Sprawozdanie wstępne

Projekt: Mini TLS – wariant W1 (Encrypt-then-MAC)

Alesia Filinkova, Weronika Maślana, Diana Pelin

1. Cel projektu

Celem projektu jest zaprojektowanie oraz implementacja uproszczonego, szyfrowanego protokołu komunikacyjnego typu klient–serwer, inspirowanego protokołem TLS, działającego na bazie protokołu TCP. Protokół zapewnia poufność przesyłanych danych oraz mechanizmy integralności i autentyczności komunikatów przy użyciu schematu **encrypt-then-MAC**.

2. Architektura systemu

System składa się z dwóch głównych komponentów:

- **Serwer** – obsługuje jednocześnie wielu klientów (liczba klientów określana parametrem przy uruchomieniu).
- **Klient** – inicjuje połączenie z serwerem, wysyła wiadomości oraz kończy sesję.

Komunikacja odbywa się w sieci dockerowej. Wszystkie dane przechowywane są w pamięci operacyjnej.

3. Struktura wiadomości protokołu

3.1 ClientHello

Wiadomość inicjująca połączenie, wysyłana przez klienta w postaci **nieszyfrowanej**.

Struktura:

```
ClientHello:

- msg_type: "ClientHello"
- client_public_key
```

Opis:

- `client_public_key` – klucz publiczny klienta używany w algorytmie wymiany kluczy Diffie-Hellman.

3.2 ServerHello

Wiadomość odpowiedzi serwera, również **nieszyfrowana**.

Struktura:

```
ServerHello:
- msg_type: "ServerHello"
- server_public_key
```

Opis:

- `server_public_key` – klucz publiczny serwera wykorzystywany do wyznaczenia wspólnego klucza sesyjnego.

3.3 Wiadomość szyfrowana (EncryptedMessage)

Wiadomość przesyłana po zakończeniu wymiany kluczy, zabezpieczona zgodnie ze schematem encrypt-then-MAC. Wiadomość jest najpierw szyfrowana, a następnie obliczany jest MAC dla zapewnienia integralności i autentyczności.

Struktura:

```
EncryptedMessage:
- msg_type: "Data"
- ciphertext
- mac
```

Opis:

- `ciphertext` – zaszyfrowana treść wiadomości
- `mac` – kod uwierzytelniający wiadomość (Message Authentication Code)

3.4 EndSession

Wiadomość kończąca sesję, **szyfrowana**.

Struktura:

```
EndSession:
- msg_type: "EndSession"
```

Po odebraniu tej wiadomości sesja zostaje zakończona, a dalsza komunikacja wymaga ponownego wysłania ClientHello. Może być wysłana przez klienta lub serwer.

4. Wykorzystane algorytmy kryptograficzne

4.1 Wymiana kluczy – Diffie-Hellman

Do ustalenia wspólnego klucza sesyjnego wykorzystywany jest algorytm **Diffie-Hellman key exchange**.

Parametry:

- Liczba pierwsza p (mała, np. 23)
- Generator g (np. 5)

Przebieg:

1. Klient generuje prywatny klucz a i oblicza klucz publiczny: `A = g^a mod p`

2. Serwer generuje prywatny klucz b i oblicza klucz publiczny: `B = g^b mod p`
3. Wymiana kluczy publicznych przez `ClientHello` i `ServerHello`.
4. Obliczenie wspólnego sekretu:
   - Klient: s = B^a mod p
   - Serwer: s = A^b mod p

Klucz sesyjny wykorzystywany jest zarówno do szyfrowania danych, jak i do generowania MAC.

4.2 Szyfrowanie danych

Algorytm: One-Time Pad (OTP).

Opis: Każda wiadomość jest szyfrowana XOR-em z kluczem sesyjnym K. Klucz K jest rozszerzany (powtarzany) do długości wiadomości. Dla każdej nowej wiadomości generowany jest nowy segment klucza (np. przesunięcie w strumieniu), aby uniknąć ponownego użycia tego samego klucza.

5. Mechanizm integralności i autentyczności (Encrypt-then-MAC)

W wariancie W1 zastosowano schemat **encrypt-then-MAC**, który polega na:

Proces wysyłania:

- Najpierw przygotujemy oryginalną wiadomość (plaintext).
- Szyfrujemy text przy użyciu OTP z kluczem K, uzyskując ciphertext.
- Obliczamy MAC na ciphertext.
- Wysyłamy `EncryptedMessage.`

Proces odbioru:

- Server odbiera ciphertext i MAC.
- Server oblicza MAC na odebranym ciphertext i porównuje z odebranym MAC. Jeśli nie pasuje, odrzuca wiadomość (naruszenie integralności/autentyczności).
- Jeśli MAC jest poprawny, następuje odszyfrowanie ciphertext przy użyciu OTP.

Schemat ten zapewnia:

- **Integralność danych** – wykrywanie modyfikacji wiadomości.
- **Autentyczność nadawcy** – tylko strona posiadająca klucz sesyjny może wygenerować poprawny MAC.

6. Przykładowy scenariusz komunikacji

1. Klient wysyła `ClientHello`.
2. Serwer odpowiada `ServerHello`.
3. Następuje obliczenie wspólnego klucza sesyjnego.
4. Klient wysyła zaszyfrowaną wiadomość typu `EncryptedMessage`.
5. Serwer weryfikuje MAC, odszyfrowuje dane i wyświetla treść.
6. Klient lub serwer wysyła `EndSession`.

7. Połączenie zostaje zakończone.