9. Dystrybucja kluczy kryptograficznych

9.1 Algorytm Diffiego-Hellmanna

Jeśli mamy ustalony system kryptograficzny to o jego bezpieczeństwie decyduje sposób uzgadniania kluczy kryptograficznych.

Sposób uzgadniania kluczy nazywa się również
- systemem wymiany kluczy lub
- systemem dystrybucji kluczy lub
- algorytmem wymiany kluczy lub
- protokołem wymiany kluczy

Dobrym algorytmem często stosowanym np. do uzgadniania kluczy w symetrycznych systemach kryptograficznych (np. tzw. kluczy sesyjnych) jest algorytm Diffiego – Hellmanna uzgadniania klucza.

Algorytm Diffiego-Hellmana uzgadniania klucza

1.Wybieramy duże ciało skończone F_q , liczba q (liczba elementów w ciele F_q) jest publicznie znana. Liczba elementów q w ciele skończonym jest zawsze potęgą pewnej liczby pierwszej.

2.Wybieramy generator g grupy multiplikatywnej F_q^*. Z pewnych względów jest lepiej jeśli g jest generatorem grupy multiplikatywnej F_q^* ale nie jest to konieczne. Element g jest publicznie znany. Istnieje twierdzenie, które mówi, że każda grupa multiplikatywna ciała skończonego jest cykliczna tzn. zawsze istnieje w niej generator.

3.Strony A i B chcą uzgodnić klucz symetryczny, jakiego będą używać. Strona A wybiera losowo tajną liczbę $a \in <2, q-2>$.Ogólnie rzecz biorąc możemy przyjąć, że $a \in <1, q-1>$ ale $a = 1$ i $a = q-1$ nie są bezpiecznymi wyborami. Następnie strona A oblicza g^a i podaje g^a do publicznej wiadomości lub przesyła do strony B. Wybór a jest tajemnicą strony A.

4. Strona B wybiera losowo tajną liczbę $b \in <2, q-2>$. Ogólnie rzecz biorąc możemy przyjąć, że $b \in <1, q-1>$ ale $b = 1$ i $b = q-1$ nie są bezpiecznymi wyborami. Następnie strona B oblicza g^b i podaje g^b do publicznej wiadomości lub przesyła do strony A. Wybór b jest tajemnicą strony B.

5.Wspólny tajny klucz to g^{ab}. Obie strony A i B potrafią łatwo obliczyć tę liczbę, bo

$$(g^a)^b = g^{ab} \text{ i } (g^b)^a = g^{ab}$$

natomiast inne osoby nie znają a i b, znają tylko g^a i g^b i nie potrafią znaleźć g^{ab}!

Bezpieczeństwo algorytmu Diffiego-Hellmana.

Bezpieczeństwo algorytmu Diffiego-Hellmana wynika z faktu, że tzw. problem Diffiego-Hellmana polegający na obliczeniu g^{ab}, gdy znane są jedynie g^a oraz g^b jest problemem obliczeniowo trudnym.

Zauważmy, że gdyby istniał efektywny algorytm rozwiązujący problem logarytmu dyskretnego to również istniałby efektywny algorytm rozwiązujący problem Diffiego-Helmana.

Reasumując bezpieczeństwo algorytmu Diffiego-Hellmana wynika z faktu, że zarówno problem Diffiego-Hellmana jak i problem logarytmu dyskretnego są trudne obliczeniowo.

Algorytm Diffiego-Hellmana w wersji podanej wyżej jest podatny na atak typu „man in the midlle”.

9.2 Protokół szerokogębnej żaby

Protokół szerokogębnej żaby (ang. Wide-Mouth Frog protocol) jest prostym protokołem wykorzystującym szyfr z kluczem symetrycznym E do dystrybucji klucza. Celem jest przesłanie klucza stronie B przez stronę A. Protokół wykorzystuje tzw. zaufaną trzecią stronę T .

Strona A wybiera klucz sesyjny K. Za pomocą protokołu szerokogębnej żaby przekazuje się klucz od A do B za pośrednictwem zaufanej trzeciej strony T w dwu krokach. Stosuje się przy tym zsynchronizowane zegary (i znaczniki czasu) podnoszące bezpieczeństwo protokołu.

Strona A i B mają swoje wspólne z T klucze oznaczmy je przez k_A i k_B .

Protokół szerokogębnej żaby składa się z 2 kroków i jest następujący.

1. Strona A wylosowała klucz i wysyła do strony T takie dane:

$$A,\ E('T_1, B, K', k_A)$$

gdzie A jest identyfikatorem strony A a wiadomość $'T_1, B, K'$ podaje znacznik czasu T_1 , identyfikator strony B i proponowany klucz K.

2. Zaufana trzecia strona odszyfrowuje wiadomość strony A i przesyła do strony B kryptogram

$$E('T_2, A, K', k_B)$$

z własnym znacznikiem czasu T_2, identyfikatorem strony A i zaproponowanym przez stronę A kluczem K .

Istota rzeczy. Zaufana trzecia strona pełni rolę zaufanego operatora centrali telefonicznej. Jako strona A nie musimy znać klucza k_B strony B wystarczy znać tylko jeden klucz k_A .

9.3 Protokół Needhama-Schroedera

Protokół Needhama-Schroedera jest protokołem wykorzystującym szyfr z kluczem symetrycznym E do dystrybucji klucza. Celem jest przesłanie klucza stronie B przez stronę A. Protokół wykorzystuje tzw. zaufaną trzecią stronę T . Strony A i B mają swoje wspólne z T klucze oznaczmy je przez k_A i k_B.
Protokół Needhama-Schroedera składa się z 6 kroków i jest następujący.

1.Strona A przesyła do T wiadomość

$$‘A, B, r_A’$$

gdzie A jest identyfikatorem strony A, B jest identyfikatorem strony B a r_A jest liczbą losową.

2. Zufana trzecia strona T generuje klucz sesyjny K i przesyła go stronie A w kryptogramie:

$$E('r_A, B, K, E('K, A', k_B)', k_A)$$

3. Strona A deszyfruje kryptogram otrzymuje własne r_A co ją upewnia, że protokół nie jest powtórzeniem jakiegoś fragmentu starej wymiany danych (atak typu „replay”). Strona A otrzymuje jednocześnie klucz sesyjny K i kryptogram $E('K, A', k_B)$, który odsyła stronie B.

4. Strona B deszyfruje soim kluczem k_B kryptogram $E('K, A', k_B)$ otrzymując klucz sesyjny K i identyfikator strony A. Następnie strona B losuje liczbę r_B i szyfruje ją kluczem K wysyłając do strony A kryptogram

$$E(r_B, K)$$

5. Strona A deszyfruje kluczem K kryptogram $E(r_B, K)$ uzyskując r_B. Oblicza liczbę $r_B - 1$ i szyfruje ją kluczem K odsyłając do B kryptogram

$$E(r_B - 1, K)$$

6.Strona B deszyfruje kryptogram $E(r_B - 1, K)$. Jeśli po deszyfracji uzyskana liczba jest równa $r_B - 1$ to znaczy, że połączenie działa poprawnie.

Literatura

[1] A. Menezes, P. Oorschot, S. Vanstone; Handbook of Applled Cryptography; CRC Press Inc., 1997. (treść jest na stronie: http://cacr.math.uwaterloo.ca/hac)

[2] J.Stokłosa, T.Bilski,T.Pankowski; Bezpieczeństwo danych w systemach informatycznych; PWN, Warszawa 2001.

[3] B.Schneier; Kryptografia dla praktyków; WNT, 2002.