수업 명 : 시스템프로그래밍

과제 이름 : Proxy 2-3

학 과: 컴퓨터정보공학부

담당 교수님: 김태석 교수님

분 반: 월5, 수6

학 번: 2023202043

성 명: 최은준

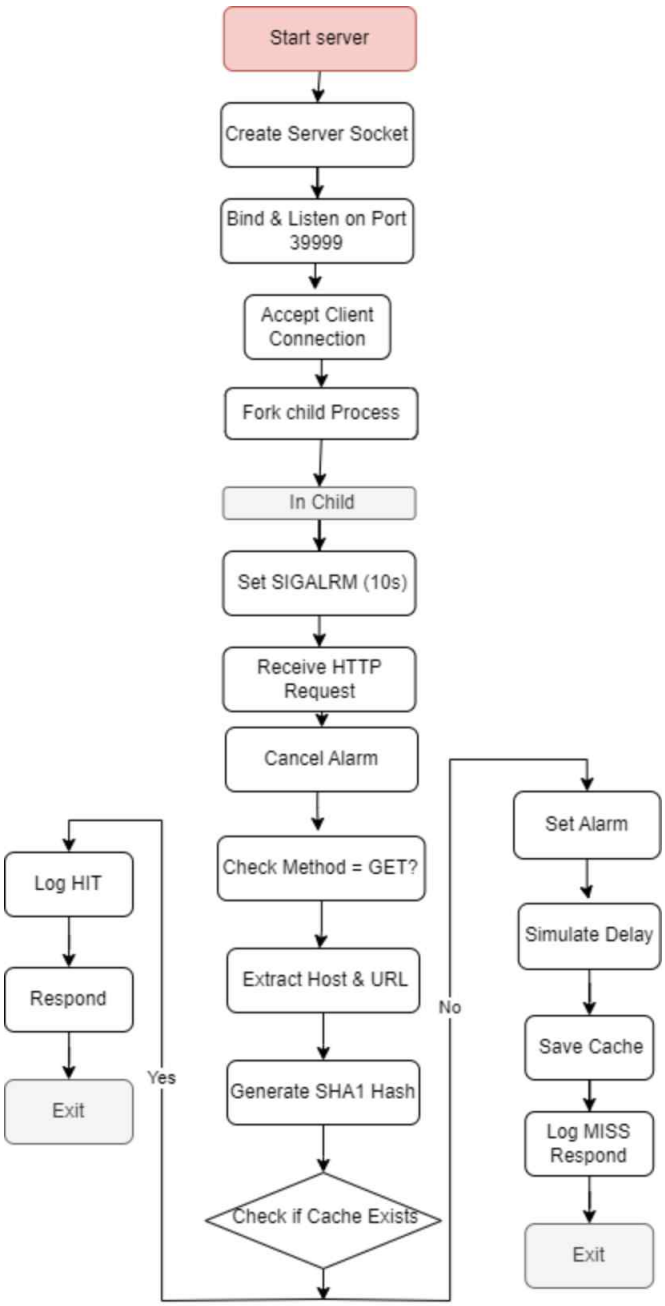
0. Introduction

이 과제에서 웹 브라우저를 클라이언트로 사용하는 프록시 서버를 구현하였다. 이때 프록시 서버는 클라이언트의 HTTP 요청을 수신하고, 해당 요청이 이전에 처리된 적이 있는지 캐시 디렉토리를 통해 판단한다. 요청한 URL이 캐시에 존재하면 HIT, 존재하지 않으면 MISS로 판단하여 그에 맞는 응답을 브라우저로 전송한다. 이 프록시 서버는 SHA1 해시 함수를 이용하여 URL을 해싱하고, 이를 기반으로 디렉토리 구조를 구성하여 캐시 파일을 관리한다. 클라이언트가 접속할 때마다 fork()를 이용해 자식 프로세스를 생성하고, 병렬 처리를 통해 다수의 요청을 동시에 처리할 수 있도록 설계되었다.

또한, 웹 서버 응답이 지연될 경우를 처리하기 위해 SIGALRM 신호를 사용하여 10초 내 응답이 없으면 해당 자식 프로세스를 종료시키도록 하였다. 본 과제를 통해 프록시 서버의 개념을 효과적으로 이해할 수 있다.

1. Flow Chart

1-1. proxy server #2-3 코드 작성 순서도



위 코드 순서도의 동작 방식은 다음과 같다.

1. 서버 소켓을 생성하여 포트 39999에 바인딩하고 리스닝 시작

2. 클라이언트의 요청을 accept()로 대기

3. 클라이언트 접속 시 fork()로 자식 프로세스 생성

4. 자식 프로세스는 10초 내에 HTTP 요청을 수신하지 않으면 종료

5. 요청이 "GET"이면 Host 헤더와 URL을 추출

6. Host+URL을 SHA1 해싱하여 캐시 경로 생성

7. 해당 파일이 캐시에 존재하면 HIT 처리: 로그 작성, 응답 전송

8. 존재하지 않으면 MISS 처리: sleep(13)으로 응답 지연 시도, 캐시 파일 생성, 로그 작성, 응답 전송

9. 연결 종료 후 자식 프로세스 종료

2. Pseudo code

2-1. proxy client #2-3 알고리즘

```
Function main():
    Set SIGCHLD handler to avoid zombies
    Set SIGALRM handler for timeout
    Create TCP server socket
    Set socket options (reuse address)
    Bind and listen on PORT
    While true:
        Accept client connection
        Fork child process
        If child:
            Close server socket
            Call handle_client()
        Else:
```

```
            Close client socket
    Close server socket
End Function
```

```
Function sigalrm_handler(signo):
    Print timeout message
    Exit process
End Function
```

```
Function sigchld_handler(signo):
    While there are terminated children:
        Reap with waitpid()
End Function
```

```
Function error_handling(msg):
    Print error using perror
    Exit process
End Function
```

```
Function sha1_hash(input, output):
    Compute SHA1 hash of input
    Convert hash to hex string
    Return output
End Function
```

```
Function is_cache_hit(path):
    Return whether file at path exists
End Function
```

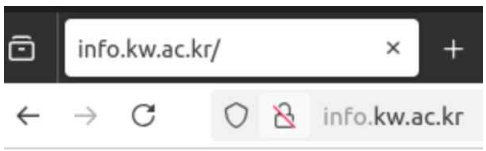
```
Function write_log(status, url, ip, port):
    Create log directory if not exist
    Open logfile in append mode
    Get current time
    Write log line with status, ip, port, url, and timestamp
    Close logfile
End Function
```

```
Function send_response(clnt_sock, status, ip, port, url):
    Build body and HTTP header
    Write header and body to client socket
End Function
```

```
Function handle_client(clnt_sock, clnt_addr):
    Set 10-second alarm
    Read HTTP request
    Cancel alarm
    Parse method and URL
    If method not GET:
        Close socket and exit
    Extract Host header
    If URL is for favicon or portal:
        Exit
    Print connection log
    Generate SHA1 hash of host
    Determine cache path
    If cache hit:
        Log HIT
        Send HIT response
    Else:
        Set alarm and sleep (simulate delay)
        Create cache file
        Log MISS
        Send MISS response
    Print disconnection log
    Close socket and exit
End Function
```

3. 결과화면

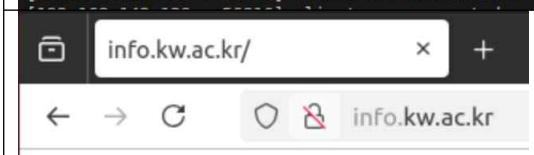
코드 작성 후 make 한 뒤 proxy_cache가 생성된 것을 확인할 수 있다.



MISS
192.168.142.129:33646
http://info.kw.ac.kr/
kw2023202043

```
kw2023202043@ubuntu:~/Proxy2-3$ ./proxy_cache
Proxy server is running on port 39999...
[192.168.142.129 : 33646] client was connected
================================================
Request from [192.168.142.129 : 33646]
GET http://info.kw.ac.kr/ HTTP/1.1
Host: info.kw.ac.kr
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:136.0) Gecko/2010010
1 Firefox/136.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Priority: u=0, i

================================================
[192.168.142.129 : 33646] client was disconnected
```



HIT
192.168.142.129:56816
http://info.kw.ac.kr/
kw2023202043

```
[192.168.142.129 : 33646] client was disconnected
[192.168.142.129 : 56816] client was connected
================================================
Request from [192.168.142.129 : 56816]
GET http://info.kw.ac.kr/ HTTP/1.1
Host: info.kw.ac.kr
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:136.0) Gecko/2010010
1 Firefox/136.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Priority: u=0, i

================================================
[192.168.142.129 : 56816] client was disconnected
```

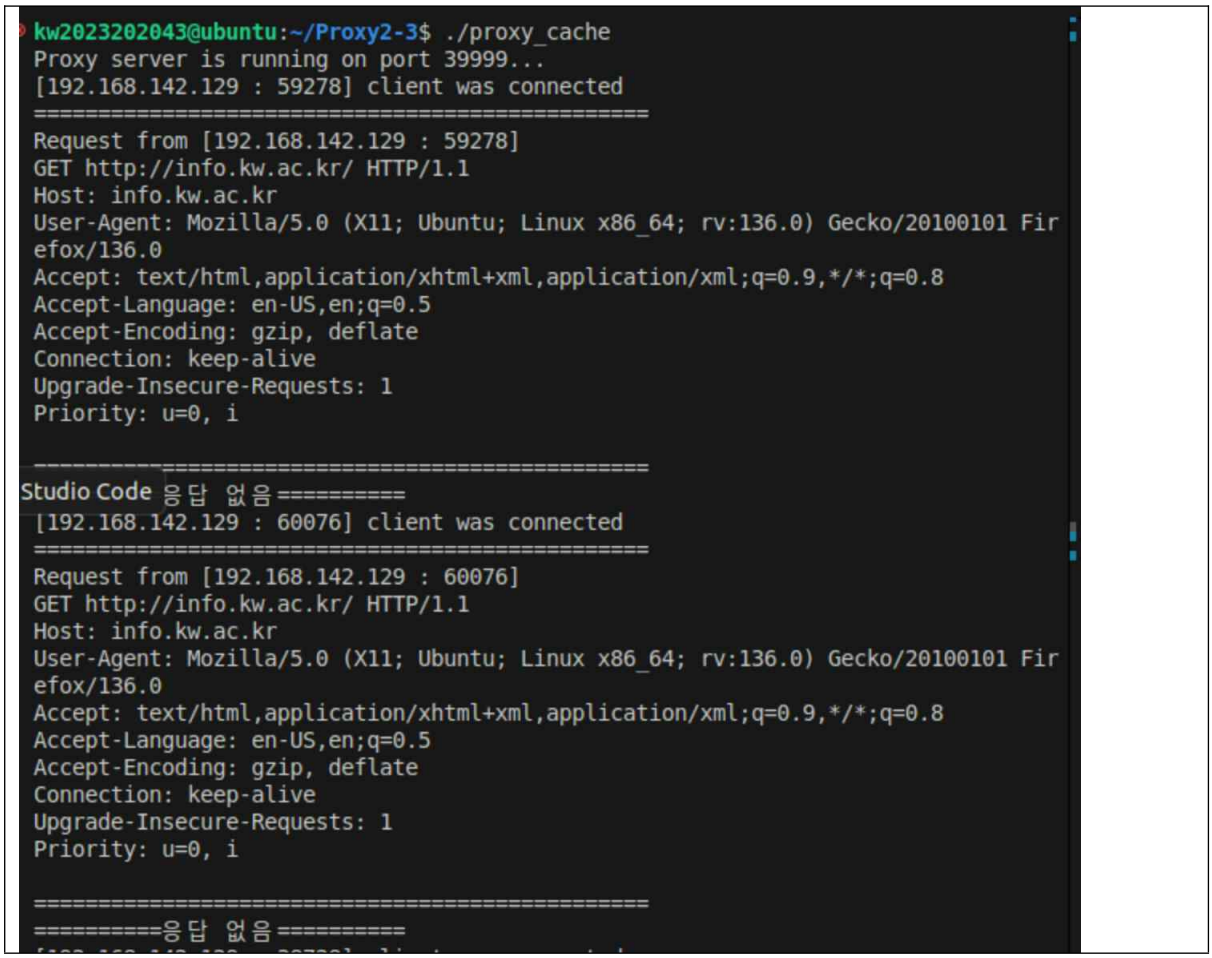
Open ~/Proxy2-3/logfile Save

```
1 [MISS] 192.168.142.129:33646 | http://info.kw.ac.kr/ - [2025/05/15
  18:07:13]
2 [HIT] 192.168.142.129:56816 | http://info.kw.ac.kr/ - [2025/05/15
  18:07:28]
```

proxy_cache.c > handle_client(int, sockaddr_in)

```
159  void handle_client(int clnt_sock, struct sockaddr_in clnt_
213      if (is_cache_hit(full_path)) {
215          send_response(clnt_sock, "HIT", client_ip, client_
         } else {
218          alarm(10);          // Set timeout
219          sleep(13);          // Simulate delay
220
221          FILE *fp = fopen(full_path, "w");
222          if (fp) {
223              fprintf(fp, "Cached data for %s\n", host);
224              fclose(fp);
225          }
226          alarm(0); // Cancel alarm
227
228          write_log("MISS", url, client_ip, client_port);
229          send_response(clnt_sock, "MISS", client_ip, client
230      }
231
232      printf("[%s : %d] client was disconnected\n", client_i
233      close(clnt_sock);
234      exit(0);
235  }
```

Software



서버는 클라이언트의 접속을 기다리다가 요청이 오면 fork()를 통해 자식 프로세스를 생성하여 요청을 처리한다. http://info.kw.ac.kr에 접속하면, 서버는 해당 URL을 SHA1 해시하여 캐시 디렉토리를 확인한다. 첫 번째 요청: 캐시 파일이 존재하지 않음 → MISS(원 서버에서 리소스를 받아 캐시에 저장 후 브라우저로 전송, logfile/log.txt에 MISS 로그가 기록됨), 두 번째 요청: 캐시 파일이 이미 존재함 → HIT(캐시 파일을 브라우저에 전송, logfile/log.txt에 HIT 로그가 기록됨)

alarm(10)함수 호출 후 10초 내에 처리가 완료되지 않으면, SIGALRM이 발생한다. 이 경우 sigalrm_handler()가 실행되어 응답없음의 메시지를 출력하고 자식 프로세스를 종료한다.

요청이 10초 이내에 처리되면 alarm(0)으로 타이머가 해제되고, 정상적으로 HIT/MISS 응답이 클라이언트로 전송된다. 반면 **sleep(13)등으로 의도적 지연**을 주면, SIGALRM이 먼저 발생하여 로그 기록, 응답 전송 등이 수행되지 않고 종료된다. 이 경우 로그 파일에 기록되지 않는다. 응답이 없을 경우 write_log()가 호출되지 않기 때문에, logfile/log.txt에 해당 요청 정보가 기록되지 않기 때문이다.

클라이언트(브라우저)는 응답이 오지 않으면 계속 로딩 상태로 남고, 일정 시간 후 자체적으로 연결을 종료하거나 재요청을 시도한다. 서버가 Connection: close를 하지 못하고 중단되었기 때문에 발생하는 현상이다.

4. 고찰

이번 2-3과제를 통해 다중 프로세스 기반의 프록시 서버 구현 방법과 SIGALRM, SIGCHLD를 이용한 예외 처리 기법을 실습할 수 있었다. 특히, 브라우저를 실제 클라이언트로 사용하여 HTTP 요청을 처리하고, 캐시를 통해 HIT/MISS를 판단하고 처리하는 로직을 구현해봄으로써 네트워크 서버의 동작 흐름을 이해할 수 있었다. 타이머를 활용하여 웹 서버의 응답 지연 상황도 처리할 수 있었으며, 이를 통해 실시간 시스템에서의 타임아웃 대응 방법에 대한 이해를 높일 수 있었다. 실습 과정에서 발생하는 여러 요청과 연결을 fork()를 통해 병렬로 처리함으로써 실제 운영체제에서 사용되는 서버 구조와 유사한 환경을 구현한 것이 매우 의미 있었다.

sleep 함수를 사용하였습니다.