FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO
U. PORTO
U. PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Application Layer

Redes de Computadores

2021/22

Pedro Brandão

References

- These slides are from “Computer Networking: A Top Down Approach 5th edition. Jim Kurose, Keith Ross Addison-Wesley, April 2009”
  - With adaptations/additions by Manuel Ricardo and Pedro Brandão

RCom 21/22 - Application Layer

Driving questions…

- *What is the need for application protocols?*
- *What is underneath a web page request?*
- *Why the verboseness of the IETF protocols?*
- *How to we really get from a name to a network address?*

3

FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO

U.PORTO

U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

RCom 21/22 - Application Layer

Principles of network applications

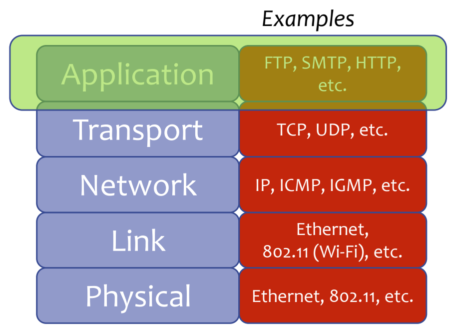
Network layer

RCom 21/22 - Application Layer

U.PORTO

Internet protocol stack

- Application: network processes
- Transport: data transfer between processes
- Network: packet routing between source and destination
- Link: data transfer between adjacent network elements
- Physical: bits on the "wire"

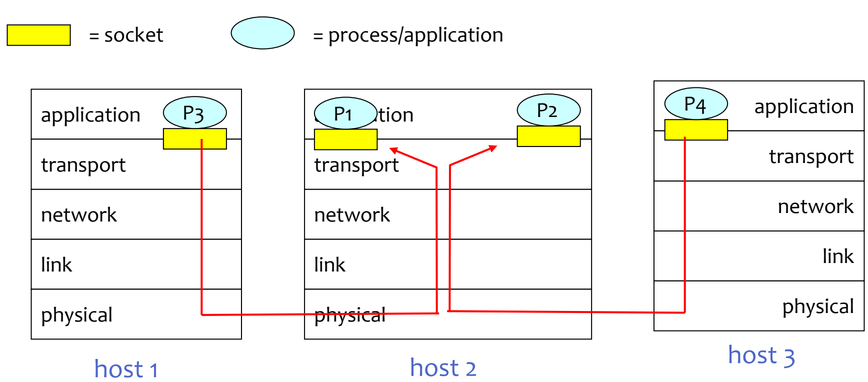


5

RCom 21/22 - Application Layer

U.PORTO

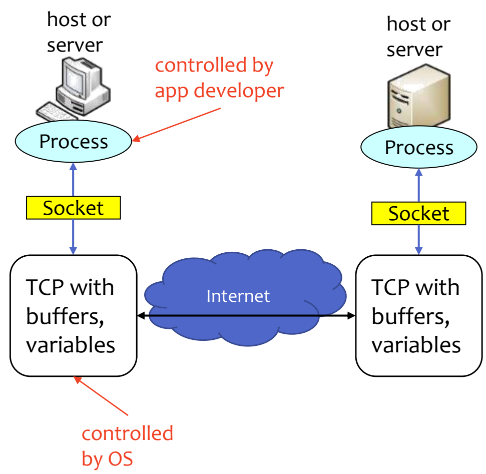
Applications and Sockets



6

Sockets

- process sends/receives messages to/from its socket
- socket analogous to door
  - sending process shoves message outdoor
  - sending process relies on transport infrastructure on other side of door which brings message to socket at receiving process
- API: (1) choice of transport protocol; (2) ability to fix a few parameters



App-layer protocol defines

- Types of messages exchanged,
  - e.g., request, response
- Message syntax:
  - what fields in messages & how fields are delineated
- Message semantics
  - meaning of information in fields
- Rules for when and how processes send & respond to messages

Public-domain protocols:

- defined in RFCs
- allows for interoperability
- e.g., HTTP, SMTP, BitTorrent

Proprietary protocols:

- e.g., Skype, ppstream, Spotify

RCom 21/22 - Application Layer

U.PORTO

What transport service does an app need?

Data loss

- some apps (e.g., audio) can tolerate some loss
- other apps (e.g., file transfer, HTTP) require 100% reliable data transfer

Timing

- some apps (e.g., Internet telephony, interactive games) require low delay to be "effective"

Throughput

- some apps (e.g., multimedia) require minimum amount of throughput to be "effective"
- other apps ("elastic apps") make use of whatever throughput they get

Security

- Encryption, data integrity, …

9

RCom 21/22 - Application Layer

U.PORTO

Transport service requirements of common apps

Application	Data Loss	Throughput	Time sensitive
File transfer	No loss	elastic	No
email	No loss	elastic	No
Real-time audio/video	Loss-tolerant	audio: 5kbps-1Mbps video:10kbps-5Mbps[1]	Yes, 100's ms
Stored audio/video	Loss-tolerant	Same as above	Yes, few secs
Interactive games	Loss-tolerant	100's kbps up[2]	Yes, 10's ms
Instant messaging	No loss	elastic	Yes and no
Web documents	No loss	elastic	No
Prog. Web Apps	No loss	elastic	Yes, few secs

[1] may be higher for higher definitions or audio quality

[2] this is for locally running games not cloud gaming services

10

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

RCom 21/22 - Application Layer

Web and HTTP

HTTP Working Group

Network layer

RCom 21/22 - Application Layer

Web and HTTP jargon

- Web page consists of objects
- Object can be HTML file, image, audio/video file, css file, js file, …
- Web page consists of base HTML-file which includes several referenced objects
- Each object is addressable by a URL
- Example URL:

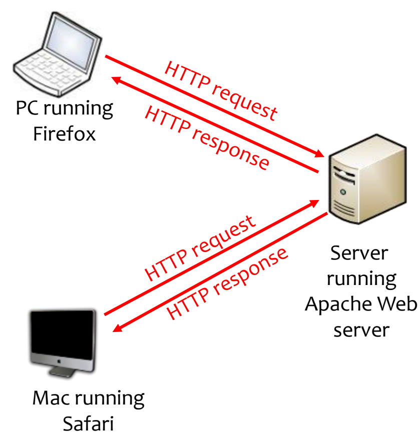
$$\underbrace{\texttt{sigarra.up.pt}}_{\text{host name}}\underbrace{\texttt{/js/up-plugins.js}}_{\text{path name}}$$

12

HTTP overview

HTTP: hypertext transfer protocol

- Web's application layer protocol
- client/server model
  - client: browser that requests, receives, "displays" Web objects
  - server: Web server sends objects in response to requests



13

HTTP overview (continued)

Uses TCP:

- client initiates TCP connection (creates socket) to server, port 80
- server accepts TCP connection from client
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- TCP connection closed

HTTP is "stateless"

- server maintains no information about past client requests

aside

Protocols that maintain "state" are complex!

- past history (state) must be maintained
- if server/client crashes, their views of "state" may be inconsistent, must be reconciled

14

RCom 21/22 - Application Layer

U.PORTO

HTTP connections

Nonpersistent HTTP

- At most one object is sent over a TCP connection.

Persistent HTTP

- Multiple objects can be sent over single TCP connection between client and server.

15

RCom 21/22 - Application Layer

U.PORTO

Nonpersistent HTTP

Suppose user enters URL

(contains text, references to images, css files and js files)

`sigarra.up.pt/up/pt/web_base.gera_pagina?p_pagina=home`

1a. HTTP client initiates TCP connection to HTTP server (process) at `sigarra.up.pt` on port 80

1b. HTTP server at host `sigarra.up.pt` waiting for TCP connection at port 80. "accepts" connection, notifying client

2. HTTP client sends HTTP *request message* (containing URL) into TCP connection socket. Message indicates that client wants object `up/pt/web_base.gera_pagina?p_pagina=home`

3. HTTP server receives request message, forms *response message* containing requested object, and sends message into its socket

time

16

Nonpersistent HTTP

4. HTTP server closes TCP connection.

5. HTTP client receives response message containing html file, displays html. Parsing html file, finds references to other objects

6. Steps 1 to 5 repeated for each object. Note that **includes new TCP connections**.

time

file transmission time
total = 2RTT+ file transmit time

17

Persistent HTTP

Nonpersistent HTTP issues:

- requires 2 RTTs per object
- OS overhead for each TCP connection
- browsers often open parallel TCP connections to fetch referenced objects

Persistent HTTP

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server sent over open connection
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

18

HTTP request message

- two types of HTTP messages: *request*, *response*
- HTTP request message:
  - ASCII (human-readable format)

request line (GET, POST, HEAD, …)

header lines

```
GET /up/pt/ HTTP/1.1
Host: sigarra.up.pt
User-agent: Mozilla/5.0
Connection: close
Accept-language: pt
```

Carriage return, line feed indicates end of message

(extra carriage return, line feed)

Example

HTTP request message: general format

- **HTTP headers, from Mozilla**
- **Message Headers from IANA**

sp: space
cr: carriage return
lf: line feed

Method	sp	URL	sp	Version	cr	lf	
Header field name	:	Value	cr	lf			
…							
Header field name	:	Value	cr	lf			
Body							

RCom 21/22 - Application Layer

Uploading form input

Post method:

- Web page often includes form input
- Input is uploaded to server in entity body

URL method:

- Uses GET method
- Input is uploaded in URL field of request line:

`sigarra.up.pt/feup/pt/ucurr_geral.ficha_uc_view?pv_ocorrencia_id=484435`

21

RCom 21/22 - Application Layer

HTTP response message

status line (protocol status code status phrase)

header lines

data, e.g., requested HTML file

```
HTTP/1.1 200 OK
Connection close
Date: Thu, 26 Nov 2019:09:15 GMT
Server: Apache/2.4.52 (Unix)
Last-Modified: Mon, 21 Nov 2019 …...
Content-Length: 6821
Content-Type: text/html

data data data data data ...
```

22

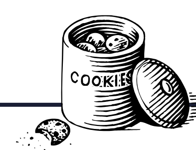
HTTP response status codes

- In first line in server→ client response message.

Some codes:

- 200 OK
  - request succeeded, requested object later in this message
- 301 Moved Permanently
  - requested object moved, new location specified later in this message (Location:)
- 400 Bad Request
  - request message not understood by server
- 404 Not Found
  - requested document not found on this server
- 505 HTTP Version Not Supported

User-server state: cookies



Example:

- Susan access Internet always from PC
- visits specific e-commerce site for first time
- when initial HTTP requests arrives at site, site creates:
  - unique ID
  - entry in backend database for ID

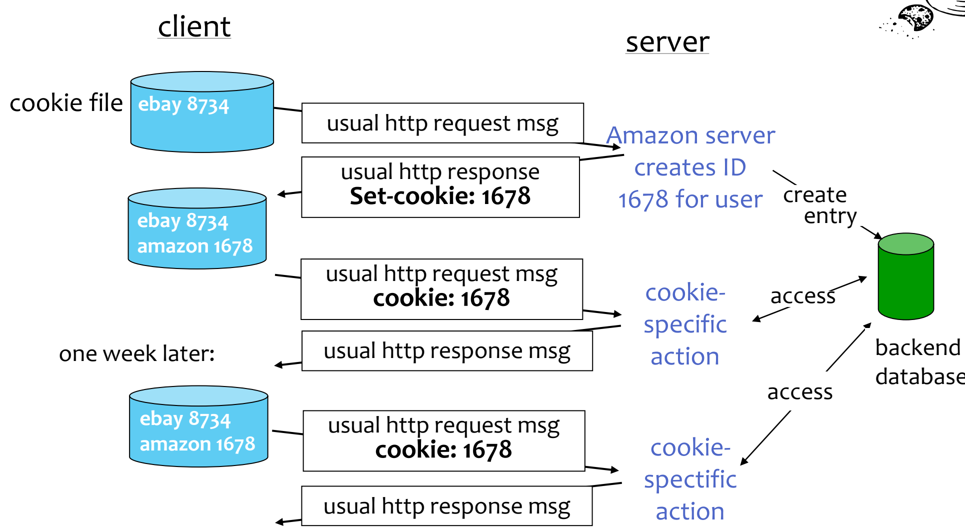
Many major Web sites use cookies

Four components:

1) cookie header line of HTTP *response* message
2) cookie header line in HTTP *request* message
3) cookie file kept on user's host, managed by user's browser
4) back-end database at Web site

- RFC 2965 - HTTP State Management Mechanism

Cookies: keeping "state" (cont.)



Cookies (continued)

What cookies can bring:

- authorization
- shopping carts
- recommendations
- user session state (Web e-mail)

aside

Cookies and privacy:

- cookies permit sites to learn a lot about you
- And there are other ways to *finger* you (are you unique, can you cover your tracks)

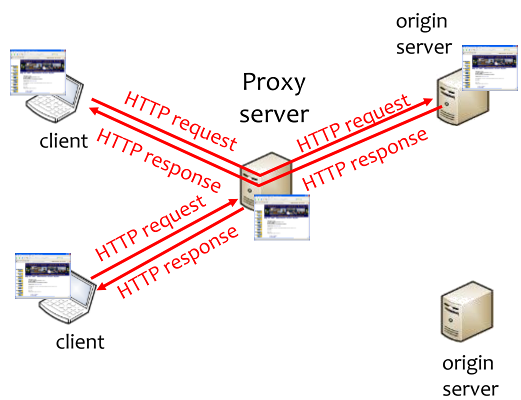
How to keep "state":

- protocol endpoints: maintain state at sender/receiver over multiple transactions
- cookies: http messages carry state

U.PORTO

Web caches (proxy server)

- Goal: satisfy client request without involving origin server
- user sets browser: Web accesses via cache
- browser sends all HTTP requests to cache
  - object in cache: cache returns object
  - else cache requests object from origin server, then returns object to client



27

RCom 21/22 - Application Layer

U.PORTO

More about Web caching

- cache acts as both client and server
- typically cache is installed by ISP (university, company, residential ISP)

Why Web caching?

- reduce response time for client request
- reduce traffic on an institution's access link.

- "Caching" is also used in Content Delivery Networks (CDN) for better user experience

28

FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO
U.PORTO
U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

RCom 21/22 - Application Layer

DNS Domain Name Service

Network layer

RCom 21/22 - Application Layer

DNS: Domain Name System

- People: many identifiers:
  - BI, name, passport nr
- Internet hosts, routers:
  - IP address (32 bit) - used for addressing datagrams
  - "name", e.g., ww.google.com - used by humans
- Q: 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"

U.PORTO

30

DNS

RCom 21/22 - Application Layer

DNS services

- hostname to IP address translation
- host aliasing
  - Canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: set of IP addresses for one canonical name

Why not centralize DNS?

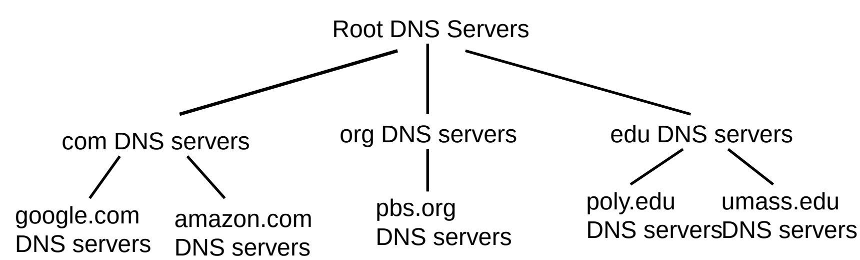
- single point of failure
- traffic volume
- distant centralized database
- maintenance

- doesn't scale!

31

Distributed, Hierarchical Database

RCom 21/22 - Application Layer



Client wants IP for www.amazon.com; 1st approx:

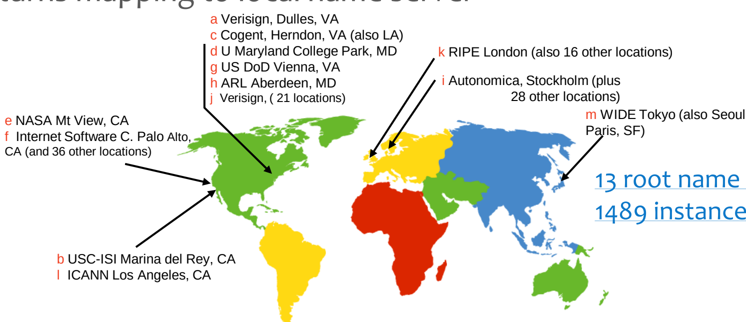
- client queries a root server to find `com` DNS server
- client queries `com` DNS server to get `amazon.com` DNS server
- client queries `amazon.com` DNS server to get IP address for `www.amazon.com`

32

RCom 21/22 - Application Layer

DNS: Root name servers

- contacted by local name server that can not resolve name
- root name server:
  - contacts authoritative name server if name mapping not known
  - gets mapping
  - returns mapping to local name server

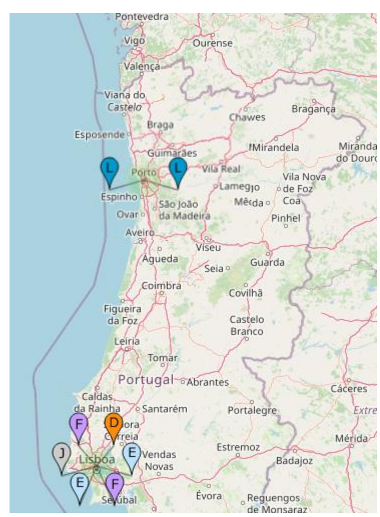


13 root name servers worldwide, with 1489 instances (20/Jan/2022 12:21)

RCom 21/22 - Application Layer

DNS: Root name servers

- On root-servers the location of the several instances are available.



RCom 21/22 - Application Layer

U.PORTO

TLD and Authoritative Servers

- Top-level domain (TLD) servers:
  - responsible for com, org, net, edu, etc, and all top-level country domains uk, fr, ca, jp.
  - and generic TLDs
  - and sponsored TLDs

- Authoritative DNS servers:
  - organization's DNS servers, providing authoritative hostname to IP mappings for organization's servers (e.g., Web, mail).
  - can be maintained by organization or service provider

35

RCom 21/22 - Application Layer

U.PORTO

Local Name Server

- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one.
  - also called "default name server"
- when host makes DNS query, query is sent to its local DNS server
  - acts as proxy, forwards query into hierarchy

36

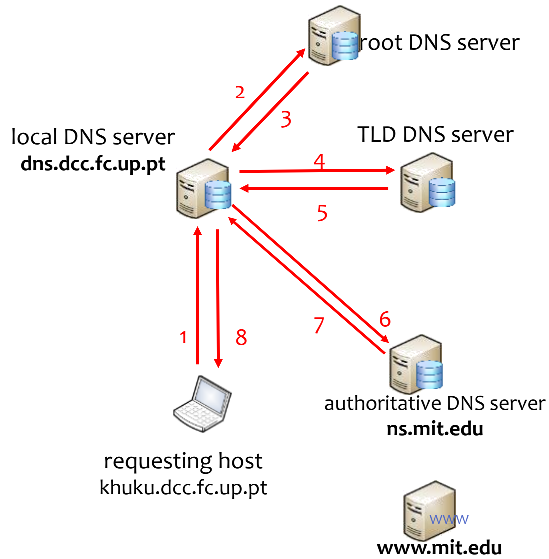
RCom 21/22 - Application Layer

DNS name resolution example

- Host at khuku.dcc.fc.up.pt wants IP address for www.mit.edu

iterated query:

- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



37

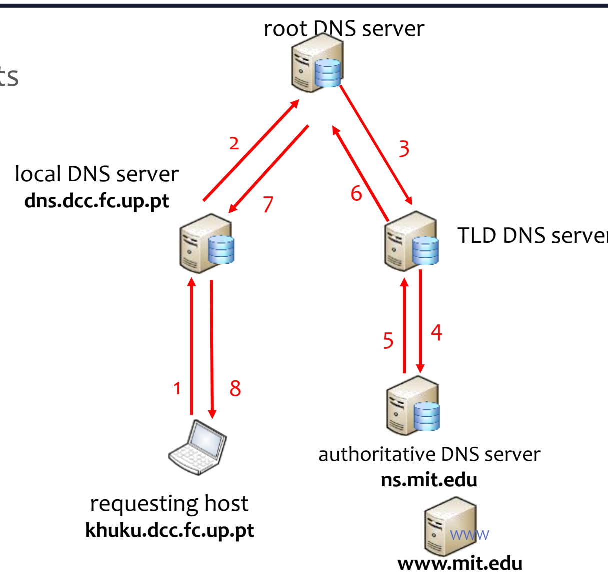
RCom 21/22 - Application Layer

DNS name resolution example

- Host at khuku.dcc.fc.up.pt wants IP address for www.mit.edu

recursive query:

- puts burden of name resolution on contacted name server
- heavy load?



38

DNS records

distributed db storing resource records (RR)

- Type=A
  - **name** is hostname
  - **value** is IP address
- Type=NS
  - **name** is domain (e.g. foo.com)
  - **value** is hostname of authoritative name server for this domain
- Type=CNAME
  - **name** is alias name for some "canonical" (the real) name `www.fe.up.pt` is really `sifeup.fe.up.pt`
  - **value** is canonical name
- Type=MX
  - **value** is name of mailserver associated with **name**

```
RR format: (name, value, type, ttl)
```


DNS protocol, messages

- DNS protocol : query and reply messages, both with same message format

msg header

- identification: 16-bit nr for query, reply to query uses same nr
- flags:
  - query or reply
  - recursion desired
  - recursion available
  - reply is authoritative

Identification	Flags
Number of questions	Number of answers RRs
Number of authority RRs	Number of additional RRs
Questions (variable nr)	
Answers (variable nr of RR)	
authority (variable nr of RR)	
Additional info (variable nr of RR)	

12 bytes

DNS protocol, messages

	Identification	Flags
	Number of questions	Number of answers RRs
	Number of authority RRs	Number of additional RRs
Name, type fields for a query	Questions (variable nr)	
RRs in response to query	Answers (variable nr of RR)	
records for authoritative servers	authority (variable nr of RR)	
additional "helpful" info that may be used	Additional info (variable nr of RR)	

Inserting records into DNS

- example: new startup "Network Utopia"
- register name networkuptopia.com at DNS registrar (e.g., Network Solutions)
  - provide names, IP addresses of authoritative name server (primary and secondary)
  - registrar inserts two RRs into com TLD server:

```
(networkutopia.com, dns1.networkutopia.com, NS)
(dns1.networkutopia.com, 212.212.212.1, A)
```

- create authoritative server Type A record for www.networkuptopia.com; Type MX record for networkutopia.com
- How do people get IP address of your Web site?

Security in DNS

Why is DNS security important?, from CloudFlare

- Your IP is public
- Your query (the name you want resolve) is public
  - Can be eavesdrop on the way to the DNS server
- When iterative every contacted server knows the name to resolve
- Several solutions
  - DNSSec (RFC4033): not widely adopted by clients
  - DNSCrypt: needs a DNS proxy installed
  - DNS over TLS (DoT, RFC7858): needs support of TLS from DNS server
  - DNS over HTTPS (DoH, RFC8484): the query needs to go a WebServer

43

Security in DNS – DoH

- Using DoH the name resolution query is sent as an HTTP POST using TLS for security (authentication of the server and content encryption)

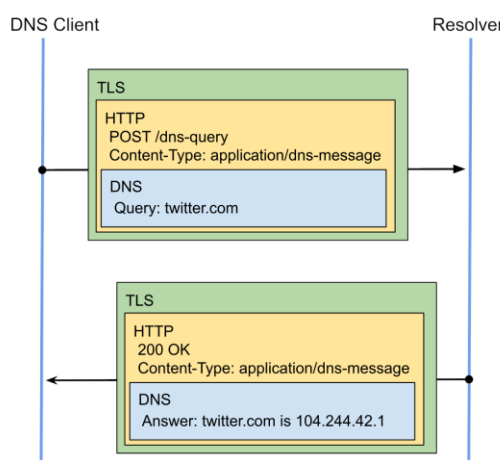


Image from DNS Encryption Explained, from CloudFlare

44

U.PORTO

Summary

- Principles of Application Protocols
- The HTTP protocol
  - Messages
  - Connection
  - *Keeping state*
  - Caching
- DNS
  - Hierarchy and domain levels
  - Queries
  - Security

U.PORTO

Homework

1. Review slides
2. Read from Tanenbaum
   - Section 7.1 –DNS - The Domain Name System
   - Section 7.3.5 – HTTP - The HyperText Transport Protocol
3. ***No*** questions at Moodle