wykonał: Michał Hałuszczak (158069)

System zdalnego zamykania systemów operacyjnych

1. Opis protokołu komunikacyjnego

WebSocket to standardowy protokół do dwukierunkowego przesyłania danych między klientem a serwerem. Protokół WebSockets nie działa przez HTTP, zamiast tego jest oddzielną implementacją na szczycie TCP. Połączenie WebSocket umożliwia pełnodupleksową komunikację między klientem a serwerem, dzięki czemu każda ze stron może przesyłać dane do drugiej za pośrednictwem ustanowionego połączenia. Powodem, dla którego WebSockets, wraz z powiązanymi technologiami Server-sent Events (SSE) i kanałami danych WebRTC, są ważne, jest to, że HTTP nie jest przeznaczony do utrzymywania otwartego połączenia dla serwera w celu częstego przesyłania danych do przeglądarki internetowej.

2. Opis implementacji

```
├── server
  ├── cli.py - Interfejs administratora serwera
  └── server.py - Implementacja serwera opartego o websocket
|── client
  └──client.py - Implementacja komunikacji clienta ze serwerem
```

Serwer jest zaprojektowany, aby nasłuchiwać na porcie *8765* i obsługiwać przychodzące połączenia od klientów. Głównym zadaniem serwera jest przyjmowanie wiadomości od klientów, wysyłanie odpowiedzi oraz zarządzanie zapisami IP klientów w dwóch plikach: **`lista`** i **`wylacz`**.

Obsługa połączeń WebSocket:

Funkcja *handle_connection* obsługuje pojedyncze połączenie WebSocket. Po nawiązaniu połączenia:

- Pobierany jest adres IP klienta (*websocket.remote_address[0]*).
- Otwierany jest plik **wylacz**, aby sprawdzić, czy IP klienta znajduje się na liście zablokowanych. Jeśli tak, serwer wysyła odpowiedź „yep”, informując, że klient został zablokowany, i usuwa to IP z pliku **wylacz**.
- Jeśli klient nie jest zablokowany, serwer sprawdza plik lista, aby sprawdzić, czy adres IP klienta został już zapisany. Jeśli nie, dodaje go do pliku lista.

Obsługa komunikacji z klientem:

- Po zakończeniu wstępnej weryfikacji serwer wysyła wiadomość do klienta z informacją o jego adresie IP.
- Następnie serwer nasłuchuje wiadomości wysyłanych przez klienta w ramach pętli asynchronicznej (`async for message in websocket`). Każda wiadomość odebrana od klienta jest logowana w konsoli, a następnie odsyłana do klienta jako odpowiedź z prefiksem „Echo:”.
- Jeśli połączenie z klientem zostanie przerwane, serwer wyświetla odpowiedni komunikat w konsoli.

3. Sposób uruchamiania projektu

Projekt wymaga, aby środowisko uruchomieniowe obsługiwało jedną z najnowszych wersji Python 3 (3.11+) oraz bibliotekę websockets. Na odpowiednio skonfigurowanym środowisku serwer można uruchomić przykładowo za pomocą komendy: */path/to/python3 /path/to/server.py*. W repozytorium znajduje się przykładowa infrastruktura oparta na konteneryzacji (Docker), umożliwiająca utworzenie serwera z działającą aplikacją oraz trzech klientów połączonych w jednej sieci 192.168.100.0/24.

Bibliografia:

websockets 14.1 documentation
websocket-client/websocket-client: WebSocket client for Python
stopping-docker-container-from-inside