Warszawa dn.19.02.2017

# Implementacja protokołu Tree-based Group Diffie-Hellman(TGDH) z szyfrem AES na przykładzie chatu grupowego

Autorzy:
Michał Krzemiński
Patryk Bąk

Spis treści:



## 1. Opis aplikacji:

Projekt składa się z dwóch autonomicznych aplikacji realizujących scenariusz klient-serwer. Rolą aplikacji klienckiej jest wysyłanie zaszyfrowanej wiadomości do serwera, który pracuje w trybie broadcast i wysyła tą wiadomość do wszystkich aplikacji klienckich zarejestrowanych w chacie. Możliwe też jest komunikowanie się z wybranym użytkownikiem z chatu. Obie aplikacje są zaimplementowane w środowisku Java, są to aplikacje konsolowe.

## 2. Opis protokołu TGDH

Protokół TGDH służy do obliczania przez wszystkich klientów jednego, wspólnego klucza grupowego który jest następnie używany do szyfrowania wiadomości algorytmem symetrycznym AES. Protokół ten zakłada istnienie abstrakcyjnego drzewa binarnego, w którym każdy user jest reprezentowany jako liść w tym drzewie. Każdy liść (user) posiada klucz prywatny K - wygenerowany pseudolosowy klucz AES 128 bitów, oraz klucz publiczny BK(blind key) obliczony używając K oraz p i g (grupa multiplikatywna dostarczona przez serwer). Liczby p i g zostały wzięte ze standardu IETF (https://www.ietf.org/rfc/rfc5114.txt).

p jest to duża liczba pierwsza:
p = B10B8F96 A080E01D DE92DE5E AE5D54EC 52C99FBC FB06A3C6
9A6A9DCA 52D23B61 6073E286 75A23D18 9838EF1E 2EE652C0
13ECB4AE A9061123 24975C3C D49B83BF ACCBDD7D 90C4BD70
98488E9C 219A7372 4EFFD6FA E5644738 FAA31A4F F55BCCC0
A151AF5F 0DC8B4BD 45BF37DF 365C1A65 E68CFDA7 6D4DA708
DF1FB2BC 2E4A4371

g jest to generator grupy multiplikatywnej:

g = A4D1CBD5 C3FD3412 6765A442 EFB99905 F8104DD2 58AC507F
D6406CFF 14266D31 266FEA1E 5C41564B 777E690F 5504F213
160217B4 B01B886A 5E91547F 9E2749F4 D7FBD7D3 B9A92EE1
909D0D22 63F80A76 A6A24C08 7A091F53 1DBF0A01 69B6A28A
D662A4D1 8E73AFA3 2D779D59 18D08BC8 858F4DCE F97C2A24
855E6EEB 22B3B2E5

$$BK_{<2,0>} = g^{K<2,0>} \bmod p$$

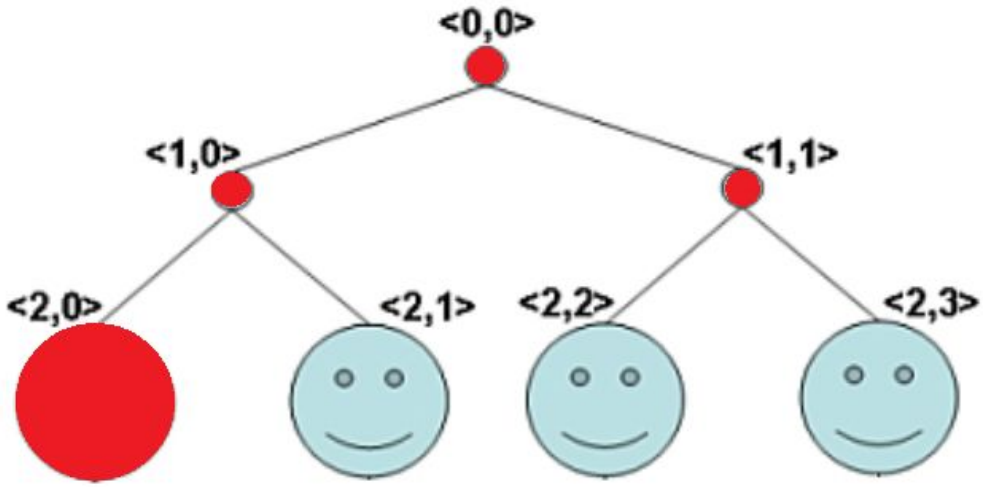
Po obliczeniu klucza publicznego (BK) przez każdego usera z pary liści w drzewie binarnym następuje wymiana kluczy publicznych sąsiadów z tej samej gałęzi drzewa binarnego. Dla przykładu przyjmijmy lewa gałąź <2,0> i <2,1>. Używając klucz publiczny sąsiada każdy węzeł oblicza klucz prywatny swojego rodzica K<1,0> w drzewie:

$$K_{<1,0>} = BK_{<2,1>}{}^{K<2,0>} \bmod p$$

Następnie obliczany jest klucz publiczny dla tego węzła drzewa (BK)

$$BK_{<1,0>} = g^{K<1,0>} \bmod p$$

Kolejnym krokiem obliczania klucza grupowego jest obliczenie klucza prywatnego korzenia drzewa K<0,0>. Każdy user liczy go lokalnie używając kluczy publicznych ścieżki między swoją pozycją i korzeniem oraz klucza publicznego drugiego dziecka korzenia (na poniższym drzewie zaznaczono na czerwono):



$$K_{<0,0>} = BK_{<1,1>}^{K<1,0>} \bmod p$$

Ten klucz nie jest używany do szyfrowania wiadomości. Używając funkcji skrótu MD5 z wyliczonego 128 bajtowego klucza (K<0,0>) generujemy 128 bitowy skrót którego będziemy używać do szyfrowania wiadomości szyfrem AES.

## 3. Bezpieczeństwo

Siła protokołu jest oparta na trudności obliczenia logarytmów dyskretnych w ciałach skończonych, dlatego też potencjalny atakujący który przechwyci wiadomość wysyłaną kanałem publicznym nie jest w stanie odtworzyć z niej klucza prywatnego który znajduje się lokalnie u użytkownika. Wspólny sekret dla pary liści jest obliczany w każdym z nich dlatego też nie jest wymagane istnienie bezpiecznego kanału lub zaufanej trzeciej strony. Protokół TGDH rozszerza możliwości protokołu Diffiego-Hellmana i pozwala na dzielenie się sekretem przez większą liczbę użytkowników.

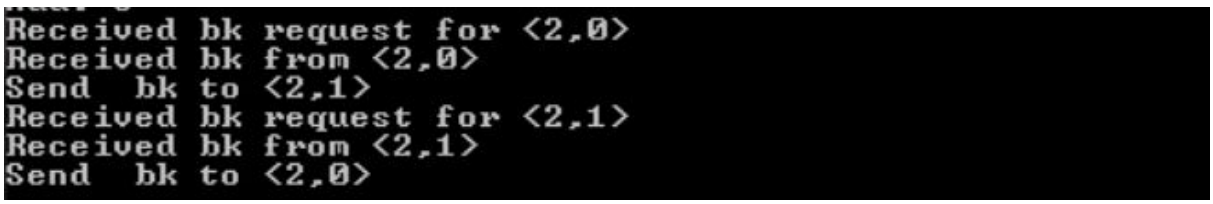
Do cech protokołu TGDH należy:

1. **Bezpieczeństwo klucza grupowego**, polegające na tym że nawet przy znajomości wartości klucza grupowego nie jest możliwe obliczenie kluczy prywatnych użytkowników
2. **Bezpieczeństwo "w przód"**, polegające na tym że znając podzbiór kluczy prywatnych użytych do obliczenia poprzedniego klucza grupowego nie jest możliwe odgadnięcie podzbioru kluczy prywatnych które będą służyły w obliczeniu kolejnego klucza grupowego
3. **Bezpieczeństwo "w tył"**, polegające na tym że znając podzbiór kluczy prywatnych który będzie użyty do obliczenia przyszłego klucza grupowego, nie jest możliwe odgadnięcie podzbioru kluczy prywatnych, które służyły w obliczeniu poprzedniego klucza grupowego
4. **Niezależność klucza prywatnego**, polegająca na tym że atakujący który zna poprawny podzbiór kluczy prywatnych użyty do obliczenia bieżącego klucza grupowego, nie jest w stanie odgadnąć reszty kluczy prywatnych ze zbioru kluczy prywatnych
5. Klucz grupowy GK obliczony przez każdego z użytkowników, w trakcie komunikacji pomiędzy nimi jest odświeżany co ustaloną liczbę wysłanych wiadomości i ponownie liczony, operacja ta jest niewidoczna z punktu zwykłego użytkownika i nie utrudnia potencjalnej komunikacji w trakcie obliczania nowego klucza. Operacja ta zapewnia świeżość klucza grupowego.

W celu zwiększenia bezpieczeństwa i zapewnieniu autentyczności wysyłanych kluczy, dane wysyłane publicznym kanałem pomiędzy użytkownikami a serwerem są podpisywane cyfrowo przez serwer używając standardu podpisu cyfrowego RSA. Poprawny podpis oznacza, że wiadomość pochodzi od właściwego nadawcy, który nie może zaprzeczyć faktowi jej nadania oraz, że wiadomość nie została zmieniona podczas transmisji. Autentyczność kanału pomiędzy serwerem a klientem zapewnia obronę przed atakiem “Man in the middle”, w którym to atakujący pośredniczy w komunikacji pomiędzy nadawcą a adresatem wiadomości bez ich wiedzy podsłuchując tym samym i wykradając cenne informacje.

## 4. Sposób działania aplikacji

Scenariusz realizujący chat grupowy dla 4 użytkowników jest generowany poprzez uruchomienie pliku wykonywalnego standardScenario.bat, który uruchamia serwer broadcastowy oraz 4 użytkowników podłączających się kolejno do serwera.

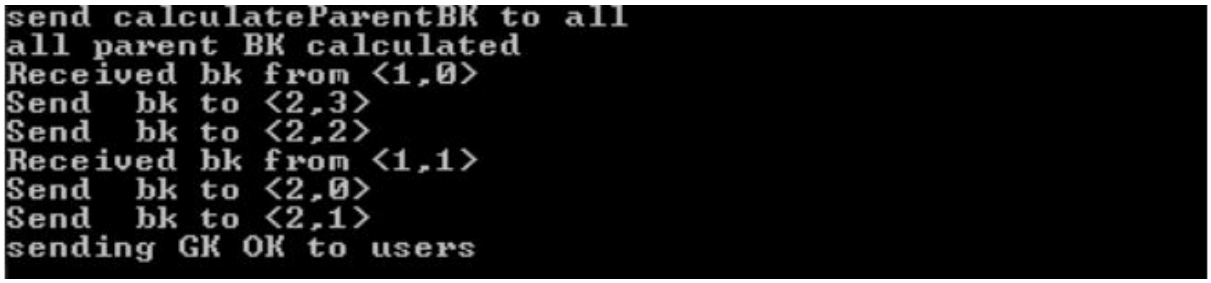



*Zrzut z konsoli serwera*

Kolejnym krokiem jest inicjalizacja protokołu Diffiego-Hellmana pomiędzy sąsiadującymi liśćmi w drzewie binarnym użytkowników poprzez poproszenie przez serwer BK pierwszego użytkownika <2,0> oraz przesłanie otrzymanego BK do użytkownika <2,1>. Następnie użytkownik <2,1> wysyła swój BK do użytkownika <2,0>. Po otrzymaniu BK sąsiada każdy użytkownik wysyła potwierdzenie do serwera że otrzymał klucz od sąsiada i jest gotowy do liczenia klucza prywatnego rodzica. Analogiczna procedura dystrybucji BK pomiędzy użytkownikami, wykonuje się równolegle dla prawej gałęzi drzewa binarnego(użytkownicy <2,2> ; <2,3>).

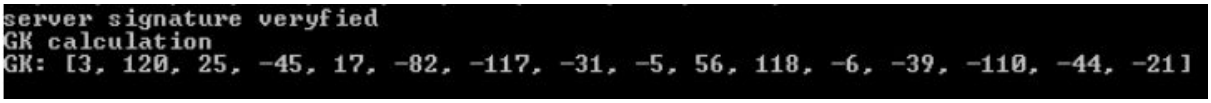

*Zrzut z konsoli serwera*

Analogicznie dla drugiej gałęzi (userzy 2 i 3). Gdy obie gałęzie mają BK swoich sąsiadów przechodzimy do kolejnego punktu.




*Zrzut z konsoli serwera*

Serwer wysyła do każdego usera żądanie policzenia swojego rodzica, następnie zachodzi dystrybucja kluczy publicznych rodziców, tak aby userzy z gałęzi lewej (0 i 1) mieli BK rodzica prawej gałęzi <1,1> oraz aby userzy z gałęzi prawej (2i 3) mieli BK rodzica lewej <1,0>. Gdy każdy user ma wszystkie BK potrzebne do obliczenia klucza grupowego, serwer wysyła komunikat GK ok do wszystkich klientów. Zezwala to użytkownikom na policzenie klucza grupowego lokalnie.




*Zrzut z konsoli klienta*

Klucz grupowy obliczany jest używając wzoru (Przykład dla użytkownika 0):

$$\mathrm{K}_{<0,0>} = \mathrm{BK}_{<1,1>}\text{^}(\mathrm{BK}_{<2,1>}{}^{\mathrm{K}<2,0>} \bmod \mathrm{p}) \bmod \mathrm{p}$$

Wynikiem jest klucz K<0,0> 128 bajtowy, nie jest on używany do szyfrowania. Do szyfrowania służy funkcja skrótu MD5 (128 bitów) tego klucza, szyfrem symetrycznym AES.