**LAPORAN PRAKTIKUM IMPLEMENTASI WEB SERVER SEDERHANA MENGGUNAKAN PYTHON SOCKET DAN SSL/TLS**

****

|  |  |
| --- | --- |
| Nama | : L Hafidl Alkhair |
| Nim | : 2023903430060 |
| Kelas | : TRKJ-2C |
| Jurusan | : Teknologi Informasi dan Komputer |
| Progam Studi | : Teknologi Rekayasa Komputer dan Jaringan |
| Dosem Pengampu | : Afla Nevrisa S.Kom, M.Kom |

****

**PROGRAM STUDI TEKNOLOGI REKAYASA KOMPUTER JARINGAN**

**JURUSAN TEKNOLOGI INFORMASI DAN KOMPUTER**

**POLITEKNIK NEGERI LHOKSEUMAWE 2025**

# **LEMBAR PENGESAHAN**

|  |  |
| --- | --- |
| No. Praktikum | : 05 /TIK/TRKJ-2C/ Pemrograman Jaringan |
| Judul | : Implementasi Web Server Sederhana Menggunakan Python Socket Dan Ssl/Tls |
| Nama | : L Hafidl Alkhair |
| Nim | : 202390343060 |
| Kelas | : TRKJ-2C |
| Jurusan | : Teknologi Informasi Dann Komputer |
| Program Studi | : Teknologi Rekayasa Komputer Jaringan |
| Tanggal Praktikum | : 20 Mei 2025 |
| Tanggal Penyerahan | : 16 Juni 2025 |

|  |
| --- |
| Buketrata, 16 Juni 2025 |
| Dosen Pembimbing, |
| **Afla Nevrisa S.Kom, M.Kom**  NIP. 199211172022032007 |

# **DAFTAR ISI**

[LEMBAR PENGESAHAN i](#_Toc201006790)

[DAFTAR ISI ii](#_Toc201006791)

[BAB I 1](#_Toc201006792)

[A. Dasar teori 1](#_Toc201006793)

[B. Tujuan Praktikum 1](#_Toc201006794)

[C. Alat dan bahan 1](#_Toc201006795)

[BAB II 3](#_Toc201006796)

[1. Membuat Program Server 3](#_Toc201006797)

[2. Output program 6](#_Toc201006798)

[3. Penjelasan dan Analisa 7](#_Toc201006799)

[BAB III 8](#_Toc201006800)

[A. Kesimpulan 8](#_Toc201006801)

**BAB I**

**PENDAHULUAN**

1. **Dasar teori**

Pada dasarnya, web server adalah sistem yang merespons permintaan dari klien (browser atau aplikasi) melalui protokol HTTP. Dalam praktikum ini, server dibangun menggunakan bahasa Python dan komunikasi dilakukan melalui socket. Agar komunikasi lebih aman, protokol SSL/TLS digunakan, yang menjamin bahwa data antara client dan server dienkripsi.

Penggunaan socket di Python memungkinkan pembuatan koneksi antar perangkat melalui jaringan. Untuk keamanan, digunakan library ssl yang membungkus socket agar mendukung komunikasi terenkripsi. Sertifikat self-signed (cert.pem dan key.pem) digunakan agar server dapat melakukan autentikasi dasar kepada klien.

1. **Tujuan Praktikum**
2. Membangun sebuah web server sederhana menggunakan Python socket.
3. Mengimplementasikan SSL/TLS agar komunikasi antara client dan server menjadi aman.
4. Memahami proses pembuatan sertifikat digital (self-signed certificate).
5. Melakukan pengamatan terhadap trafik jaringan menggunakan Wireshark, untuk melihat enkripsi SSL/TLS pada koneksi.
6. **Alat dan bahan**
7. **Perangkat Lunak:**

* Python (Interpreter + Library socket dan ssl)
* OpenSSL (untuk membuat sertifikat)
* Wireshark (untuk monitoring trafik jaringan)

1. **Perangkat Keras:**

* Laptop atau PC yang mendukung instalasi Python
* Jaringan lokal (localhost)

1. **File/Script yang Dibuat:**

* server\_ssl.py → Menjalankan server dengan socket dan SSL
* client\_ssl.py → Menghubungkan ke server dengan koneksi terenkripsi
* cert.pem, key.pem → Sertifikat dan kunci privat untuk SSL

**BAB II**

**PRAKTIKUM**

1. **Membuat Program Server**

**Membuat file server\_ssl.py**

*import socket*

*import ssl*

*# Buat socket TCP*

*server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)*

*server\_socket.bind(('0.0.0.0', 8443))*

*server\_socket.listen(5)*

*# Konfigurasi SSL*

*context = ssl.SSLContext(ssl.PROTOCOL\_TLS\_SERVER)*

*context.load\_cert\_chain(certfile='cert.pem', keyfile='key.pem')*

*print("Menunggu koneksi di port 8443...")*

*try:*

*while True:*

*try:*

*client\_socket, addr = server\_socket.accept()*

*# Bungkus koneksi client dalam SSL*

*ssl\_client\_socket = context.wrap\_socket(client\_socket, server\_side=True)*

*print(f"Terhubung dengan {addr}")*

*ssl\_client\_socket.sendall(b"Selamat datang di server SSL!\n")*

*ssl\_client\_socket.close()*

*except ssl.SSLError as e:*

*print(f"[SSL ERROR] Koneksi dari {addr} ditolak: {e}")*

*client\_socket.close()*

*except Exception as e:*

*print(f"[ERROR] Koneksi gagal: {e}")*

*except KeyboardInterrupt:*

*print("\nServer dihentikan oleh user.")*

*finally:*

*server\_socket.close()*

**Membuat file client\_ssl.py**

*import socket*

*import ssl*

*# Buat context SSL dan set agar percaya pada cert.pem*

*context = ssl.create\_default\_context(cafile="cert.pem")*

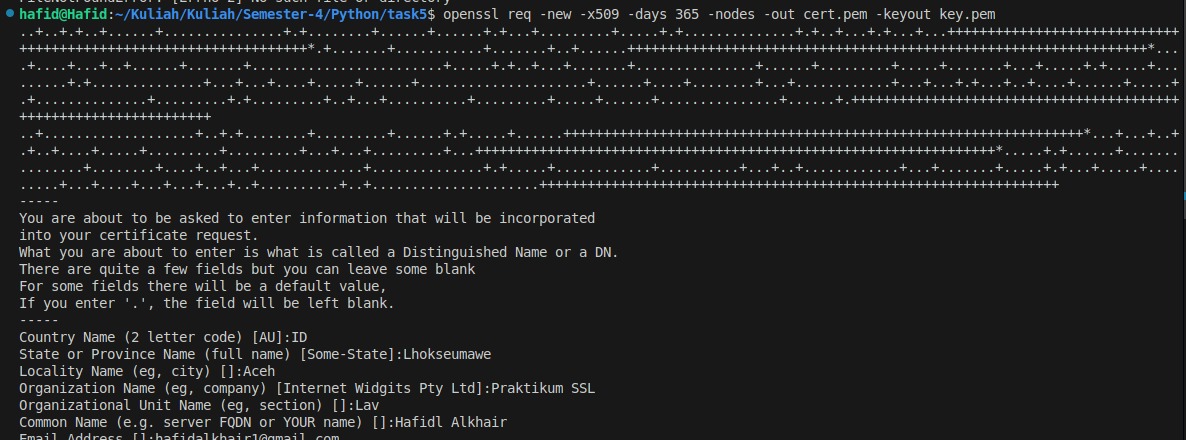
*with context.wrap\_socket(socket.socket(socket.AF\_INET), server\_hostname='Hafidl Alkhair') as conn:*

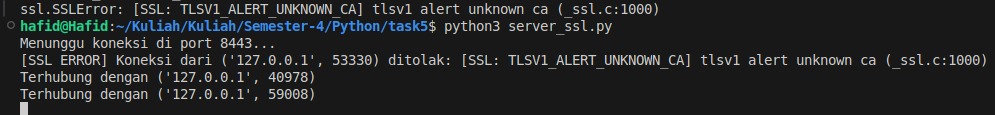
*conn.connect(('localhost', 8443))*

*print(conn.recv(4096).decode())*

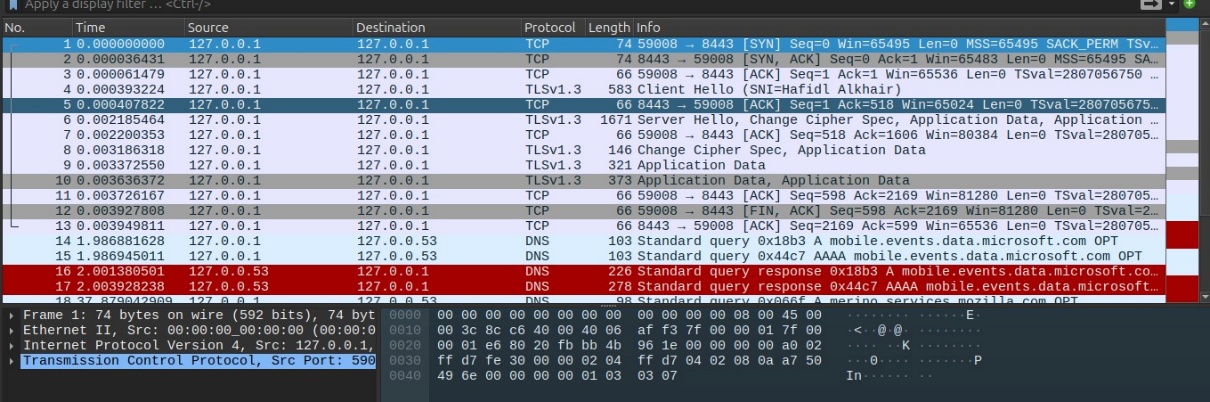
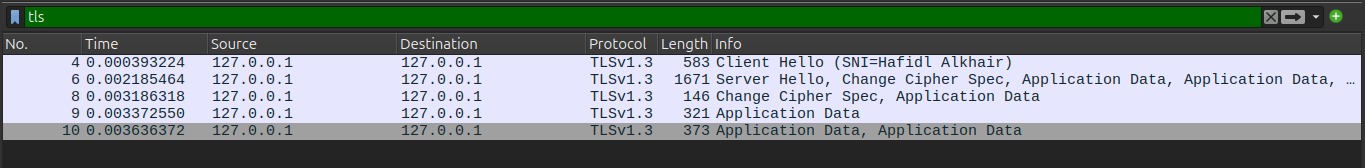
**openssl req -new -x509 -days 365 -nodes -out cert.pem -keyout key.pem**

digunakan untuk membuat sertifikat SSL self-signed beserta private key-nya. Ini diperlukan agar server kamu bisa melakukan komunikasi aman (encrypted) menggunakan SSL/TLS.



1. **Output program**
2. **Server dijalankan**
3. **client dijalankan**



1. **wireshrak**
2. **tls di wireshark**
3. **Penjelasan dan Analisa**
4. **Membuat Program Server dan Client**

* server\_ssl.py membuat server socket TCP di port 8443.
* SSL digunakan melalui konfigurasi SSLContext.
* Saat ada client terhubung, server menyambut dengan pesan teks terenkripsi.
* Error handling juga dilakukan untuk mengatasi kesalahan SSL dan koneksi.
* client\_ssl.py bertindak sebagai klien yang menggunakan SSLContext untuk terhubung dengan server secara aman. Sertifikat cert.pem digunakan agar client "percaya" pada server.

1. **Pembuatan Sertifikat**

Perintah berikut digunakan:

bash

CopyEdit

openssl req -new -x509 -days 365 -nodes -out cert.pem -keyout key.pem

Ini menghasilkan file .pem untuk digunakan sebagai identitas digital server.

1. **Output Program**

* Server menampilkan status bahwa koneksi ditunggu.
* Saat client berhasil terhubung, muncul pesan "Terhubung dengan..." dan client menerima teks sambutan.
* Wireshark digunakan untuk menganalisis komunikasi, dan hasilnya menunjukkan protokol **TLS**, yang menandakan bahwa data telah terenkripsi dengan baik.

1. **Analisis**

Dengan SSL, komunikasi client-server menjadi aman. Socket biasa tidak cukup untuk keamanan karena data bisa disadap, namun dengan SSL, data disandikan sehingga sulit dimanipulasi pihak ketiga. Penggunaan sertifikat meskipun bersifat self-signed, tetap penting untuk membangun dasar komunikasi terenkripsi.

**BAB III**

**PENUTUP**

1. **Kesimpulan**

Berdasarkan hasil praktikum:

1. Server dan client berhasil dibuat dengan Python menggunakan socket.
2. Komunikasi berhasil diamankan menggunakan SSL dengan sertifikat self-signed.
3. Trafik jaringan berhasil diamati dan menunjukkan koneksi terenkripsi.
4. Praktikum ini menunjukkan pemahaman dasar tentang implementasi web server dan keamanan jaringan melalui SSL.