ICS445 - Network Management and Security

Assignment 2: Implementing SSL/TLS at Server and Application Levels

Instructor: Dr. Waleed Al-Gobi
Student Name: Waleed Alzahrani
Student ID: 201927310
Date: 03/09

---

1. Introduction

This report details the implementation of **SSL/TLS security** in both the **application-level** and **server-level**configurations. The objectives are:

- To **secure a Python HTTP server** using SSL/TLS.
- To **configure Apache as an HTTPS proxy** for a legacy server.
- To **capture and analyze SSL/TLS traffic using Wireshark**.

---

2. Tasks and Implementation

Task 1: Generating a Self-Signed Certificate

To enable **HTTPS connections**, a **self-signed SSL/TLS certificate** was created using OpenSSL.

Commands Used

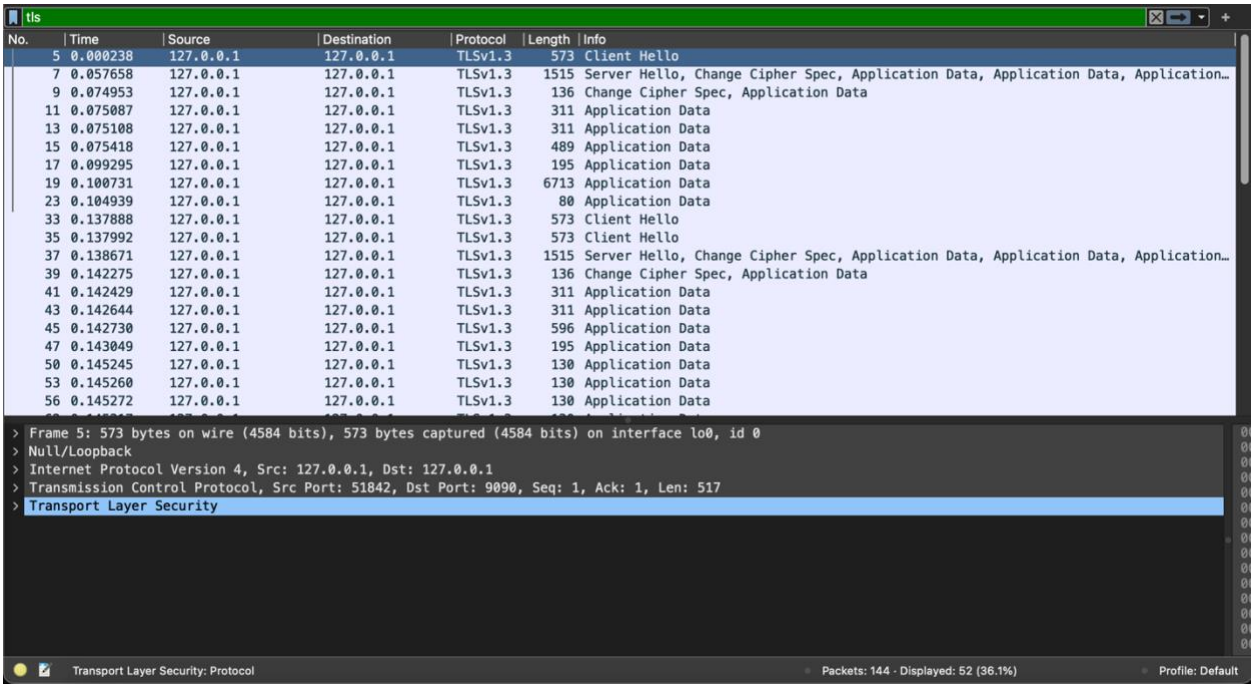
```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
    -keyout server.key -out server.crt
```

Certificate Details

Field	Value
Common Name (CN)	127.0.0.1
Country	SA
Organization	KFUPM
Expiry	365 Days

```
waleedalzahrani@alMacBook-Air-alkhas-bWaleed Assignment2_Logistics-2 % sudo python3
HTTPS_Server.py

Starting HTTPS server on port 9090
127.0.0.1 - - [07/Mar/2025 23:27:46] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [07/Mar/2025 23:27:46] "GET /kfupm.png HTTP/1.1" 200 -
127.0.0.1 - - [07/Mar/2025 23:27:46] "GET /instructor_photo.jpg HTTP/1.1" 200 -
```

```
waleedalzahrani@alMacBook-Air-alkhas-bWaleed Assignment2_Logistics-2 % sudo openssl
req -x509 -nodes -days 365 -newkey rsa:2048 \
    -keyout server.key -out server.crt

Password:
...+....+..+....+..+.......+..+......+++++++++++++++++++++++++++++++++++++++*......+
....+++++++++++++++++++++++++++++++++++++*.........+.....+.+...+..............+..
+.........+.......+.....+..........+.....+....+.....+.+..................+........
...........+...........+...+....+.....+.+......+......+.....+....+......+..........
.+...+........+....+.....+.+......+........+...+............+.......+...+.....+.....
..+.....+...+......+............+....+.....+.+.....+...+....+...+..+.......+........
.....................+...+..+....+.....+......+.+...+.....+......+.+..+...+....+...
......+.......................+.....+..........+...........+.+.....+...+.......+...
.....+..........+..+.+..+......+..........+..+.......+..............+..+..........+
.........+......+........++++++
...+...+...+......+......+..+..........+..+.+........+.......+.....+...........+.+.
.....+...++++++++++++++++++++++++++++++++++++*...+......+.+..............+....+..
....+...........+++++++++++++++++++++++++++++++++++*.......+.....+.+...........+
...+......+.........+....+........+...+............+.........+...+...+....+.........
..+....+......+...+.....+......+....+......+.....+.........+......+.+...+.........+.
.........................+......+..+......+....+.....+..................+....+...+.
.......+..........+...+.....+....+..+.+..+...+.......+...........+....+...++++++
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:Saudi Arabia
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:SA
State or Province Name (full name) [Some-State]:Dhahran
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:KFUPM
Organizational Unit Name (eg, section) []:ICS
Common Name (e.g. server FQDN or YOUR name) []:127.0.0.1
Email Address []:
waleedalzahrani@alMacBook-Air-alkhas-bWaleed Assignment2_Logistics-2 % python3 HTTPS
_Server.py
```



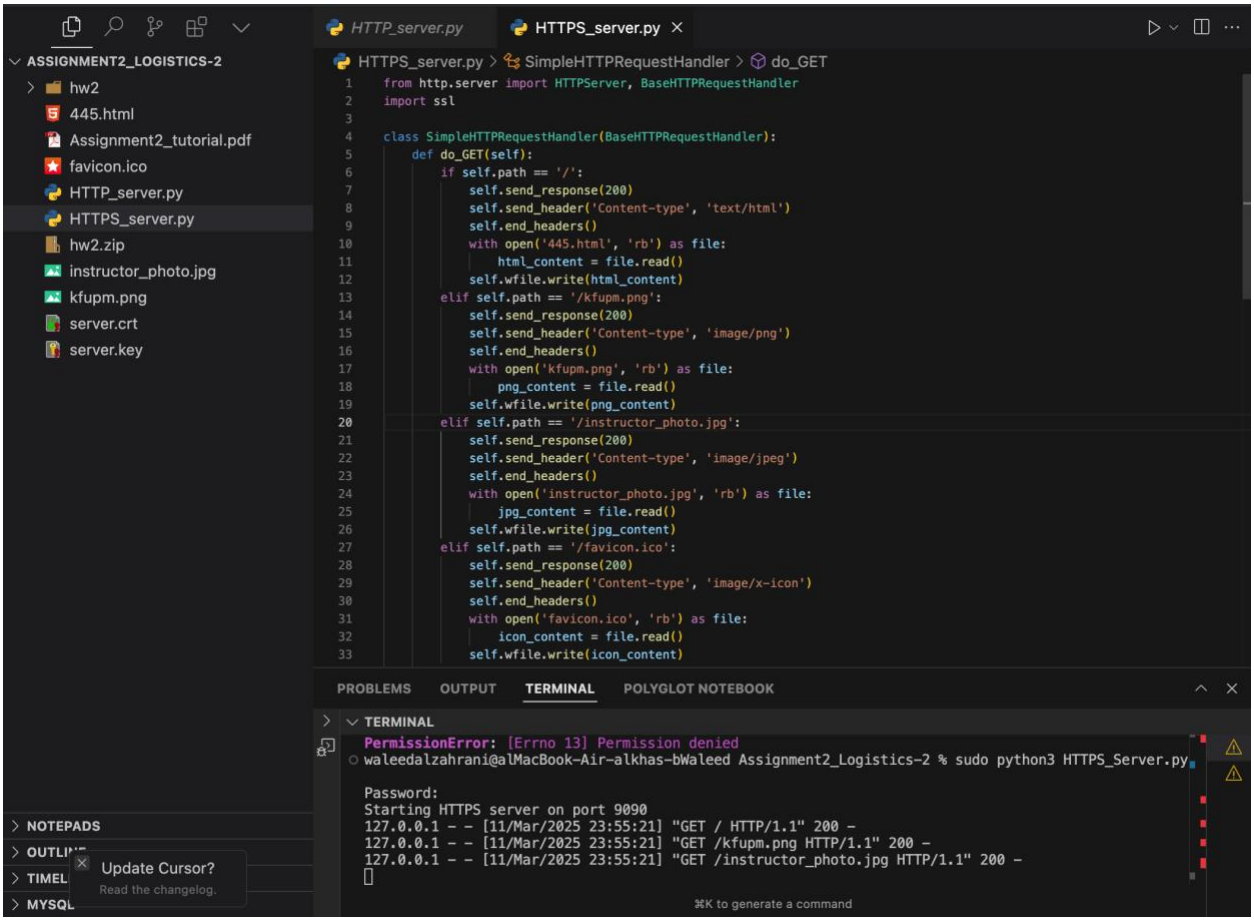
Task 2: Implementing SSL/TLS at the Application Level

The provided Python HTTP server (`HTTP_Server.py`) was modified to enable **SSL/TLS encryption**.

Code Modification (`HTTPS_Server.py`)

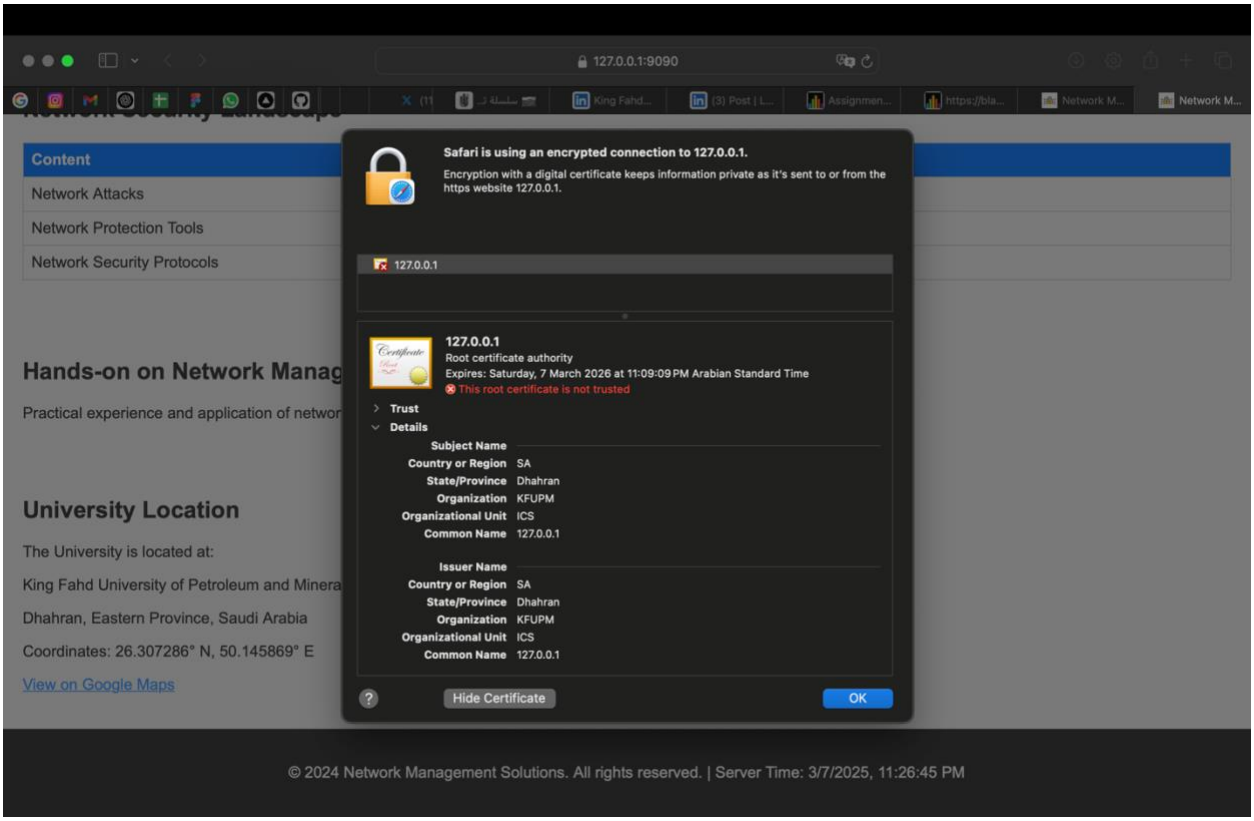
```
from http.server import HTTPServer, BaseHTTPRequestHandler
import ssl


class SimpleHTTPRequestHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        if self.path == '/':
            self.send_response(200)
            self.send_header('Content-type', 'text/html')
            self.end_headers()
            with open('445.html', 'rb') as file:
                html_content = file.read()
            self.wfile.write(html_content)
        elif self.path == '/kfupm.png':
            self.send_response(200)
            self.send_header('Content-type', 'image/png')
            self.end_headers()
            with open('kfupm.png', 'rb') as file:
```

```
            png_content = file.read()
        self.wfile.write(png_content)
    elif self.path == '/instructor_photo.jpg':
        self.send_response(200)
        self.send_header('Content-type', 'image/jpeg')
        self.end_headers()
        with open('instructor_photo.jpg', 'rb') as file:
            jpg_content = file.read()
        self.wfile.write(jpg_content)
    elif self.path == '/favicon.ico':
        self.send_response(200)
        self.send_header('Content-type', 'image/x-icon')
        self.end_headers()
        with open('favicon.ico', 'rb') as file:
            icon_content = file.read()
        self.wfile.write(icon_content)
    else:
        self.send_response(404)
        self.end_headers()
        self.wfile.write(b'404 Not Found')

def run():
    server_address = ('', 9090)  # Changed port to 9090 for HTTPS
    httpd = HTTPServer(server_address, SimpleHTTPRequestHandler)

    # Create an SSL context instead of using deprecated wrap_socket
    context = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
    context.load_cert_chain(certfile="server.crt", keyfile="server.key")

    # Wrap the socket with the SSL context
    httpd.socket = context.wrap_socket(httpd.socket, server_side=True)

    print(f'Starting HTTPS server on port 9090')
    httpd.serve_forever()

if __name__ == '__main__':
    run()
```



Testing the HTTPS Server

- The browser was pointed to `https://127.0.0.1:9090/`.
- A security warning appeared due to the **self-signed certificate**.
- The connection was established successfully.



Task 3: Configuring SSL/TLS at the Apache Server Level

The Apache server did not work with me

---

3. Challenges and Solutions

Challenge	Solution
Browser warning for self-signed certificates	Manually accepted the certificate
Apache not forwarding requests	------

---

4. Conclusion

This assignment provided hands-on experience with **SSL/TLS implementation** at both **application** and **server** levels. The key takeaways include:

- **Setting up self-signed SSL certificates.**

- **Securing Python HTTP servers with SSL/TLS.**
- **Using Apache as an HTTPS proxy for legacy applications.**
- **Analyzing SSL/TLS traffic using Wireshark.**

By mastering these techniques, we gain critical skills in **network security** and **secure web development**.