Projekt PSI - Sprawozdanie wstępne

Zespół 37 - Jakub Bąba, Angelika Ostrowska, Aleksandra Szymańska

Założenia projektu

Implementację rozwiązania w postaci architektury klient-serwer opartej na protokole TCP dostarczymy w Pythonie. Komunikacja będzie sterowana z wiersza poleceń: klienci będą mogli wybrać między zainicjowaniem połączenia z serwerem, wysłaniem wiadomości i zakończeniem połączenia z serwerem, serwer będzie mógł zamknąć połączenie dla wybranego klienta. Dodatkowo serwer będzie w stanie obsłużyć wielu klientów (ich liczba będzie przekazywana jako argument wywołania) i wyświetli tych aktualnie połączonych. Projekt będzie uruchamiany w sieci dockerowej minimalną liczbą poleceń.

Do zrealizowania projektu wybraliśmy wariant W1, a więc jako mechanizm integralności i autentyczności wykorzystamy mechanizm encrypt-then-mac dla wysyłanych szyfrowanych wiadomości.

Serwer sam będzie przydzielał klientom kolejne ID. Klient ich nie wysyła, bo nie moglibyśmy zapewnić ich unikalności.

Struktura wiadomości

- ClientHello: “ClientHello|<podstawa>|<moduł>|<klucz_publiczny_klienta>”
- ServerHello: “ServerHello|<klucz_publiczny_serwera>”
- Wiadomość (szyfrowana): “<długość_zaszyfrowanej_wiadomości>|<treść_wiadomości>|<MAC>”
- EndSession (szyfrowana): “{EndSession}|<MAC>”

Podstawa (liczba całkowita) i moduł (liczba pierwsza) ustalane przez klienta będą używane do ustalenia klucza sesji w algorytmie wymiany kluczy Diffie-Hellmana, który opiszemy później.
Znaki | są używane jako separatory np. między wiadomością a kodem MAC.

Używamy długości przesyłanej wiadomości, żeby mieć pewność, że nie podzielimy jej w złym momencie, jeśli w zaszyfrowanej wiadomości znajdzie się separator |.

Napis DONE jest dodawany pod koniec wiadomości. Dzięki temu będzie zawsze wiadomo, kiedy wiadomość się kończy, co jest ważne w przypadku tych komunikatów, które nie mieszczą się w buforze - jeśli wiadomość musi być podzielona na kilka części, DONE będzie dodany tylko na końcu ostatniej z części.
Podobnie jak reszta wiadomości, napis DONE jest szyfrowany.

Wykorzystane algorytmy

Do wymiany kluczy wykorzystamy algorytm Diffie-Hellman key exchange. Klient przekaże podstawę oraz moduł i razem z serwerem wymienią klucze publiczne w pierwszych wiadomościach, by ustalić wspólny klucz sesji. Odbędzie się to w następujący sposób:

Klient wygeneruje losowy klucz prywatny, ustali podstawę i moduł do obliczenia klucza publicznego i przekaże te informacje (poza kluczem prywatnym) serwerowi, który tak samo losowo wygeneruje swój klucz prywatny i użyje otrzymanych podstawy i modułu do obliczenia klucza publicznego, który odeśle klientowi.

$$\text{Klucz publiczny} = \{\text{podstawa}\}^{\{\text{klucz prywatny}\}} \bmod \{\text{moduł}\}$$

Klucz sesji dla klienta będzie obliczany ze wzoru:

$$\{\text{klucz sesji}\} = \{\text{klucz publiczny serwera}\}^{\{\text{klucz prywatny klienta}\}} \bmod \{\text{moduł}\}.$$

Analogicznie dla serwera:

$$\{\text{klucz sesji}\} = \{\text{klucz publiczny klienta}\}^{\{\text{klucz prywatny serwera}\}} \bmod \{\text{moduł}\}.$$

Z właściwości obliczeń wynika to, że klucz sesji po obu stronach będzie taki sam.

Podsumowując:

- Klient przekazuje podstawę, moduł oraz swój klucz publiczny
- Serwer w ramach odpowiedzi przesyła swój klucz publiczny
- Obie strony posiadają swoje klucze prywatne, a także są w stanie obliczyć wspólny klucz sesji

Przykładowo, rozpoczęcie komunikacji może wyglądać następująco:

- Klient ustala wartość podstawy (23) oraz modułu (5)
- Klient ustala wartość klucza prywatnego (4) i wylicza klucz publiczny (4)
- Klient przesyła wiadomość ClientHello|23|5|4
- Serwer ustala wartość klucza prywatnego (3) i wylicza klucz publiczny (10)
- Serwer odsyła wiadomość ServerHello|10
- Obie strony wyliczają wartość klucza sesji, który jest wykorzystywany później (18)

Do szyfrowania wiadomości wykorzystamy prosty OTP. Na podstawie ustalonego klucza sesji powiększonego o numer wiadomości, który będzie ziarnem losowości będzie generowany klucz OTP o tej samej długości co wiadomość. W ten sposób klucz będzie losowy i unikalny dla każdej wiadomości. Wiadomości będą szyfrowane przez zastosowanie funkcji XOR na szyfrowanej wiadomości używając klucza OTP. Odbiorca wiadomości będzie wykonywał te same kroki - generując klucz OTP i wykonując operację XOR na otrzymanej wiadomości. Zapewnimy spójność informacji przez zliczanie wiadomości po odczytaniu komunikatu DONE oraz zapobieganiu wysyłania kolejnych przez nadawcę przed otrzymaniem odpowiedzi od odbiorcy.

Przykładowo, dla wyżej ustalonego klucza sesji (18), oraz numeru wiadomości (1), zaszyfrowanie może wyglądać następująco:

- Klient chce przesłać wiadomość “HelloDONE”, czyli [72, 101, 108, 108, 111, 68, 79, 78, 69], wiadomość ma długość 9
- Załóżmy że za pomocą ziarna (19) można wygenerować następujący klucz OTP: [10, 15, 12, 14, 9, 11, 13, 16, 18]
- Za pomocą operacji XOR wyliczamy zaszyfrowaną wiadomość [66, 90, 96, 98, 102, 79, 66, 94, 87]
- Klient wysyła wiadomość 9|BZ`bfOB^W|<MAC>
- Serwer także znając ziarno wylicza klucz OTP, używa operacji XOR i czyta wiadomość HelloDONE

Do wyznaczenia kodu MAC użyjemy algorytmu HMAC, który wykorzystuje klucz sesji, zaszyfrowaną wiadomość i funkcję hashującą SHA-256. Najpierw hashowany jest klucz sesji poddany XORowaniu z ustaloną wartością wewnętrznego klucza ipad (54) i skonkatenowany z wiadomością, a docelową wartość otrzymujemy przez hashowanie klucza sesji XORowanego z ustaloną wartością zewnętrznego klucza opad (92) i skonkatenowanego z wynikiem poprzedniej operacji.

Sposób realizacji mechanizmu integralności i autentyczności

W naszym projekcie implementujemy mechanizmy zapewniające integralność i autentyczność poprzez encrypt-then-mac.
Najpierw szyfrujemy wiadomość, a następnie dołączamy do niej kod uwierzytelnienia wiadomości. Po otrzymaniu takiej wiadomości odbiorca będzie mógł używając klucza sesji sam obliczyć kod MAC i potwierdzić autentyczność oraz integralność otrzymanego komunikatu.
Jeśli kod MAC byłby niezgodny oznaczałoby to, że nadawca nie użył poprawnego klucza sesji lub wiadomość została zmieniona w trakcie przesyłania. Poza tym, dzięki zastosowaniu szyfrowania wiadomości, po jej przechwyceniu, treść nie zostanie ujawniona.