HTTP to HTTPS redirect (HSTS Header)

```
├── libs
│   ├── asio
│   │   └── asio
│   └── crow
│       ├── include
│       └── lib
└── strict_https
    ├── CMakeLists.txt
    ├── main.cpp
    ├── server.crt
    └── server.key
```

```
cmake_minimum_required(VERSION 3.10)

project(StrictHTTPSServer)

# Set C++ standard
set(CMAKE_CXX_STANDARD 17)
set(CMAKE_CXX_STANDARD_REQUIRED ON)

# Add include directories for libs
include_directories(../libs/crow/include)
include_directories(../libs/asio/asio/include)

# Find required packages
find_package(OpenSSL REQUIRED)
find_package(ZLIB REQUIRED)
find_package(Threads REQUIRED)

# Set Crow configuration
add_definitions(-DCROW_ENABLE_SSL)
add_definitions(-DCROW_ENABLE_COMPRESSION)

# Create executable
add_executable(StrictHTTPSServer main.cpp)

# Link libraries
target_link_libraries(StrictHTTPSServer
    OpenSSL::SSL
    OpenSSL::Crypto
```

```
    ZLIB::ZLIB
    Threads::Threads
)

# Copy SSL certificates to build directory
file(COPY server.crt server.key DESTINATION ${CMAKE_BINARY_DIR}/)
```

```
#include <crow.h>
#include <thread>
#include <string>
#include <iostream>
#include <chrono>

// Helper function to add security headers
crow::response add_security_headers(crow::response res) {
    res.set_header("Strict-Transport-Security", "max-age=31536000;
includeSubDomains; preload");
    res.set_header("X-Frame-Options", "DENY");
    res.set_header("X-Content-Type-Options", "nosniff");
    res.set_header("X-XSS-Protection", "1; mode=block");
    return res;
}

class HTTPSRedirectApp {
public:
    void run() {
        // HTTP server for redirects
        std::thread http_thread([this]() {
            crow::SimpleApp http_app;

            // Catch all HTTP requests and redirect to HTTPS
            CROW_ROUTE(http_app, "/<path>")
            ([](const crow::request& req, const std::string& path) {
                crow::response res(301);
                std::string host = req.get_header_value("Host");
                if (host.empty()) {
                    host = "localhost:8443";
                } else {
                    // Replace port 8081 with 8443 for testing
                    size_t port_pos = host.find(":8081");
                    if (port_pos != std::string::npos) {
                        host = host.substr(0, port_pos) + ":8443";
                    } else if (host.find(":") == std::string::npos) {
                        host += ":8443";
```

```
            }
        }
        res.set_header("Location", "https://" + host + "/" + path);
        res.set_header("Cache-Control", "no-cache");
        return res;
    });

    // Root redirect
    CROW_ROUTE(http_app, "/")
    ([](const crow::request& req) {
        crow::response res(301);
        std::string host = req.get_header_value("Host");
        if (host.empty()) {
            host = "localhost:8443";
        } else {
            // Replace port 8081 with 8443 for testing
            size_t port_pos = host.find(":8081");
            if (port_pos != std::string::npos) {
                host = host.substr(0, port_pos) + ":8443";
            } else if (host.find(":") == std::string::npos) {
                host += ":8443";
            }
        }
        res.set_header("Location", "https://" + host + "/");
        res.set_header("Cache-Control", "no-cache");
        return res;
    });

    std::cout << "HTTP redirect server starting on port 8081...\n";
    http_app.port(8081).multithreaded().run();
});

// Give the HTTP server a moment to start
std::this_thread::sleep_for(std::chrono::milliseconds(100));

// HTTPS server with HSTS
crow::SimpleApp https_app;

// Sample HTTPS routes with security headers
CROW_ROUTE(https_app, "/")
([]() {
    crow::response res(200, "text/html",
        "<h1>Welcome to Secure HTTPS Server!</h1>"
```

```
                "<p>This connection is secured with HTTPS and HSTS headers.
</p>"
                "<p>Try accessing this via HTTP at <a
href='http://localhost:8081'>http://localhost:8081</a> - you'll be
redirected here!</p>"
                "<p><a href='/api/status'>Check API Status</a> | <a
href='/security-headers'>View Security Headers</a></p>");
            return add_security_headers(std::move(res));
        });

        CROW_ROUTE(https_app, "/api/status")
        ([]() {
            crow::json::wvalue response;
            response["status"] = "secure";
            response["protocol"] = "https";
            response["hsts_enabled"] = true;
            response["port"] = 8443;
            response["redirect_port"] = 8081;
            crow::response res(200, "application/json", response.dump());
            return add_security_headers(std::move(res));
        });

        CROW_ROUTE(https_app, "/security-headers")
        ([]() {
            crow::response res(200, "text/plain",
                "Security Headers Information:\n"
                "- Strict-Transport-Security: max-age=31536000;
includeSubDomains; preload\n"
                "- X-Frame-Options: DENY\n"
                "- X-Content-Type-Options: nosniff\n"
                "- X-XSS-Protection: 1; mode=block\n\n"
                "These headers are automatically added to all HTTPS
responses.\n");
            return add_security_headers(std::move(res));
        });

        // Configure SSL/TLS certificates
        std::cout << "HTTPS server starting on port 8443 with SSL...\n";
        std::cout << "Using SSL certificates: server.crt and server.key\n";
        std::cout << "\nTesting URLs:\n";
        std::cout << "- HTTP (will redirect): http://localhost:8081\n";
        std::cout << "- HTTPS (with HSTS): https://localhost:8443\n";
```

```
        https_app.port(8443)
                 .ssl_file("server.crt", "server.key")
                 .multithreaded()
                 .run();

        http_thread.join();
    }
};

int main() {
    HTTPSRedirectApp app;
    app.run();
    return 0;
}
```

Move the ssl certs inside the build directory from where executable runs.

```
./StrictHTTPSServer > server_ssl.log 2>&1 &
```

> test the HTTP to HTTPS redirect:

```
# curl -I -k -L http://localhost:8081

HTTP/1.1 301 Moved Permanently
Content-Length: 0
Cache-Control: no-cache
Location: https://localhost:8443/
Server: Crow/1.2.1
Date: Sat, 26 Jul 2025 06:48:07 GMT
Connection: Keep-Alive

HTTP/1.1 200 OK
Content-Length: 331
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
Content-Type: text/html
Server: Crow/1.2.1
Date: Sat, 26 Jul 2025 06:48:07 GMT
Connection: Keep-Alive
```

> test the HTTPS endpoints directly and check for HSTS headers:

```
# curl -k -I https://localhost:8443
HTTP/1.1 200 OK
Content-Length: 331
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
Content-Type: text/html
Server: Crow/1.2.1
Date: Sat, 26 Jul 2025 06:48:22 GMT
Connection: Keep-Alive
```

> test the API status endpoint:

```
# curl -k -s https://localhost:8443/api/status | jq
{
  "port": 8443,
  "redirect_port": 8081,
  "hsts_enabled": true,
  "protocol": "https",
  "status": "secure"
}
```

> test the security headers endpoint:

```
# curl -k -s https://localhost:8443/security-headers
Security Headers Information:
- Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
- X-Frame-Options: DENY
- X-Content-Type-Options: nosniff
- X-XSS-Protection: 1; mode=block

These headers are automatically added to all HTTPS responses.
```

> test the redirect with a specific path:

```
# curl -I -k -L http://localhost:8081/api/status
HTTP/1.1 301 Moved Permanently
Content-Length: 0
Cache-Control: no-cache
Location: https://localhost:8443/api/status
```

```
Server: Crow/1.2.1
Date: Sat, 26 Jul 2025 06:49:09 GMT
Connection: Keep-Alive

HTTP/1.1 200 OK
Content-Length: 91
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
Content-Type: application/json
Server: Crow/1.2.1
Date: Sat, 26 Jul 2025 06:49:09 GMT
Connection: Keep-Alive
```

Crow C++ webserver in the strict_https directory that enforces HTTPS with proper redirects and HSTS headers.

Features:

1. HTTP to HTTPS Redirect: All HTTP requests on port 8081 are redirected to HTTPS on port 8443 using 301 status codes
2. HSTS Headers: All HTTPS responses include strict transport security headers:
    - Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
    - X-Frame-Options: DENY
    - X-Content-Type-Options: nosniff
    - X-XSS-Protection: 1; mode=block
3. SSL/TLS Support: Uses proper SSL certificates (server.crt and server.key) copied from the ssl_example
4. Multiple Endpoints:
    - Root endpoint with security information
    - /api/status - JSON API showing server status
    - /security-headers - Text endpoint showing security headers info

Testing Results:

- HTTP redirects (301) work correctly for all paths
- HTTPS responses include all security headers
- SSL connections work properly
- Path-specific redirects maintain the original path

- JSON API endpoint works with security headers