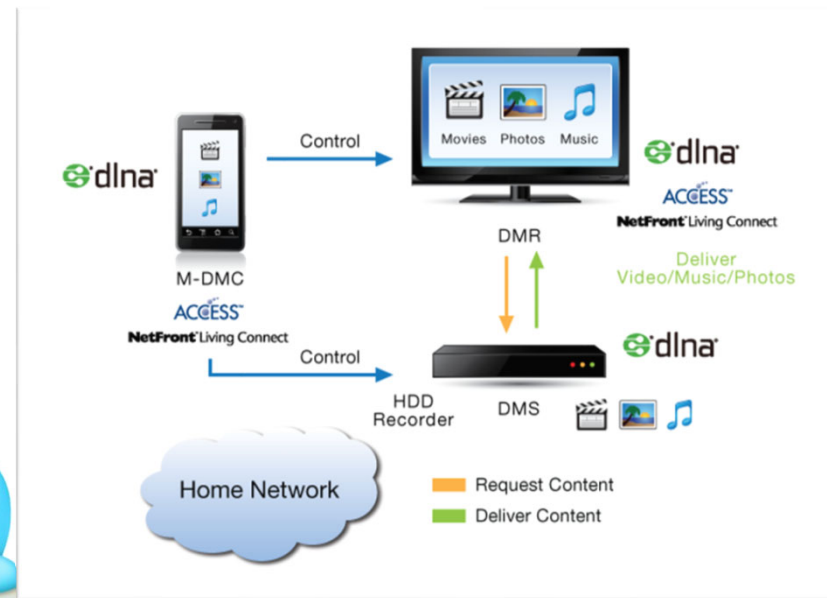




# Application Layer

Jean-Yves Le Boudec

2020

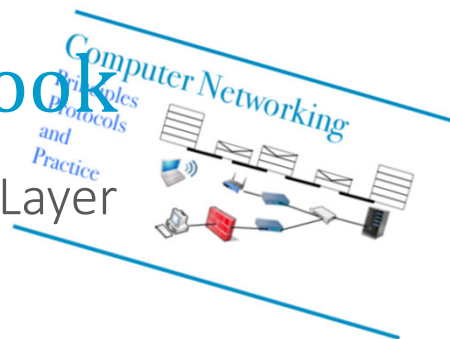


# Contents

1. The Application Layer
  2. QUIC
3. The Domain Name System
4. Application Layer Gateways
  5. IPv4 / IPv6
  6. ALG46

## Textbook

Chapter 2: The Application Layer



# 1. The Application Layer

The application layer of TCP/IP consists of  
the distributed applications themselves – it is the topic of courses such as information systems, distributed systems, industrial informatics  
a number of generic intermediate “layers”

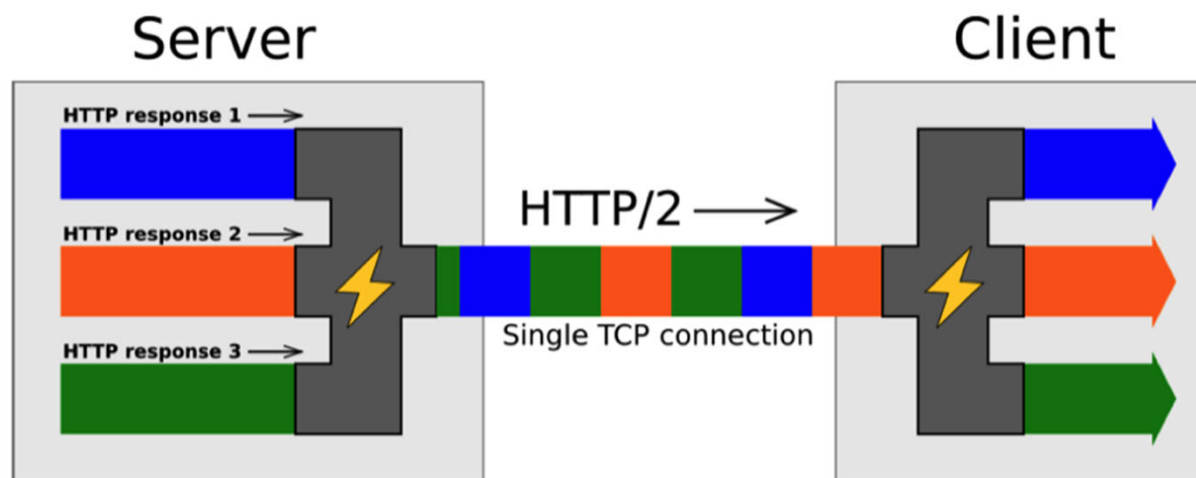
You did several application layers in the socket programming lab

In this module, we focus on generic issues, not on specific apps

## Example: World Wide Web (WWW)

HTTP/1.1 uses one TCP connection per object

HTTP/2 allows to use the same TCP connection for several objects with parallel processing of requests (at the application layer).



<https://blog.sessionstack.com/how-javascript-works-deep-dive-into-websockets-and-http-2-with-sse-how-to-pick-the-right-path-584e6b8e3bf7>

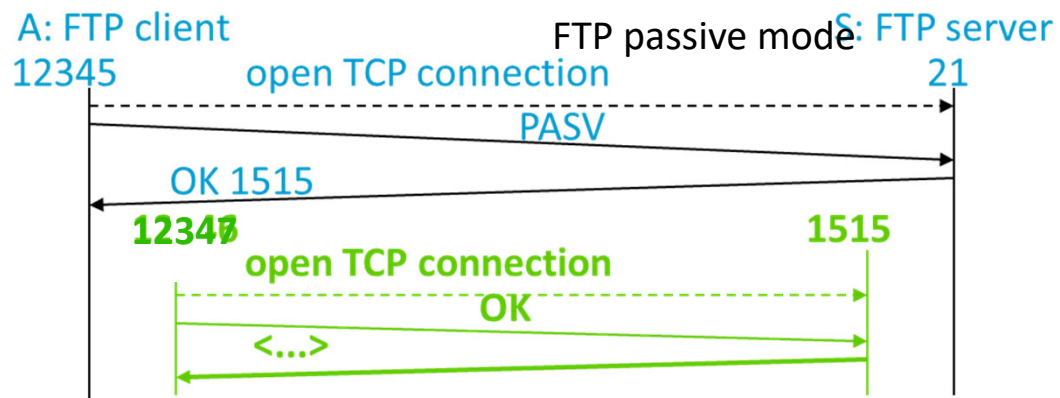
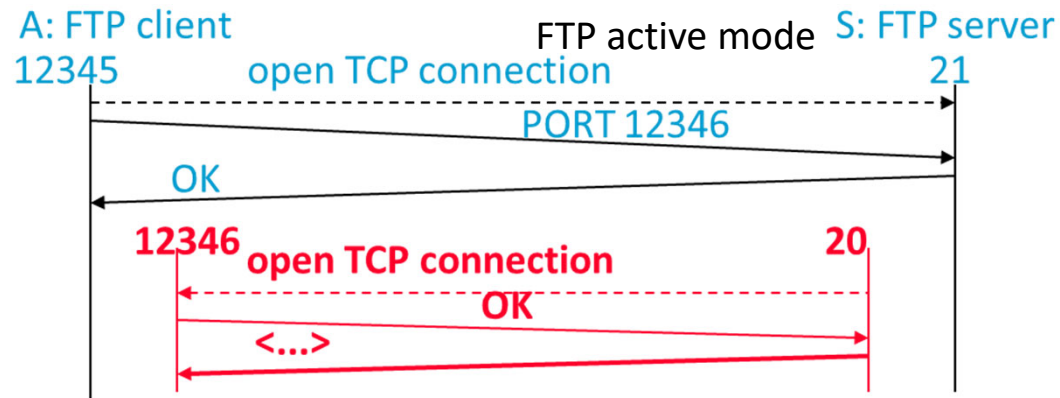
A server sends us a web page with 10 embedded objects (images, scripts, etc) using http/2 over one single TCP connection. Our browser displays the elements of the page as soon as they arrive, without waiting for the page to be complete.

One packet of the first object is lost (and will be retransmitted).

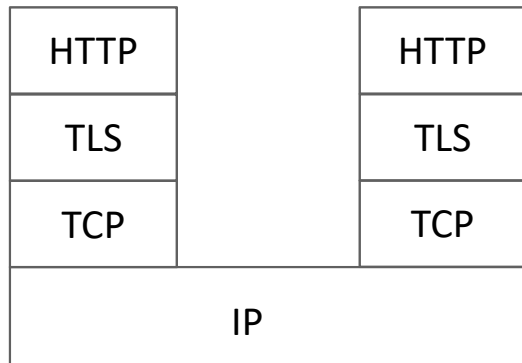
- A. The second object can be displayed before the lost packet is repaired because HTTP/2 multiplexes objects
- B. The second object must wait for the first object to be completely received
- C. It depends which version of TCP is used
- D. I don't know

## Which of these port numbers are TCP server ports ?

- A. 20
- B. 21
- C. 12346
- D. 20 and 21
- E. 20 and 12346
- F. 21 and 12346
- G. all of these three
- H. none of these three
- I. I don't know



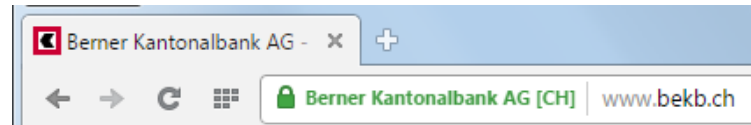
## 2. QUIC



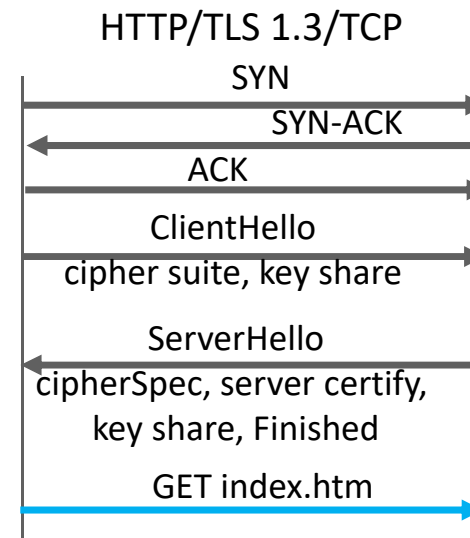
Recall that https means:

HTTP over  
secure transport (TLS)

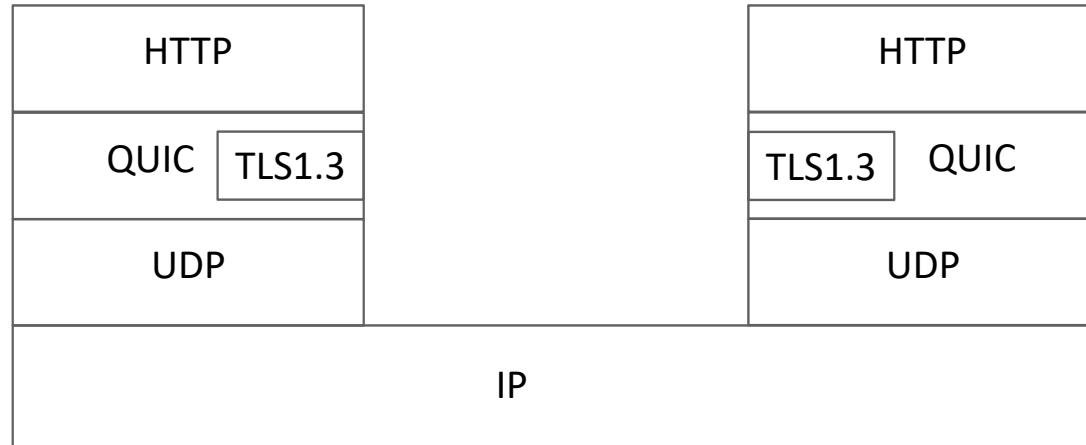
At least 2 RTTs before transmission of  
user data can start



http used over TLS and port 443 = https



# QUIC



## Why ?

1. Avoid latency and head-of-the line blocking of TLS over TCP
2. Modifications to TCP are often impractical because of filtering router that discard TCP options

## How ? All in one !

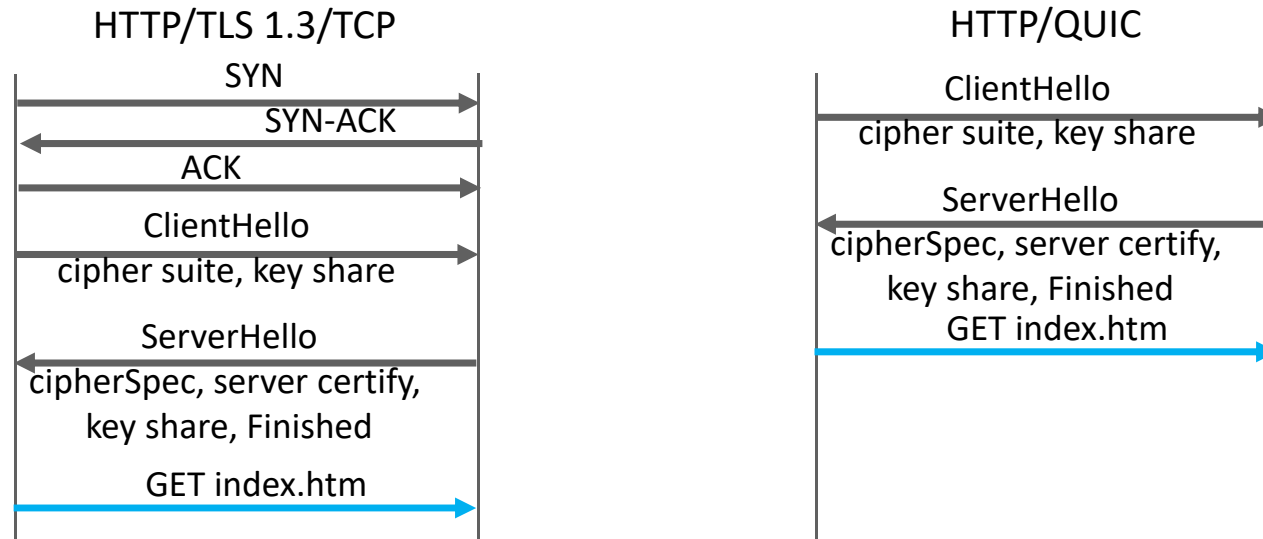
QUIC opens one TLS 1.3 session first

QUIC runs over UDP

Data is always secured (with TLS 1.3 by default).



# QUIC versus TCP with TLS1.3 (1-RTT case)



QUIC establishes a TLS1.3 session first.

QUIC assumes  $MTU \geq 1280$  B.

Implemented in google apps (chrome, youtube app, etc.) where it replaces TCP (try it: [wireshark](#) or [chrome://net-export/](#) and [https://netlog-viewer.appspot.com/](#))

# QUIC Packets, Streams and Frames

One QUIC session (also called “connection”) has multiples **streams**.

Every stream is reliable, like TCP, and data of this stream is delivered in sequence. But no sequence across different streams.

Streams can be created and deleted on-the-fly.

Stream 0 is for TLS1.3 and connection management.

One QUIC **packet** has a unique (increasing) packet number (64bits) – never wraps. A retransmitted packet has a different number.

A packet contains **frames** for stream data and other data.

|   |                      |                                   |                      |                 |
|---|----------------------|-----------------------------------|----------------------|-----------------|
| QUIC hdr: Connection Id,<br>Packet number |                      | stream 0<br>crypto handshake data |                      | ACK             |
| stream 1<br>app data                      | stream 2<br>app data | stream 3<br>app data              | stream 4<br>app data | MAX_STREAM_DATA |

# Reliable Streams

QUIC obtains **reliability similar to TCP**: missing packets are detected and retransmitted.

Packet loss detection incorporates the best known algorithms of TCP such as Fast Retransmit, RACK, Tail Loss Probe etc.

ACK frames contains up to 256 ack blocks (compare to TCP: 3 blocks).

**Flow control** is both per stream and per connection, using an explicit offset (instead of window).

**Congestion control** is similar to TCP Reno or Cubic, per connection (not per stream).

**Unreliable streams** are also foreseen; they do not have packet retransmission (in principle) but are subject to flow control and to congestion control.

## Example: Ephemeral-QUIC

PMU streaming application sends one measurement in one packet every 20 msec. If one packet is lost but not recovered before 20msec, it is better to forget about it and use the newer measurement. With TCP we have head-of-the line blocking, a new measurement is delayed if previous one is lost.

Weiyu Zhang's solution with QUIC [Master thesis 2018]

- send one measurement in one reliable stream
- one stream per measurement
- kill stream if a new measurement is available and previous stream is not acknowledged

Lisa, behind a NAT, uses the youtube app. Will the app be able to use QUIC ?

- A. No because the NAT will not see TCP port numbers
- B. No because QUIC uses UDP and the UDP port numbers are encrypted
- C. Yes because QUIC uses UDP and the NAT can manipulate UDP port numbers
- D. I don't know

## 3.The Domain Name System (DNS)

*Why* invented ?

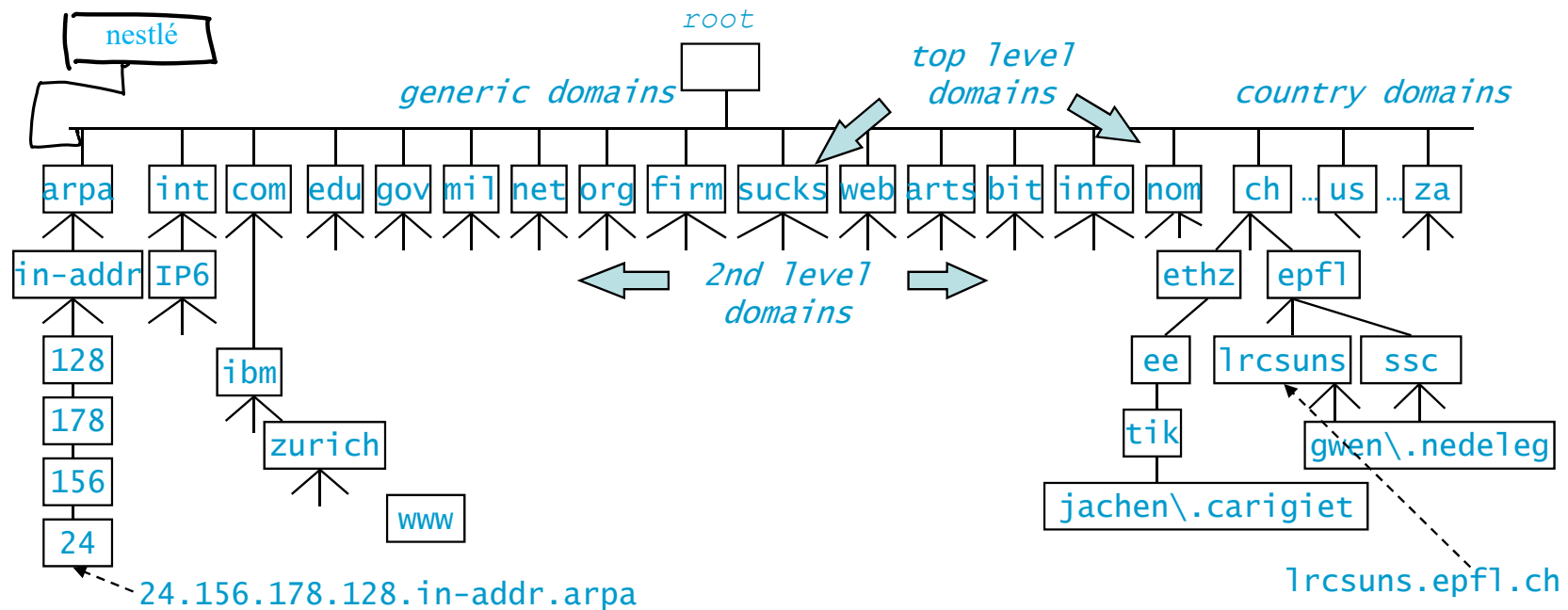
support user friendly naming of resources: computers,  
printers, mailboxes,...

hide IP address changes on servers

*What* does it do ?

map DNS names (ex: ssc.epfl.ch) to IP addresses

# DNS Names



every node on the tree represents one or a set of resources

every node on the tree has a label `1rcsuns` and a domain name `1rcsuns.epfl.ch`

label is made of 1 to 63 characters **a-z**, **0-9** or **-**

Other characters are allowed but real name is translated with *punycode*

Ex: `www.öpfl.ch` is in fact `www.xn--pfl-rna.ch`

# How Does DNS Work ?

When Lisa's machine needs to map name to IP address

DNS resolver in Lisa's machine contacts a DNS server

IP address of DNS server is known to Lisa's machine at configuration time

DNS server may not know answer: in such a case, DNS server needs to do several iterations, as shown on next example.

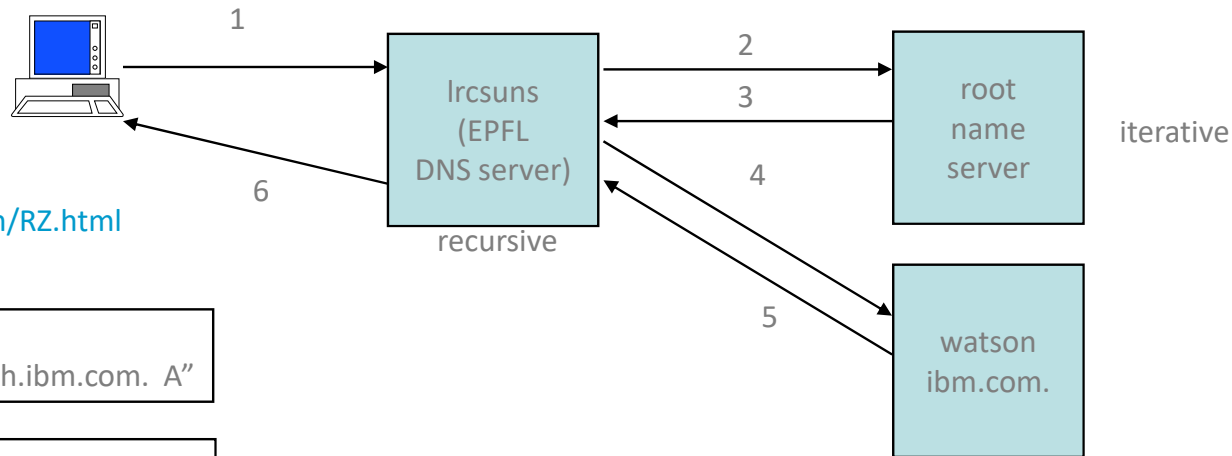
A cache is used at DNS resolver and at DNS server to avoid repeating the same requests frequently.

DNS uses UDP for queries and responses (in principle) but can also use TCP if server accepts





Lisa clicks:  
<http://www.zurich.ibm.com/RZ.html>



1 query, RD=yes  
question = "www.zurich.ibm.com. A"

2,4 query, RD=no  
question = "www.zurich.ibm.com. A"

3 answer  
question = "www.zurich.ibm.com. A"  
answer = ""  
authority= "ibm.com. NS watson.ibm.com.  
          NS ns.austin.ibm.com.  
          NS ns.almaden.ibm.com."  
additional="watson.ibm.com. A 192.35.232.34  
          ns.austin.ibm.com. A 129.34.139.4  
          ns.almaden.ibm.com A 198.4.83.134"

5,6 answer  
question = "www.zurich.ibm.com. A"  
answer = "www.zurich.ibm.com. 7200 A 193.5.61.131"

The previous slide shows an example of name resolution.

1. an application on lrcsuns requests a name resolution (find the IP address of www.zurich.ibm.com), a request is sent to the name server configured at lrcsuns
2. the epfl name server does not know the answer, but, as any name server, knows the IP address of root name servers.
3. a root name server knows the IP addresses of all name servers of all top level domains (such as .com, .info etc) and also of many level-2 domains. Thus, it informs lrcsuns of the IP address of the name servers responsible for the ibm.com domain
4. the epfl name server sends the same request now to the ibm name server
- 5 the ibm name server gives the IP address of www.zurich.ibm.com back to the epfl name server. The epfl name server keeps the address in its cache, this will be used if the same request comes again
- 6 the epfl name server gives the IP address of www.zurich.ibm.com back to lrcsuns. End of the resolution !

The request sent by lrcsuns is *recursive* (RD=yes): lrcsuns will receive only the final answer. In contrast, the request sent by the epfl name server is *iterative* (RD=no): it receives only partial answers that help towards the solution.

# A (=IPv4) and AAAA (= IPv6) records

```
C:\Users\leboudec> dig a lca.epfl.ch
```

```
<<>> DiG 9.3.2 <<>> a lca.epfl.ch
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 652
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;lca.epfl.ch.                IN      A

;; ANSWER SECTION:
lca.epfl.ch.      86400   IN      CNAME   lca1srv2.epfl.ch.
lca1srv2.epfl.ch.86400   IN      A       128.178.156.24
```

```
C:\Users\leboudec> dig aaaa lca.epfl.ch
```

```
; <<>> DiG 9.3.2 <<>> aaaa lca.epfl.ch
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 415
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;lca.epfl.ch.                IN      AAAA

;; ANSWER SECTION:
lca.epfl.ch.      86400   IN      CNAME   lca1srv2.epfl.ch.
lca1srv2.epfl.ch.86400   IN      AAAA    2001:620:618:19c:1:80b2:9c18:1
```

A and AAAA records map name to IP address

CNAME record maps name to name

```
lca.epfl.ch.      86400   IN       CNAME    lca1srv2.epfl.ch.
```

here lca1srv2.epfl.ch is the canonical name, lca.epfl.ch is an alias

The DNS uses many other types of records. E.g.

MX: mail server

```
epfl.ch.          86400   IN       MX       50 mx1.epfl.ch.epfl.ch.      86400
```

SRV: generic service

```
_sip._tcp.epfl.ch.  86400   IN       SRV      10 10 5060 uc-edge-ext1.epfl.ch.
```

```
_h323ls._udp.epfl.ch. 86400   IN       SRV      10 10 1719 uc-edge-ext1.epfl.ch.
```

```
_sip._tls.epfl.ch.  86400   IN       SRV      10 10 5061 uc-edge-ext1.epfl.ch.
```

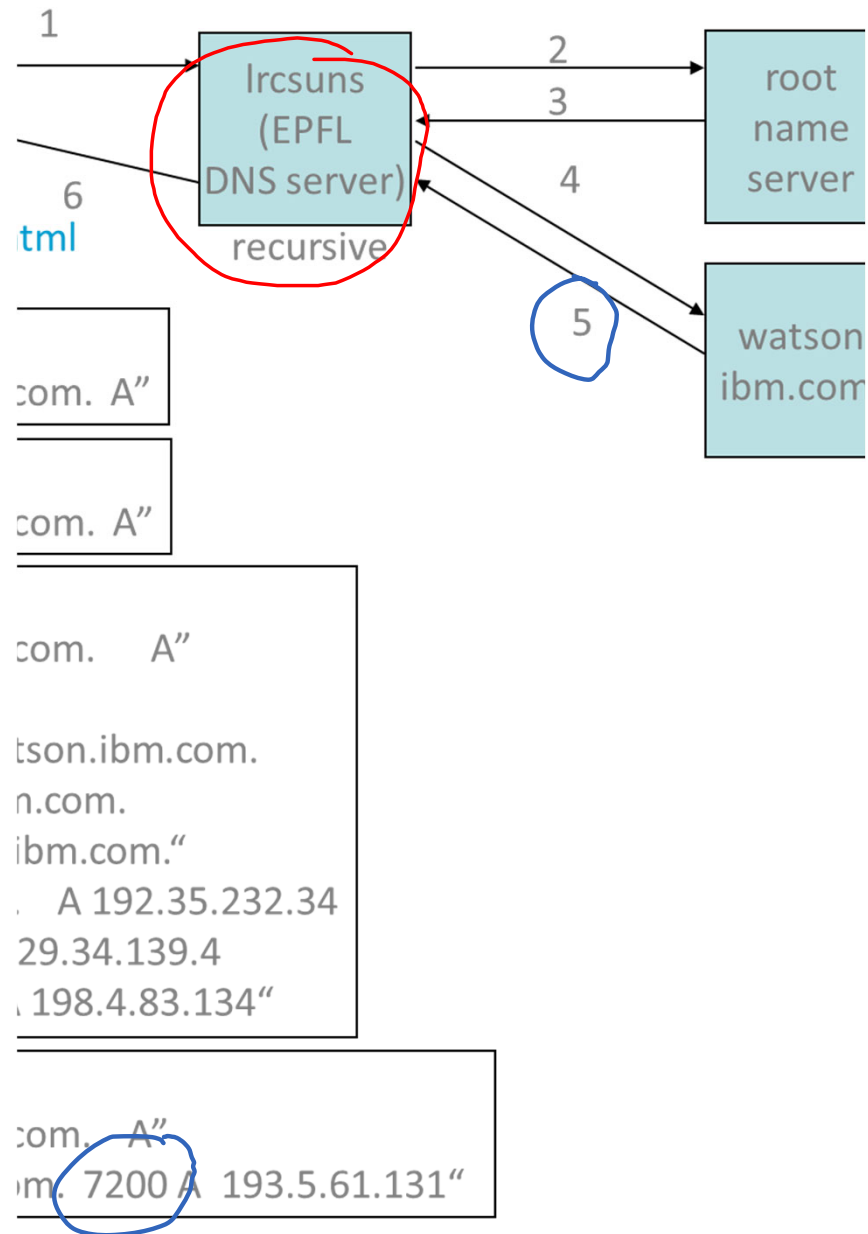
(here used for IP telephony)

# Caching and Time to Live

when receiving message 5, [stisun1.epf.ch](#) keeps answer from [watson.ibm.com](#) in a *cache* and does  $TTL = 7200$  s for this cache entry. TTL decreases by 1 every second. When  $TTL = 0$ , cache entry is discarded by [lrcsuns](#)

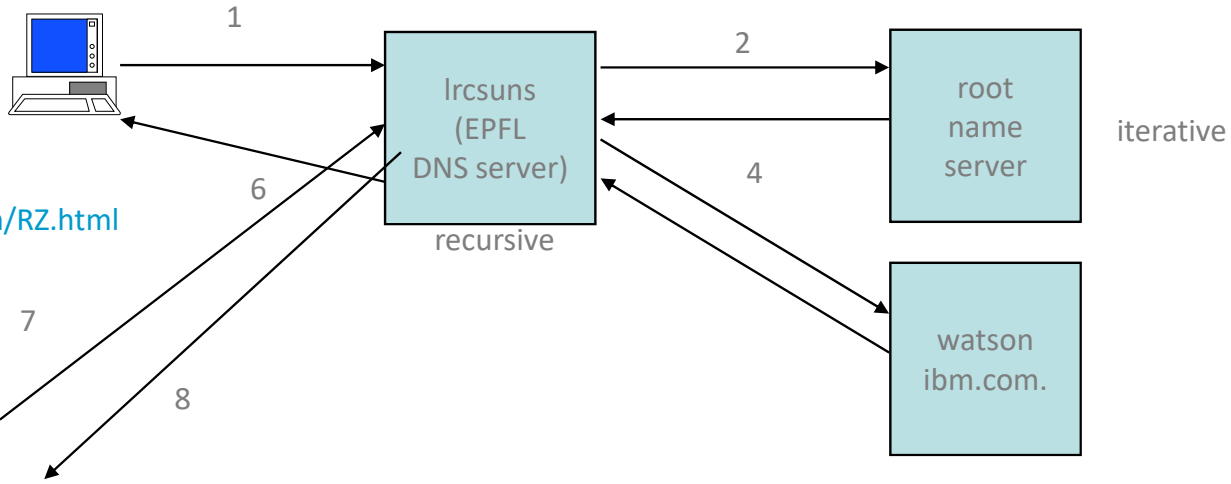
During next 2 hours, further requests are answered directly by [lrcsuns](#)

[watson.ibm.com](#) is the “authoritative” DNS server for this record (i.e. the origin) -- decides value of *TTL* to be used by caching servers





Lisa clicks:  
<http://www.zurich.ibm.com/RZ.html>



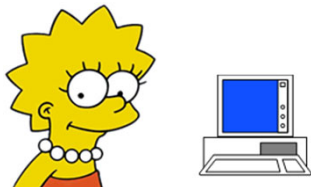
1mn later, Bart clicks:  
<http://www.zurich.ibm.com/RZ.html>

7 query, RD=yes  
question = "www.zurich.ibm.com. A"

8 answer  
question = "www.zurich.ibm.com. A"  
answer = "www.zurich.ibm.com. 7140 A 193.5.61.131"

# Reverse DNS Lookup

Reverse Query = find name, given some IP address; used e.g. by traceroute



Lab 1a

Tracing route to read.more.at.beaglenetworks.net [216.81.59.173]  
over a maximum of 64 hops:

```
 1      2 ms      <1 ms      <1 ms      cv-ic-dit-v151-ro.epfl.ch [128.178
 2      <1 ms     <1 ms     <1 ms     cv-gigado-v100.epfl.ch [128.178.10
 3      <1 ms     <1 ms     <1 ms     c6-ext-v200.epfl.ch [128.178.200.1
 4      <1 ms     <1 ms     <1 ms     swiel2.epfl.ch [192.33.209.33]
 5      18 ms     <1 ms     3 ms     swiLS2-10GE-1-2.switch.ch [130.59.
 6      4 ms      4 ms      4 ms     swiE71-10GE-2-7.switch.ch [130.59
```



## How does traceroute do DNS lookup ?

```
Tracing route to read.more.at.beaglenetworks.net [216.81.59.173]  
over a maximum of 64 hops:
```

```
 1      2 ms      <1 ms      <1 ms      cv-ic-dit-v151-ro.epfl.ch [128.178.100.100]  
 2      <1 ms     <1 ms     <1 ms     cv-gigado-v100.epfl.ch [128.178.100.100]  
 3      <1 ms     <1 ms     <1 ms     cv-gigado-v100.epfl.ch [128.178.100.100]
```

- A. DNS client tries all possible DNS names and verifies each of them with DNS server
- B. DNS client asks DNS server: *what is the name corresponding to this IP address ?* and DNS server searches its database to find an answer
- C. Traceroute asks intermediate routers: *what is your DNS name ?*
- D. None of the above
- E. I don't know



```
Z:\>dig 1.15.178.128.in-addr.arpa PTR
```

we are looking for the name corresponding to the IPv4 address 128.178.151.1

```
; <<>> DiG 9.3.2 <<>> 1.15.178.128.in-addr.arpa PTR
;; global options: printcmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 368
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
```

```
;; QUESTION SECTION:
;1.15.178.128.in-addr.arpa.  IN  PTR
```

```
;; ANSWER SECTION:
1.15.178.128.in-addr.arpa. 86400 IN  PTR  c6-slb-1-v15-ro.epfl.ch.
```

the name is found from a PTR record in DNS

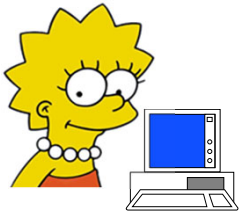
```
;; Query time: 0 msec
;; SERVER: 128.178.15.227#53(128.178.15.227)
;; WHEN: Thu Aug 28 12:10:55 2014
;; MSG SIZE rcvd: 80
```

```
Z:\>dig c6-slb-1-v15-ro.epfl.ch
```

we verify that the IPv4 address for this name is as expected

```
; <<>> DiG 9.3.2 <<>> c6-slb-1-v15-ro.epfl.ch
;; ANSWER SECTION:
c6-slb-1-v15-ro.epfl.ch. 86400 IN  A  128.178.15.1
```

it is as expected, i.e. reverse record (PTR) and direct record (A) are consistent



```

37  149 ms  149 ms  149 ms  freedom.to.the.galaxy [206.214.251.89]
38  149 ms  150 ms  149 ms  0-----I-----I-----0 [206.214.251.94]
39  150 ms  151 ms  150 ms  0-----0 [206.214.251.97]
40  151 ms  150 ms  150 ms  0-----0 [206.214.251.102]

```

Z:\>dig 89.251.214.206.in-addr.arpa PTR

we are looking for the name corresponding to the IPv4 address 206.214.251.89

```

;<<>> DiG 9.3.2 <<>> 89.251.214.206.in-addr.arpa PTR
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1223
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

```

the name is found from a PTR record in DNS

```

;; QUESTION SECTION:
;89.251.214.206.in-addr.arpa. IN PTR

```

```

;; ANSWER SECTION:
89.251.214.206.in-addr.arpa. 257 IN PTR freedom.to.the.galaxy.

```

```

;; Query time: 2 msec
;; SERVER: 128.178.15.227#53(128.178.15.227)
;; WHEN: Thu Aug 28 11:43:34 2014
;; MSG SIZE rcvd: 80

```

we verify that the IPv4 address for this name is as expected

```

Z:\>ping freedom.to.the.galaxy.
Ping request could not find host freedom.to.the.galaxy.
Please check the name and try again.

```

this DNS name is bogus, there is no A (nor AAAA) record for it. The PTR record is bogus.

# DNS authority

DNS **authority** is hierarchical

Top level authority allocates top-level domains (e.g. .ch)

The top level is currently run by PTI (Public Technical Identifiers – formerly ICANN)

The 13 top-level DNS servers (root servers) are run by multiples companies

National **registry** allocates .ch domain names (in Switzerland: SWITCH)

**Registar** is a commercial organization accredited by registry to sell names (e.g. infomaniak.ch, register.ch, switchplus.ch)

DNS servers are run by ISPs or other organizations.

**namecoin** is an attempt to provide peer-to-peer (= without authority) DNS names for the domain .bit. It uses the blockchain technology.

# DNS and Load Balancing

## Why ?

sovkom.com is very popular and hosted on many servers. Traffic should be split between sites

## How ?

One solution is to use a load balancing DNS server:

sovkom.com is mapped to many different addresses (or CNAMEs)

load balancing DNS server  
chooses one address (based  
on load, response time...)

TTL is very short to allow for  
frequent changes

```
sovkom.com 300 AAAA 2001: babe::b0b2  
sovkom.com 300 AAAA 2001: b88b::b0bc  
sovkom.com 300 AAAA 2001: b55b::b0ba  
...
```

# DNSSEC

DNS alone is unsecure even if servers are trusted  
anyone can send (incorrect) answers to DNS queries

DNSSEC solves this problem

- Records are signed, using public key cryptography

- Chain of trust is initialized with root servers

- Some domains (such as .se) implement it systematically

- Some DNS servers (Google's 8.8.8.8 and 8.8.4.4) implement DNSSEC systematically

Alternative: DNS over TLS / over QUIC

# Zeroconf / Bonjour Server-less DNS

Why invented ?

For names of local importance only : e.g. your printer at home

hp-laser2719.local

top level domain **local** is reserved for names visible in subnet

Uses the zeroconf multicast address **224.0.0.251/ff02::fb**, UDP port **5353**

Works only in one (possibly bridged) LAN

Allows not only to resolve local names but also to discover devices

Universal Plug and Play (UPnP) also does similar things.

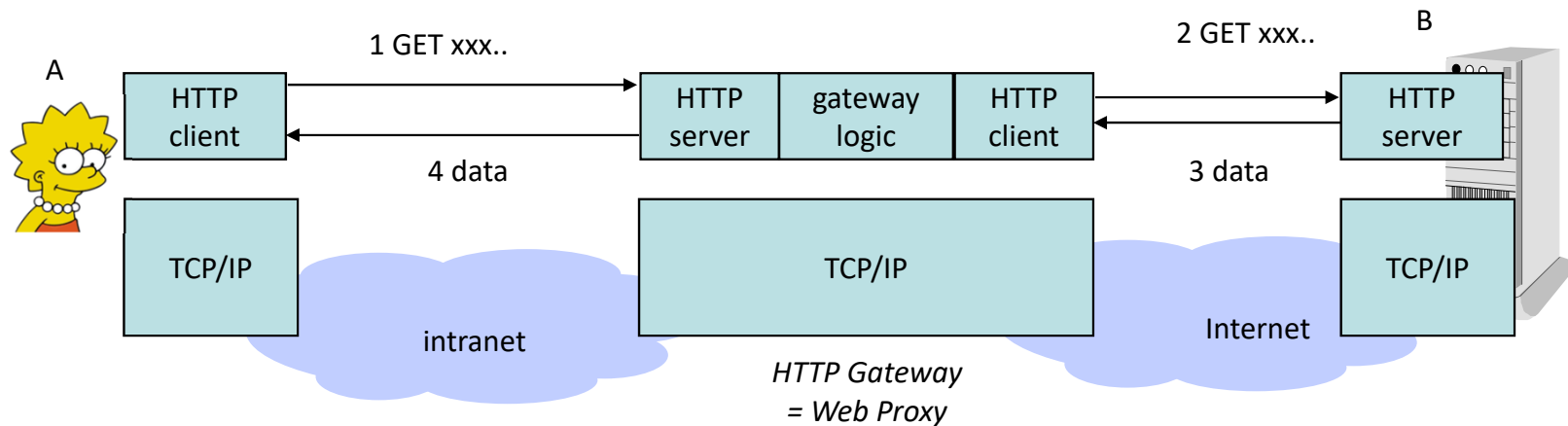
We change the IPv4 address of lca.epfl.ch.  
How long does it take for the whole world to be informed of the change ?

- A. less than a minute
- B. Up to 6 hours
- C. Up to 1 day
- D. Up to 2 days
- E. None of these
- F. I don't know

```
C:\Users\leboudec> dig a lca.epfl.ch
<<>> DiG 9.3.2 <<>> a lca.epfl.ch
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, flags: qr aa rd ra; QUE
;; QUESTION SECTION:
;lca.epfl.ch.
;; ANSWER SECTION:
lca.epfl.ch.      86400
lca1srv2.epfl.ch. 86400
```

EPFL names have TTL= 86400  
secs, i.e. 1 day

## 4. Application Layer Gateways (ALGs)



Definition: an *application layer gateway* is an application layer intermediate system. It terminates the TCP connections (if the application layer uses TCP) and does “store and forward” for the application layer data – it is another example of “middle box”

*Example:* HTTP gateway, also called “web proxy”

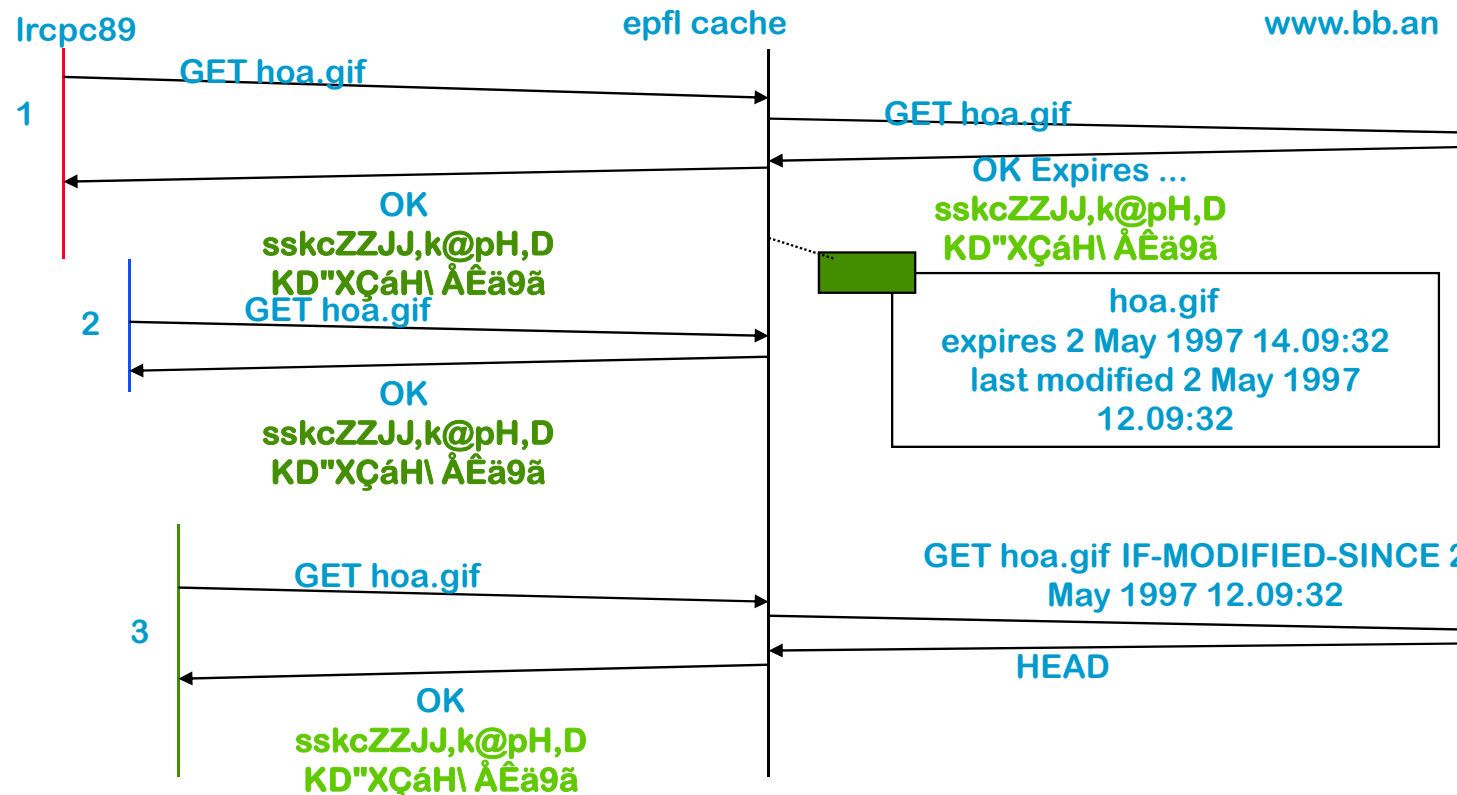
A sends HTTP request to gateway, gateway sends another HTTP request to server, server sends objects to gateway, gateway sends objects to A

Gateway terminates the TCP connections and does “store and forward”



# Web Proxy can be deployed to reduce traffic

Web proxy can keep frequently asked documents close to user  
requested files are kept in a cache  
similar systems deployed in content distribution networks

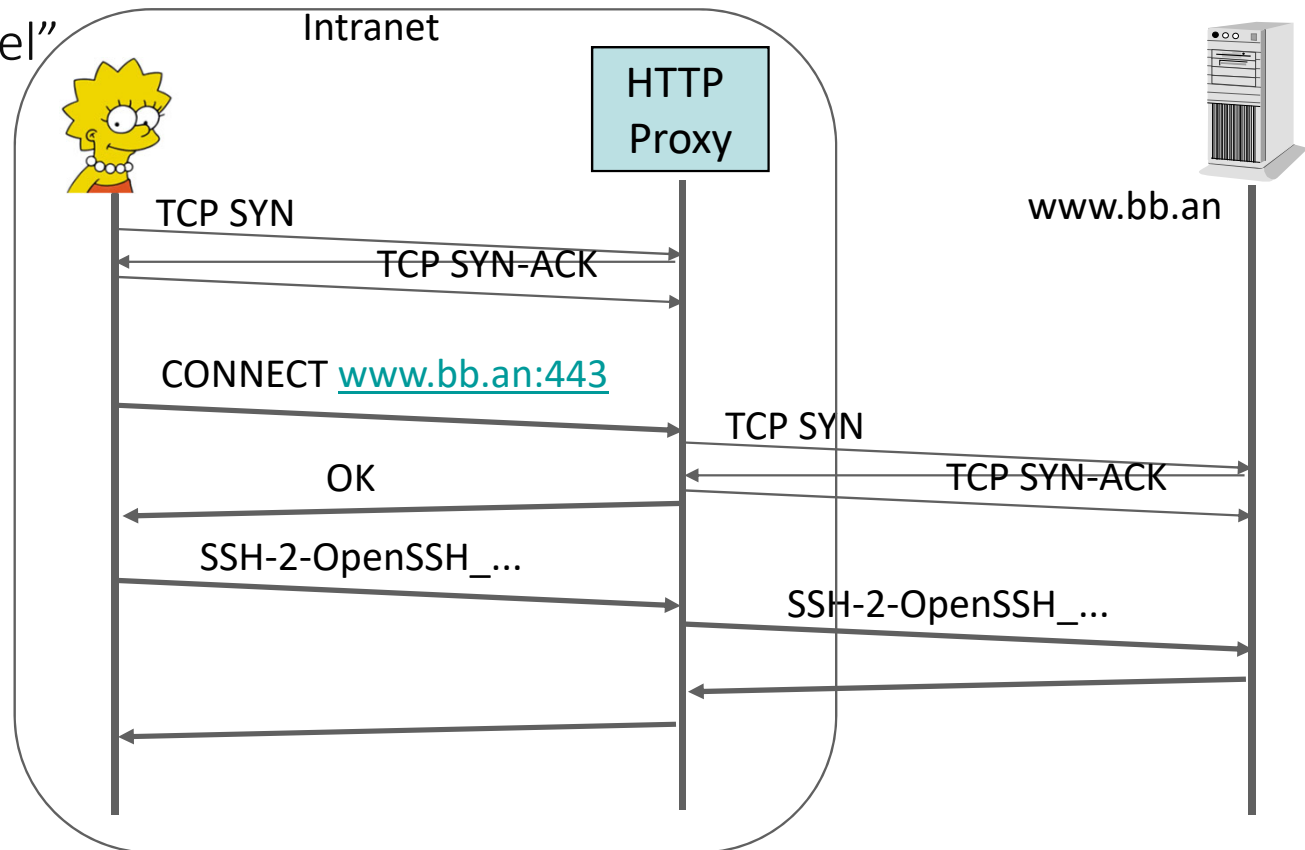


# HTTP Tunnel

Web proxy opens a TCP connection to remote server on behalf of client;  
After that, contents of TCP segments are relayed unmodified  
supports end-to-end TLS.

This is called an “HTTP Tunnel”

There are two TCP connections, one client-proxy and one proxy-remote server.



# The “End-to-end” Principle

The “end-to-end” principle of the Internet says that any layer above the network layer should avoid intermediate systems

Thus: Application Layer Gateways should be avoided

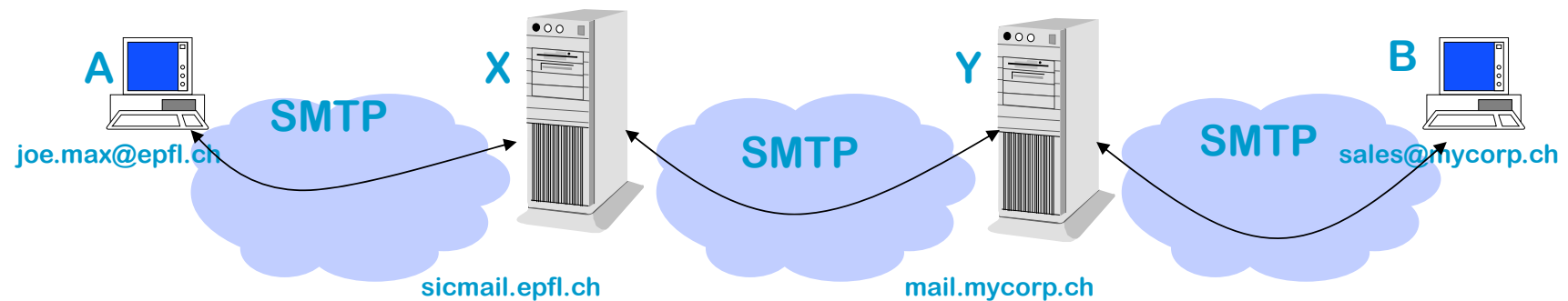
*Why?*

Simplify the network. The network is independent of applications and can be run more safely.

Allow easy deployment of applications. Ex: the web was deployed in 1994 in a few months. Before that, TCP/IP existed, but not HTTP.

Performance is better – no store and forward

# The “End-to-end” Principle for Email



Q. what would a strict application of the end-to-end principle on the figure give ?

# The End-to-end Principle is not always Applicable

Application layer gateways are still desirable in some cases.

For Example

1. **Mobility** (or partial connectivity). On the previous figure, this is why we send email to Y and not to B.
2. **Security / Access Control**. X connects to the internet through web proxy S. S can deny access to some web sites.



**Attention!**

L'accès à cette page a été bloqué pour des raisons de sécurité.

IP: 10.4.147.134

Application: slideshare-base

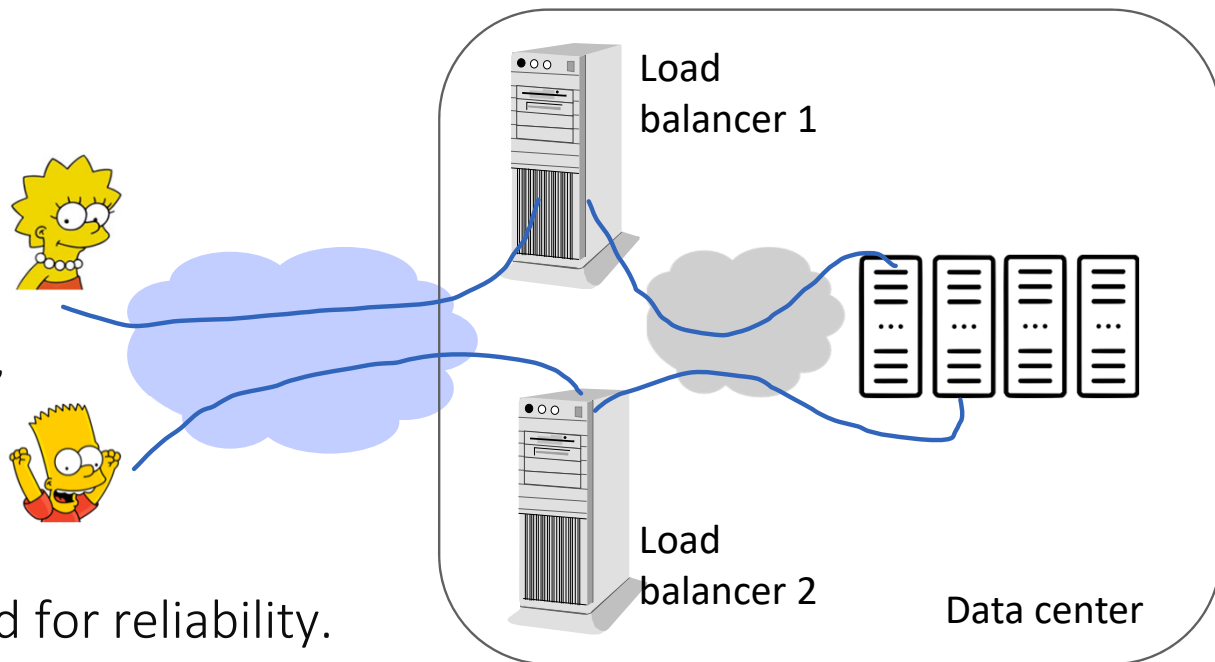
3. **Interworking**. Interworking between IPv4 and IPv6.
4. **Performance**. Counter TCP deficiencies on wireless links, Load balancers

# Server-Side Load Balancers (reverse proxies)

Server-side Load balancers are application-layer gateways: they terminate TCP or QUIC connections and re-direct user request to one of the application servers.

Also called “reverse-proxies” (server side).

Load balancers are replicated for reliability.  
How is user traffic split among load balancers ?

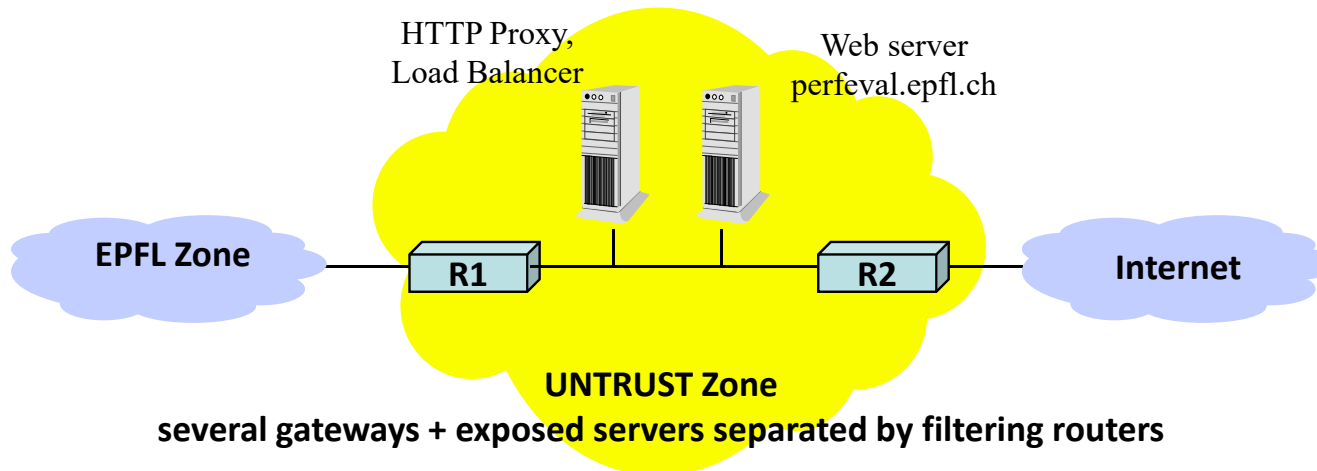


# Firewalls use ALGs

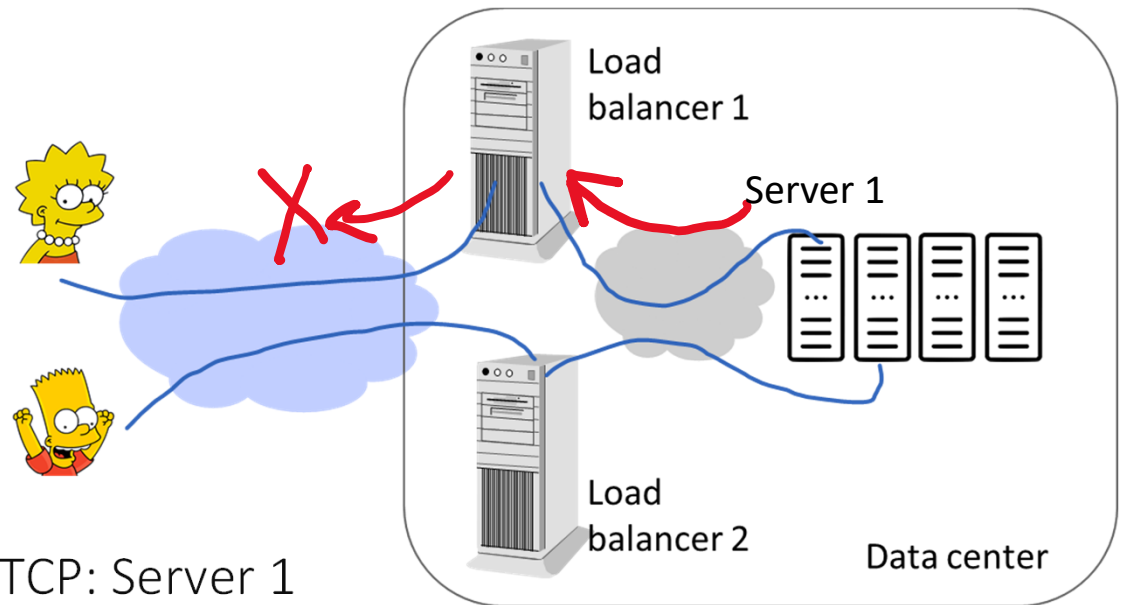
Firewall = a system that separates Internet from intranet: all traffic must go through firewall; only authorized traffic may go through; firewall itself cannot be penetrated (as one thinks)



**Firewall =  
one dual homed application layer gateway**



A packet is lost in the transfer from data center to Lisa. Which system will retransmit it ?



- A. Load balancer 1
- B. With QUIC: Load Balancer 1; with TCP: Server 1
- C. With TCP: Load Balancer 1; with QUIC: Server 1
- D. Server 1
- E. I don't know

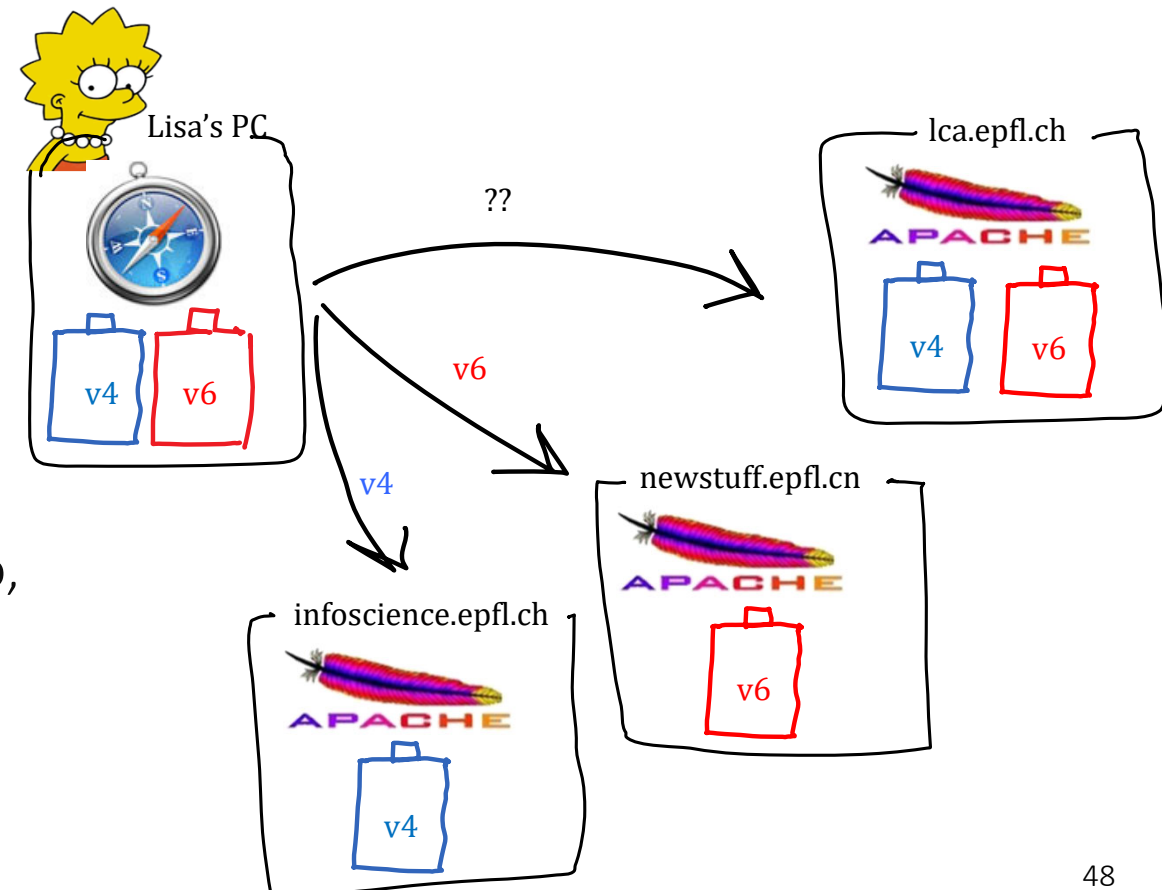


## 5. Side Effects of IPv6 vs IPv4

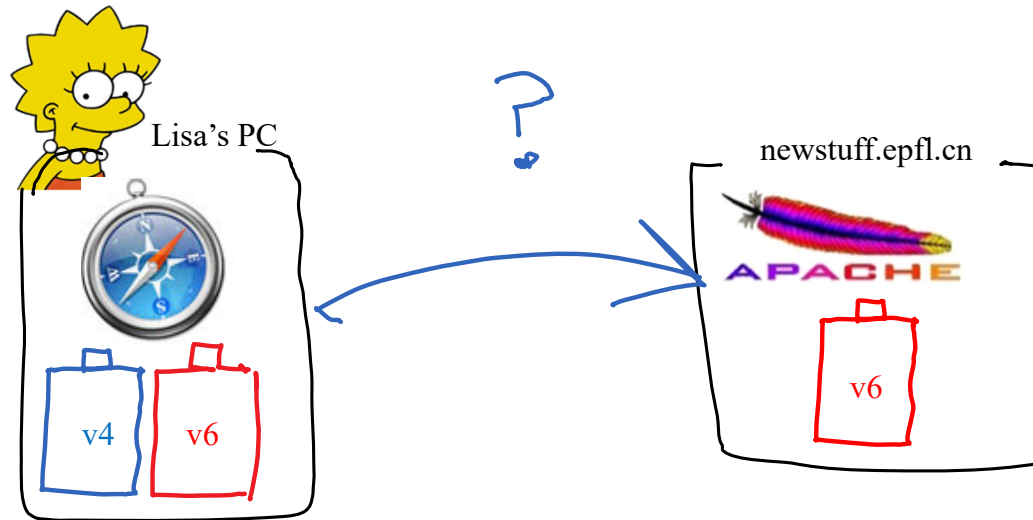
Application layer *protocols* (such as HTTP) are independent of the version of IP (IPv4/IPv6). However, an application layer program is dependent on the type of socket it uses (IPv4, IPv6, dual stack socket on some OSs). Recent code *should* work with both IPv6 and IPv4.

Will Lisa use IPv4 or IPv6 to connect to lca.epfl.ch ?

Many systems by default prefer IPv6. If so, Lisa will connect via IPv6.

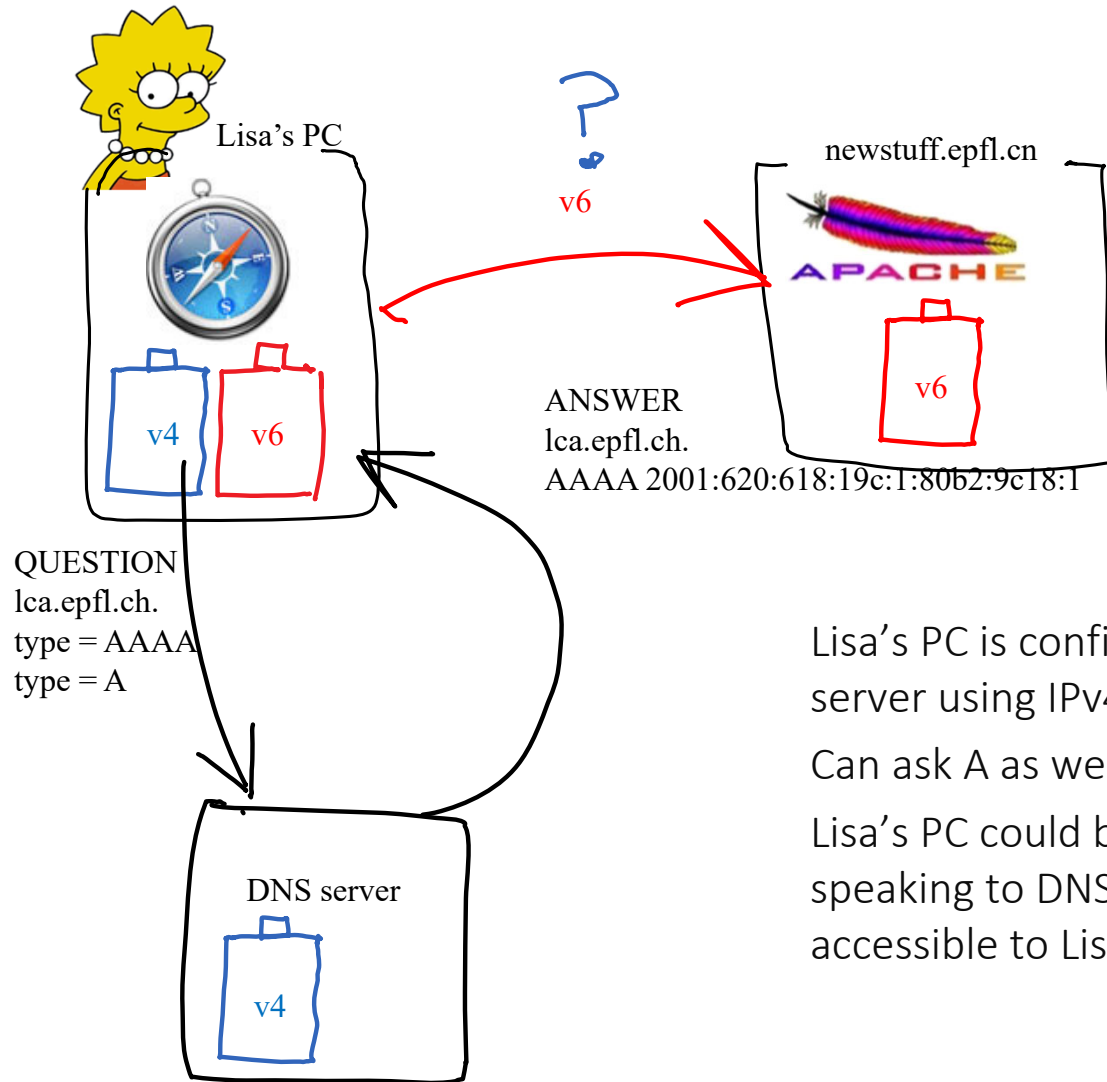


## How does Lisa know whether other end is v4 or v6 ?



- A. Lisa's PC sends a ping6 and a ping4 to newstuff.epfl.cn
- B. Lisa's PC tries http using both IPv4 and IPv6 in parallel and sees what works
- C. None of the above
- D. I don't know

# DNS can be accessed via IPv4 or IPv6



Lisa's PC is configured to talk to a DNS server using IPv4

Can ask A as well as AAAA questions

Lisa's PC could be configured to use v6 for speaking to DNS server, if a v6 DNS server is accessible to Lisa's PC

```
PS C:\Users\leboudec> nslookup
```

```
Default Server: stisun1.epfl.ch
```

```
Address: 2001:620:618:10f:1:80b2:f08:1
```

```
> infoscience.epfl.ch
```

```
Server: stisun1.epfl.ch
```

```
Address: 2001:620:618:10f:1:80b2:f08:1
```

```
Name: ec2-34-250-186-131.eu-west-1.compute.amazonaws.com
```

```
Address: 34.250.186.131
```

```
Aliases: infoscience.epfl.ch
```

```
> www.epfl.ch
```

```
Name: www.epfl.ch.cdn.cloudflare.net
```

```
Addresses: 2606:4700:10::6814:e42a
```

```
2606:4700:10::6814:e52a
```

```
104.20.228.42
```

```
104.20.229.42
```

```
Aliases: www.epfl.ch
```

## 6. Application Layer Gateway (ALG46) for IPv4 and IPv6 Interworking

IPv4 and IPv6 are  
incompatible

v4 only host cannot handle  
IPv6 packets

v6 only host cannot handle  
IPv4 packets



*What* needs to be solved:

Interworking (h4 to h6): allow IPv6-  
only hosts and IPv4-only hosts to  
communicate

example: IPv6 host connects to an  
IPv4 web server

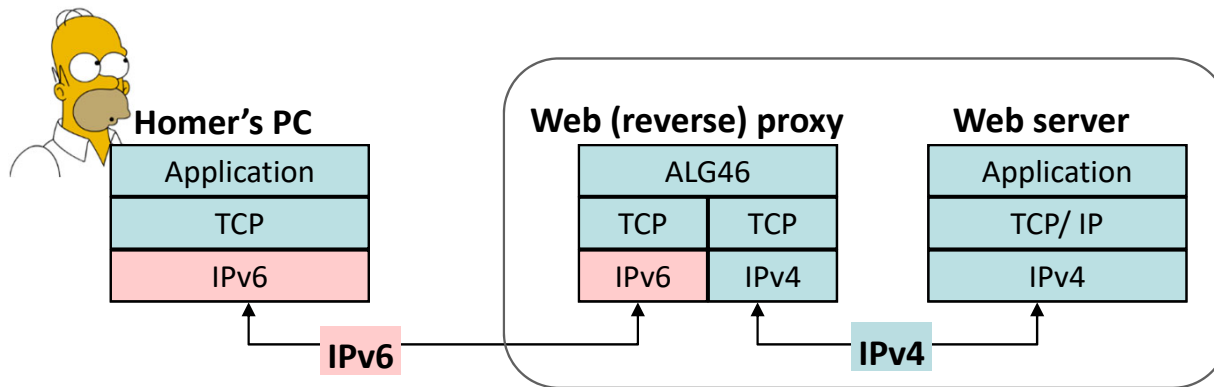
Like to like access

6 to 6 over IPv4 infrastructure; ex:  
IPv6 host at home connects to  
IPv6 server at EPFL

4 to 4 over IPv6 (legacy devices)

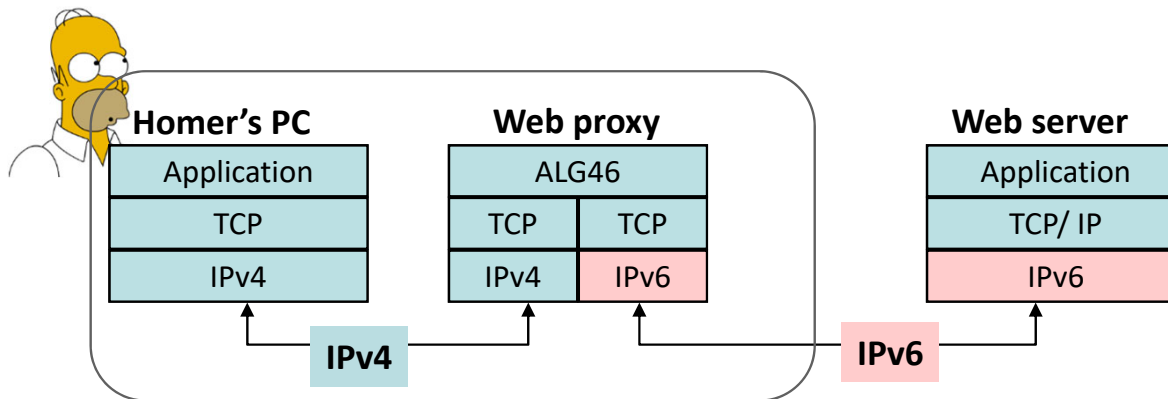
In this section we study interworking.

# Dual Stack Application Layer Gateways (ALG46s) for h4 to h6 Interworking

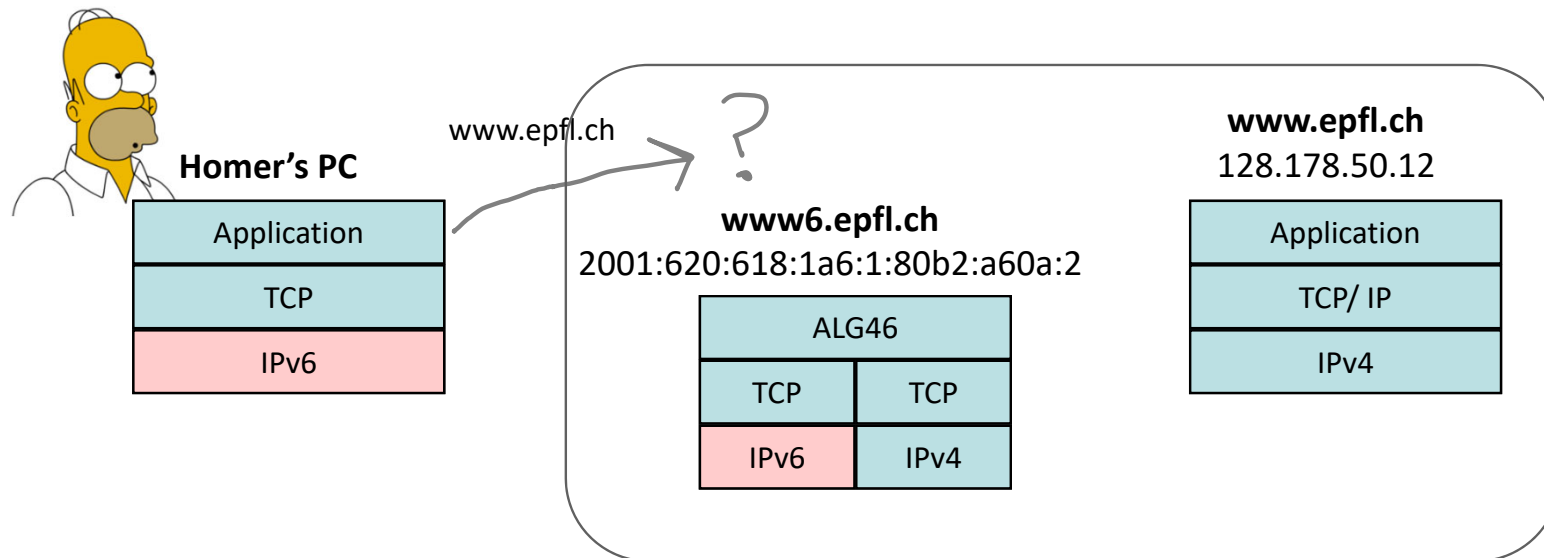


E.g. Enterprise does not support IPv6 on its legacy server. Same could happen with v4 instead of v6.

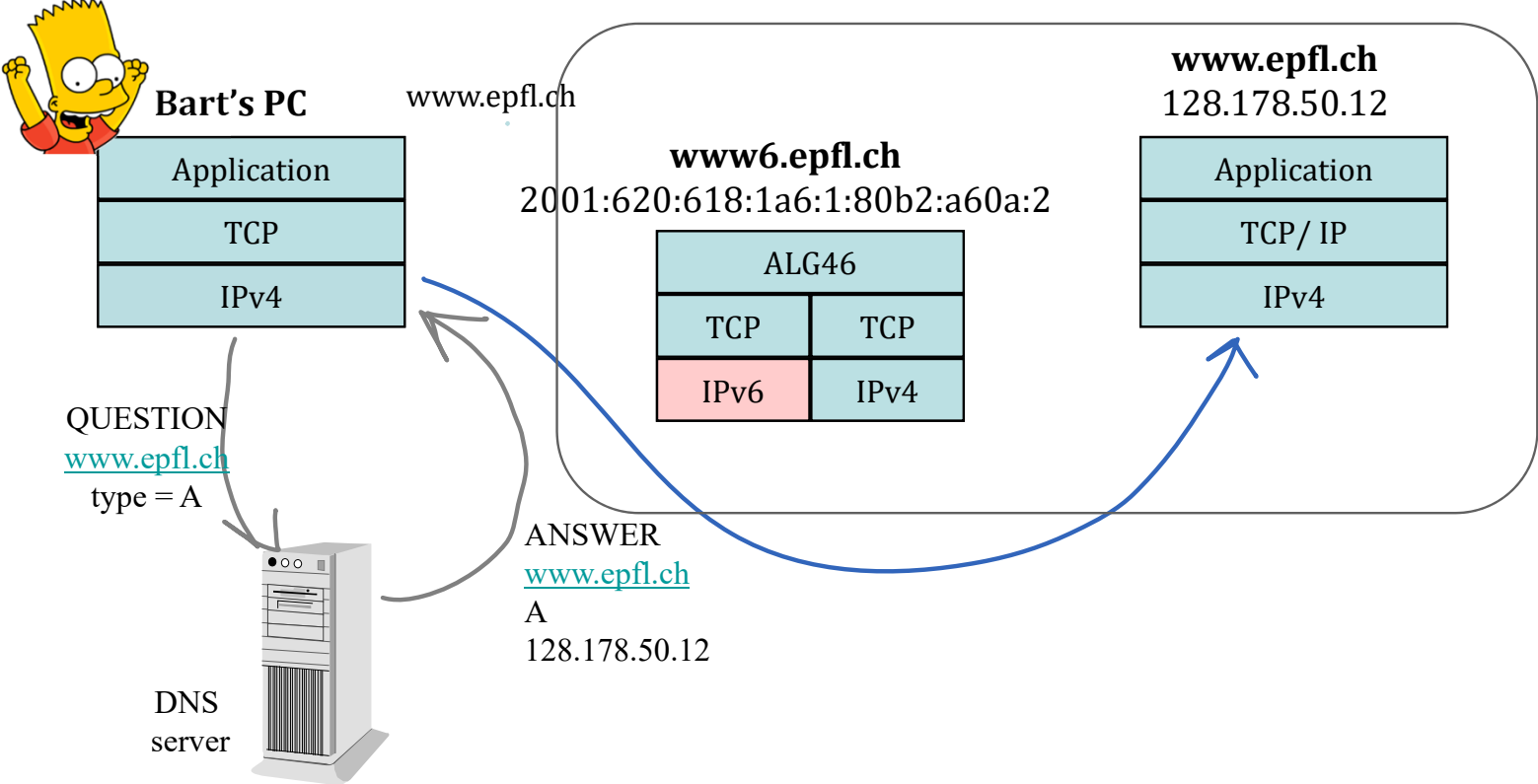
E.g. Homer's enterprise provides HTTP tunnels to give Homer access to v6 services without deploying IPv6 internally. Same could happen with v4 instead of v6.



# How does Homer's PC know it should go to the ALG instead of the final web server ?

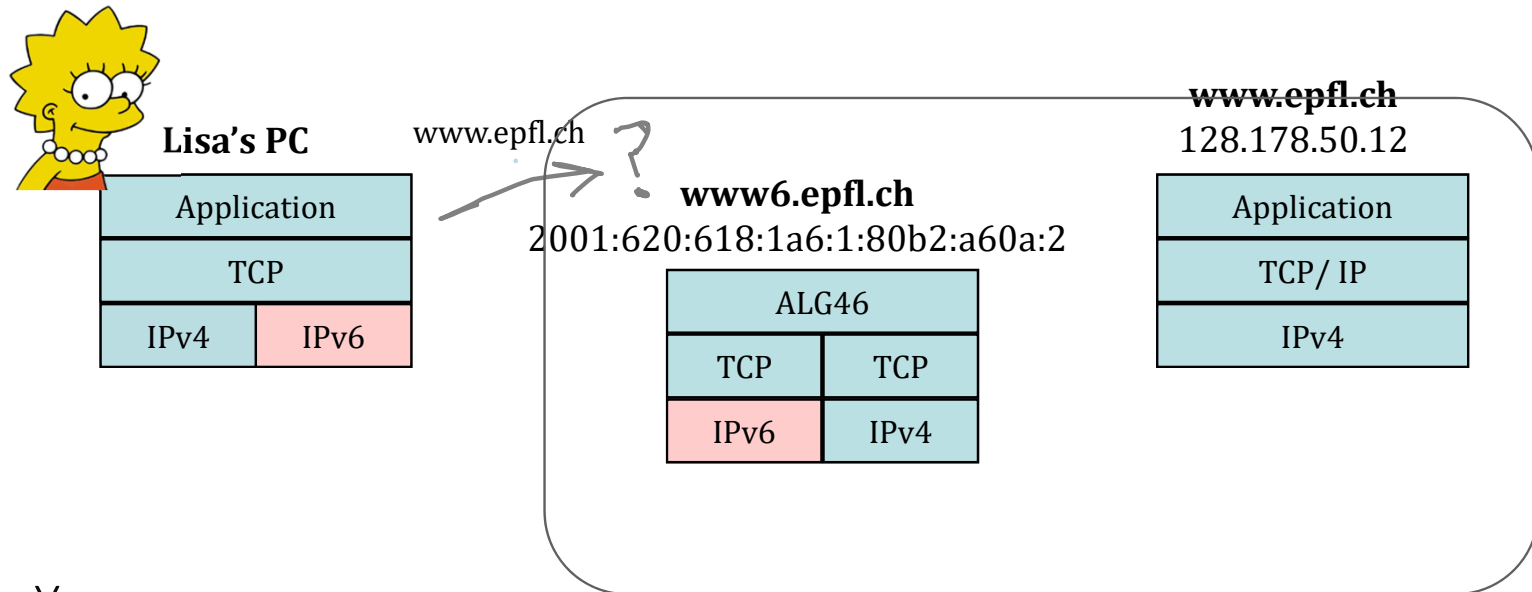


# Bart's PC will not use the ALG





# Will Lisa's PC use the ALG ?



- A. Yes
- B. No
- C. It depends on the configuration
- D. I don't know

# Facts to Remember

QUIC can be used as a replacement for TCP. Main differences are: TLS is integrated + multiple streams.

The end-to-end principle states that Application layer gateways (ALGs) should be avoided whenever possible. ALGs are still deployed for security / performance.

DNS is used by apps on dual stack machines to know whether to use IPv4 or IPv6.

ALG46s (web proxies, HTTP tunnels) can be used to solve the h4 to h6 interworking problem.