Uniwersytet w Siedlcach

Wydział Nauk Ścisłych i Przyrodniczych

Kierunek Informatyka  
Programowanie w Sieciach Komputerowych

**Aplikacja w Architekturze Mikroserwisów**

**z szyfrowaniem**

***Opracował***:

Karol Przewuski

***Prowadzący:***

dr hab. Prof. Ucz. Stanisław Ambroszkiewicz

Siedlce, 2025

Spis treści

[1. Założenia projektowe 2](#_Toc187342643)

[2. Schemat blokowy 6](#_Toc187342644)

[3.Opis poszczególnych aplikacji 7](#_Toc187342645)

[Client 7](#_Toc187342646)

[Server (API Gateway) 9](#_Toc187342647)

[Login 10](#_Toc187342648)

[Register 11](#_Toc187342649)

[Tablica 13](#_Toc187342650)

[Chat 13](#_Toc187342651)

[Transfer plików wejście 15](#_Toc187342652)

[Transfer plików wyjście 16](#_Toc187342653)

[SecurityUtils(Narzędzia bezpieczeństwa) 16](#_Toc187342654)

[4. Monitorowanie i Analiza Ruchu Sieciowego za pomocą WireShark w Projekcie 18](#_Toc187342655)

[5. Kody Źródłowe 22](#_Toc187342656)

[SecurityUtils 22](#_Toc187342657)

[Client 25](#_Toc187342658)

[Server 34](#_Toc187342659)

[Login 40](#_Toc187342660)

[Register 41](#_Toc187342661)

[Tablica 42](#_Toc187342662)

[Chat 43](#_Toc187342663)

[FileTransferIn 44](#_Toc187342664)

[FileTransferOut 45](#_Toc187342665)

# 1. Założenia projektowe

Głównym celem projektu jest stworzenie zintegrowanego systemu komunikacji, który umożliwia użytkownikom realizację następujących funkcji:

* Rejestracja: Umożliwienie nowym użytkownikom tworzenia kont w systemie.
* Logowanie: Uwierzytelnianie użytkowników i dostęp do funkcji systemu.
* Wymiana wiadomości tekstowych: Tworzenie i zarządzanie dialogiem między użytkownikami.
* Transfer plików: Przesyłanie danych z/na serwer plików.

Struktura Systemu:

System będzie się składać z czterech głównych komponentów:

1. Interfejs Użytkownika (CLI):

* **Opis**: Konsolowy interfejs użytkownika umożliwiający interakcję z systemem.
* **Funkcje:** 
  + - Rejestracja i logowanie użytkowników.
    - Wysyłanie wiadomości tekstowych.
    - Przeglądanie tablicy wiadomości.
    - Zarządzanie plikami: wysyłanie i pobieranie.

1. API Gateway:

* **Opis:** Centralny punkt dostępu do mikroserwisów. Odpowiada za:
  + - Przekierowywanie żądań użytkownika do odpowiednich serwisów.
    - Implementację mechanizmów bezpieczeństwa, takich jak szyfrowanie i uwierzytelnianie.
* **Funkcje:**
  + Szyfrowanie i deszyfrowanie wiadomości.
  + Obsługa podpisów cyfrowych.
  + Autoryzacja użytkowników.

1. Mikroserwisy: Niezależne serwisy zapewniające specyficzne funkcje:

* Serwis Logowania: Uwierzytelnianie użytkownika i sesji
* Serwis Rejestracji: Tworzenie nowych kont użytkowników
* Serwis Tablica: Wyświetlanie ostatnich 10 komunikatów użytkowników
* Serwis Czatu: Wymiana komunikatów między zalogowanymi użytkownikami.
* Serwis Transferu Plików: Wysyłanie i odbieranie plików

1. Back-end jako usługa (BaaS): Zestaw usług back-endowych w tym:

* Bazy Danych: Przechowywanie danych użytkowników oraz historii czatów
* Serwer Plików(Folder): Przechowywanie i zarządzanie plikami przesłanymi przez użytkowników.

Technologie:

Projekt będzie wykorzystywał następujące technologie:

* Język programowania: Java
* Protokoły komunikacyjne: TCP/IP dla komunikacji sieciowej
* Zarządzanie bazą danych: MySQL
* Mechanizmy Bezpieczeństwa:
  + RSA (2048-bit) do podpisów cyfrowych.
  + AES (256-bit) do szyfrowania wiadomości.
  + Wymiana kluczy publicznych między CLI a API Gateway.

Szczegółowy Opis Operacji Kryptograficznych w Systemie

**1. Początkowa Wymiana Kluczy i Uwierzytelnienie**

**Przygotowanie:**

* Klient generuje własną parę kluczy RSA: klucz prywatny (K-CLI) i publiczny (K+CLI), te klucze są unikalne dla każdego uruchomienia aplikacji i służą do uwierzytelnienia oraz podpisywania wiadomości.
* API Gateway posiada swoją stałą parę kluczy RSA: klucz prywatny (K-API) i publiczny (K+API), te klucze są stałe dla serwera.

**Proces Wymiany gdy klient chce połączyć się z serwerem,:**

1. Klient przesyła swój klucz publiczny (K+CLI) do API Gateway
2. API Gateway przesyła swój klucz publiczny (K+API) do klienta
3. Obie strony przechowują klucze publiczne partnera do późniejszego użycia.

**Uwierzytelnienie Dwustronne:**

1. API Gateway → Klient (Serwer weryfikuje klienta):
   * API Gateway generuje losowy ciąg (nonce)
   * Wysyła go do klienta
   * Klient podpisuje go swoim kluczem prywatnym (K-CLI) i odsyła podpisaną wartość
   * API Gateway weryfikuje podpis kluczem publicznym klienta (K+CLI)
   * Jeśli weryfikacja się powiedzie, serwer ma pewność, że klient jest autentyczny
2. Klient → API Gateway (Klient weryfikuje serwer):
   * Klient generuje losowy ciąg (nonce)
   * Wysyła go do serwera
   * API Gateway podpisuje go swoim kluczem prywatnym (K-API) i odsyła podpisaną wartość
   * Klient weryfikuje podpis kluczem publicznym API Gateway (K+API)
   * Jeśli weryfikacja się powiedzie, klient ma pewność, że serwer jest autentyczny

**2. Szyfrowanie Komunikacji**

**Inicjalizacja Sesji:**

1. Klient generuje klucz symetryczny AES dla sesji
2. Klucz AES jest szyfrowany kluczem publicznym API Gateway (K+API)
3. Zaszyfrowany klucz AES jest przesyłany do API Gateway
4. API Gateway deszyfruje klucz AES swoim kluczem prywatnym (K-API)
5. Od tego momentu obie strony mają ten sam klucz AES do szybkiej komunikacji, który będzie używany do szyfrowania wszystkich dalszych wiadomości w tej sesji

**Proces Szyfrowania Wiadomości:**

1. Nadawca:
   * Tworzy oryginalną wiadomość
   * Szyfruje ją kluczem AES
   * Tworzy podpis cyfrowy swoim kluczem prywatnym
   * Łączy zaszyfrowaną wiadomość i podpis
2. Odbiorca:
   * Odbiera zaszyfrowaną wiadomość i podpis
   * Deszyfruje wiadomość kluczem AES
   * Weryfikuje podpis kluczem publicznym nadawcy
   * Przetwarza wiadomość jeśli weryfikacja się powiedzie

**3. Transfer Plików**

**Wysyłanie Pliku:**

1. Plik jest dzielony na segmenty
2. Dla każdego segmentu:
   * Szyfrowanie segmentu kluczem AES
   * Tworzenie podpisu segmentu kluczem prywatnym nadawcy
   * Przesłanie zaszyfrowanego segmentu i podpisu

**Odbieranie Pliku:**

1. Dla każdego odebranego segmentu:
   * Weryfikacja podpisu kluczem publicznym nadawcy
   * Deszyfrowanie segmentu kluczem AES
   * Składanie segmentów w całość

**4. Komunikacja z Mikrousługami**

**Przepływ Danych:**

1. Klient → API Gateway:
   * Klient szyfruje dane kluczem AES
   * Dodaje swój podpis cyfrowy (RSA)
   * Wysyła zabezpieczony pakiet
2. API Gateway → Mikrousługa:
   * API Gateway deszyfruje dane
   * Sprawdza podpis klienta
   * Przekazuje czytelne dane do odpowiedniej mikrousługi
3. Mikrousługa → API Gateway:
   * Mikrousługa przetwarza żądanie
   * Wysyła odpowiedź do API Gateway
   * Dane są w formie niezaszyfrowanej (komunikacja wewnętrzna)
4. API Gateway → Klient:
   * API Gateway szyfruje odpowiedź kluczem AES
   * Dodaje swój podpis cyfrowy (RSA)
   * Wysyła zabezpieczony pakiet do klienta

**5. Zakończenie Sesji (Gdy klient kończy prace).**

* System usuwa tymczasowy klucz AES
* Czyści wszystkie dane sesji z pamięci
* Zamyka bezpiecznie wszystkie połączenia
* Usuwa tymczasowe klucze i dane uwierzytelniające

Ten system zapewnia:

* Poufność danych (szyfrowanie AES)
* Autentyczność (podpisy RSA)
* Integralność (weryfikacja podpisów)
* Bezpieczną wymianę kluczy (RSA)
* Wydajność (AES do dużych danych)

# 

# 2. Schemat blokowy

Obraz zawierający tekst, zrzut ekranu, diagram, krąg

Opis wygenerowany automatycznie

Rysunek 1 Schemat blokowy aplikacji

# 3.Opis poszczególnych aplikacji

# Client

Opis: Aplikacja klienta (CLI) jest uruchamiana po stronie użytkownika i umożliwia interakcję z systemem poprzez konsolowy interfejs użytkownika. Obsługuje różne funkcje systemu, takie jak rejestracja, logowanie, wyświetlanie tablicy wiadomości, czat oraz transfer plików. Komunikacja z serwerem odbywa się za pomocą protokołu TCP/IP z wykorzystaniem szyfrowania AES oraz podpisów cyfrowych opartych na RSA.

**Funkcje:**

1. **Rejestracja:**
   * Użytkownik wprowadza dane (login i hasło), które są przesyłane do serwera za pośrednictwem API Gateway.
   * Dane są szyfrowane kluczem AES i podpisywane kluczem prywatnym klienta.
2. **Logowanie:**
   * Uwierzytelnienie użytkownika na podstawie loginu i hasła.
   * Sesja jest uwierzytelniana dwukierunkowo (klient i serwer wymieniają wyzwania).
3. **Wyświetlanie tablicy:**
   * Pobranie ostatnich 10 wiadomości z bazy danych za pośrednictwem API Gateway.
   * Dane są deszyfrowywane na kliencie po odebraniu.
4. **Czat:**
   * Wysyłanie wiadomości tekstowych do innych użytkowników.
   * Wiadomości są przechowywane w bazie danych i dostępne na żądanie.
5. **Transfer plików:**
   * Wysyłanie i odbieranie plików z serwera.
   * Pliki są dzielone na pakiety, szyfrowane i przesyłane w formacie Base64.
6. **Pomoc:**
   * Wbudowany system pomocy dostępny za pomocą komendy man.
   * Wyświetla listę dostępnych funkcji i ich opisów, ułatwiając korzystanie z aplikacji.

**Architektura Bezpieczeństwa:**

* **Szyfrowanie AES:** Wszystkie dane przesyłane między klientem a serwerem są szyfrowane kluczem AES.
* **Podpisy cyfrowe RSA:** Wiadomości są podpisywane kluczem prywatnym klienta i weryfikowane kluczem publicznym serwera.
* **Wymiana kluczy:**
  + Klucz AES jest szyfrowany kluczem publicznym serwera i przesyłany do klienta.
  + Klient używa klucza prywatnego RSA do odszyfrowania klucza AES.

Przykłady z klasy Client:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 2 Ekran powitalny klienta

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 3 Menu po wybraniu funkcji man (pomoc)

# Server (API Gateway)

**Opis:** API Gateway pełni funkcję centralnego punktu dostępowego do systemu mikroserwisów. Jego głównym zadaniem jest obsługa żądań od klientów (aplikacji CLI), przekierowywanie ich do odpowiednich mikroserwisów oraz zarządzanie odpowiedziami. API Gateway zapewnia bezpieczeństwo komunikacji poprzez szyfrowanie danych oraz uwierzytelnianie dwukierunkowe między klientem a serwerem.

**Funkcje:**

1. **Obsługa zapytań od klientów:**
   * API Gateway odbiera żądania przesyłane przez klienta.
   * Weryfikuje integralność wiadomości poprzez weryfikację podpisów cyfrowych RSA.
   * Odszyfrowuje dane zaszyfrowane kluczem AES.
2. **Przekierowywanie do mikroserwisów:**
   * Na podstawie typu żądania (login, register, chat, tablica, FTI, FTO), API Gateway kieruje zapytanie do odpowiedniego mikroserwisu.
   * Komunikacja z mikroserwisami odbywa się poprzez protokół TCP/IP.
3. **Przetwarzanie odpowiedzi od mikroserwisów:**
   * API Gateway odbiera odpowiedź od mikroserwisu, podpisuje ją swoim kluczem prywatnym RSA i szyfruje kluczem AES, zanim przekaże ją klientowi.
4. **Mechanizmy bezpieczeństwa:**
   * Wymiana kluczy RSA z klientem na początku sesji.
   * Wysyłanie zaszyfrowanego klucza AES do klienta.
   * Podpisywanie odpowiedzi i weryfikacja podpisów wiadomości od klienta.

Przykłady z klasy Server:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 4 Przykład pracy serwera po połączeniu nowego klienta

# Login

**Opis:** Mikroserwis Login umożliwia użytkownikom uwierzytelnianie poprzez wprowadzenie loginu i hasła. Jego głównym zadaniem jest sprawdzanie poprawności danych logowania w bazie danych i zwracanie odpowiedniego statusu operacji do API Gateway.

**Funkcje:**

1. **Weryfikacja danych logowania:**
   * Otrzymuje zaszyfrowane żądanie logowania od API Gateway.
   * Po odszyfrowaniu sprawdza w bazie danych, czy podany login i hasło są prawidłowe.
   * Weryfikuje dane przy użyciu zapytania SQL.
2. **Obsługa błędów:**
   * Zwraca status status:OK, jeśli dane logowania są poprawne.
   * Zwraca status status:NO, jeśli login lub hasło są nieprawidłowe.
3. **Bezpieczeństwo:**
   * Dane logowania są odbierane po wcześniejszym odszyfrowaniu w API Gateway.
   * Żadne dane uwierzytelniające nie są przesyłane w postaci czystego tekstu.

Przykłady z klasy Login:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 4 Logowanie

# Register

**Opis:** Mikroserwis Register jest odpowiedzialny za obsługę rejestracji nowych użytkowników. Jego głównym zadaniem jest weryfikacja, czy nazwa użytkownika jest unikalna, a następnie zapisanie nowego konta w bazie danych. Mikroserwis komunikuje się z API Gateway, który przesyła odpowiednie żądania od klientów.

**Funkcje:**

1. **Weryfikacja unikalności nazwy użytkownika:**
   * Przed zapisaniem nowego użytkownika w bazie danych, mikroserwis sprawdza, czy podany login już istnieje.
   * W przypadku kolizji nazwy użytkownika zwraca status status:NO.
2. **Rejestracja użytkownika:**
   * Po pozytywnej weryfikacji unikalności, dane nowego użytkownika (login i hasło) są zapisywane w bazie danych.
3. **Obsługa błędów:**
   * Jeśli w trakcie rejestracji wystąpi błąd (np. problem z bazą danych), mikroserwis zwraca status status:NO.

Przykłady z klasy Register:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 5 Przykład próby rejestracji użytkownika pod nazwą która istnieje

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 6 Przykład rejestracji użytkownika

# Tablica

**Opis:** Mikroserwis Tablica umożliwia wyświetlenie ostatnich dziesięciu wpisów z tablicy ogłoszeń. Jest dostępny zarówno dla użytkowników zalogowanych, jak i niezalogowanych, dzięki czemu zapewnia ogólnodostępny wgląd w aktywność systemu.

**Funkcje:**

1. **Pobieranie wpisów:**
   * Mikroserwis odczytuje z bazy danych ostatnie 10 wpisów uporządkowanych według daty dodania.
   * Obsługuje zapytania w trybie "tylko do odczytu".
2. **Obsługa zapytań:**
   * Mikroserwis odbiera żądania od API Gateway w celu zwrócenia najnowszych wpisów.
   * Dane są zwracane w formacie zgodnym z wymaganiami klienta (CLI).
3. **Brak konieczności logowania:**
   * Usługa jest dostępna dla wszystkich użytkowników, niezależnie od ich statusu logowania.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 7 Przykład wiadomości w tablicy

# Chat

**Opis:** Mikroserwis Chat umożliwia zalogowanym użytkownikom dodawanie nowych wpisów do tablicy ogłoszeń. Wpisy są przechowywane w bazie danych i stają się widoczne dla innych użytkowników korzystających z funkcji tablicy.

**Funkcje:**

1. **Dodawanie wpisów:**
   * Mikroserwis odbiera wiadomości tekstowe od API Gateway i zapisuje je w bazie danych jako nowy wpis.
   * Każdy wpis zawiera nazwę autora oraz treść wiadomości.
2. **Zarządzanie liczbą wpisów:**
   * Jeśli liczba wpisów na tablicy przekracza 10, najstarszy wpis jest automatycznie usuwany, aby utrzymać limit.
3. **Obsługa błędów:**
   * W przypadku problemów z bazą danych mikroserwis zwraca odpowiedni status błędu (status:NO).
4. **Ograniczenie dostępu:**
   * Funkcja jest dostępna wyłącznie dla użytkowników zalogowanych, co jest weryfikowane przez API Gateway przed przesłaniem żądania.

Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie

Rysunek 8 Przykład dodawania wpisu na chacie

# Transfer plików wejście

**Opis:** Mikroserwis FileTransferIn umożliwia użytkownikom pobieranie plików z serwera. Użytkownik podaje nazwę pliku, a mikroserwis przesyła dane w formie pakietów, zapewniając jednocześnie informację o statusie operacji.

**Funkcje:**

1. **Obsługa żądania pobrania pliku:**
   * Mikroserwis przyjmuje od API Gateway nazwę pliku do pobrania.
   * Weryfikuje, czy plik istnieje na serwerze.
2. **Podział pliku na pakiety:**
   * Plik jest dzielony na mniejsze części (maksymalnie 1024 bajty), które są wysyłane pojedynczo w formacie Base64.
   * Każdy pakiet zawiera dane o:
     + Nazwie pliku.
     + Liczbie pakietów do przesłania.
     + Zawartości bieżącego pakietu.
3. **Informacja o statusie:**
   * Jeśli plik istnieje i zostanie poprawnie przesłany, mikroserwis zwraca status status:OK.
   * W przypadku błędu (np. brak pliku), zwracany jest status status:NO.

Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie

Rysunek 9 Ściąganie pliku z serwera

# Transfer plików wyjście

**Opis:** Mikroserwis FileTransferOut umożliwia użytkownikom wysyłanie plików do serwera. Użytkownik określa nazwę pliku i jego zawartość, która następnie jest przesyłana w formie pakietów danych. Każdy pakiet zawiera szczegółowe metadane, takie jak nazwa pliku, liczba pakietów i treść bieżącego pakietu.

**Funkcje:**

1. **Odbieranie pliku od użytkownika:**
   * Mikroserwis odbiera dane w formie pakietów przesyłanych przez API Gateway.
   * Obsługuje wielopartyjne przesyłanie danych, co umożliwia przesyłanie dużych plików.
2. **Składanie pliku:**
   * Po odebraniu wszystkich pakietów mikroserwis składa je w jeden plik i zapisuje go w katalogu docelowym.
3. **Informacja o statusie:**
   * Jeśli plik zostanie poprawnie zapisany, mikroserwis zwraca status status:OK.
   * W przypadku błędu (np. brak pakietów lub problem z zapisem) zwracany jest status status:NO.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 10 Wysyłanie pliku na serwer

# SecurityUtils(Narzędzia bezpieczeństwa)

**Opis:** Klasa SecurityUtils jest centralnym modułem obsługującym mechanizmy bezpieczeństwa w systemie. Odpowiada za szyfrowanie i deszyfrowanie danych, zarządzanie kluczami kryptograficznymi (AES i RSA), generowanie i weryfikację podpisów cyfrowych, a także uwierzytelnianie dwukierunkowe między klientem a serwerem. Jest używana przez API Gateway oraz mikroserwisy, zapewniając integralność, poufność i autentyczność danych.

**Funkcje:**

1. **Zarządzanie kluczami kryptograficznymi:**
   * Generowanie par kluczy RSA (klucz publiczny i prywatny).
   * Generowanie i zarządzanie kluczem AES do szyfrowania symetrycznego.
   * Ustawianie klucza publicznego partnera komunikacji.
2. **Szyfrowanie i deszyfrowanie danych:**
   * **AES:** Szyfrowanie i deszyfrowanie wiadomości za pomocą klucza AES.
   * **RSA:** Szyfrowanie klucza AES kluczem publicznym partnera i deszyfrowanie klucza AES za pomocą klucza prywatnego.
3. **Podpisy cyfrowe:**
   * Generowanie podpisów cyfrowych wiadomości za pomocą klucza prywatnego RSA.
   * Weryfikacja podpisów wiadomości za pomocą klucza publicznego partnera.
4. **Uwierzytelnianie dwukierunkowe:**
   * Generowanie wyzwań (nonce) do weryfikacji tożsamości.
   * Podpisywanie i weryfikacja wyzwań w celu zapewnienia autentyczności komunikacji.
5. **Kodowanie i dekodowanie wiadomości:**
   * Łączenie zaszyfrowanej wiadomości i podpisu w jeden komunikat.
   * Rozdzielanie i deszyfrowanie otrzymanej wiadomości oraz weryfikacja podpisu.

**Procesy i Algorytmy:**

1. **Szyfrowanie danych (AES):**
   * Dane są szyfrowane za pomocą klucza AES w trybie szyfrowania blokowego.
   * Wynik szyfrowania jest kodowany w Base64 przed przesłaniem.
2. **Podpisy cyfrowe (RSA):**
   * Dane są hashowane za pomocą SHA-256.
   * Hash jest podpisywany kluczem prywatnym, generując podpis cyfrowy.
3. **Uwierzytelnianie:**
   * Serwer generuje nonce (liczbę jednorazową), którą klient podpisuje i zwraca w celu weryfikacji.
   * Analogiczny proces działa w odwrotnym kierunku.

# Monitorowanie i Analiza Ruchu Sieciowego za pomocą WireShark w Projekcie

WireShark w tym projekcie jest wykorzystywany do analizy ruchu sieciowego pomiędzy klientem (CLI), API Gateway oraz mikroserwisami. Jego głównym celem jest weryfikacja implementacji mechanizmów bezpieczeństwa (szyfrowania i podpisów cyfrowych) oraz identyfikacja potencjalnych słabości w przesyłaniu danych.

Analiza Ruchu Sieciowego za pomocą Wiresharka

Przeprowadzona została analiza ruchu sieciowego w celu weryfikacji bezpieczeństwa i integralności komunikacji w projekcie. Głównym celem było sprawdzenie, czy przesyłane dane są odpowiednio szyfrowane i podpisywane cyfrowo.

**Kroki Testowe**

**Przygotowanie:**

1. Uruchomiono Wiresharka jako administrator.
2. Wybrano interfejs sieciowy **Localhost**, ponieważ komunikacja w projekcie odbywa się lokalnie.
3. Ustawiono filtr: tcp.port == 1410 (port używany przez API Gateway).
4. Uruchomiono wszystkie komponenty programu w kolejności:
   * Mikroserwisy (Login, Register, Tablica, Chat, FileTransferIn, FileTransferOut).
   * Server.java (API Gateway).
   * Client.java (interfejs użytkownika).

**Przeprowadzone Testy i Wyniki**

**1. Test Rejestracji**

* **Opis:** Użytkownik wprowadził dane rejestracyjne (login i hasło).
* **Wynik w Wireshark:**
  + Przechwycono zaszyfrowaną wiadomość rejestracyjną w formacie type:register#login:<login>#haslo:<hasło> zaszyfrowaną AES.
  + Wysłane dane były nieczytelne w przechwyconych pakietach.
  + Każda wiadomość zawierała podpis cyfrowy, który został poprawnie zweryfikowany.

**2. Test Logowania**

* **Opis:** Użytkownik wprowadził dane logowania.
* **Wynik w Wireshark:**
  + Przechwycono wymianę kluczy RSA na początku sesji.
  + Klucz AES został przesłany w zaszyfrowanej formie za pomocą RSA.
  + Wymiana challenge-response była widoczna jako zaszyfrowane wiadomości.
  + Właściwe dane logowania były zaszyfrowane AES i podpisane cyfrowo.

**3. Test Wysłania Wiadomości na Tablicę**

* **Opis:** Użytkownik dodał wpis do tablicy.
* **Wynik w Wireshark:**
  + Przechwycono zaszyfrowaną wiadomość type:chat#login:<login>#tresc:<treść wiadomości> przesłaną do serwera.
  + Odpowiedź serwera (status:OK) była również zaszyfrowana i podpisana cyfrowo.
  + Treść wiadomości nie była widoczna w przechwyconych pakietach.

**4. Test Transferu Pliku**

* **Opis:** Użytkownik wysłał plik do serwera.
* **Wynik w Wireshark:**
  + Przechwycono wiele pakietów, ponieważ plik został podzielony na mniejsze fragmenty (1024 bajty).
  + Każdy fragment pliku był zaszyfrowany AES i podpisany cyfrowo.
  + Odpowiedź serwera (status:OK) potwierdzająca odbiór pliku była zaszyfrowana.

**5. Test Pobierania Pliku**

* **Opis:** Użytkownik pobrał wcześniej przesłany plik.
* **Wynik w Wireshark:**
  + Przechwycono żądanie pobrania pliku w formacie type:FTI#login:<login>#filename:<nazwa\_pliku>.
  + Odpowiedź serwera zawierała zaszyfrowane fragmenty pliku.
  + Pakiety z danymi pliku były dłuższe, co wynika z większej ilości przesyłanych danych.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Równolegle

Opis wygenerowany automatycznie

Rysunek 11 Podgląd każdej wykonanej operacji podczas testowania bezpieczeństwa.

Powyższy zrzut ekranu przedstawia analizę przechwyconej komunikacji sieciowej pomiędzy klientem a API Gateway w projekcie. Widoczny przepływ danych obrazuje wymianę informacji w trakcie realizacji różnych operacji, takich jak logowanie, rejestracja, czy transfer plików.

Czerwony: pakiety od klienta do serwera

Niebieski: pakiety od serwera do klienta

# 5. Kody Źródłowe

# SecurityUtils

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

# Client

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, menu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, menu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

# Server

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

# Login

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

# Register

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

# Tablica

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie Obraz zawierający zrzut ekranu, tekst, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

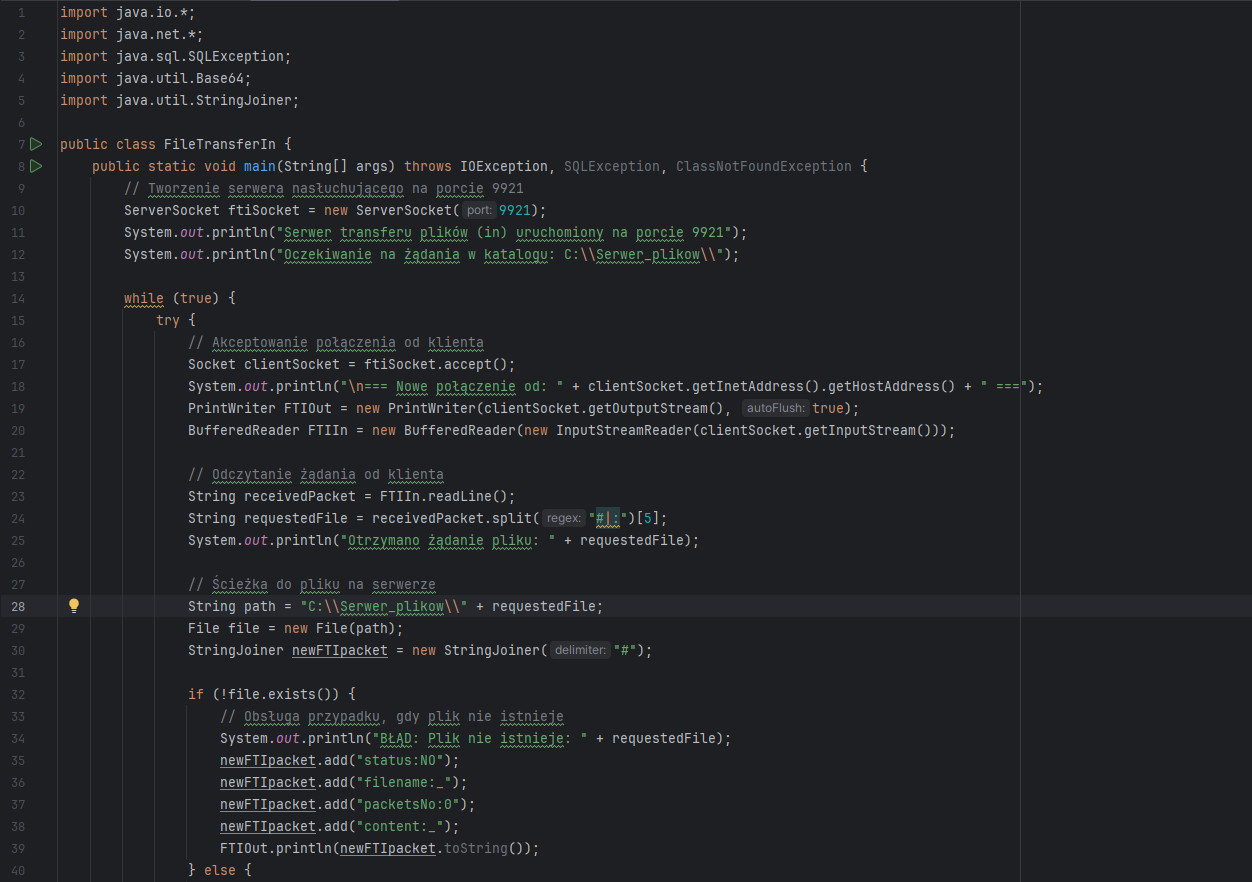
# Chat

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, System operacyjny

Opis wygenerowany automatycznie Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

# FileTransferIn

 Obraz zawierający tekst, zrzut ekranu, menu

Opis wygenerowany automatycznie

# FileTransferOut

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie