Sieci komputerowe

Wykład 11: Podstawy kryptografii

Marcin Bieńkowski

Instytut Informatyki
Uniwersytet Wrocławski

Spis treści

Szyfrowanie

Szyfrowanie symetryczne

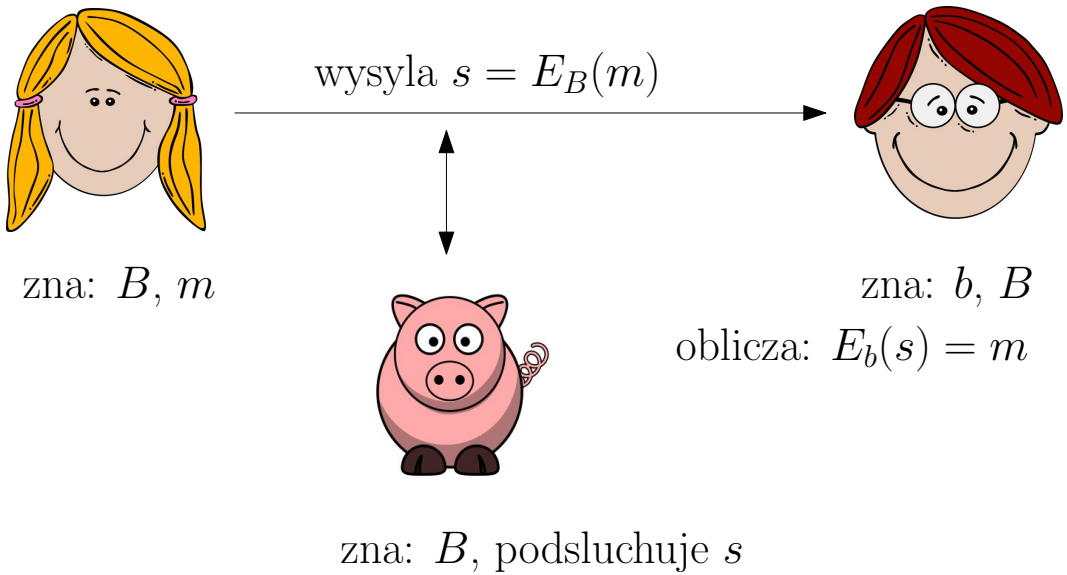
Główny problem: jak ustalić wspólny klucz K?

Rozwiązanie 1: przesłać innym, *zabezpieczonym* kanałem (zazwyczaj niepraktyczne / drogie)

Rozwiązanie 2: zastosować szyfrowanie asymetryczne (do przesyłania klucza lub całej wiadomości)

Szyfrowanie asymetryczne, założenia

- Bob ma dwa klucze
  - *klucz* B (na stronie WWW)
  - *klucz* b (w sejfie)
- Istnieje algorytm szyfrujący E, taki że dla dowolnej wiadomości m zachodzi $E_b(E_B(m)) = m$.



- b i m są trudno obliczalne na podstawie $s = E_B(m)$ i B!

Efekt

Każdy może wysłać wiadomość do Boba,
ale odczytać może ją tylko Bob.

Czy takie szyfrowanie jest w ogóle możliwe do zrealizowania?

- Tak! Idea: pewne *odwracalne* operacje są łatwiejsze do wykonania niż ich odwrotności (np. mnożenie liczb pierwszych kontra rozkład na czynniki pierwsze).
- Przykładowy algorytm: RSA $\rightarrow$ notatki.

Efekt

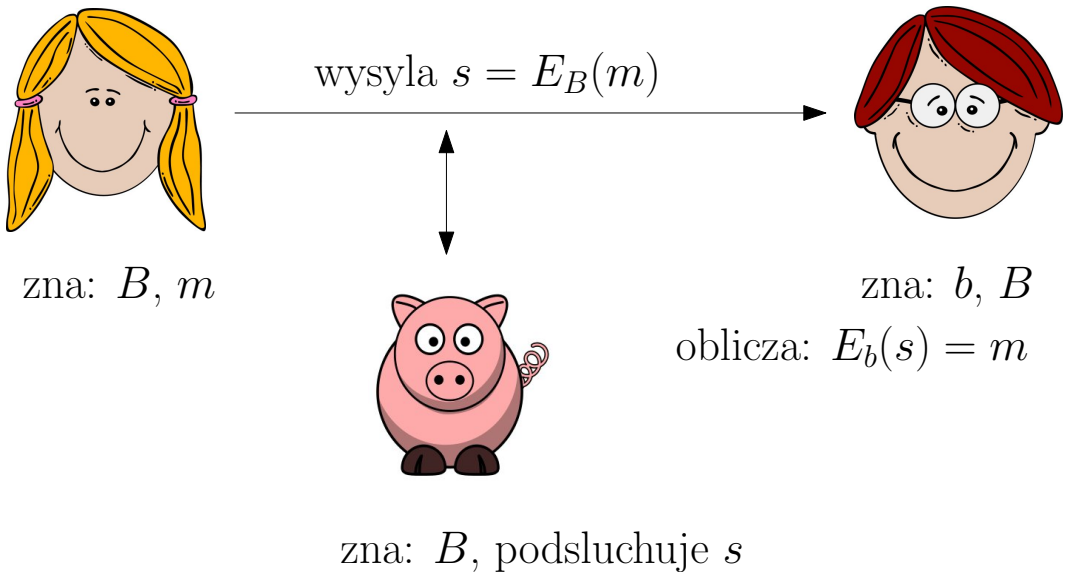
Każdy może wysłać wiadomość do Boba,
ale odczytać może ją tylko Bob.

Czy takie szyfrowanie jest w ogóle możliwe do zrealizowania?

- Tak! Idea: pewne *odwracalne* operacje są łatwiejsze do wykonania niż ich odwrotności (np. mnożenie liczb pierwszych kontra rozkład na czynniki pierwsze).
- Przykładowy algorytm: RSA $\rightarrow$ notatki.

Szyfrowanie asymetryczne

- Bob ma dwa klucze
  - *klucz* B (na stronie WWW)
  - *klucz* b (w sejfie)
- Istnieje algorytm szyfrujący E, taki że dla dowolnej wiadomości m zachodzi $E_b(E_B(m)) = m$.



- b i m są trudno obliczalne na podstawie $s = E_B(m)$ i B!

Uwierzytelnianie

Normalny scenariusz

Założenia

- Alicja zna klucz B (publiczny Boba).
- Alicja wysyła wiadomość do Boba zaszyfrowaną B.

Co wiedzą poszczególne osoby?

- Bob nie musi się uwierzytelniać.
  - Wiadomość jest zaszyfrowana kluczem B, więc przeczytać może tylko posiadacz b, czyli Bob.
- Ale Bob nie wie, kto wysłał wiadomość!

Normalny scenariusz

Założenia

- Alicja zna klucz B (publiczny Boba).
- Alicja wysyła wiadomość do Boba zaszyfrowaną B.

Co wiedzą poszczególne osoby?

- Bob nie musi się uwierzytelniać.
  - Wiadomość jest zaszyfrowana kluczem B, więc przeczytać może tylko posiadacz b, czyli Bob.
- Ale Bob nie wie, kto wysłał wiadomość!

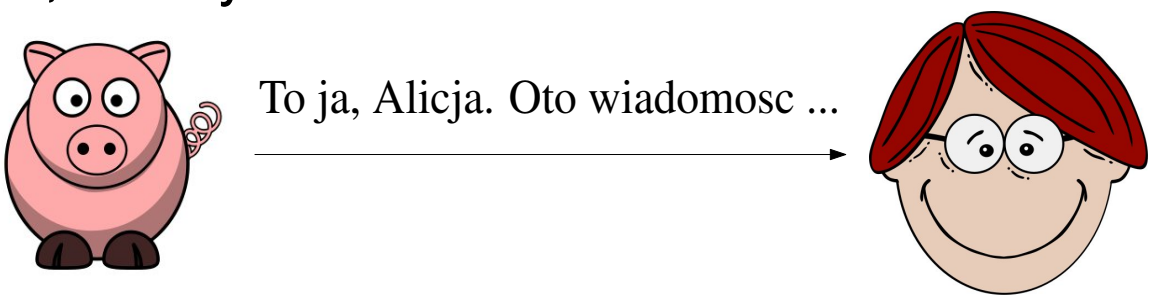
Normalny scenariusz

Założenia

- Alicja zna klucz B (publiczny Boba).
- Alicja wysyła wiadomość do Boba zaszyfrowaną B.

Co wiedzą poszczególne osoby?

- Bob nie musi się uwierzytelniać.
  - Wiadomość jest zaszyfrowana kluczem B, więc przeczytać może tylko posiadacz b, czyli Bob.
- Ale Bob nie wie, kto wysłał wiadomość!



Normalny scenariusz

Założenia

- Alicja zna klucz B (publiczny Boba).
- Alicja wysyła wiadomość do Boba zaszyfrowaną B.

Co wiedzą poszczególne osoby?

- Bob nie musi się uwierzytelniać.
  - Wiadomość jest zaszyfrowana kluczem B, więc przeczytać może tylko posiadacz b, czyli Bob.
- Ale Bob nie wie, kto wysłał wiadomość!

Uwaga na marginesie: w przypadku szyfrowania symetrycznego nie mamy tego problemu, bo Alicja udowadnia swoją tożsamość znajomością klucza (który jest znany tylko Alicji i Bobowi).

Wróćmy do algorytmu RSA

- Klucz prywatny B i publiczny b można zamienić miejscami, tj.

$$E_b(E_B(m)) = m \ ,$$

ale również

$$E_B(E_b(m)) = m \ .$$

- $E_b(m)$ nazywamy podpisem cyfrowym tekstu m.
  - To nie do końca prawda.
- Zweryfikować taki podpis może każdy, wystarczy znać klucz publiczny B.
- Tylko posiadacz klucza prywatnego b może tak podpisać m.
- Jak wykorzystać to w uwierzytelnianiu?

Wróćmy do algorytmu RSA

- Klucz prywatny B i publiczny b można zamienić miejscami, tj.

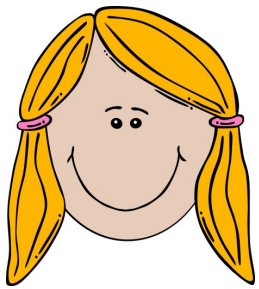
$$E_b(E_B(m)) = m \ ,$$

ale również

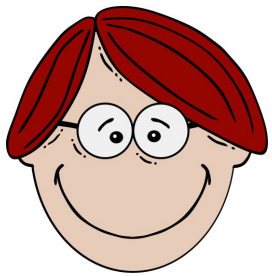
$$E_B(E_b(m)) = m \ .$$

- $E_b(m)$ nazywamy podpisem cyfrowym tekstu m.
  - To nie do końca prawda.
- Zweryfikować taki podpis może każdy, wystarczy znać klucz publiczny B.
- Tylko posiadacz klucza prywatnego b może tak podpisać m.
- Jak wykorzystać to w uwierzytelnianiu?

Złe rozwiązanie



To ja, Alicja.
Wysylam X oraz $Y = E_a(X)$



klucz prywatny: a

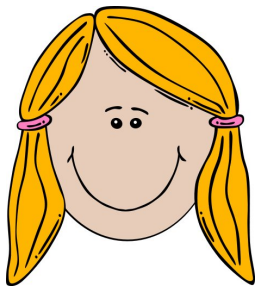
klucz publiczny: A

zna: A

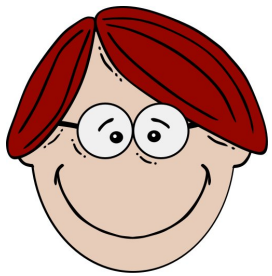
sprawdza, czy $E_A(Y) = X$

Problem: Atak powtórzeniowy: świnia może nagrać tę transmisję i odtworzyć później w komunikacji z Bobem.

Złe rozwiązanie



To ja, Alicja.
Wysylam X oraz $Y = E_a(X)$



klucz prywatny: a

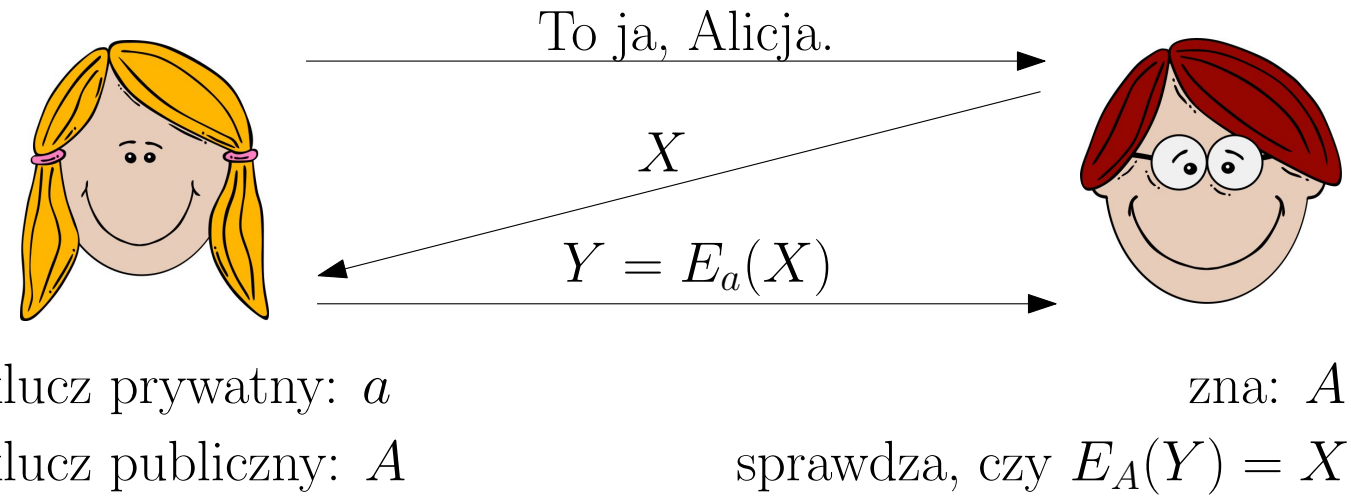
klucz publiczny: A

zna: A

sprawdza, czy $E_A(Y) = X$

Problem: Atak powtórzeniowy: świnia może nagrać tę transmisję i odtworzyć później w komunikacji z Bobem.

Uwierzytelnianie za pomocą podpisu cyfrowego



- Bob wybiera unikatowe, wcześniej niewykorzystywane X.
- Alicja udowadnia w ten sposób że jest posiadaczką klucza prywatnego pasującego do klucza publicznego Alicji.

Podsumowanie szyfrowania i uwierzytelniania

- Każda strona komunikacji generuje parę kluczy: publiczny i prywatny
- Alicja: (A, a), Bob: (B, b).
  - wiadomość: m
  - wiadomość + podpis kluczem prywatnym Alicji $m, E_a(m)$ (każdy może zweryfikować, że to Alicja napisała)
  - wiadomość zaszyfrowana kluczem publicznym Alicji $E_A(m)$ (przeczytać może tylko Alicja)
- Zastosowanie: PGP (*Pretty Good Privacy*).

Podpisy cyfrowe raz jeszcze

Rozwiązanie 1: Alicja wysyła parę $(m, E_a(m))$.

- Wada: zużywa 2 x więcej miejsca
- Wada: obliczanie $E_a(\cdot)$ długo trwa.

Rozwiązanie 2: Alicja wysyła parę $(m, E_a(h(m))$, gdzie h to kryptograficzna funkcja skrótu.

Podpisy cyfrowe raz jeszcze

Rozwiązanie 1: Alicja wysyła parę $(m, E_a(m))$.

- Wada: zużywa 2 x więcej miejsca
- Wada: obliczanie $E_a(\cdot)$ długo trwa.

Rozwiązanie 2: Alicja wysyła parę $(m, E_a(h(m))$, gdzie h to kryptograficzna funkcja skrótu.

Dystrybucja kluczy publicznych

Skąd wziąć czyjś klucz publiczny? (1)

Szyfrowanie i uwierzytelnianie zakłada, że nie tylko znamy czyjś klucz publiczny, ale wiemy też, że należy on do konkretnej osoby.

- Szyfrujemy kluczem publicznym B, odczytać może tylko posiadacz b, ale czy jest to Bob?
- Uwierzytelnianie: wiemy, że druga osoba posiada klucz prywatny a pasujący do klucza publicznego A, ale czy jest to Alicja?

Skąd wziąć czyjś klucz publiczny? (2)

Pomysł 1: Spotkanie fizyczne / telefoniczne / videokonferencja (czasem niepraktyczne).

- Poza tym wtedy można ustalić klucz symetryczny, po co nam kryptografia asymetryczna?

Pomysł 2: Klucz publiczny dostępny na stronie WWW.

- Bezpieczeństwo oparte na tym, że nikt go nie podmieni!
- Konieczna dodatkowa weryfikacja, np.:
  - Alicja umieszcza na stronie WWW klucz A.
  - Bob pobiera ze strony klucz A'.
  - Alicja i Bob weryfikują telefonicznie, czy $h(A) = h(A')$.

Skąd wziąć czyjś klucz publiczny? (3)

Poprzednie pomysły działają tylko przy komunikacji fizycznych osób.

Komunikacja z instytucją:

- Wchodzimy na stronę banku.
- Bank mówi „mój klucz publiczny = ..., szyfruj do mnie dane tym kluczem”.
- Skąd wiemy, że łączymy się faktycznie z bankiem?

Certyfikaty

Załóżmy, że Alicja:

- Ma klucz publiczny B i wie, że należy on do Boba.
- Wierzy w to, że Bob świadomie używa podpisów cyfrowych.
- Ma wiadomość „klucz publiczny Charliego to C” podpisaną kluczem b.
  - Ta wiadomość to certyfikat.

Na tej podstawie Alicja może zweryfikować, że wiadomość napisał Bob i uwierzyć w nią. $\rightarrow$ poznaje klucz publiczny Charliego.

Certyfikaty

Załóżmy, że Alicja:

- Ma klucz publiczny B i wie, że należy on do Boba.
- Wierzy w to, że Bob świadomie używa podpisów cyfrowych.
- Ma wiadomość „klucz publiczny Charliego to C” podpisaną kluczem b.
  - Ta wiadomość to certyfikat.

Na tej podstawie Alicja może zweryfikować, że wiadomość napisał Bob i uwierzyć w nią. $\rightarrow$ poznaje klucz publiczny Charliego.

Certyfikaty w PGP

Na stronie WWW Charlie może umieścić

- swój klucz publiczny C.
- certyfikat „klucz publiczny Charliego to C” podpisany kluczami różnych osób.

Uwagi:

- Umożliwia budowanie grafu certyfikacji.
  - Ścieżki certyfikacji mogą być długie.
- Podpisywanie kluczy publicznych: częste w środowisku programistów open source.
- PGP wykorzystywane do podpisywania oprogramowania.

A co z kluczami publicznymi instytucji?

Również certyfikaty:

- Ale generowane przez specjalne (zaufane) urzędy certyfikacji (CA).
- Można zgłosić się do CA, żeby dostać certyfikat (żeby CA podpisało nasz klucz publiczny).
  - CA powinno zweryfikowac, czy jesteśmy tym, za kogo się podajemy.

Ale skąd wziąć klucz publiczny CA?

- Mamy je wpisane w przeglądarkę WWW $\rightarrow$ prezentacja

SSL / TLS

SSL (Secure Sockets Layer), TLS (Transport Layer Security)

- Warstwa pośrednicząca pomiędzy warstwą transportową i warstwą aplikacji.
- Odpowiada za szyfrowanie i uwierzytelnianie.
- Większość popularnych usług ma swoje warianty wykorzystujące SSL działające na innym porcie (np. HTTPS = HTTP over SSL, port 443)

Łączenie z serwerem HTTPS

- Serwer WWW wysyła certyfikat (klucz publiczny + dane o stronie) podpisany przez pewne CA.
- Przeglądarka sprawdza, czy:
  - posiada klucz publiczny tego CA i sprawdza prawdziwość podpisu CA.
  - dane o stronie opisują tę stronę, z którą zamierzamy się łączyć.
- Mamy uwierzytelniony serwer i możemy szyfrować wiadomości dla serwera WWW jego kluczem publicznym.
- Zazwyczaj nie uwierzytelnia się użytkownika, choć jest to teoretycznie możliwe.

Rodzaje certyfikatów

Zwykłe: zaświadczenie, że łączymy się z konkretną stroną `abc.org`.

- To nie znaczy, że łączymy się ze stroną należącą do instytucji ABC!

Rozszerzone: zaświadczenie, że łączymy się ze stroną danej instytucji.

Wadliwe: najczęściej klucz publiczny podpisany nim samym.

$\rightarrow$ prezentacja

Klucze sesji

Powiedzieliśmy: „mamy uwierzytelniony serwer i możemy szyfrować wiadomości dla serwera WWW”

Problemy:

- Serwer musi też jakoś szyfrować dane do nas.
  - Moglibyśmy mu teraz wysłać swój klucz publiczny...
- Szyfrowanie asymetryczne jest nieefektywne (RSA jest ok. 1000 razy wolniejszy niż AES)

Rozwiązanie:

- Przeglądarka generuje symetryczny klucz sesji.
- Przeglądarka szyfruje go kluczem publicznym serwera WWW i wysyła do serwera WWW.
- Dalsza komunikacja jest szyfrowana kluczem sesji.

Uwagi końcowe

Dwie historie na zakończenie

Kryptografia $\neq$ bezpieczeństwo

- Kryptografia dostarcza *matematycznych metod*.
- Te metody mogą być wadliwe same w sobie.
- Ale zazwyczaj po prostu są źle wykorzystywane.
- Trudność: nie umiemy zdefiniować matematycznie, co to jest bezpieczeństwo.

Historia nr 1 (1)

Alicja pisze list rekomendacyjny

- Alicja jest bardzo zajęta (jest profesorem).
- Alicja chce rekomendować na stanowisko osobę X.
- Alicja zleca napisanie listu Bobowi, dając mu wytyczne co ma w nim być.
- Po tym jak Bob napisze rekomendację m, Alicja ją przeczyta, obliczy $E_a(h(m))$ i Bob wyśle $(m, E_a(h(m))$ do pracodawcy.
- Zakładamy, że funkcja h generuje 80-bitowy skrót.

Gdzie tu jest problem?

- Ustalmy parę $(m, E_a(h(m))$. Jeśli chcemy znaleźć m' takie, że $E_a(h(m)) = E_a(h(m'))$ to musimy sprawdzić średnio $2^{80}/2$ wiadomości.
- A co może zrobić Bob jeśli nie lubi osoby X?

Historia nr 1 (1)

Alicja pisze list rekomendacyjny

- Alicja jest bardzo zajęta (jest profesorem).
- Alicja chce rekomendować na stanowisko osobę X.
- Alicja zleca napisanie listu Bobowi, dając mu wytyczne co ma w nim być.
- Po tym jak Bob napisze rekomendację m, Alicja ją przeczyta, obliczy $E_a(h(m))$ i Bob wyśle $(m, E_a(h(m))$ do pracodawcy.
- Zakładamy, że funkcja h generuje 80-bitowy skrót.

Gdzie tu jest problem?

- Ustalmy parę $(m, E_a(h(m))$. Jeśli chcemy znaleźć m' takie, że $E_a(h(m)) = E_a(h(m'))$ to musimy sprawdzić średnio $2^{80}/2$ wiadomości.
- A co może zrobić Bob jeśli nie lubi osoby X?

Historia nr 1 (1)

Alicja pisze list rekomendacyjny

- Alicja jest bardzo zajęta (jest profesorem).
- Alicja chce rekomendować na stanowisko osobę X.
- Alicja zleca napisanie listu Bobowi, dając mu wytyczne co ma w nim być.
- Po tym jak Bob napisze rekomendację m, Alicja ją przeczyta, obliczy $E_a(h(m))$ i Bob wyśle $(m, E_a(h(m))$ do pracodawcy.
- Zakładamy, że funkcja h generuje 80-bitowy skrót.

Gdzie tu jest problem?

- Ustalmy parę $(m, E_a(h(m))$. Jeśli chcemy znaleźć m' takie, że $E_a(h(m)) = E_a(h(m'))$ to musimy sprawdzić średnio $2^{80}/2$ wiadomości.
- A co może zrobić Bob jeśli nie lubi osoby X?

Historia nr 1 (2)

- Alicja chce rekomendować na stanowisko osobę X.
- Alicja zleca napisanie listu Bobowi, dając mu wytyczne co ma w nim być.
- Po tym jak Bob napisze rekomendację m, Alicja ją przeczyta, obliczy $E_a(h(m))$ i Bob wyśle $(m, E_a(h(m))$ do pracodawcy.
- Zakładamy, że funkcja h generuje 80-bitowy skrót.

Gdzie tu jest problem?

- Bob generuje 2^{40} listów rekomendacyjnych polecających X (zbiór M_X) i 2^{40} listów rekomendacyjnych polecających Y (zbiór M_Y)
  - listy różnią się spacjami, przecinkami, drobnymi słowami, etc.
- h zachowuje się jak funkcja zwracająca losowy ciąg 80-bitowy.
- Ze stałym prawdopodobieństwem, istnieją $m_x \in M_X$ i $m_y \in M_Y$, takie że $h(m_x) = h(m_y)$! ($\rightarrow$ ćwiczenie (paradoks urodzin))
- Bob pokazuje Alicji m_x, Alicja oblicza $E_a(h(m_x))$, Bob wysyła poprawnie podpisany list $(m_y, E_a(h(m_x))$.

Historia nr 1 (2)

- Alicja chce rekomendować na stanowisko osobę X.
- Alicja zleca napisanie listu Bobowi, dając mu wytyczne co ma w nim być.
- Po tym jak Bob napisze rekomendację m, Alicja ją przeczyta, obliczy $E_a(h(m))$ i Bob wyśle $(m, E_a(h(m))$ do pracodawcy.
- Zakładamy, że funkcja h generuje 80-bitowy skrót.

Gdzie tu jest problem?

- Bob generuje 2^{40} listów rekomendacyjnych polecających X (zbiór M_X) i 2^{40} listów rekomendacyjnych polecających Y (zbiór M_Y)
  - listy różnią się spacjami, przecinkami, drobnymi słowami, etc.
- h zachowuje się jak funkcja zwracająca losowy ciąg 80-bitowy.
- Ze stałym prawdopodobieństwem, istnieją $m_x \in M_X$ i $m_y \in M_Y$, takie że $h(m_x) = h(m_y)$! ($\rightarrow$ ćwiczenie (paradoks urodzin))
- Bob pokazuje Alicji m_x, Alicja oblicza $E_a(h(m_x))$, Bob wysyła poprawnie podpisany list $(m_y, E_a(h(m_x))$.

Historia nr 1 (3)

Morał: dla funkcji skrótu ważne są dwie własności kryptograficzne

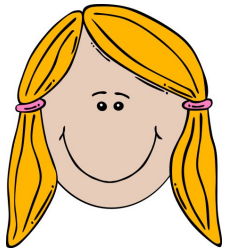
- Słaba własność: dla ustalonego x znalezienie y, takiego że $h(x) = h(y)$ jest obliczeniowo trudne.
- Silna własność: znalezienie x i y, takich że $h(x) = h(y)$ jest obliczeniowo trudne.
- MD5 nie ma silnej własności, SHA-2 jak dotąd ma.
- Rozwiązaniem jest zazwyczaj podwojenie długości skrótu zwracanego przez h.

Historia nr 2 (1)

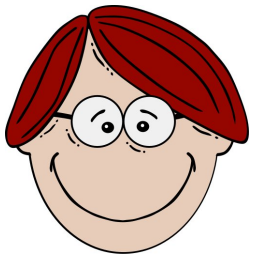
Alicja pisze wiadomość m. Chce ją podpisać i zaszyfrować do Boba. W jakiej kolejności powinna to robić?

Historia nr 2 (2)

Wariant 1: zaszyfruj, potem podpisz



$E_a(E_B(m))$



klucz publiczny: A

klucz prywatny: a

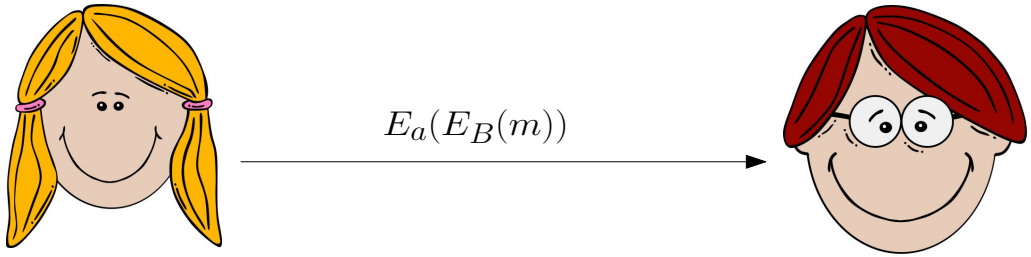
m = „moj genialny pomysl ...”

klucz publiczny: B

klucz prywatny: b

Historia nr 2 (2)

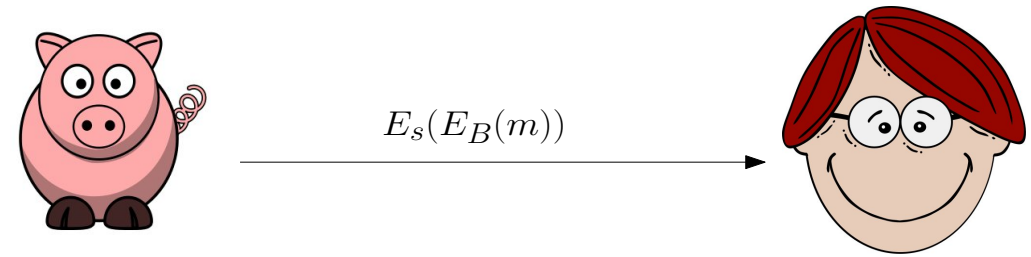
Wariant 1: zaszyfruj, potem podpisz



klucz publiczny: A
klucz prywatny: a
m = „moj genialny pomysl ...”

klucz publiczny: B
klucz prywatny: b

Oczywiście świnia przechwytuje wiadomość i wysyła własną.



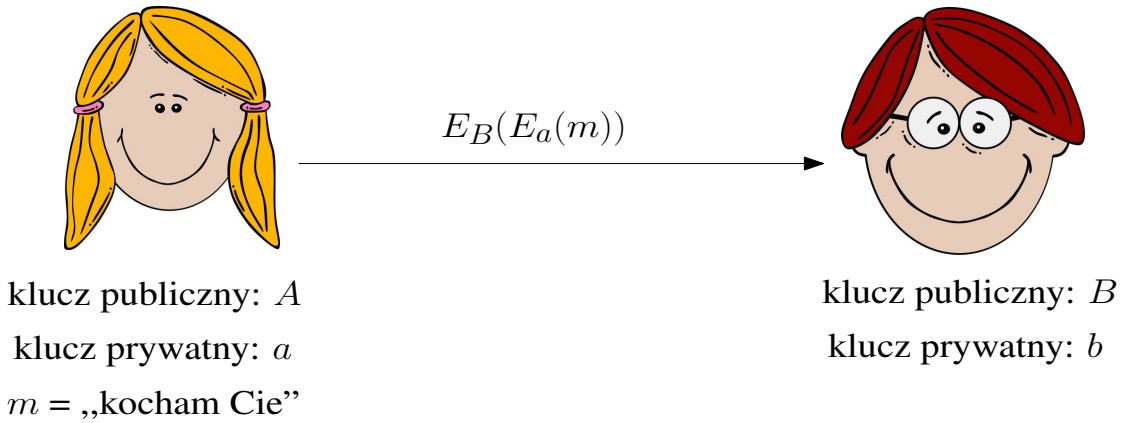
klucz publiczny: S
klucz prywatny: s

klucz publiczny: B
klucz prywatny: b

Bob myśli, że genialny pomysł miała świnia (świnia nawet nie wie jaki).

Historia nr 2 (3)

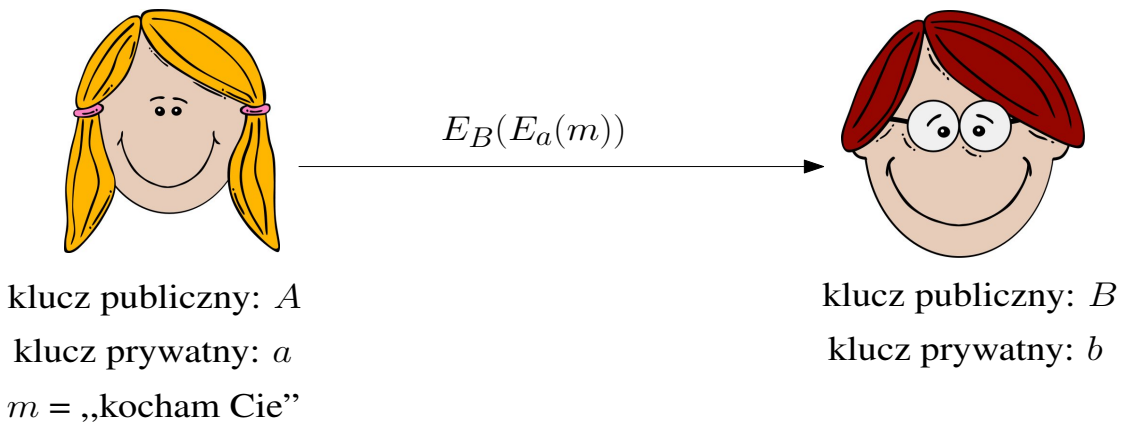
Wariant 2: podpisz, potem zaszyfruj



Tym razem Bob to świnia, wysyła $E_C(E_a(m))$ do swojego kumpla Charliego (dysponującego parą kluczy (C, c)).

Historia nr 2 (3)

Wariant 2: podpisz, potem zaszyfruj

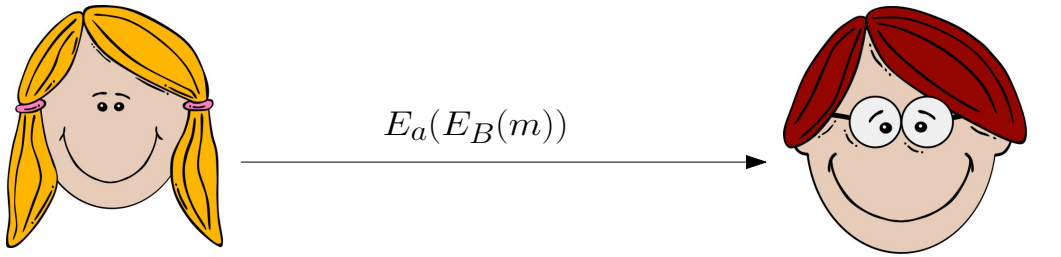


Tym razem Bob to świnia, wysyła $E_C(E_a(m))$ do swojego kumpla Charliego (dysponującego parą kluczy (C, c)).

Historia nr 2 (4)

Jeden ze sposobów naprawy:

- Wariant „szyfruj, potem podpisuj” jeszcze raz:



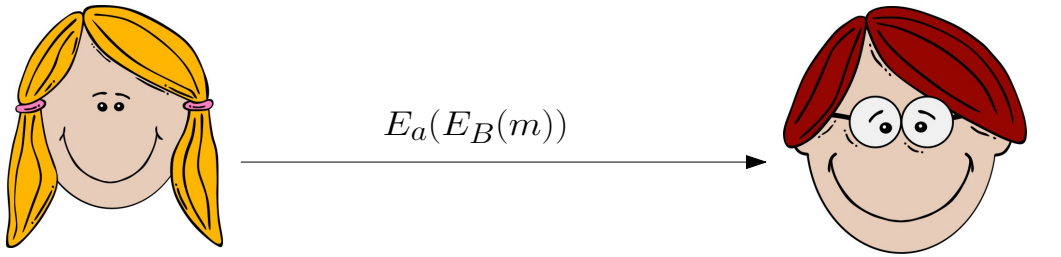
klucz publiczny: A
klucz prywatny: a
m = „moj genialny pomysl ...”

klucz publiczny: B
klucz prywatny: b

Historia nr 2 (4)

Jeden ze sposobów naprawy:

- Wariant „szyfruj, potem podpisuj” jeszcze raz:



klucz publiczny: A
klucz prywatny: a
m = „moj genialny pomysl ...”

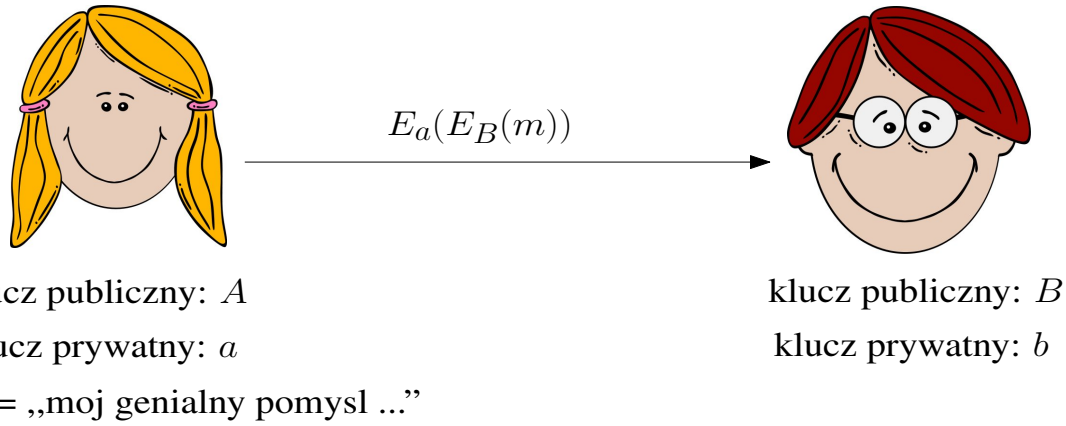
klucz publiczny: B
klucz prywatny: b

- Wystarczyłoby, żeby Alicja zmieniła wiadomość na: „To ja, Alicja. Mój genialny pomysł ...”.

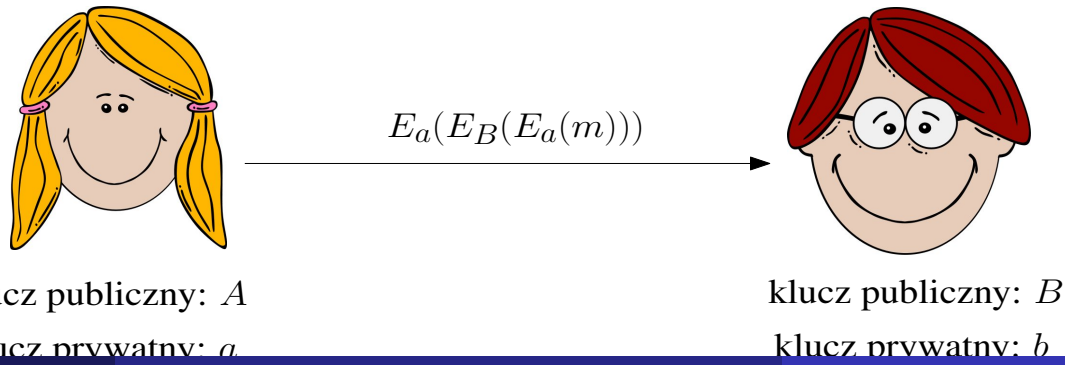
Historia nr 2 (4)

Jeden ze sposobów naprawy:

- Wariant „szyfruj, potem podpisuj” jeszcze raz:



- Wystarczyłoby, żeby Alicja zmieniła wiadomość na: „To ja, Alicja. Mój genialny pomysł ...”.
- Automatyzacja: podpisz, zaszyfruj, podpisz:



Lektura dodatkowa

- Kurose, Ross: rozdział 8
- Tanenbaum: rozdział 8