



香港中文大學
The Chinese University of Hong Kong

CSCI2720 - Building Web Applications

Lecture 13: Server, Client, and HTTP

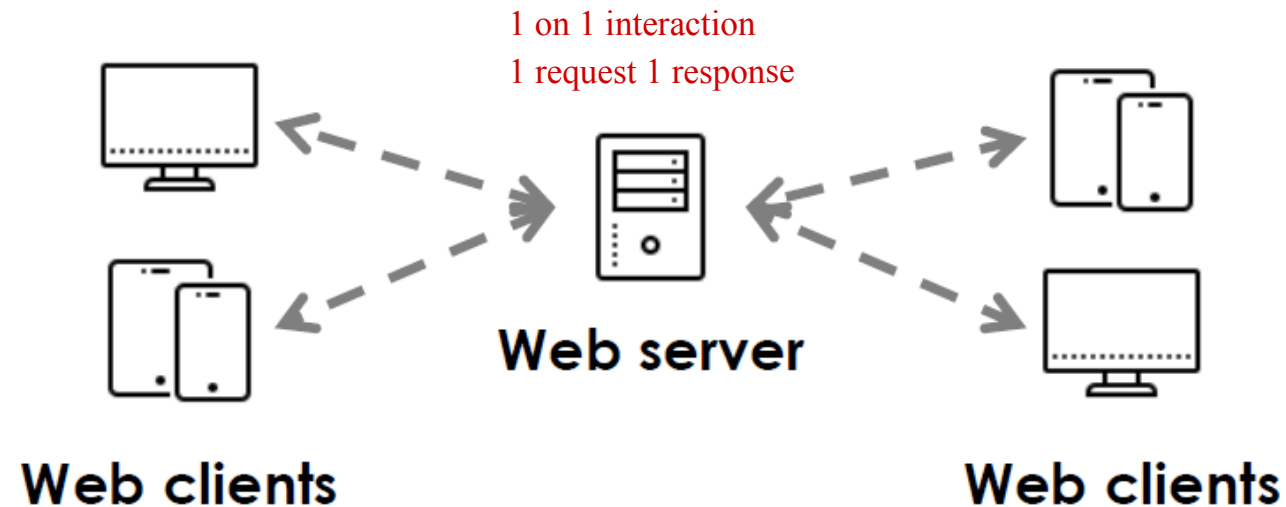
Dr Colin Tsang

Outline

- OSI model and layers
- Protocols
- MAC, IP
- TCP/UDP
- Ports
- URL and path
- Client-server, listening for connections
- Localhost
- HTTP: Client and Servers, Request and Response

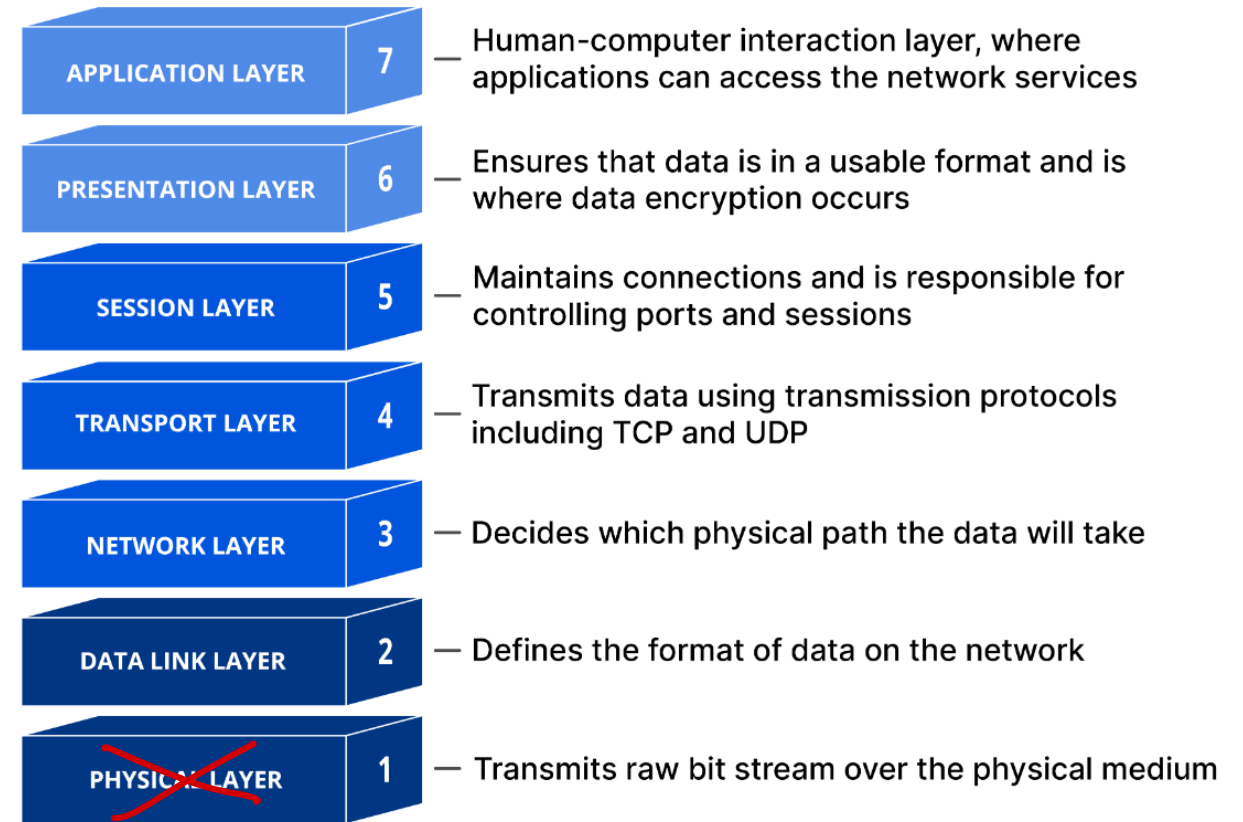
Client-Server architecture

- World Wide Web (WWW) uses the client-server architecture:
 - Clients obtain service from a centralized server
 - Server waits for client requests and make response
 - See: <https://www.britannica.com/technology/client-server-architecture>



OSI network model

- Open Systems Interconnection (OSI)
- Communication between clients and servers can be seen in multiple layers
 - Abstraction = reducing complexity of problems to smaller ones
 - Division of labour
- See: <https://www.cloudflare.com/en-gb/learning/ddos/glossary/open-systems-interconnection-model-osi/>



Communication protocols

- Clear definition of steps is needed for two computers to communicate
 - Rules
 - Syntax
 - Semantics
 - Synchronization of communication
 - Error recovery methods
- There are protocols for every layer in the networking model
- The *Internet Engineering Task Force* (IETF) develops and promotes voluntary internet standards.

Some important layers

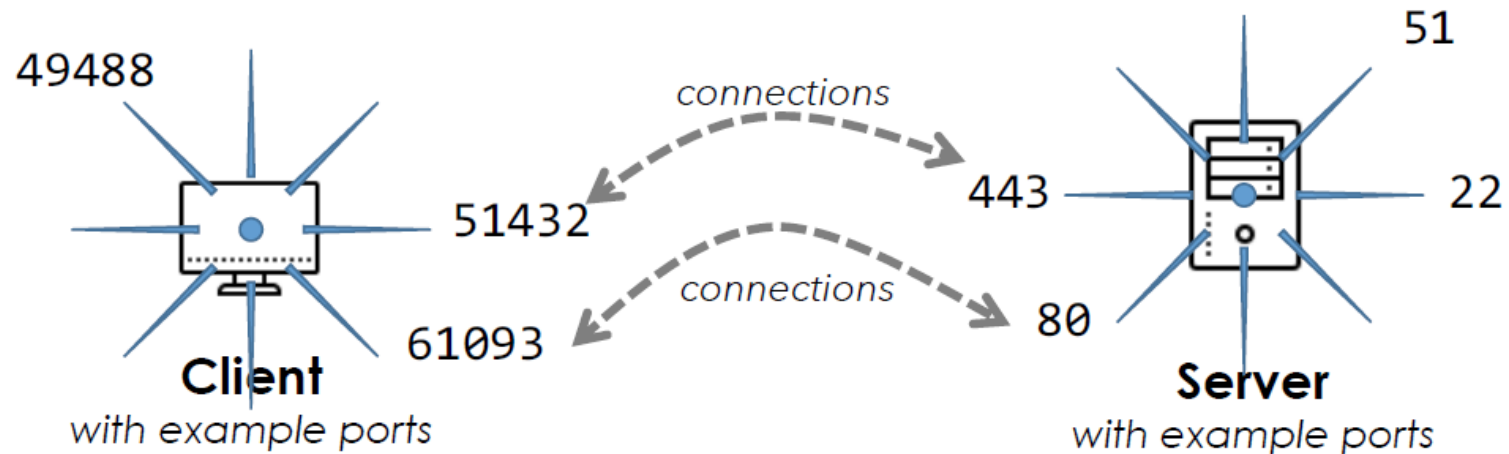
- MAC address (layer 2): locating a piece of communication device on a **local network**
- IP address (layer 3): identifying a **network interface** in networks
 - IPv4 (32-bit) vs IPv6 (128-bit)
 - Public addresses vs private addresses
 - Private addresses limit datagrams to be sent within local network only
 - E.g., 192.168.1.123 is only meaningful within a local network
- Transport (layer 4): reliability of data transmission
 - TCP (more reliable) vs UDP (faster but not guaranteed transfers)
 - Easier to recover error
 - Live streaming

Communicating over ports

- In networking, connections are made on ports of a network device.
- Each port is “listened to ” serve by one piece of software (server/client)
 - Well known ports: 0 – 1023 (HTTP: 80, HTTPS: 433)
 - Registered ports: 1024 – 49151
 - Private ports: 49152 – 65535
- See: https://www.webopedia.com/quick_ref/portnumbers.asp

Communicating over ports

The client normally use a random private port for every new connection

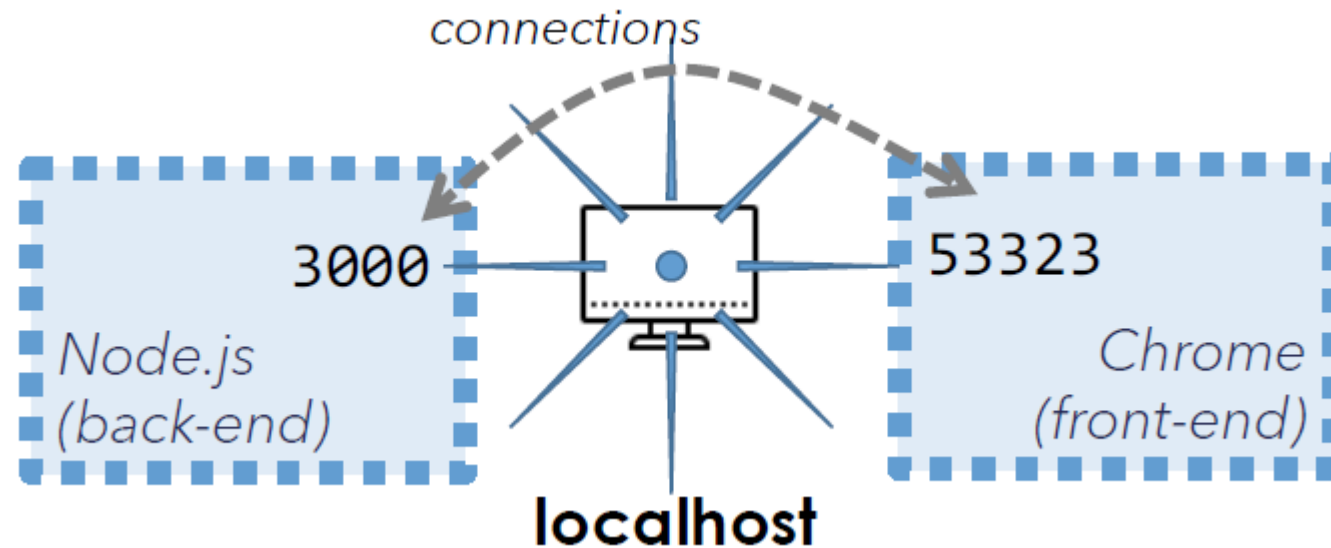


Socket

- In networking programming, a network socket is an endpoint for communication
- Socket = ^{TCP}transport protocol + ^{IP}IP address + ^{MAC}port number
- Implementation depends on the programming language/environment.

localhost

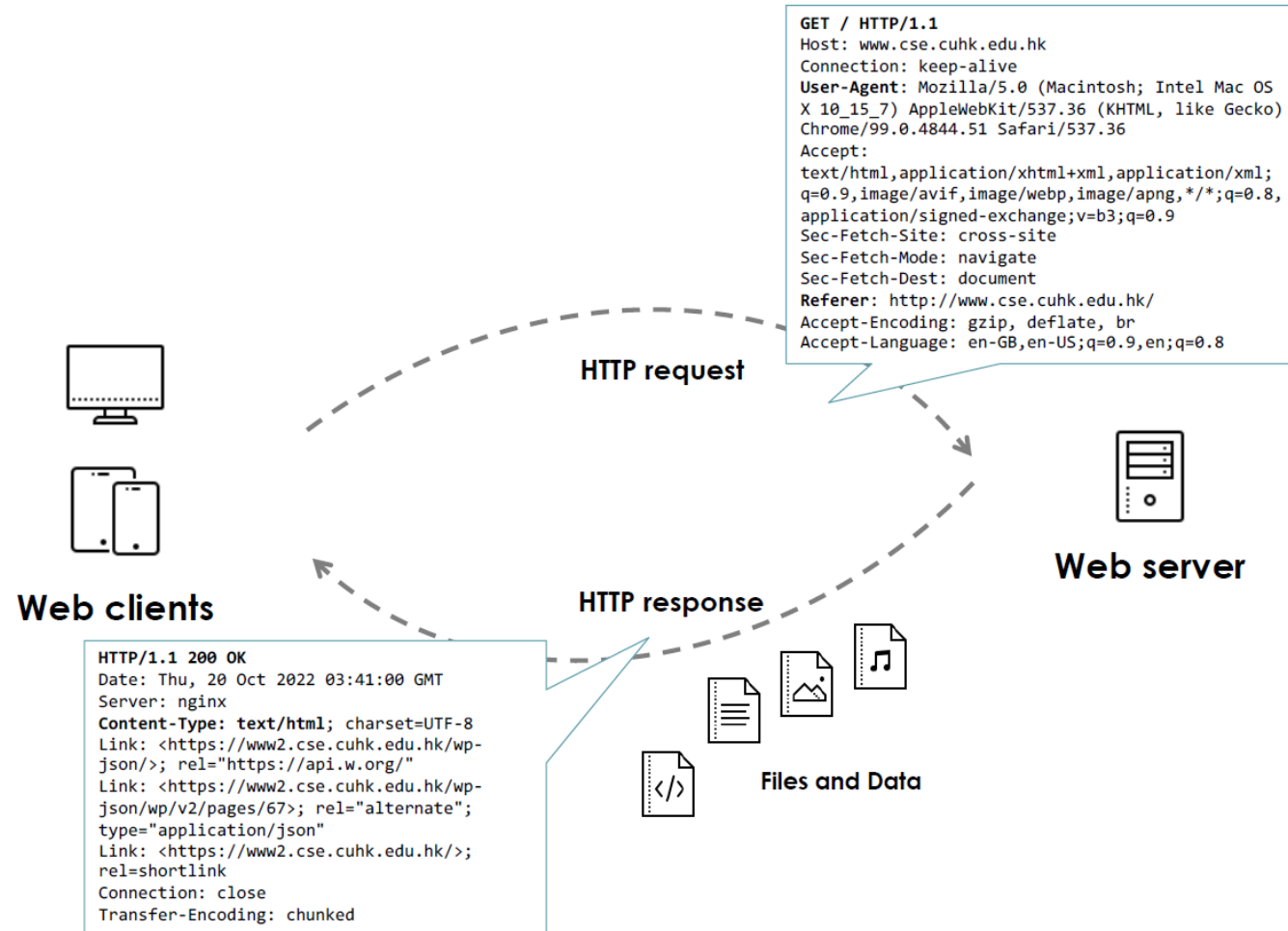
- It is also possible to have *both ends of communication only on one computer*
 - Although it is in the same computer, this way allows communication to still go through network layers in the OS
- Usually identified in the computer as *localhost* or *127.0.0.1*



HTTP

- HTTP – HyperText Transfer Protocol
 - A protocol for transferring data between a web client and a web server
- Communication is initiated by a client
 - A client sends a *HTTP request* to a server
 - The server returns a *HTTP response* to the client
 - *HTTP Server Push* (not commonly used now)
- It is *stateless*
 - Every request is treated as an independent request
 - The protocol itself does not offer any mean to relate two separate requests
- HTTP/1.1, HTTP/2, HTTP/3
 - Different version co-exist

HTTP Connections



HTTP Clients

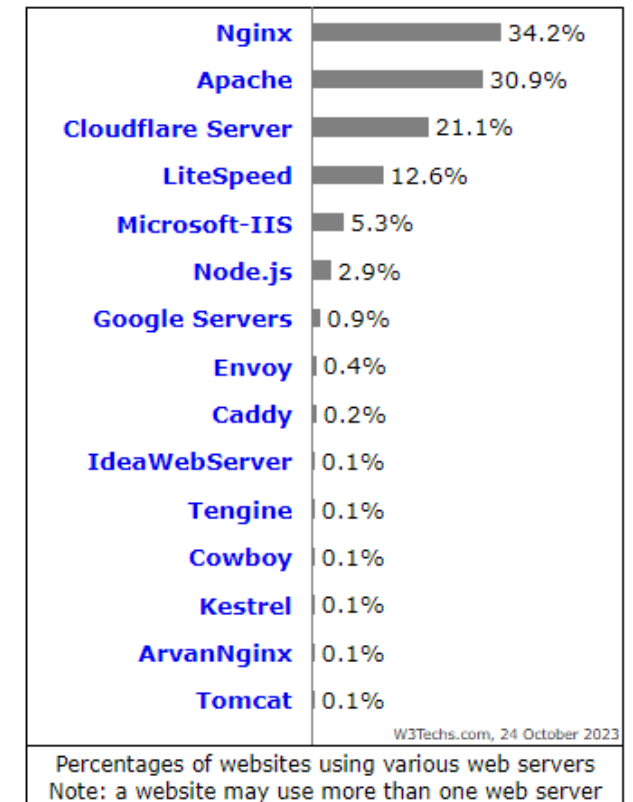
- Web client (“user-agent”): a software communicating with the web server over HTTP or HTTPS
 - Web browsers: for end-user experience
 - HTML/CSS rendering
 - JavaScript execution
 - Web technologies
 - Web crawlers / search engines
 - Programmatic interfaces
 - E.g., Postman (See: <https://www.postman.com/>)

				
	Google Chrome	Apple Safari	Mozilla Firefox	Microsoft Edge
Browser Engine	Blink	WebKit	Gecko	Blink
JavaScript Engine	V8	JavaScriptCore	SpiderMonkey	V8

https://www.youtube.com/watch?v=H52DmvfzDWM&ab_channel=JamesEagle

HTTP Servers

- Web server: handling requests for web documents with HTTP/HTTPS, supporting server-side scripts
 - Apache (was still the 1st two years ago)
 - Nginx
 - Microsoft IIS
 - Node.js
- Web caching
 - Forward/reverse proxies
 - Content delivery networks (CDN)
 - E.g., Cloudflare
- See: https://w3techs.com/technologies/overview/web_server



Uniform Resource Locator (URL)

<https://www.cuhk.edu.hk/english/index.html>

Protocol

Hostname

Path

Filename

- A web resource can be identified and located using the above information
- “Really, if you think about it, it doesn't need the slash slash. People are having to use that finger so much ... not to mention the human labour and time spent typing those two keystrokes countless millions of times in browser address boxes.” *Sir Tim Berners Lee*

HTTP Request

HTTP Method	Request URI	HTTP Version	
POST /cuhk/csci2720/new HTTP/1.1			Request Line
Host: localhost User-Agent: Mozilla/5.0 ... Accept: ... Accept-Language: en-us,en;q=0.5 Accept-Encoding: gzip,deflate Accept-Charset: ... Keep-Alive: 115 Connection: keep-alive Referer: http://localhost/... Content-Type: application/x-www-form-urlencoded Content-Length: 13			Headers
name=Student+One			Body

HTTP Methods

- **GET** and **POST** are the most used methods
- **GET** method is used when
 - A user clicks on a link, select a bookmarked URL, enter the URL in browser location bar
 - Browser retrieves files (images, CSS, etc) needed in a HTML document
- Form submission – either **GET** or **POST**
 - For sending data back to server
 - We also use fetch with **PUT** in assignment 1
- **DELETE**, **HEAD**, ..., etc

GET Vs. POST

GET	POST
Data only delivered <i>inside the request URL</i> in text form	Data can be encapsulated inside request body
Only limited amount of data (~2k) due to URL length	Data size is only <i>limited by the body size</i> (~1MB to 2GB) depending on the HTTP server
The request URL can be bookmarked, and is visible in the location bar ➔ security concern!	Only URL can be saved but not the data in the body
The request URL would stay in the browser history, and can also be found on HTTP server log ➔ security concern!	Only URL is recorded but not the data in the body

Request URL

- A file path-like string in the form
 - `/path`
- A query sting typically contains URL-encoded name-value pairs
 - Most web frameworks support URL encoding/decoding
 - The exact structure of the query string is not standardised
 - `/path?query`
- e.g.,
 - `/main/index.html`
 - `/foo/bar/index?123456789`
 - `/foo/bar/index?name1=value1&name2=value2`

Headers in an HTTP Request

- Contain multiple header fields
- Each header field has a name and value
- For a complete list of HTTP headers
 - See: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>

Headers in an HTTP Request *from client*

- **User-Agent**: *Browser* data about the web client and the client's operating system
- **Referer**: the URL of the webpage that brings the client to the requested page
- **Cookie**: cookies
- **Content-Type**: indicates the media type of the resource in the body
 - In a request, it typically indicates how the data are encoded in the body
- **Accept, Accept-Encoding, Accept-Language, Accept-Charset**:
 - Content types, compression schemes, languages, and character sets that the web client handle (for content negotiation)

*html
img*

HTTP Request Body

- Carries data when request method is **POST** or **PUT**
- The data in the body can be encoded in various format
 - The encoding scheme is indicated by the header **Content-Type**.

HTTP Response

Status code

If there is error (eg 404), there is no body information but still haeder

Version

HTTP/1.1 200 OK

Status Line

Date: Mon, 7 Mar 2022 03:08:29 GMT
Server: nginx
Content-Length: 235
Connection: Keep-Alive
Content-Type: text/html

Headers

<html>
...
</html>

Body

Data in an HTTP Response

- Status Line
 - Status code and Reason-Phrase
 - E.g., 404 Not Found
 - 100-199: Informational responses
 - 200-299: Successful responses
 - 300-399: Redirects
 - 400-499: Client errors
 - 500-599: Server errors
 - See: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>
 - Body:
 - Containing the requested resources, which can be
 - A static or dynamically generated file
 - Data encoded in some encoding scheme
- html file only May receive information from client, by key:value

Headers in an HTTP Response

- Location: Indicates the URL to redirect a page
 - *Location: login.php*
- **Cache-Control, Expires, ETag**: Caching-related headers, e.g.,
 - *Cache-Control: no cache, must-revalidate*
- **Set-Cookie**: Send cookies to a client
 - Most framework offers API for setting and retrieving cookies

Headers in an HTTP Response

Static

- **Content-Type:** Indicates the media type of the resource in the body
 - *Content-Type: text/plain*
 - *Content-Type: text/html*
 - *Content-Type: text/csv*
 - A script can dynamically create text file, CSV file, image, etc
- **Content-Disposition:** Request a web client to save (instead of display) the content
 - *Content-Disposition: attachment; filename="image.jpg"*

More terms in Web

- Client Vs. Server
 - Client-side rendering
 - Server-side rendering
- Front-end Vs. Back-end
 - Presentation
 - HTML/CSS/JS
 - Data access
 - Data store, cloud logic
 - Authentication



- See: <https://www.system-concepts.com/insights/service-blueprint/>

Further readings

- MDN tutorial on HTTP:
 - <https://developer.mozilla.org/en-US/docs/Web/HTTP>