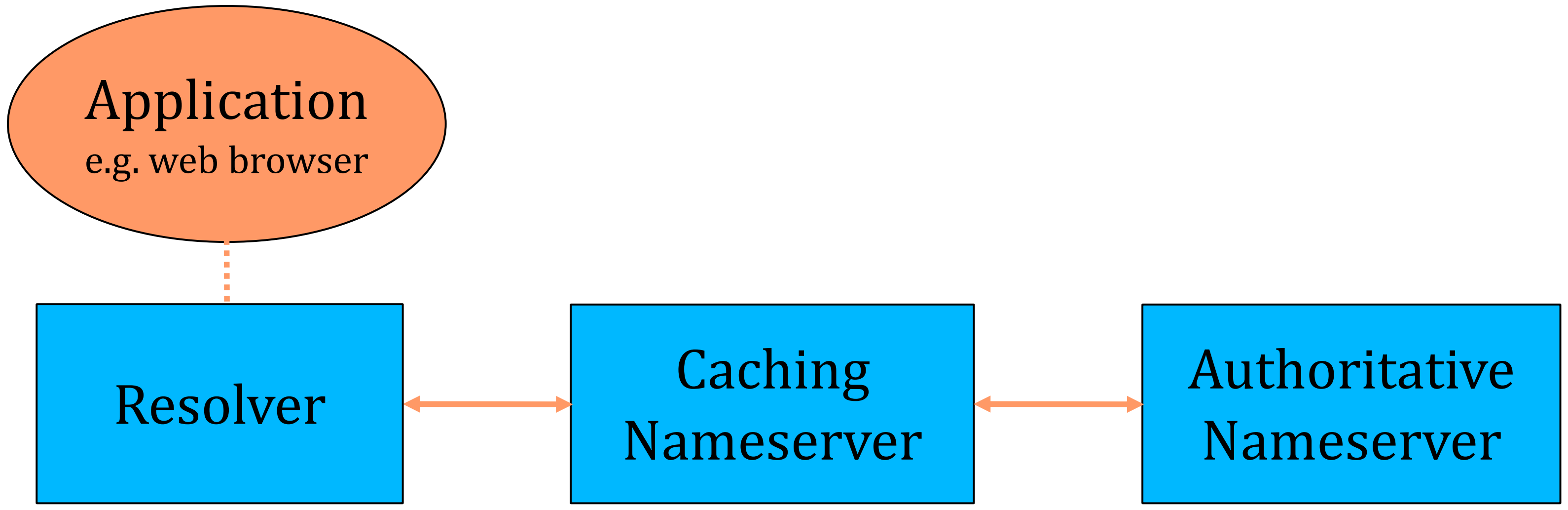
Caching

DNS Distributed Database

- There is one primary server for a domain, and typically a number of secondary servers containing replicated databases.



There are three roles involved in DNS



- *RESOLVER*
  - Takes request from application, formats it into UDP packet, sends to cache
- *CACHING NAMESERVER*
  - Returns the answer if already known
  - Otherwise searches for an authoritative server which has the information
  - Caches the result for future queries
  - Also known as RECURSIVE nameserver
- *AUTHORITATIVE NAMESERVER*
  - Contains the actual information put into the DNS by the domain owner

DNS Clients

- A DNS client is called a *resolver*.
- A call to **`gethostbyname()`** is handled by a resolver (typically part of the client).
- Most Unix workstations have the file **`/etc/resolv.conf`**

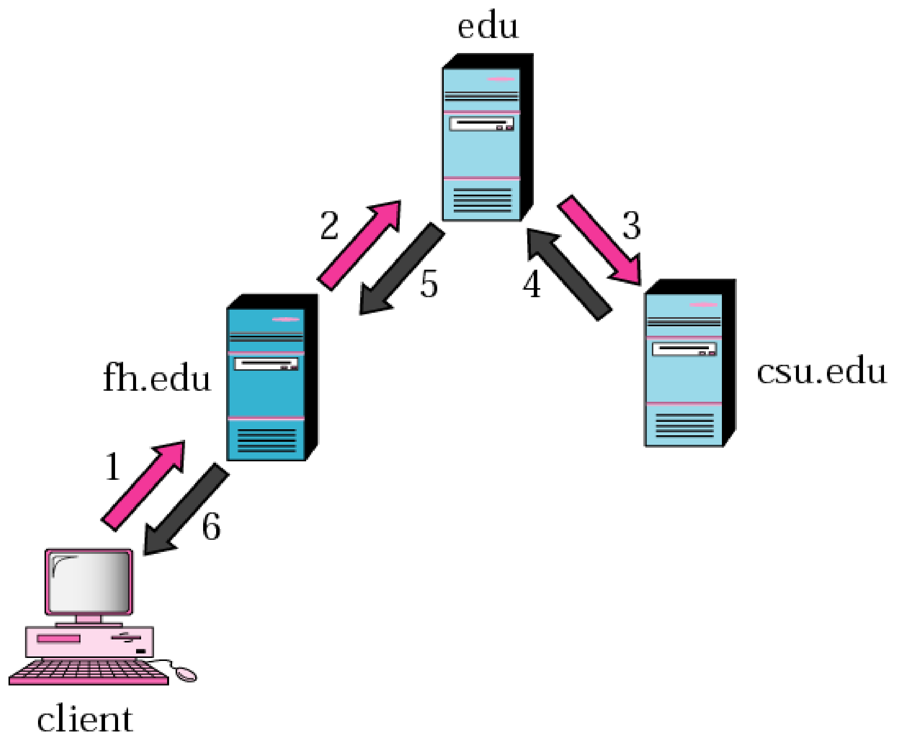
that contains the local domain and the addresses of DNS servers for that domain.

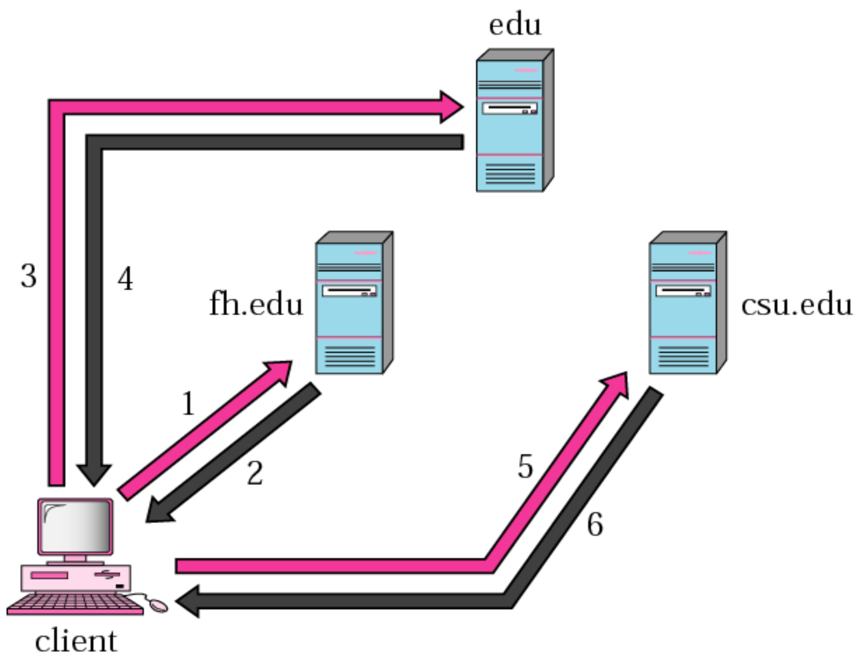
Recursive / Iterative resolution

Name – address Resolution mapping

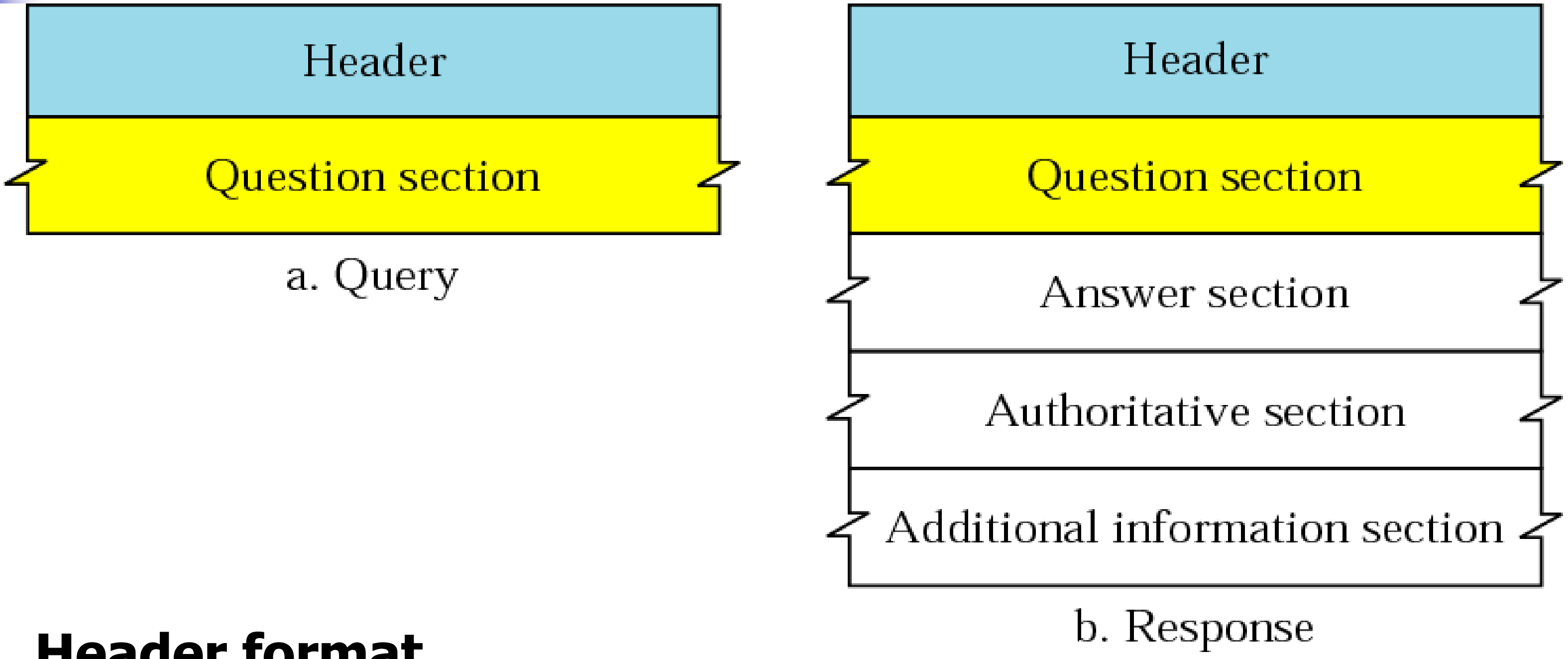
a name to an address or

an address to a name





Query and response messages



Header format

2 bytes	2 bytes
Identification	Flags
Number of question records	Number of answer records (All 0s in query message)
Number of authoritative records (All 0s in query message)	Number of additional records (All 0s in query message)