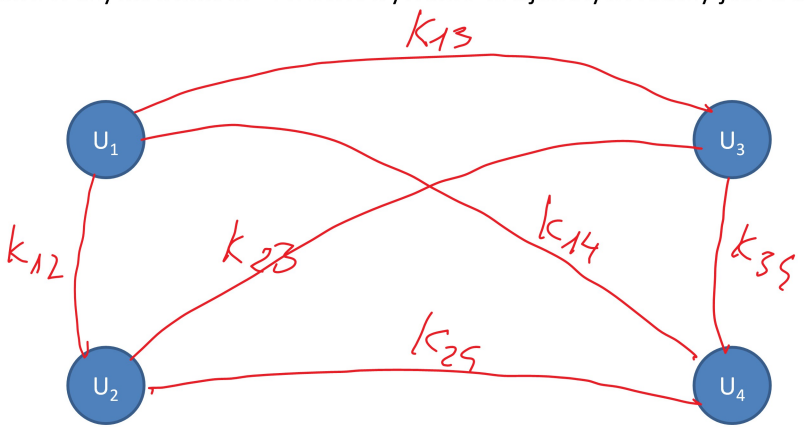
Kryptografia i bezpieczeństwo danych - wymiana kluczy

Sławomir Samolej
ssamolej.kia.prz.edu.pl
ssamolej@prz.edu.pl

W naszym wykładzie dochodzimy do końca dyskusji na temat szyfrowania z kluczem symetrycznym. Pozostało nam kilka zagadnień, które uzupełnią potrzebne informacje, aby zamknąć ten temat.

Zarządzanie kluczami

Problem: n użytkowników. Przechowywanie wzajemnych kluczy jest trudne.



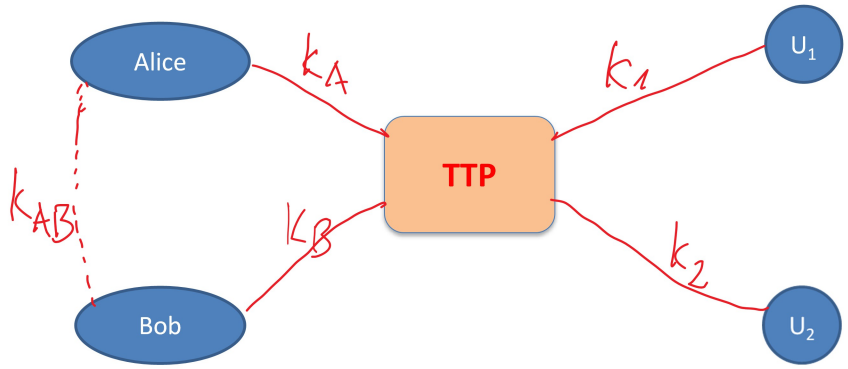
Całkowita liczba: O(n) kluczy na użytkownika

2

Wiemy, jak zaszyfrować i zapewnić integralność wiadomości. Problem, którego jeszcze nie rozstrzygnęliśmy jest jak ustalić wspólny klucz. Będzie to wprowadzenie w kolejny obszar kryptografii: kryptografii z kluczem publicznym.
Załóżmy, że na świecie istnieje n użytkowników i chcą oni zarządzać kluczami do szyfrowania informacji do siebie. Załóżmy, że mamy 4 użytkowników. Jedną z możliwych opcji jest, że każda para użytkowników współdzieli ze sobą tajny klucz. Możemy wtedy dla szyfrowanej komunikacji w takim gronie zdefiniować 6 kluczy, odpowiednio: K_{12}, K_{13}, K_{14}, K_{23}, K_{24}, K_{34}. Problem, który się tu pojawia, to konieczność zarządzania przez danego użytkownika tymi kluczami. Każdy z użytkowników musi zarządzać N kluczami, jeśli chce prowadzić wymianę informacji z N współpracownikami. Powstaje więc pytanie, czy można zarządzanie kluczy w takiej strukturze usprawnić.

Lepsze rozwiązanie

Zaufana trzecia strona online (Online Trusted 3rd Party (TTP))

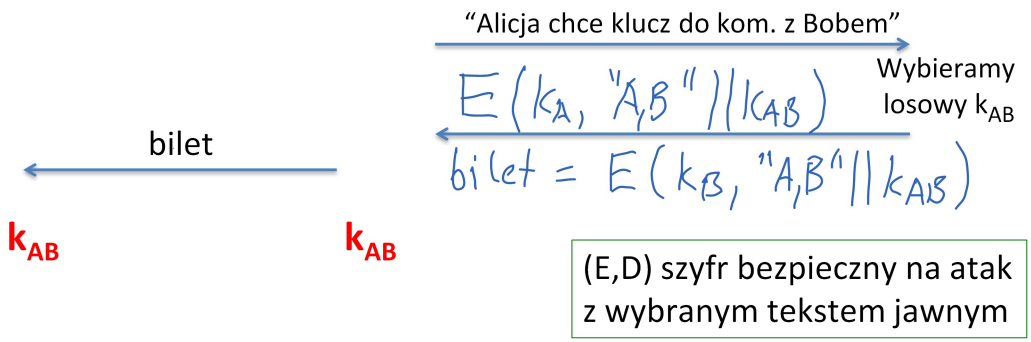


3

Okazuje się, że tak… Sposób zarządzania kluczami może zostać wsparty z zastosowaniem osobnej instytucji działającej w sieci, do której mamy zaufanie (ang. Online Trusted 3rd Party – TTP). Wtedy wystarczy, że każdy użytkownik jest w posiadaniu jednego klucza, do komunikacji właśnie z tą instytucją. Tak więc U_1 ma klucz K_1, U_2 ma klucz K_2, Alice ma klucz K_A, a Bob klucz K_B. Co zachodzi, jeśli Alice chce w sposób chroniony wymienić informację z Bobem? Muszą się oni zaangażować w pewien protokół, który na końcu zdefiniuje klucz szyfrowania komunikacji między nimi K_{AB}. Powstaje pytanie, jak wygenerować taki klucz. Zaproponujemy na początek pewien prosty protokół.

Generowanie kuczy: prosty protokół

Alice chce klucz współdzielony z Bobem. Bezpieczeństwo tylko przeciw podsłuchiwaniu.



(E,D) szyfr bezpieczny na atak z wybranym tekstem jawnym

4

Rozważmy pewien prosty przykładowy protokół wymiany kluczy. Bob posiada klucz K_B, Alice – klucz K_A. Te klucze są współdzielone z instytucją zaufaną. Instytucja zaufana posiada oba klucze. Rozważamy protokół wymiany kluczy, który zapewnia tylko ochroną przed podsłuchaniem. Bardziej zaawansowane protokoły zostaną omówione później. Nasz prosty protokół zaczyna się od wysłania informacji do TTP, że Alicja chce otrzymać klucz do komunikacji z Bobem. TTP (Trusted Third Party) wybiera losowy klucz K_{AB} i wysyła do Alice wiadomość składającą się z dwóch części. Pierwsza część zawiera zaszyfrowaną z zastosowaniem klucza Alice K_A wiadomość zawierającą (1) informację że przesyłany jest klucz połączenia między Alice and Bobem, (2) dołączony do tej informacji klucz K_{AB}. Druga część informacji nazywana jest biletem (ticket) i zawiera zaszyfrowaną z zastosowaniem klucza Boba K_B (1) informację, że przesyłany jest klucz do komunikacji między Alice i Bobem, (2) klucz komunikacji K_{AB}. Za każdym razem informacja (1) i klucz (2) są traktowane jako jedna wiadomość do zaszyfrowania. Obie części wiadomości wysyłane są do Alice. System szyfrowania zastosowany tutaj to system odporny na atak z wybranym tekstem jawnym (CPA- secure cipher). Jeśli Alice chce komunikować się z Bobem, to odszyfrowuje część wiadomości, która była zaszyfrowana jej kluczem i uzyskuje klucz K_{AB}. Potem wysyła „bilet" do Boba, który odszyfrowuje go swoim kluczem K_B i w takim razie oboje posiadają klucz K_{AB} do bezpiecznej wymiany informacji między nimi.

Pierwszym pytaniem, na które musimy odpowiedzieć, to dlaczego taki protokół jest bezpieczny, nawet jeśli założyliśmy tylko bezpieczeństwo algorytmu szyfrującego tylko na podsłuchiwanie.

Prosty protokół generowania kluczy – bezpieczeństwo, zastosowania

Alice chce mieć klucz do komunikacji z Bobem. Zapewnione jest tylko bezpieczeństwo przed podsłuchiwaniem.

Podsłuchujący widzi: $E(k_A,\ \text{“A, B”} \ll k_{AB})$; $E(k_B,\ \text{“A, B”} \ll k_{AB})$

(E,D) jest bezpieczne na atak z wybranym tekstem jawnym $\Rightarrow$
atakujący niczego się nie dowie o kluczu k_{AB}

Uwaga: TTP jest potrzebne do każdej wymiany kluczy, zna wszystkie klucze sesji.

(podstawa systemu Kerberos)

5

Bez wchodzenia w rozważania teoretyczne, atakujący widzi ciągle serię szyfrogramów i nie jest w stanie ich odróżnić od siebie. Nie jest w stanie odróżnić szyfrogramu klucza szyfru od szyfru przypadkowego zlepku bitów. Rozwiązanie jest bezpieczne na atak z wybranym tekstem jawnym.
Zwróćmy uwagę na rolę zaufanej instytucji trzeciej (TTP). Po pierwsze jest ona zaangażowana w każdą wyminę kluczy. Alice i Bob nie mogą rozpocząć bezpiecznej wymiany informacji, jeśli TTP nie jest online i nie jest dostępne dla nich obojga. TTP zna również wszystkie klucze sesji. Wtedy, gdy zaufana instytucja trzecia zostanie „przekupiona” lub zaatakowana może to doprowadzić do wycieku kluczy. Właśnie dlatego ta instytucja nazywa się ZAUFANĄ, bo zna wszystkie klucze.
Pokazane rozwiązanie w dalszym ciągu stosuje „tylko” kryptografię z kluczem symetrycznym. Jest szybkie i efektywne. Trochę trudno wyobrazić sobie takie rozwiązanie funkcjonujące jako węzeł w sieci WWW. Nie bardzo wiadomo, kto mógłby być traktowany jako ta trzecia zaufana instytucja. Natomiast jeśli chodziłoby o opracowanie systemu chronionej wymiany informacji w jednej instytucji, to takie rozwiązanie miałoby sens. Można ustalić konkretny komputer, jako tą zaufaną instytucję. W oparciu o mechanizm wymiany kluczy podobny do opisanego pracuje system Kerberos.

Podstawowy protokół: nie jest bezpieczny na aktywne ataki

Przykład: brak bezpieczeństwa na ataki z ponowieniem

Atakujący „nagrywa” sesję pomiędzy Alice i Bobem
- Na przykład zamówienie na książkę

Atakujący ponownie rozpoczyna taką samą sesję z Bobem
- Bob myśli, że Alice potrzebuje następnej kopii książki

6

Należy podkreśli, że omówiony właśnie protokół jest bardzo podstawowy. Miał on na celu pokazanie na przykładzie problemu dystrybucji kluczy. Jest on odporny tylko na podsłuchiwanie i zupełnie nie oferuje bezpieczeństwa przeciw atakowi aktywnemu. Rozważmy przykład, jak aktywny atakujący może zniszczyć ten protokół. Rozważmy atak powtórzeniowy (ang. replay attack). Atakujący może „nagrać” „konwersację” między Alice i Bobem. Możemy sobie wyobrazić, że Alice chce kupić książkę od Boba. Transakcja zachodzi, Bob otrzymuje zapłatę i wysyła kopię książki do Alice. Atakujący może ponownie „odtworzyć” konwersację pomiędzy Alice i Bobem. Bob nie będzie w stanie sprawdzić, czy pod Alice ktoś się nie podszył, więc wyśle jej ponownie książkę i rachunek do zapłacenia.
Protokół nie nadaje się wiec do aplikacji. Jego bezpieczna wersja zostanie om omóiona później.

Kluczowe pytanie

Czy możemy wygenerować współdzielone klucze bez trzeciej zaufanej instytucji dostępnej **online**?

Odpowiedź: Tak!

Punkt początkowy do rozważań nad kryptografią z kluczem publicznym:

- Merkle (1974), Diffie-Hellman (1976), RSA (1977)
- Ostatnio: Szyfrowanie na bazie ID (BF 2001),
  Funkcjonalne szyfrowanie (BSW 2011)

7

Omówiony protokół może doprowadzić do postawienia ważnych pytań. Czy możemy zbudować protokół wymiany kluczy , który jest bezpieczny zarówno na podsłuchiwanie jak i na aktywne ataki? Czy możemy z budować protokół wymiany kluczy niewymagający ciągle aktywnej i zawsze widocznej w Internecie zaufanej instytucji trzeciej?
Okazuje się, że odpowiedzi na oba pytania brzmią tak. I jest to możliwe dzięki nowym konstrukcjom kryptograficznym nazywanym kryptografią z kluczem publicznym. U podstaw tych konstrukcji leżą trzy pomysły. Pierwszy, opublikowany przez Merkle, prawie 50 lat temu dotyczył sposobu wymiany kluczy. Drugi opracowany przez dwie osoby Diffie'go i Hellman'a dotyczył koncepcji kryptografii z kluczem publicznym. Trzeci, opracowany przez Rivest'a, Shamir'a i Adlemman'a dotyczył algorytmu szyfrowania stosującego kryptografię z kluczem publicznym. Algorytm RSA jest wciąż popularnym narzędziem szyfrowania według tej „nowej" koncepcji kryptografii. Watro zaznaczyć, że szyfrowanie z kluczem publicznym jest w dalszym ciągu rozwijane, o czym mogą świadczyć pokazane na slajdzie kolejne koncepcje rozwijające ten pomysł. Pierwsza z prac omawia nowy sposób zarzadzania kluczami, druga dotyczy podawania kluczy, które tylko częściowo odszyfrowują wiadomości.

Wymiana kluczy bez zaufanej trzeciej instytucji będącej online (1)

Cel: Alice i Bob chcą współdzielonego klucza, ukrytego przed podsłuchaniem.

- Dotąd: zapewnialiśmy tylko bezpieczeństwo przeciw podsłuchiwaniu

Czy to jest wykonalne z zastosowaniem szyfrowania symetrycznego?

8

Rozważamy, czy możemy skonstruować protokół wymiany kluczy bez trzeciej instytucji zaufanej. Protokół ma zapewnić tylko ochronę przed podsłuchiwaniem. Alice i Bob , którzy nigdy się wcześniej nie spotkali na koniec protokołu otrzymują wspólny klucz do prowadzenia chronionej komunikacji. Będziemy się też zastanawiali, czy taka konstrukcja jest w ogóle możliwa przy zastosowaniu tylko technik znanych z kryptografii z kluczem symetrycznym.

Łamigłówka Merkle'a (1974)

Odpowiedź: tak, ale rozwiązanie jest nieefektywne

Główne narzędzie: łamigłówki

- Problemy mogą być rozwiązane z pewnym wysiłkiem
- Przykład: E(k,m) jest szyfrem symetrycznym, gdzie $k \in \{0,1\}^{128}$
  - **łamigłówka(P) = E(P, "message")** gdzie $P = 0^{96} \| b_1 \ldots b_{32}$
  - Cell: znajdź P przez wypróbowanie wszystkich 2^{32} możliwości

9

Okazuje się, że istnieje takie rozwiązanie, ale jest niewydajne…
Możemy dokonać wymiany kluczy z zastosowaniem znanych nam do tej pory mechanizmów szyfrowania blokowego, obliczania funkcji hash itp. bez konieczności istnienia dołączonej do sieci zaufanej trzeciej instytucji. Jednak otrzymane protokoły są bardzo niewydajne i w konsekwencji nieużywane w praktyce.
Przeanalizujemy jedno z takich rozwiązań. Zostało opracowane przez Ralph'a Merkle, kiedy był na studiach inżynierskich w ramach seminarium. Jego profesor nie zrozumiał znaczenia tego odkrycia. Merkle skończył studia inżynierskie i przeniósł się do Stanford, gdzie dał duży wkład w rozwój kryptografii z kluczem publicznym, współpracując z Marty Hellman'em. Głównym narzędziem w opracowanym protokole wymiany kluczy jest „łamigłówka". Łamigłówka to problem, który jest trudny do rozwiązania, ale może być rozwiązany przy włożeniu pewnego wysiłku. Przykładem takiej łamigłówki może być następujące zadanie. Mamy szyfr symetryczny, który stosuje klucze o długości 128 bitów (np. AES). Przygotowujemy specjalny klucz, który składa się z 96 bitów zawierających zera a pozostałe 32 bitów jest losowa wartością. Teraz szyfrujemy pewien ustalony (niezmienny) tekst (np. „message") z zastosowaniem naszego spreparowanego klucza. Wynik nazywany jest łamigłówką, ponieważ tak naprawdę nie jest tak bardzo trudno znaleźć klucz P. Wystarczy sprawdzić maksymalnie 2^{32} wartości.

Łamigłówka Merkle'a - protokół

Alicja: przygotowuje 2^{32} łamigłówek

- Dla i=1, …, 2^{32} wybierz losowy $\mathbf{P_i} \in \{0,1\}^{32}$ i $\mathbf{x_i}, \mathbf{k_i} \in \{0,1\}^{128}$
  ustaw łamigłówkę$_i$ ⟵ E(0^{96} ll $\mathbf{P_i}$, **"Puzzle # x_i" ll k_i**)
- Wysyła łamigłówkę$_1$, … , łamigłówkę$_{2^{32}}$ do Bob'a

Bob: wybiera losową łamigłówkę$_j$ i rozwiązuje ją. Otrzymuje (x_j, k_j) .

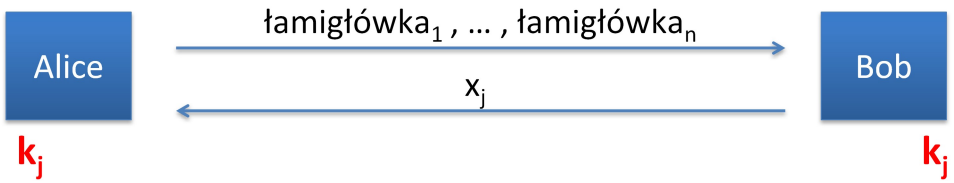
- Wysyła x_j do Alice

Alice: poszukuje łamigłówki numer x_j . Stosuje k_j jako współdzielony klucz szyfrowania symetrycznego

10

Wymiana klucza z zastosowaniem łamigłówki wygląda w następujący sposób. Alicja losuje 2^{32} łamigłówek. W czasie każdego losowania wybiera P_i (32-btową końcówkę klucza 128-bitowego) oraz dwie liczby x_i i k_i, obie 128-bitowe. Następnie szyfruje za pomocą „ograniczonego" klucza wyraz „Puzzle # x_i" połączony z k_i. Wszystkie 2^{32} łamigłówek jest przesłanych do Bob'a. Bob wybiera losową łamigłówkę i ją rozwiązuje. Odsyła do Alice wartość x_j. Jest to numer jednej z wysłanych łamigłówek. Alice odnajduje k_j powiązany z tym numerem i ustala go jako klucz sesji.

Schematycznie



Praca Alice: O(n) (przygotowanie n łamigłówek)

Praca Bob'a: O(n) (rozwiązanie jednej łamigłówki)

Praca podsłuchiwacza: $O(n^2)$ (np. 2^{64} czasu)

11

Spoglądając na protokół z innej strony, Alice wysyła do Bob'a n łamigłówek. Bob rozwiązuje jedną i odsyła wartość x_j. Przygotowanie i wysłanie n łamigłówek zajmuje n jednostek czasu, rozwiązanie jednej z łamigłówek również zajmuje n jednostek czasu. Alice na podstawie wartości x_j odnajduje „uzgodniony" klucz szyfrowania symetrycznego. Podsłuchiwacz widzi n łamigłówek i jedną wartość x_j. Nie wie, która z łamigłówek zawierała x_j. Żeby złamać protokół musi on rozwiązać wszystkie łamigłówki (w najgorszym wypadku), żeby znaleźć klucz k_j. Złożoność obliczeniowa takiego zadania jest n^2, jeśli było 2^{32} łamigłówek, to trzeba wykonać 2^{64} obliczeń.
Widać tutaj pewne problemy z praktyczną realizacją tego protokołu. Zarówno Alice, jak i Bob muszą wykonać wiele obliczeń. Ponadto Alice musi wysłać bardzo wiele danych do Boba (16 do 32 GB w zależności od wielkości każdej z łamigłówki). Po takim wysiłku atakującemu wystarczy wykonać 2^{64} obliczeń, co obecnie nie jest uznawane za specjalnie bezpieczne. W rezultacie, jeśli atakujący będzie naprawdę chciał złamać ten protokół, to go złamie. Żeby naprawdę zabezpieczyć ten protokół, trzeba zwiększyć w nim parametr n, np. przesyłając 2^{64} łamigłówek do rozwiązania. Wtedy atakujący musiałby poświęcić czas 2^{128} do złamania protokołu, co już jest uznawane za bezpieczne. Ale trzeba zwrócić uwagę, że uczestnicy protokołu wymiany kluczy muszą spędzić 2^{64} czasu na swoje procedury, co staje się już nieakceptowalne… Dlatego ten protokół nie jest stosowany w praktyce. Warto jednak zwrócić uwagę na interesujący pomysł polegający na tym, że uczestnicy protokołu spędzają „tylko" liniowy w stosunku od ilości przesłanych łamigłówek czas obliczeń, podczas gdy atakujący musi spędzić na złamanie ten czas podniesiony do potęgi drugiej. Mówi się wtedy o „kwadratowej luce" pomiędzy szyfrującymi, a atakującymi.

Niemożliwość znalezienia wyniku

Czy możemy opracować konstrukcję z większą luką z zastosowaniem metod szyfrowania symetrycznego?

Odpowiedź: nie wiadomo…

Ale: z grubsza mówiąc,

kwadratowa luka jest uważana za najlepszą z możliwych, kiedy traktujemy szyfr jako czarną skrzynkę do kryptoanalizy [IR'89, BM'09]

12

Naturalnym pytaniem jest, czy możemy zbudować podobną konstrukcję, która będzie miała większą lukę z zastosowaniem szyfrów symetrycznych? Okazuje się, że na razie nie wiemy. Na razie uznajemy, że kwadratowa luka to jest coś najlepszego, co potrafimy osiągnąć, jeśli atakujący nie zna żadnych dodatkowych informacji na temat wymiany informacji (np. zastosowanych algorytmów szyfrowania i konstrukcji kryptograficznych) za wyjątkiem szyfrogramów i pewnych odpowiedzi. Wtedy najlepsze, co potrafimy zrobić, to wydłużyć czas łamania do wartości n^2.
Na końcu prezentacji znajdują się odwołania do literatury. Druga z wymienionych na tym slajdzie prac pokazuje, że kwadratowa luka to najwięcej, co możemy zaprojektować.

Powstaje teraz pytanie, co robić dalej. Wydaje się, że w jakimś sensie utknęliśmy. Dysponując metodami szyfrowania symetrycznego możemy osiągnąć co najwyżej kwadratowa lukę. Dalsze rozwiązania są możliwe w kryptografii z kluczem jawnym. Będziemy potrzebować czegoś więcej niż dotychczasowe funkcje szyfrujące i obliczające skróty (HASH). Będziemy potrzebować funkcji o bardzo specjalnych właściwościach wywodzących się z algebry.

Wymiana kluczy bez zaufanej trzeciej instytucji będącej online (2)

Cel: Alice i Bob chcą współdzielonego klucza, ukrytego przed podsłuchaniem.

- Dotąd: zapewnialiśmy tylko bezpieczeństwo przeciw podsłuchiwaniu

Czy to jest wykonalne z **wykładniczą luką**?

13

Przypomnijmy, że rozważamy, czy możemy skonstruować protokół wymiany kluczy bez trzeciej instytucji zaufanej. Protokół ma zapewnić tylko ochronę przed podsłuchiwaniem. Alice i Bob, którzy nigdy się wcześniej nie spotkali na koniec protokołu otrzymują wspólny klucz do prowadzenia chronionej komunikacji. Tym razem będziemy się zastanawiać, czy nie można w jakiś sposób powiększyć luki (pomiędzy pracą konieczną do wykonania przez uczestników wymiany, a pracą konieczną do wykonania przez atakującego, żeby złamać protokół), np. do wartości wykładniczej.

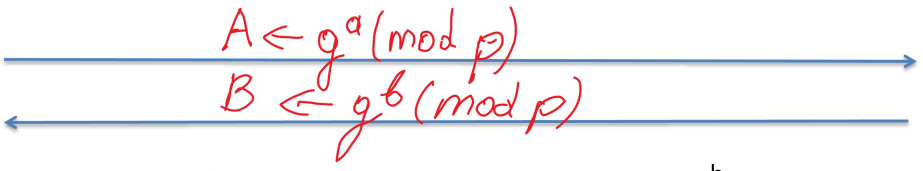
Protokół Diffie-Hellman'a (nieformalnie)

Ustal dużą liczbę pierwszą **p** (np. 600 cyfr)

Ustal wartość całkowitą **g** w przedziale {1, …, p}

Alice **Bob**

Wybierz losową wartość **a** z {1,…,p-1} Wybierz losową wartość **b** in {1,…,p-1}



$B^a \pmod p = (g^b)^a = k_{AB} = g^{ab} \pmod p = (g^a)^b = A^b \pmod p$

14

Lukę można powiększyć stosując protokół Diffie-Hellman'a, który teraz zostanie omówiony w sposób nieformalny. Wykonanie protokołu zaczyna się od wybrania pewnej dużej liczby pierwszej (np. składającej się z 600 cyfr w systemie dziesiętnym, około 2000 bitów w systemie dwójkowym). Jej zapisanie zajmuje około 2 Kb (kilo-bity). Nazwijmy ją **p**. Wybieramy również liczbę **g**, która jest liczbą całkowitą z przedziału **1** do **p**. te dwie liczby p i g są parametrami protokołu Diffie-Hellman'a. Są one wybrane raz i ustalane na zawsze.

Protokół działa w następujący sposób. Alicja wybiera losową liczbę całkowitą z zakresu 1 do p-1. Wtedy oblicza $g^a \pmod p$ (takie obliczenia są do wykonania w efektywny sposób). Przypisuje otrzymany wynik do zmiennej A. Wartość A jest wysyłana bo Bob'a. Bob robi to samo. Wybiera liczbę losową z przedziału 1 do p-1 i oblicza $g^b \pmod p$. Przypisuje otrzymany wynik do zmiennej B. Wartość B jest wysyłana do Alice. Teraz może zostać wygenerowany współdzielony tajny klucz…

Klucz k_{AB} jest obliczany jako $g^{ab} \pmod p$. Ciekawym spostrzeżeniem twórców protokołu było to, że każdy z uczestników był w stanie wyliczyć tę wartość g^{ab}. Alice mogła wyliczyć wartość $B^a \pmod p$, która po podstawieniu dawała $(g^b)^a$. Bob również mógł obliczyć te wartość, ponieważ $A^b \pmod p$ wynosi $(g^a)^b$.

Omówiony protokół został opublikowany w 1976 roku i stanowił przełom w kryptografii. Rozpoczął erę projektowania protokołów algebraicznych. Opiera się na własności potęgowania, która mówi, że $(g^b)^a$ jest równe $(g^a)^b$.

Bezpieczeństwo (pierwsze spostrzeżenia)

Podsłuchujący widzi: p, g, $A=g^a \pmod p$ i $B=g^b \pmod p$

Czy może wyliczyć $g^{ab} \pmod p$??

Bardziej ogólnie: zdefiniujmy $DH_g(g^a, g^b) = g^{ab} \pmod p$

Jak złożona obliczeniowo jest funkcja DH mod p?

15

Łatwo zauważyć, że Alice i Bob wymienili ze sobą klucz. Powstaje pytanie, czy odbyło się to w bezpieczny sposób. Czy podsłuchujący dysponując p, g, A i B może wyliczyć g^{ab}? Problem można uogólnić. Mamy pewną funkcję DH, zbudowaną na bazie g i pytamy się, czy dysponując g^a i g^b jesteśmy w stanie obliczyć g^{ab}? Pytanie sprowadza się jak złożona obliczeniowo (modulo p) jest funkcja, która na podstawie tych g^a i g^b wyliczy g^{ab}?

Jak złożona obliczeniowo jest funkcja DH mod p?

Zakładamy, że liczba p ma długość n bitów.

Najlepszy znany algorytm (GNFS): czas wykonywania exp($\tilde{O}(\sqrt[3]{n})$)

rozm. klucza szyfr.	rozmiar modułu	Rozmiar krzywych eliptycznych
80 bitów	1024 bity	160 bitów
128 bitów	3072 bitów	256 bitów
256 bitów (AES)	**15360** bitów	512 bitów

W rezultacie:

następuje wolne przejście od (mod p) do krzywych eliptycznych

16

Załóżmy, że p ma długość n-bitów (u nas ok. 2000 bitów). Okazuje się, że najlepszy algorytm wyszukujący rozwiązania funkcji DH działa w przybliżeniu w czasie proporcjonalnym do e do pierwiastka trzeciego stopnia z n (tak naprawdę to jest algorytm rozwiązujący problem dyskretnego logarytmu, ale przyjmujemy, że rozwiązuje również nasz problem; algorytm nazywa się „general numer field sieve" (GNFS) – ogólne sito ciała liczbowego). Złożoność nie jest wykładnicza, bo byłaby proporcjonalna do e do n, a jest do e do pierw. 3-go stopnia z n.
Spróbujmy rozważyć pewne wyniki na liczbach.
Jeśli rozmiar modułu (liczby p) wynosiłby 1024 bity, to czas obliczania funkcji wynosiłby ok. e^{10}, co jest małą liczbą. W przypadku algorytmu GNFS tak na prawdę ta wartość wynosi e^{80}, ponieważ na jego złożoność wpływają też inne czynniki, które tu zostały pominięte.
Tabelę należy rozumieć w następujący sposób. Podany jest rozmiar liczby p w bitach i w zależności od liczby tych bitów odniesiono czas rozwiązania funkcji DH do czasu złamania szyfru blokowego o określonej długości klucza. Czyli, jeśli długość liczby p będzie wynosić ok. 1000 bitów, to będzie można znaleźć rozwiązanie funkcji w czasie porównywalnym do złamania szyfru blokowego z kluczem 80 bitowym (czyli 2^{80}). Gdybyśmy zastosowali liczbę długości ok 3000 bitów, to czas rozwiązania funkcji byłby taki sam jak czas złamania AES z kluczem o 128 bitów. Aby osiągnąć czas obliczeń funkcji DH porównywalny do złamania AES 256 (2^{256} w idealnym przypadku), to długość liczby p musiałaby być bardzo duża.
Okazuje się więc, że wprowadzenie bezpieczeństwa na odpowiednim poziome wymaga obliczeń na bardzo dużych liczbach, które są wolne. Powstaje pytanie, czy można znaleźć jakiś lepsze rozwiązanie tego problemu. Okazuje się, że tak. W czasie omawiania protokołu Diffie-Hellman'a zastosowałem metodę zawartą w oryginalnym artykule z 1976 bazująca na arytmetycznym module z liczby pierwszej. Powoduje ona, że czas obliczeń funkcji Diffie-Hellman'a jest długi, ale równocześnie nie dający zbyt wielkiego bezpieczeństwa.
Przeprowadzenie protokołu Diffie-Hellman'a można przekształcić z obliczeń arytmetycznego modułu liczb pierwszych w inny obiekt algebraiczny, a tam obliczenie funkcji Diffie-Hellmana staje się dużo bardziej złożone algorytmicznie. Ten nowy model algebraiczny nazywa się krzywymi eliptycznymi. Ponieważ problem ma znacznie większą złożoność obliczeniową, to możemy prowadzić obliczenia na krótszych liczbach pierwszych. I tak stosując rozwiązanie funkcji Diffie-Hellmana w dziedzinie krzywych eliptycznych wystarczą długości liczb pierwszych dwukrotnie dłuższe od porównywalnych długości kluczy w szyfrach blokowych.
W nowych rozwiązaniach kryptograficznych następuje stopniowe przechodzenie z protokołu DH opartego na modułowi z liczb pierwszych na rzecz obliczeń w domenie krzywych eliptycznych.

Brak bezpieczeństwa na atak aktywny (man-in-the-middle)

Protokół nie jest bezpieczny na ataki **aktywne**

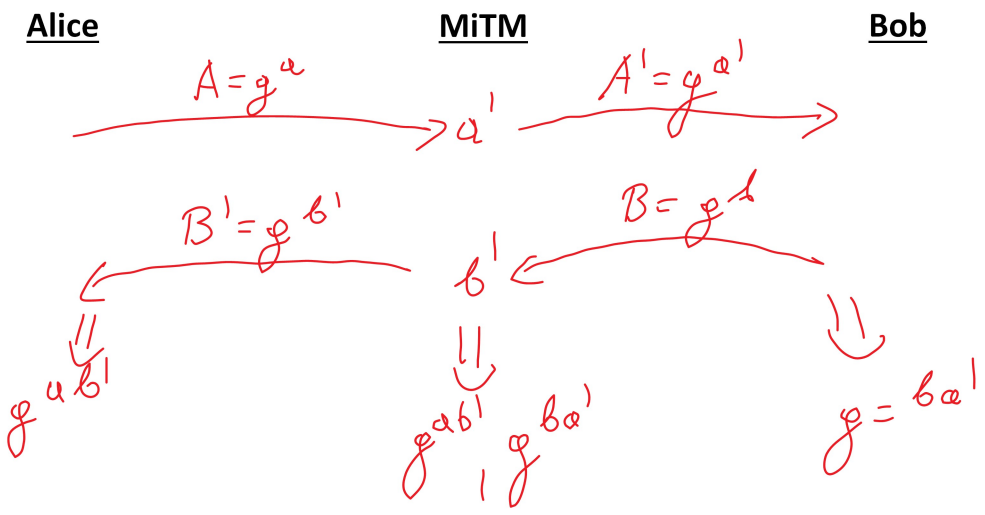
Alice **MiTM** **Bob**

17

Protokół Diffie-Hellmana nie jest bezpieczny na aktywny atak. Do zabezpieczenia tego protokołu będzie potrzebny pewien dodatek, który zostanie omówiony później.
Na dowód, rozważmy pewien atak. W systemie pojawia się „podsłuchiwacz" MiMT. Przechwytuje on wiadomość od Alice ($A=g^a$) i zamienia ją na swoją ($A'=g^{a'}$). Bob nie wie, że wiadomość została sfałszowana. Odsyła do nadawcy (Myśląc, że to Alice) wartość $B=g^b$. MiTM przechwytuje tę wartość i zamienia na inną: $B'=g^{b'}$. Podczas generowania kluczy Alice otrzymuje wartość gab', natomiast Bob gba'. To nie są te same klucze. Ale ponieważ MiTM zna wszystkie wartości a, b, a', b', to może sobie stworzyć klucze do komunikacji z Alice (gab') i Bobem (gba'). Wtedy, gdy Alice wysyła wiadomość do Boba, MiTM może ją odszyfrować, sfałszować i używając klucza komunikacji z Bobem wysyłać wiadomości w imieniu Alice. Alice i Bobowi wydaje się, że wymieniają bezpiecznie między sobą wiadomości, podczas gdy za każdym razem przechodzą one przez „pośrednika". Protokół nie jest więc bezpieczny na atak „men-in-the-middle".

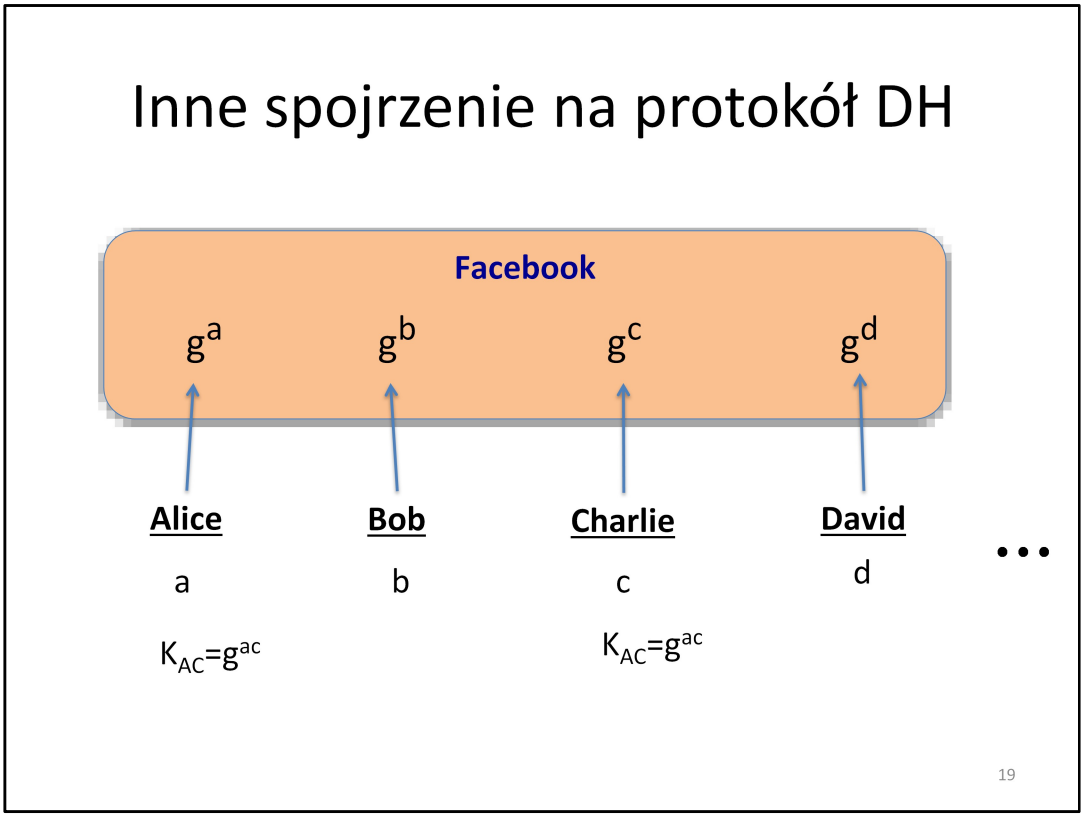
Brak bezpieczeństwa na atak aktywny (man-in-the-middle)

Protokół nie jest bezpieczny na ataki **aktywne**

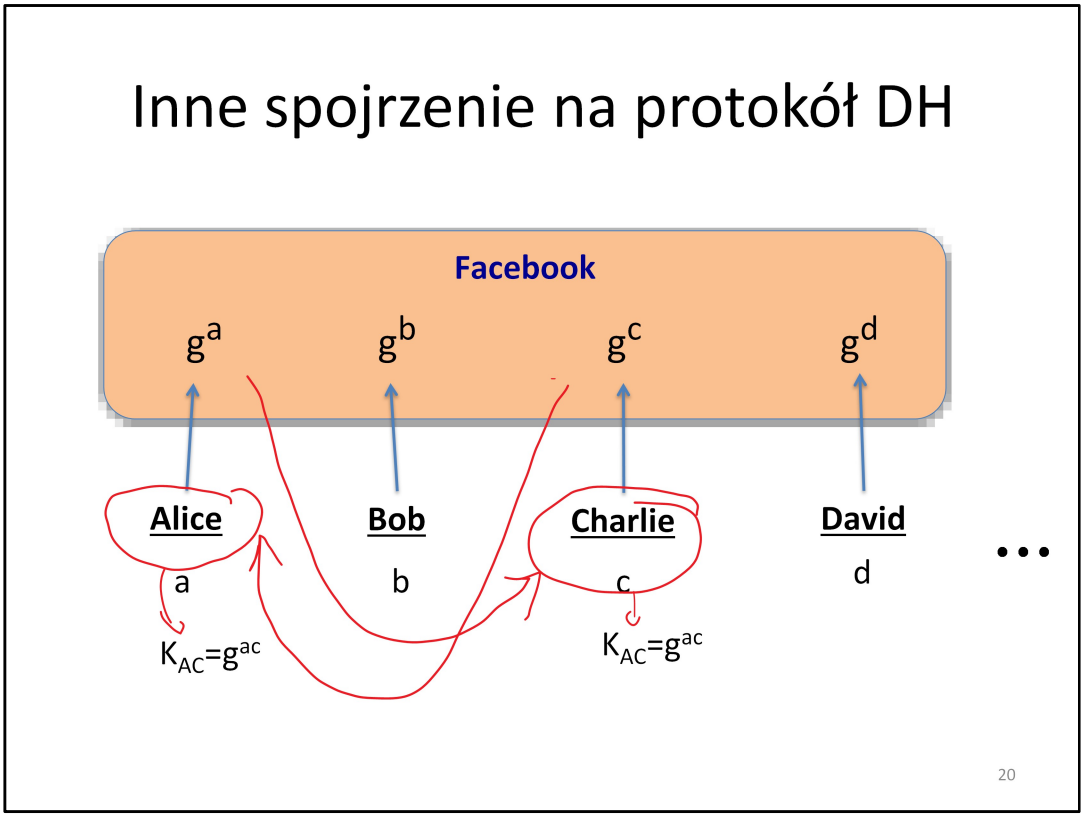


18

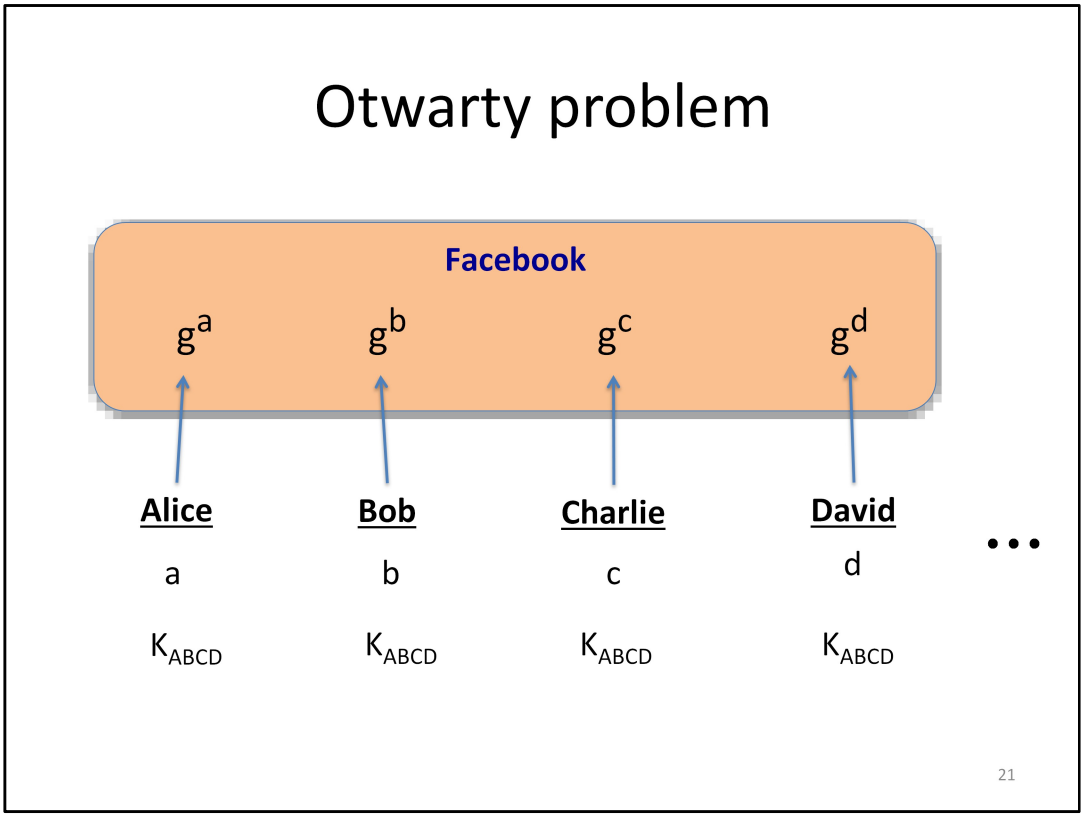
Protokół Diffie-Hellmana nie jest bezpieczny na aktywny atak. Do zabezpieczenia tego protokołu będzie potrzebny pewien dodatek, który zostanie omówiony później.
Na dowód, rozważmy pewien atak. W systemie pojawia się „podsłuchiwacz" MiMT. Przechwytuje on wiadomość od Alice ($A=g^a$) i zamienia ją na swoją ($A'=g^{a'}$). Bob nie wie, że wiadomość została sfałszowana. Odsyła do nadawcy (Myśląc, że to Alice) wartość $B=g^b$. MiTM przechwytuje tę wartość i zamienia na inną: $B'=g^{b'}$. Podczas generowania kluczy Alice otrzymuje wartość $g^{ab'}$, natomiast Bob $g^{ba'}$. To nie są te same klucze. Ale ponieważ MiTM zna wszystkie wartości a, b, a', b', to może sobie stworzyć klucze do komunikacji z Alice ($g^{ab'}$) i Bobem ($g^{ba'}$). Wtedy, gdy Alice wysyła wiadomość do Boba, MiTM może ją odszyfrować, sfałszować i używając klucza komunikacji z Bobem wysyłać wiadomości w imieniu Alice. Alice i Bobowi wydaje się, że wymieniają bezpiecznie między sobą wiadomości, podczas gdy za każdym razem przechodzą one przez „pośrednika". Protokół nie jest więc bezpieczny na atak „men-in-the-middle".



Okazuje się, że na protokół można spojrzeć z innej perspektywy. Wtedy okazuje się on pozbawiony interakcji pomiędzy osobami wymieniającym dane.
Załóżmy, że mamy bardzo wielu użytkowników (miliony). Każdy z nich wybiera jedną losową ukrytą wartość (a, b, c, d, …). Każda z nich zapisuje na swoim koncie Facebook zwój wkład w protokół DH, czyli wartości g^a, g^b, g^c, … Interesującą sprawą jest, że jeśli Alice i Charlie chcą wymieniać ze sobą informacje w sposób chroniony, to wcale nie muszą nawiązać bezpośredniej komunikacji. Wystarczy, że Alice odczyta dane z publicznego profilu Charlie'go, a Charlie odczyta dane z publicznego profilu Alice. Oboje mogą zestawić teraz klucz g^{ac}. Bez zawiązywania bezpośredniego „dialogu" otrzymują oni możliwość wygenerowania tajnego klucza. Ta właściwość jest czasami nazywana nie-interaktywną własnością protokołu Diffie-Hellman'a.



Okazuje się, że na protokół można spojrzeć z innej perspektywy. Wtedy okazuje się on pozbawiony interakcji pomiędzy osobami wymieniającym dane.
Załóżmy, że mamy bardzo wielu użytkowników (miliony). Każdy z nich wybiera jedną losową ukrytą wartość (a, b, c, d, …). Każda z nich zapisuje na swoim koncie Facebook zwój wkład w protokół DH, czyli wartości g^a, g^b, g^c, … Interesującą sprawą jest, że jeśli Alice i Charlie chcą wymieniać ze sobą informacje w sposób chroniony, to wcale nie muszą nawiązać bezpośredniej komunikacji. Wystarczy, że Alice odczyta dane z publicznego profilu Charlie'go, a Charlie odczyta dane z publicznego profilu Alice. Oboje mogą zestawić teraz klucz g^{ac}. Bez zawiązywania bezpośredniego „dialogu" otrzymują oni możliwość wygenerowania tajnego klucza. Ta właściwość jest czasami nazywana nie-interaktywną własnością protokołu Diffie-Hellman'a.



Z omówionym wcześniej zagadnieniem wiąże się problem, który nie został jeszcze rozwiązany… Czy można odczytać dane z wielu profili i od razu uzyskać „zbiorowy klucz" do komunikacji z wieloma użytkownikami? Zakładamy, że dany użytkownik pobiera dane z n profili (np. 4) i chce na podstawie tej informacji ustalić tajny klucz dający możliwość wymiany informacji ze wszystkimi w grupie.
Okazuje się, że jeśli w grupie jest 2 osoby, to mamy DH,
Jeśli w grupie jest 3 osoby, to też mamy rozwiązanie – protocol due to Joux (używa dość skompilowanych matematycznych przekształceń).
Natomiast jeśli w grupie jest 4 lub więcej osób, jeszcze nie udało się tego rozwiązać…

Ustalanie wspólnego sekretu

Cel: Alice i Bob chcą współdzielonego sekretu, ukrytego przed podsłuchaniem.

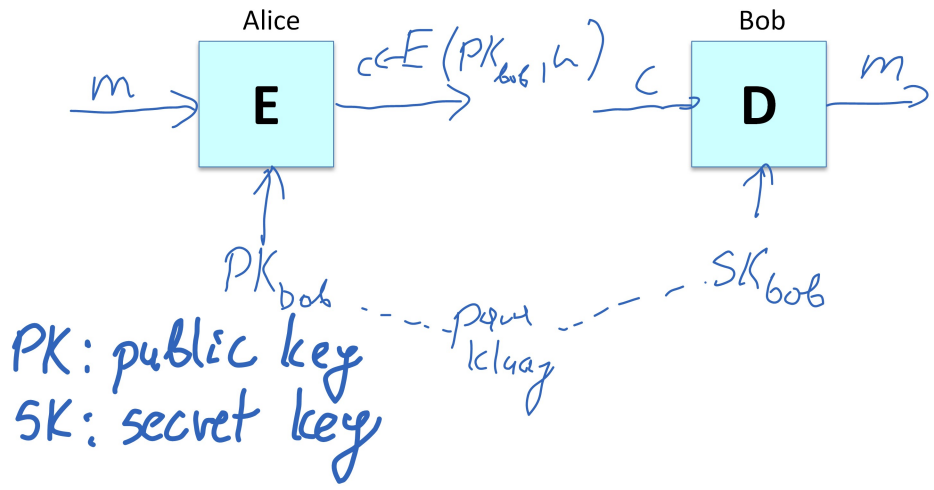
- Dotąd: zapewnialiśmy tylko bezpieczeństwo przeciw podsłuchiwaniu

Teraz: proponujemy nowe podejście

22

Alice i Bob , którzy nigdy się wcześniej nie spotkali ale na koniec protokołu otrzymują wspólny klucz do prowadzenia chronionej komunikacji. Istnieje możliwość podsłuchiwania komunikacji, ale nie jej modyfikacji (brak aktywnych ataków). Jak dotąd pokazaliśmy nieefektywny mechanizm opierający się na kryptografii klucza symetrycznego, a potem mechanizm wymiany kluczy Diffie-Hellman'a z uwzględnieniem wysiłku, jaki musi w złamanie protokołu włożyć atakujący. Warto nadmienić, że protokół DH jest powszechnie stosowany w Internecie. Teraz zajmiemy się innym podejściem bazującym na kryptografii klucz apublicznego.

Kryptografia klucza publicznego



23

Czym jest kryptografia klucza publicznego? Podobnie jak w przypadku szyfrowania symetrycznego dysponujemy dwoma algorytmami: szyfrującym (E) i deszyfrującym (D). Tym razem do szyfrowania stosujemy jeden klucz, nazywany kluczem publicznym. W naszym przypadku nazwijmy go PK_{bob}, ale do odszyfrowywania stosujemy inny klucz, nazywany kluczem tajnym (Sk_{bob}). Te dwa klucze tworzą parę kluczy. Szyfrowanie polega na uruchomieniu algorytmu szyfrującego wiadomość za pomocą klucza publicznego. Odszyfrowywanie polega na uruchomieniu algorytmu odszyfrowującego z zastosowaniem klucza tajnego.

Szyfrowanie z kluczem publicznym

Definicja: system szyfrowania z kluczem publicznym to trzy algorytmy (G, E, D)

- G(): alg. losowy generujący parę kluczy (pk, sk)
- E(pk, m): alg. losowy biorący m∈M i zwaracający c ∈C
- D(sk,c): alg. deterministyczny biorący c∈C i zwracający m∈M lub ⊥

Spójność: ∀(pk, sk) zwracanych przez G :

$$\forall m \in M: \quad D(sk, E(pk, m)) = m$$

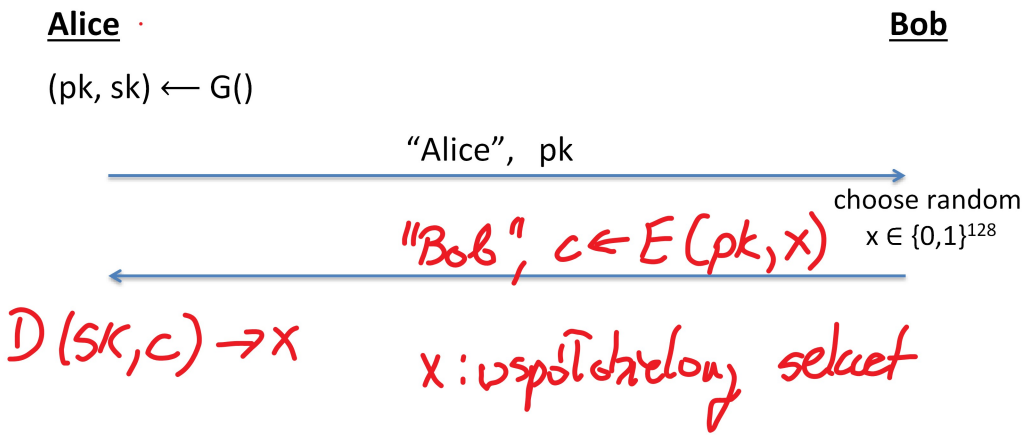
Semantyczne bezpieczeństwo:

System szyfrowania z kluczem publicznym jest bezpieczny semantycznie.

24

Bardziej formalnie, Szyfrowanie z kluczem publicznym można zdefiniować, jako złożenie trzech algorytmów. Algorytmu G, który służy do generacji kluczy. Po jego działaniu otrzymujemy parę kluczy: publiczny i tajny. Algorytm E, to algorytm szyfrujący, który z zastosowaniem klucza publicznego szyfruje wiadomości. Algorytm D to algorytm odszyfrowujący, który odszyfrowuje szyfrogram z zastosowaniem klucza tajnego, lub zwraca „bottom” (znacznik, że odszyfrowanie się nie udało). Podobnie, jak to miało miejsce w szyfrowaniu symetrycznym algorytmy szyfrowania i deszyfrowania muszą spełnić własność spójności. Dla każdej pary kluczy pk i sk wygenerowanej przez algorytm G wiadomość zaszyfrowana z zastosowaniem klucza publicznego da się odszyfrować z zastosowaniem klucza tajnego.

Ustalanie współdzielonego sekretu



- Semantyczne bezpieczeństwo dla takiego jest udowodnione.
- Protokół jest podatny na atak man-in-the-middle.

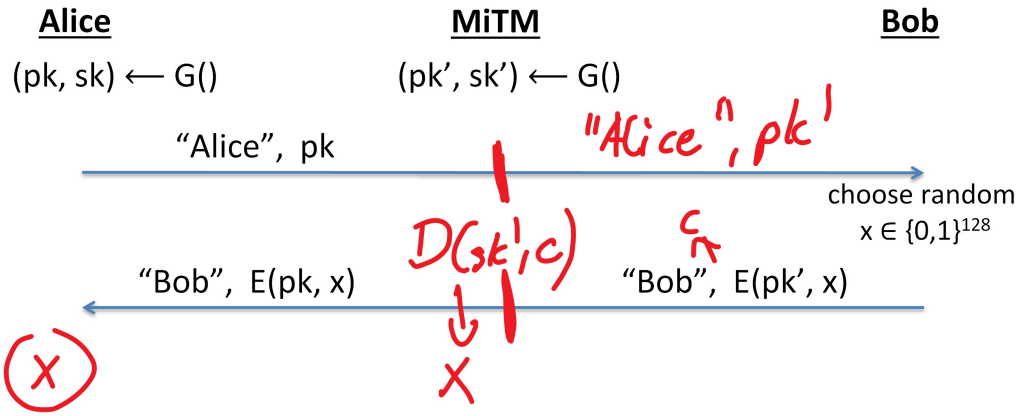
25

Jak zastosować kryptografię z kluczem publicznym do wymiany kluczy (tajnych informacji?).
Alice generuje parę kluczy (pk i sk). Następnie odsyła pk do Bob'a z informacją, że klucz pochodzi od Alice. Bob generuje losową wartość x (klucz?) o długości 128 bitów, szyfruję ją z zastosowaniem klucza publicznego Alice i dołącza informację, że odsyła zaszyfrowaną wiadomość od siebie (od Boba). Alice otrzymuje szyfrogram, odszyfrowuje go i otrzymuje wartość x. x jest współdzieloną tajną wartością pomiędzy Alice i Bob.

W tym podejściu jest różnica w odniesieniu do protokołu Diffie-Hellman'a. Tutaj musi zajść sekwencja zdarzeń. Bob nie może odesłać sekretu, jeśli nie otrzyma wcześniej wiadomości od Alice. W protokole Diffie-Hellman'a kolejność wysyłania danych nie maiła znaczenia, dodatkowo protokół ten pozwalał ustalić chronione połączenie na podstawie danych umieszczonych np. w profilu na Facebook'u. Zastosowanie kryptografii z kluczem publicznym, do rozpoczęcia połączenia wymaga jeszcze „odesłania" informacji do osoby publikującej klucz publiczny…

Podatność na atak man-in-the-middle

As described, the protocol is insecure against **active** attacks



26

Jak już wspominałem, omówiony protokół nie jest odporny na atak z aktywnym przeciwnikiem. Kiedy Alice generuje swoje klucze, to własne klucze generuje również MiTM. Atakujący przechwytuje wiadomość Alice, a następnie przesyła jej „podrobioną" wersję z własnym kluczem pk'. Bob'owi wydaje się, że otrzymał wiadomość od Alice. Odsyła do nadawcy wiadomości zaszyfrowaną przy pomocy pk' losową wartość x. Ta wiadomość jest przechwytywana przez MiTM i odszyfrowywana. MiTM podszywa się teraz pod Bob'a i odsyła do Alice wartość x zaszyfrowaną jej kluczem publicznym. Alice i Bob'owi wydaje się, że x jest tajna i współdzielona tylko między nimi, podczas gdy jest ona również w posiadaniu MiTM. Protokół nie jest więc bezpieczny, jeśli dopuścimy do możliwości fałszowania wiadomości. Taki protokół można zabezpieczyć, ale zastosowaniem podpisu elektronicznego i będzie to pokazane w dalszej części wykładu.

Konstrukcje stosowane w szyfrowaniu z kluczem publicznym

Konstrukcje w kryptografii z kluczem publicznym oparte są na problemach trudnych pochodzących z teorii liczb lub algebry. Oznacza to, że do rozwiązania problemu potrzeba bardzo dużo czasu lub zasobów i w praktyce na razie są nierozwiązywalne

Następny wykład:

- Przegląd wybranych zagadnień z teorii liczb będących podstawą tworzenia systemów kryptografii z kluczem publicznym.

27

Jak dotąd wyjaśniliśmy, jak na wysokim poziomie działa kryptografia z kluczem publicznym, ale nie pokazaliśmy jak takie systemy się konstruuje. Okazuje się, że te systemy bazują na teorii liczb i algebrze, podobnie jak protokół Diffie-Hellman'a.

Literatura uzupełniająca

- Merkle Puzzles are Optimal,
  B. Barak, M. Mahmoody-Ghidary, Crypto ’09
- On formal models of key exchange (sections 7-9)
  V. Shoup, 1999

28

Jeśli chodzi o wybrane prace uzupełniające materiały pokazane na wykładzie, to pierwsza odnosi się do problemu „kwadratowej luki”, jeśli w systemie wymiany kluczy stosujemy tylko łamigłówki Merkle’a i system szyfrujący widziany w postaci czarnej skrzynki (nie znamy zastosowanej konstrukcji kryptograficznej i kluczy).

Druga praca jest przeglądem protokołów wymiany kluczy i metod ich zabezpieczania przed atakiem men-in-the-middle.