Komunikator z szyfrowaniem - sprawozdanie 30.10.2018

Mariusz Najwer 218592

Cel

Należało przygotować komunikator (chat) klient-serwer wspierający bezpieczną wymianę sekretu (protokół Diffie-Hellman) oraz obsługujący zadany format komunikacji.

Zadania, które zostały wykonane

1. Kodowanie wiadomości: base64
2. Szyfrowanie wiadomości: Cezar, XOR (jednobajtowy), none
3. Dynamiczny Protokół D-H (inny dla każdego klienta i każdej wiadomości)
4. Format JSON
5. Testy szyfrów i protokołu D-H

Sposób wykonania zadania

1. Do zrealizowania zadania wykorzystano Node JS web framework express, który w łatwy sposób umożliwił komunikację pomiędzy klientami i serwerem. Przy wykonywaniu zadania pomocne były dokumentacje związane z JS i socket.io.

Testy

- ✓ W czacie może brać udział nieskończona liczba klientów (uruchomiono kilku klientów)
- ✓ Poprawne zaimplementowanie protokołu D-H (szyfrowane wiadomości docierają do klientów, każdy klient może decydować o sposobie szyfrowania)
- ✓ Poprawne szyfrowanie z wykorzystaniem cezara, XOR. Funkcje były sprawdzone „przed" implementacją do głównego programu (jak? przykładowe przesunięcia, krańcowe litery, przykładowe ciągi znaków)
- ✓ Poprawna obsługa formatu JSON
- ✓ Poprawna generacja kluczy, sekrety są niedostępne
- ✓ Przy szyfrowaniu z wykorzystaniem algorytmu cezara, zostały uwzględnione polskie litery
- ✓ Nagła zmiana tajnego sekretu (próba ataku) powoduje oczekiwany błąd w szyfrowaniu (brak możliwości odczytania wiadomości)
- ✓ Klient może nasłuchiwać czatu, znając sam adres (udostępniony link)
    - Można ustawić sposób w jaki będzie nasłuchiwał przed przystąpieniem do czatu, domyślnie ustawiono brak szyfrowania (encryption:none)

Wnioski

1. **Zrealizowano cel zadania**
2. Dokumentacja socket.io mogłaby być lepsza. Byłem zagubiony na początku
3. Warto dołączyć bazę danych do aplikacji, która zapamiętywałaby historię rozmów
4. Warto dodać opcję logowania do serwisu, aby każdy klient miał ZAWSZE stały identyfikator
5. Warto rozszerzyć aplikację o możliwość informacji o uczestnikach czatu.
6. W przyszłości warto rozbudować aplikację o „pokoje tematyczne"
7. Czat głosowy
8. Warto dodać arkusze stylów do aplikacji, aby zachęcała do użytkowania

9. Wdrożenie aplikacji na komercyjny serwer wymaga początkowego kapitału
10. Znany jest algorytm szyfrowania, bezpieczeństwo wymiany danych opiera się na problemie faktoryzacji dużych liczb, należy dobrać odpowiednio duże parametry startowe
11. Ale nie za duże, bo:
    a. Math.pow(2, 53) == Math.pow(2, 53) + 1 // true w JS
12. JavaScript nie jest najlepszym językiem do zadań matematycznych
13. Ciekawostka. Powyższe zadanie będzie realizowane na warsztatach w PGS Software we Wrocławiu, w ramach nauki .Neta i Reacta, lecz bez protokołu D-H (9.11-6.12.2018r.)

Literatura

1. https://socket.io/
2. https://nodejs.org/en/
3. https://www.geeksforgeeks.org/implementation-diffie-hellman-algorithm/
4.