DNS

DOMAIN NAME SYSTEM

INTRODUCTION TO DNS, THE INTERNET'S DISTRIBUTED NAMING SYSTEM

Peter R. Egli
INDIGOO.COM

Contents

1. Purpose of DNS (RFC1034 / RFC1035)

DNS purpose:

DNS provides a mapping between symbolic names and IP addresses in a worldwide distributed and hierarchic database.

Addressing before DNS was introduced:

Prior to the introduction of DNS, symbolic name to IP address mappings were stored in the file hosts on each computer or host. Naturally, this scheme did not scale well because updates to the hosts file were necessary on each host every time a new host joined the network.
This file still exists and may contain static mappings, e.g. `localhost` to `127.0.0.1` and `::1`:

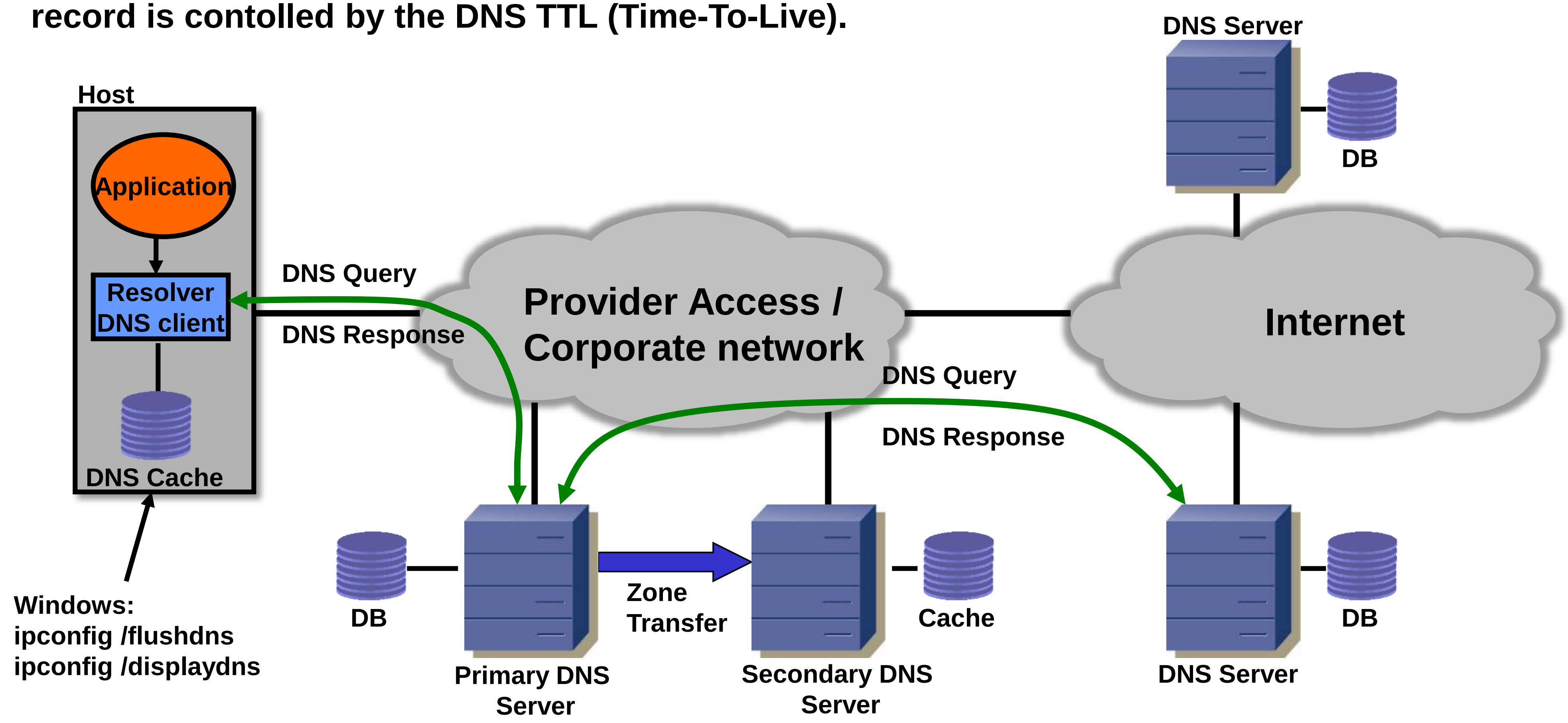
Windows:	`C:\Windows\system32\drivers\etc\hosts`
Unix / Linux:	`/etc/hosts`

DNS key characteristics:

- DNS is a distributed system (many servers cooperating, worldwide).
- Hierarchy & delegation (if one server does not know the binding, it goes up the hierarchy).
- Names are organized in a tree-structure allowing delegation of responsibility.

2. DNS elements

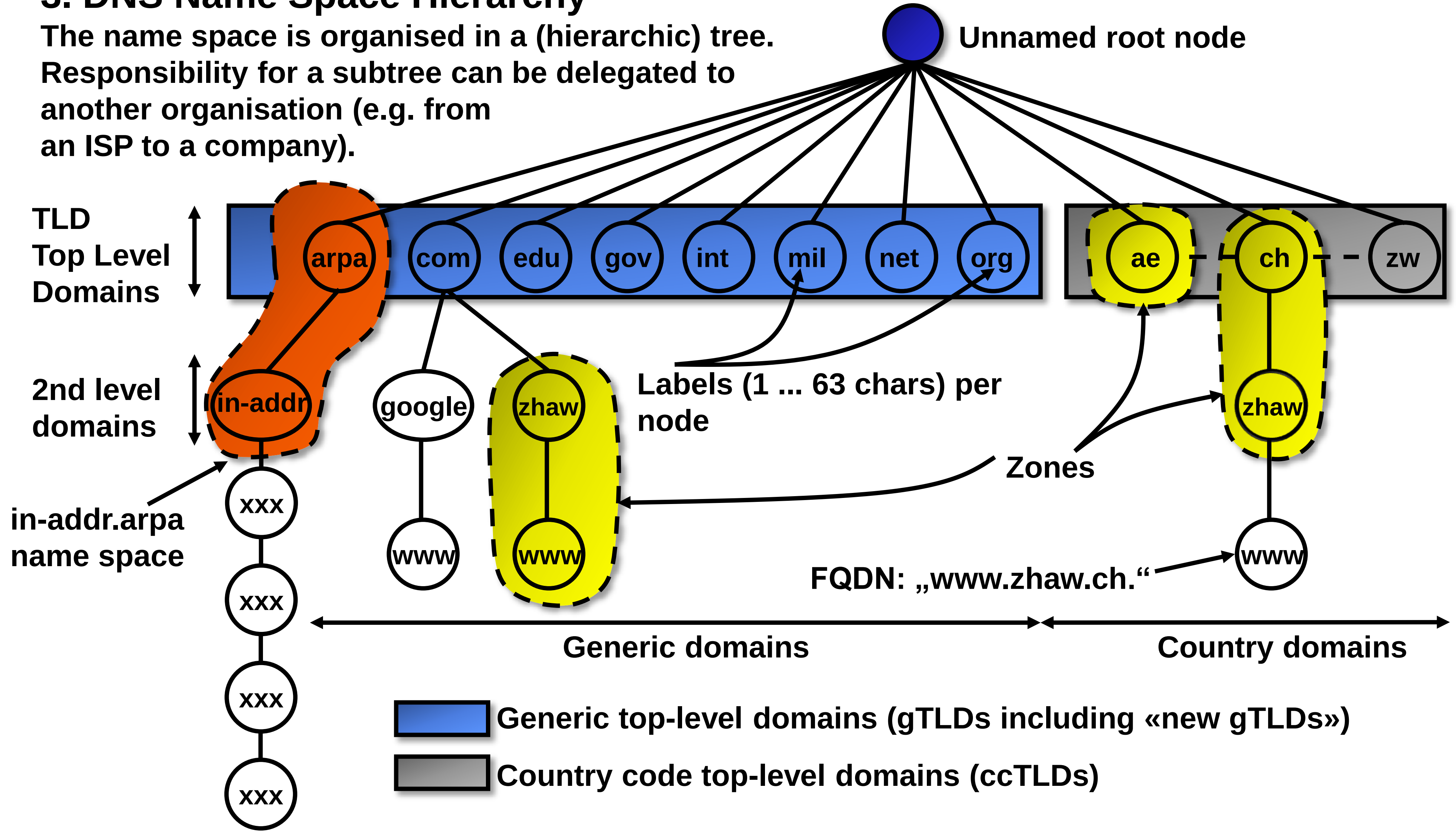
Every organisation running DNS must operate 2 DNS servers (redundancy).
Like HTTP pages, DNS records can be cached (in the client or in DNS servers). The lifetime of a record is contolled by the DNS TTL (Time-To-Live).



The primary server transfers the entire set of DNS records (mapping name to IP) in a zone transfer. Both the primary and secondary (and ternary if available) DNS server deliver authoritative records.

3. DNS Name Space Hierarchy

The name space is organised in a (hierarchic) tree.
Responsibility for a subtree can be delegated to another organisation (e.g. from an ISP to a company).



🟦 Generic top-level domains (gTLDs including «new gTLDs»)

▨ Country code top-level domains (ccTLDs)

4. DNS terms (1/4)

Root (name) server:
Servers that have in their database IPs of top level servers (gTLD servers).
Every server knows at least 2 root servers which in turn know all top level domains.
Where are the root servers? http://www.root-servers.org/

DNS client:
Performs lookups (resolver).
Sometimes the requesting application and not the resolver is called client. But from the server‘s point of view the resolver is the client.

DNS server:
a. When RR is in local database (authoritative or cached), the server returns requested RR (Resource Record with mapping name→IP).
b. When RR is not in local database, the server performs lookup on behalf of client (recursive query).
c. When RR is not in local database, the server returns IP address of DNS server higher up in the hierarchy (iterative query).

DNS resolver:
Process/program that performs name lookup on behalf of application.
Access to DNS resolver from applicaton is through OS calls: gethostbyname(), gethostbyaddress().

4. DNS terms (2/4)

Primary name server:
Authoritative name server for zone. The databases of the primary and secordary name server contain the authoritative RRs (changes to the database are made here).

Secondary name server:
Serves as (hot standby) server for primary server.
The secondary name server is also authoritative.
The secondary name server maintains a database with cached name records from the primary server (through zone transfers every 3 hours or so).

TLD (Top Level Domain):
All nodes in name tree directly underneath the root node are TLDs.
The TDLs are: arpa, com, edu, gov, int, mil, net, org, and all country domains.

Generic domain:
Top-level domains that are not country level domains: arpa, com, edu, gov, int, mil, net, org.

Resource record RR:
Record that contains mapping name→IP.

4. DNS terms (3/4)

Name space:
Defines a hierarchic tree of names and labels.

Label:
„Token“ of DNS name (the pieces between the dots). E.g. in www.zhaw.ch. *www*, *zhaw* and *ch* are labels.

Zone:
Part of name tree that is separately administered. Zones may contain smaller zones in a hierarchic way.
A zone that contains another zone delegates administration and responsibility for the name space of the contained zone (to the contained zone). Each zone must have one primary and at least 1 secondary name server (redundancy).

FQDN (Fully Qualified Domain Name):
Name that fully specifies a host. Example: www.zhaw.ch. is a FQDN.
Note: FQDNs have a dot at the end to indicate that it is an FQDN (the dot represents the root node).

Relative domain name:
All non-FQDN are relative domain names. Example: e.g. zhaw.ch.

Authoritative record:
Record that comes from the authority that manages the record (opposite: cached records in non-authoritative servers). DNS servers responsible for a zone return authoritative records (primary, secondary DNS server).

4. DNS terms (4/4)

Registrar:
In each country an organisation is responsible for name registration.
E.g. Switzerland: Switch (www.switch.ch). Germany: Denic.

Iterative vs. recursive lookup:
Recursive lookup: DNS server performs lookup on behalf of the client (if RR not contained in local database).
Iterative lookup: DNS server returns IP address of the next or root DNS server to client (if RR not contained in local database).

Pointer query:
Reverse lookup IP→name. Pointer queries may be used for verification/authentication, e.g. of email senders.

Glue record:
A (Address) record for name server that has a name within the domain served by the server. Glue records are required to break the query deadlock in referrals which return name servers in the queried domain.

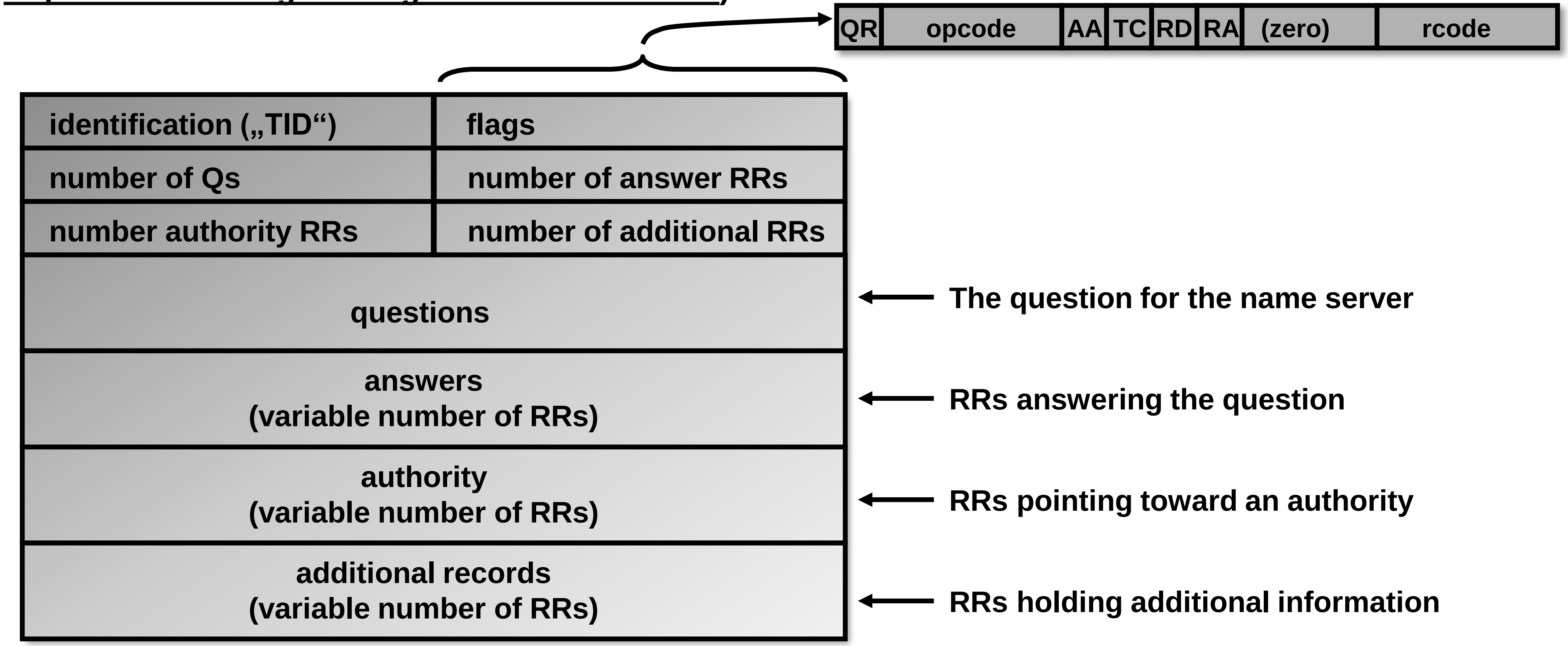
Referral:
A queried name server returns a name or address of a server that is ‚closer' to the answer domain.

5. DNS Packet

➔ DNS uses the same format for query and response.

➔ DNS uses UDP (port 53), but for large transfers (zone transfers) it uses TCP (DNS then uses format of zone file, see http://www.isoc.org/briefings/020/zonefile.shtml).

QR: query (0) response (1)
opcode: 0=standard query, 1=inverse query, 2=server status query
AA: 1=authoritative answer
TC: 1=truncated; if UDP then size exceeded 512 bytes
RD: 1=recursion desired, client can request server to handle request recursively
RA: 1=recursion available (set in response if server support recursion)
rcode: 0=no error, 3=name error


QR
opcode
AA
TC
RD
RA
(zero)
rcode
identification („TID“)
flags
number of Qs
number of answer RRs
number authority RRs
number of additional RRs
questions
answers (variable number of RRs)
authority (variable number of RRs)
additional records (variable number of RRs)
The question for the name server
RRs answering the question
RRs pointing toward an authority
RRs holding additional information

6. DNS Resource Record

A DNS record contains the information queried for (value field) in the question plus additional information on the record (TTL, class, type of record).

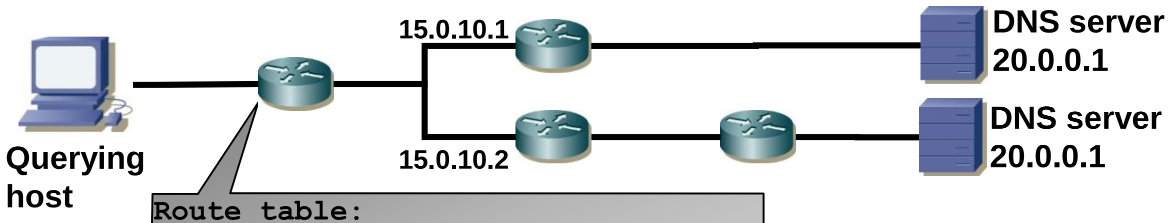
`domain_name`:		Domain name
`TTL`:		Defines how long the RR may be cached before authoritative server should be queried again (usually TTL = 1 or 2 days).
`class`:		"IN" für Internet (DNS supports also other classes such SNA, DECbit etc.)
`type`:		Kind of record.
`SOA`:	Start Of Authority	Zone admin info (primary name server name etc.).
`A`:	Address	Host IP address.
`NS`:	Name Server	Authoritative name server.
`CNAME`:	Canonical NAME	Canonical name for an alias.
`PTR`:	PoinTer Record	Pointer (IP→name).
`HINFO`:	Host INFO	Host info (host / server's type of CPU/OS).
`MX`:	Mail eXchange record	Name of host of zone that can accept mail.
`xyz`		Many other types of lesser use.
`value`:		Mapping/binding (IP address for A type).

7. DNS root servers

➔ DNS root servers are the most critical component in the entire DNS.
List of root servers: http://www.root-servers.org/.
Root servers basically publish the „root zone file“ – a file containing all names and IP addresses of all top-level domains (gTLDs and ccTLDs). See http://www.isoc.org/briefings/020/zonefile.shtml.
Root zone file excerpt:

```
…
$ORIGIN .
LU 172800 IN NS MERAPI.SWITCH.CH.
$ORIGIN SWITCH.CH.
MERAPI 172800 IN A 130.59.211.10
172800 IN AAAA 2001:620::5
$ORIGIN .
LU 172800 IN NS SUNIC.SUNET.SE.
…
```

There are 13 logical root servers, named ‚A‘ through ‚M‘, each administered by a different organisation. Some organisations (such as RIPE, K-root-server) chose to run multiple redundant physical root servers (called „mirrors“, DNS server clusters) distributed worldwide. See http://k.root-servers.org/ for RIPE‘s root servers. These mirrored root servers are reachable through IPv4 anycast (same IP address, but depending on the host‘s location the nearest root server is reached by a querying host):



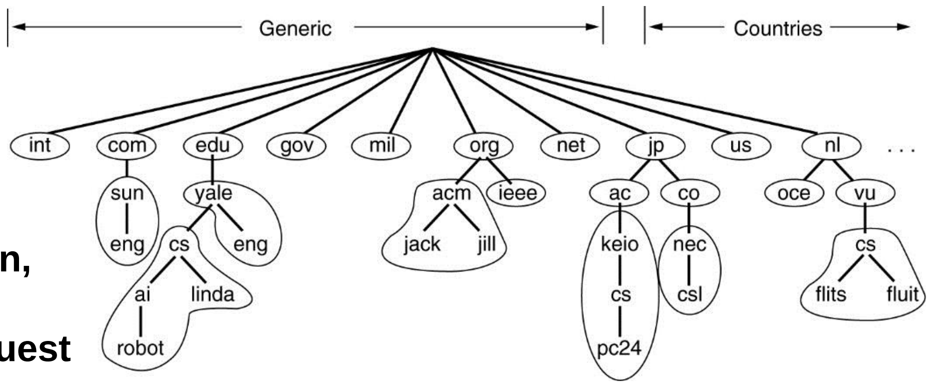
```
Route table:
Dest.     Mask  Next-Hop   Distance
15.0.0.0  29    127.0.0.1  0
20.0.0.1  32    15.0.10.1  1
20.0.0.1  32    15.0.10.2  2
```

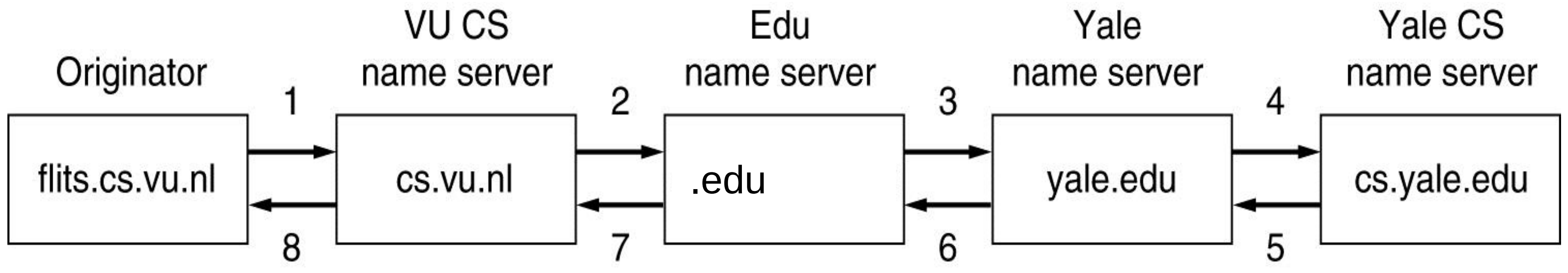
Mirrored DNS root servers are reachable through different paths. Normal IP routing ensures that the querying host reaches the „nearest“ mirrored DNS root server (BGP4 distributes routes in the network).

8. DNS Operation

Case A. Server does not know binding (not cached locally):

Example: flits.cs.vu.nl looks up linda.cs.yale.edu
1. flits.cs.vu.nl asks local DNS server cs.vu.nl.
2. cs.vu.nl does not have binding, but as per RFC1035 must have IP address of server for 'edu' = edu-server.net. cs.vu.nl forwards request to edu-server.net.
3. edu-server.net must know IP address of all of its children, so it forwards the request to yale.edu.
4. yale.edu must know all of its children so it forwards request to cs.yale.edu.
5. cs.yale.edu is authoritative server for all children underneath cs.yale.edu so it replies with the corresponding record.
6./7./8. The reply goes back to the client (through all servers).





Case B. Server knows binding (cached) and directly responds to client.

9. DNS recursive versus iterative queries

<u>Recursive:</u> The first contacted DNS server performs the lookup on behalf of the client; resolvers (clients) use recursive query.

<u>Iterative:</u> The first contacted DNS server refers the client to some other server in the hierarchy; servers use iterative query (referral).

<u>nslookup example for www.indigoo.com (via root servers):</u>

```
cmd> nslookup
> set norecursive                                       (Force iterative queries)
> set type=ns                                           (Query for name server addresses)
> .                                                     (Query for root servers)
> (root)  nameserver = i.root-servers.net
> ...
> (root)  nameserver = m.root-servers.net
> server b.root-servers.net                             (Select B root server for queries)
> com.                                                  (Query for com TLD)
> com     nameserver = a.gtld-servers.net (192.5.6.30)
> ...
> com     nameserver = f.gtld-servers.net (193.0.9.1)
> server 192.5.6.30
> indigoo.com.                                          (Query for domain indigoo.com)
> indigoo.com     nameserver = ns131.hoststar.ch
> server ns131.hoststar.ch (85.10.192.4)
> set type=A
> www.indigoo.com.                                      (Query for www.indigoo.com)
> www.indigoo.com Address:  85.10.192.4
```

10. DNS Pointer Query

Pointer query = Lookup IP address to name (= inverse lookup).

Problem:

DNS name space is hierarchic and allows quick lookup name→IP.
Because the tree is hierarchic the lookup is like a mathematical one-way function (easy lookup in one direction, virtually impossible in the other direction).

Solution:

Special name space `in-addr.arpa` in DNS name space.
Beneath this name space every organization is responsible for a portion of the in-addr.arpa name space. For example the owner of the IP address range 193.5.54.0/24 is responsible for serving pointer queries to this address.
N.B.: 33.13.252.140.in-addr.arpa. is the FQDN for a host with IP address 140.252.13.33 (note reversed order of IP address bytes).

Demo nslookup:

```
>cmd nslookup
>set type=ptr                       (set query type to pointer)
>193.5.54.112
>112.54.4.193.in-addr.arpa          (reversed IP address!)
```

11. Dynamic DNS dynDNS RFC2136

Problem:

DNS is pretty static which means it does not allow to quickly change the IP address to name binding (this takes days to propagate through the network because of caching).
Because of IP address scarcity, providers (ISPs) have fewer public IP addresses than customers (overbooking: only a portion of customers is online at any time).
This was ok some years ago but people (customers) start to run more elaborate applications like Internet Telephony.

Solution:

Dynamic DNS enhances DNS with the capability to register a name and IP address with a server.
The lookup to the server is still plain vanilla DNS, but dynDNS makes it possible to re-register the IP address with the server once it has changed (e.g. DSL access with dynamic IP addresses, DHCP).
As opposed to standard DNS, DynDNS uses very low TTL values (~2 minutes or so).

There are also proprietary protocols used for dynamic DNS name registration with a server.
Often some REST-style protocol (HTTP-based) is used.

12. IDN International Domain Names RFC5890

Problem:

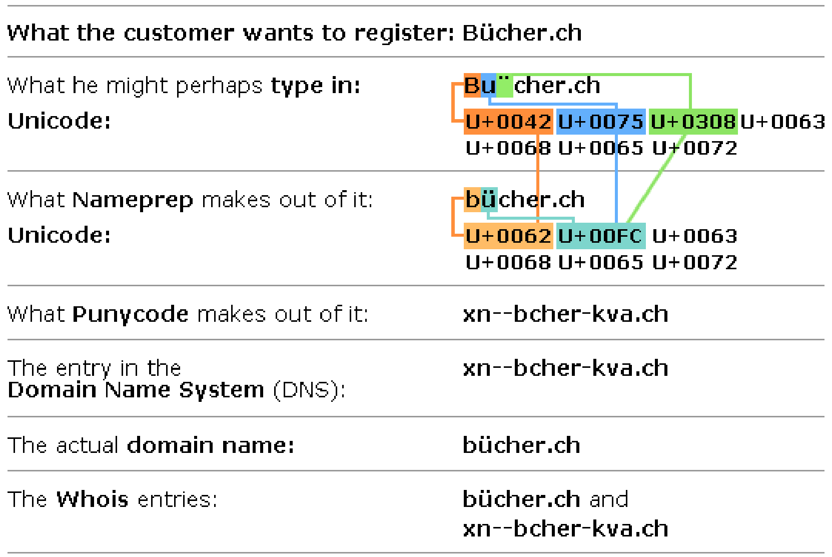
Classical DNS allows only domain names with characters from the ASCII repertoire. Special characters are not possible. Introduction of simple Unicode for DNS names is not possible because it would break backward compatibility with legacy DNS systems.

Solution:

RFC5890 introduces IDNA (International Domain Names for Applications) by allowing special characters from the Unicode repertoire to be used for domain names. The Unicode characters are encoded into an ASCII format to achieve backward compatibility with legacy DNS systems.

Encoding scheme:

IDNA uses the Punicyode encoding scheme defined in RFC3492.
Example: www.bücher.ch



What the customer wants to register: Bücher.ch	
What he might perhaps type in: Unicode:	Bu¨cher.ch U+0042 U+0075 U+0308 U+0063 U+0068 U+0065 U+0072
What Nameprep makes out of it: Unicode:	bücher.ch U+0062 U+00FC U+0063 U+0068 U+0065 U+0072
What Punycode makes out of it:	xn--bcher-kva.ch
The entry in the Domain Name System (DNS):	xn--bcher-kva.ch
The actual domain name:	bücher.ch
The Whois entries:	bücher.ch and xn--bcher-kva.ch