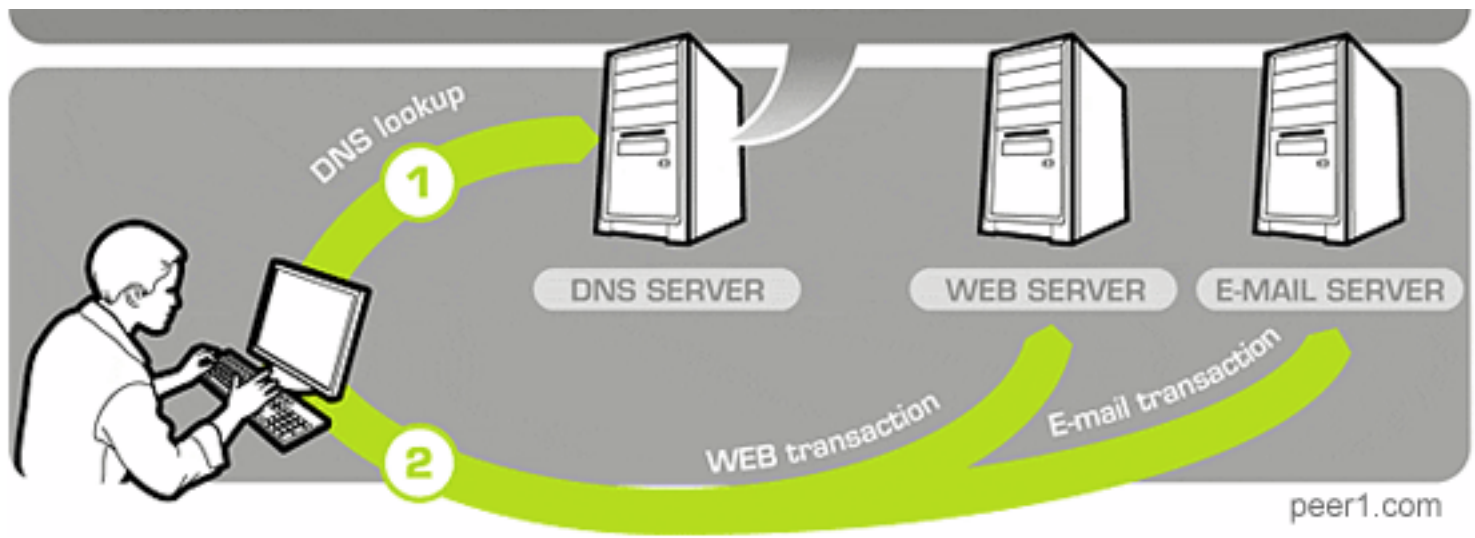
the Domain Name System service



DNS, Domain Name System

- Domain Name System (DNS) converts the name of a Web site or any host. service or resource to an IP address www.linuxhomenetworking.com ➛ 65.115.71.34
- important, as the IP address not the name, is used in routing traffic over the Internet
- without DNS access, a host would be as disconnected from the Internet

DNS, Domain Name System

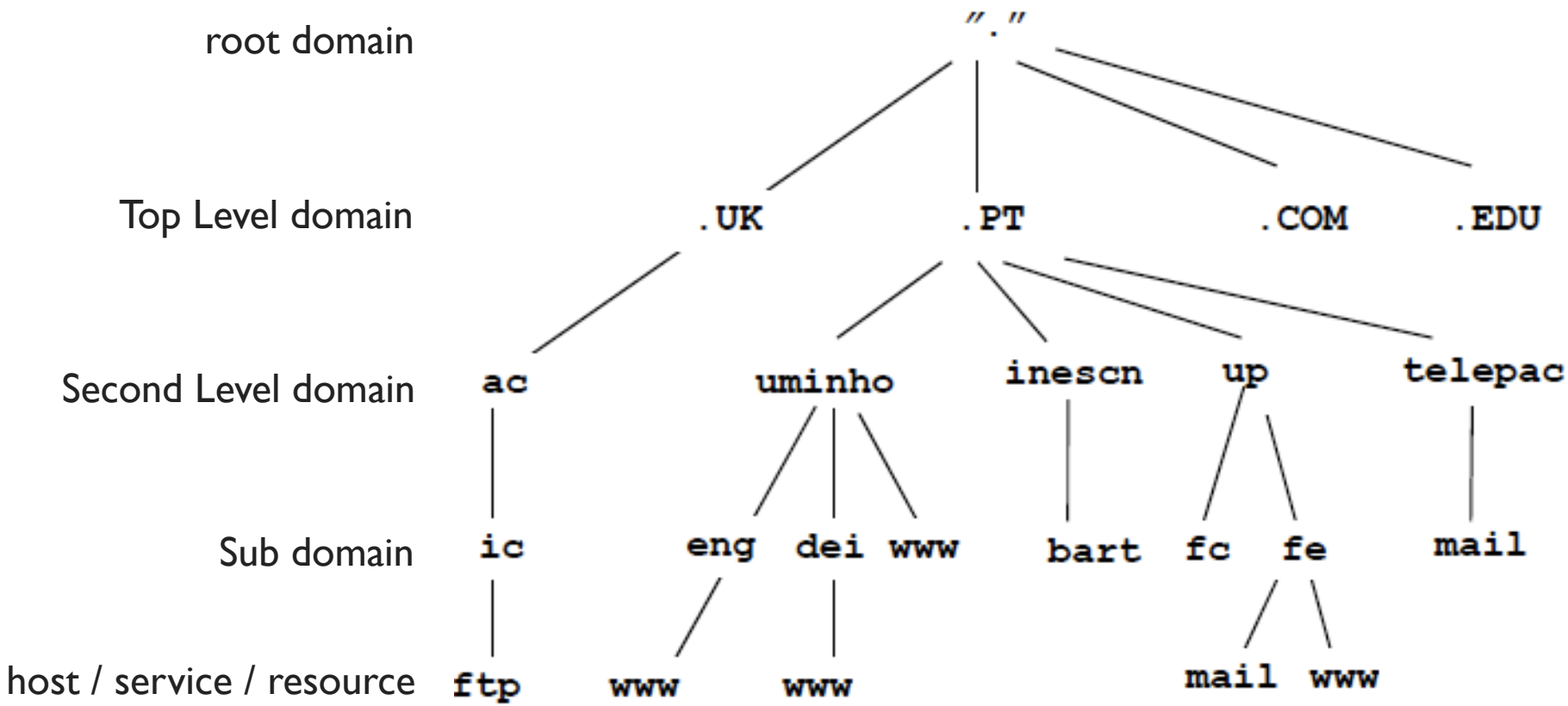
- converts the name of a host , server, site, into an IP address
- there are in the whole Internet a limited number (13) of Authoritative DNS servers
  - ultimately these are the ones to be contacted (hierarchy of DNS servers)
  - maintain the registry of all DNS servers in the Internet
- the alternative to DNS
  - to memorize all IP addresses !!!
  - maintain a local list
    - in Unix-based systems there's a *hosts* file in the */etc* directory

```
#
#       TCP/IP hosts database
#
127.0.0.1         localhost loopback lb
192.35.246.1      animal.inescn.pt animal
192.35.246.7      gonzo.inescn.pt
192.35.246.9      bart.inescn.pt bart
```

    - however it is not practical nor feasible to maintain updated

DNS

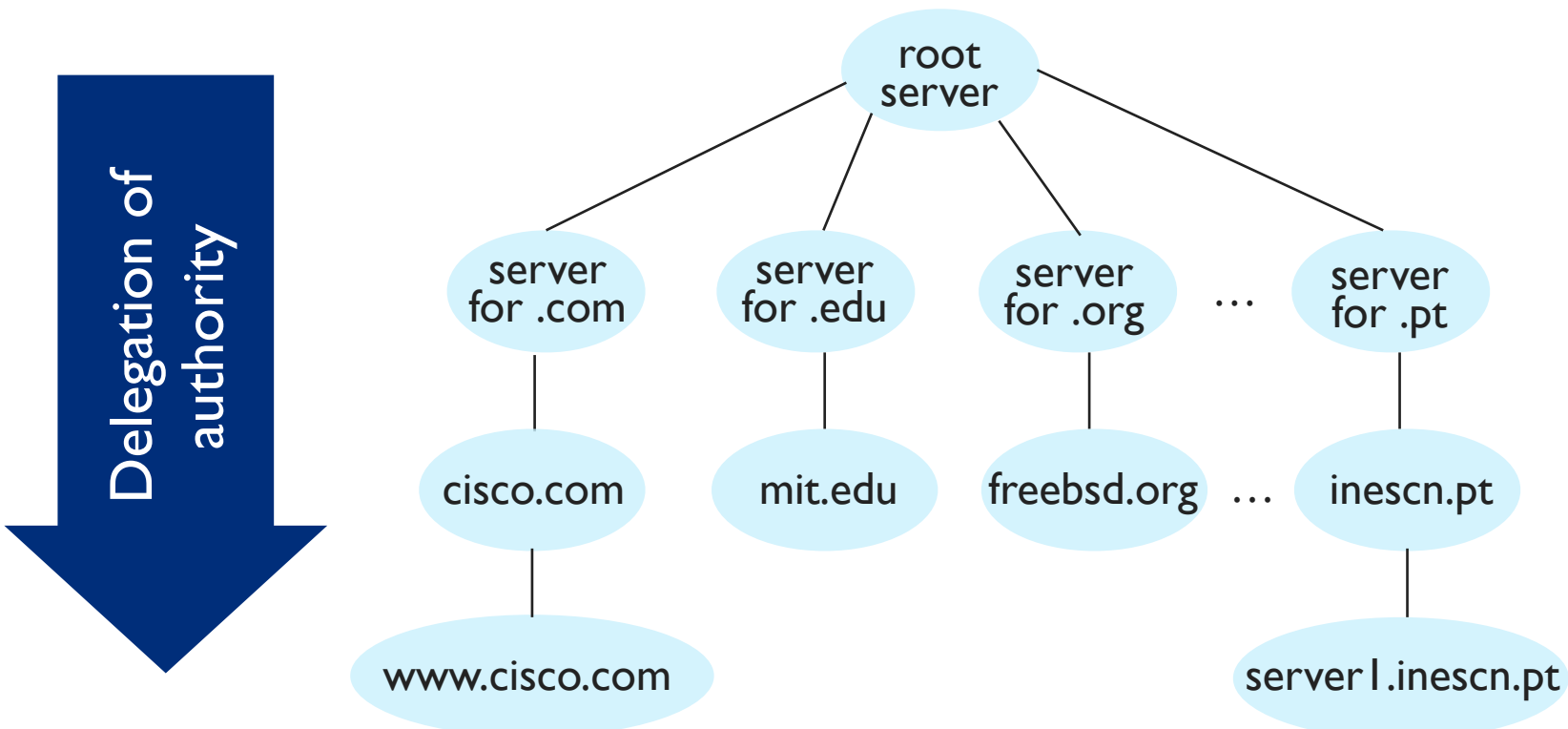
- DNS Hierarchy



host / service . [subDomain] . [Domain] . [TopLevelDomain]

DNS hierarchy

- Delegation of authority
  - root server handles (has authority for) all Top Level Domains
- TLDs servers have authority for organizations, etc
  - organizations' servers have authority for their different sub-domains or single resources



DNS hierarchy - root servers

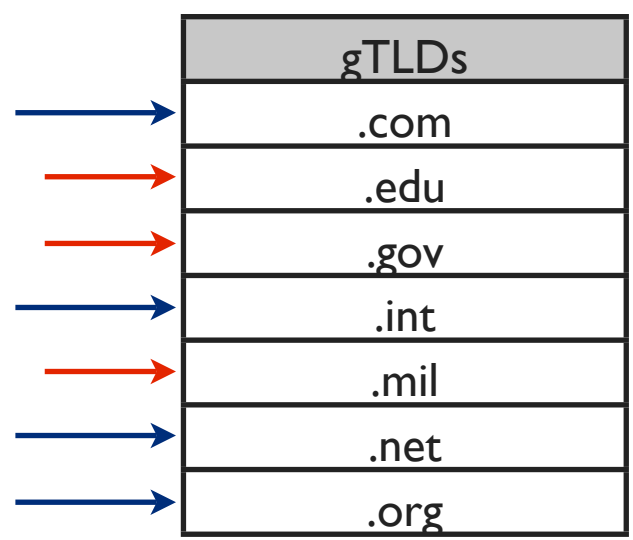
- root servers
  - authoritative name servers that serve the DNS root zone
  - network of hundreds of servers in many countries
- configured in the DNS root zone as 13 named authorities

Hostname	IP Addresses	Manager
a.root-servers.net	198.41.0.4, 2001:503:BA3E::2:30	VeriSign, Inc.
b.root-servers.net	192.228.79.201	University of Southern California (ISI)
c.root-servers.net	192.33.4.12	Cogent Communications
d.root-servers.net	128.8.10.90, 2001:500:2D::D	University of Maryland
e.root-servers.net	192.203.230.10	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241, 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4	US Department of Defence (NIC)
h.root-servers.net	128.63.2.53, 2001:500:1::803f:235	US Army (Research Lab)
i.root-servers.net	192.36.148.17, 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30, 2001:503:c27::2:30	VeriSign, Inc.
k.root-servers.net	193.0.14.129, 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42, 2001:500:3::42	ICANN
m.root-servers.net	202.12.27.33, 2001:dc3::35	WIDE Project

DNS

- Initial Top Level Domains (TLDs)
  - ccTLDs (Country Codes)
    - ISO 3166
  - gTLDs (Generic codes)
    - ugTLDs (Unsponsored)
    - sgTLDs (Sponsored)
  - gTLDs divide the ARPAnet

ccTLDs	
Portugal	.pt
Hungary	.hu
Brasil	.br
Germany	.de
France	.fr
Denmark	.dk
United Kingdom	.uk
...	



DNS

- Some well known TLDs

Domain	Type of Organization
.com	commercial organizations
.edu	educational institutions
.org	non-profit organizations
.net	backbone networks
.gov	non-military government organizations
.mil	military government organizations
.xx	two-letter country code
.arpa	Address and Routing Parameters Area (ARPA) - reverse DNS

DNS

- Registry in the .pt domain
  - IANA has delegated upon FCCN (Fundação para a Computação Científica Nacional) the management of the ".pt" domain
  - Top level domains
    - .pt
    - .com.pt
    - .edu.pt
    - .gov.pt
    - .int.pt
    - .net.pt
    - .nome.pt
    - .org.pt
    - .publ.pt
  - in http://www.dns.pt

DNS decentralizing

- delegating authority for parts of the namespace
  - distributing responsibility for the mapping between names and addresses
- allows the existence of multiple hierarchies
- an organization is partitioned into divisions
  - appoints "executive" responsible for each division
- divisions have autonomy within specified limits
- modifications within a division do not necessarily affect the top level
  - for a given division, mappings are performed inside the division
- one given name may map to more than one address
  - different types may be specified (Web server, email server, mailbox, client, etc)

DNS organization

- DNS is divided into domains and sub-domains (zones), organized in a hierarchy
  - the name of hosts in the network is generally of the form
    - *nodeName.subDomain.Domain.mainDomain*
  - *foo.theCompany.com.pt* is part of the sub-domain *theCompany*, which is part of the domain *com.pt*, which in turn is part of the *pt* main domain
  - the top is the root zone, upon which the entire DNS system depends
- For each domain
  - one DNS *primary* server that is responsible for providing information about it and for looking up information in other domains
  - possibly several *secondary* servers with copies of information from the primary, acting as backups (not only)

DNS servers

- a single domain may be hosted by multiple servers, but only one is the master - all the rest are slaves

- For a domain hosted by a DNS server to be available to DNS clients that do not query that server directly, it must be registered in the parent zone

- DNS servers can be grouped in
  - **Authoritative**
    - Primary / Master
    - Secondary /Slave
  - **Non-authoritative**

DNS zones and servers

- Authoritative servers
  - those that have been designated to answer authoritatively for the designated zone
  - need to be listed in the delegation (in the parent zone)
  - provide original and definitive answers to DNS queries
    - not just cached answers, obtained from another name server

∴ an authoritative server only return answers to queries about domain names that are installed in its configuration system

- Two types of Authoritative Name Servers:
  - **Master** nameservers
  - **Slave** nameservers

DNS zones and servers

- **Master** or Primary
  - stores the original master copies of all zone records
  - eventual changes in the zone records are made only in the master server
  - only one master per domain
- **Slave** or Secondary
  - exact replica of master server (identical copy of the master records)
    - gets updates via special automatic updating mechanism of the DNS protocol
  - used to share DNS server load and to improve DNS zone availability in case master server fails
- recommended to have at least 2 slave servers and one master server for each domain name
- Slaves do not act only as a backup in case of failure of the Master
  - enable to balance the load
  - are authoritative for their domain

DNS zones and servers

- to find out the authoritative name servers
  - in Linux, Unix and MacOS

```
> host -t ns fe.up.pt

fe.up.pt name   server   magoo.fe.up.pt.
fe.up.pt name   server   ns1.fe.up.pt.
fe.up.pt name   server   ns2.fe.up.pt.
```

```
> nslookup
> set query=ns
> dnsknowledge.com

Server:         193.136.28.10
Address:        193.136.28.10#53

Non-authoritative answer:
dnsknowledge.com  name server=ns2.softlayer.com.
dnsknowledge.com  name server=ns1.softlayer.com.

Authoritative answer can be found from:
ns1.softlayer.com   internet address = 67.228.254.4
ns1.softlayer.com   has AAAA address 2607:f0d0:0:f:1::1
ns2.softlayer.com   internet address = 67.228.255.5
ns2.softlayer.com   has AAAA address 2607:f0d0:0:f:2::1
> exit
```

DNS zones and servers

- to find out the authoritative name servers (2)

```
> dig ns fe.up.pt

; <<>> DiG 9.4.3-P3 <<>> ns fe.up.pt
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41688
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 5

;; QUESTION SECTION:
;fe.up.pt.                      IN      NS

;; ANSWER SECTION:
fe.up.pt.               86400   IN      NS      magoo.fe.up.pt.
fe.up.pt.               86400   IN      NS      ns1.fe.up.pt.
fe.up.pt.               86400   IN      NS      ns2.fe.up.pt.

;; ADDITIONAL SECTION:
ns1.fe.up.pt.           86400   IN      A       193.136.28.10
ns1.fe.up.pt.           86400   IN      AAAA        2001:690:2200:9a01::10
ns2.fe.up.pt.           86400   IN      A       193.136.28.9
ns2.fe.up.pt.           86400   IN      AAAA        2001:690:2200:9a01::9
magoo.fe.up.pt.             86400   IN      A       193.136.28.37

;; Query time: 3 msec
;; SERVER: 193.136.28.10#53(193.136.28.10)
```

```
> dig +short ns fe.up.pt

ns2.fe.up.pt.
ns1.fe.up.pt.
magoo.fe.up.pt.
```

DNS zones and servers

- Non-authoritative name servers
  - do not contain copies of any domains
  - only a cache file built from all the DNS lookups performed in the past (only authoritative responses)
  - when they query an authoritative server, they pass the answer to the client and thus provide an authoritative reply
  - most often, they answer with a previous lookup retrieved from its lookup cache
    - any such answer is non-authoritative because it did not come from an authoritative server
  - they are simple; do not maintain a database; enable to speed up the name resolution process

DNS zones and servers - summary

- normally more than one domain name server for any domain
- Authoritative servers
  - the main nameserver is called primary / master
  - the extra nameservers can be seen as backups and as load distributors
    - secondary / slave name servers
  - they automatically download the names database from the primary nameserver
    - a change in the name database takes some time to propagate
  - authoritative nameservers only return answers to queries about domain names that have been specifically configured by the administrator
    - authoritative answers
- Non-authoritative servers
  - load distributors
  - may pass the query they have received onto other ns
  - may return authoritative and non-authoritative (cached) answers

DNS model

- DNS works according to a client-server model
  - the server is called *nameserver*
    - program that replies to queries from other programs , using the hosts database
  - the client is called *resolver*
    - set of methods invoked by the user applications
    - sends requests to the *nameserver*; processes the received replies; sends back to the application the requested data
- port 53 is reserved for DNS traffic
  - either TCP or UDP
- main components of DNS
  - resolver
  - nameserver
  - database of resource records (RRs)

DNS model

- *resolver*

NAME

res_query, **res_search**, **res_mkquery**, **res_send**, **res_init**, **dn_comp**, **dn_expand**, **dn_skipname**, **ns_get16**, **ns_get32**, **ns_put16**, **ns_put32** -- resolver routines

SYNOPSIS

```
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/nameser.h>
#include <resolv.h>

int
res_query(const char *dname, int class, int type, u_char *answer,
    int anslen);

int
res_search(const char *dname, int class, int type, u_char *answer,
    int anslen);

int
res_mkquery(int op, const char *dname, int class, int type,
    const u_char *data, int datalen, const u_char *newrr_in, u_char *buf,
    int buflen);
```

DESCRIPTION

These routines are used for making, sending and interpreting query and reply messages with Internet domain name servers.

DNS queries

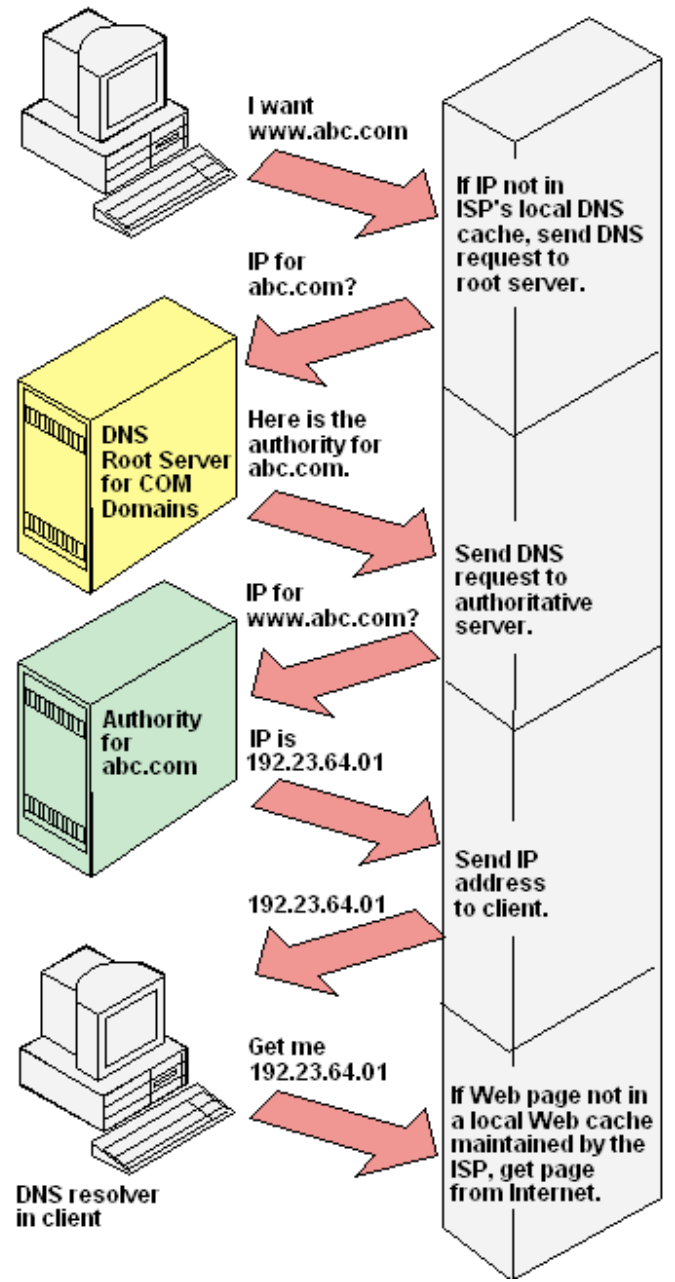
- can be sent from a DNS client (resolver) to a DNS server (nameserver)
- or between two DNS servers
- two types of DNS queries
  - Recursive (forward lookup query)
  - Iterative
- normally, clients issue recursive queries
- the DNS server is forced to send a reply
  - either successful or failure
  - if the query cannot be resolved from local data (local zone files or cache of previous queries), the DNS server must contact any other DNS servers it needs to resolve the request
  - the recursive query must be escalated to a root DNS server

DNS queries

- iterative queries
  - the DNS server is expected to respond with the best local information it has
    - based on what the DNS server knows from local zone files or from caching
    - if it does not have any local information that can answer the query, it simply sends a negative response

DNS queries

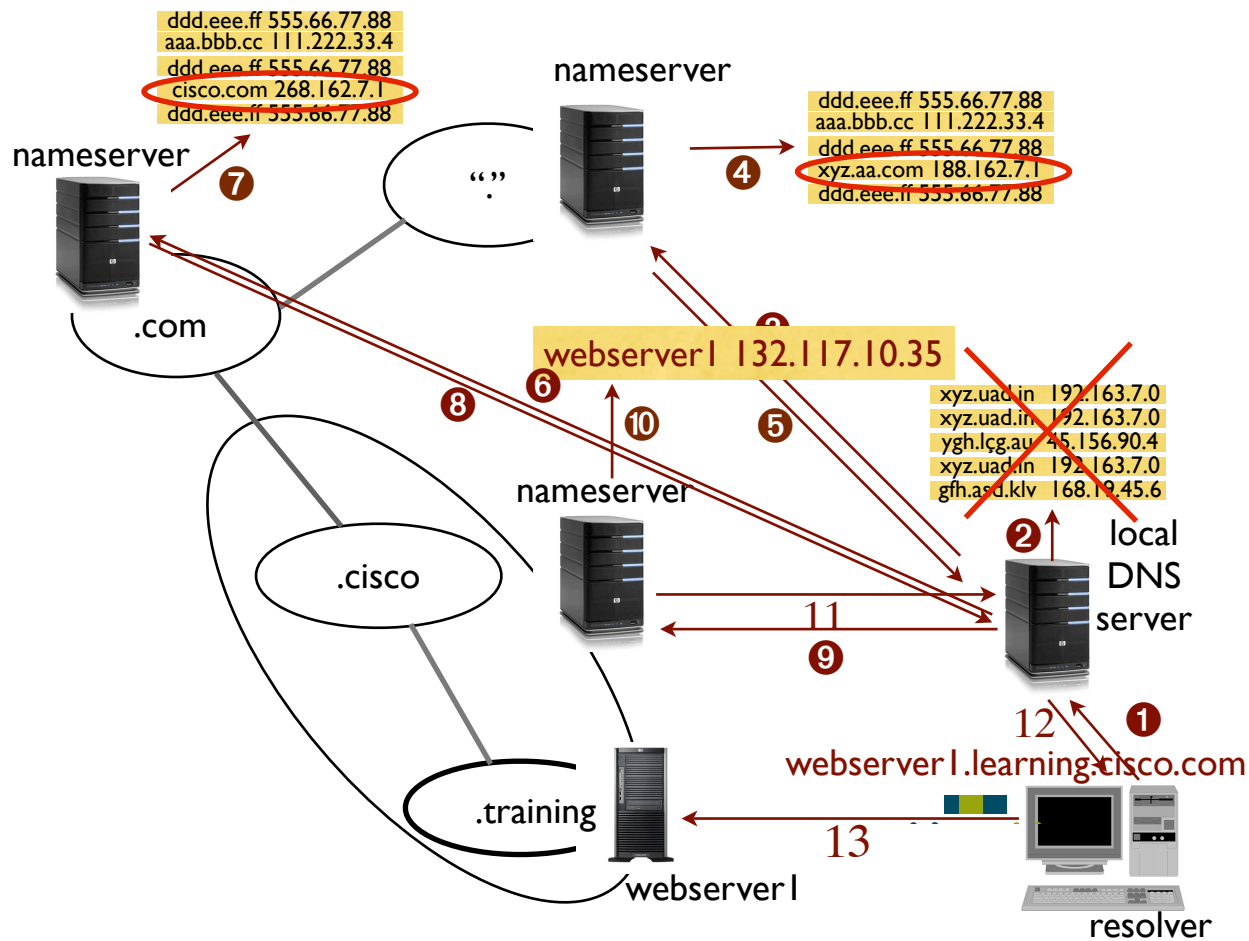
- recursive or forward lookup query
  - *resolver* sends a recursive query



source: http://images.yourdictionary.com/DNS

DNS queries

- forward lookup query
  - a user wants to contact "webserver1.training.cisco.com"
    - the resolver sends a request with a fully qualified domain name



DNS caching

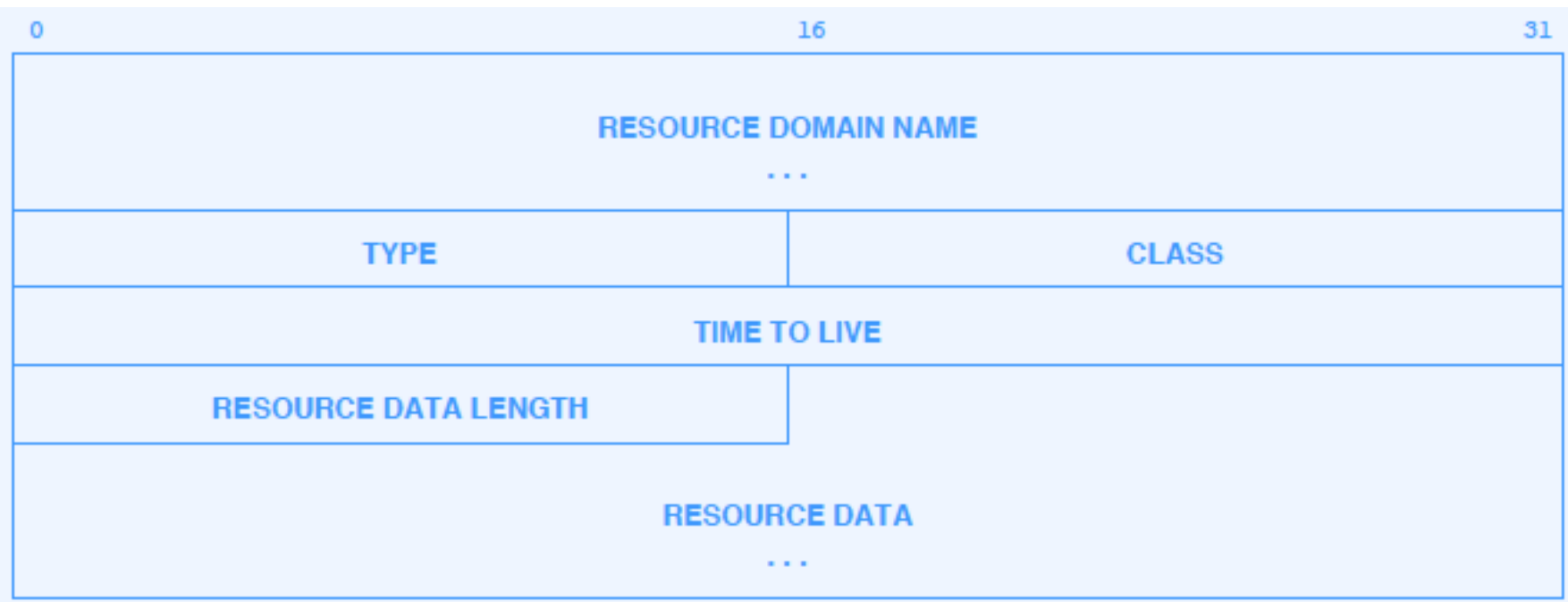
- lookup queries represent a heavy burden
  - even if queries go directly to the DNS server that has the authority
- to improve performance of the DNS system, caching is used for non-local names
  - each server keeps in cache a list of recently used names
  - and the indication of where that info was obtained
- when client issues a request, the DNS server
  1. checks if it has authority for that name
  2. checks its cache
- if the data is obtained from the cache
  - the reply is sent as *non-authoritative binding* and the name of the server is sent as well
  - client may choose to use directly the provided binding or to contact the server

DNS caching

- names may change
- caches need to be updated
  - servers mark the time of each entry in the cache
  - after exceeding a given period of time, entries are removed
  - this period is set by the authoritative server
- when a authoritative server replies with the name-address binding, it includes the parameter TTL (*Time To Live*)
- both local name servers as well as clients may cache data
  - resolver software caches DNS entries in the host itself

DNS database

- a given name may map to more than one address
  - the client must specify what type of object (host/ resource) it is looking for, when resolving a name
- the DNS server returns objects of that type
  - it keeps in its DB one record per resource with varied information, among which the type (*class*)



DNS Generic Record Format

- textual format and a binary or wire-format
- The textual format has the following generic form:
  - `name ttl class type type-specific-data`
  - `ttl` ➛ 32 bit value. The Time to Live in seconds (range is 1 to 2147483647) and indicates how long the RR may be cached. The value zero indicates the data should not be cached
  - `class` ➛ 16 bit value which defines the protocol family or an instance of the protocol. The normal value is IN = Internet protocol
  - `types` ➛ resource record type which determines the value(s) of the type-specific-data field

DNS database

- types of records

value	description
A	assigns a 32-bit IP address to a domain or subdomain name
AAAA	assigns a 128-bit IP v6 address to a domain or subdomain name
CNAME	canonical name that makes one domain name an alias of another
MX	maps a domain name to a list of mail exchange servers for that domain
PTR	maps an IPv4 address to the canonical name for that host; implements reverse DNS lookup for that address
NS	maps a domain name to a list of DNS servers authoritative for that domain
SOA	start of authority record specifies the DNS server providing authoritative information about an Internet domain, the email of the domain
SRV	allows to find the host name that provides a given service on behalf of the domain
TXT	allows an administrator to insert arbitrary text into a DNS record
NAPTR	Naming Authority Pointer

DNS database

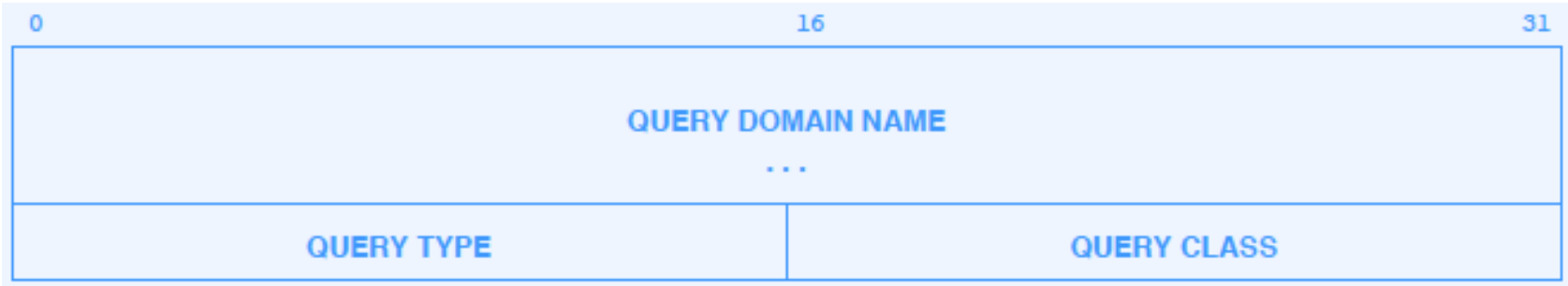
- **A** record
  - example.com IN A 98.76.54.3
    - IN ⇒ Internet
    - A ⇒ Address record type
- **CNAME**
  - associates a new subdomain to an already existing A record
    - mail.example.com IN CNAME mail.example.net
- **MX**
  - all emails @ mydomain.com should be routed to the mail server at mydomain.com
    - mydomain.com. 14400 IN MX 0 mydomain.com
- **NS**
  - maps a domain name to a list of DNS servers authoritative for that domain
  - example.com. IN NS ns1.live.secure.com
- **SOA**
  - specifies the DNS server providing authoritative information about a domain, the email of the administrator, the serial number, and timers relating to refreshing the zone

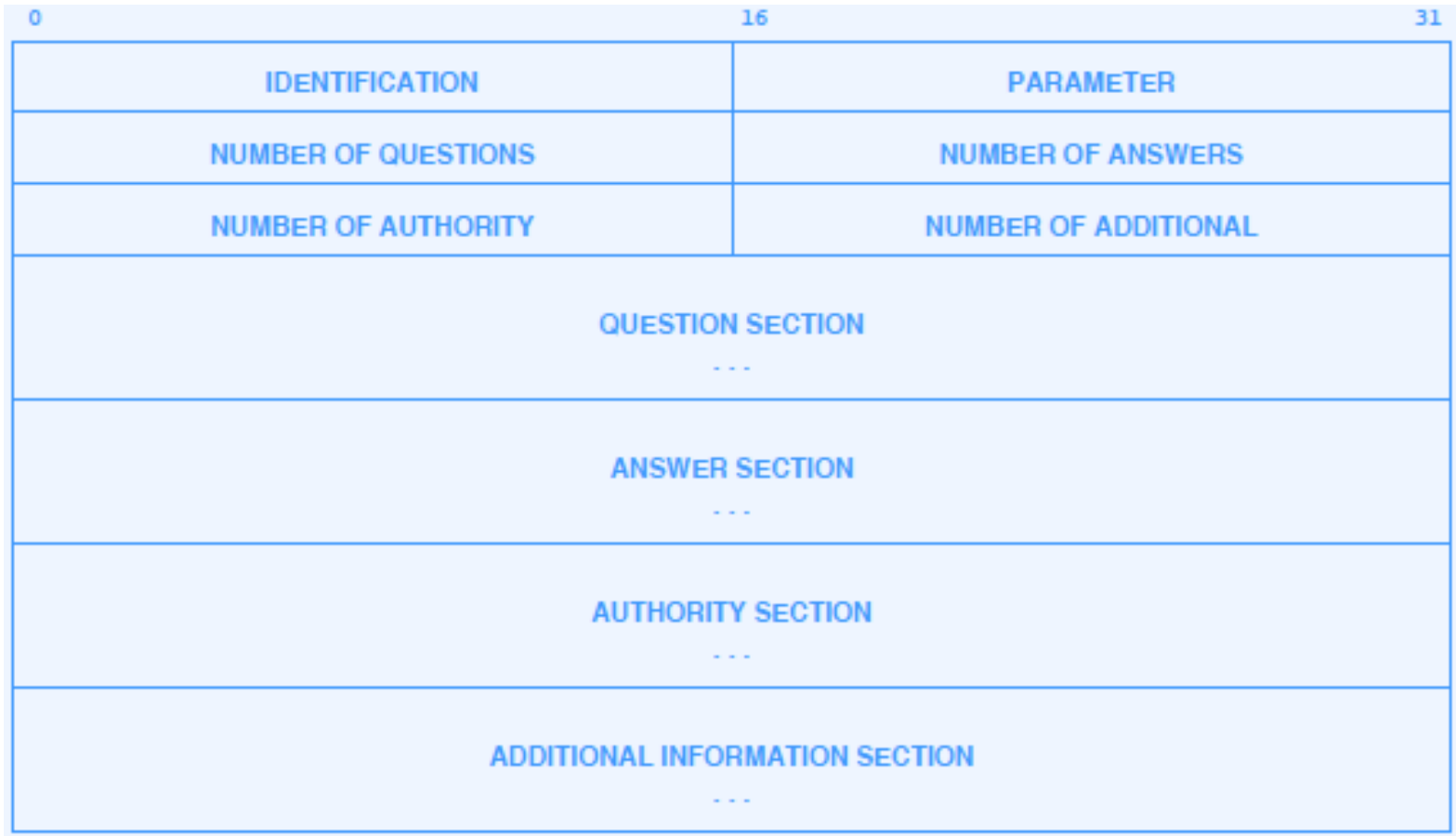
DNS database

- **SOA** record (Start Of Authority)

```
; name TTL class rr Nameserver email-address
mydomain.com. 14400 IN SOA ns.mynameserver.com. root.ns.mynameserver.com. (
2004123001 ; Serial number
86000 ; Refresh rate in seconds
7200 ; Update Retry in seconds
3600000 ; Expiry in seconds
600 ; minimum in seconds )
```

DNS query & response format





split DNS

- within an organization, hosts are divided into internal (*inside local* addresses) and external (*inside global*)
- Internal hosts access external resources by having their internal addresses rewritten (using NAT) to an address within the global address block

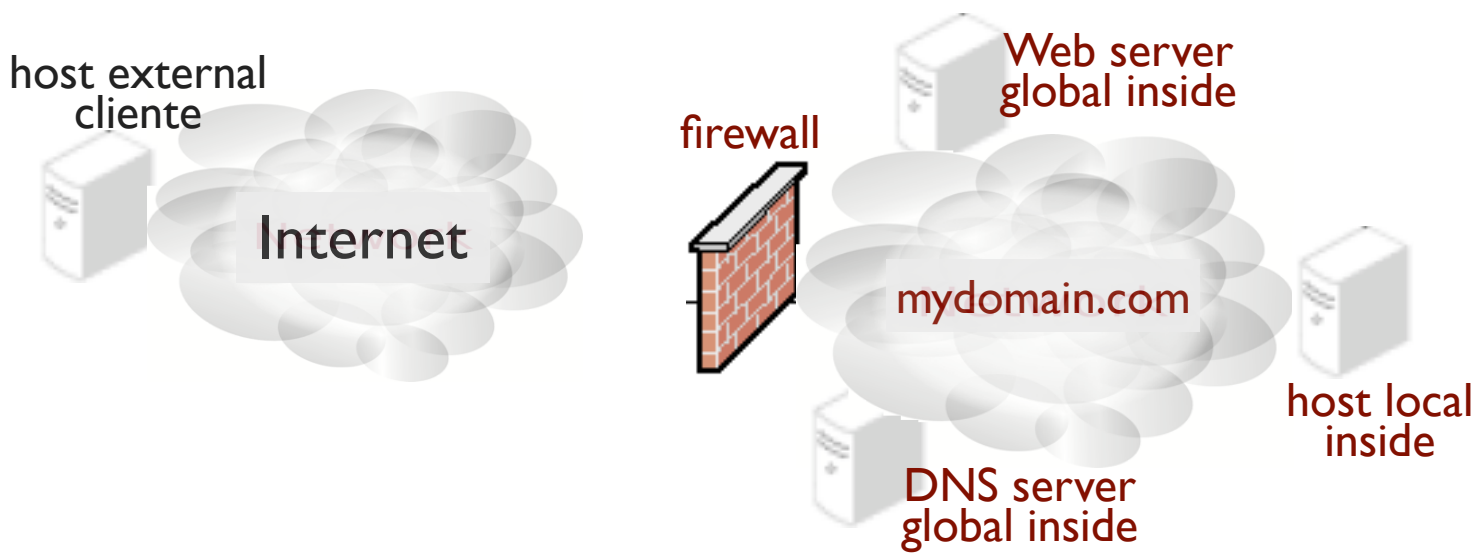
- The main concept of split DNS is
  - internal hosts are directed to an internal domain name server for name resolution
  - external hosts are directed to an external domain name server for name resolution
    - internal names (inside local addresses) are not disclosed for the outside world

split DNS

- Split DNS is simply a configuration in which the IP address to which a DNS name resolves is dependent on the location of the client
  - most often used in a NAT environment
  - insures that local clients resolve the DNS names of local servers to their RFC 1918 addresses ("Address Allocation for Private Internets" - inside local or internal)
  - and external clients resolve the same server names to their public counterparts
  - internal addresses are not disclosed to the public
- The responses to queries for names, depend upon whether the query is resolved in a local or a global context
  - whether the query is sent from a local or global address

Example: **non-split** versus split DNS

- Setup:
  - domain *mydomain.com*
    - DNS server without split
    - Web server
    - both servers with global inside addresses (externally accessible)
    - internal client with local inside address (private)
  - external client (Internet) trying to access internal Web server
  - internal client trying to access Web server

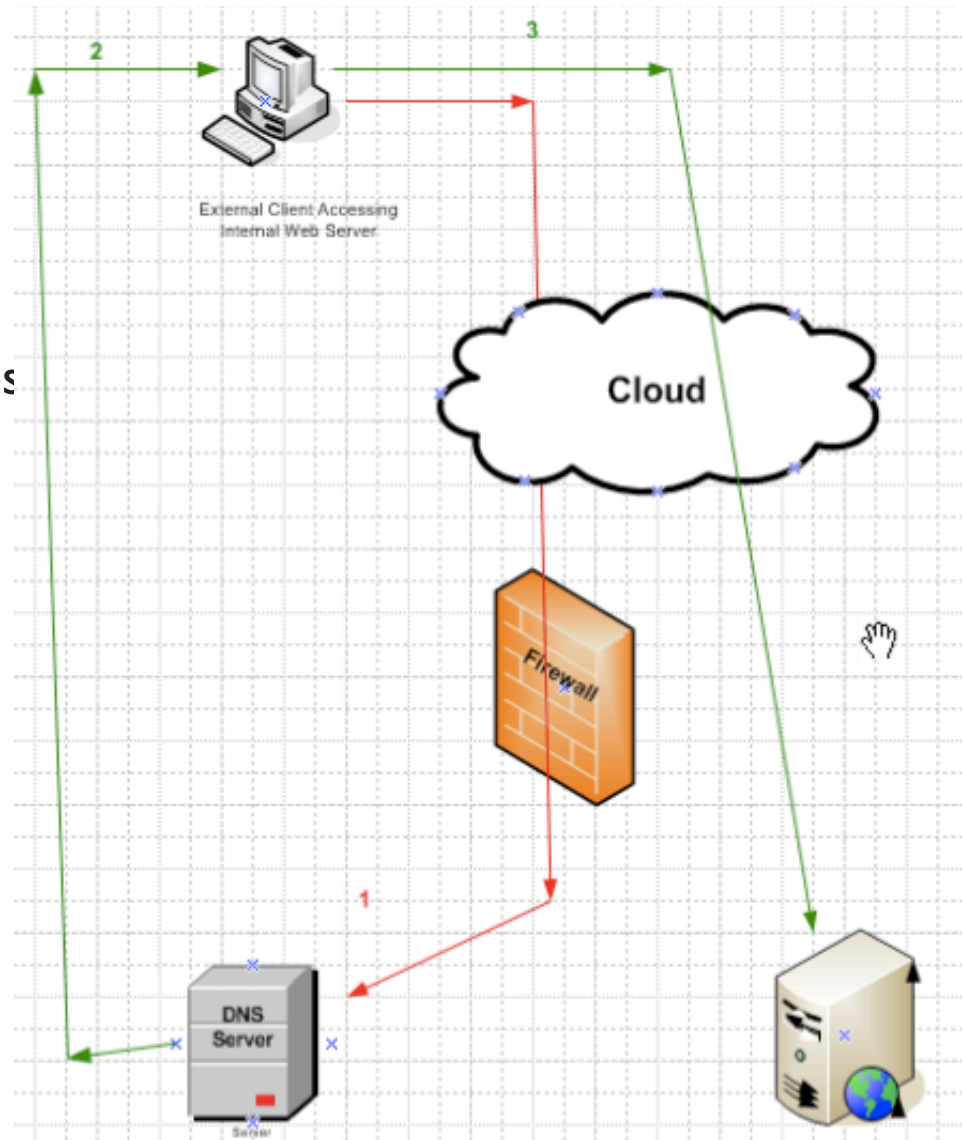


non-split versus split DNS

- non split: one single zone for mydomain.com
  - externally accessible resources: dns.mydomain.com (222.144.222.2), www.mydomain.com (222.144.222.3), mail.mydomain.com, ..
  - internally accessible resources: webserver.mydomain.com (192.168.1.3), ...
- external client (Internet) trying to access www.mydomain.com
  - resolves the address via firewall server to the dns.mydomain.com
  - the DNS server responds with the external address of the Web server (222.144.222.3)
  - the client connects to this ip accessing directly the Web server of mydomain.com

non-split versus split DNS

- external (Internet) client trying to access www.mydomain.com
  - no problems, everything goes well!

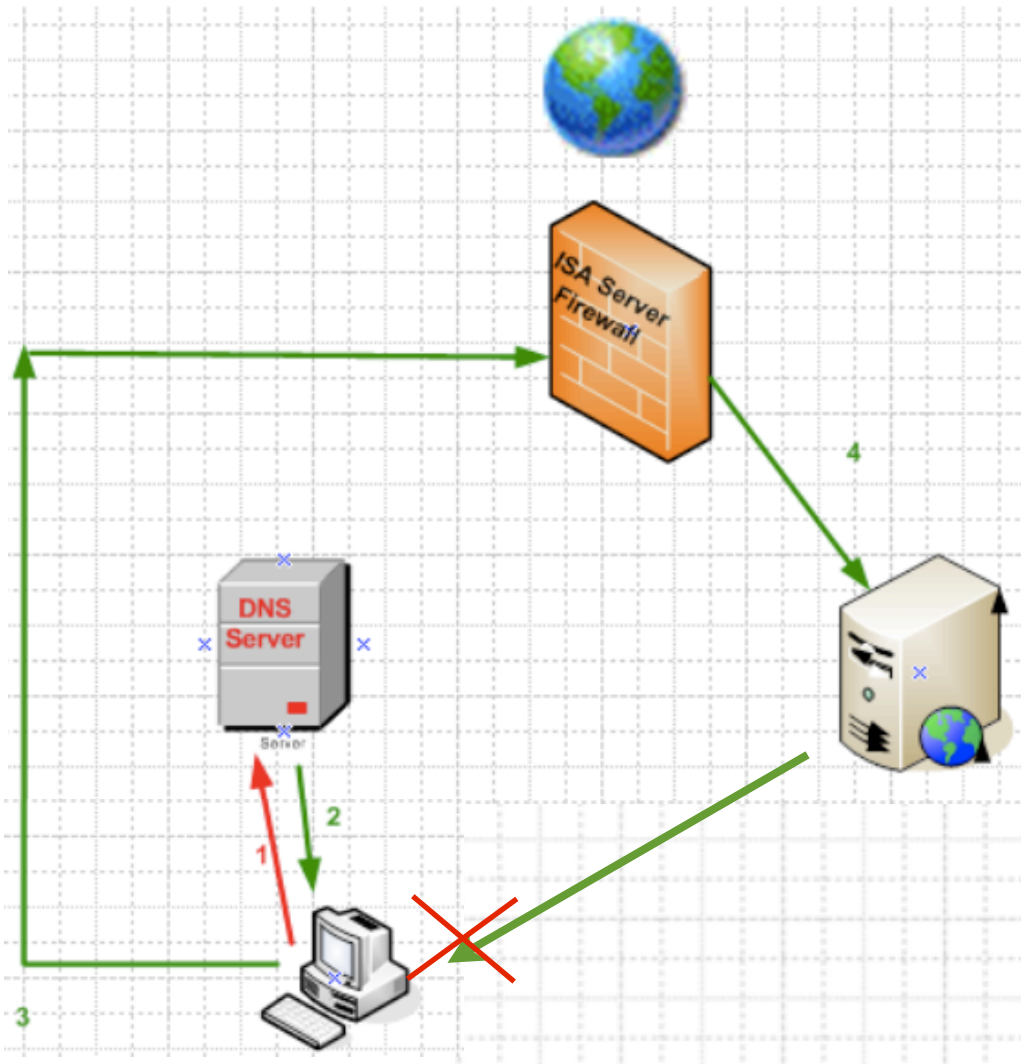


non-split versus split DNS

- internal client (inside local address) trying to access www.mydomain.com
  - queries the DNS server
  - the DNS sends back the same externally accessible ip (222.144.222.3)
  - the client connects to that ip having to go through the firewall because it is an externally visible ip
  - the packet goes through the firewall server to the Web server
- if it is a secured NAT client, then the Web server replies directly to that client (not through the firewall)
- the client rejects the reply packet because it was waiting for a reply from the firewall and not directly from the Web server

non-split versus split DNS

- internal client (secured NAT) trying to access www.mydomain.com
  - with a no-split solution, the DNS server replies always in the same way
    - regardless of the origin of the query
    - because the DNS server supplied the externally accessible address of the Web server to the local inside client
      - the client is expecting always a communication through the firewall
- but the Web server replies directly to the internal client



non-split versus **split** DNS

- split DNS allows to create more than one zone in the mydomain.com
- the external zone always resolves to externally accessible addresses
  - used when the query is received from an external client
- the internal zone has a local address table with appropriate entries for the internal network
  - maps the servers/resources names to their internal addresses
  - the servers then contact directly the internal clients using a local domain table

BIND, Berkeley Internet Name Domain

- a DNS server requires BIND (Berkeley Internet Name Domain)
  - suite of software programs that implement DNS
    - most common DNS server for Unix systems
  - most well known program is the BIND daemon, "*named*" which responds to requests
  - hosts that do not have BIND, require that the DNS server is explicitly indicated
- in all unix machines, a host learns its domain servers from the file /etc/resolv.conf

BIND views

- BIND support of *views*
  - a *view* is a set of domains that are visible to only some DNS clients
    - implementation of split DNS
  - restricts the visibility of some domains to only particular clients, identified by their IP addresses
  - useful for creating zones that are only visible to systems on an internal network, even if the DNS server is connected to the Internet
    - hide internal zones from the Internet
    - partition external (Internet) from internal (LAN) DNS information
  - each view has a unique name, and a list of matching IPs addresses and IP networks that determine which clients and servers it is visible to

BIND views

- solve the problems
  - of having a DNS server for the internal network of the company
  - of having another DNS server for external servers, for external clients, etc.
  - if only one server is configured with all internal names, public and private, the Internet will be "polluted" with private addresses
    - also it will reveal the world, part of the topology of your internal network
    - only good for possible attacker/cracker
- it's more efficient to resolve to internal IPs when you are inside and external IPs when you are outside
  - with hosts that have public and private connections.

Configuration files of BIND

- *resolv.conf*
  - in each host
- *named.conf*
  - contains all of the domains that the server hosts, as well as global configuration settings that apply to all domains
  - indicates the different DNS zones
  - for each zone, indicates the file with names to be used
- *named.conf.options*
  - for additional info such as DNS cache
- *zone files*

Configuration files of BIND

- *named.conf (excerpt)*

```
include "/etc/bind/named.conf.options";
zone "." {
        type hint;
        file "/etc/bind/db.root";
};
zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};
zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};
```

named.conf

```
    //
options {
  directory "/var/named";
  ...
// required zone for
recursive queries
zone "." {
  type hint;
  file "root.servers";
};
zone "example.com" in{
  type master;
  file "master/
master.example.com";
  // enable slaves only
  allow-transfer
{192.168.23.1;192.168.23.2;);
};
// required local host domain
zone "localhost" in{
  type master;
  file "master.localhost";
  allow-update{none;};
};
// localhost reverse map
zone "0.0.127.in-addr.arpa"
in{
  type master;
  file "localhost.rev";
  allow-update{none;};
};
// reverse map for class C
192.168.0.0
zone "0.168.192.IN-ADDR.ARPA"
in{
  type master;
  file "192.168.0.rev";
};
```

named.conf using views

```
options {...};
// other clauses/statements (as required)
view "first" {
 options{...};
       // zones clauses including 'required' zones
 zone {...};
 .....
 zone {...};
};
view "second" {
 options {...};
 // zones clauses including 'required' zones
 zone {...};
 .....
 zone {...};
};
```

Required zone files

- located in /var/named/
- root.servers
  - contains addresses of servers which can supply a list of the root servers (also called named.ca or named.root in standard BIND distributions)
  - defines a list of name servers where BIND can get a list of TLD servers for the particular TLD e.g. ".com"

```
zone "." in{
  type hint;
  file "root.servers";
};
```

- localhost
  - This zone allows resolution of the name 'localhost' to the loopback address 127.0.0.1 when using the DNS server

```
zone "localhost" in{
  type master;
  file "master.localhost";
};
```

- 0.0.127.IN-ADDR.ARPA
  - reverse mapping using the special domain IN-ADDR.ARPA

```
zone "0.0.127.in-addr.arpa" in{
  type master;
  file "localhost.rev";
};
```

DNS Sample Domain Zone file

- both externally visible (public) and internal hosts defined in this file
- 2 name servers; ns1 (internal) and ns2 (external)
- FTP and WWW services are provided by the same host
- two hosts named bill and fred
- host addresses are all in the class C private address range 192.168.0.0
- Resource Records defined separately.

```
$TTL 86400 ; 24 hours
@  1D  IN  SOA ns1.example.com. hostmaster.example.com. (
                    2002022401 ; serial
                    3H ; refresh
                    15 ; retry
                    1w ; expire
                    3h ; minimum
                   )
       IN  NS     ns1.example.com. ; in the domain
       IN  NS     ns2.smokeyjoe.com. ; external to domain
       IN  MX  10 mail.another.com. ; external mail provider
; server host definitions
ns1    IN  A      192.168.0.1  ;name server definition
www    IN  A      192.168.0.2  ;web server definition
ftp    IN  CNAME  www.example.com.  ;ftp server definition
; non server domain hosts
bill   IN  A      192.168.0.3
fred   IN  A      192.168.0.4
```

Some concerns when managing DNS

- Each domain should have at least two nameservers
- secondaries should be placed in external networks
- avoid the use of CNAME (aliases)
  - only for services such as www, ftp, or pop
  - never declare a CNAME as a NS
  - recommended syntax for the field "serial" is YYYMMDDHHnn
- the use of wildcard "*" in the MX type has the consequence of accepting email for unexistent machines