網頁程式設計
Web Programming

Ch1: Introduction (WWW and HTTP)

111-2-r1

Outline

- Why Study Web Programming?
- World Wide Web
- Hypertext Transfer Protocol

REF: 新觀念PHP7+MySQL+AJAX網頁設計範例教本(第五版)，旗標。

REF: 輕鬆玩HTML5+CSS3+JavaScript網頁程式設計，全華。

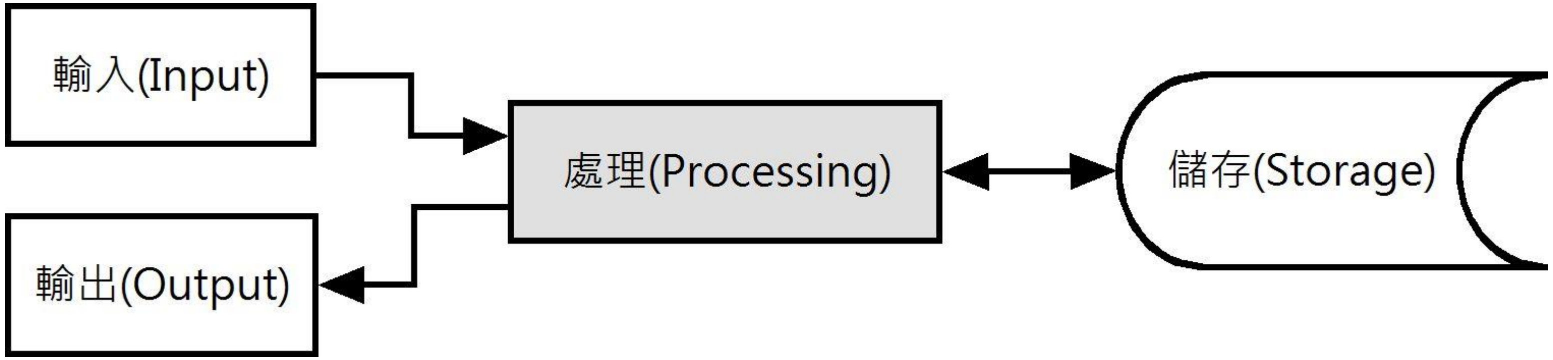
Why Study Web Programming?

- A web application basically includes user-side (presenting text, figure, video, gaming…) and server-side (processing actions) these two parts.
- In the past, web application can only be running at the server-side, presenting its exportation results on the client-side browser (now it can be mutual).
- Most web applications aim to response client requests and make interaction. For instance, Facebook, Plurk, and Twitter are providing such scenario.
- You may use C/C++, PHP, Perl, Python, Ruby, and other web programming languages to develop the web application.

Why Study Web Programming? (cont.)

- Web application is one of Information Processing Systems, with input, output, and processing actions.

- The invention of web has improved the information dissemination in the world.



Why Study Web Programming? (cont.)

- Basic elements for enabling the web service:

- Webpage content (text, figure, video…)
- Web application (collaborate with database, accounting system…)
  - Programming languages (like Ruby, JavaScript…)
  - Web frameworks (like Ruby on Rails, NodeJS…)
- Webserver software (Apache, IIS, Tomcat, Nginx…)
- Webhosting (PC, Server, Virtual Machine, VPC w/ cPanel…)

- Presenting to the client → front-end
- Others → back-end

(前後端簡略的區分方式)

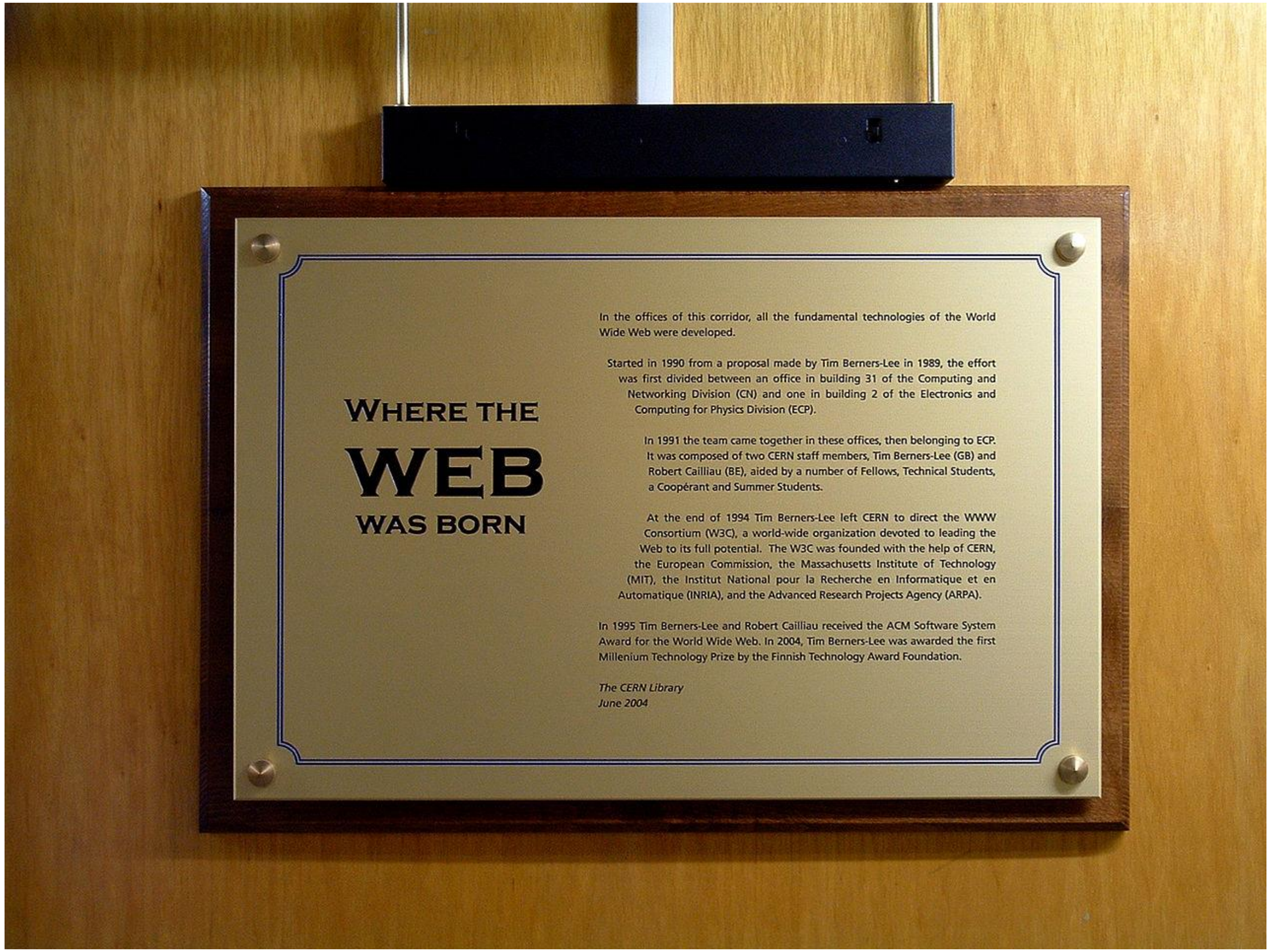
World Wide Web

- World Wide Web (WWW, 全球資訊網), called **web** in short, is an information system that invented by research team of CERN (歐洲核子研究組織), leading by Sir Timothy Berners-Lee in 1989.

- In WWW, it uses Uniform Resource Locator (**URL**, 統一資源定位符) and Uniform Resource Identifier (**URI**, 統一資源識別碼) to identified available resources, and these resources can be inter-linked by **hyperlink** (超鏈結).

REF: https://zh.wikipedia.org/wiki/%E4%B8%87%E7%BB%B4%E7%BD%91#/media/File:WWW_logo_by_Robert_Cailliau.svg

World Wide Web (cont.)



WHERE THE

WEB

WAS BORN

In the offices of this corridor, all the fundamental technologies of the World Wide Web were developed.

Started in 1990 from a proposal made by Tim Berners-Lee in 1989, the effort was first divided between an office in building 31 of the Computing and Networking Division (CN) and one in building 2 of the Electronics and Computing for Physics Division (ECP).

In 1991 the team came together in these offices, then belonging to ECP. It was composed of two CERN staff members, Tim Berners-Lee (GB) and Robert Cailliau (BE), aided by a number of Fellows, Technical Students, a Coopérant and Summer Students.

At the end of 1994 Tim Berners-Lee left CERN to direct the WWW Consortium (W3C), a world-wide organization devoted to leading the Web to its full potential. The W3C was founded with the help of CERN, the European Commission, the Massachusetts Institute of Technology (MIT), the Institut National pour la Recherche en Informatique et en Automatique (INRIA), and the Advanced Research Projects Agency (ARPA).

In 1995 Tim Berners-Lee and Robert Cailliau received the ACM Software System Award for the World Wide Web. In 2004, Tim Berners-Lee was awarded the first Millenium Technology Prize by the Finnish Technology Award Foundation.

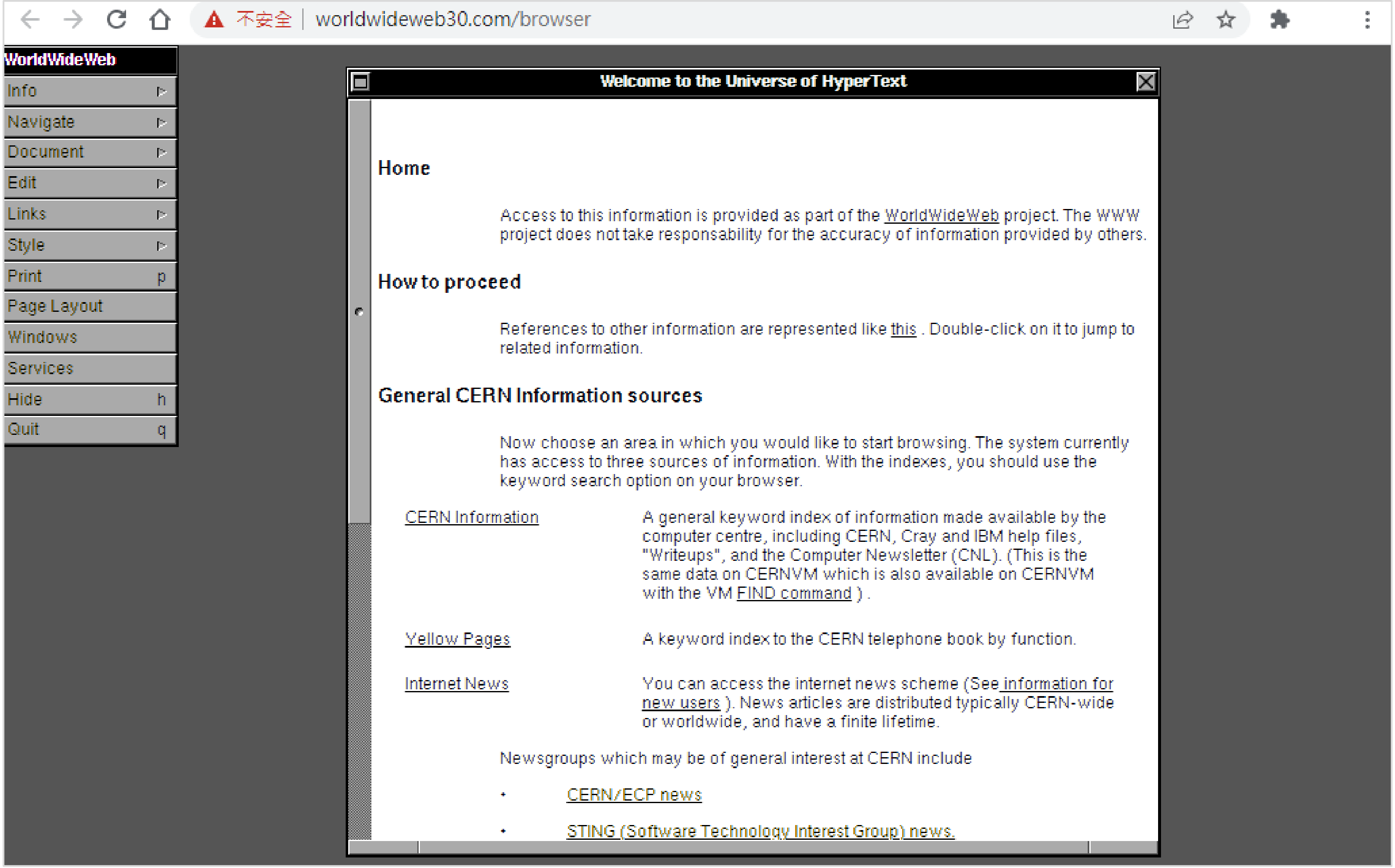
The CERN Library
June 2004

REF: https://zh.wikipedia.org/wiki/File:Where_the_WEB_was_born.jpg

World's First Web Server (at CERN)

REF: https://en.wikipedia.org/wiki/CERN_httpd#/media/File:First_Web_Server.jpg

CERN 2019 WorldWideWeb 30th Rebuild



Home

Access to this information is provided as part of the WorldWideWeb project. The WWW project does not take responsability for the accuracy of information provided by others.

How to proceed

References to other information are represented like this . Double-click on it to jump to related information.

General CERN Information sources

Now choose an area in which you would like to start browsing. The system currently has access to three sources of information. With the indexes, you should use the keyword search option on your browser.

CERN Information	A general keyword index of information made available by the computer centre, including CERN, Cray and IBM help files, "Writeups", and the Computer Newsletter (CNL). (This is the same data on CERNVM which is also available on CERNVM with the VM FIND command) .
Yellow Pages	A keyword index to the CERN telephone book by function.
Internet News	You can access the internet news scheme (See information for new users). News articles are distributed typically CERN-wide or worldwide, and have a finite lifetime.

Newsgroups which may be of general interest at CERN include

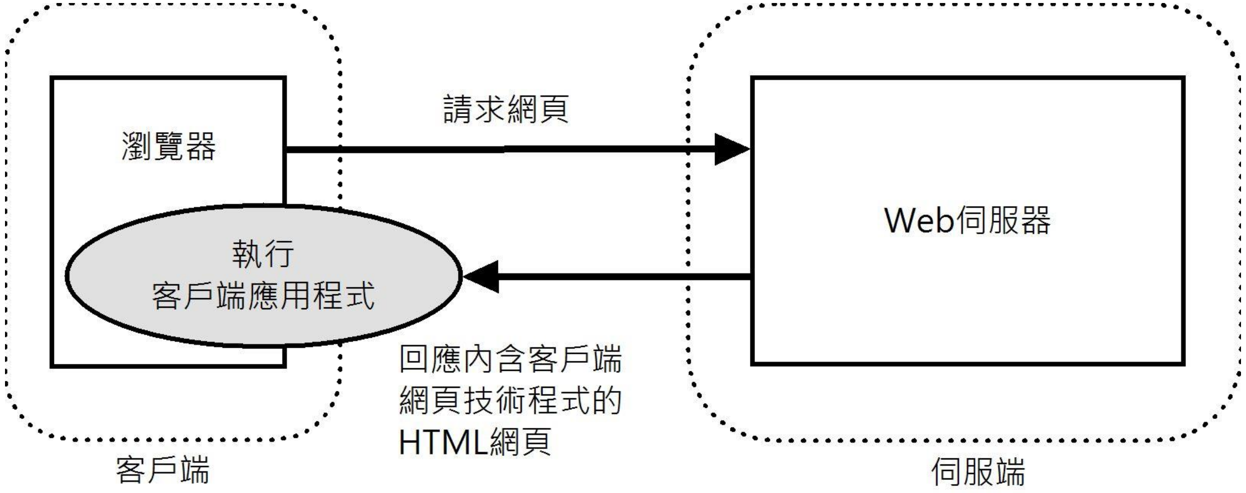
- CERN/ECP news
- STING (Software Technology Interest Group) news.

REF: https://worldwideweb30.com/

World Wide Web (cont.)

- **Hypertext Markup Language** (HTML, 超文本標記語言) is a standard language used to create webpage. A webpage may involve html and additional information (e.g., Cascading Style Sheets/CSS and JavaScript/JS) to present its content.

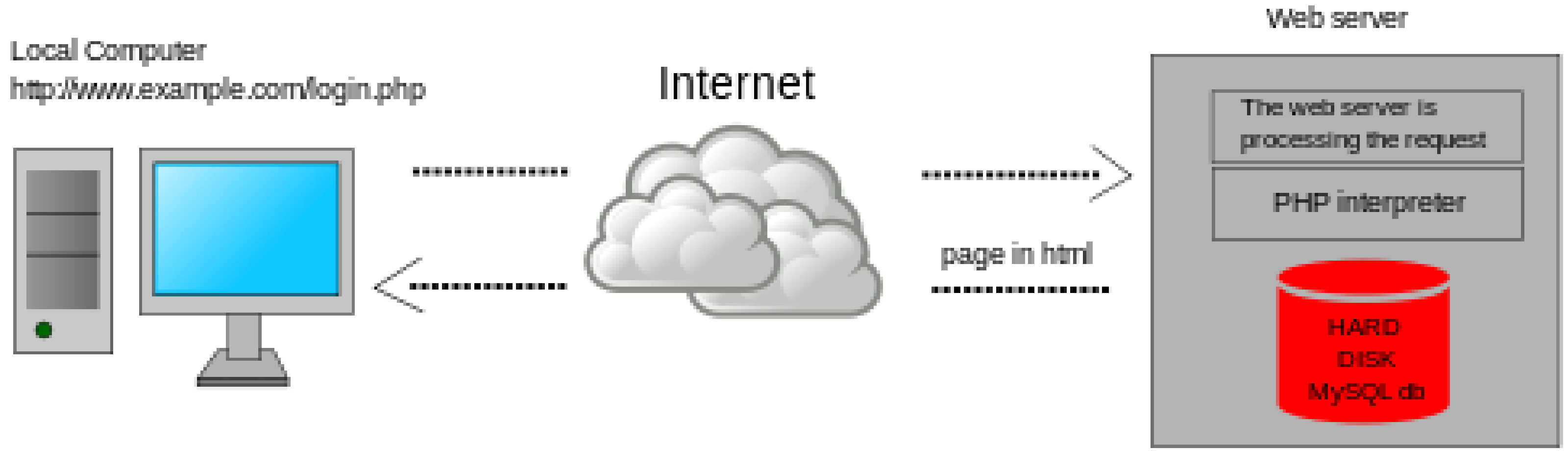
- The resources on web are transferred via the **Hypertext Transfer Protocol** (HTTP, 超文本傳輸協定). By using **web browser** (網頁瀏覽器), user are able to access web resources through the Internet.



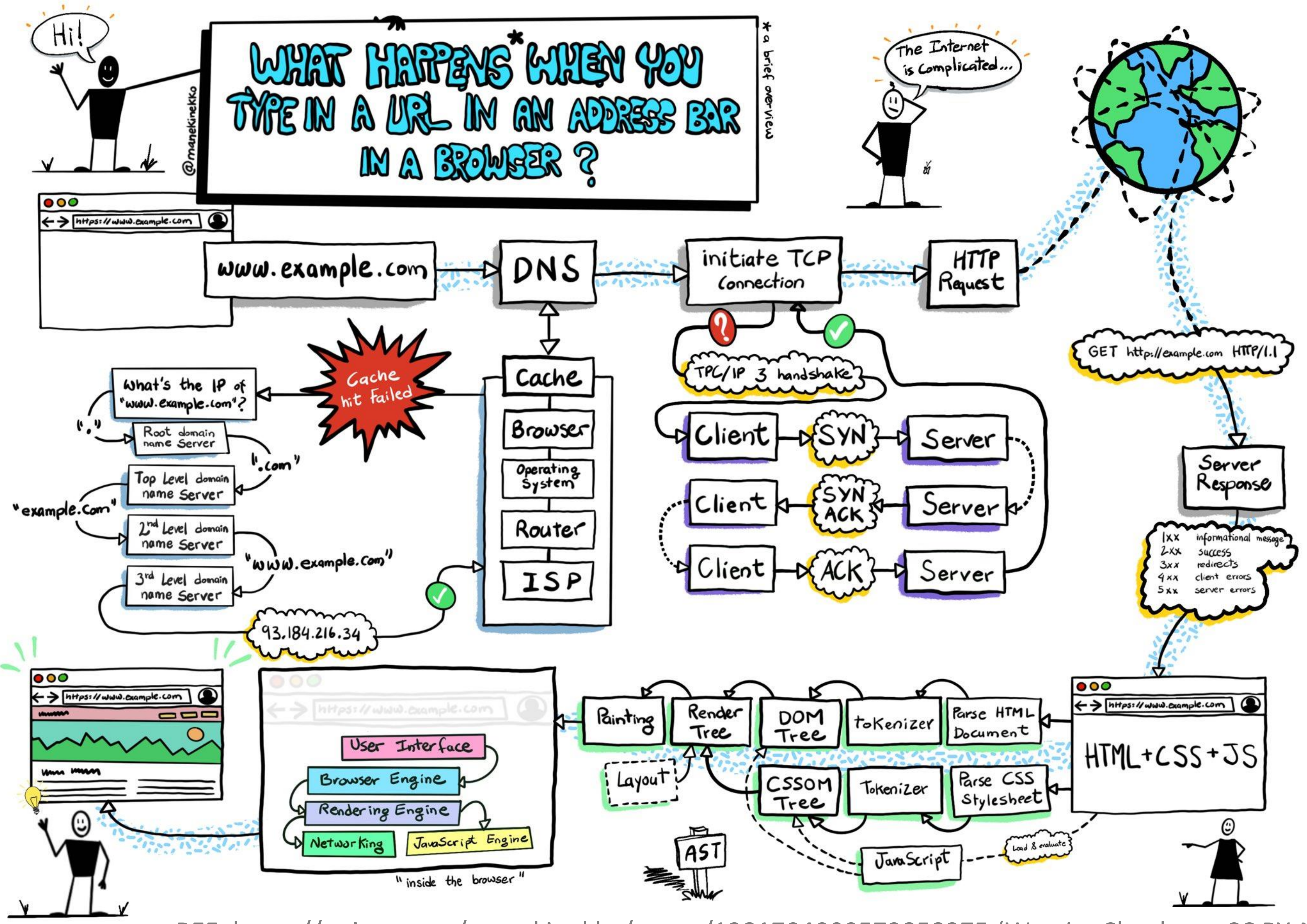
World Wide Web (cont.)

- Website aggregates resources in web server, enabling web service—a daemon that responds to requests made over the Internet from web client(s).

- Web server and client leverages HTTP to make communication, which are application-level protocols in the layer-7 of Open Systems Interconnection (OSI) reference model.

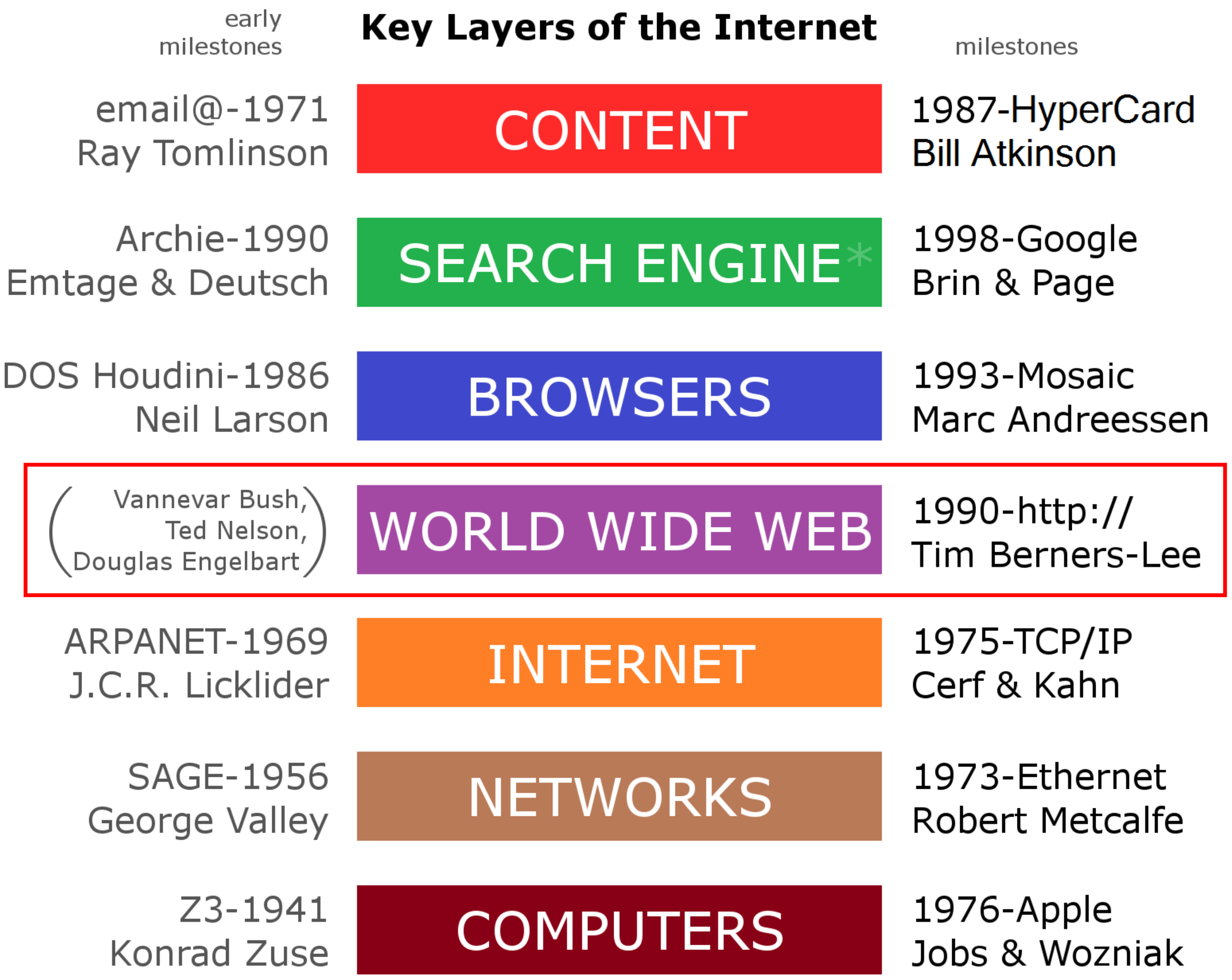


REF: https://en.wikipedia.org/wiki/File:Scheme_dynamic_page_en.svg



REF: https://twitter.com/manekinekko/status/1281704000572858375 (Wassim Chegham, CC BY-NC-SA 2.0)

World Wide Web (cont.)

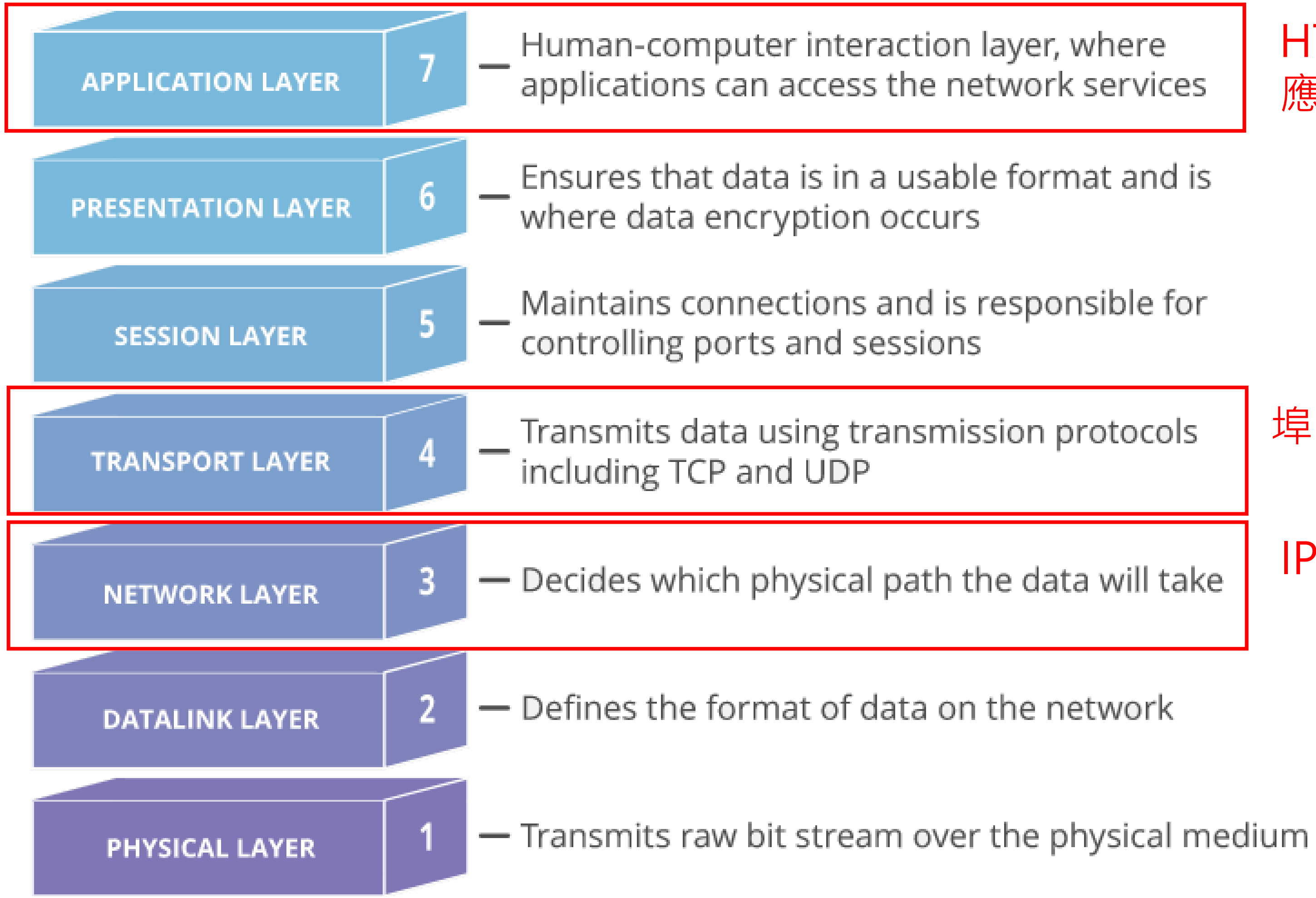


Key Layers of the Internet

early milestones		milestones
email@-1971 Ray Tomlinson	CONTENT	1987-HyperCard Bill Atkinson
Archie-1990 Emtage & Deutsch	SEARCH ENGINE*	1998-Google Brin & Page
DOS Houdini-1986 Neil Larson	BROWSERS	1993-Mosaic Marc Andreessen
(Vannevar Bush, Ted Nelson, Douglas Engelbart)	WORLD WIDE WEB	1990-http:// Tim Berners-Lee
ARPANET-1969 J.C.R. Licklider	INTERNET	1975-TCP/IP Cerf & Kahn
SAGE-1956 George Valley	NETWORKS	1973-Ethernet Robert Metcalfe
Z3-1941 Konrad Zuse	COMPUTERS	1976-Apple Jobs & Wozniak

REF: https://upload.wikimedia.org/wikipedia/commons/3/39/Internet_Key_Layers.png

Open Systems Interconnection (OSI) Reference Model



APPLICATION LAYER 7 — Human-computer interaction layer, where applications can access the network services

HTTP 屬於 Layer-7 的應用層協定

PRESENTATION LAYER 6 — Ensures that data is in a usable format and is where data encryption occurs

SESSION LAYER 5 — Maintains connections and is responsible for controlling ports and sessions

TRANSPORT LAYER 4 — Transmits data using transmission protocols including TCP and UDP

埠號 (port number)

NETWORK LAYER 3 — Decides which physical path the data will take

IP 位址 (IP address)

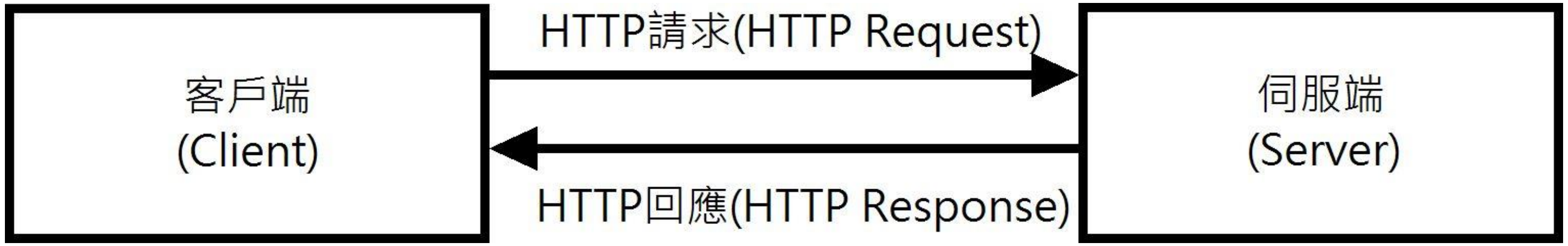
DATALINK LAYER 2 — Defines the format of data on the network

PHYSICAL LAYER 1 — Transmits raw bit stream over the physical medium

REF: https://www.cloudflare.com/zh-tw/learning/ddos/what-is-layer-7/

Hypertext Transfer Protocol

- HTTP is the general way of making data communication to transmit web resources through the network.
- A simple interaction between HTTP server and client is shown below:



Hypertext Transfer Protocol (cont.)

- HTTP Requests for Comments (RFCs) has became a standard of Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C). The first documented version of HTTP was HTTP/0.9 in 1991.

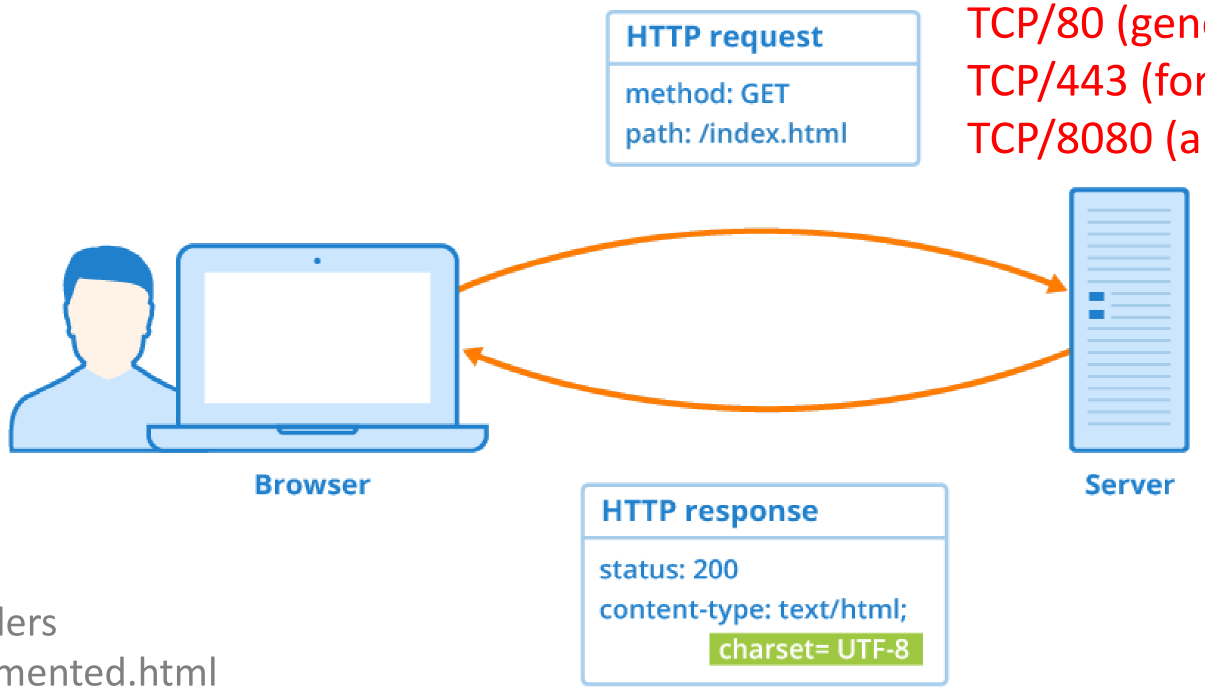
- Dave Raggett led the HTTP Working Group (HTTP WG) in 1995 and wanted to expand the protocol. The initial work was to document existing practice and short-term extensions. Subsequent work was to extend and revise the protocol.

- The WG directions include improving efficiency, extending operations & negotiation, getting richer meta-information, and adding security concepts.

REF: https://www.w3.org/Arena/webworld/httpwgcharter.html

Hypertext Transfer Protocol (cont.)

- Connection
  - The client makes a TCP-IP connection to the host using the domain name or IP address, and the port number given in the address.
  - If the port number is not specified, 80 is always assumed for HTTP.
  - The server accepts the connection.



default server port:
TCP/80 (general)
TCP/443 (for HTTPS)
TCP/8080 (alternative)

default client port:
a large random number

REF: https://www.seobility.net/en/wiki/HTTP_headers
REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

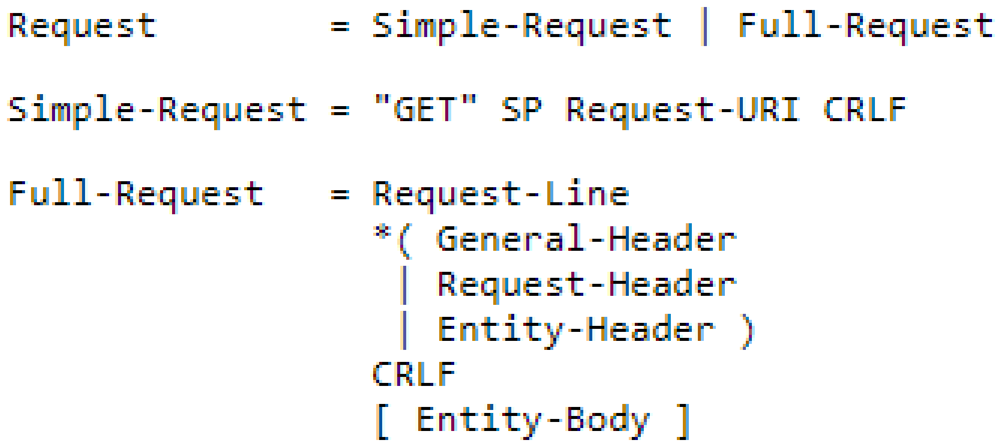
Hypertext Transfer Protocol (cont.)

- Request
  - The client sends a document request consisting of a line of ASCII characters. A well-behaved server will not require the carriage return character.
  - This request consists of the word "GET", a space, the document address , omitting the "http:, host and port parts when they are the coordinates just used to make the connection.
  - The document address will consist of a single word.
  - The search functionality of the protocol lies in the ability of the addressing syntax to describe a search on a named index .
  - A search should only be requested by a client when the index document itself has been described as an index with tag .

REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

Hypertext Transfer Protocol (cont.)

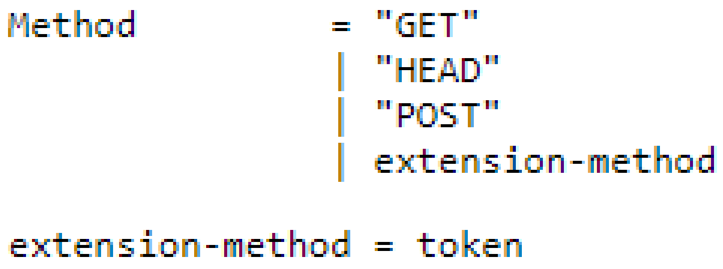
- A **request message** from a client to a server, which includes the type of that message, the method used to be applied for the resource, the identifier of the resource, and the protocol version in use.



```
Request        = Simple-Request | Full-Request

Simple-Request = "GET" SP Request-URI CRLF

Full-Request   = Request-Line
                 *( General-Header
                  | Request-Header
                  | Entity-Header )
                 CRLF
                 [ Entity-Body ]
```

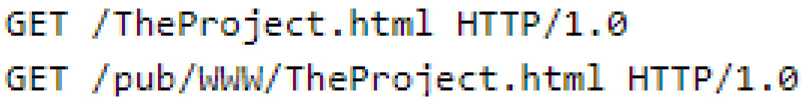


```
Method         = "GET"
               | "HEAD"
               | "POST"
               | extension-method

extension-method = token
```

REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

Hypertext Transfer Protocol (cont.)

- GET (用於取得資料)
  - The GET method means retrieve whatever information (in the form of an entity) is identified by the Request-URI.
  - 

```
GET /TheProject.html HTTP/1.0
GET /pub/WWW/TheProject.html HTTP/1.0
```

- HEAD (類似 GET 但只需要使用標頭回應, 例如用於判斷資源是否存在)
  - The HEAD method is identical to GET except that the server must **not return any Entity-Body** in the response.
  - The meta information contained in the HTTP headers in response to a HEAD request should be identical to the information sent in response to a GET request.

REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

Hypertext Transfer Protocol (cont.)

- POST (上傳資料, 資料存放位置和如何接收處理由伺服器端決定)
  - The POST method is used to request that the destination server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI in the Request-Line.
  - POST is designed to allow a uniform method to cover the following functions:
    - Annotation of existing resources
    - Posting a message to a bulletin board, newsgroup, mailing list, or similar group of articles
    - Providing a block of data, such as the result of submitting a form, to a data-handling process
    - Extending a database through an append operation
- PUT, DELETE, TRACE, OPTIONS, CONNECT (also defined in RFC 7231)
- PATCH (defined in RFC 5789)

REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

Hypertext Transfer Protocol (cont.)

- **Response**
  - The response to a simple GET request is a message in hypertext mark-up language (HTML).
  - It is a byte stream of ASCII characters.
  - Lines shall be delimited by an optional carriage return followed by a mandatory line feed character. The client should not assume that the carriage return will be present.
  - The format of the message is HTML. This format allows for menus and hit lists to be returned as hypertext. It also allows for plain ASCII text to be returned following the PLAINTEXT tag .
  - The message is terminated by the closing of the connection by the server.
  - Well-behaved web client will read the entire document as fast as possible. Web client should not wait for human-user's action before reading the whole of the document.
  - The server may impose a timeout of the order on inactivity.
  - Error responses are supplied in human readable text in HTML syntax.

REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

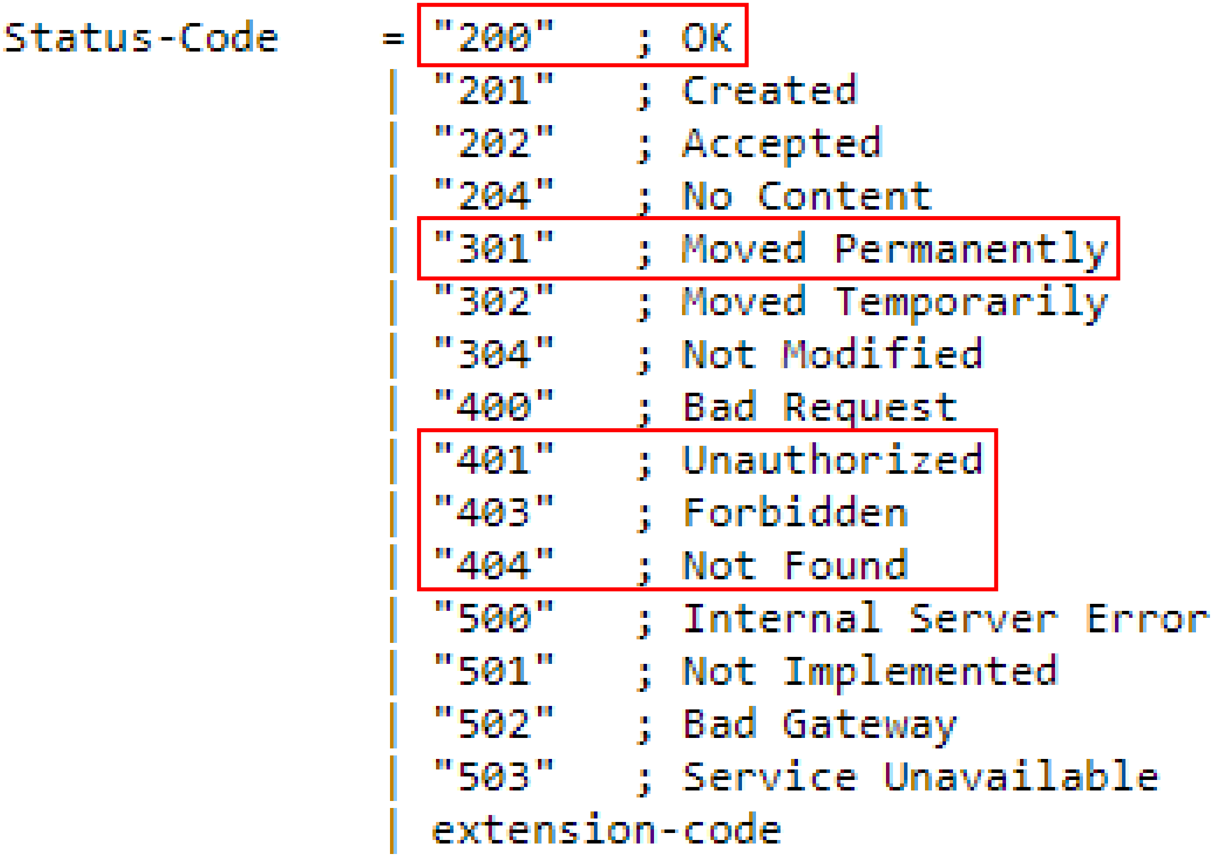
Hypertext Transfer Protocol (cont.)

- Status-Code in **response message**
  - The Status-Code element is a 3-digit integer result code of the attempt to understand and satisfy the request. Web client is not required to examine or display the Reason-Phrase for human-user.

- 1xx: Informational - Not used, but reserved for future use
- 2xx: Success - The action was successfully received, understood, and accepted.
- 3xx: Redirection - Further action must be taken in order to complete the request
- 4xx: Client Error - The request contains bad syntax or cannot be fulfilled
- 5xx: Server Error - The server failed to fulfill an apparently valid request

REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

Hypertext Transfer Protocol (cont.)

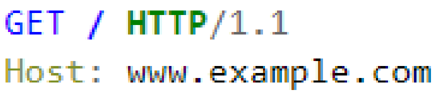


```
Status-Code    = "200"   ; OK
               | "201"   ; Created
               | "202"   ; Accepted
               | "204"   ; No Content
               | "301"   ; Moved Permanently
               | "302"   ; Moved Temporarily
               | "304"   ; Not Modified
               | "400"   ; Bad Request
               | "401"   ; Unauthorized
               | "403"   ; Forbidden
               | "404"   ; Not Found
               | "500"   ; Internal Server Error
               | "501"   ; Not Implemented
               | "502"   ; Bad Gateway
               | "503"   ; Service Unavailable
               | extension-code
```

REF: https://www.w3.org/Protocols/HTTP/1.0/spec.html

Hypertext Transfer Protocol (cont.)

- Web Client (request)

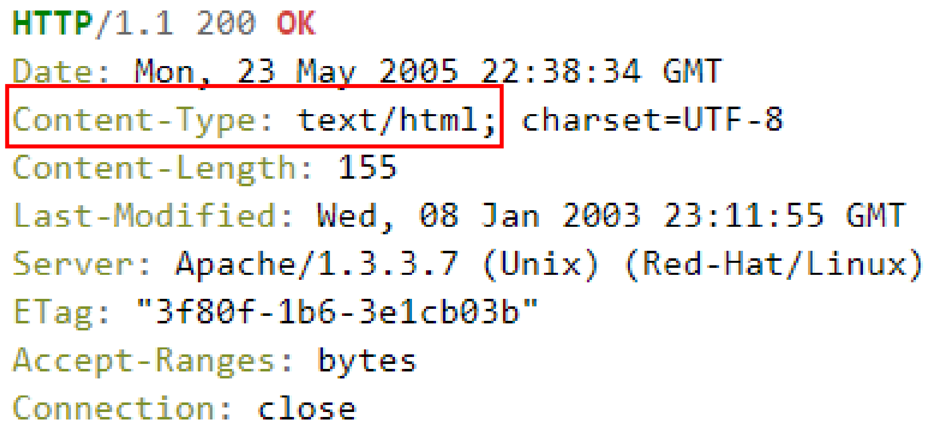


```
GET / HTTP/1.1
Host: www.example.com
```

網址會先透過 DNS 機制取得解析後的 IP 位址，若此值為 IP 地址則不需要解析

- Web Server (response)

狀態碼 →

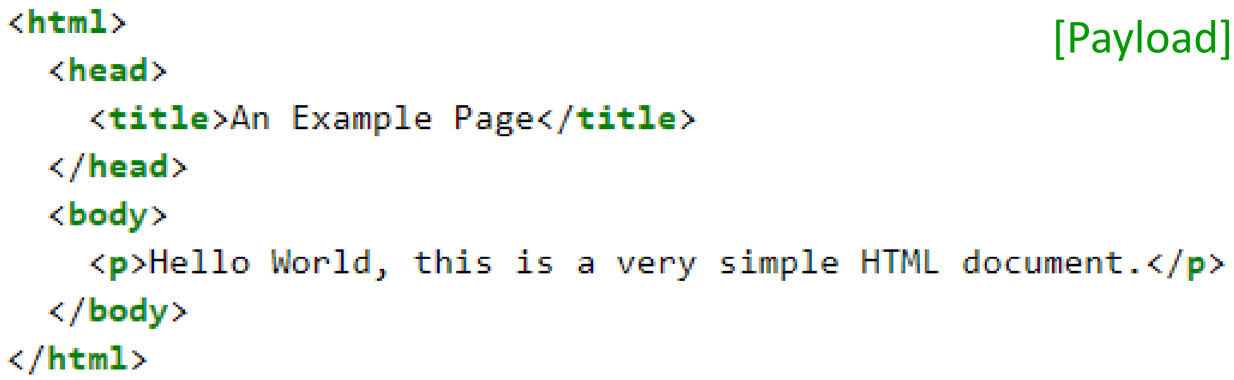


```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 155
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
ETag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Connection: close
```

[Header]

[Payload]

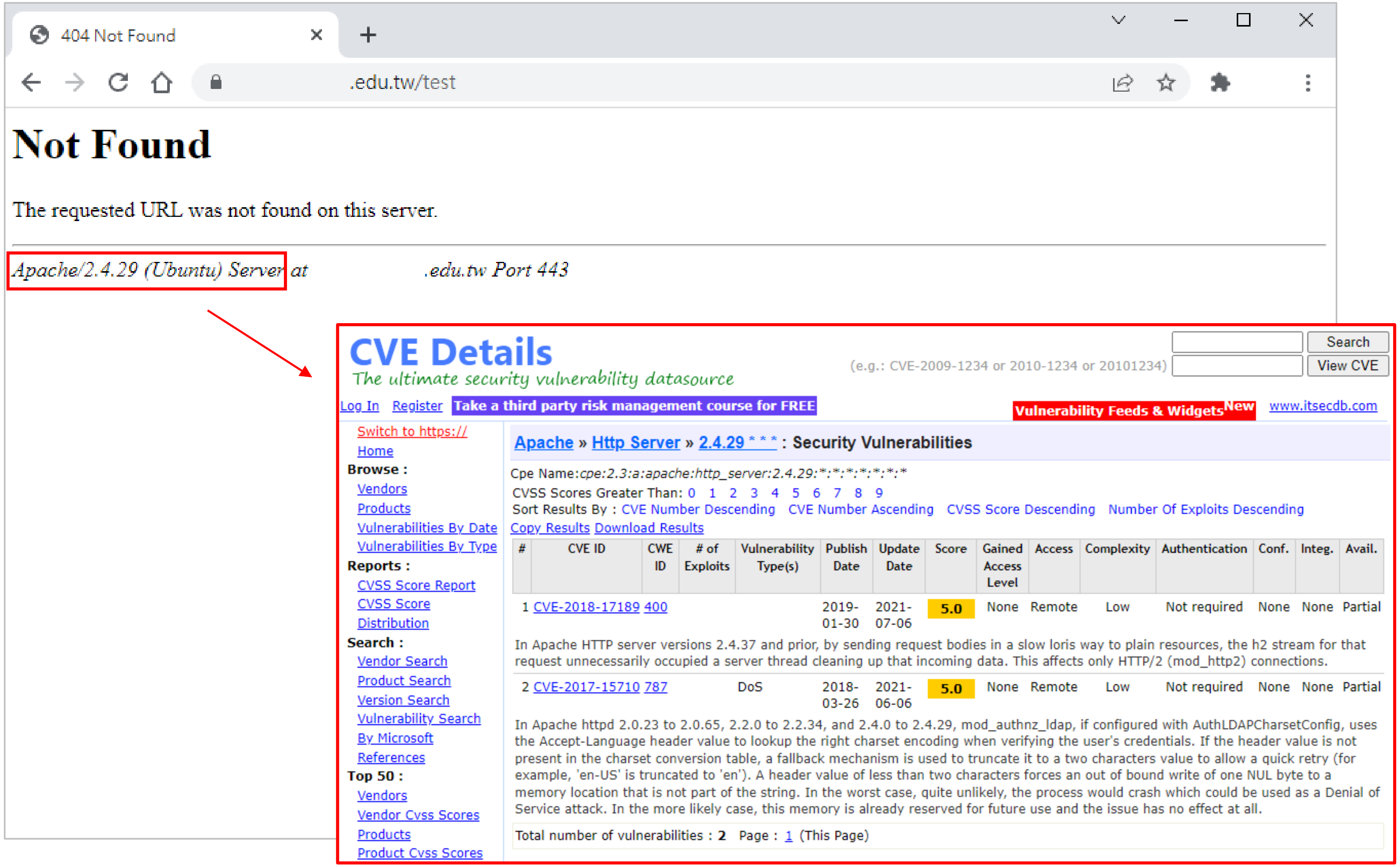
網頁內容 →



```
<html>
  <head>
    <title>An Example Page</title>
  </head>
  <body>
    <p>Hello World, this is a very simple HTML document.</p>
  </body>
</html>
```

REF: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

Status Code Leakage



Not Found

The requested URL was not found on this server.

Apache/2.4.29 (Ubuntu) Server at .edu.tw Port 443

CVE Details

The ultimate security vulnerability datasource

Apache » Http Server » 2.4.29 * * * : Security Vulnerabilities

Cpe Name:*cpe:2.3:a:apache:http_server:2.4.29:*:*:*:*:*:*:*

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

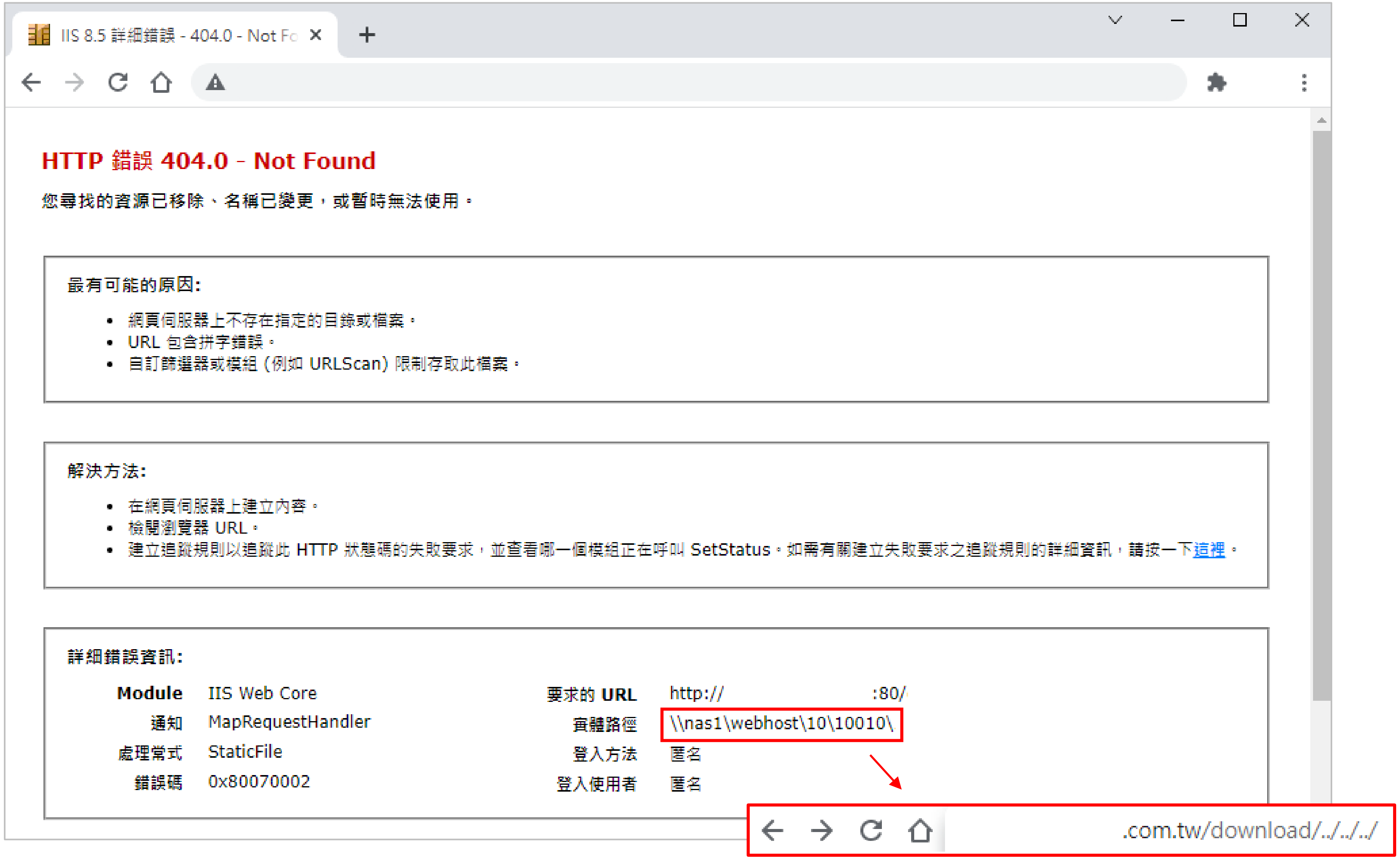
Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending

Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2018-17189	400			2019-01-30	2021-07-06	5.0	None	Remote	Low	Not required	None	None	Partial
In Apache HTTP server versions 2.4.37 and prior, by sending request bodies in a slow loris way to plain resources, the h2 stream for that request unnecessarily occupied a server thread cleaning up that incoming data. This affects only HTTP/2 (mod_http2) connections.														
2	CVE-2017-15710	787		DoS	2018-03-26	2021-06-06	5.0	None	Remote	Low	Not required	None	None	Partial
In Apache httpd 2.0.23 to 2.0.65, 2.2.0 to 2.2.34, and 2.4.0 to 2.4.29, mod_authnz_ldap, if configured with AuthLDAPCharsetConfig, uses the Accept-Language header value to lookup the right charset encoding when verifying the user's credentials. If the header value is not present in the charset conversion table, a fallback mechanism is used to truncate it to a two characters value to allow a quick retry (for example, 'en-US' is truncated to 'en'). A header value of less than two characters forces an out of bound write of one NUL byte to a memory location that is not part of the string. In the worst case, quite unlikely, the process would crash which could be used as a Denial of Service attack. In the more likely case, this memory is already reserved for future use and the issue has no effect at all.														

Total number of vulnerabilities : 2 Page : 1 (This Page)

Status Code Leakage (cont.)



HTTP 錯誤 404.0 - Not Found

您尋找的資源已移除、名稱已變更，或暫時無法使用。

最有可能的原因:

- 網頁伺服器上不存在指定的目錄或檔案。
- URL 包含拼字錯誤。
- 自訂篩選器或模組 (例如 URLScan) 限制存取此檔案。

解決方法:

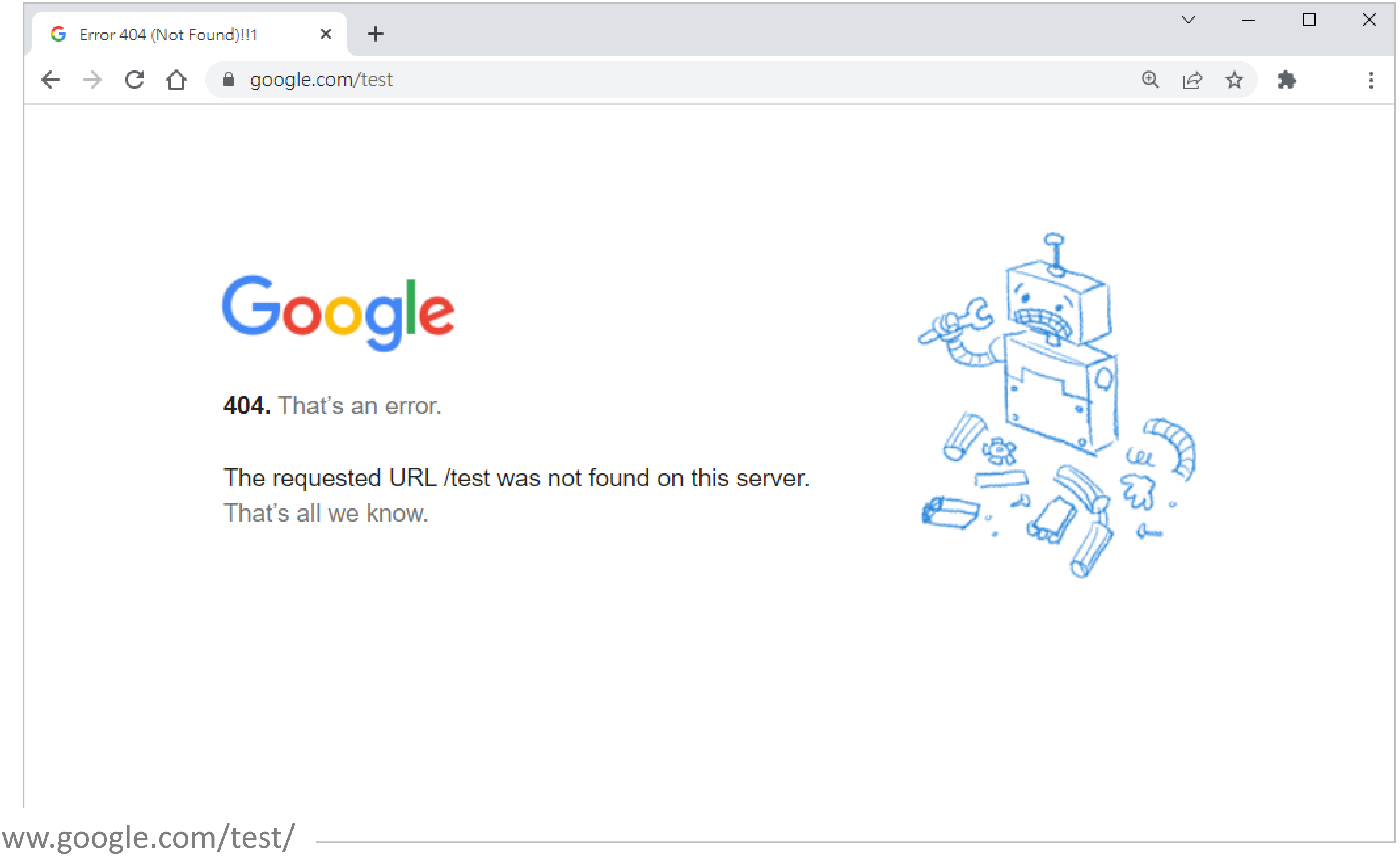
- 在網頁伺服器上建立內容。
- 檢閱瀏覽器 URL。
- 建立追蹤規則以追蹤此 HTTP 狀態碼的失敗要求，並查看哪一個模組正在呼叫 SetStatus。如需有關建立失敗要求之追蹤規則的詳細資訊，請按一下這裡。

詳細錯誤資訊:

Module	IIS Web Core	**要求的 URL**	http:// :80/
通知	MapRequestHandler	實體路徑	\\nas1\webhost\10\10010\
處理常式	StaticFile	登入方法	匿名
錯誤碼	0x80070002	登入使用者	匿名

.com.tw/download/../../../

Customized Response Message

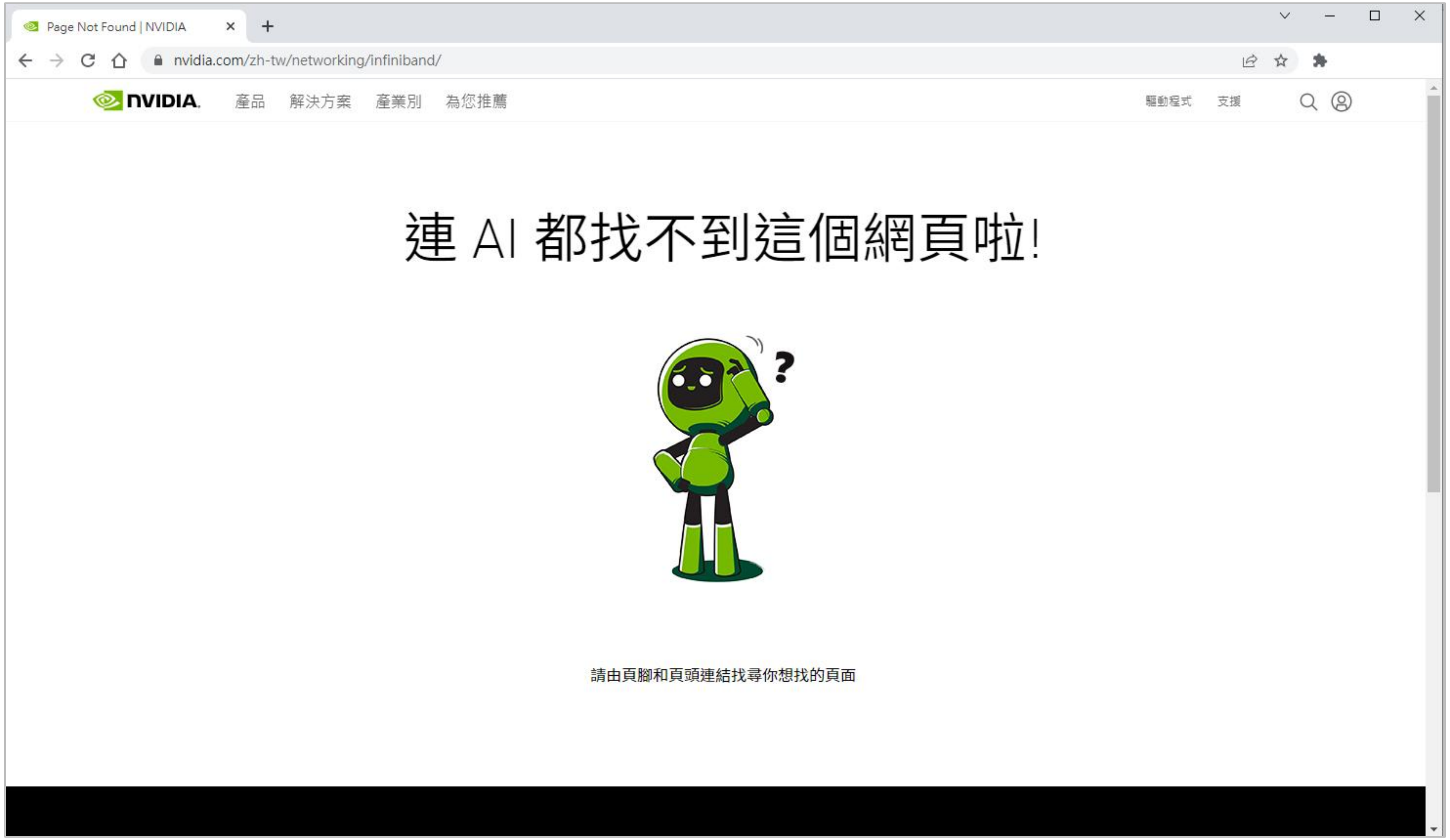


404. That's an error.

The requested URL /test was not found on this server. That's all we know.

REF: https://www.google.com/test/

Customized Response Message (cont.)



連 AI 都找不到這個網頁啦!

請由頁腳和頁頭連結找尋你想找的頁面

REF: https://www.nvidia.com/zh-tw/test/

Hypertext Transfer Protocol (cont.)

- **Disconnection**
  - The TCP-IP connection is broken by the server when the whole document has been transferred.
  - Web client may abort the transfer by breaking the connection before this, in which case the server shall not record any error condition.
  - Web server need not store any information about the request after disconnection.

REF: https://www.w3.org/Protocols/HTTP/AsImplemented.html

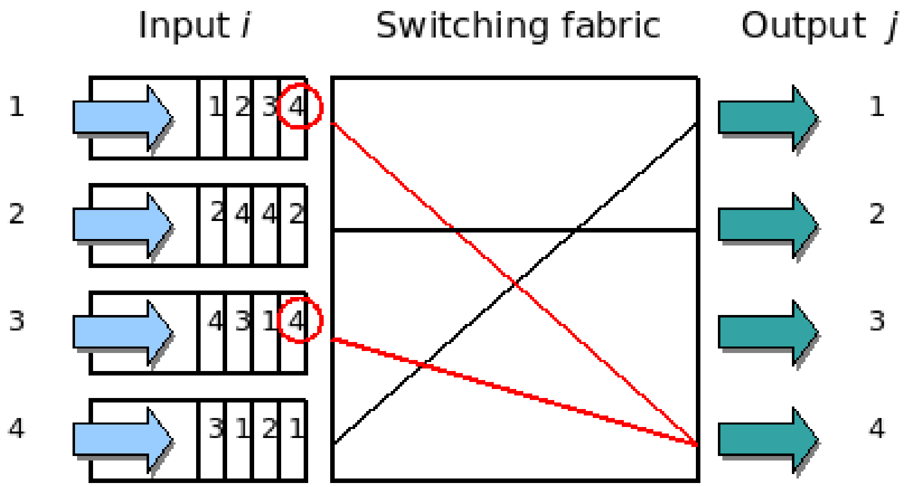
Hypertext Transfer Protocol (cont.)

- HTTP/1.0
  - Documented in 1996 as RFC 1945, revised from HTTP/0.9
  - It separates connection for every resource request even accessing the same web server
  - With **head of line blocking** problem (HOL)

Hypertext Transfer Protocol (cont.)

- Head of line blocking problem

- The 1st and 3rd input flows are competing to send packets to the same output interface
- If the switching fabric decides to transfer the packet from the 3rd input flow, the 1st input flow cannot be processed in the same time slot.
- The 1st input flow is blocking a packet for output interface 3, which is available for processing.



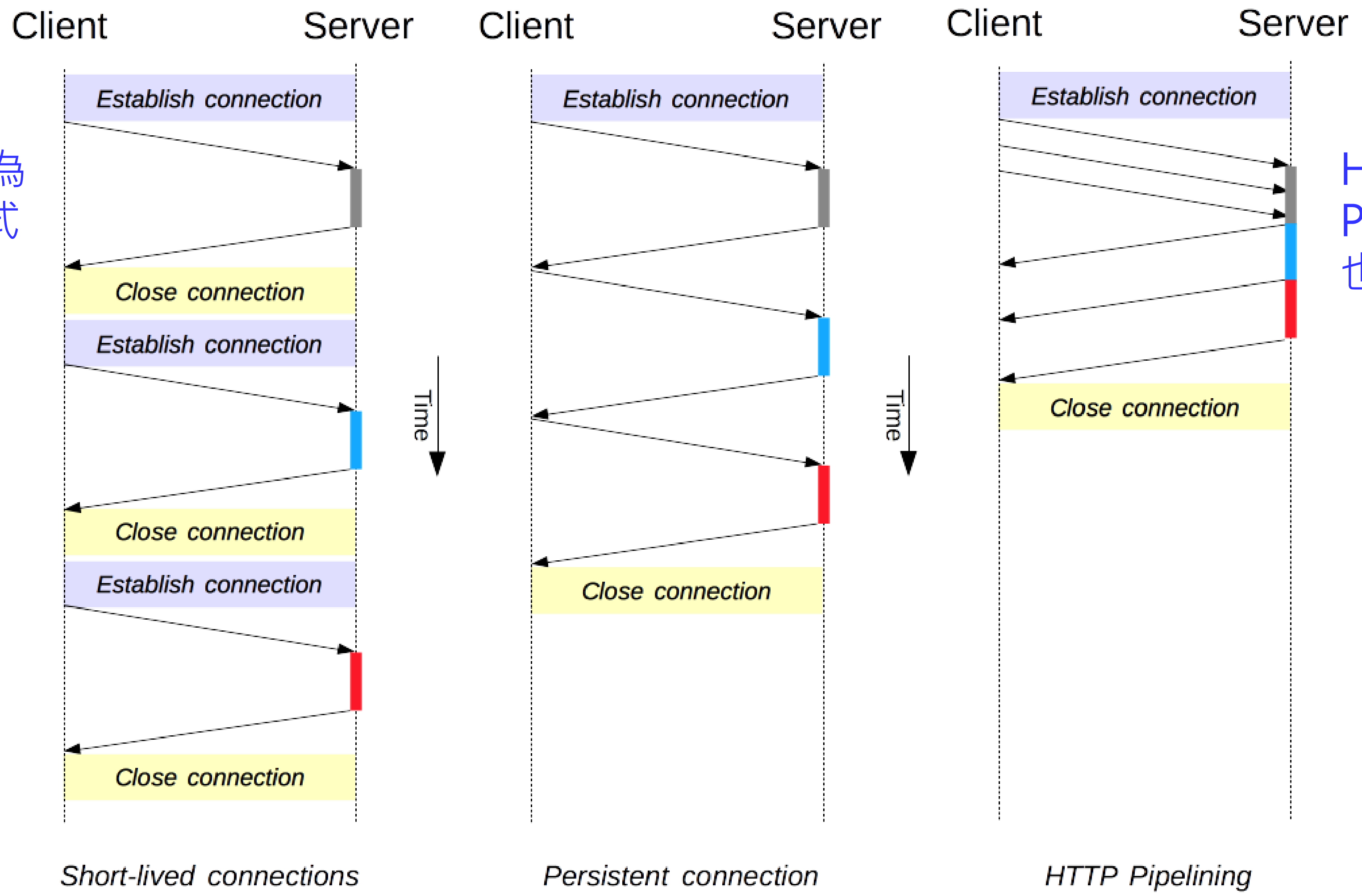
REF: https://en.wikipedia.org/wiki/Head-of-line_blocking

Hypertext Transfer Protocol (cont.)

- HTTP/1.1
  - A revision of the HTTP/1.0
  - It can reuse the connection to access resource from the same web
  - “HTTP/1.1 200 OK” is the common example used to test HTTP server-client connection
  - When parallel requests in the browser is coming, the subsequent requests have to wait for the former ones to be completely served in HTTP/1.1, it still meets the head of line blocking problem

Connection management in HTTP/1.x

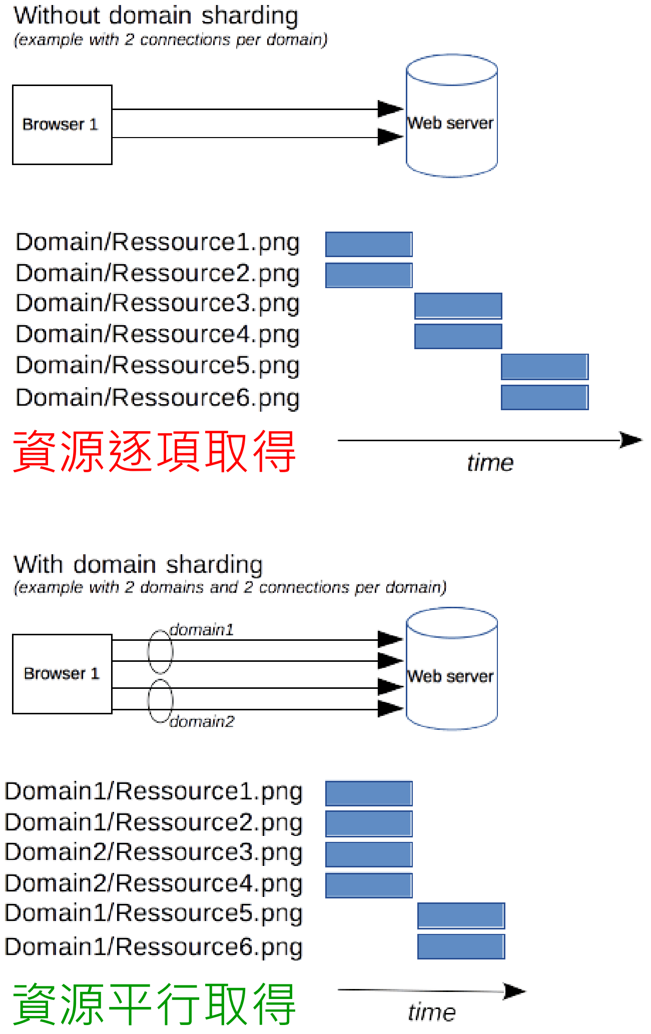
HTTP/1.0 預設為 Short-lived 模式



HTTP/1.1 將預設改為 Persistent，除此之外也支援 Pipeline 模式

REF: https://developer.mozilla.org/en-US/docs/Web/HTTP/Connection_management_in_HTTP_1.x

Connection management in HTTP/1.x



資源逐項取得

資源平行取得

在 HTTP/1.x 使用 domain sharing 可以加快取得需要的資源 (建立很多 connection)

但在 HTTP/2 因為能更有效率地平行處理 (connection coalescing)，故啟用這個功能反而可能會降低效率

REF: https://developer.mozilla.org/en-US/docs/Web/HTTP/Connection_management_in_HTTP_1.x

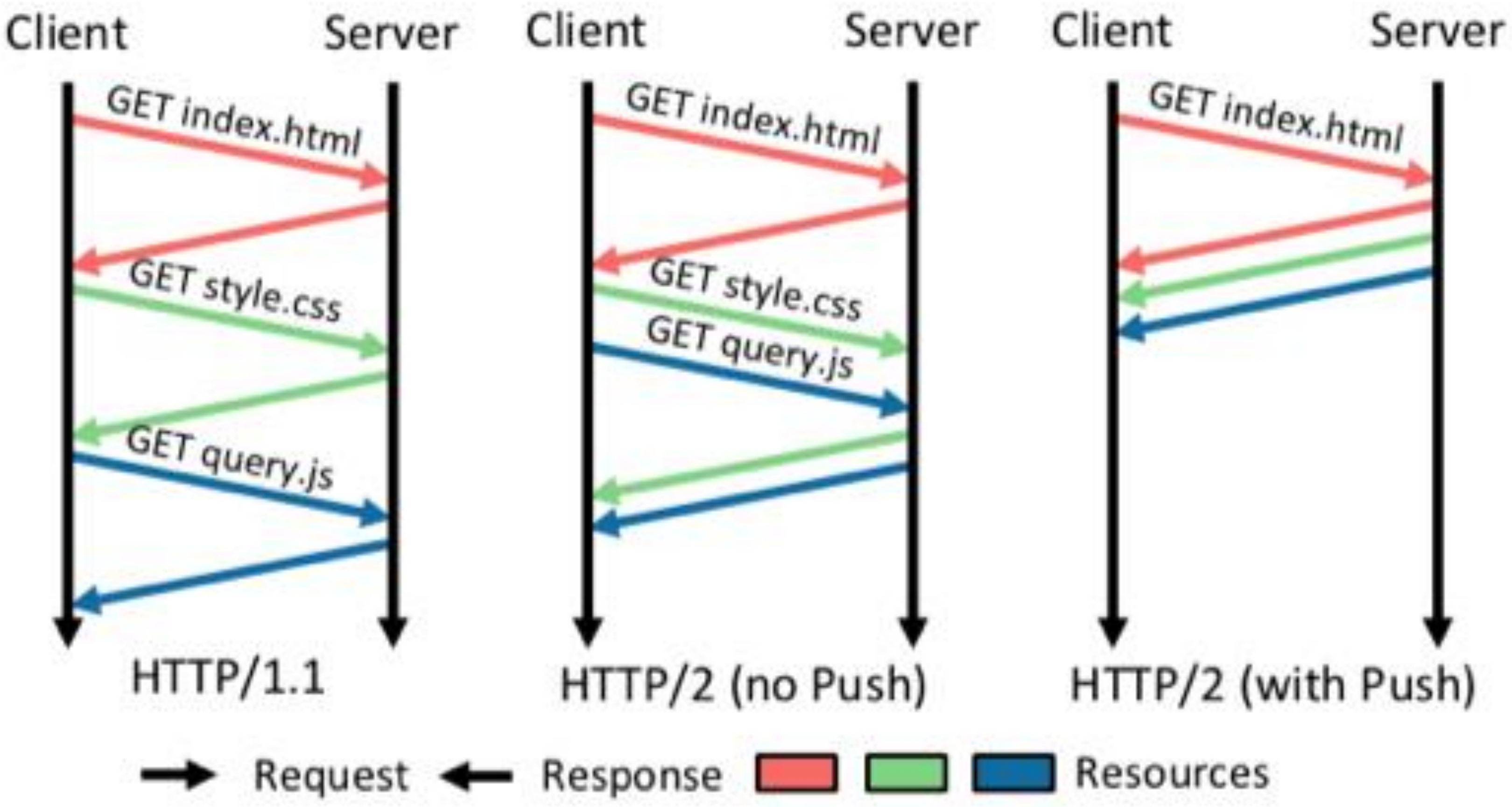
Hypertext Transfer Protocol (cont.)

- HTTP/2 (HTTP/2.0)
  - Published in 2015, and it is a major revision of the HTTP/1.1
  - Supported by most mainstream web browsers
  - Allow HTTP/2 **Server Push** (to respond with data for more queries than the client requested)
  - Pipelining of requests
  - Fix head of line blocking problem by multiplexing multiple requests over a single TCP connection

- SPDY (pronounced like "speedy") from HTTP/1.1 to HTTP/2
  - It was a previous HTTP-replacement protocol developed by Google (at session layer)
  - The HTTP WG considered proposed SPDY protocols to be used HTTP/1.1
  - HTTP/2 was built with SPDY concepts, each user action is given a "stream ID", providing is a single TCP channel connecting the user to the server

REF: https://en.wikipedia.org/wiki/HTTP/2

Server Push



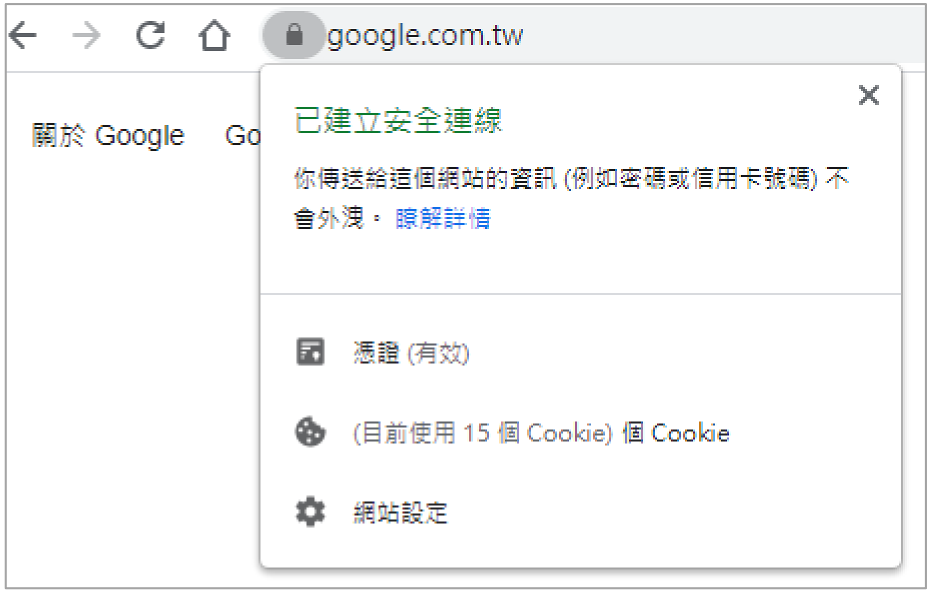
Server 主動將其他需要搭配的資源一起主動推送給 Client，以節省 Client 送出 request 的耗時，進而達到提升效率之目的

REF: https://blog.apnic.net/2018/04/26/adoption-performance-and-human-perception-of-http-2-server-push/

Hypertext Transfer Protocol Secure (HTTPS)

- Hypertext Transfer Protocol Secure (HTTPS) is a secured extension of the HTTP. Its communication encryption relies on SSL/TLS (HTTP over SSL/TLS).

- The SSL/TLS protocols are implemented as a transparent wrapper around the HTTP protocol in session layer.



已建立安全連線

你傳送給這個網站的資訊 (例如密碼或信用卡號碼) 不會外洩。 瞭解詳情

憑證 (有效)

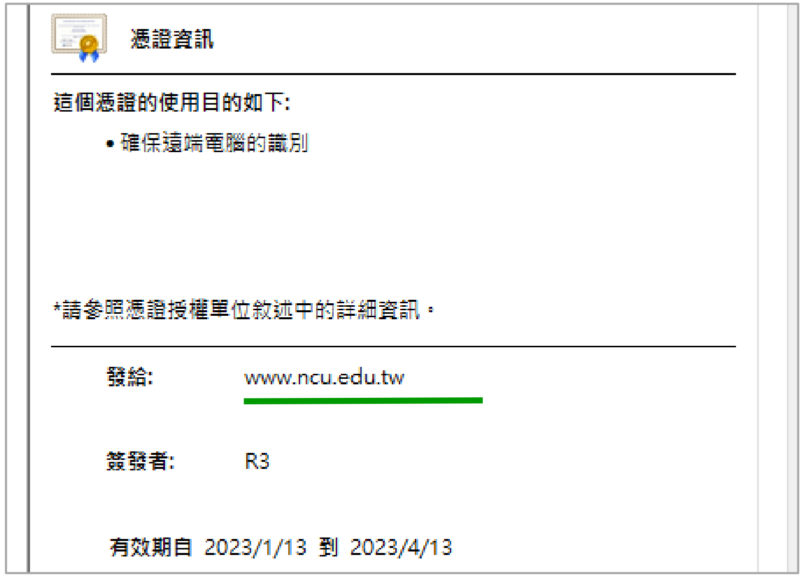
(目前使用 15 個 Cookie) 個 Cookie

網站設定

Hypertext Transfer Protocol Secure (HTTPS) (cont.)

- The motivations for HTTPS are security browsing and host authentication for the website to against MITM attack (sniffer and tamper)

- The aspect of trust websites is depending on certificate authorities (CA) build-in the browsers, such as Chrome, Firefox, and IE



憑證資訊

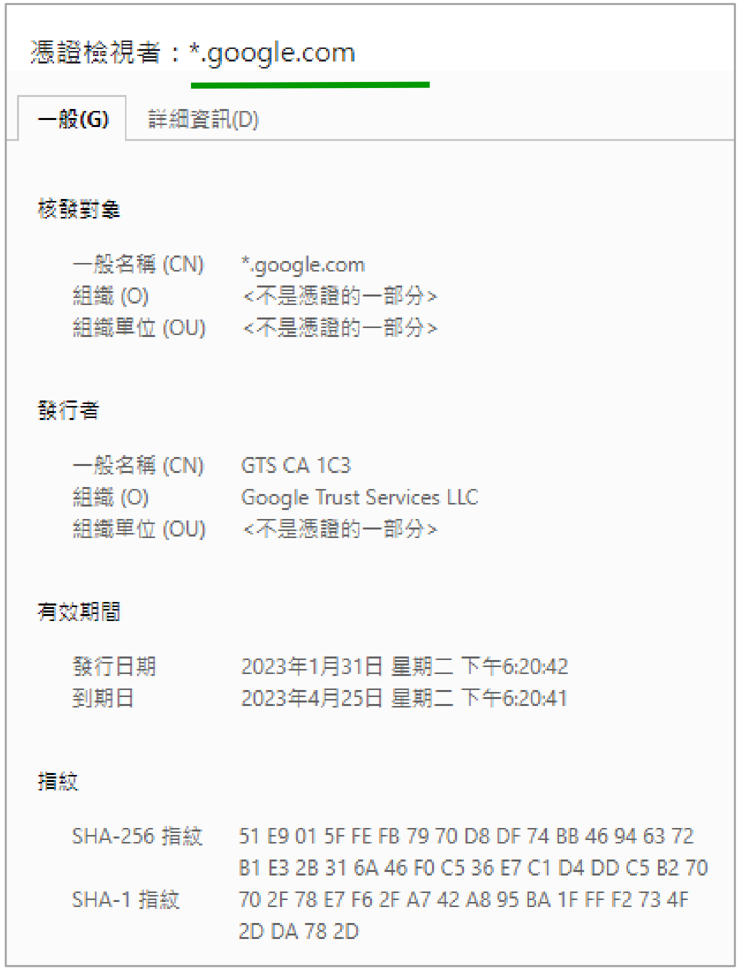
這個憑證的使用目的如下:

- 確保遠端電腦的識別

*請參照憑證授權單位敘述中的詳細資訊。

發給:	www.ncu.edu.tw
簽發者:	R3

有效期自 2023/1/13 到 2023/4/13



憑證檢視者：*.google.com

核發對象

一般名稱 (CN)	*.google.com
組織 (O)	<不是憑證的一部分>
組織單位 (OU)	<不是憑證的一部分>

發行者

一般名稱 (CN)	GTS CA 1C3
組織 (O)	Google Trust Services LLC
組織單位 (OU)	<不是憑證的一部分>

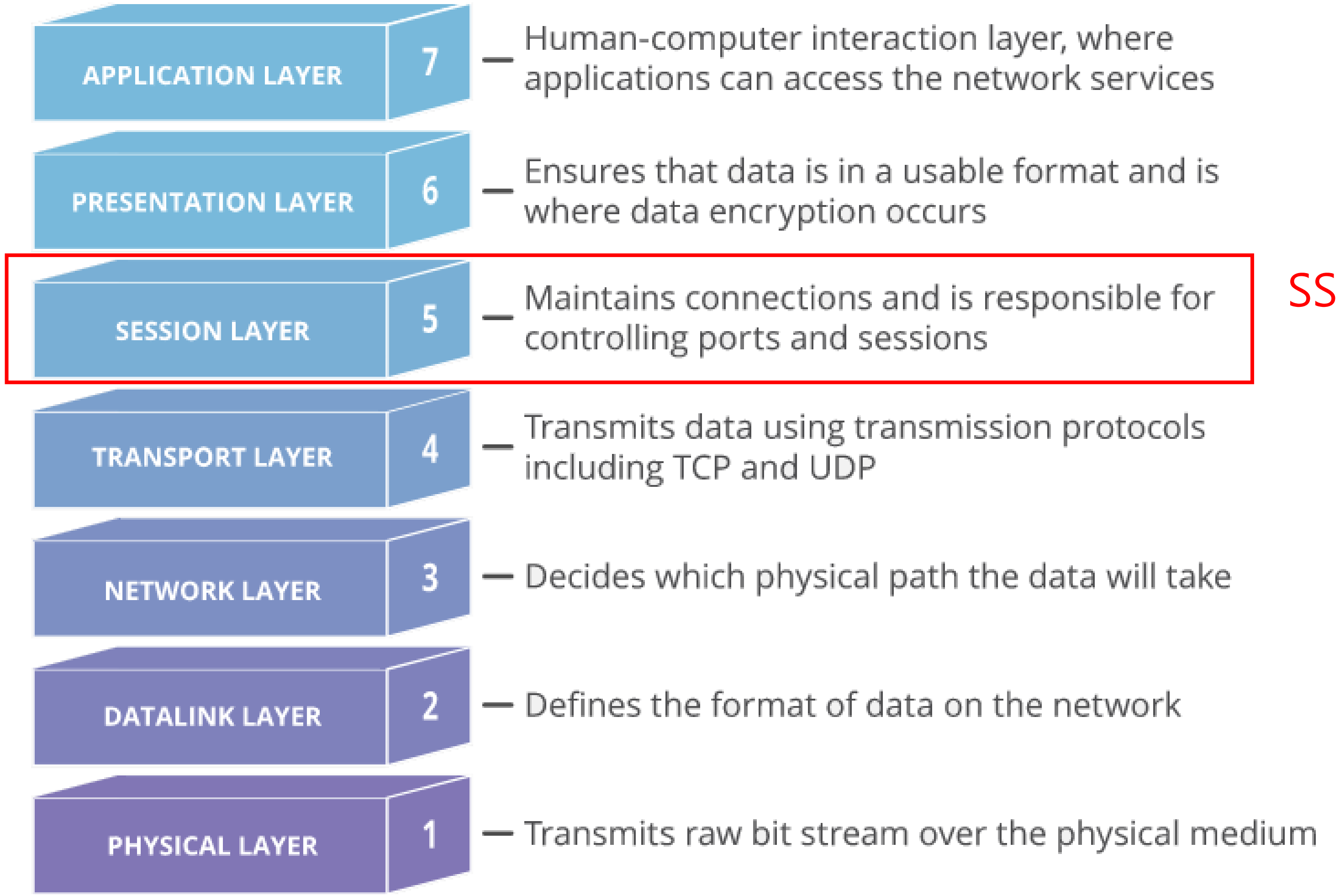
有效期間

發行日期	2023年1月31日 星期二 下午6:20:42
到期日	2023年4月25日 星期二 下午6:20:41

指紋

SHA-256 指紋	51 E9 01 5F FE FB 79 70 D8 DF 74 BB 46 94 63 72 B1 E3 2B 31 6A 46 F0 C5 36 E7 C1 D4 DD C5 B2 70
SHA-1 指紋	70 2F 78 E7 F6 2F A7 42 A8 95 BA 1F FF F2 73 4F 2D DA 78 2D

Open Systems Interconnection (OSI) Reference Model



APPLICATION LAYER 7 — Human-computer interaction layer, where applications can access the network services

PRESENTATION LAYER 6 — Ensures that data is in a usable format and is where data encryption occurs

SESSION LAYER 5 — Maintains connections and is responsible for controlling ports and sessions

TRANSPORT LAYER 4 — Transmits data using transmission protocols including TCP and UDP

NETWORK LAYER 3 — Decides which physical path the data will take

DATALINK LAYER 2 — Defines the format of data on the network

PHYSICAL LAYER 1 — Transmits raw bit stream over the physical medium

SSL/TLS 歸類在第五層

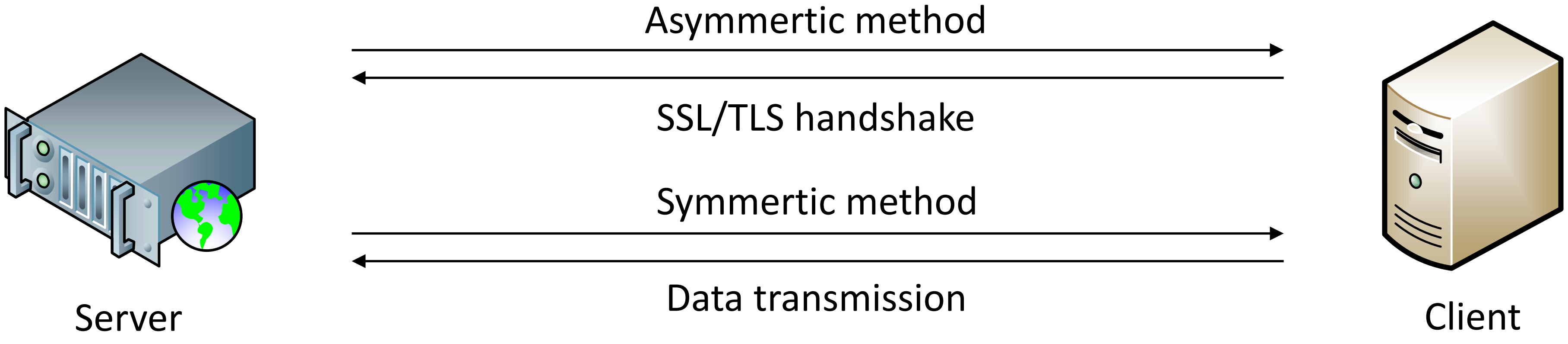
REF: https://www.cloudflare.com/zh-tw/learning/ddos/what-is-layer-7/

Hypertext Transfer Protocol Secure (HTTPS) (cont.)

- HTTPS is an extension of HTTP, it is also referred to as HTTP over SSL/TLS.

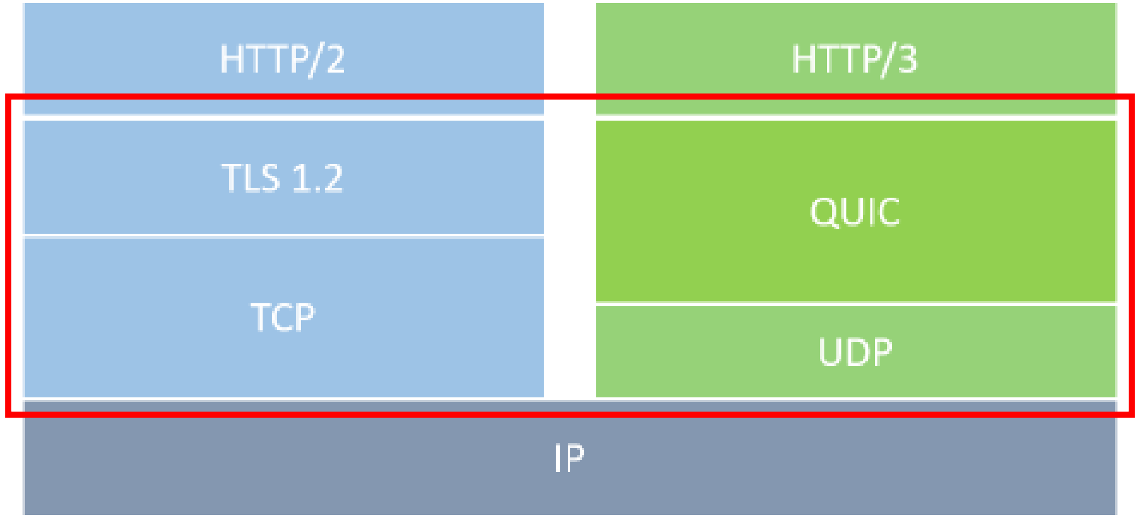
(1-1) Client uses server's public key to encrypt message (send session key).
(1-2) Server uses its private key to decrypt message, getting client's public key, and uses this key to encrypt message (receive session key) and return an ack.
(1-3) handshake complete

(2-1) Using symmetric session key to transmission the data.



Hypertext Transfer Protocol (cont.)

- HTTP/3
  - It is a new upcoming version of HTTP
  - The name was HTTP-over-QUIC originally
  - HTTP/3 uses **QUIC** technology, it aims to fix the head-of-line blocking issues completely
  - The change of migrating from legacy transmission way to QUIC speeding up the performance while it might sacrifice the reliability

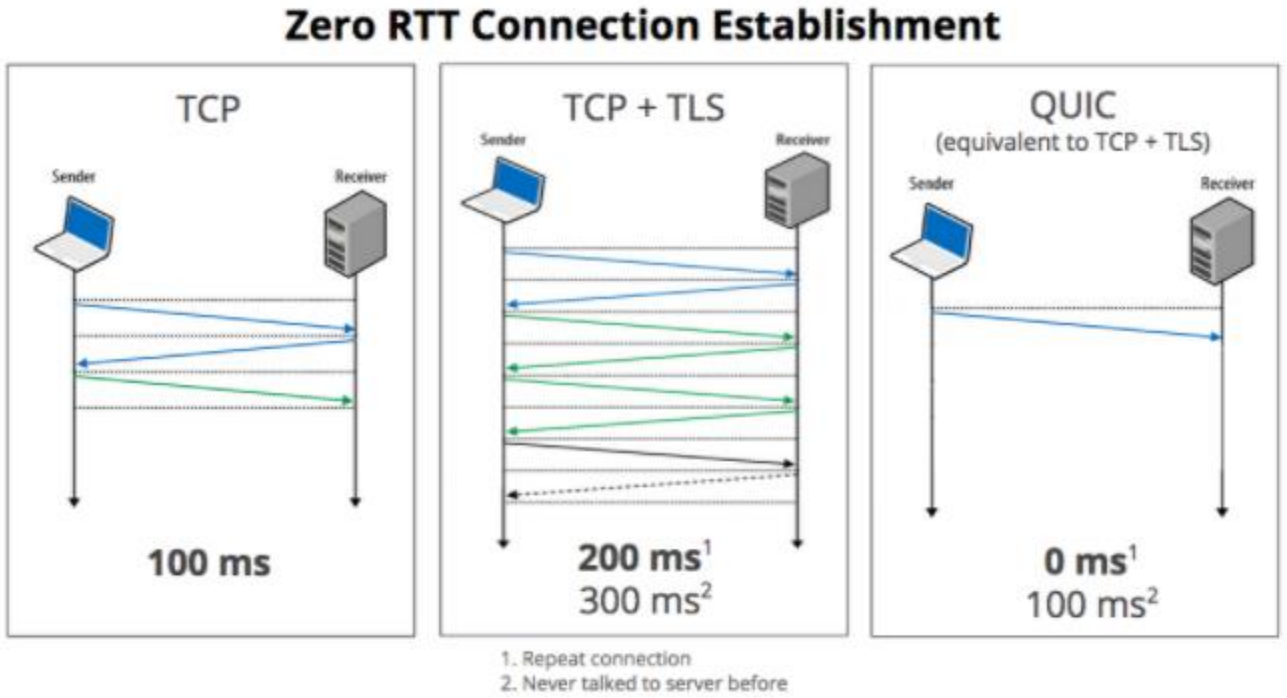


REF: https://en.wikipedia.org/wiki/HTTP/3

REF: https://solidgeargroup.com/en/quic-what-hides-behind-http-3/

Hypertext Transfer Protocol (cont.)

- Quick UDP Internet Connection (QUIC)
  - It is a transport layer network protocol developed initially by Google
  - Congestion control is used over User Datagram Protocol (UDP), not original Transmission Control Protocol (TCP) in HTTP/1.x and HTTP/2



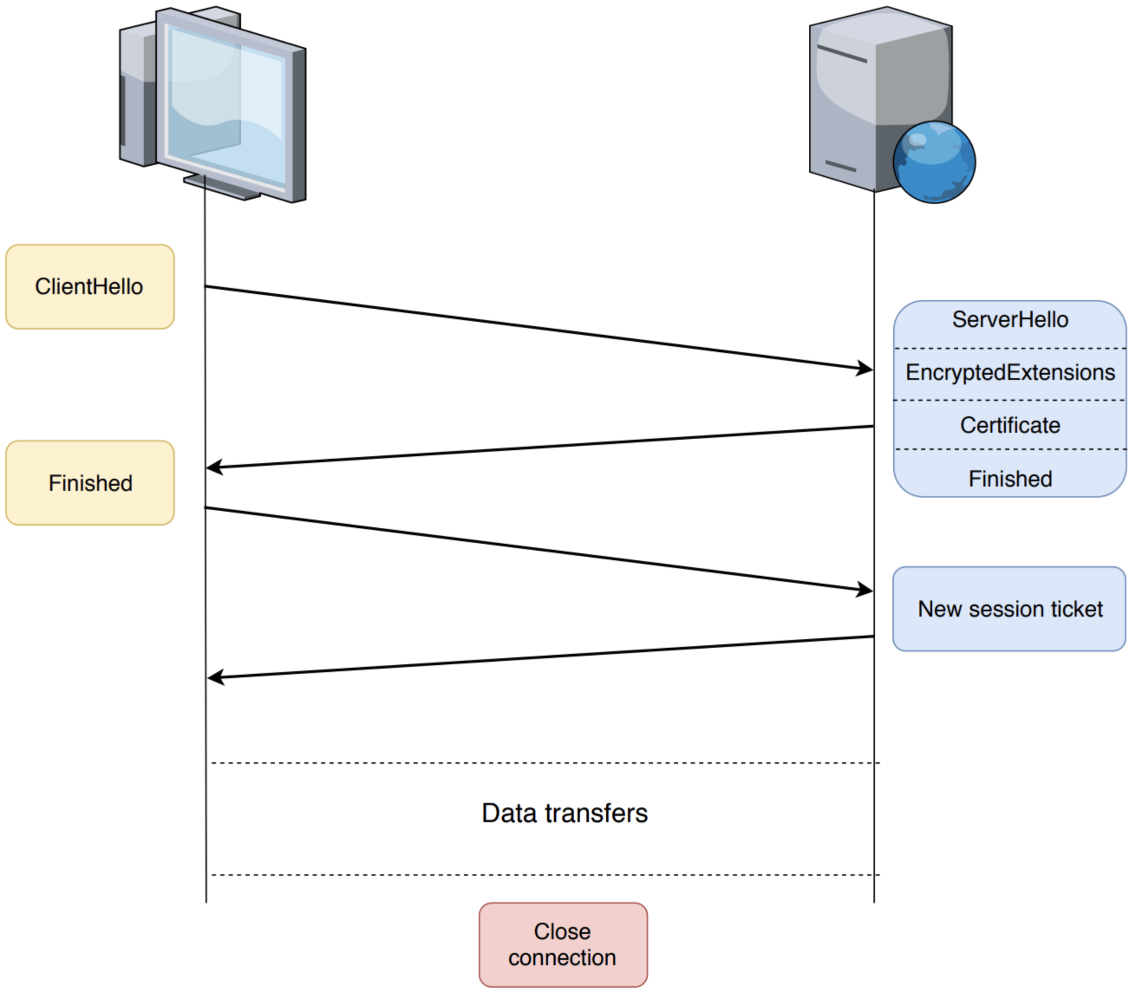
REF: https://solidgeargroup.com/en/quic-what-hides-behind-http-3/

Hypertext Transfer Protocol (cont.)

- Quick UDP Internet Connection (QUIC)
  - QUIC supports a set of multiplexed connections over UDP/443 and has been designed to provide equivalent security to TLS/SSL by reducing latency in both connection and data transport
  - It makes web client-server communications to be more efficient
  - When each resource is assigned an individual conceptual flow, the transport layer may know that when a packet is lost, subsequent packets can still be used if they contain data from another resource that was not in the lost packet
  - It solves is the slow establishment of the TLS/SSL connection, which requires several information exchanges (round trip times, RTTs), while QUIC reduces the cost to practically zero RTTs

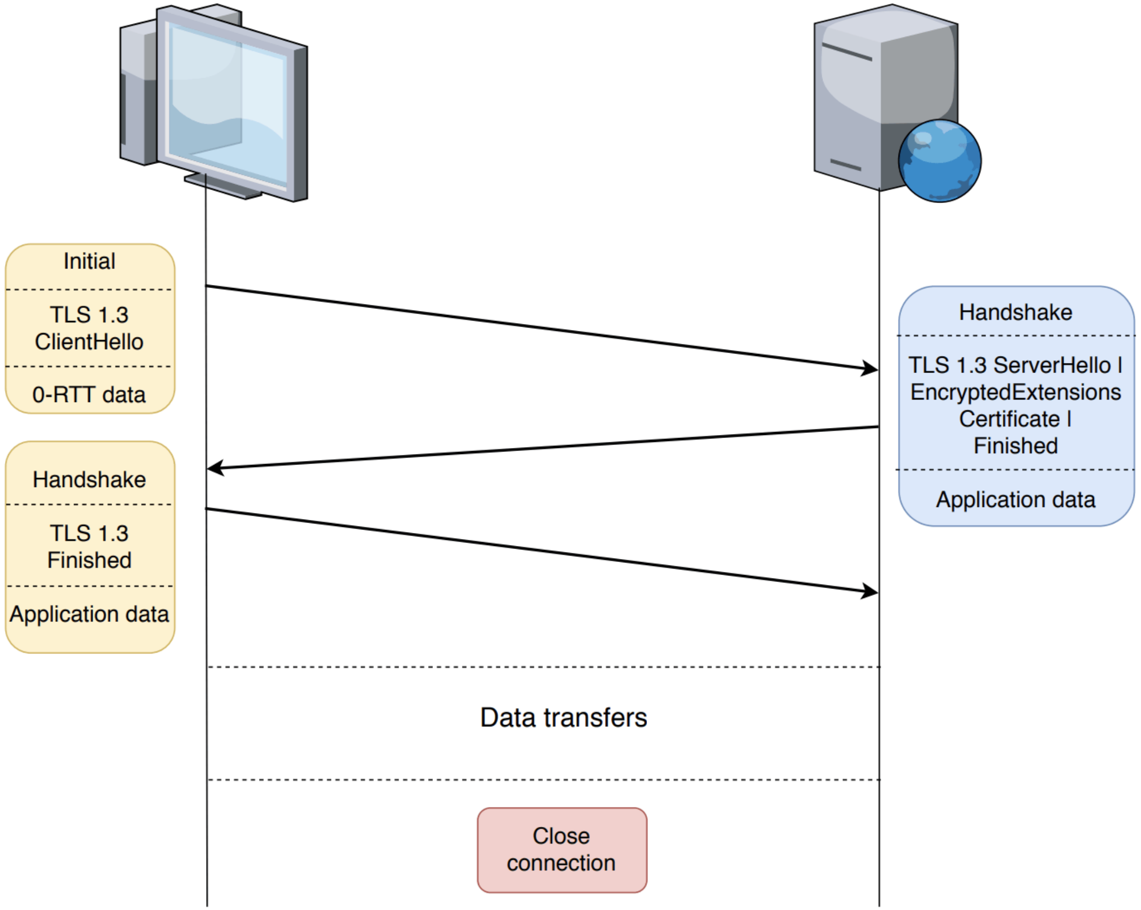
REF: https://solidgeargroup.com/en/quic-what-hides-behind-http-3/

Hypertext Transfer Protocol (cont.)



REF: https://solidgeargroup.com/en/quic-what-hides-behind-http-3/

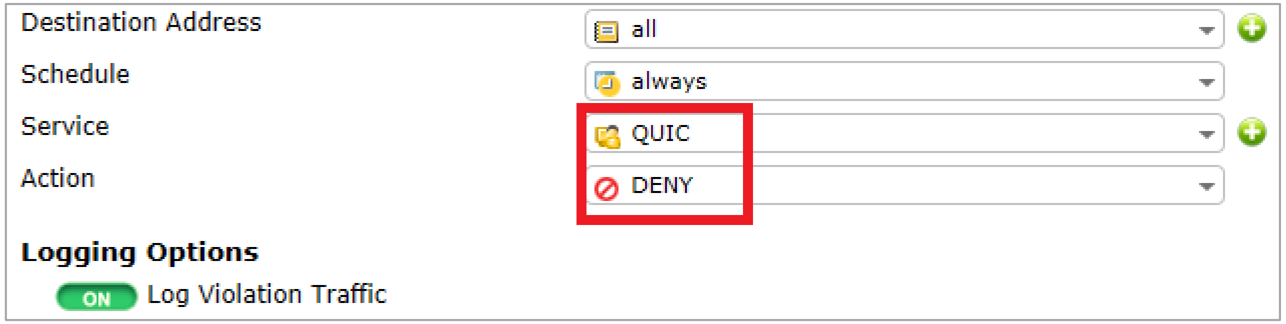
Hypertext Transfer Protocol (cont.)



REF: https://solidgeargroup.com/en/quic-what-hides-behind-http-3/

Hypertext Transfer Protocol (cont.)

- **However, some antivirus/firewall are still blocking QUIC by default**
  - *`Most firewalls have extensive functionality when dealing with HTTP and HTTPS traffic`*
  - *`In most architectures, when HTTP traffic is detected, it is passed on to a web protection module that performs web filtering, deep packet inspection`*
  - *`Enabling QUIC may cause network security appliances to be unavailable to restrict access, which also increases the risk of downloading malware when browsing the web`*



Destination Address	all
Schedule	always
Service	QUIC
Action	DENY

Logging Options

ON Log Violation Traffic

REF: https://kb.fortinet.com/kb/viewContent.do?externalId=FD36529

REF: https://www.fastvue.co/fastvue/blog/googles-quic-protocols-security-and-reporting-implications/

使用不同語言編寫的 Web Application

- VBScript
  - VBScript 是 Visual Basic 語言家族的成員，全名 Microsoft Visual Basic Scripting Edition，簡稱 VBScript。VBScript 是完全免費的直譯語言，早期的微軟 IE 瀏覽器都有支援 VBScript，能讓網頁設計者開發互動多媒體的網頁內容，但現在已經較少看到。

- Java Applet
  - 使用 Java 語言撰寫的一種 Java 應用程式，我們需要使用編譯器將原始程式碼編譯成位元組碼，即 Java Applet，瀏覽器需要使用 Java 直譯器 JVM (Java Virtual Machine) 來執行。
  - 可以在網頁讀取的時候看到這樣的動畫標示

REF: https://java.com

使用不同語言編寫的 Web Application (cont.)

- ActionScript and Flash
  - ActionScript 是 Macromedia 公司開發的 Script 腳本語言 (後來被 Adobe 收購)，可以讓 Flash 動畫電影檔產生互動效果，這是一種類似 JavaScript 語法的腳本語言。瀏覽器只需安裝 Flash 播放程式，就可以在網頁顯示 Flash 所建立的動畫效果，但 Flash 因為資安問題已經已經被主流瀏覽器停止支援，同時 Adobe 也不再提供維護。

- Silverlight
  - Silverlight 是一套用來開發豐富網際網路應用程式 (Rich Internet Application, RIAs) 的工具程式，其定位和 Macromedia 公司的 Flash 相同，屬微軟家族的 RIA 工具，提供網頁設計師另一種建立網頁內容的選擇，但由於發展因素已由微軟宣布終止維護。

使用不同語言編寫的 Web Application (cont.)

- JavaScript (JS)
  - JavaScript 是 Netscape 開發的 Script 腳本語言，程式語法容易，初學者也可以快速在網頁上建立互動效果。Jscript 則為微軟推出相容 JavaScript 的 Script 語言，在過去的時代中 Netscape 或 Mozilla Firefox 支援 JavaScript 但 Internet Explorer 支援 Jscript。

- AJAX
  - AJAX 是 Asynchronous JavaScript And XML 的縮寫，譯成中文就是非同步 JavaScript 和 XML 技術，AJAX 是由 HTML 和 CSS、XML、XML DOM 及 XMLHttpRequest 等物件組成。

使用不同語言編寫的 Web Application (cont.)

- ASP (Active Server Pages)
  - 英文字面上是一種讓網頁在伺服器上動起來的技術，能夠將 Script 語言內嵌 HTML 標籤的網頁，在伺服器端產生動態網頁內容，這是一種在伺服器端以直譯方式執行的網頁技術。
  - ASP.NET 是繼 ASP 3.0 後，微軟開發的伺服端網頁技術，以 CLR (Common Language Runtime) 架構建立的 .NET 程式設計平台，可以讓開法者運用該語言在伺服器端建立Web應用程式。
  - 可以在使用微軟的 IIS 加設網頁伺服器的系統常看到 .asp 或 .aspx 的網頁檔

使用不同語言編寫的 Web Application (cont.)

- JSP (Java Server Pages)
  - JSP 是 Java 家族中能和 ASP 一較長短的伺服器端網頁技術，結合 HTML 和 Java Servlet；其中 Java Applet 是下載到客戶端執行的程式檔，Java Servlet 則是在伺服端執行提供服務。

- Node.js
  - 以往 JavaScript 只能用於使用者端，但 Node.js 提供了同時運用在伺服器端和使用者端的方式，可支援密集和高互動性的應用情境。可透過 NPM 管理器取得其他開源的第三方程式整合到自己所設計的網頁中，搭配框架及模組化設計提升網頁開發與建構的速度。

使用不同語言編寫的 Web Application (cont.)

- PHP (PHP: Hypertext Preprocessor)
  - 同樣是開放原始碼的伺服器端 Script 語言，可以直接內嵌於 HTML 網頁，特別適用在 Web 網站開發，主要是使用在 Unix 作業系統的伺服端網頁技術，目前也支援 Windows 作業系統。
  - PHP 可以用來取代 CGI 類型的文字應用設計 (例如將 Perl CGI 實作的功能改用 PHP 來建立)，PHP 語言只需配合 Web 伺服器和瀏覽器的開發環境，就可以透過瀏覽器執行 PHP 語法，並且讓使用者的瀏覽器顯示執行結果的網頁內容。
  - PHP 亦可配合伺服端的資料庫系統建立網頁資料庫 (Web Database)，直接連接和存取資料庫的記錄資料在瀏覽器顯示。
  - Linux + Apache + MySQL(MariaDB) +PHP 通稱 LAMP，能提供基礎的 web 開發服務環境。目前最新的主流版本雖然是 PHP8，但仍然可以看到許多系統仍繼續停在 PHP7 環境。

共通匣道介面

- CGI (Common Gateway Interface)
  - 共通匣道介面提供 Web 伺服器執行外部程式的管道，CGI 應用程式是一種外部程式的執行檔，能夠使用各種程式語言來開發，例如：Visual Basic、C、C++ 和 Perl，程式需要編譯成執行檔案，以便在伺服器端執行。
  - CGI 能夠讓 web service 提供使用者執行類似 console application (透過命令提示字元運作的應用) 之程式。CGI 可以在 web server 上實現動態 HTML 頁面的展現 (例如隨著 user 透過網頁輸入的值或者一天的 24 小時有所變化)。
  - 使用 CGI 的 web 應用程式設計中須包含 CGI 指令碼。通常情況下，一次的 client request 會對應一個 CGI 指令碼的執行，並產生相對應的 HTML 以 response。
  - 現在已經較少看到 CGI 的描述，很多網頁應用情境會說明其使用了 Rest API 或 HTML5/Web Socket (提供更豐富的雙向互動) 的方式來實作 。

Q & A

Thanks for your time and attention.