Sprawozdanie wstępne – projekt PSI

Przedmiot: Programowanie Sieciowe
Autorzy: Marek Dzięcioł, Jakub Mieczkowski, Tomasz Nejman
Data sporządzenia: 02.01.2026
Wersja: 1.0

Treść:

Celem projektu jest zaprojektowanie oraz implementacja szyfrowanego protokołu opartego na protokole TCP, tzw. mini TLS.

Wybrany wariant funkcjonalny

• W2 – wykorzystanie mechanizmu mac-then-encrypt dla wysyłanych szyfrowanych wiadomości jako mechanizm integralności i autentyczności, implementacja w Pythonie.

Struktura wiadomości

- ClientHello
  - 4 bajty – sygnatura wiadomości: “HELO”
  - 16 bajtów – liczby p,g (2x int64)
  - 8 bajty – liczba A (int64)

- ServerHello
  - 4 bajty – sygnatura wiadomości: “EHLO”
  - 8 bajty – liczba B (int64)

- Szyfrowana wiadomość
  - 28 bajty – kod uwierzytelnienia HMAC-SHA224
  - 4 bajty – typ wiadomości
  - 8 bajty – rozmiar wiadomości
  - n bajtów – zawartość wiadomości

- EndSession
  - specjalny typ szyfrowanej wiadomości, o specjalnym typie wiadomości = 3
  - zawartość wiadomości “EDNSSION”

Wykorzystane algorytmy

- Do wymiany kluczy stosujemy algorytm Diffiego-Hellmana, gdzie używane są liczby p, g, A, B (publiczne), wymienione wcześniej w wiadomościach.
- Do szyfrowania stosujemy algorytm: OTP. Strumień szyfrujący jest generowany na podstawie klucza sesji (np. poprzez generator pseudolosowy) i XOR-owany ze strumieniem danych.
- Integralność i autentyczność: algorytm HMAC-SHA224. Mechanizm realizowany jest w trybie mac-then-encrypt.

Przykładowy scenariusz użycia:

- p = 23, g = 5
- klient wybiera a = 6 i oblicza A = 5^6 (mod 23) = 8
- klient wysyła wiadomość: HELO2358
    - Sygnatura: HELO
    - Liczby p, g: 23, 5
    - Klucz publiczny A: 8
- serwer odbiera wiadomość, odczytuje p, g
- serwer wybiera b = 15 i oblicza B = 5^{15} (mod 23) = 19
- serwer wysyła wiadomość: EHLO19
    - Sygnatura: EHLO
    - Klucz publiczny B: 19
- wspólny klucz K = 19^6 (mod 23) = 8^{15} (mod 23) = 2
- klient wysyła szyfrowaną wiadomość:
    - Przygotowanie danych jawnych:
        - typ wiadomości: 1 (zwykła wiadomość)
        - rozmiar: 6 (długość słowa “TAJNE” + “\0”)
        - zawartość: “TAJNE\0”, zaszyfrowane używając klucza K=2
    - Obliczenie autentyczności (MAC):
        - HMAC-SHA224(klucz, dane do podpisu)
        - Doklejenie 28-bajtowego skrótu na początek wiadomości
    - Budowa bloku do zaszyfrowania:
        - [HMAC (28B)] + [Typ (4B)] + [Rozmiar (8B)] + [Treść (6B)]
        - Łączna długość bloku: 46 bajtów
    - Szyfrowanie:
        - Cały blok jest szyfrowany algorytmem OTP z użyciem klucza sesji.
    - Wysłanie wiadomości serwerowi
- klient wysyła wiadomość o zakończeniu sesji:
    - typ wiadomości: 3 (kod zakończenia)
    - rozmiar: 9 (długość “EDNSSION” + “\0”)
    - zawartość: “EDNSSION\0”, zaszyfrowane używając klucza K=2

    - suma kontrolna HMAC-SHA224

Sposób realizacji mechanizmu integralności i autentyczności dla szyfrowanych wiadomości:

projekcie zastosowano podejście mac-then-encrypt, co zapewnia ochronę przed modyfikacją szyfrogramu (pod warunkiem weryfikacji MAC po odszyfrowaniu).

1. Nadawca tworzy paczkę danych (typ + rozmiar + treść).
2. Nadawca oblicza kod HMAC-SHA224 z tej paczki, używając wspólnego klucza sesji.
3. Kod HMAC jest doklejany na początku paczki.
4. Całość (HMAC + dane) jest szyfrowana.
5. Odbiorca deszyfruje wiadomość, rozdziela HMAC od danych, samodzielnie oblicza HMAC z odszyfrowanych danych i porównuje go z otrzymanym (odszyfrowanym) kodem HMAC.

Dodatkowe informacje

- Całość będzie realizowana w pamięci RAM, bez wykorzystania bazy danych.
- Aplikacja będzie sterowana z wiersza poleceń zgodnie z wymaganiami.
- Aplikacja będzie pozwalała na utrzymywanie wielu sesji w tym samym czasie (maksymalna liczba sesji do określenia jako argument wywołania).