

Protokoły z hasłem. Część I

Stefan Dziembowski

Na czym polega problem?

Bezpieczeństwo opieramy się na hasle zapamiętanym przez człowieka.

Problem:

Jeśli zbiór \mathcal{H} potencjalnych haseł nie jest zbyt duży, to takie hasła są podatne na atak słownikowy.

Atak słownikowy — przykład:

π — hasło (zapamiętane przez człowieka)

$C = E(\pi, M)$ — szyfrogram wiadomości M

Przeciwnik robi tak:

Szuka takiego $\tilde{\pi} \in \mathcal{H}$ że $D(\tilde{\pi}, C)$ „ma sens”.

Jeśli M ma jakąś nadmiarowość, to w ten sposób możemy znaleźć właściwe hasło,

albo przynajmniej uda mu się zredukować liczbę potencjalnych haseł.

Plan

- **Dzisiaj:**

- Protokoły symetryczne — obie strony znają tylko hasło. To się nazywa **Encrypted Key Exchange (EKE)**
Rozważamy protokoły oparte na:
 - Protokole Diffiego-Hellmana
 - RSA

- **Za tydzień:**

- Protokoły asymetryczne — jedna ze stron („klient”) zna klucz publiczny drugiej strony („serwera”).
- Zagadnienia praktyczne.

EKE

Protokoły **Encrypted Key Exchange (EKE)** zostały wprowadzone przez Bellare i Merritt w pracy [BM92].

Idea

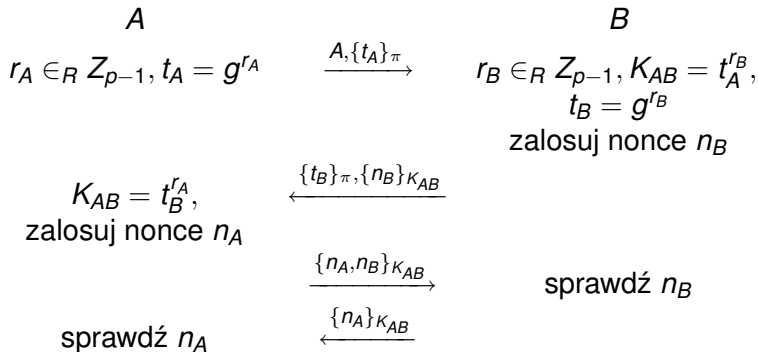
π — dzielone hasło (albo hash hasła)

Użyjemy Diffiego-Hellmana

szyfrując komunikaty kluczem π .

Pierwszy pomysł na EKE (trochę uproszczony)

Działamy w grupie Z_p^*
 g — generator, L — parametr



Co jest szyfrowane za pomocą hasła π

Za pomocą π szyfrowane są $t_A = g^{r_A}$ i $t_B = g^{r_B}$.

Przeciwnik widzi kryptogramy $c_A = \{t_A\}_\pi$ i $c_B = \{t_B\}_\pi$

Niech $D_\pi(C)$ oznacza odszyfrowanie C za pomocą klucza π .

Możemy więc przeprowadzić atak słownikowy

\mathcal{H} — zbiór wszystkich haseł.

Niech:

$$\mathcal{H}' := \{\tilde{\pi} : D_\pi(c_A) \text{ i } D_\pi(c_B) \text{ należą do } Z_p^*\}$$

Atak słownikowy - nieformalna analiza

n — długość wejścia szyfru symetrycznego.

Jeśli p jest losową liczbą pierwszą długości n , to przeciętnie

$$p \approx \frac{n}{2}.$$

Czyli szansa, że $D_\pi(C_A) \in Z_p^*$ wynosi około $\frac{1}{2}$.

Możemy więc oczekiwać, że $|\mathcal{H}'| \approx \frac{|\mathcal{H}|}{4}$.

Zatem po logarytmicznej liczbie zaobserwowanych sesji przeciwnik znajdzie hasło π .

Ten rodzaj ataku nazywa się **partition attack**.

Jak poradzić sobie z tym atakiem?

Nieeleganckie rozwiązanie problemu

Można wybrać p równe „prawie” 2^n .

Na szczęście są lepsze rozwiązania. . .

Pierwszy pomysł

Pomińmy szyfrowanie za pomocą π . . .

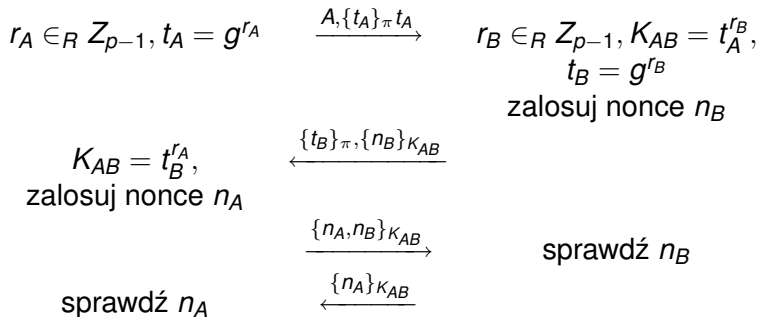
To nie ma sensu, bo gdzieś to π trzeba użyć.

Może można chociaż pominąć jedno z szyfrowań za pomocą π ?

W oryginalnej pracy [BM92] twierdzono, że można.

A tymczasem **nie** można.

Co będzie gdy usuniemy szyfrowanie pierwszej wiadomości



Wygląda OK, bo bez znajomości π trudno wyprodukować $\{t_B\}_\pi$
(ze znajomością t_B)

Tymczasem możliwy jest taki atak

Przeciwnik C podszywa się pod A

$$r_C \in_R \mathbb{Z}_{p-1}, t_A = g^{r_C} \xrightarrow{A, t_A} r_B \in_R \mathbb{Z}_{p-1}, K_{AB} = t_A^{r_B},$$
$$t_B = g^{r_B}$$

$$\underbrace{\{t_B\}_\pi}_{c_1}, \underbrace{\{n_B\}_{K_{AB}}}_{c_2}$$

złóż nonce n_B

Jeśli n_B zawiera nadmiarowość, to możliwa jest taka eliminacja potencjalnego hasła $\tilde{\pi}$:

- 1 Odszyfrowujemy c_1 hasłem $\tilde{\pi}$.
Niech \tilde{t}_B będzie wynikiem odszyfrowania.
- 2 Odszyfrowujemy c_2 za pomocą $\tilde{t}_B^{r_C}$.
Sprawdzamy, czy to co wyszło „ma sens”

PPK i PAK — eleganckie rozwiązanie problemów EKE

Protokół **Password Protected Key (PPK) exchange** jest modyfikacją EKE mającą na celu zapobieżenie „partition attacks”.

Protokół ten jest wariantem protokołu **Password Authenticated Key (PAK) exchange**.

Szyfrowanie odbędzie się za pomocą metod „algebraicznych”.

Szczegóły można znaleźć w [BMP00], oraz w pracach tam cytowanych.

PPK — c.d.

Wszystko będzie działać się w grupie Z_p^* , gdzie $p = rq + 1$ i $r \perp q$.

Konkretnie:

- 1 Diffie-Hellman odbędzie się w podgrupie G , gdzie $G \subset Z_p^*$ jest rzędu q .

Zauważmy: jeśli γ jest generatorem Z_p^* , to

$$G = \{\gamma^{1 \cdot r}, \gamma^{2 \cdot r}, \dots, \gamma^{(q-1)r}\}.$$

- 2 „Zaszyfrowanie” $x \in G$ odbędzie się przez pomnożenie przez element $P \in G$ obliczony na podstawie π .

Jak takie P uzyskać?

H_1 — funkcja hashująca dająca wartości w Z_p^*

$$P := H_1(A, B, \pi)^r$$

Bo jeśli $H_1(A, B, \pi)$ jest równe γ^i (dla pewnego i), to

$$H_1(A, B, \pi)^r = \gamma^{ir} \in G \text{ (dla pewnego } j = 1, \dots, q-1)$$

PAK

H_1, H_2, H_3 — funkcje hashujące
 $P_1 = H_1(A, B, \pi)^r$ i $P_2 = H_2(A, B, \pi)^r$

A		B
$r_A \in_R Z_q, t_A = g^{r_A}, m = t_A P_1$	\xrightarrow{m}	$t_A = m/P_1, r_B \in Z_q, t_B = g^{r_B}$
$t_B = m'/P_2$	$\xleftarrow{m'}$	$m' = t_B P_2$
$Z_{AB} = t_B^{r_A}$		$Z_{AB} = t_A^{r_B}$

$$K_{AB} = H_3(A, B, m, m', Z_{AB}, \pi)$$

Do tego warto by dodać jakiś handshake żeby strony uzyskały potwierdzenie, że klucz został uzgodniony.

Rozszerzone (augmented) EKE

Obserwacja

Często jest tak, że nie chcemy, żeby serwer znał nasze hasło π .
Zamiast tego dajemy mu tylko hash tego hasła $H(\pi)$.

Pomysł

Przy wykonaniu protokołu będziemy dowodzić znajomości hasła π .

Serwer będzie potrafił to zweryfikować tylko na podstawie $H(\pi)$.

Użyjemy **podpisów ElGamala**.

Podpisy ElGamala

Dane

Grupa Z_p^* i jej generator g .

Klucze

Kluczem **prywatnym** jest $a \in Z_{p-1}^*$. Kluczem **publicznym** jest $\beta = g^a$

Idea

Podpisem na x będzie taka wartość którą można obliczyć tylko przy znajomości a .
A zweryfikować ją można za pomocą β .

Podpisy ElGamala - c.d.

$$\beta = g^a$$

Podpis na x

$Sig_K(x, k) = (\gamma, \delta)$, gdzie k jest losowe i $\gamma = g^k$.

Jaki jest wzór na δ ?

Łatwiej zacząć od weryfikacji

Weryfikacja: $g^x = \beta^\gamma \cdot \gamma^\delta$

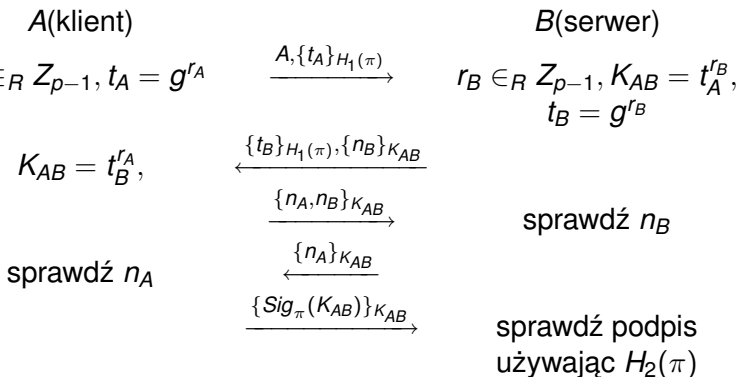
Stąd można wyprowadzić δ :

$$g^x = g^{a\gamma} \cdot g^{k\delta}$$

$$\text{Stąd: } \delta = (x - a\gamma)k^{-1} \bmod (p - 1)$$

Jak można wykorzystamy podpisy ElGamala?

Zamiast π użyjemy $H_1(\pi)$ (gdzie H_1 jest funkcją hashującą).
Oprócz tego serwer będzie znał $H_2(\pi) = g^\pi$.



Mały problem

$$\{Sig_{\pi}(K_{AB})\}_{K_{AB}}$$

Czy użycie K_{AB} w ostatniej wiadomości jest rozsądne?

Co jeśli K_{AB} kiedyś wycieknie?

Przeciwnik odszyfruje $Sig_{\pi}(K_{AB})$ i przeprowadzi atak słownikowy.

Czyli lepiej tego nie robić. . .

Jak sobie z tym poradzić?

Z klucza K_{AB} wyprowadzimy osobno klucz sejsyjny i osobno klucz na własne potrzeby protokołu ustalania klucza.

Jak to zrobić z RSA

Pierwszy pomysł

„Klucze” Diffiego-Hellmana zastępujemy efemerycznymi kluczami RSA.

zamiast:

$$\begin{array}{c} \xrightarrow{\{g^{r_A}\}_\pi} \\ \xleftarrow{\{g^{r_B}\}_\pi} \end{array}$$

będziemy mieć:

$$\begin{array}{c} \xrightarrow{\{(n, e)\}_\pi} \\ \xleftarrow{\{K_{AB}^e \bmod n\}_\pi} \end{array}$$

Problem

n nie wygląda losowo (bo losowa liczba będzie miała małe dzielniki), więc łatwo można przeprowadzić atak słownikowy. . .

Naprawa

Nie szyfrujemy modułu

$$\begin{array}{c} n, \{e\}_\pi \\ \xrightarrow{\hspace{1.5cm}} \\ \{K_{AB}^e \bmod n\}_\pi \\ \xleftarrow{\hspace{1.5cm}} \end{array}$$

Uwaga

e (takie, że $e \perp \varphi(n)$) może być wybrane losowo z jakiegoś przedziału $1, \dots, 2^L$,

ale trzeba uważać, bo takie e jest zawsze nieparzyste, więc lepiej wogóle nie transmitować najmniej znaczącego bitu e .

Poza tym trzeba zadbać, żeby n było niewiele mniejsze od potęgi 2.

Kolejny problem

Jeśli wycieknie klucz sesyjny K_{AB} , to przeciwnik może sprawdzić potencjalne hasło $\tilde{\pi}$:

- 1 Rozszyfruj pierwszy szyfrogram $\{e\}_\pi$ za pomocą potencjalnego hasła $\tilde{\pi}$. Niech \tilde{e} będzie wynikiem.
- 2 Oblicz $\{K_{AB}^{\tilde{e}} \bmod n\}_{p_i}$ i sprawdź czy wyszło to samo co w drugim kroku protokołu.

Jak temu zaradzić?

Zrandomizować szyfrowanie.

Atak

Ten atak pochodzi z [Pat97].

Przeciwnik wybiera n' w taki sposób, że $n' = pq$, gdzie $3|(p-1)$ i $3|(q-1)$:

$$\xrightarrow{n', X}$$

Lemat

Lemat

$f(x) = x^3$ jest (w Z_n^*) funkcją „9-na-1”.

Dowód

Z Chińskiego Twierdzenia o Resztach wystarczy udowodnić, że w Z_p^* i Z_q^* ta funkcją 3-na-1. Dla ustalenia weźmy Z_p^* .

γ — generator Z_p^*

$$p = 3w + 1.$$

γ^i, γ^j — jakieś elementy.

Wtedy mamy $(\gamma^i)^3 = (\gamma^j)^3$ wttw gdy:

$\gamma^{3i-3j} = 1$. A to zachodzi wttw gdy:

$(3w) | 3(i - j)$. A to to samo co $w | (i - j)$

Czyli i musi być równe $j, j + w$, albo $j + 2w$

Co dalej?

B „odszyfrowuje” X za pomocą hasła π i otrzymuje jakiś e' .

Założmy, że e' jest podzielne przez 3 (wpp powtarzamy).

B wysła do A wiadomość $C = \{K_{AB}^{e'} \bmod n\}_{\pi}$.

Przeciwnik może zweryfikować potencjalne hasła $\tilde{\pi}$ przez sprawdzenie, czy odszyfrowane C jest resztą sześcienną.

Protokoły oparte na RSA

Istnieją bezpieczne protokoły oparte na RSA:

- **Open Key Exchange (OKE)** [Luc98]
- **Secure Network Authentication with Password Information (SNAPI)** [MPS00]

Literatura I



Steven M. Bellovin and Michael Merritt.

Encrypted key exchange: Password-based protocols secure against dictionary attacks.

In *Proc. IEEE Computer Society Symposium on Research in Security and Privacy*, pages 72–84, May 1992.

pdf.



Victor Boyko, Philip MacKenzie, and Sarvar Patel.

Provably secure password-authenticated key exchange using Diffie-Hellman.

In *Advances in Cryptology - Eurocrypt 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 156–171, 2000.

Literatura II



Stefan Lucks.

Open key exchange: How to defeat dictionary attacks without encrypting public keys.

In *Security Protocols – 5th International Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 79–90, 1998.



Philip MacKenzie, Sarvar Patel, and Ram Swaminathan.

Password-authenticated key exchange based on RSA.

In *Advances in Cryptology - Asiacrypt 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 599–613, 2000.

Literatura III



Sarvar Patel.

Number theoretic attacks on secure password schemes.

In *RSP: 18th IEEE Computer Society Symposium on Research in Security and Privacy*, pages 236–247, 1997.