

# Web

tsaimh (2022, CC BY)  
jnlm (2019-2021, CC BY)  
? (?-2018)

交大資工系資訊中心

Computer Center of Department of Computer Science, NCTU

# Outline

- Web hosting
  - Basics
  - Client-Server architecture
  - HTTP protocol
  - Static vs. dynamic pages
  - Virtual hosts
- Proxy
  - Forward proxy
  - Reverse proxy

# Web Hosting – Basics (1)

- Three major techniques in WWW (World Wide Web) System
  - HTML (JavaScript / CSS)
  - HTTP
  - URL
- HTML (1) – HyperText Markup Language
  - Providing a means to describe the structure of text-based information in a document.
  - The original HTML is created by Tim Berners-Lee.
  - Published in 1993 by the IETF as a formal "application" of SGML (with an SGML Document Type Definition defining the grammar).
  - The HTML specifications have been maintained by the World Wide Web Consortium (W3C).
    - <http://www.w3.org/>

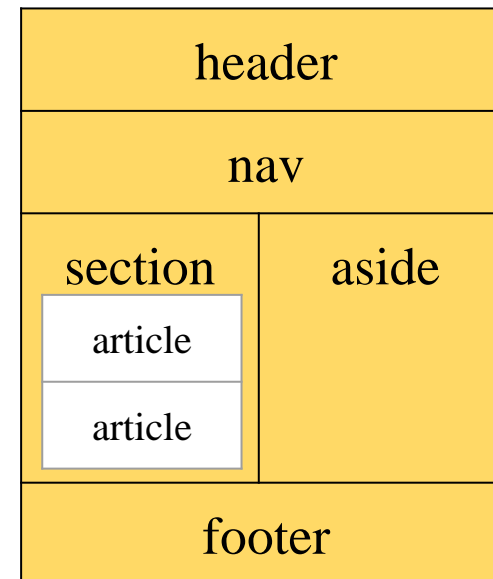
# Web Hosting – Basics (2)

- HTML (2)
  - Mark-up the text and define presentation effect by HTML Tags.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
  <head>
    <title>Hello World!</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

# Web Hosting – Basics (3)

- HTML5
  - Published in October 2014 by the World Wide Web Consortium (W3C)
  - Many new syntactic features are included.
  - article, aside, footer, header, nav, section, ...
  - include and handle multimedia and graphical content
  - video, canvas, audio
  - `<!DOCTYPE html>`



# Web Hosting – Basics (4)

- HTTP – Hyper-Text Transfer Protocol
  - A TCP-based protocol
  - Stateless
  - Communication method between client and server. All browsers and web servers have to follow this standard.
  - Originally designed to transmit HTML pages.
  - Now it is used to format, transmit, and link documents of variety media types
    - Text, picture, sound, animation, video, ...
    - Mobile App APIs
      - <https://developer.pixnet.pro/#!/doc/pixnetApi/oauthApi>
      - [https://developers.facebook.com/docs/graph-api?locale=zh\\_TW](https://developers.facebook.com/docs/graph-api?locale=zh_TW)
  - HTTPS – secured version.

# Web Hosting – Basics (5)

- URL – Uniform Resource Locator
  - Describe how to access an object shared on the Internet (RFC 1738)
  - Format
  - Protocol `://` [ [ username [ :password ] @ ] hostname [ :port ] ] [ /directory ] [ /filename ]
  - e.g.,
    - `http://www.cs.nctu.edu.tw/`
    - `ftp://ca.nctu.edu.tw/`
    - `telnet://ptt.cc/`

## WHERE

The file is on the machine localhost in the directory /page

<http://localhost/page/index.html>

## HOW

Hyper-Text Transfer Protocol

## WHAT

The file I want is index.html

# Web Hosting – Basics (6)

- URL Protocols

Proto	What it does	Example
http	Accesses a remote file via HTTP	<a href="http://www.cs.nctu.edu.tw">http://www.cs.nctu.edu.tw</a>
https	Accesses a remote file via HTTP/SSL	<a href="https://www.cs.nctu.edu.tw">https://www.cs.nctu.edu.tw</a>
ftp	Accesses a remote file via FTP	<a href="ftp://ftp.cs.nctu.edu.tw/">ftp://ftp.cs.nctu.edu.tw/</a>
file	Access a local file	<a href="file:///home/tsaimh/.tcshrc">file:///home/tsaimh/.tcshrc</a>
mailto	Sends mail	<a href="mailto:tsaimh@cs.nctu.edu.tw">mailto:tsaimh@cs.nctu.edu.tw</a>
news	Accesses Usenet newsgroups	<a href="news:tw.bbs.comp.386bsd">news:tw.bbs.comp.386bsd</a>

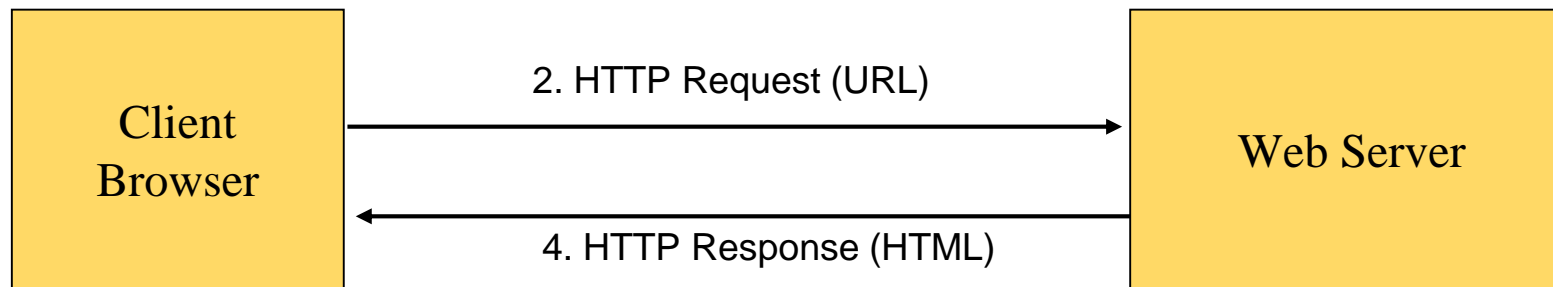


# Web Hosting – Client-Server Architecture (1)

- Client-server architecture
  - Web Server: Answer HTTP request
  - Web Client: Request certain page using URL

1. User enters a URL in browser

3. Search for the HTML resource pointed by URL



5. Show the data which HTML resource describes.

# Web Hosting – Client-Server Architecture (2)

- Using "telnet" to retrieve data from web server

```
$ telnet www.cs.nctu.edu.tw 80
Trying 140.113.235.47...
Connected to www.cs.nctu.edu.tw.
Escape character is '^]'.
GET /~tsaimh/sa.html HTTP/1.0

HTTP/1.1 200 OK
Server: nginx/0.7.62
Date: Sat, 12 Dec 2009 02:14:45 GMT
Content-Type: text/html
Connection: close
Last-Modified: Sat, 12 Dec 2009 02:14:09 GMT
Accept-Ranges: bytes
Content-Length: 201
Vary: Accept-Encoding

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01//EN">
<html>
  <head>
    <title>Hello World!</title>
  </head>
  <body>
    <p>Hello Wrold!</p>
  </body>
</html>
```

# Web Hosting – The HTTP Protocol (1)

- HTTP: **H**ypertext **T**ransfer **P**rotocol
  - RFCs: (HTTP 1.1)
    - <http://www.faqs.org/rfcs/rfc2068.html>
    - <http://www.faqs.org/rfcs/rfc2616.html> (Updated Version)
  - Useful Reference: <http://jmarshall.com/easy/http/>
  - A network protocol used to deliver virtually all files and other data on the World Wide Web.
    - HTML files, image files, query results, or anything else.
  - Client-Server Architecture
    - A browser is an HTTP client because it sends requests to an HTTP server (Web server), which then sends responses back to the client.

# Web Hosting – The HTTP Protocol (2)

- Clients:

- ※ Send Requests to Servers

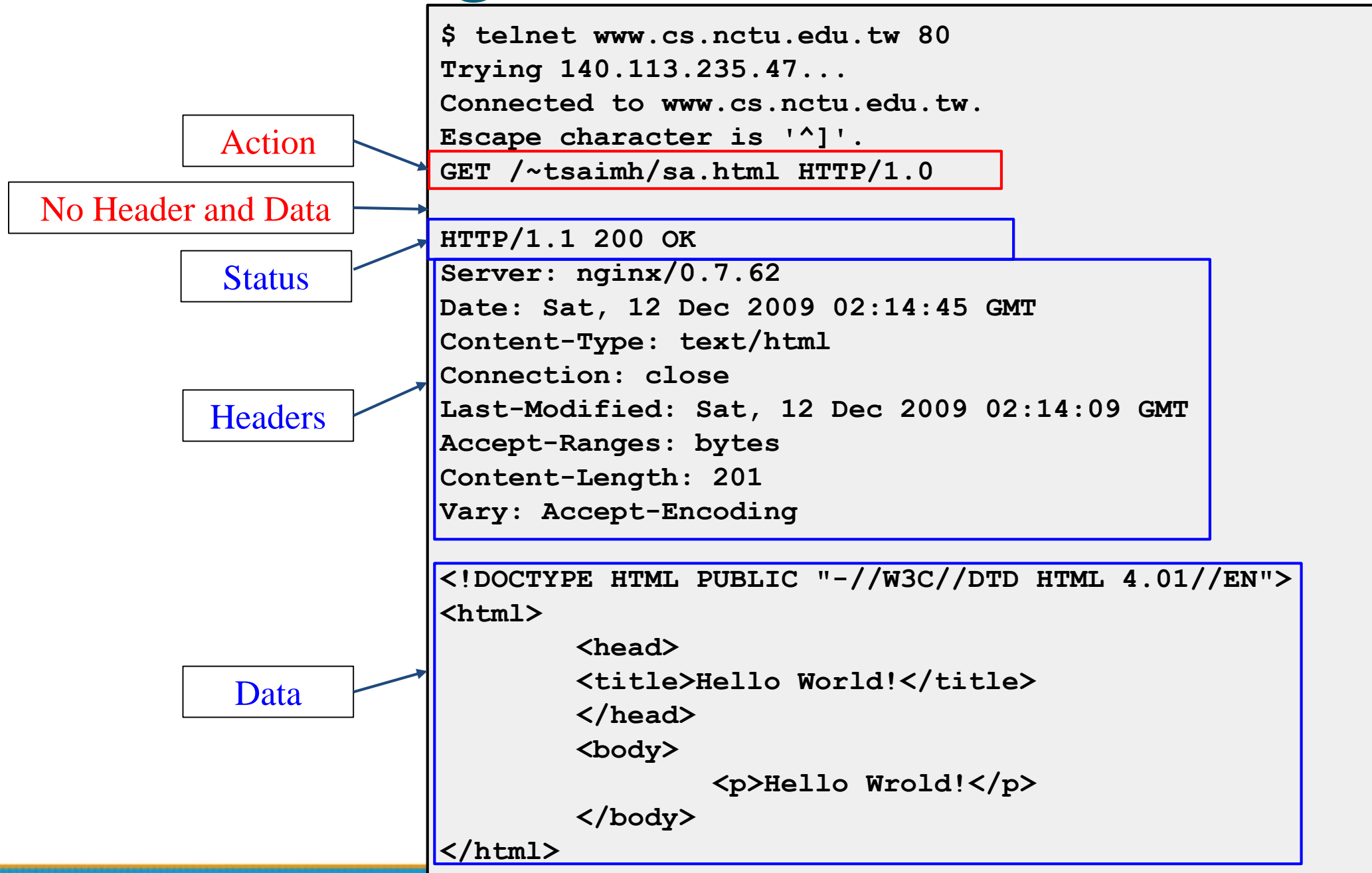
- Action "path or URL" Protocol
  - Actions: GET, POST, HEAD
  - e.g. : GET /index.php HTTP/1.1
- Headers
  - Header\_Name: value
  - e.g. : Host: [www.cs.nctu.edu.tw](http://www.cs.nctu.edu.tw)
- (blank line)
- Data ...

- Servers:

- ※ Respond to the clients

- Status:
  - 200: OK
  - 403: Forbidden
  - 404: Not Found
  - 426: Upgrade Required
  - ...
  - e.g. : HTTP/1.1 200 OK
- Headers
  - Same as clients
  - e.g. : Content-Type: text/html
- (blank line)
- Data...

# Web Hosting – The HTTP Protocol (3)



# Web Hosting – The HTTP Protocol (4)

- GET vs. POST (client side)

- GET:

- Parameters in URL: GET /get.php?a=1&b=3 HTTP/1.1

- No data content

- Corresponding in HTML files

- Link URL: <http://nasa.cs.nctu.edu.tw/get.php?a=1&b=3>

- Using Form: <form method="GET" action="get.php"> ... </form>

- POST:

- Parameters in Data Content

- POST /post.php HTTP/1.1

- Corresponding in HTML files

- Using Form: <form method="POST" action="post.php"> ... </form>

# Web Hosting – The HTTP Protocol (5)

- Get vs. Post Security Issue
  - GET:
    - GET requests can be cached
    - GET requests remain in the browser history
    - GET requests can be bookmarked
    - GET requests should never be used when dealing with sensitive data
    - GET requests have length restrictions
    - GET requests should be used only to retrieve data
  - POST:
    - POST requests are never cached
    - POST requests do not remain in the browser history
    - POST requests cannot be bookmarked
    - POST requests have no restrictions on data length

# Web Hosting – The HTTP Protocol (6)

	<b>GET</b>	<b>POST</b>
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)
Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL  Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL



# Web Hosting – The HTTP Protocol (7)

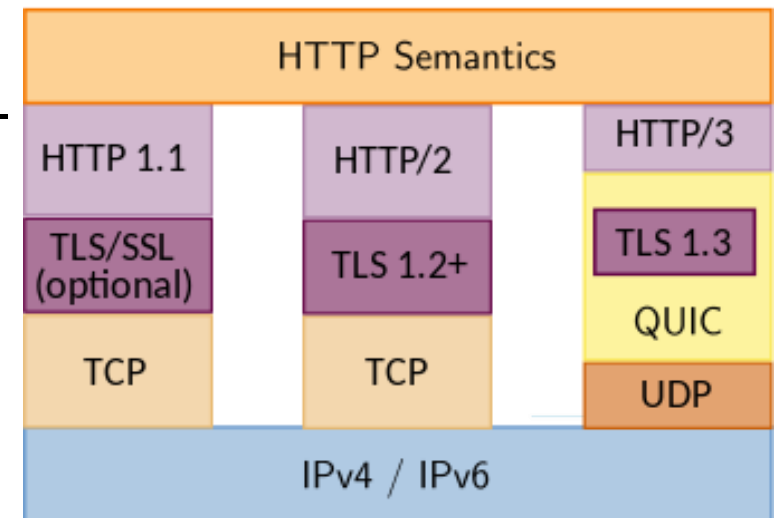
- HTTP Headers:
  - What HTTP Headers can do?
    - Content information (type, date, size, encoding, ...)
    - Cache control
    - Authentication
    - URL Redirection
    - Transmitting cookies
    - Knowing where client come from
    - Knowing what software client use
    - ...

# Web Hosting – The HTTP Protocol (7)

- HTTP/2:
  - RFC 7540
    - <https://tools.ietf.org/html/rfc7540>
  - Solve some problems of HTTP/1.1
    - Server Push
    - Multiplexing
      - Previously: domain sharding
      - Reuse TCP Connection
    - Smaller header
    - HPACK Compression

# Web Hosting – The HTTP Protocol (8)

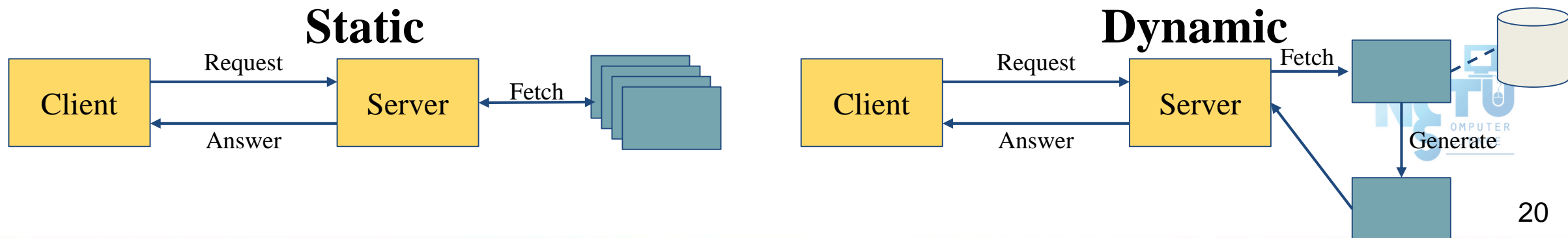
- HTTP/3:
  - RFC 9114. (Jun. 2022)
    - <https://tools.ietf.org/html/rfc9114>
  - Solve head-of-line blocking problems of HTTP/2.0 (over TCP)
    - Use UDP instead of TCP
    - lower latency and loads more quickly in real-world usage
    - in some cases over 3× faster than with HTTP/1.1 (which remains the only HTTP version deployed by many websites)



# Web Hosting – Static vs. Dynamic Pages

## (1)

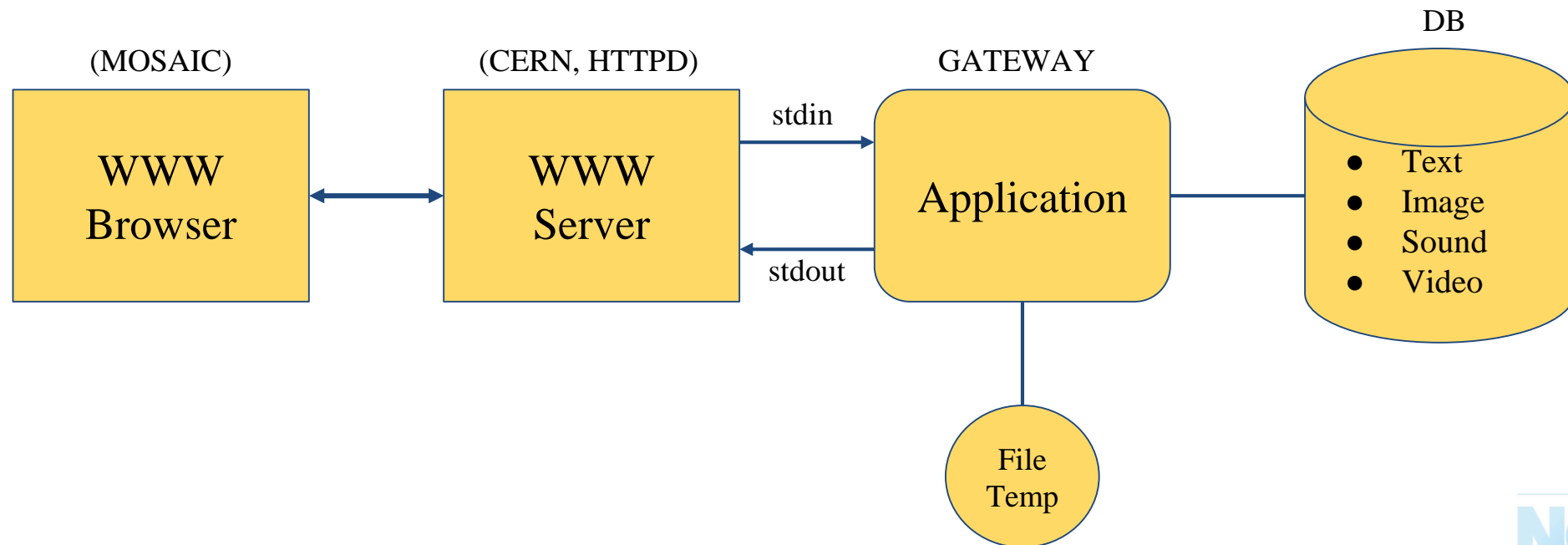
- Static vs. Dynamic Pages
  - Technologies of Dynamic Web Pages
    - Client Script Language
      - JavaScript, Jscript, VBScript
    - Client Interactive Technology
      - AJAX, SPA (Single Page Application)
    - Server Side
      - CGI
      - Languages: Perl, ASP, JSP, PHP, NodeJS, Golang, C/C++, ...etc.



# Web Hosting – Static vs. Dynamic Pages

## (2)

- CGI (Common Gateway Interface)
  - A specification that allows an HTTP server to exchange information with other programs



# Web Hosting – Virtual Hosting (1)

- Providing services for more than one domain-name (or IP) in one web server.
- IP-Based Virtual Hosting vs. Name-Based Virtual Hosting
- IP-Base – Several IPs (or ports)
- Name-Base – Single IP, several hostnames
- Example (Apache configuration)

```
NameVirtualHost 140.113.17.225  
  
<VirtualHost 140.113.17.225>  
ServerName nabsd.cs.nctu.edu.tw  
DocumentRoot "/www/na"  
</VirtualHost>  
  
<VirtualHost 140.113.17.225>  
ServerName sabsd.cs.nctu.edu.tw  
DocumentRoot "/www/sa"  
</VirtualHost>
```

```
<VirtualHost 140.113.17.215:80>  
DocumentRoot /www/sabsd  
ServerName sabsd.cs.nctu.edu.tw  
</VirtualHost>  
  
<VirtualHost 140.113.17.221:80>  
DocumentRoot /www/tphp  
ServerName tphp.cs.nctu.edu.tw  
</VirtualHost>
```

# Web Hosting – Virtual Hosting (2)

Q: How Name-Based Virtual Hosting works?

A: It takes use of HTTP Headers.

```
$ telnet www.cs.nctu.edu.tw 80
Trying 140.113.235.47...
Connected to www.cs.nctu.edu.tw.
Escape character is '^]'.
GET / HTTP/1.0
Host: www.cs.nctu.edu.tw

HTTP/1.1 301 Moved Permanently
Server: nginx/0.7.62
Date: Sat, 12 Dec 2009 02:50:22 GMT
Content-Type: text/html
Connection: close
Cache-Control: no-cache, must-revalidate
Location: cht/announcements/index.php
Vary: Accept-Encoding

Connection closed by foreign host.
```

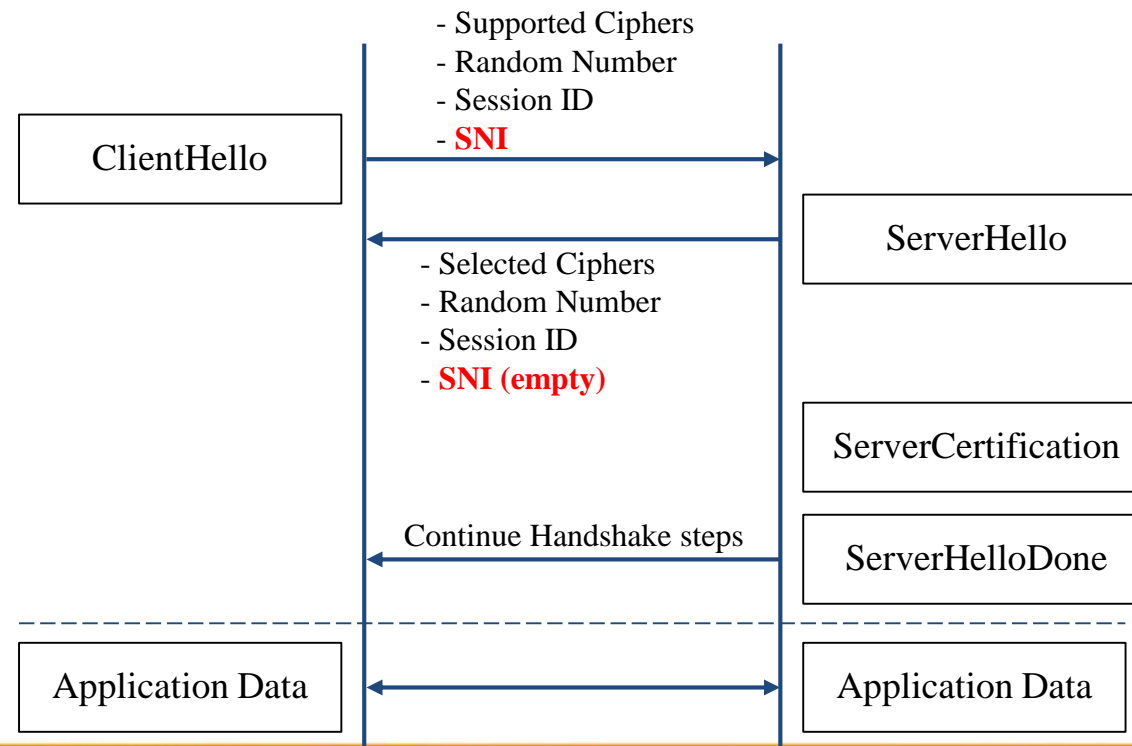
```
$ telnet www.cs.nctu.edu.tw 80
Trying 140.113.235.47...
Connected to www.cs.nctu.edu.tw.
Escape character is '^]'.
GET / HTTP/1.0
Host: www.ccs.nctu.edu.tw

HTTP/1.1 200 OK
Server: nginx/0.7.62
Date: Sat, 12 Dec 2009 02:51:43 GMT
Content-Type: text/html
Connection: close
Vary: Accept-Encoding

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html lang="zh-Hant">
<head>
    <meta http-equiv="content-type"
content="text/html; charset=utf-8">
    <title>國立交通大學資訊學院</title>
    ...
```

# Web Hosting - Server Name Indication (SNI)

- An extension to the TLS protocol by which a client indicates which hostname it is attempting to connect to at the start of the handshaking process.
  - Use single IP Address for multiple SSL sites
- No encryption of hostname
  - Possible information leak





# Proxy

- Proxy

- A proxy server is a server which services the requests of its clients by:

- Making requests to other servers
- Caching some results for further same requests

- Goals:

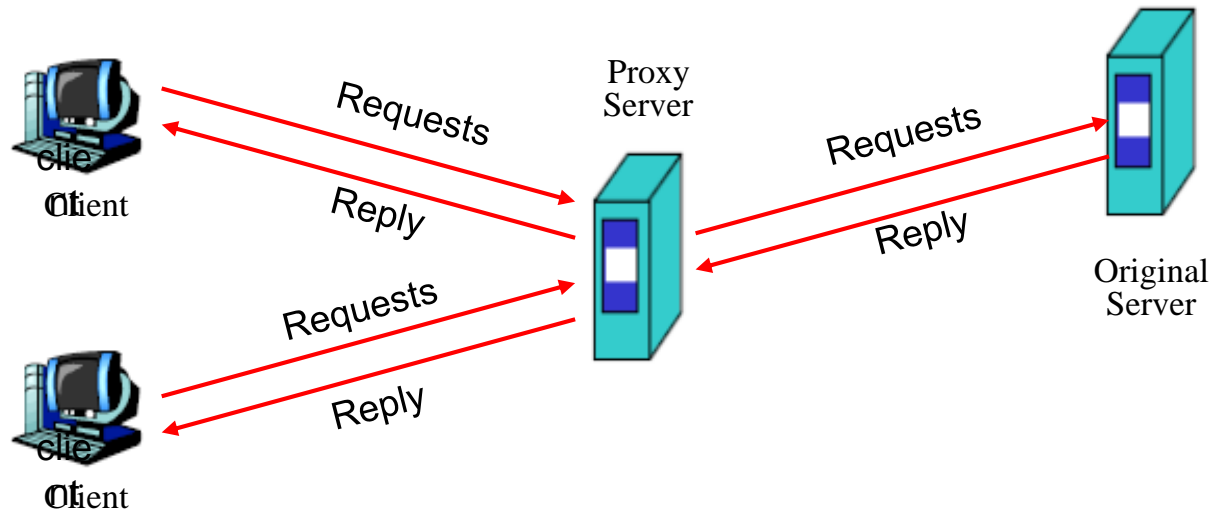
- Performance
- Stability
- Central Control
- ...etc.

- Roles:

- Forward Proxy
- Reverse Proxy

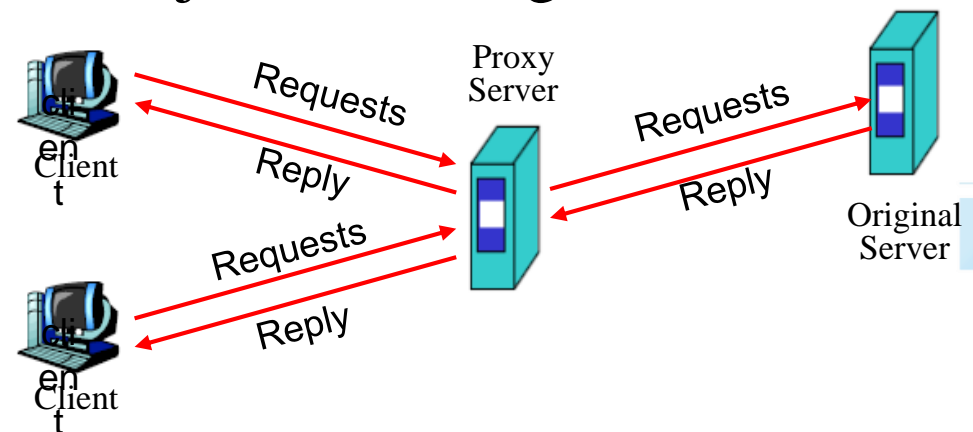
- Targets

- Web pages/FTP files
- TCP/IP Connections
- ...etc.



# Proxy – The Forward Proxy

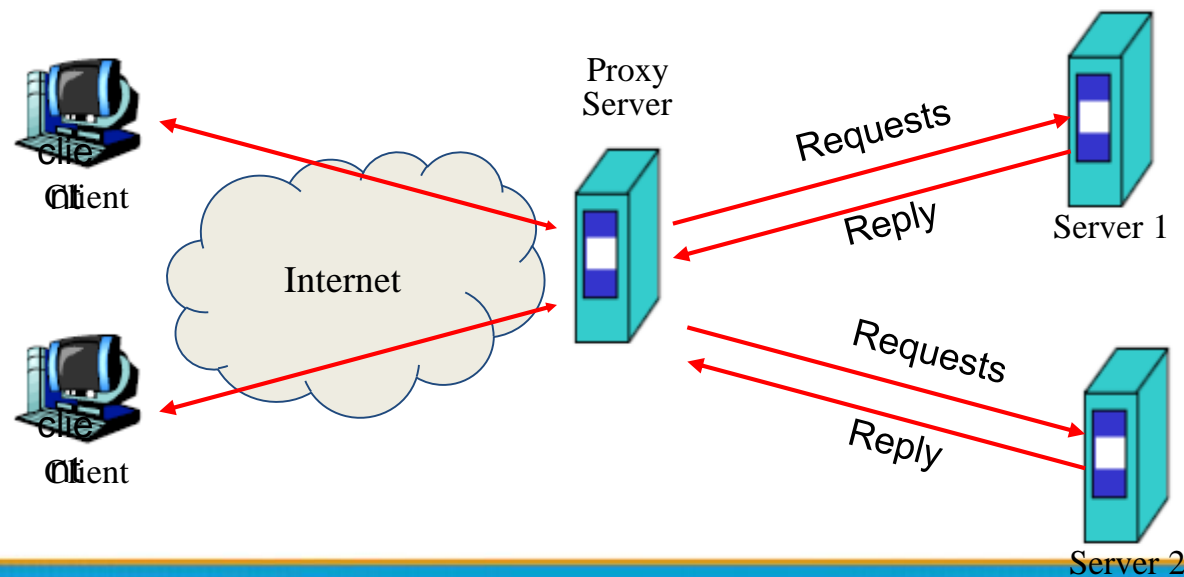
- Forward Proxy
  - Proxy the outgoing requests, for the reason of
    - Bandwidth saving
    - Performance
    - Central control
  - When objects requested are
    - In cache, return the cached objects
    - Otherwise, proxy server requests object from origin server, then cache it and return to client



# Proxy – The Reverse Proxy

- Reverse Proxy

- Proxy the incoming requests, for the reason of
  - Reducing Server Load (by caching)
  - Load Balance
  - Fault Tolerant
- Reverse proxy acts as the original server, accept incoming requests, reply corresponding result. **SEAMLESS for clients!**



# Proxy – The Reverse Proxy - Cont.

- Modem Hardware Server Load Balance
  - Application layer load balancing (L7)
  - Application layer service health check
  - Global server load balancing
  - SSL off load
  - Data acceleration
    - Cache
    - Compression (gzip)
  - Programmable server load balancing
    - F5 iRule
    - A10 aFlex