Web

hwlin1414

Outline

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

Web Hosting – Basics (1)

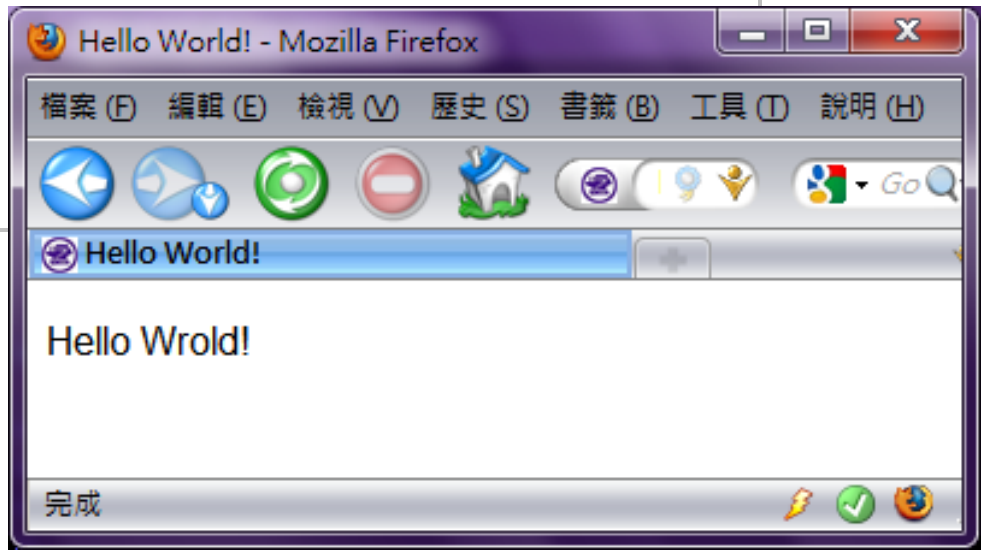
- Three major techniques in WWW (World Wide Web) System
  - HTML
  - HTTP
  - URL
- HTML (1) – HyperText Markup Language
  - Providing a means to describe the structure of text-based information in a document.
  - The original HTML is created by Tim Berners-Lee.
  - Published in 1993 by the IETF as a 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 Wrold!</p>
        </body>
</html>
```

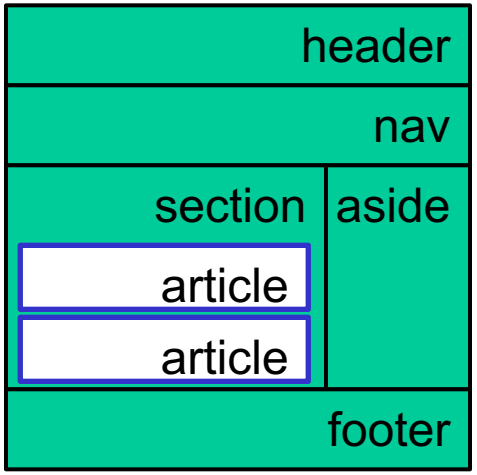


Web Hosting – Basics (3)

❑ HTML 5

- 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>



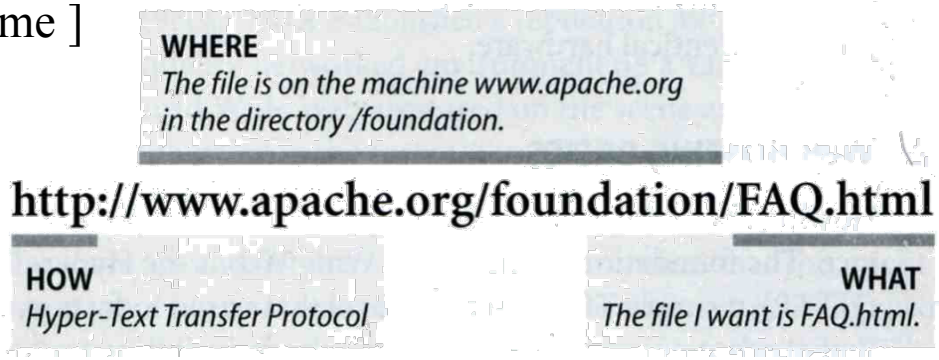
Web Hosting – Basics (4)

- HTTP – Hyper-Text Transfer Protocol
  - A TCP-based protocol
  - 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, ...
  - 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]



- ex:
  - http://www.cs.nctu.edu.tw/
  - ftp://ftp.cs.nctu.edu.tw/
  - telnet://bs2.to/

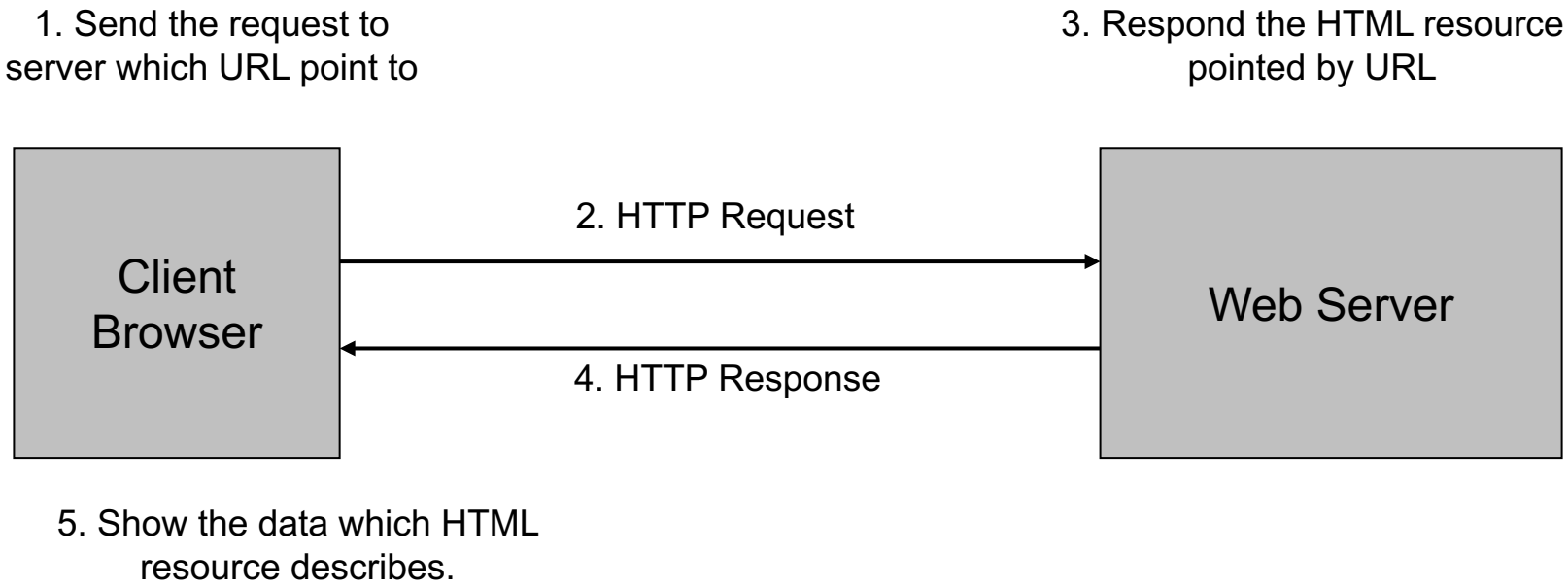
Web Hosting – Basics (6)

❑ URL Protocols

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

Web Hosting – Client-Server Architecture (1)

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



Web Hosting
– Client-Server Architecture (2)

- Using "telnet" to retrieve data from web server

```
liuyh@bsd5 ~/public_html $ telnet www.cs.nctu.edu.tw 80
Trying 140.113.235.47...
Connected to www.cs.nctu.edu.tw.
Escape character is '^]'.
GET /~liuyh/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: Hypertext Transfer Protocol
  - 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.

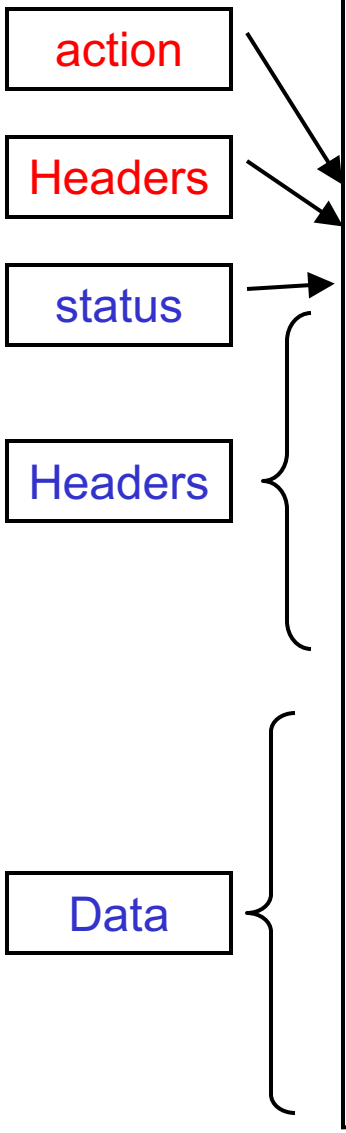
Web Hosting – The HTTP Protocol (2)

- Clients:
  - ※ Send Requests to Servers
    - Action "path or URL" Protocol
      - Actions: GET, POST, HEAD
      - Ex. GET /index.php HTTP/1.1
    - Headers
      - Header_Name: value
      - Ex.

      Host: www.cs.nctu.edu.tw
    - (blank line)
    - Data …
- Servers:
  - ※ Respond to the clinets
    - Status:
      - 200: OK
      - 403: Forbidden
      - 404: Not Found
      - 426: Upgrade Required
      - …
      - Ex. HTTP/1.1 200 OK
    - Headers
      - Same as clients
      - Ex.

      Content-Type: text/html
    - (blank line)
    - Data…

Web Hosting – The HTTP Protocol (3)




```
liuyh@bsd5 ~/public_html $ telnet www.cs.nctu.edu.tw 80
Trying 140.113.235.47...
Connected to www.cs.nctu.edu.tw.
Escape character is '^]'.
GET /~liuyh/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 (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

- https://www.w3schools.com/tags/ref_httpmethods.asp

Web Hosting – The HTTP Protocol (6)

	GET	POST
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

- https://www.w3schools.com/tags/ref_httpmethods.asp

Web Hosting – The HTTP Protocol (7)

❑ HTTP Headers:

- What HTTP Headers can do?

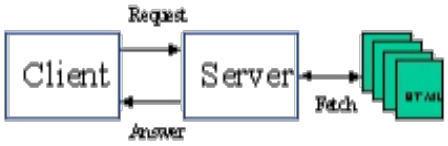
  [Ref] http://www.cs.tut.fi/~jkorpela/http.html

  - ➢ Content information (type, date, size, encoding, …)
  - ➢ Cache control
  - ➢ Authentication
  - ➢ URL Redirection
  - ➢ Transmitting cookies
  - ➢ Knowing where client come from
  - ➢ Knowing what software client use
  - ➢ …

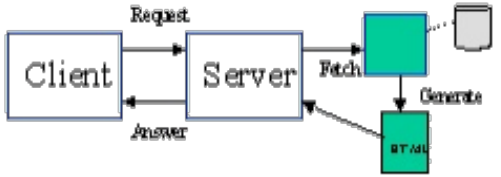
Web Hosting – Static vs. Dynamic Pages (1)

- Static vs. Dynamic Pages

Static vs. Dynamic



An HTML document stored in a file is a static Web page. Unless the file is edited, its content does not change.

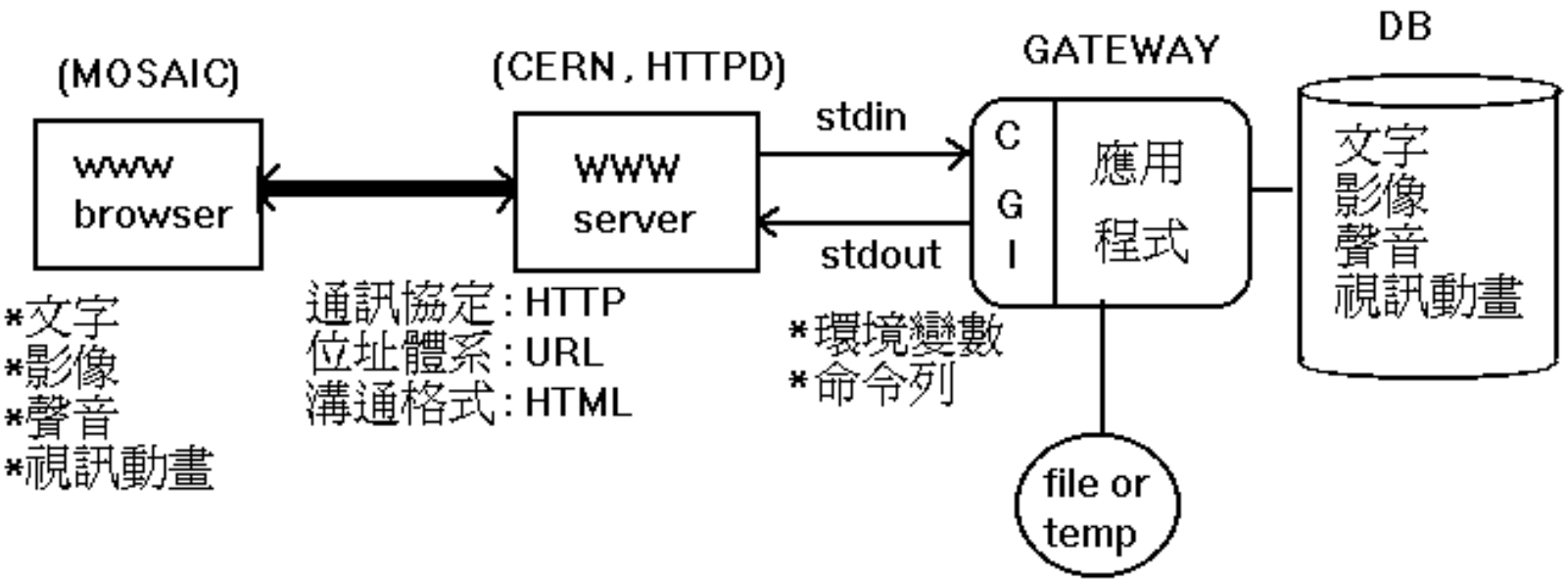


A dynamic Web page is generated or partially generated each time it is accessed.

- Technologies of Dynamic Web Pages
  - Client Script Language
    - JavaScript, Jscript, VBScript
  - Client Interactive Technology
    - Java Applet, Flash, XMLHTTP,AJAX
  - Server Side
    - CGI
    - Languages: Perl, ASP, JSP, PHP, 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



(圖 1)WWW主從架構應用示意圖

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 – Singe 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.
```

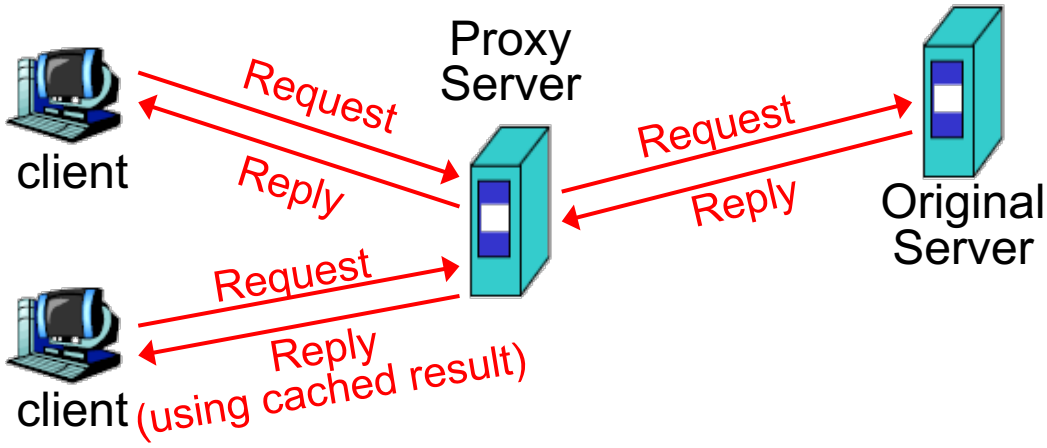
```
$ telnet www.cs.nctu.edu.tw 80
Trying 140.113.235.47...
Connected to www.cs.nctu.edu.tw.
Escape character is '^]'.
GET / HTTP/1.0
Host: www.ccs.nctu.edu.tw

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>
    ...
```

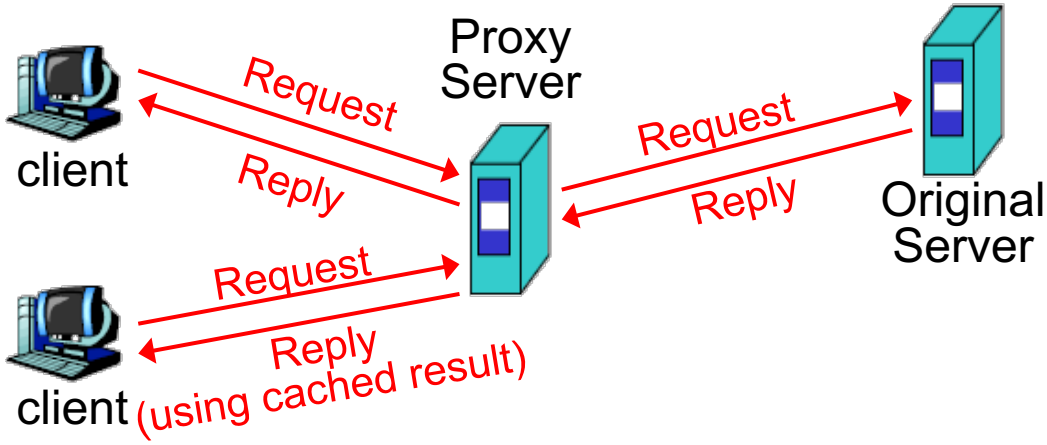
Proxy

- 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.



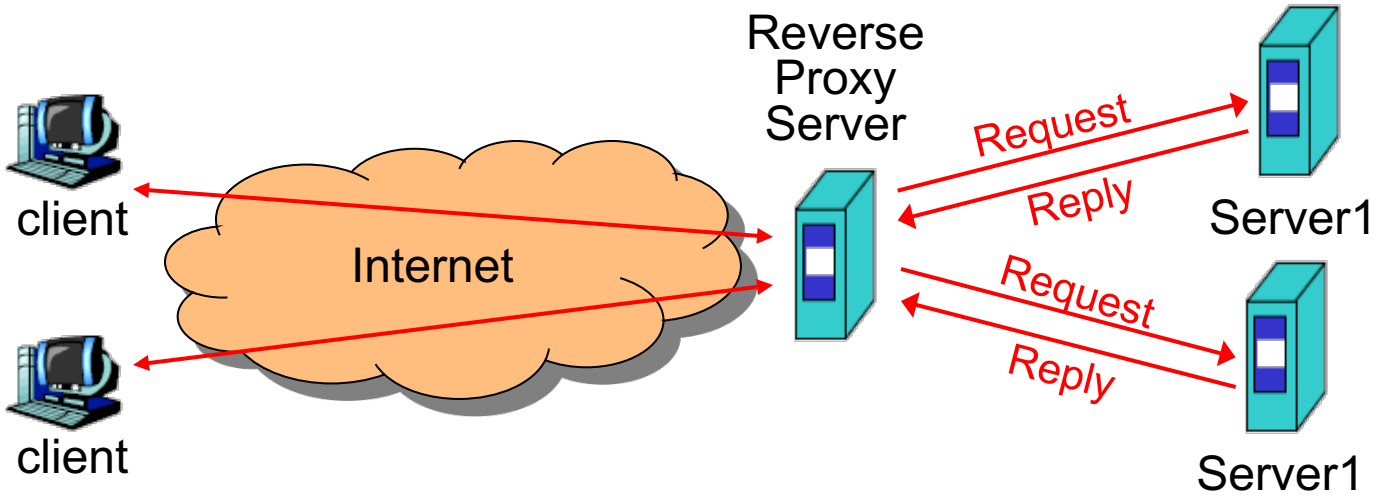
Proxy
– The Forward Proxy

- 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

Appendix

Proxy – SQUID

- A web proxy server & cache daemon.
  - Supports HTTP, FTP
  - Limited support for TLS, SSL, Gopher, HTTPS
- Port install: /usr/ports/www/squid{,30,31}
- Startup:
  - /etc/rc.conf
    - squid_enable="YES"
  - /usr/local/etc/rc.d/squid start
- Configuration Sample/Documents:
  - /usr/local/etc/squid/squid.conf.default

Proxy – SQUID Configuration (1)

- Listen Port
  - Service Port
    - http_port 3128
  - Neighbored Communication
    - icp_port 3130
- Logs
  - access_log
    - access_log /var/log/squid/access.log squid
  - cache_log
    - cache_log /var/log/squid/cache.log
  - cache_store_log
    - cache_store_log /var/log/squid/store.log

Proxy – SQUID Configuration (2)

❑ Access Control

- acl – define an access control list
  - ➢ Format: acl *acl-name acl-type data*

```
acl all src 0.0.0.0/0.0.0.0
acl NCTU srcdomain .nctu.edu.tw
acl YAHOO dstdomain .yahoo.com
acl allowhost src "/usr/local/etc/squid.squid.allow"
```

- http_access – define the control rule
  - ➢ Format: http_access *allow|deny acl-name*

```
http_access allow NCTU
http_access allow allowhost
http_access deny all
```

Proxy – SQUID Configuration (3)

- Proxy Relationship
  - Protocol: ICP (Internet Cache Protocol)
    RFC 2186 2187, using *UDP*
  - Related Configuration
    - cache_peer *hostname type http_port icp_port [options]*
    - cache_peer_domain *cache-host domain [domain ...]*
    - cache_peer_access *cache-host allow|deny acl-name*

Proxy – SQUID Configuration (4)

- Cache Control
  - cache_mem 256 MB
  - cache_dir ufs /usr/local/squid/cache 100 16 256
  - cache_swap_low 93
  - cache_swap_high 98
  - maximum_object_size 4096 KB
  - maximum_object_size_in_memory 8 KB

Proxy – SQUID Configuration (5)

- Sample: Proxy Configuration

```
http_port 3128
icp_port 3130

cache_mem 32 MB
cache_dir ufs /usr/local/squid/cache 100 16 256

access_log /var/log/squid/access.log squid
cache_log /var/log/squid/cache.log
cache_store_log /var/log/squid/store.log
pid_filename /usr/local/squid/logs/squid.pid

visible_hostname nabsd.cs.nctu.edu.tw
acl allowhosts src "/usr/local/etc/squid/squid.allow “
http_access allow allowhosts
http_access deny all
```

Proxy – SQUID Configuration (6)

- Sample: Reverse Proxy Configuration

```
http_port 80 vhost
icp_port 3130

cache_mem 32 MB
cache_dir ufs /usr/local/squid/cache 100 16 256

access_log /var/log/squid/access.log squid
cache_log /var/log/squid/cache.log
cache_store_log /var/log/squid/store.log
pid_filename /usr/local/squid/logs/squid.pid

visible_hostname nabsd.cs.nctu.edu.tw
url_rewrite_program /usr/local/squid/bin/redirect.sh
acl cswww dstdomain csws1 csws2
http_access allow all cswww
always_direct allow cswww
```

Proxy
– SQUID Configuration (7)

```
% cat /usr/local/squid/bin/redirect.sh

#!/bin/sh

while read line
do
    TIME=`date "+%S"`
    SERV=`expr $TIME % 2 + 1`
    echo $line | sed -e  \ "s/^http:\/\/www\.cs\.nctu\.edu\.tw\//http:\/\/csws$SERV\.cs\.nctu\.edu\.tw\//"
done
```