Final Project: Web Proxy

Files:
proxy.h
myproxy.c
logging.c
filtering.c
connection.c

Functions/structs definition:
proxy.h:

```
typedef struct {
        int client_fd;
        struct sockaddr_in client_addr;
        int allow_untrusted;
} client_info;
```

myproxy.c:
In this file, it include the main function and how to start the proxy.
Inside the function start_proxy, it uses pthread to handle multi or concurrent clients. It also block sites given the forbidden sites.
void handle_sigint(int signo);
*void start_proxy(int port, const char *forbidden_sites_path, const char *log_path, int allow_untrusted);*
*int main(int argc, char *argv[]);*

logging.c:
In this file, it handles open outfile directory and loging the request to the outfile.
*void create_log_directory(const char *log_path);*
*FILE *open_log_file(const char *log_path);*
*void log_request(const char *log_path, const char *client_ip, const char *request_line, int status, int response_size);*

filtering.c:
In this file, it handles if the site is blocked with some helper functions.
void load_forbidden_sites(const char *filename);
static int compare_strings(const void *a, const void *b);
void sort_forbidden_sites();
int is_site_blocked(const char *host);
void reload_forbidden_sites(int signo);

connection.c:
In this file, it handles all the request from the clients. In handle_client, it test the request against ssl certificate to verify. It also extract the url to see if it is HTTP or HTTPS inorder to see if it is

GET or HEAD request. This function also handles writing to the host server and read the response.

*char *strcasestr(const char *haystack, const char *needle);*
*void send_error(int fd, int code, const char *msg);*
*int extract_host(const char *url, char *host, size_t host_len);*
*int connect_to_server(const char *host, int port, int *server_fd, SSL **ssl, SSL_CTX **ssl_ctx, int allow_untrusted);*
*void forward_data(int src_fd, int dst_fd, SSL *ssl, const char *buffer, size_t len);*
*void *handle_client(void *arg);*

TEST CASES

1. [curl -v -x http://127.0.0.1:9090 http://example.com] test GET request
   Expected: 200 ok
   Result:

```
* Rebuilt URL to: http://example.com/
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
> GET http://example.com/ HTTP/1.1
> Host: example.com
> User-Agent: curl/7.61.1
> Accept: */*
> Proxy-Connection: Keep-Alive
>
< HTTP/1.1 200 OK
< Content-Type: text/html
< ETag: "84238dfc8092e5d9c0dac8ef93371a07:1736799080.121134"
< Last-Modified: Mon, 13 Jan 2025 20:11:20 GMT
< Cache-Control: max-age=829
< Date: Wed, 12 Mar 2025 05:42:09 GMT
< Alt-Svc: h3=":443"; ma=93600,h3-29=":443"; ma=93600,quic=":443"; ma=93600; v="43"
< Content-Length: 1256
< Connection: close
<
<!doctype html>
<html>
<head>
```

```
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
    @media (max-width: 700px) {
        div {
            margin: 0 auto;
            width: auto;
        }
    }
    </style>
</head>

<body>
<div>
    <h1>Example Domain</h1>
    <p>This domain is for use in illustrative examples in documents. You may use this
    domain in literature without prior coordination or asking for permission.</p>
    <p><a href="https://www.iana.org/domains/example">More information...</a></p>
</div>
</body>
</html>
* Closing connection 0
[ychen729@unix2 WebProxy]$
```

2. [curl -I -x http://127.0.0.1:9090 http://example.com] test HEAD request
   Expected: only return the headers and 200 OK
   Result:

```
2025-03-12T05:43:47.000Z 127.0.0.1 "HEAD http://example.com/ HTTP/1.1
Host: example.com
User-Agent: curl/7.61.1
Accept: */*
Proxy-Connection: Keep-Alive
X-Forwarded-For: 127.0.0.1


" 200 0
```

```
HTTP/1.1 200 OK
Content-Type: text/html
ETag: "84238dfc8092e5d9c0dac8ef93371a07:1736799080.121134"
Last-Modified: Mon, 13 Jan 2025 20:11:20 GMT
Cache-Control: max-age=608
Date: Wed, 12 Mar 2025 05:43:47 GMT
Alt-Svc: h3=":443"; ma=93600,h3-29=":443"; ma=93600,quic=":443"; ma=93600; v="43"
Connection: close
```

3. Test forbidden sites
   Expected: HTTP/1.1 403 Forbidden
   Result:

```
*   Trying 127.0.0.1:9090...
* Connected to 127.0.0.1 (127.0.0.1) port 9090
> GET http://example.com/ HTTP/1.1
> Host: example.com
> User-Agent: curl/8.4.0
> Accept: */*
> Proxy-Connection: Keep-Alive
>
< HTTP/1.1 403 Forbidden
< Connection: close
< Content-Length: 0
<
```

```
2025-03-12T06:20:37.000Z 127.0.0.1 "GET http://example.com/ HTTP/1.1
Host: example.com
User-Agent: curl/8.4.0
Accept: */*
Proxy-Connection: Keep-Alive

" 403 0
```

4. Test X-Forward-For
   Expected: include the X-Forward-Far tag
   Result:

```
2025-03-12T06:22:35.000Z 127.0.0.1 "GET http://httpbin.org/get HTTP/1.1
Host: httpbin.org
User-Agent: curl/8.4.0
Accept: */*
Proxy-Connection: Keep-Alive
X-Forwarded-For: 127.0.0.1
```

5. Sigint handling on blocklist update
   Expected: should update the block list and block the new added sites
   Result:

```
SIGINT received: Reloading forbidden sites list...
Forbidden sites list reloaded. Total blocked sites: 6
Blocklist updated. New count: 6 sites
Checking if site is blocked: github.com
BLOCKED (Blocked entry had 'www.'): github.com
Blocking site: github.com
```