



HTTP: the language of web communication

Claudia Hauff

cse1500-ewi@tudelft.nl



Web technology overview

1. **HTTP: the language of web communication**
2. **HTML** & web app design
3. **JavaScript**: interactions in the browser
4. **Node.js**: JavaScript on the server
5. **CSS**: adding style
6. Node.js: advanced topics
7. Cookies & sessions
8. Web security

Source: <https://vimeo.com/110256895>

1:50 min

Learning goals

- **Describe** how web servers and clients interact with each other.
- **Request** resources from web servers and understand the responses.
- **Describe** the different URL components.
- **Understand** and **employ** basic HTTP authentication.
- **Explain** the difference between HTTP and HTTPS.

World Wide Web

vs.

Internet

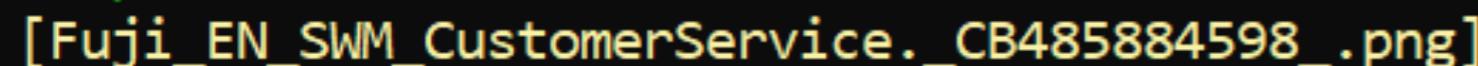
The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet
(envisioned already in 1945)



- **1960s:** Precursor to the Internet (ARPANET) devised by the US department of Defense
 - Initial services: electronic mail, file transfer
- Late **1980s:** Internet opened to commercial interests
- **1989:** WWW created by Tim Berners-Lee (CERN)



[Skip to main content](#) [Fuji_EN_SWM_CustomerService._CB485884598_.png]

All

Select the department you want to search in [All Departments_____]

Go Go

Search _____

Deliver toNetherlands

EN Hello, Sign in Account & Lists Sign in Account & Lists Orders Cart 0

Departments

Today's Deals Your Amazon.com Gift Cards Help Registry Sell Disability Customer Suppo

Welcome to Amazon.com. If you prefer a simplified shopping experience, try the mobile
www.amazon.com/access. The mobile web version is similar to the mobile app. Stay on /
all the features of the main Amazon website.[Previous page](#)

1. We ship over 45 million products around the world
- 2.
- 3.
- 4.

[Next page](#)[Shop by Category](#)[Computers & Accessories](#)[Computers & Accessories](#)[Video Games](#)

(NORMAL LINK) Use right-arrow or <return> to activate.

Arrow keys: Up and Down to move. Right to follow a link; Left to go back.

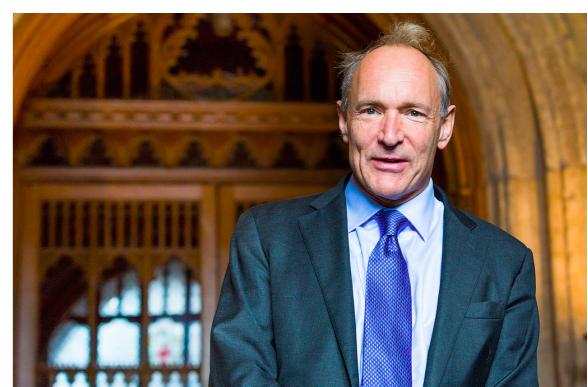
H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet
(envisioned already in 1945)



- **1960s:** Precursor to the Internet (ARPANET) devised by the US department of Defense
 - Initial services: electronic mail, file transfer
- Late **1980s:** Internet opened to commercial interests
- **1989:** WWW created by Tim Berners-Lee (CERN)
- **1994:** Netscape released its first Web browser





Location: about:

[Guided Tour](#) [What's New](#) [Questions](#) [Net Search](#) [Net Directory](#) [Newsgroups](#)

Mosaic Netscape version 0.9 beta

Copyright © 1994 Mosaic Communications Corporation,
All rights reserved.

This is *BETA* software subject to the license agreement set forth in the README file.
Please read and agree to all terms before using this software.

Report any problems to win_cbug@mcom.com.



Mosaic Communications, Mosaic Netscape, and the Mosaic Communications logo are trademarks of Mosaic Communications Corporation.

Any provision of Mosaic Software to the U.S. Government is with "Restricted rights" as follows: Use, duplication or disclosure by the Government is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer Restricted Rights clause at FAR 52.227-19 when applicable, or in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, and in similar clauses in the NASA FAR Supplement. Contractor/manufacturer is Mosaic Communications Corporation, 650 Castro Street, Suite 500, Mountain View, California, 94041.

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet
(envisioned already in 1945)



- **1960s:** Precursor to the Internet (ARPANET) devised by the US department of Defense
 - Initial services: electronic mail, file transfer
- Late **1980s:** Internet opened to commercial interests
- **1989:** WWW created by Tim Berners-Lee (CERN)
- **1994:** Netscape released its first Web browser
- **1995:** Microsoft released Internet Explorer v1
- **1998:** Google was founded
- **2002:** Mozilla released Firefox v1

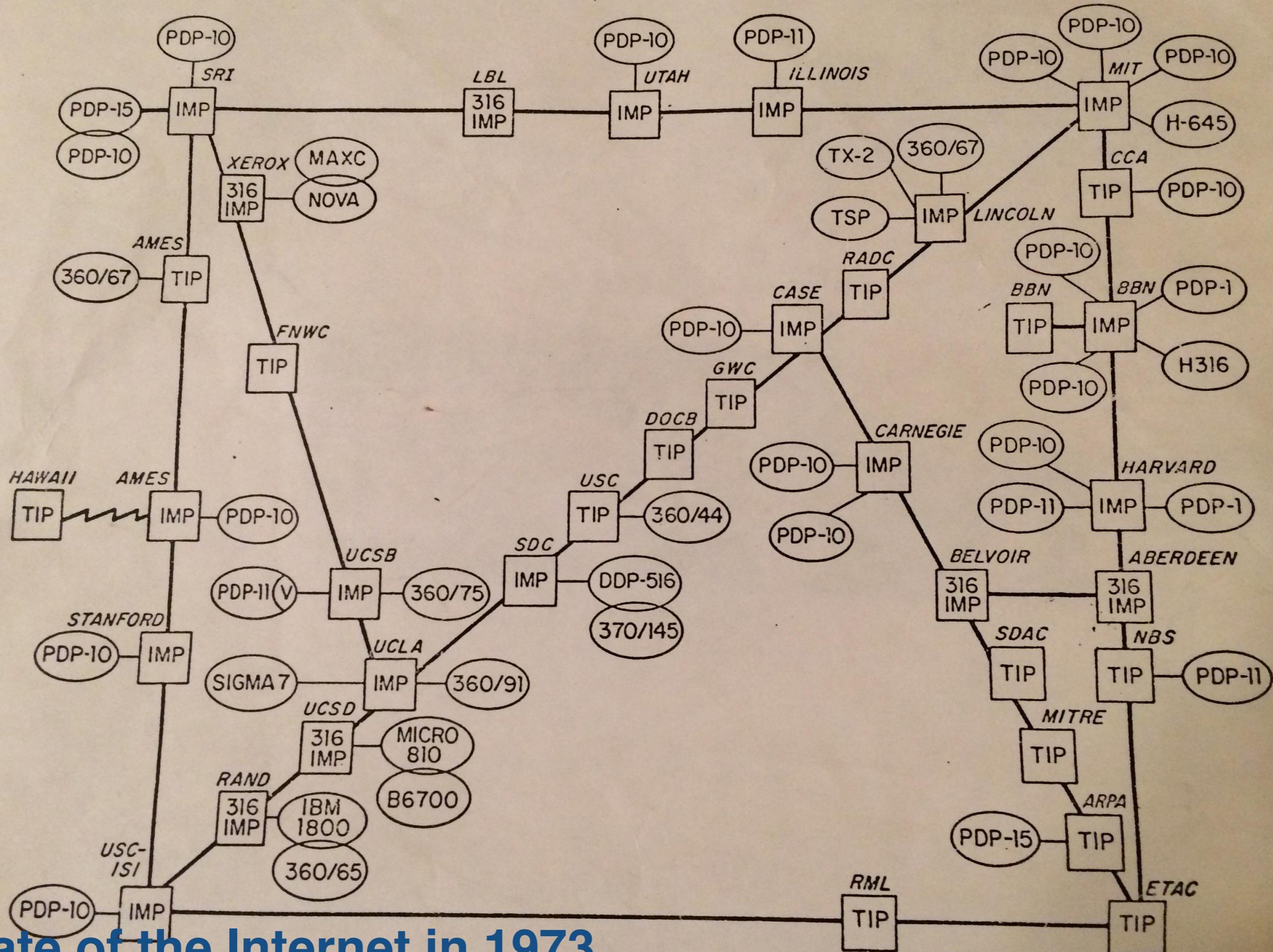


Key aspects of the Internet

Internet: interconnected computer networks that span the globe and communicate through a common standard (TCP/IP).

- Sub-networks function **autonomously**
- **No centralised** control
- Devices **dynamically** join/leave the network
- Devices interact through **open standards**
- **Easy** to use: server/client software widely available

ARPA NETWORK, LOGICAL MAP, MAY 1973



State of the Internet in 1973

Image source: <https://twitter.com/worker gnome/status/807704855276122114>

Two important organisations

Internet Engineering Task Force (IETF)

"The mission of the IETF is to make the Internet work better by producing high quality, relevant technical documents that influence the way people design, use, and manage the Internet."

Request for Comments (RFC)

World Wide Web Consortium (W3C)

"The W3C mission is to lead the World Wide Web to its full potential by developing protocols and guidelines that ensure the long-term growth of the Web."

HTTP messages

We cover HTTP/1.1
in this lecture!

HTTP/1.1

RFC 2068

1997

HTTP 0.9

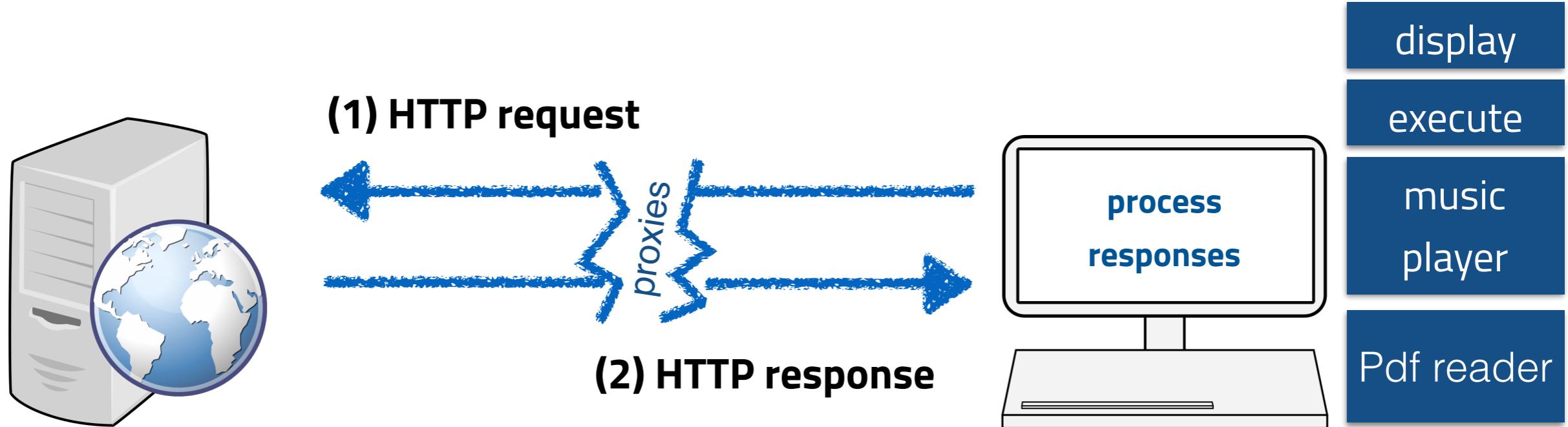
HTTP/2

G RFC 7540

2015

G HTTP/3

Web servers and clients



- Servers wait for data requests
- Answer thousands of clients simultaneously
- Host **web resources** (content with an identity)
- Proxies have several functions: filtering, caching, authentication, etc.
- Clients are often browsers
 - Telnet

Network communication

- Conceptual model **Open Systems Interconnection** (OSI) from 1995
- Network protocols matched to layers
- Many network protocols exist, we care about three in particular:

IP Internet Protocol
TCP Transmission Control Protocol
HTTP Hypertext Transfer Protocol

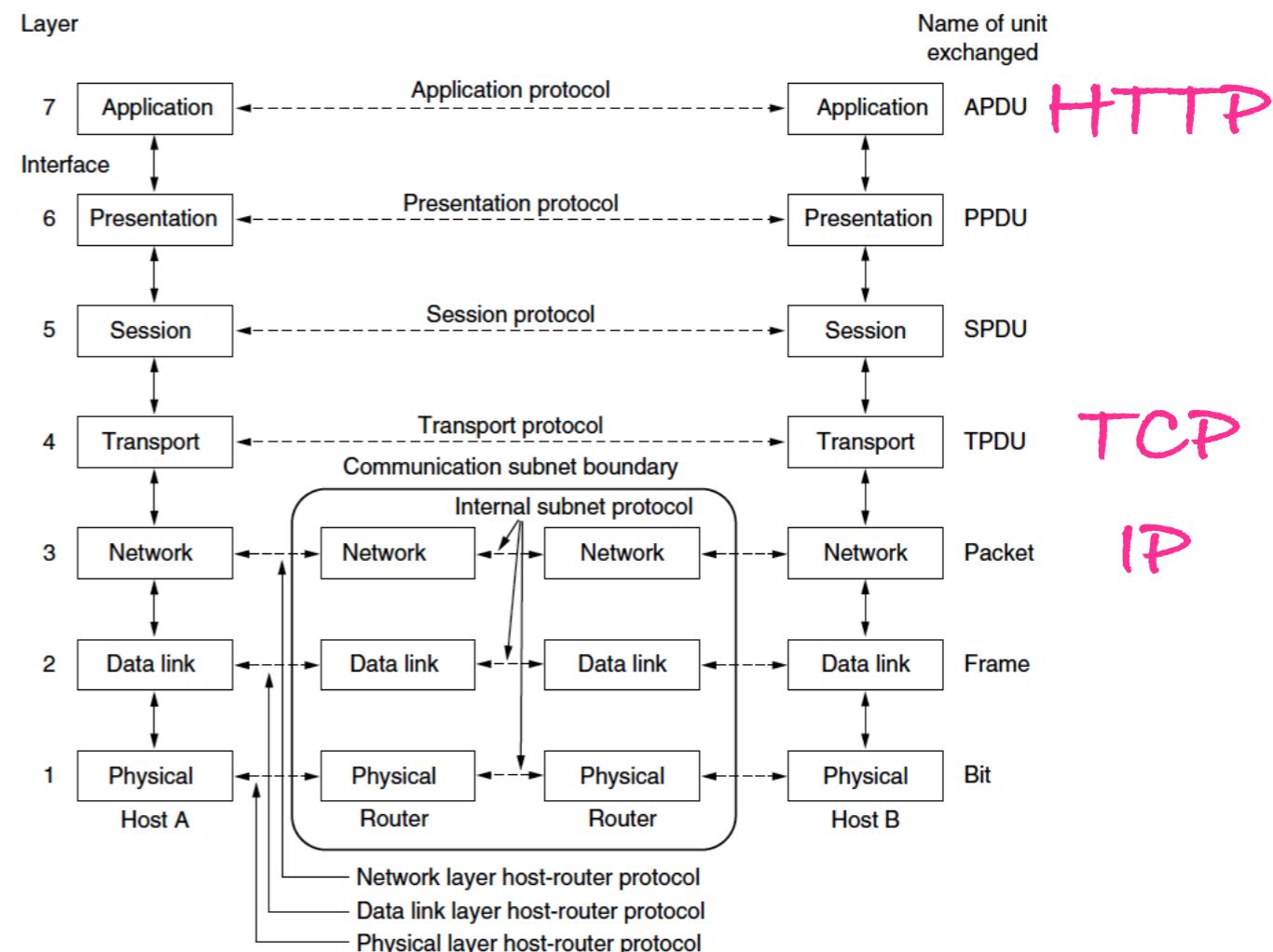
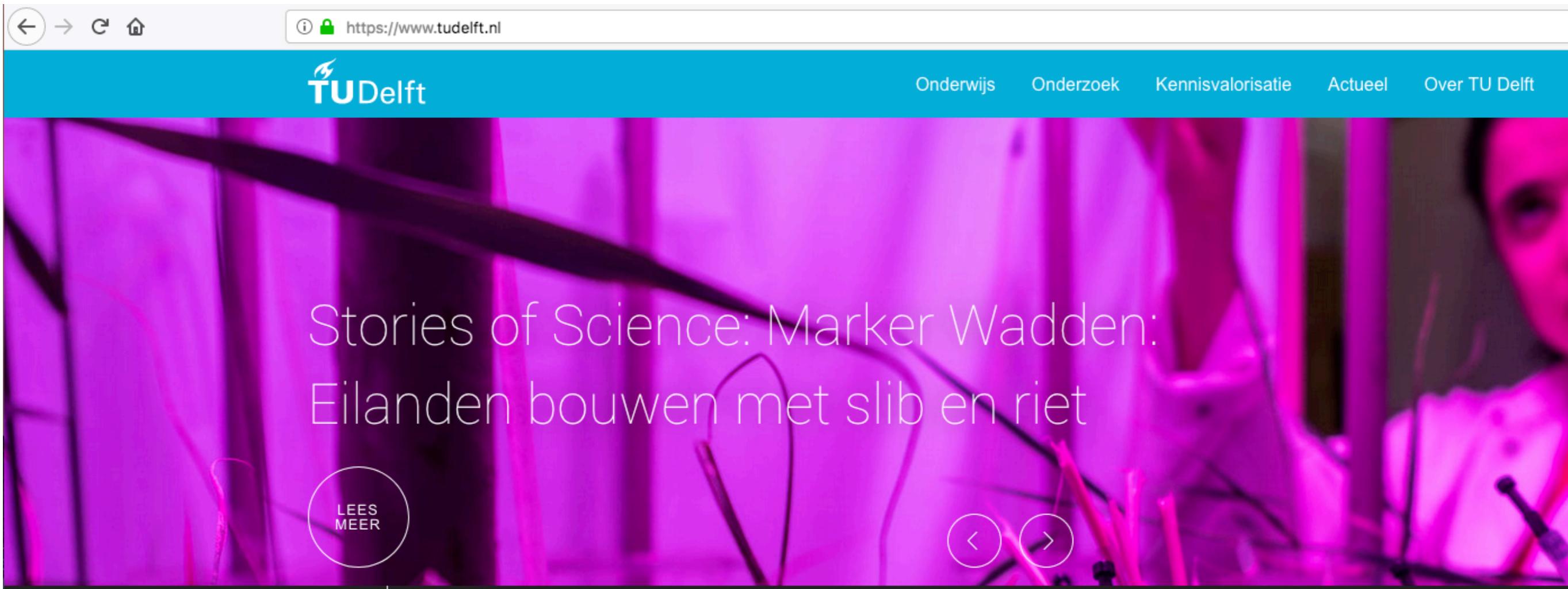


Image src: Computer Networks (5th edition), Tanenbaum & Wetherall, p. 42

HTTP uses **reliable** data-transmission protocols (inherited from TCP).



Demo time

A screenshot of a browser developer tools Network tab showing network requests for the TU Delft website. The table lists 33 requests, all successful (200 status code), transferred 2.40 MB, and finished in 2.48 seconds. The requests include files like index.html, CSS, JS, and various image files (e.g., torch.svg, logo.svg, various jpg files). The domain for most requests is www.tudelft.nl, while some images are served from siteimproveanalytics.com and d1rkab7tlqy5f1.cloudfront.net.

Status	Method	File	Domain	Cause	Type	Transferred	Size
200	GET	/	www.tudelft.nl	document	html	10.69 KB	53.84 KB
304	GET	dist.style.8be0425eaa.min.css	www.tudelft.nl	stylesheet	css	cached	177.21 KB
200	GET	modernizr.min.js	www.tudelft.nl	script	js	3.34 KB	6.68 KB
200	GET	dist.javascripts.831a266649.min.js	www.tudelft.nl	script	js	85.71 KB	279.57 KB
200	GET	logo.svg	www.tudelft.nl	img	svg	2.43 KB	5.07 KB
200	GET	torch.svg	www.tudelft.nl	img	svg	1.50 KB	2.12 KB
200	GET	siteanalyze_6005654.js	siteimproveanalytics.com	script	js	8.83 KB	26.24 KB
304	GET	csm_bachelorvoorlichting_4_fd66b0ebb8.jpg	d1rkab7tlqy5f1.cloudfront.net	img	jpeg	cached	116.95 KB
304	GET	GettyImages-691090647 hashtag twitteronderzoek_400x200_c_det.jpg	d1rkab7tlqy5f1.cloudfront.net	img	jpeg	cached	90.74 KB
304	GET	csm_News_180503_BoostPhysics_MJVanRiel_2_5517102b38.jpg	d1rkab7tlqy5f1.cloudfront.net	img	jpeg	cached	108.12 KB
200	GET	csm_GS_HARKOS 010_0355811622.jpg	d1rkab7tlqy5f1.cloudfront.net	img	jpeg	29.86 KB	29.43 KB
200	GET	csm_Backlit_keyboard_6959477114.jpg	d1rkab7tlqy5f1.cloudfront.net	img	jpeg	44.25 KB	43.82 KB
304	GET	csm_20140328_tudelft_campus_lente_R9B9442_dc999599da.jpg	d1rkab7tlqy5f1.cloudfront.net	img	jpeg	cached	166.64 KB
304	GET	csm_Building wetlands with soft mud5_b2e2da8269.jpg	d1rkab7tlqy5f1.cloudfront.net	img	jpeg	cached	113.41 KB

33 requests | 2.40 MB / 2.24 MB transferred | Finish: 2.48 s | DOMContentLoaded: 1.35 s | load: 2.49 s

HTTP request message

plain text, line-oriented character sequences

```
GET / HTTP/1.1
Host: www.tudelft.nl
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
10.9; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,*/*;q=0.8
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Cookie:
__utma=1.20923577936111.16111.19805.2;utmcmd=(none);
```

HTTP response message

```
HTTP/1.1 200 OK
```

start line

```
Date: Fri, 01 Aug 2014 13:35:55 GMT
```

```
Content-Type: text/html; charset=utf-8
```

```
Content-Length: 5994
```

```
Connection: keep-alive
```

```
Set-Cookie: fe_typo_user=d5e20a55a4a92e0;  
path=/; domain=tudelft.nl
```

```
[ ... ]
```

```
Server: TU Delft Web Server
```

```
....
```

```
....
```

header fields

name:value

body

(optional)

HTTP headers dissected

Well-known header fields

Content-Type	Entity type
Content-Length	Length/size of the message
Content-Language	Language of the entity sent (e.g. English)
Content-Encoding	Data transformations applied to the entity
Content-Location	Alternative location of the entity
Content-Range	Range defines the pieces sent for partial entities
Content-MD5	Checksum of the content
Last-Modified	Date on which this entity was created/modified
Expires	Date at which the entity will become stale
Allow	Lists the legal request methods for the entity
Connection & Upgrade	Protocol upgrade

Entity bodies contain **raw** data. **Header** needed to **interpret** the data.

Content-Type

- **MIME** types are attached to all HTTP object data

Multipurpose Internet **Mail** Extensions

historic reasons

- Type determines clients' reaction to the data
- Pattern: [primary object type] / [subtype]
 - text/plain
 - text/html
 - image/jpeg
 - video/quicktime

Content-Types are diverse

Most popular

text/html
application/xhtml+xml
application/pdf
application/rss+xml
image/jpeg
application/atom+xml
text/plain
application/pdf
application/xml
text/calendar

Least popular

video/quicktime
text/rdf+n3
application/postscript
application/x-bzip2
application/msword
application/coreldraw
audio/midi
application/vnd.ms-powerpoint
application/pgp-signature

Common Crawl

The list is based on a sample of a 2019 large-scale web crawl.



Content-Length

- Indicates the **size** of the entity body
- Necessary to detect premature message truncation
(e.g. due to a server crash, faulty proxy)
- Essential for **persistent connections** to discover where one HTTP message ends and the next begins

Persistent connections reuse the same TCP connection for multiple HTTP request/response messages.

Content-Encoding

- Commonly either `gzip`, `compress` (Unix compression),
`deflate` (zlib compression) or `identity` (no encoding)
- Servers aim for **encodings** that clients understand
 - Clients send a list of acceptable encodings in a corresponding request header: `Accept-Encoding: gzip, deflate`
- **Compression** saves network bandwidth but increases processing costs

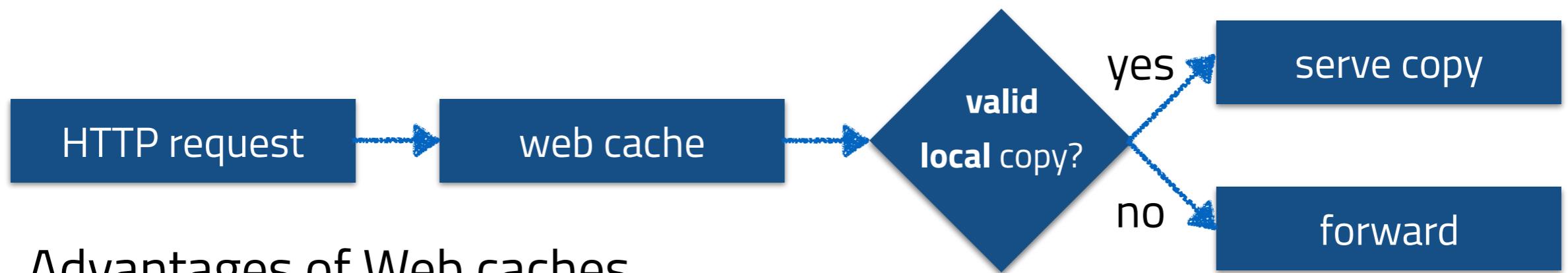
Content-MD5 (RFC 1321)

Message Digest

- HTTP messages are sent via TCP (*this will change in HTTP/3*)
- **However:** the Internet is huge, many servers interact to transport a message with different implementations (**bugs!**) of established protocols
- **Sanity check:** sender generates a **MD5 checksum** of the content (hashed into a 128 bit value) to detect unintended modifications of the content
- Has been removed from the HTTP/1.1 specification (2014), but remains in use (simple technique, easy to implement)

Expires

- **Web caches** keep copies of *popular* resources



- Advantages of Web caches
 - A. **Reduction** of redundant data transfer
 - B. **Reduction** of network bottlenecks
 - C. **Reduction** demand on origin servers
 - D. **Reduced** distance delay

- **Expires** indicates when the resource is no longer valid

Expires & Cache-Control

- Content on the origin server can change
- Caches need to ensure that their copies are **in sync** with the origin server
- Caches can revalidate their copies at any time (inefficient)
- **Expires** in HTTP response header indicates a **resource's expiration date** in **absolute** terms — date determines when the cache revalidates
- **Cache-Control** indicates a resource's expiration date in **relative** terms (seconds since being sent)

Expires & Cache-Control

- Content on the origin server
- Caches need to ensure they contact the origin server
- Caches can revalidate their content
- Expires in HTTP response
expiration date in absolute time
cache revalidates
- Cache-Control indicates relative terms (seconds):

Headers Cookies Params Response Timings Stack Trace

Request URL: <https://www.theguardian.com/uk>
Request method: GET
Remote address: 151.101.37.11
Status code: 200 ⓘ Edit and
Version: HTTP/2.0

Filter headers

Response headers (1.511 KB)

- accept-ranges: bytes
- age: 74
- cache-control: max-age=60, stale-while-revali...tale-if-error=864000, private
- content-encoding: gzip
- content-length: 138026
- content-security-policy: default-src https://data:; connect-src https: wss:
- content-type: text/html; charset=utf-8
- date: Fri, 19 Oct 2018 08:35:14 GMT
- etag: W/"hash-3272292653060572815"
- link: rel=preload; as=script; nopush
- set-cookie: GU_mvt_id=894690; expires=Thu,...th=/; domain=.theguardian.com
- set-cookie: GU_geo_continent=EU; path=/;
- strict-transport-security: max-age=31536000; includeSubDomains; preload
- vary: Accept-Encoding,User-Agent
- x-content-type-options: nosniff
- X-Firefox-Spdy: h2
- x-frame-options: SAMEORIGIN

Only a private cache (e.g. the browser cache) should cache the response.

Last-Modified

- Contains the date when the resource was last **altered**
- **No indication** about the amount of changes
- Often used in combination with **If-Modified-Since** for cache revalidation requests: origin server only returns the document if it changed since the given date
- Last-Modified dates are **not reliable**

Connection & Upgrade

Polling the 2018 Midterm Elections in Real Time

The Upshot has partnered with Siena College to poll dozens of the most competitive House and Senate races across the country. Our poll results are updated in real time, after every phone call. We hope to help you understand how polling works, and why it sometimes doesn't.

[Related article »](#)

[Sign up for alerts](#)

185,872
calls made so far

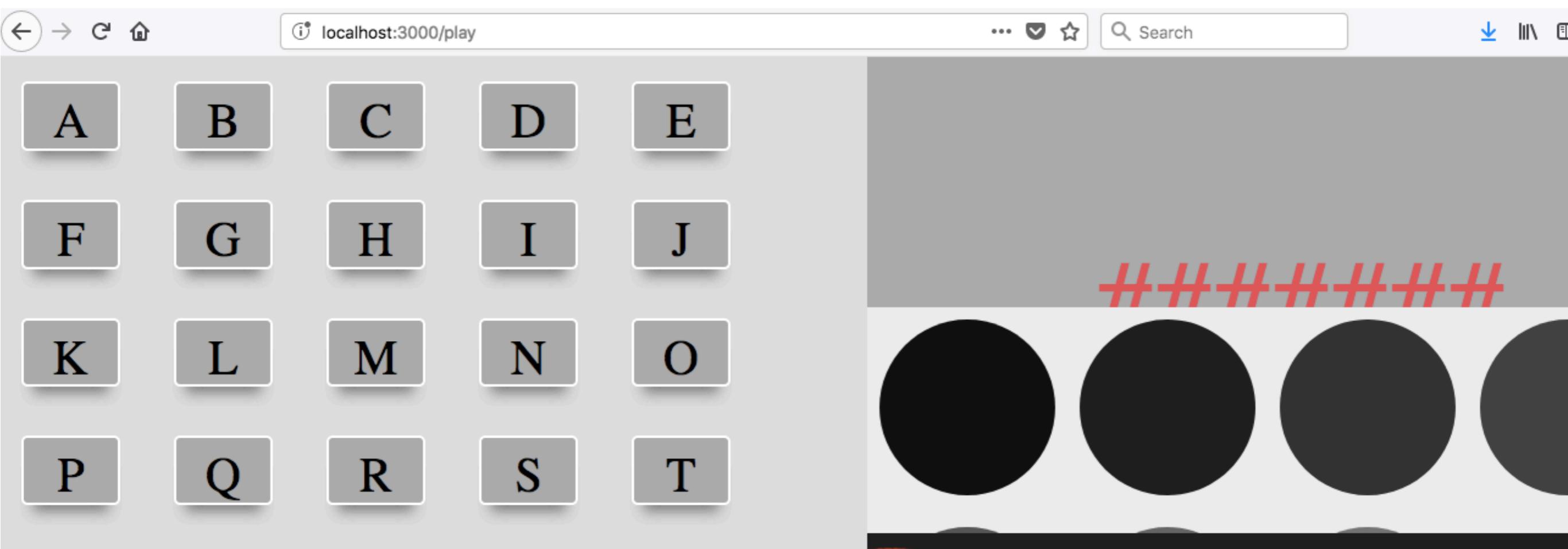
We're currently polling these races.

The screenshot shows a user profile for **Claudia Hauff** (@CharlotteHase) with 3,800 tweets, 742 following, and 1,611 followers. The bio mentions "Can a T" and "Continue". The main feed displays a tweet from **TEDxAmsterdam Ams** (@TEDxAmsterdam) with the text: "And what a brilliant #TEDxAmsterdam Award pitch it was! Applications for the 2018 Award are open until 5 Oct. Make like @GoodShipping1 did and send us your idea. Who knows how far it will take you? #thebigx #thebigunknown". A blue checkmark next to the TEDxAmsterdam handle indicates verification. A red circle highlights the number "185,872 calls made so far" in a large orange circle. Another red circle highlights the text "See 1 new Tweet" in a light orange circle.

Connection & Upgrade

- HTTP/1.1: the client **always** initiates the conversation
- Simulating a **server-side push** of data:
 - **Polling**: client regularly sends HTTP requests
 - **Long polling**: client sends an HTTP request and the server holds it open until new data arrives
 - Issues: wasted bandwidth, complex backend
- From simulation to solution (standardized in 2011): the **WebSocket protocol** enables **bidirectional communication** between client & server

Connection & Upgrade



The screenshot shows a web browser window with the URL `localhost:3000/play`. The main content area displays a 4x5 grid of letters A through T. To the right of the grid is a large, dark image of a printed circuit board (PCB) with several circular components. Red highlights are present on the PCB image, specifically along the top edge and on the right side.

The browser's developer tools are open at the bottom. The Network tab is selected, indicated by a red circle. The table below lists network requests:

Status	Method	F...	D...	Cause	Type	Transferred	Size	0 ms	640 ms	1.2	Headers	Cookies	Params	Response	Timings
200	GET	config.js	loc...	script	js	970 B	664 B	→ 28 ms			Request URL: http://localhost:3000/				
200	GET	status...	loc...	script	js	1.29 KB	0.99 KB	→ 32 ms			Request method: GET				
200	GET	messag...	loc...	script	js	2.43 KB	2.13 KB	→ 35 ms			Remote address: [::1]:3000				
200	GET	alphab...	loc...	script	js	2.86 KB	2.56 KB	→ 41 ms			Status code: 101	?	Edit and Resend	Raw headers	
200	GET	ui-right...	loc...	script	js	1.59 KB	1.29 KB	→ 37 ms			Version: HTTP/1.1				
200	GET	interact...	loc...	script	js	9.04 KB	8.74 KB	→ 37 ms			Filter headers				
200	GET	style.css	loc...	stylesheet	css	608 B	316 B	→ 4 ms			Pragma: no-cache				
101	GET	/	loc...	websocket	plain	129 B	0 B				Sec-WebSocket-Extensions: permessage-deflate				
											Sec-WebSocket-Key: PTXgPhQoo1j61/Zklj/uqQ==				
											Sec-WebSocket-Version: 13				
											Upgrade: websocket				

At the bottom of the table, a blue bar highlights the last row (Status 101). The status code 101 is circled in red. The Headers section on the right also has several entries circled in red, including "Sec-WebSocket-Extensions: permessage-deflate", "Sec-WebSocket-Key: PTXgPhQoo1j61/Zklj/uqQ==", "Sec-WebSocket-Version: 13", and "Upgrade: websocket".

At the very bottom of the browser window, there is a footer bar with the following information: 11 requests | 10.41 MB / 10.42 MB transferred | Finish: 2.10 s | DOMContentLoaded: 2.10 s | load: 2.14 s.

Connection & Upgrade

- Client and server have to agree to the **protocol upgrade**
- **Client initiates** the upgrade with two request headers:

Connection:Upgrade
Upgrade:[protocols]

- Server responds with a 101 Switching Protocols status if a protocol upgrade is possible
- Once established, both the client and server can push data

Common response status codes

10.4.3 402 Payment Required

This code is reserved for future use.

1xx	Informational	(101 Switching ...)
2xx	Success	(200 OK)
3xx	Redirected	
4xx	Client error	(404 Not Found)
5xx	Server error	

In practice only a few codes per category are supported.

HTTP methods

GET / HTTP/1.1

Host: www.tudelft.nl

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:31.0) Gecko/20100101 Firefox/31.0

Accept: text/html,application/xhtml+xml;q=0.9,*/*;q=0.8

Accept-Language: en-gb,en;q=0.5

Accept-Encoding: gzip, deflate

Common HTTP methods

GET

Get a document from the Web server.

HEAD

Get the header of a document from the Web server.

POST

Send data from the client to the server for processing.

PUT

Save the body of the request on the server.

TRACE

Trace the message through proxy servers to the server.

OPTIONS

Determine what methods can operate on a server.

DELETE

Remove a document from a Web server.

Servers may implement more or fewer methods than shown.

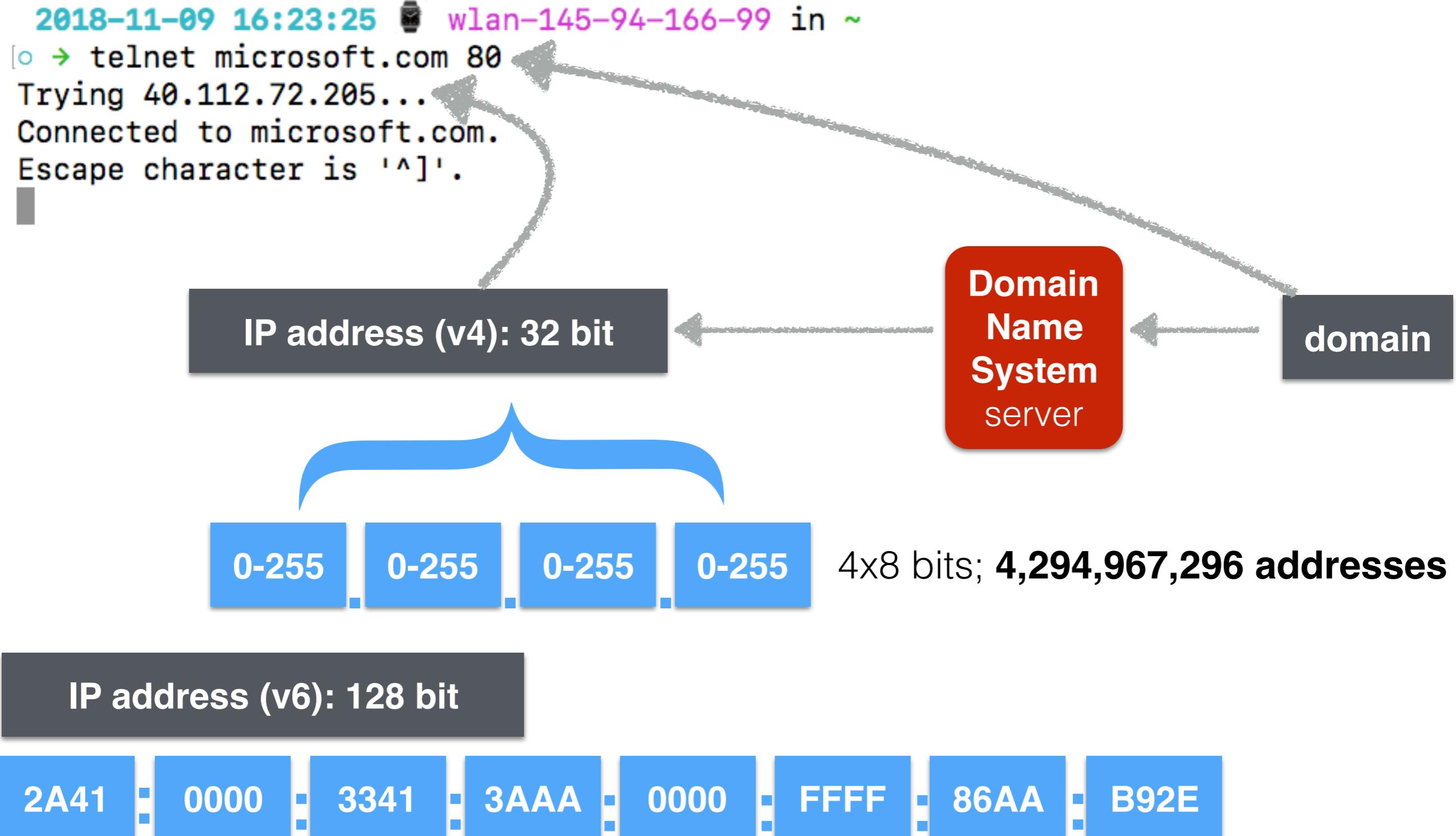
Use openssl
to test ssl/https!

```
2018-11-09 16:08:26 🕒 wlan-145-94-166-99 in ~  
[o → telnet microsoft.com 80  
Trying 40.112.72.205...  
Connected to microsoft.com.  
Escape character is '^]'.  
HEAD / HTTP/1.1  
host:microsoft.com  
  
HTTP/1.1 301 Moved Permanently  
Date: Fri, 09 Nov 2018 15:08:45 GMT  
Server: Kestrel  
Location: https://www.microsoft.com/
```

Demo time: telnet (RFC 15!)

Telnet opens a **TCP connection** to a web server; chars are typed directly into the port. The server treats telnet as web client, the returned data is displayed onscreen.

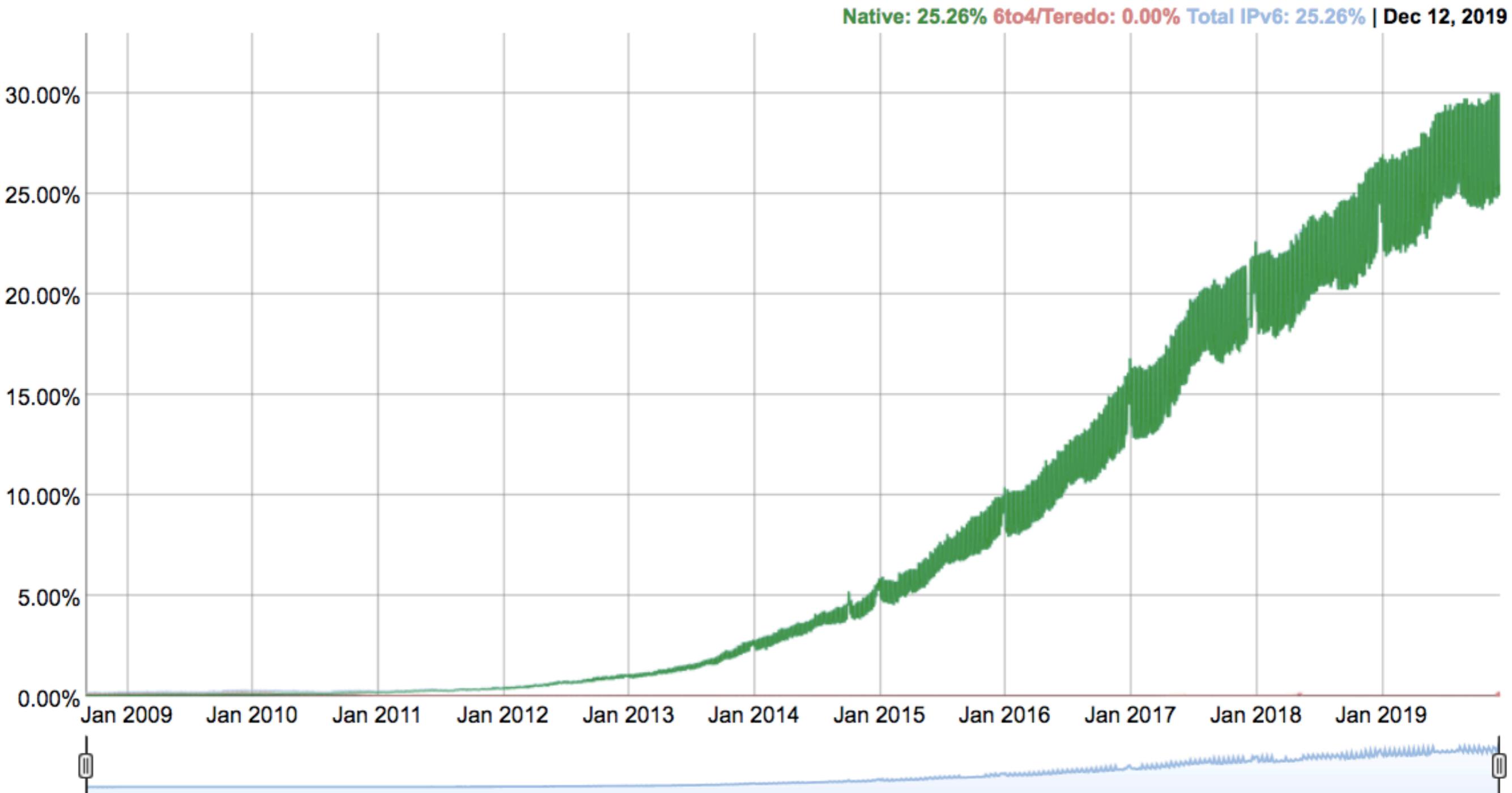
From domain to IP address



From domain to IP address

IPv6 Adoption

We are continuously measuring the availability of IPv6 connectivity among Google users. The graph shows the percentage of users that access Google over IPv6.



Uniform Resource Locators (URLs)

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

determines the protocol to use when connecting to the server

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

the username/password (to access a protected resource)

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

the domain name or numeric IP address of the server

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

the port on which the server is expecting requests

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

the local path to the resource

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

additional input parameters applications may require

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

parameters passed to gateway resources (e.g. a search engine)

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to `http`, however, the syntax slightly varies from scheme to scheme
- General format:

```
<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>
```

the name of a piece of a resource; only used by the client

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

common convention: name1=value1&name2=value2&...

WEB Images

Netherlands

30 Hotels in Delft - Best Price Guarantee. AD

Book your Hotel in Delft online. No reservation costs. Great rates.

Book Now, No Booking Fees, Book for Tonight

www.booking.com/Delft/Hotels | Report Ad

Your cityguide to Delft — Delft.com

Discover the story of William of Orange, get up close and personal with Johannes Vermeer and see how the world-famous Delft Blue is made.

D <https://www.delft.com>

Delft 2018: Best of Delft, The Netherlands Tourism - TripAdvisor

Vermeer's birthplace and a true gem, Delft sits between The Hague and Rotterdam in the country's southwest. The city's name comes from the Dutch word for digging, fitting since canals are a highlight here. Others include the 13th-century Old Church, the 15th-century New Church and the ...

» https://www.tripadvisor.com/Tourism-g188626-Delft_South_Holland_Provi...

Delft - Wikipedia

Delft ((listen)) is a city and municipality in the province of South Holland, Netherlands. It is located between Rotterdam, to the southeast, and The Hague, to the northwest.

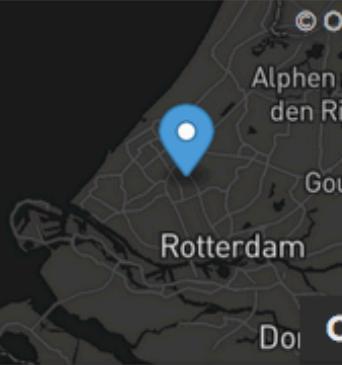
W <https://en.wikipedia.org/wiki/Delft>

Delft travel - Lonely Planet

Explore Delft holidays and discover the best time and places to visit. | An amalgam of austere medieval magnificence and Golden Age glory, Delft's exquisite town centre is a hugely popular Dutch day-trip destination, awash with visitors strolling its narrow, canal-lined streets and central Markt.

L <https://www.lonelyplanet.com/the-netherlands/the-randstad/delft>

 More Images



Delft

delft.nl

Delft is a city and municipality in the province of South Holland, Netherlands. It is located between Rotterdam, to the southeast, The Hague, to the northwest. Together with them, it is part of the Rotterdam–The Hague metropolitan area and the Randstad. More at Wikipedia

Country: Netherlands

Body: Municipal council

Mayor: Marja van Bijsterveldt (CDA)

Spode Delft Plate - Free Shipping, Orders \$99 or More AD

Free Shipping, Orders \$99 or More. Get Details and Your Pattern Replacements.com/Spode | Report Ad

CV ketel Delft Voordeelactie - Welke korting ga jij kiezen?

Alleen nog in Maart: €100 huurtegoed of gratis slimme thermos €199

Schemes: more than just http(s)

```
http://<host>:<port>/<path>?<query>#<frag>
```

```
https://<host>:<port>/<path>?<query>#<frag>
```

```
mailto:<valid-email-address>
```

```
file://<host>/<path>
```

```
file:///Users/my_home_dir/tmp.html
```

```
ftp://<user>:<passwd>@<host>:<port>/<path>;<params>
```

Relative vs. absolute URLs

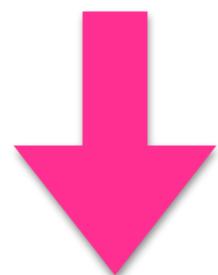
absolute

`https://www.tudelft.nl/studenten/`

```
<h1>Resources</h1>
<ol>
<li><a href="brightspace">Brightspace</a></li>
<li><a href="../disclaimer">Disclaimer</a></li>
</ol> .. parent directory
```

relative

URLs can be complex.
RFC 3986 governs conversion rules.



`https://www.tudelft.nl/studenten/brightspace`
`https://www.tudelft.nl/disclaimer`

URL design restrictions

- **Initial design goals:** *portable* across protocols and *human readable* (no invisible/non-printing characters)
- URLs initially restricted to a very small “safe” alphabet: ASCII Heavily biased in favour of English speakers.

latin alphabet, 0123456789 - _ .~
and additional reserved chars like ! () @ &
- **Added later:** character encoding, e.g. whitespace %20

Punycode (RFC 3492)

"Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It **uniquely** and **reversibly** transforms a Unicode string into an ASCII string."

`http://правительство.рф`

`=> http://xn--80aealotwbjpid2k.xn--plai`

`http://nl.wikipedia.org/wiki/Itali%C3%AB`

`=> http://nl.wikipedia.org/wiki/Itali%C3%AB`

A potential security issue in *mixed scripts*: `http://paypal.com`

Authentication

Authentication

So far: HTTP as **anonymous, stateless** request/response protocol.
The same request, sent by different clients, is treated in exactly the same manner.

Now: identification via

- A. HTTP headers
- B. Client IP address tracking
- C. Fat URLs
- D. User login (HTTP Basic Authentication)

In lecture 7: Cookies & Sessions.

User-related HTTP header fields

From	Request	User's email address	mostly web crawler
User-Agent	Request	User's browser	device customization
Referer	Request	Page the user came from	user interests
Client-IP	Request (Extension)	Client's IP address	
Authorization	Request	Username & password	

Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:69.0) Gecko/20100101 Firefox/69.0

Client IP address tracking

- Idea: **Client IP address** as user identifier
- **Several issues:**
 - A. IP addresses describe the **machine**, not the user
 - B. ISPs **dynamically assigned IP** addresses to users
 - C. Users may access the Web through **firewalls**
 - D. HTTP **proxies** and **gateways** open new TCP connections (IP of the proxy/gateway is shown), **x-Forwarded-For** might help (it quickly gets complicated)

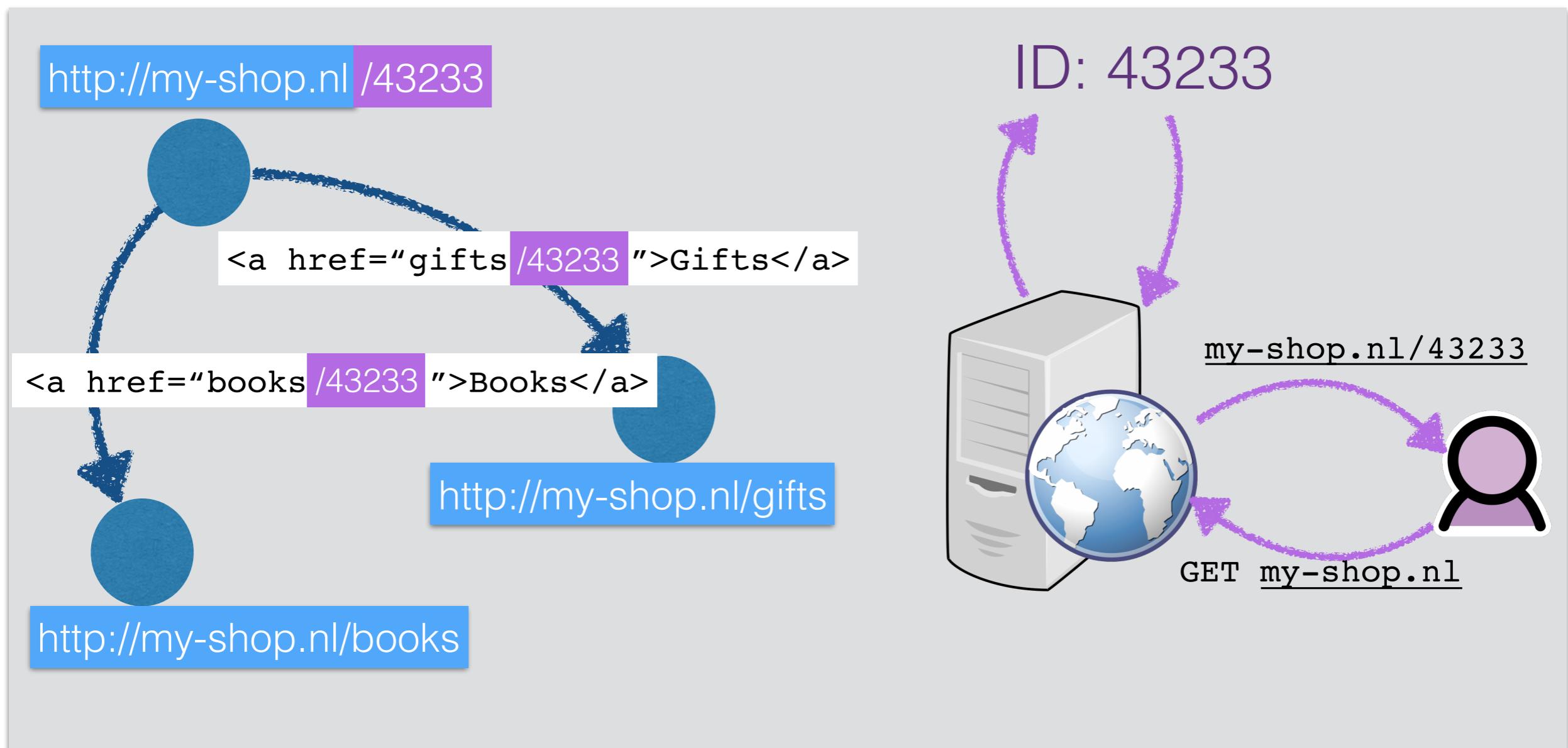
Fat URLs

- Idea: track users through the **generation of unique URLs**
 - First time a user visits a resource within a Web site, a **unique ID** is generated by the server
 - Server **redirects** client to the fat URL (URL+unique ID)
 - Server **rewrites the HTML** when an HTTP request with a fat URL is received (adds ID to all hyperlinks)

```
<a href="/browse/002-1145265-8016838">Gifts</a>
<a href="/wishlist/002-1145265-8016838">Wish List</a>
```

- Independent HTTP requests thus tied into a single session

Fat URLs



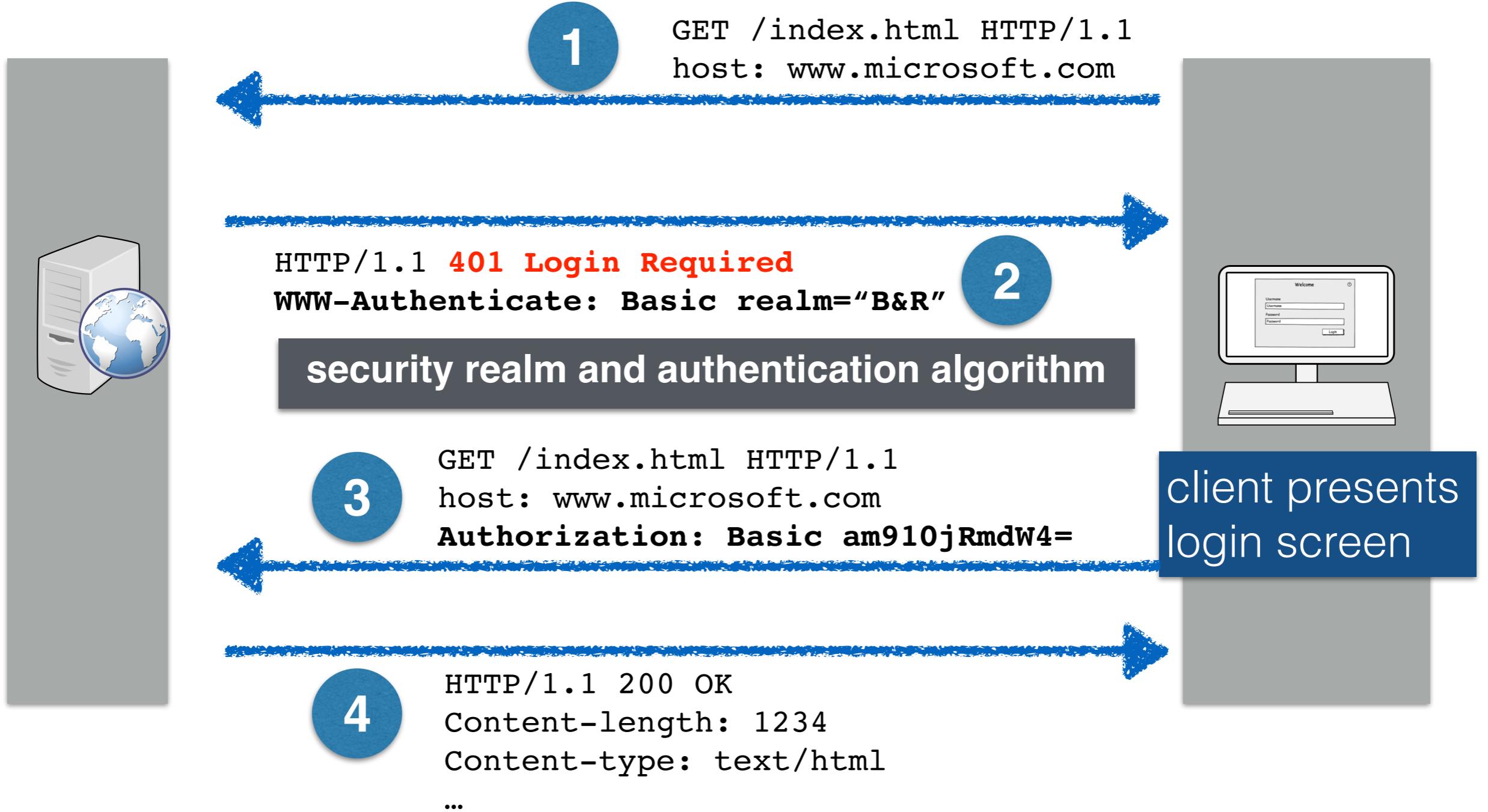
Fat URLs have issues too!

- Fat URLs are **ugly**
- Fat URLs **cannot be shared** (private info shared)
- Fat URLs **break web caching** mechanisms
- **Extra server load** (HTML page rewrites)
- Users can “**escape**” (ID is lost when user navigates away from the site)

HTTP basic authentication

- Server explicitly asks the user for authentication (**username and password**)
- HTTP has a **built-in mechanism** to support username/password based authentication via `WWW-Authenticate` and `Authorization` headers
- HTTP is **stateless**: once logged in, the client sends the login information with each request

HTTP basic authentication



In future HTTP requests to the site, the browser automatically issues the stored username/password.

HTTP basic authentication

Authorization: Basic

- Username and password are joined together by a colon and converted to **base-64 encoding** (binary-to-text encoding)
- Base-64 encoding ensures that **only HTTP compatible characters** are entered into the message (takes as input binary, text and international character data strings)

Normandië

Tm9ybWFuZGnDqw==

Delft

RGVsZnQ=

España

RXNwYcOxYQ==

HTTP basic authentication: secure?

- Username and password can be **decoded trivially** (sent over the network **in the clear**)
- **Users tend to reuse login/password combinations**; a non-critical web site may use basic authentication without SSL that an opponent can capture and try on critical sites

HTTP basic authentication: overall

Basic authentication prevents **accidental** or **casual access** by curious users (privacy is desired but not essential).

Basic authentication is useful for **personalisation** and access control within a “friendly” environment (intranet).

“In the wild”, basic authentication should always be used in combination with **secure HTTP (e.g. https)** — avoids sending username/password **in the clear** across the network.

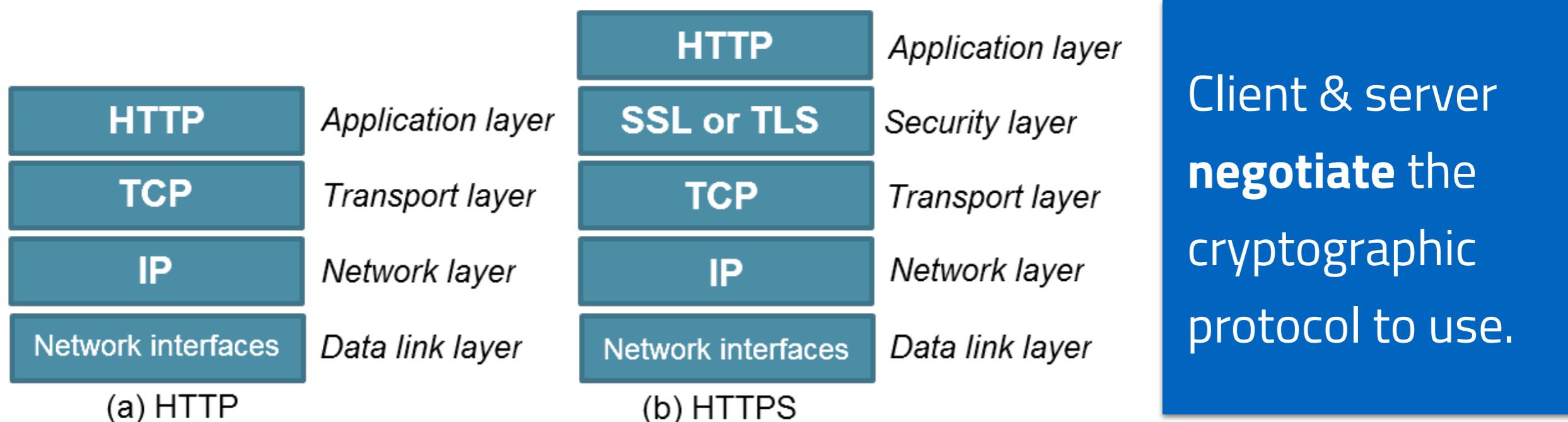
Secure HTTP

Secure HTTP

- So far: lightweight authentication
 - Not useful for purchasing, bank transactions or confidential data
- Secure HTTP should provide
 - A. Server authentication (client is sure to talk to the right server)
 - B. Client authentication (server is sure to talk to the right client)
 - C. Integrity (client and server are sure their data is intact)
 - D. Encryption
 - E. Efficiency

Secure HTTP: HTTPS

- HTTPS is the most popular secure form of HTTP
- URL scheme is `https://` instead of `http://`
- Request and response data are **encrypted** before being sent across the network (SSL: Secure Socket Layer)



- Get acquainted with the organisation of the web materials!
- Work through **Chapter 2** (intro to HTML) of the web course book **before** the next lecture!
- Start working on **Assignment 1.** 4