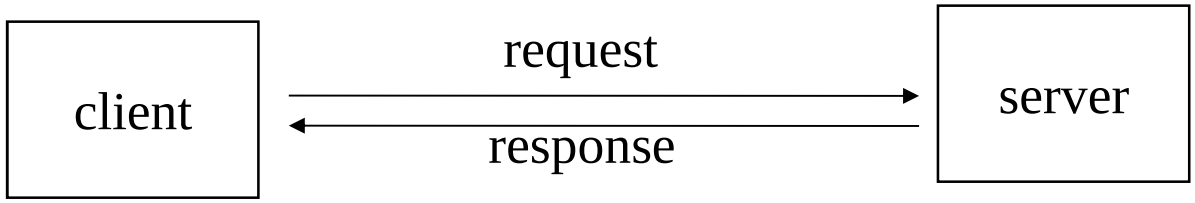
Proxy server implementation

Client and server

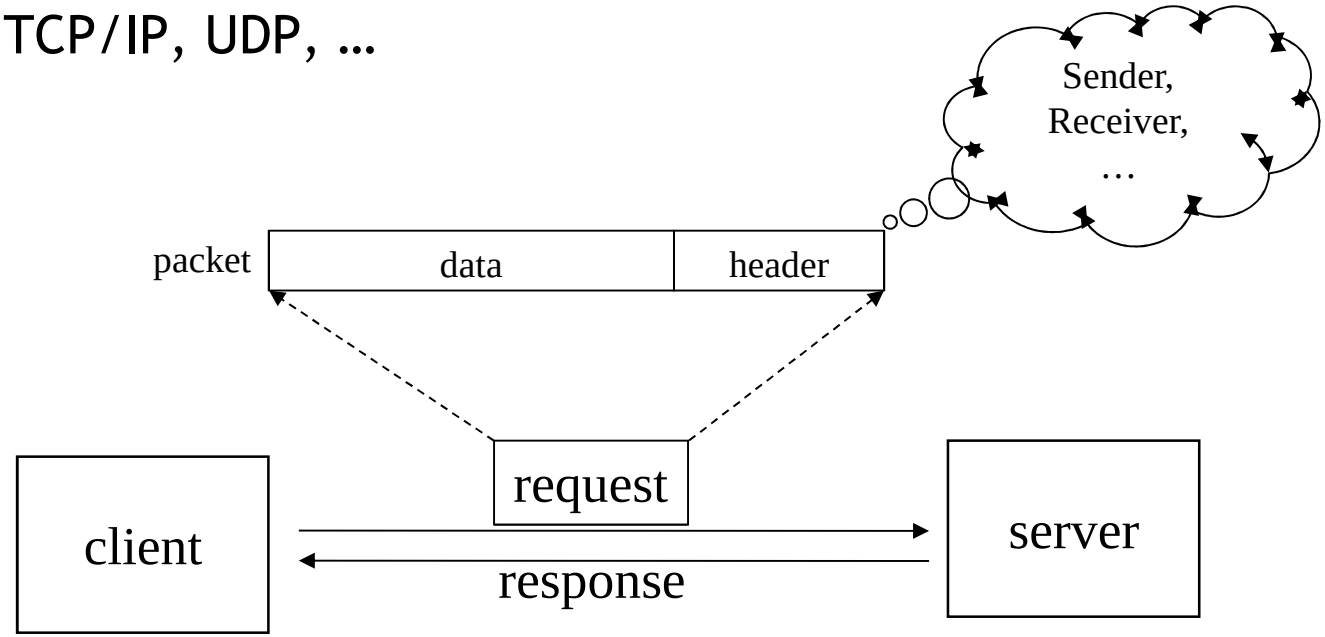
- Network programming
  - involves writing programs that communicate with other programs across a computer network

- Most network applications is divided into two pieces.
  - Client
  - Server



Client and server

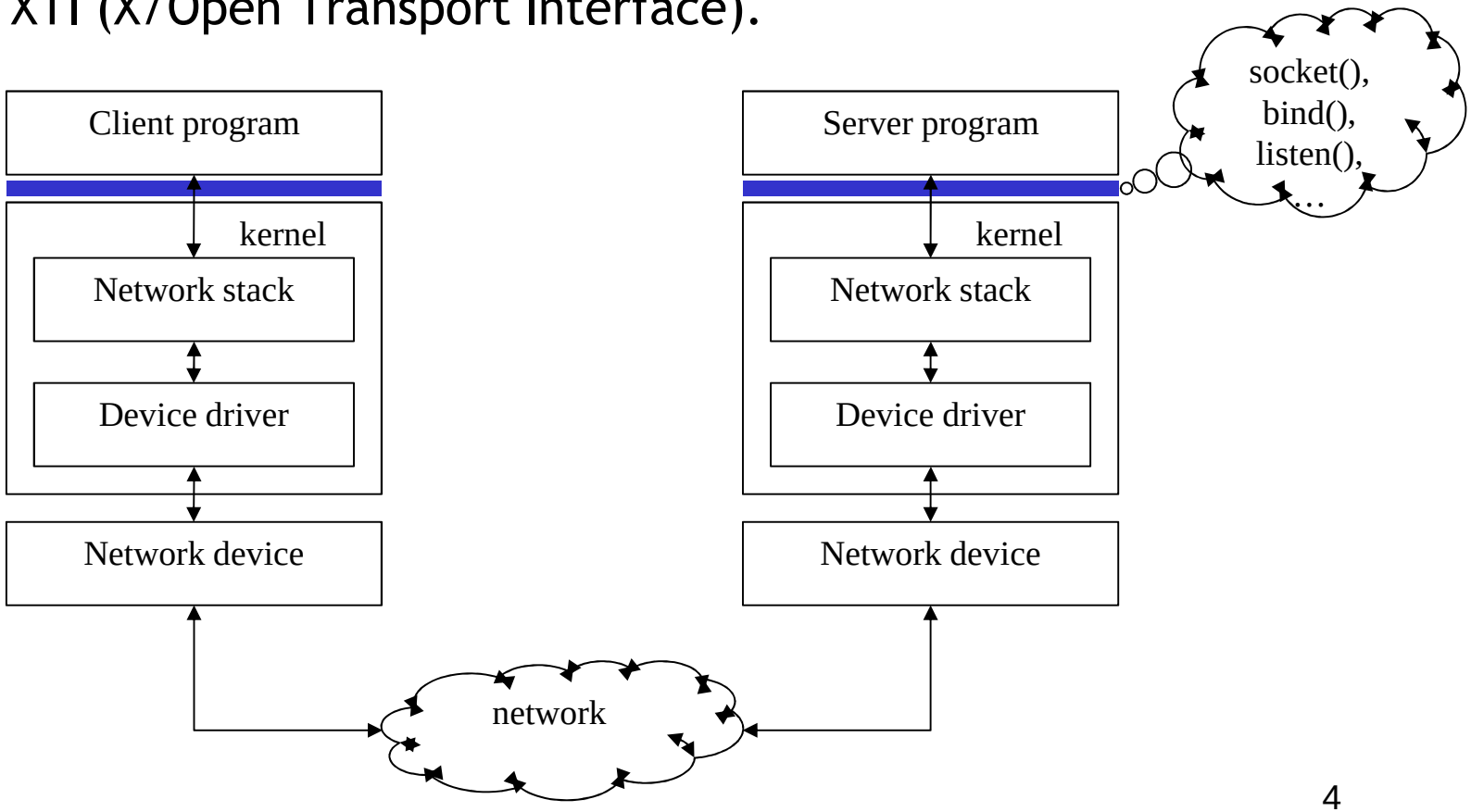
- Unix/Linux provides some low-level network services in kernel.
  - TCP/IP, UDP, …



You can learn the details in “Computer network”.

Client and server

- We just write network programs using APIs.
  - Socket (Berkeley sockets)
  - XTI (X/Open Transport Interface).

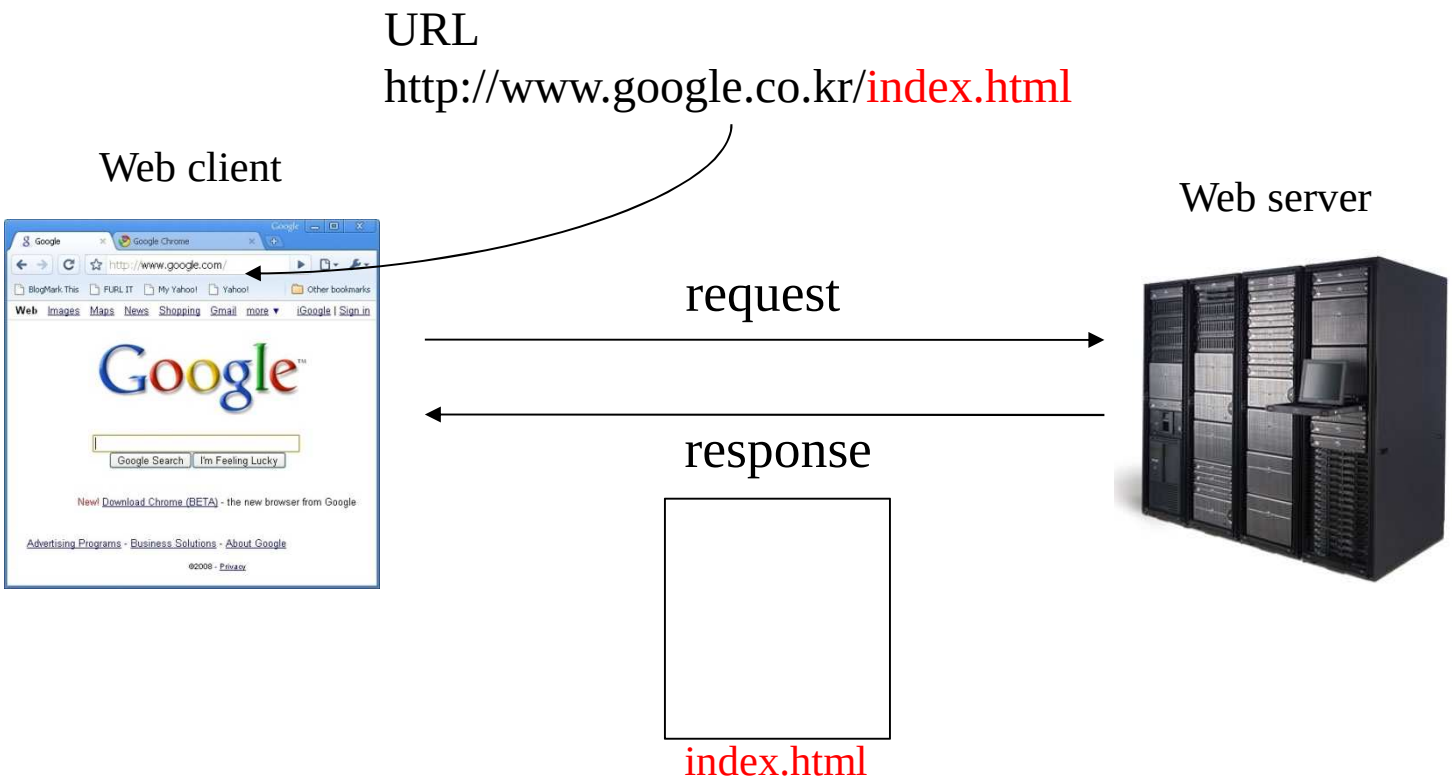


Client and server

- There are numerous examples of clients and servers.
  - Web client(browser) communicates with web server.
  - FTP client downloads a file from FTP server.
  - With telnet client, we can log in a remote host through a telnet server.
  - …
- Your project is to implement proxy server.

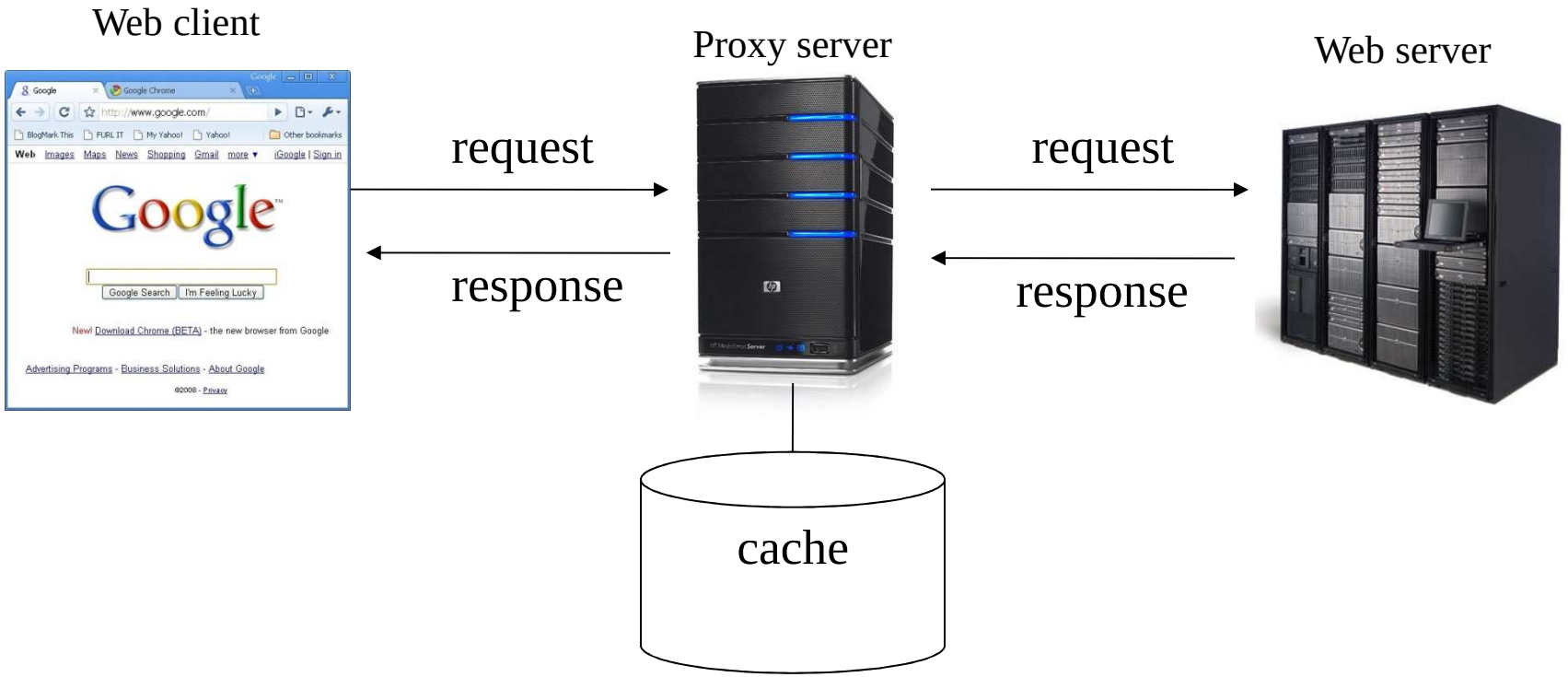
Web service

- HTTP protocol



Proxy server

- Proxy server

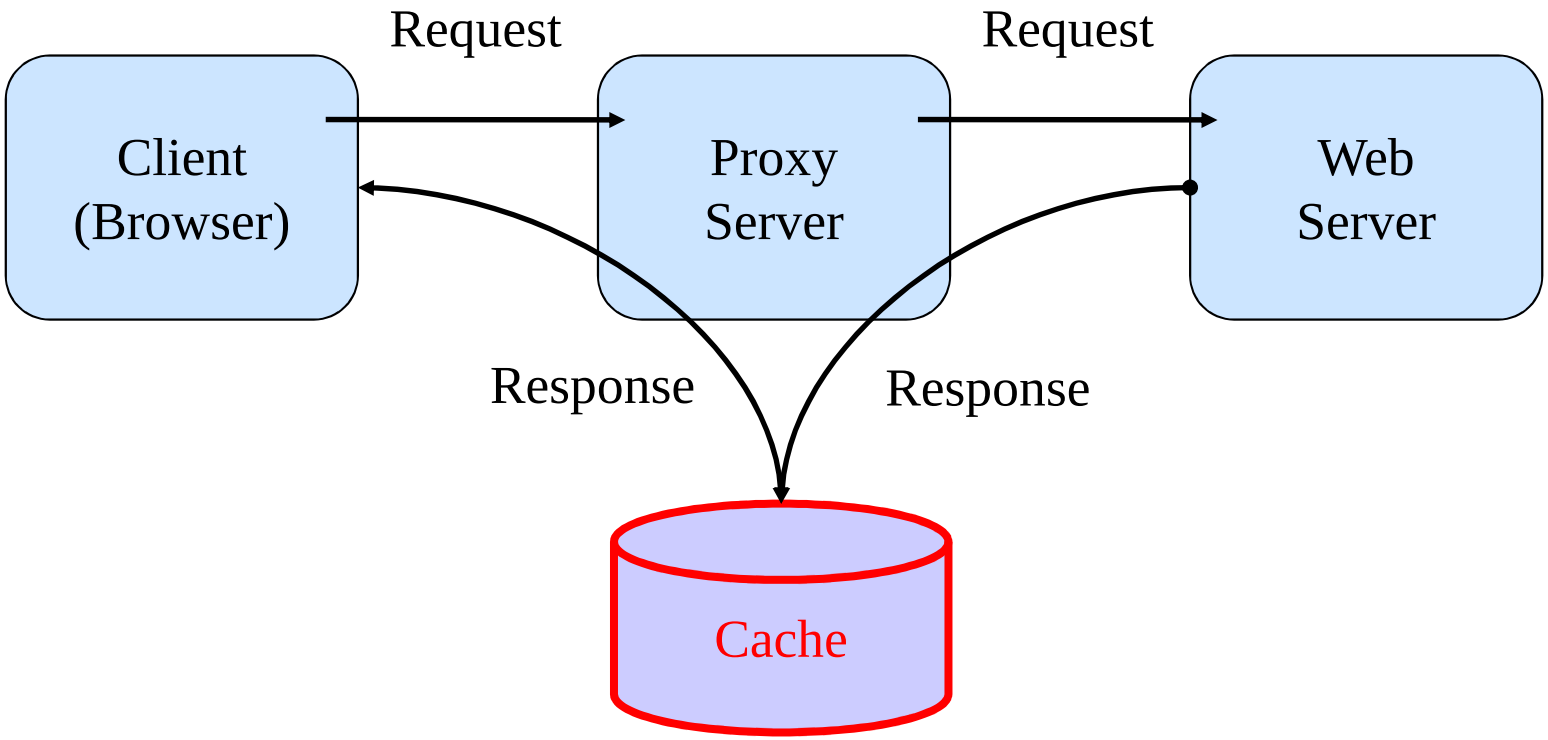


Proxy server

- Proxy server
  - 인터넷의 캐시 프로그램으로 최근에 가져온 자료들을 보관하는 저장장소를 의미
  - 같은 자료를 다시 요구할 경우 Proxy 내의 캐쉬에 있는 자료를 제공함으로써 자료가 빠르게 전송
  - 사용자는 Proxy를 사용해서 자료를 받아 오겠다는 사실을 호스트에 알려주어야 Proxy를 통해서 자료를 받는 것이 가능
  - 전송 속도가 빠를 수도 있지만 Proxy에 미리 저장되지 않은 자료를 찾는 경우는 더 느려질 수도 있다.

Proxy server implementation schedule

- 3-steps implementation

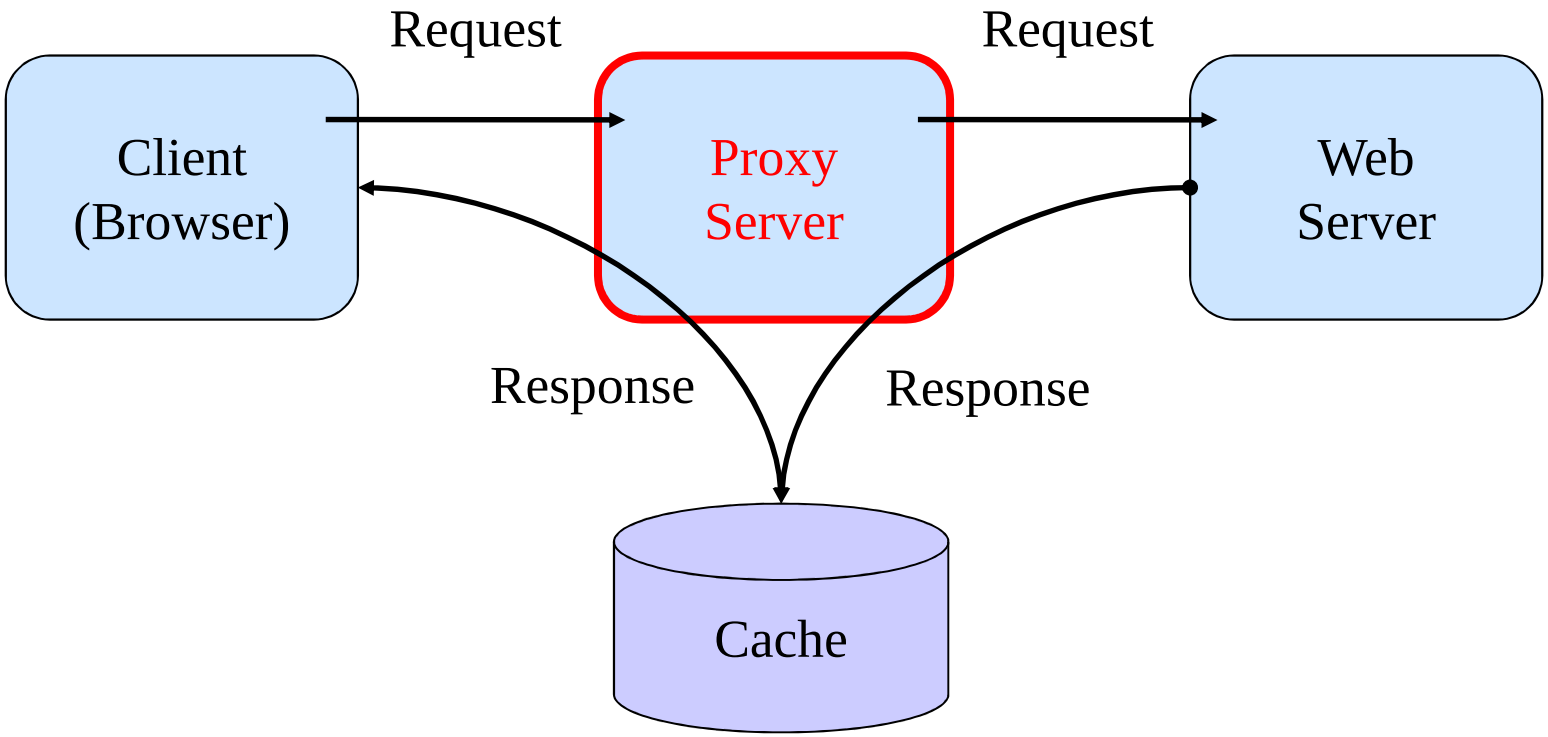


1. Cache implementation

Proxy server implementation schedule

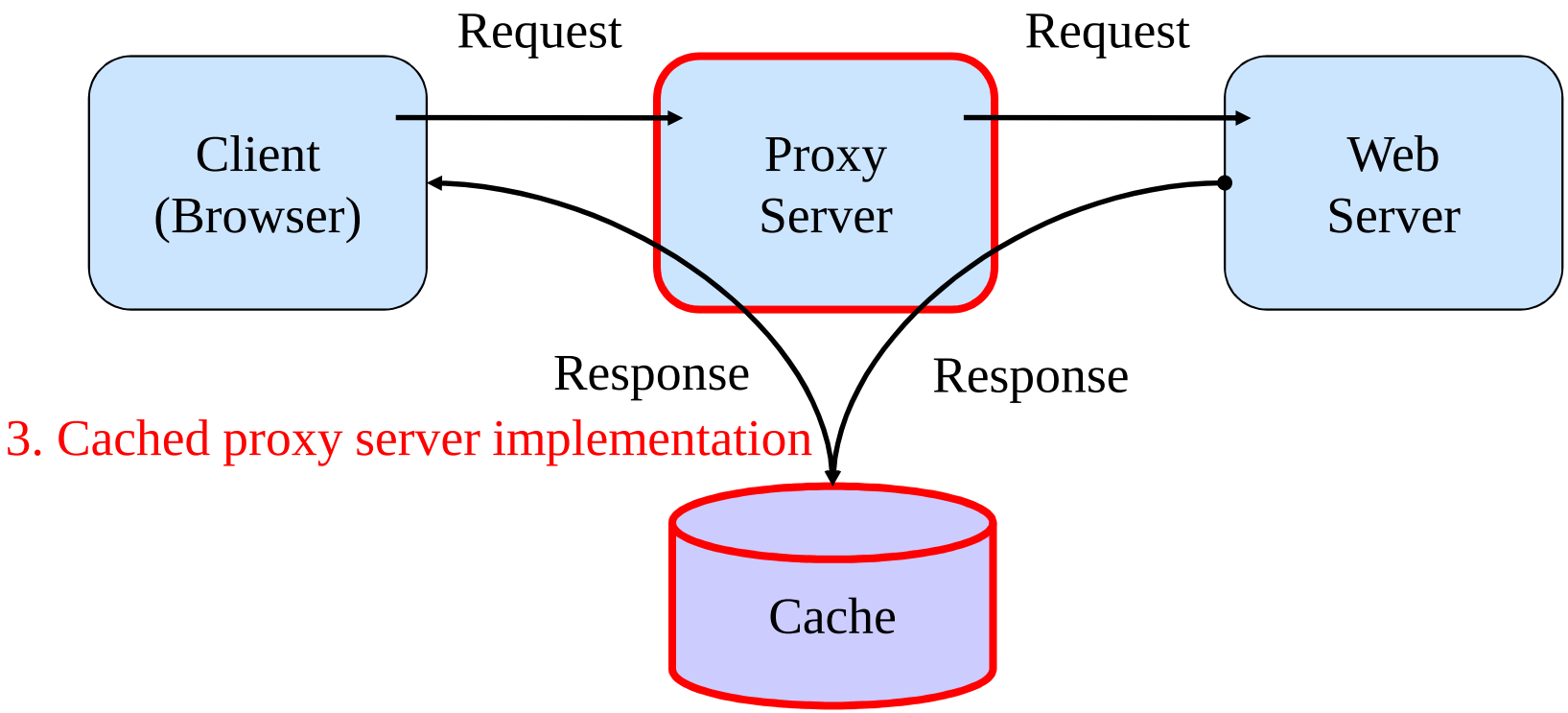
- 3-steps implementation

2. Proxy server implementation



Proxy server implementation schedule

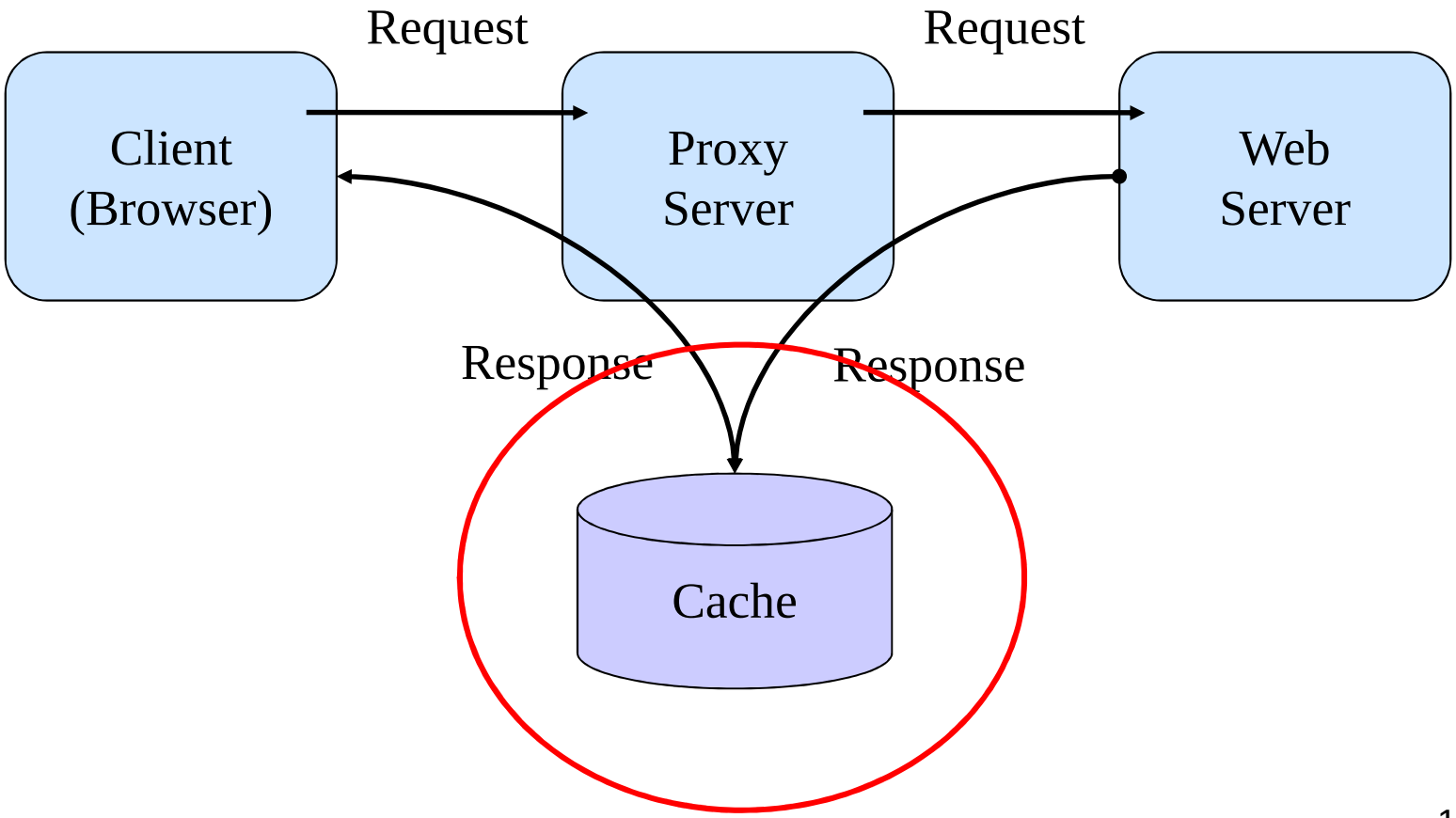
- 3-steps implementation



1. Cache implementation

Objective

- Cache implementation



Functionalities of a Cache

- Original server의 데이터를 저장함으로써 network delay를 줄임
- 요청된 data가 저장되어 있는지 검사
  - HIT: 클라이언트에 데이터 전송
  - MISS: Original server로부터 데이터를 받아서 cache에 저장, 클라이언트에 데이터 전달

Steps for cache implementation

- 표준입력(STDIN)으로부터 URL 입력
- 시스템으로부터 현재 시간 구함
- SHA-1(or MD5) 알고리즘을 사용하여 textual URL을 hashed URL로 변환
- Hashed URL을 이용하여 디렉토리와 파일 생성
- 파일에 URL과 현재 시간 기록
- Log file 생성
- Multiple processing

Operation

```
$ ./a.out
input URL> ce.kwangwoon.ac.kr/~sswlab
input URL> www.yahoo.co.kr
input URL> www.infoseek.com
input URL> ce.kwangwoon.ac.kr/~sswlab
input URL> www.ohmynews.com
input URL> bye
$
```

Steps for cache implementation

- 표준입력(STDIN)으로부터 URL 입력
- 시스템으로부터 현재 시간 구함
- SHA-1(or MD5) 알고리즘을 사용하여 textual URL을 hashed URL로 변환
- Hashed URL을 이용하여 디렉토리와 파일 생성
- 파일에 URL과 현재 시간 기록
- Log file 생성
- Multiple processing

Time and date

- Related APIs and structures
  - time_t time(time_t *calptr);
  - char *ctime(const time_t *calptr);
  - struct tm *gmtime(const time_t *calptr);
  - struct tm *localtime(const time_t *calptr);
  - tm structure

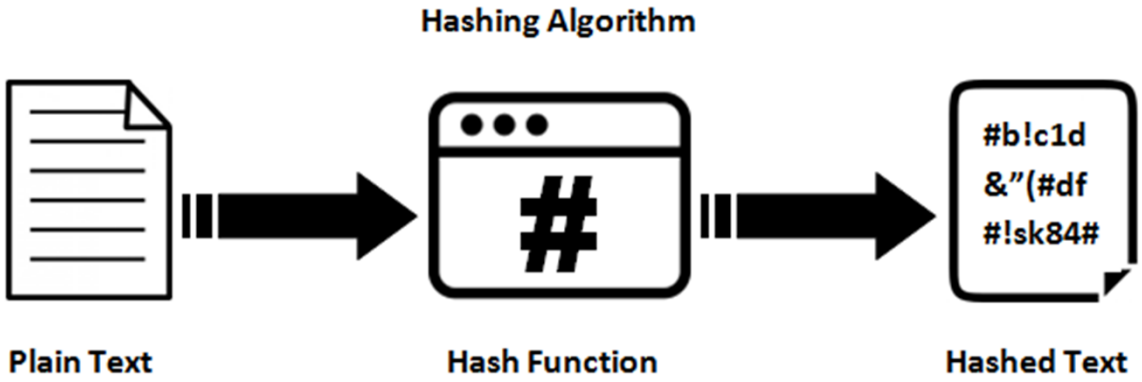
- You will learn details in “time and date” chapter.

Steps for cache implementation

- 표준입력(STDIN)으로부터 URL 입력
- 시스템으로부터 현재 시간 구함
- SHA-1(or MD5) 알고리즘을 사용하여 textual URL을 hashed URL로 변환
- Hashed URL을 이용하여 디렉토리와 파일 생성
- 파일에 URL과 현재 시간 기록
- Log file 생성
- Multiple processing

Hash function

- 해시 함수(hash function)는 임의의 길이의 데이터를 고정된 길이의 데이터로 매핑하는 함수이며 해시 함수에 의해 얻어지는 값은 해시 값, 해시 코드, 해시 체크섬 또는 간단하게 해시라고 함
- 해쉬함수의 종류로는 MD5, SHA, CRC32등이 있음



SHA-1

- SHA (Secure Hash Algorithm)
  - 미국 국립표준기술연구소(NIST)의 미국 연방정부 정보처리 표준.
  - 최초의 함수는 SHA이고, 이후 SHA-1, SHA-2, SHA-3등이 개발됨.

- SHA-1
  - 1993년 미국 국가안보국(NSA)에 의해 전자서명 알고리즘으로 개발
  - 여러 보안 프로토콜이나 프로그램에서 많이 사용됨.
    - E.g. SSH(Secure Shell), SSL(Secure Socket Layer), …
  - 입력한 text data(최대 2^{64}bits)를 160 bits의 hashed data로 반환함.

- In this assignment,
  - Cache에 저장하기 전에, URL을 hash 하기 위해 사용

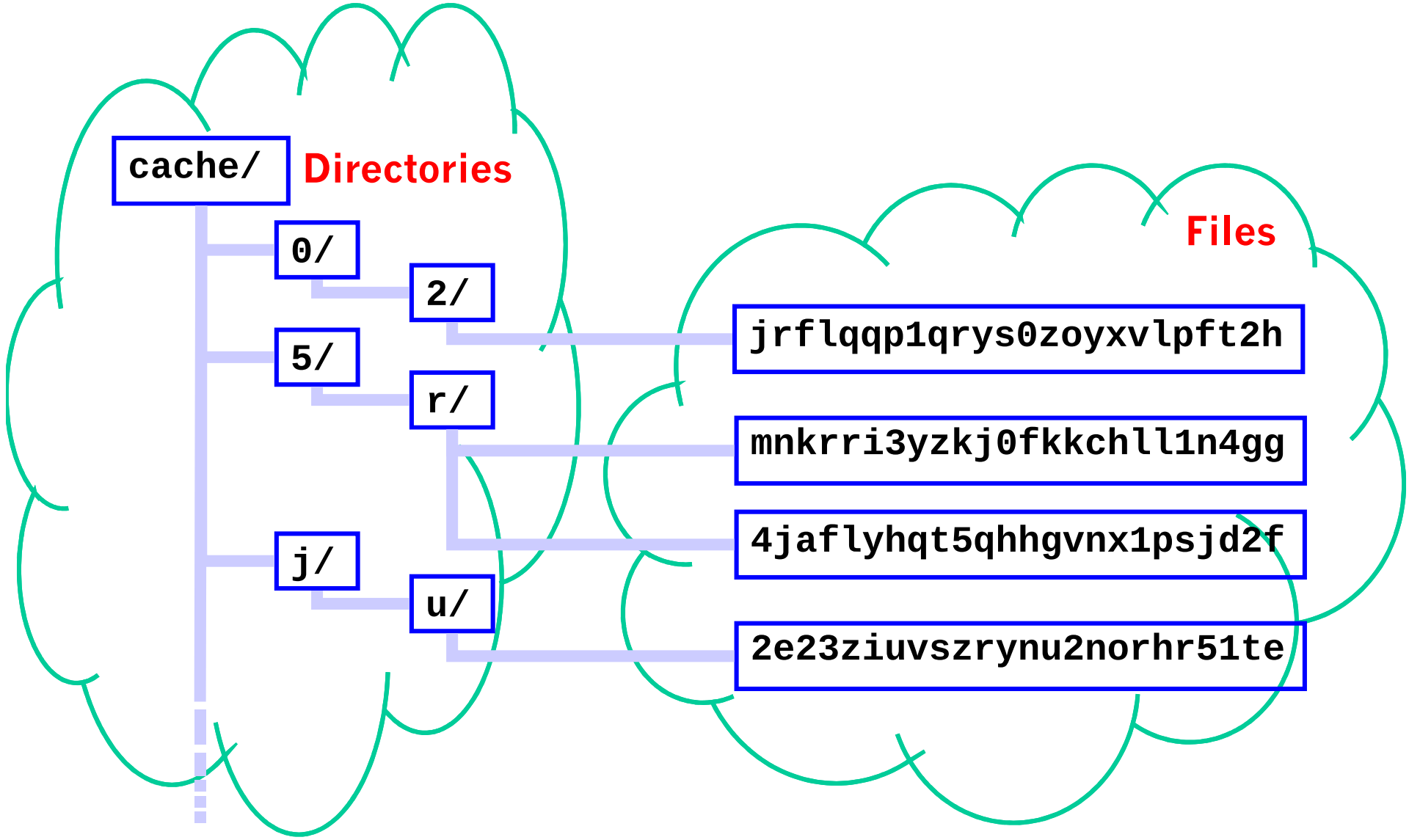
SHA-1을 이용한 URL Hashing

URL	Hashed URL
ce.kwangwoon.ac.kr/~sswlab	0/2/jrflqqp1qrys0zoyxvlpft2h
www.yahoo.co.kr	j/u/2e23ziuvszrynu2norhr51te
www.infoseek.com	5/r/mnkrri3yzkj0fkkchll1n4gg
ce.kwangwoon.ac.kr/~sswlab	0/2/jrflqqp1qrys0zoyxvlpft2h
www.ohmynews.com	5/r/4jaflyhqt5qhhgvnx1psjd2f

Steps for cache implementation

- 표준입력(STDIN)으로부터 URL 입력
- 시스템으로부터 현재 시간 구함
- SHA-1(or MD5) 알고리즘을 사용하여 textual URL을 hashed URL로 변환
- Hashed URL을 이용하여 디렉토리와 파일 생성
- 파일에 URL과 현재 시간 기록
- Log file 생성
- Multiple processing

Cached Directories and Files



Cache Directory

```
$ ls -R cache
cache:
0  5  j

cache/0:
2

cache/0/2:
jrflqqp1qrys0zoyxvlpft2h

cache/5:
r

cache/5/r:
4jaflyhqt5qhhgvnx1psjd2f  mnkrri3yzkj0fkkchll1n4gg

cache/j:
u

cache/j/u:
2e23ziuvszrynu2norhr51te
```

Output file

- Directory and file creation
  - MISS인 경우만 생성하면 됨.
    - Hashed URL에 따라 directory와 cached file을 구성한다.
    - Write URL information in cache files
  - Cache directories
    - Using hash function results
    - The root directory is “cache/”

Steps for cache implementation

- 표준입력(STDIN)으로부터 URL 입력
- 시스템으로부터 현재 시간 구함
- SHA-1(or MD5) 알고리즘을 사용하여 textual URL을 hashed URL로 변환
- Hashed URL을 이용하여 디렉토리와 파일 생성
- 파일에 URL과 현재 시간 기록
- Log file 생성
- Multiple processing

Logfile

```
$ cat logfile
ce.kwangwoon.ac.kr/~sswlab-[2012/2/23/ 20:5:7]
www.yahoo.co.kr-[2012/2/23/ 20:5:11]
www.infoseek.com-[2012/2/23/ 20:5:15]
0/2/jrflqqp1qrys0zoyxvlpft2h-[2012/2/23/ 20:5:20]
www.ohmynews.com-[2012/2/23/ 20:5:24]
$
```

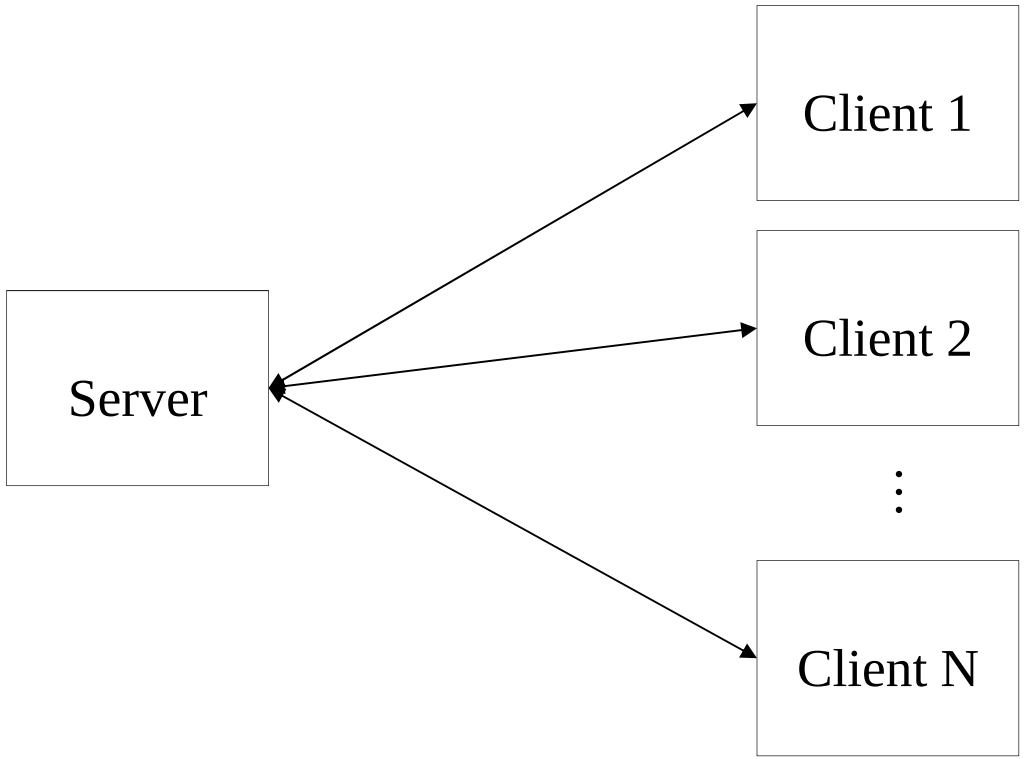
HIT인 경우
Hashed URL에 해당하는 디렉토리와 파일이 존재하는지 검사, **cached file**의 **path**를 구한 후 기록

Steps for cache implementation

- 표준입력(STDIN)으로부터 URL 입력
- 시스템으로부터 현재 시간 구함
- SHA-1(or MD5) 알고리즘을 사용하여 textual URL을 hashed URL로 변환
- Hashed URL을 이용하여 디렉토리와 파일 생성
- 파일에 URL과 현재 시간 기록
- Log file 생성
- Multiple processing

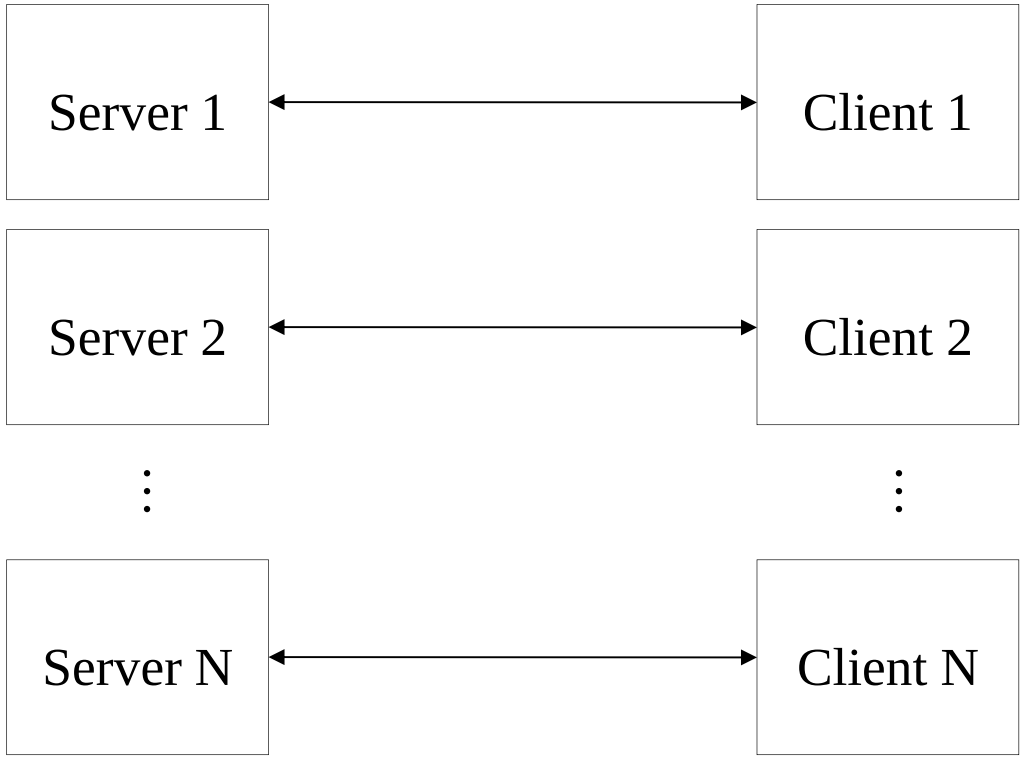
Concurrent server

- One-to-Many model
  - Conceptually, one server should service for many clients.
  - Eg. Web server, DB server, FTP server, …



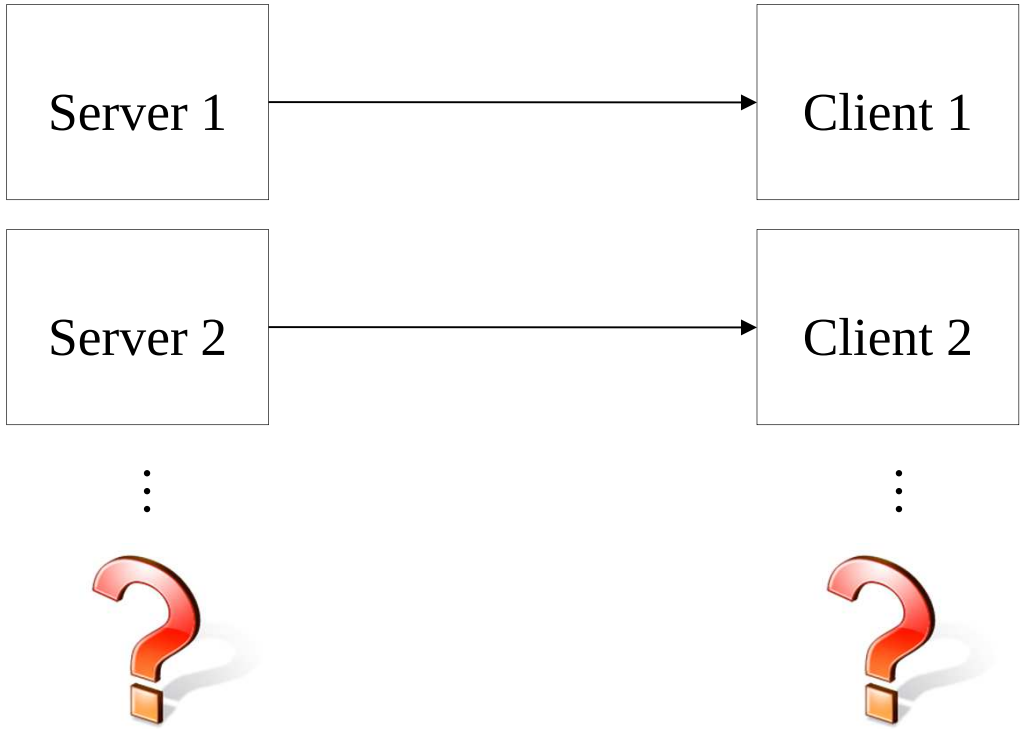
Concurrent server

- One-to-One relationship
  - Practically, one server can service for just one client.



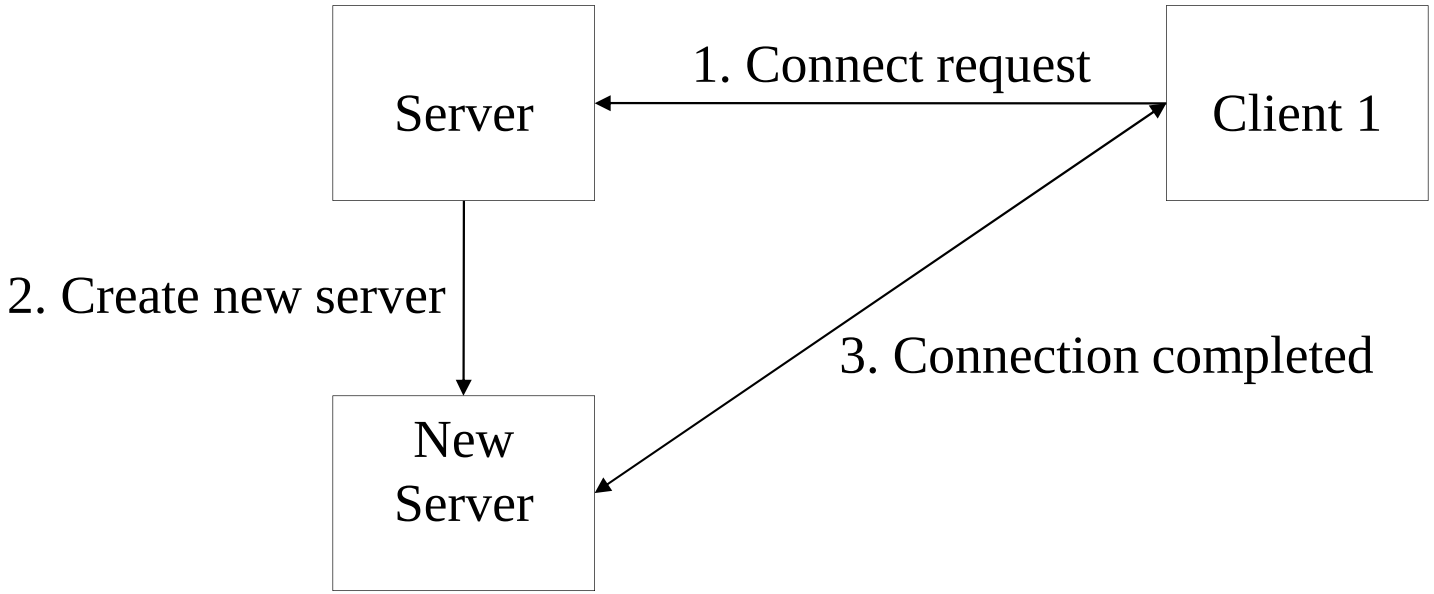
Concurrent server

- How many servers are required?
  - We don’t know how many clients want to connect.



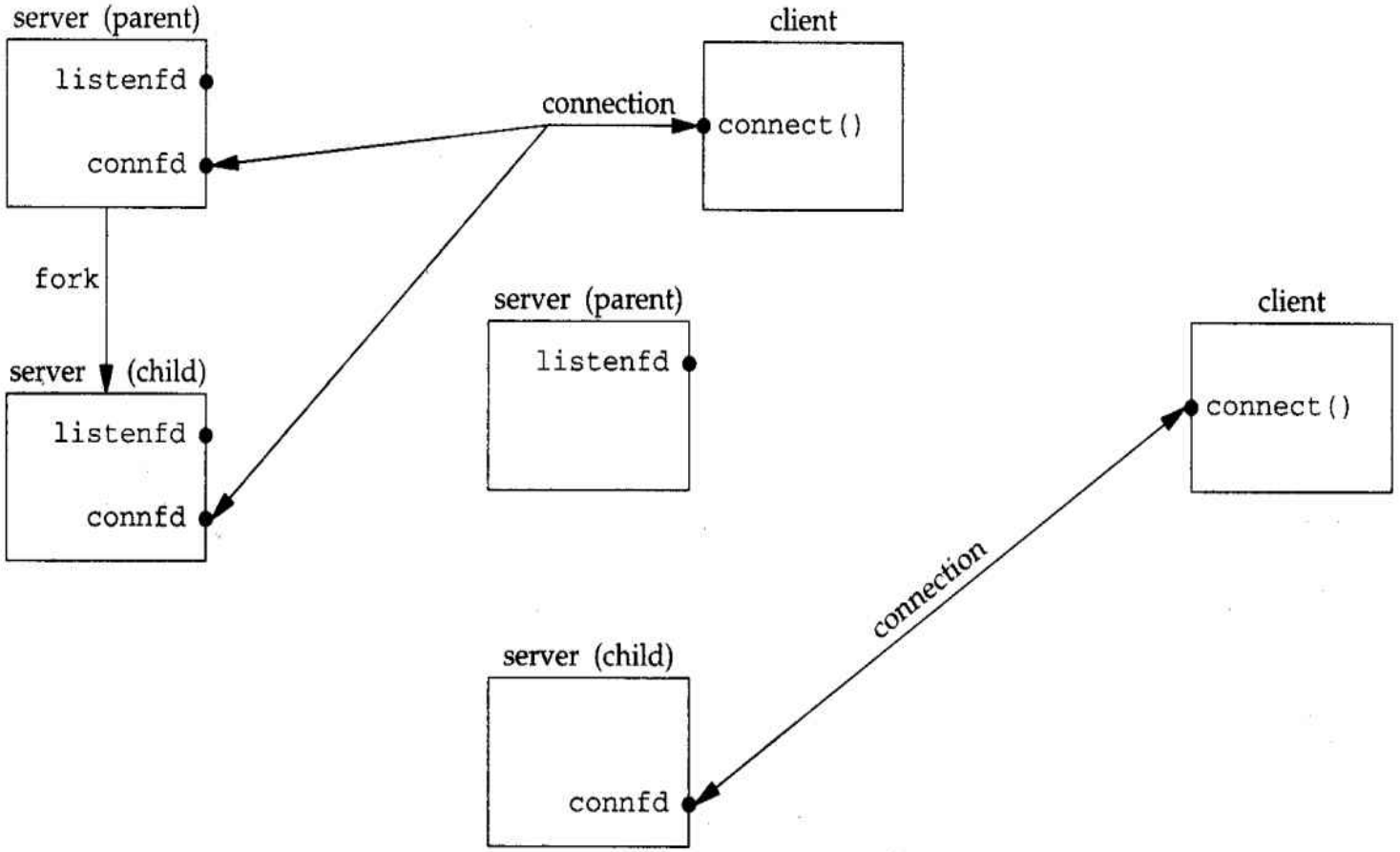
Concurrent server

- Server is created with an on-demand manner.
  - We create the server just when requested from client.



Concurrent server

- Concurrent server implementation with fork().
  - Process manipulation system calls (fork, exec, exit, wait,…)



Concurrent server

- Notification of termination by using signal.
  - When child terminates, parent is signaled.
  - signal(), kill(), …