
Podstawy kryptografii

Sieci komputerowe

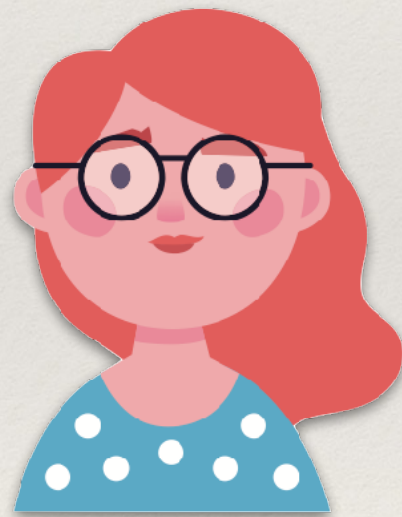
Wykład 11

Marcin Bieńkowski

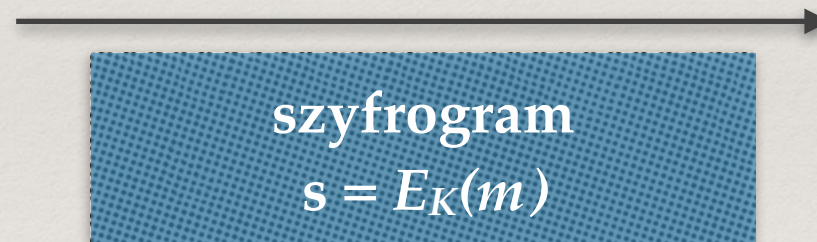
Szyfrowanie symetryczne

W poprzednim odcinku (1)

- ❖ **Szyfrowanie symetryczne** = ten sam klucz K do szyfrowania i deszyfrowania.
 - ✦ Znany obu stronom i nikomu więcej.
- ❖ Publiczny algorytm szyfrujący E i deszyfrujący D .
- ❖ $D_K(E_K(m)) = m$ dla każdego tekstu jawnego m i klucza K .



zna klucz K i tekst jawny m



zna K

oblicza $D_K(s) = D_K(E_K(m)) = m$

W poprzednim odcinku (2)

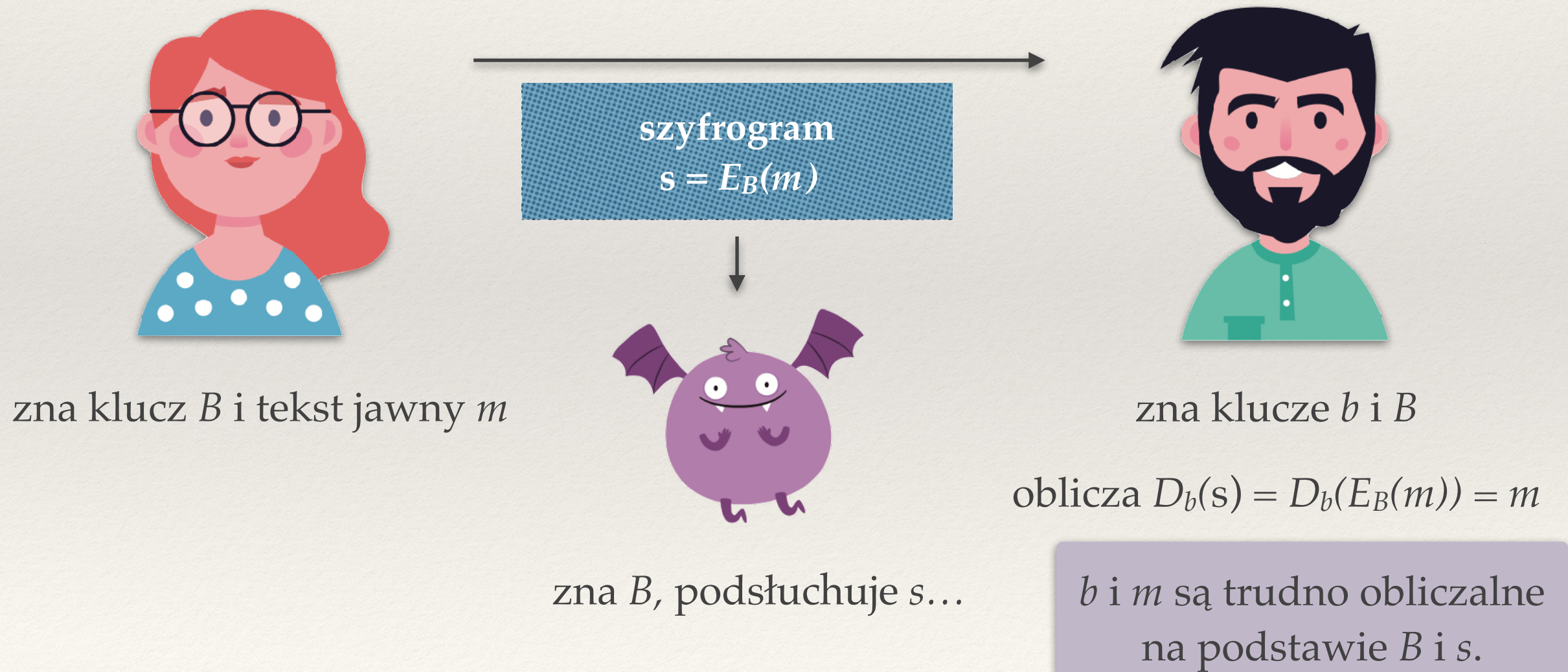
Szyfrowanie symetryczne

- ❖ Efektywne obliczeniowo.
- ❖ Zapewnia poufność, uwierzytelnianie nadawcy, integralność wiadomości.
- ❖ Problem: jak ustalić wspólny klucz?
 - ✦ Przesyłanie innym (zabezpieczonym) kanałem zazwyczaj niepraktyczne lub / i drogie.

Szyfrowanie asymetryczne

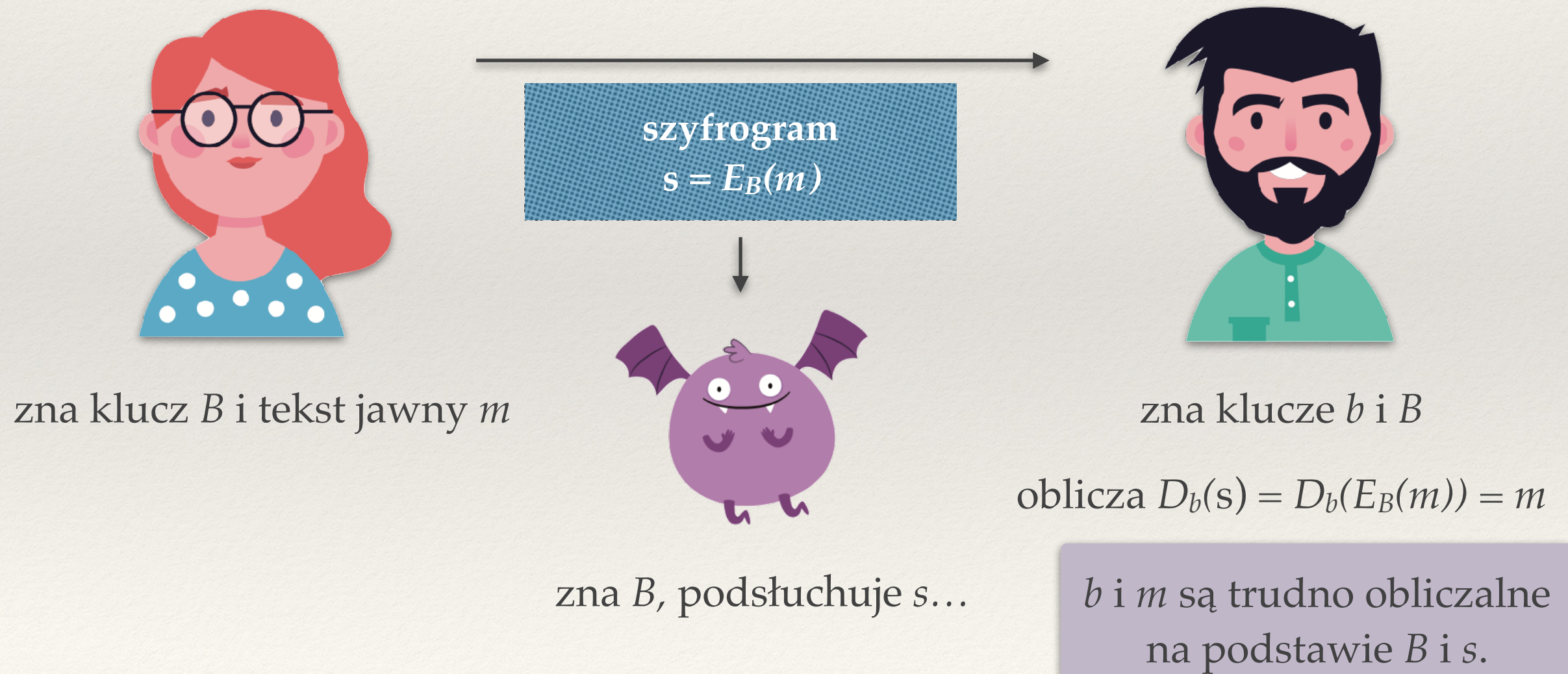
Szyfrowanie asymetryczne: założenia

- ❖ Bob ma klucz publiczny B (na stronie internetowej) i klucz prywatny b (w sejfie).
- ❖ Publiczny algorytm szyfrujący E i deszyfrujący D .
- ❖ Dla dowolnej wiadomości m zachodzi $D_b(E_B(m)) = m$.



Szyfrowanie asymetryczne: założenia

- ❖ Bob ma klucz publiczny B (na stronie internetowej) i klucz prywatny b (w sejfie).
- ❖ Szyfrować wiadomości może **każdy** znający klucz publiczny B .
- ❖ Deszyfrować te wiadomości może **tylko** znający klucz prywatny b .



Jak to jest w ogóle możliwe?

- ❖ Mamy tekst jawny m i znamy B .
- ❖ Możemy obliczyć $s = E_B(m)$.
- ❖ Obliczenie $D_b(s)$ bez znajomości b jest obliczeniowo trudne.
- ❖ B i b “pasują do siebie”, tj. $D_b(E_B(m)) = m$.
- ❖ Idea: pewne odwracalne operacje są szybsze niż ich odwrotności.
 - ✦ Mnożenie dwóch liczb pierwszych vs. rozkład na czynniki.
 - ✦ Podnoszenie do potęgi modulo vs. logarytm dyskretny.

Algorytm RSA: generowanie kluczy

Generujemy (dla siebie) klucz publiczny i prywatny:

- ❖ Wybieramy $p \neq q$ (duże liczby pierwsze).
- ❖ Niech $n = p \cdot q$.
- ❖ Wybieramy liczbę e względnie pierwszą z $\phi(n) = (p - 1) \cdot (q - 1)$.
- ❖ Znajdujemy takie d , że $d \cdot e \bmod \phi(n) = 1$ (algorytm Euklidesa).

Klucze: publiczny: (e, n) , prywatny: (d, p, q) .

RSA: Szyfrowanie i deszyfrowanie

Jak zaszyfrować liczbę $m \in [0, n)$?

- ❖ Szyfrowanie: $s = E(m) = m^e \bmod n$.
- ❖ Deszyfrowanie: $D(s) = s^d \bmod n$.
- ❖ Ta sama funkcja do szyfrowania i deszyfrowania.

Szyfrowanie całej wiadomości:

- ❖ Dzielimy na fragmenty rozmiaru $\leq \log n$, każdy szyfrujemy osobno.

RSA: Dlaczego to działa? (1)

Twierdzenie. Dla dowolnego $m \in [0, n)$ zachodzi $D(E(m)) = m$.

Dowód. $D(E(m)) = (m^e \bmod n)^d \bmod n$

$$= (m^e)^d \bmod n$$

$$= m^{k \cdot \phi(n) + 1} \bmod n, \quad \text{gdzie } k \in \mathbf{N} \cup \{0\}$$

$$= m$$



Tw Eulera: dla dowolnego
 $m \in Z_n^*$ zachodzi $m^{\phi(n)} \bmod n = 1$.

A co jeśli
 $m \notin Z_n^*$?

RSA: Dlaczego to działa? (2)

Twierdzenie. Dla dowolnego $m \in [0, n)$ zachodzi $D(E(m)) = m$.

Dowód. (dla dowolnego m)

❖ Pokażemy, że:

$$\diamond m^{k \cdot (p-1) \cdot (q-1) + 1} \equiv m \pmod{p},$$

$$\diamond m^{k \cdot (p-1) \cdot (q-1) + 1} \equiv m \pmod{q}.$$

❖ Stąd z Chińskiego twierdzenia o resztach:

$$D(E(m)) = m^{k \cdot (p-1) \cdot (q-1) + 1} \equiv m \pmod{p \cdot q}.$$



RSA: Dlaczego to działa? (3)

Lemat. $m^{k \cdot (p-1) \cdot (q-1) + 1} \equiv m \pmod{p}$.

Dowód. Niech $m = a \cdot p + m_p$, gdzie $0 \leq m_p < p$.

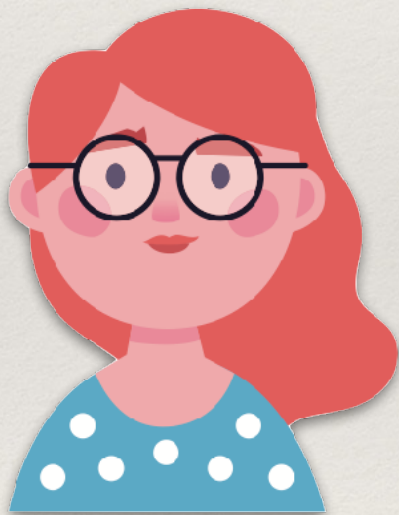
$$\begin{aligned} \text{Wtedy } m^{k \cdot (p-1) \cdot (q-1) + 1} \pmod{p} &= (m_p)^{k \cdot (p-1) \cdot (q-1) + 1} \pmod{p} \\ &= m_p \cdot [(m_p)^{(p-1)}]^{k \cdot (q-1)} \pmod{p} =: (*) \end{aligned}$$

1. Jeśli $m_p = 0$, to $(*) = 0 = m_p \equiv m \pmod{p}$.

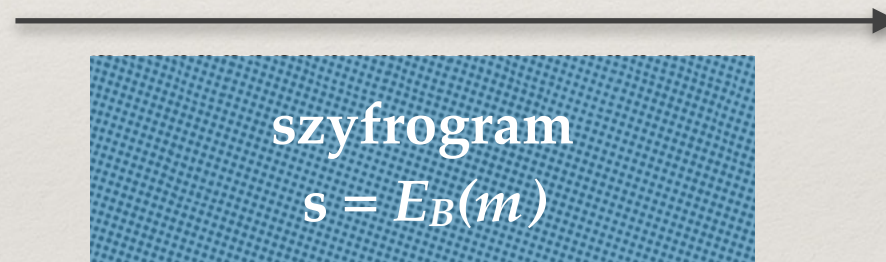
2. Jeśli $m_p > 0$, to z tw. Eulera $(m_p)^{(p-1)} \pmod{p} = 1$
i stąd $(*) = m_p \cdot 1^{k \cdot (q-1)} \pmod{p} = m_p \equiv m \pmod{p}$. ■

Szyfrowanie asymetryczne: podsumowanie

- ❖ Bob ma klucz publiczny B (na stronie internetowej) i klucz prywatny b (w sejfie).
- ❖ Szyfrować wiadomości może **każdy** znający klucz publiczny B .
- ❖ Deszyfrować te wiadomości może **tylko** znający klucz prywatny b .



zna klucz B i tekst jawny m

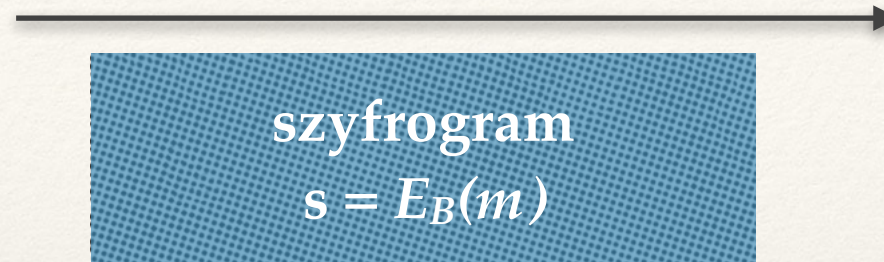
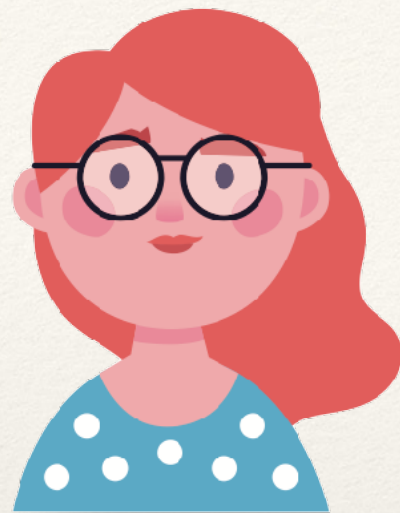


zna klucze b i B

oblicza $D_b(s) = D_b(E_B(m)) = m$

Uwierzytelnianie

Uwierzytelnianie (potwierdzanie tożsamości)



zna klucz publiczny B i tekst jawny m

zna klucz publiczny B i prywatny b

Co wiedzą poszczególne osoby?

- ❖ Alicja nie musi sprawdzać, czy po drugiej stronie jest Bob, bo szyfrogram może odszyfrować tylko Bob.
- ❖ Ale Bob nie wie, kto wysłał wiadomość!

Tego problemu nie było
w szyfrowaniu
symetrycznym.

Algorytm RSA jeszcze raz

- ❖ **RSA: ta sama funkcja szyfrująca i deszyfrująca: $E = D$.**
 - ✦ Nie tylko $E_b(E_B(m)) = m$, ale też $E_B(E_b(m)) = m$.
- ❖ **$E_b(m)$ = podpis cyfrowy wiadomości m .**
 - ✦ Prawie prawda; za chwilę zmodyfikujemy tę definicję.
 - ✦ Tylko Bob (posiadacz b) może wygenerować podpis $E_b(m)$.
- ❖ **Weryfikacja podpisu (czy Bob jest nadawcą?).**
 - ✦ Mamy parę: wiadomość m + podpis $p = E_b(m)$.
 - ✦ Znamy klucz publiczny B .
 - ✦ Sprawdzamy, czy $m = E_B(p)$.

Uwierzytelnianie za pomocą podpisu cyfrowego (1)



klucz publiczny A
i prywatny a

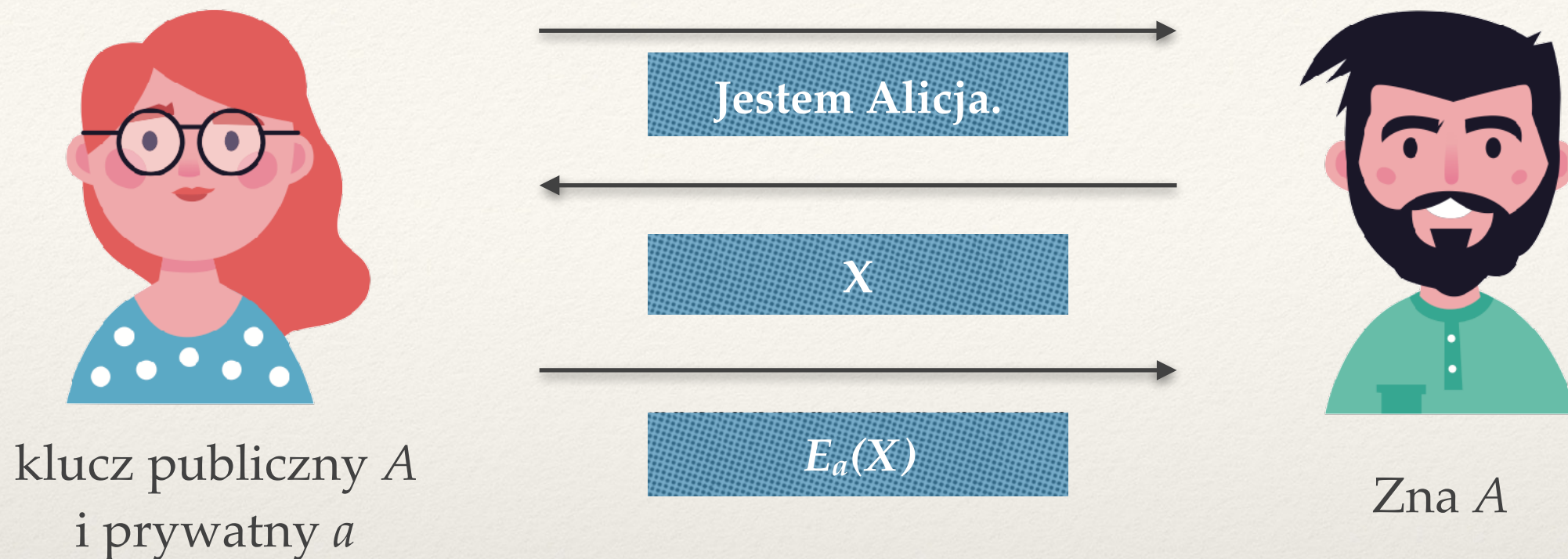
Oto dowód, że jestem
Alicją: $X, E_a(X)$



Zna A

- ❖ Tylko Alicja jest w stanie dla danego X wygenerować podpis $E_a(X)$.
- ❖ Ale możliwy atak powtórzeniowy: adwersarz przechwytuje i później odtwarza komunikat $(X, E_a(X))$.

Uwierzytelnianie za pomocą podpisu cyfrowego (2)



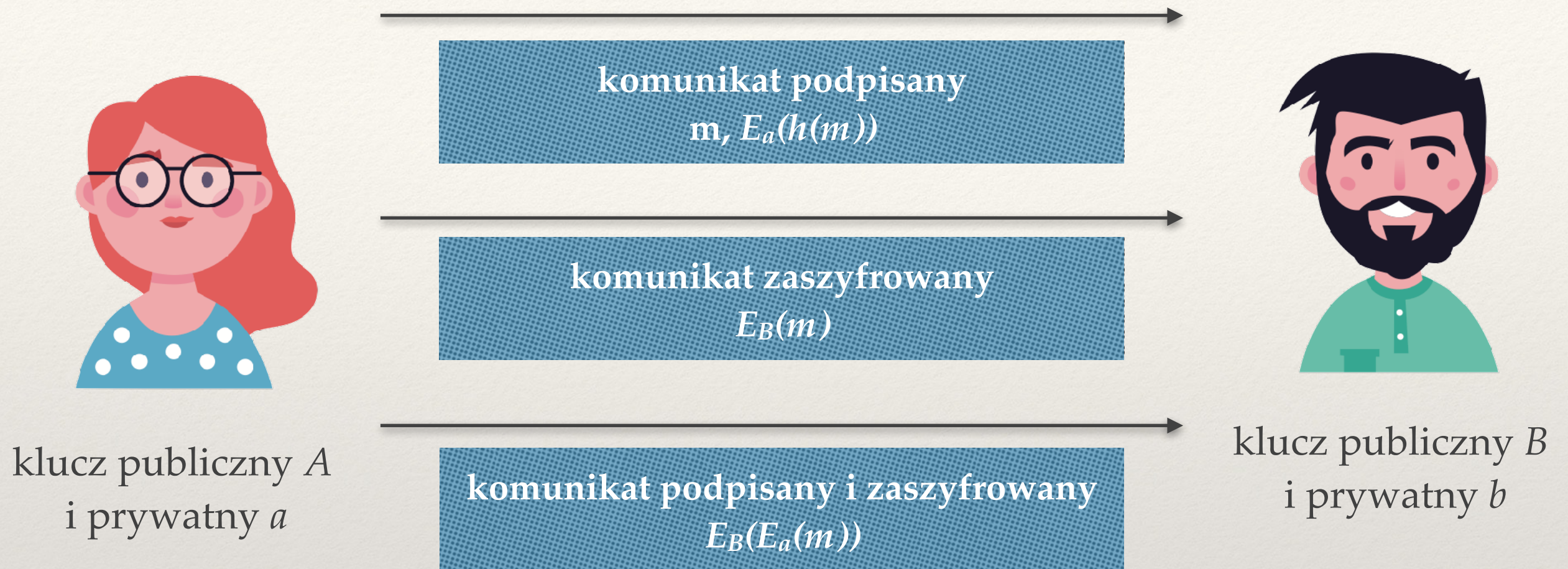
- ❖ Bob wybiera unikatowe, wcześniej niewykorzystywane X .
- ❖ Bob sprawdza, czy $E_A(E_a(X)) = X$.
- ❖ Alicja udowadnia w ten sposób że zna klucz prywatny a .

Efektywność podpisów

- ❖ Podpis zdefiniowaliśmy jako: $E_a(m)$.
 - ✦ Tekst z podpisem to para $(m, E_a(m))$.
 - ✦ Wada: długość podpisu \approx długość wiadomości.

- ❖ Lepiej:
 - ✦ Podpis to $E_a(h(m))$, gdzie h to kryptograficzna funkcja skrótu.
 - ✦ Bezpieczeństwo zakłada też trudność znalezienia kolizji dla h .

Podsumowanie: wysyłanie wiadomości m



- ❖ Ustandaryzowane jako PGP (Pretty Good Privacy).
- ❖ PGP wykorzystywane przy podpisywaniu maili i pakietów w systemach operacyjnych.

HMAC a podpisy cyfrowe

- ❖ Przypomnienie: HMAC (Message Authentication Code).
 - ♦ m = wiadomość.
 - ♦ s = sekret znany nadawcy i odbiorcy.
 - ♦ $\text{HMAC} = h(s \# h(s \# m))$.
 - ♦ Wykorzystywany do zapewniania integralności, potwierdza również znajomość sekretu s .
- ❖ HMAC nie jest podpisem cyfrowym, bo:
 - ♦ może go wykonać każda osoba znająca s ,
 - ♦ zweryfikować może go tylko osoba znająca s .

Dystrybucja kluczy publicznych

Klucze a osoby

❖ Szyfrowanie

- ♦ $E_B(m)$ może go odczytać tylko osoba znająca klucz prywatny b .
- ♦ Skąd wiemy, że tą osobą jest Bob?

❖ Podpisywanie

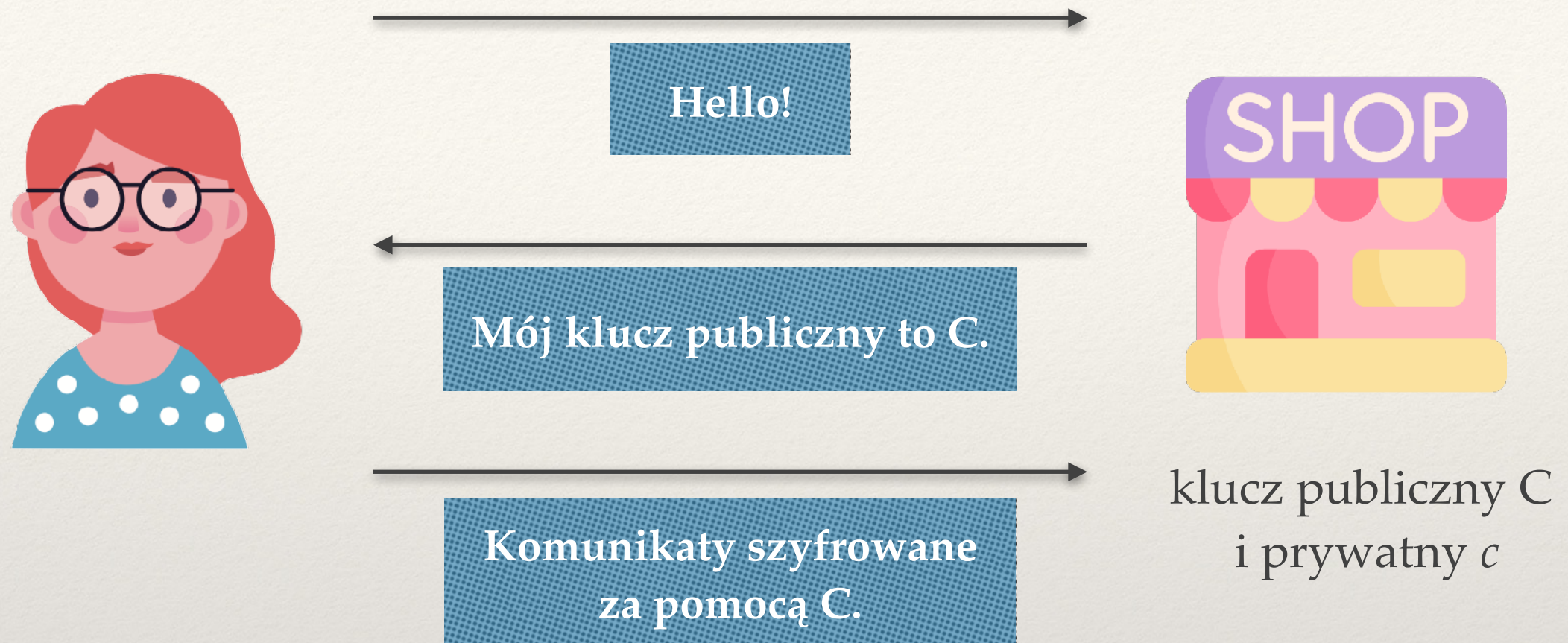
- ♦ Podpis $E_a(h(m))$ może wykonać tylko osoba znająca klucz prywatny a .
- ♦ Skąd wiemy, że tą osobą jest Alicja?

❖ Jak powiązać klucze z konkretną osobą?

Skąd wziąć czyjś klucz publiczny?

- ❖ Przekazanie klucza publicznego zabezpieczonym kanałem.
- ❖ Przekazanie klucza publicznego niezabezpieczonym kanałem + potwierdzenie funkcji skrótu zabezpieczonym kanałem.
 - ✦ Klucz publiczny Alicji (A) na stronie www.
 - ✦ Bob pobiera ze strony klucz A' .
 - ✦ Alicja i Bob weryfikują (np. telefonicznie), czy $h(A) = h(A')$.
- ❖ Pomysły akceptowalne przy komunikacji między dwiema osobami.

Komunikacja z usługą



Jak zapewnić powiązanie strona www \leftrightarrow klucz C?

Certyfikaty

Założmy, że:

- ❖ Alicja ma klucz publiczny B i wie, że należy on do Boba.
- ❖ Alicja wie, że Bob odpowiedzialnie używa podpisów cyfrowych.
- ❖ Alicja dostaje wiadomość **“klucz publiczny Charliego to C ”** podpisaną kluczem b .



certyfikat

Wtedy:

- ❖ Alicja może zweryfikować, że wiadomość napisał Bob.
- ❖ Alicja wie, Bob nie uwierzytelniałby nieprawdy.
- ❖ Alicja wie, że C to klucz publiczny Charliego.

Certyfikaty w PGP

Charlie umieszcza w publicznym repozytorium:











- ❖ swój klucz publiczny C;
- ❖ certyfikaty = wiadomości „klucz publiczny Charliego to C” podpisane kluczami różnych osób.

Umożliwia budowanie grafu certyfikacji.

- ❖ Podpisywanie kluczy publicznych: częste w środowisku programistów open source.

Certyfikaty dla usług (np. stron www)

- ❖ Certyfikaty generowane przez specjalne zaufane urzędy certyfikacji (CA).
- ❖ Certyfikat łączy adres domeny z kluczem publicznym.
- ❖ Można zgłosić się do CA, żeby dostać certyfikat (CA podpisuje nasz klucz publiczny).
- ❖ Klucze publiczne CA są wpisane w przeglądarki
 - ♦ Zbiory tych kluczy mogą się różnić przeglądarkami.

			DigiCert High Assurance EV Root CA Root certificate authority Expires: Monday, 10 November 2031 at 01:00:00 Central European Standard Time ✔ This certificate is valid		
Name		Kind	Expires		
 DigiCert Global Root G3		certificate	15 Jan 2038 at 13:00:00		
 DigiCert High Assurance EV Root CA		certificate	10 Nov 2031 at 01:00:00		
 DigiCert Trusted Root G4		certificate	15 Jan 2038 at 13:00:00		
 DST Root CA X3		certificate	30 Sep 2021 at 16:01:15		
 E-Tugra Certification Authority		certificate	3 Mar 2023 at 13:09:48		
 Echoworx Root CA2		certificate	7 Oct 2030 at 12:49:13		
 EE Certification Centre Root CA		certificate	18 Dec 2030 at 00:59:59		
 Entrust Root Certification Authority		certificate	27 Nov 2026 at 21:53:42		
 Entrust Root Certification Authority - EC1		certificate	18 Dec 2037 at 16:55:36		

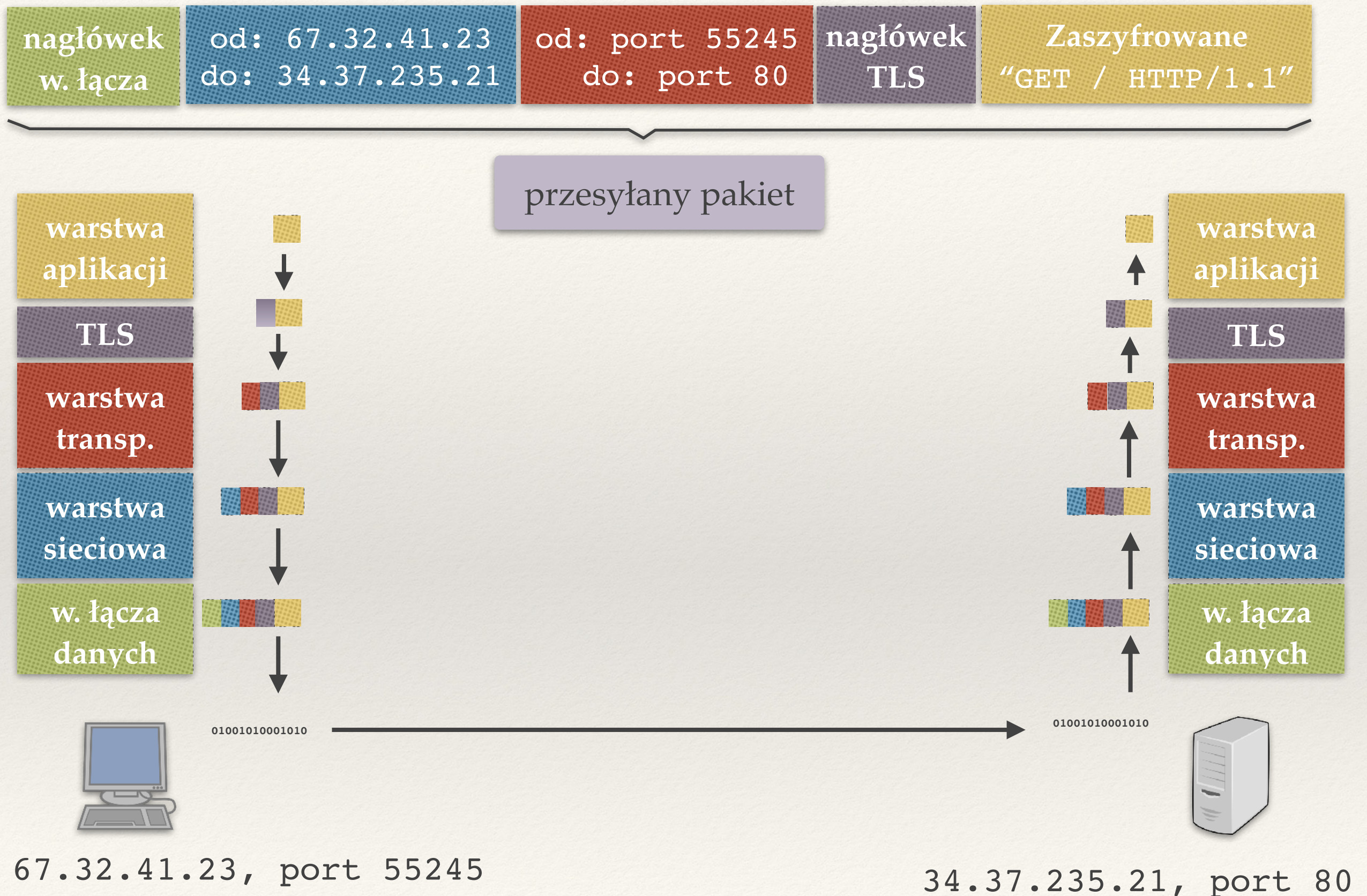
Weryfikacja

- ❖ CA powinno zweryfikować, czy faktycznie jesteśmy posiadaczami danej domeny.
 - ♦ Różne przeglądarki wierzą różnym CA.
- ❖ Obecnie dostępne darmowe CA (np. Let's Encrypt, ZeroSSL, ...).
 - ♦ Weryfikacja na podstawie umieszczenia odpowiedniego pliku na stronie www.

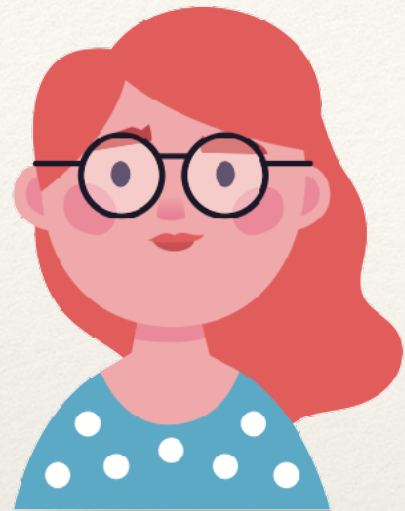
TLS (Transport Layer Security)

- ❖ Następca SSL (Secure Socket Layer).
- ❖ Warstwa pośrednicząca pomiędzy warstwą transportową i warstwą aplikacji.
- ❖ Odpowiada za szyfrowanie i uwierzytelnianie.
- ❖ Warianty usługi wykorzystujące TLS często działają na innym porcie:
 - ♦ HTTP → port 80,
 - ♦ HTTPS = TLS + HTTP → port 443.

Internetowy model warstwowy z szyfrowaniem



Łączenie z serwerem HTTPS



Hello!

$m = \text{"Klucz publiczny domeny } \text{www.example.com to } C"$
+
 $E_g(m)$, gdzie g to klucz prywatny CA.



domena
`www.example.com`,
klucz publiczny C
i prywatny c

- ❖ Przeglądarka Alicji sprawdza prawdziwość podpisu.
- ❖ Przeglądarka ma i ufa kluczowi publicznemu G .
- ❖ Od tej pory Alicja mogłaby szyfrować wiadomości dla serwera HTTP kluczem C .
 - ✦ Co z uwierzytelnianiem użytkownika?
 - ✦ Co z szyfrowaniem komunikatów od serwera HTTP?

Uwierzytelnianie użytkownika

Technicznie możliwe w TLS.

- ❖ Ale wymagałoby certyfikowanego klucza użytkownika.
- ❖ Zazwyczaj uwierzytelnianie na poziomie warstwy aplikacji przez parę użytkownik + hasło / token / plik cookie.

Klucze sesji

Serwer też powinien szyfrować dane do klienta.

- ❖ Klient zazwyczaj nie ma swojego klucza publicznego.

Rozwiązanie stosowane w TLS:

- ❖ Przeglądarka generuje **symetryczny klucz sesji** (np. AES).
- ❖ Przeglądarka szyfruje go kluczem publicznym serwera www i wysyła do serwera www.
- ❖ Dalsza komunikacja szyfrowana kluczem sesji.
- ❖ Szyfrowanie symetryczne jest wielokrotnie szybsze niż szyfrowanie asymetryczne.

- ❖ Zrobiliśmy wiele uproszczeń.
 - ✦ Ścieżki certyfikacji zamiast pojedynczych krawędzi.
 - ✦ Randomizacja kluczy sesji.
 - ✦ Współpraca z segmentacją TCP.
 - ✦ ...
- ❖ Ten sam schemat stosowany w wielu innych miejscach, np. WPA2/3 + RADIUS.

Dodatek: atak urodzinowy

List rekomendacyjny (schemat)

- ❖ Alicja chce rekomendować na stanowisko osobę X.
- ❖ Plan Alicji:
 - ✦ Zlecić napisanie listu (wiadomości m) Charliemu.
 - ✦ Sprawdzić, czy m zawiera rekomendację osoby X.
 - ✦ Obliczyć i dać Charliemu $p = E_a(h(m))$.
 - ✦ Charlie powinien wysłać $(m, p) = (m, E_a(h(m)))$ do pracodawcy.
- ❖ Charlie preferuje osobę Y i postanawia zaatakować funkcję skrótu h :
 - ✦ Funkcja h generuje 80-bitowy skrót.
 - ✦ Charlie chce napisać list m' polecający Y, taki że $h(m') = h(m)$ i wysłać (m', p) .
 - ✦ $(m', p) = (m', E_a(h(m))) = (m', E_a(h(m')))$, tj. jest poprawnie podpisana.
 - ✦ Ale znalezienie m' to sprawdzenie ok. 2^{80} wiadomości (nierealistycznie dużo).

List rekomendacyjny (problem)

- ❖ Założyliśmy, że Charlie **najpierw** wygeneruje m , a **następnie** będzie szukać m' , takiego że $h(m') = h(m)$. Koszt: sprawdzenie $\approx 2^{80}$ wiadomości.
- ❖ Ale Charlie może wygenerować dwa zbiory wiadomości, oba o liczności $\approx 2^{40}$ wiadomości.
 - ♦ M_X : wiadomości polecające X
 - ♦ M_Y : wiadomości polecające Y
 - ♦ Z prawdopodobieństwem $\Omega(1)$ istnieją $m \in M_X$ i $m' \in M_Y$, takie że $h(m') = h(m)$ (ćwiczenie).
- ❖ **Atak urodzinowy**: analogia do tzw. paradoksu urodzin.

Lektura dodatkowa

- ❖ Kurose & Ross: rozdział 8.
- ❖ Tanenbaum: rozdział 8.

Zagadnienia

- ❖ Czym szyfrowanie symetryczne różni się od asymetrycznego?
- ❖ Na czym polega bezpieczeństwo przy szyfrowaniu asymetrycznym?
- ❖ Opisz algorytm RSA.
- ❖ Czy różni się szyfrowanie od uwierzytelniania?
- ❖ Co to jest atak powtórzeniowy?
- ❖ Czy w szyfrowaniu asymetrycznym szyfrujemy kluczem publicznym czy prywatnym?
- ❖ Na czym polega podpisywanie wiadomości? Jakim kluczem to robimy?
- ❖ Jak można wykorzystać podpisy cyfrowe do uwierzytelniania?
- ❖ Czy HMAC można wykorzystać do uwierzytelniania? Czy HMAC jest podpisem cyfrowym?
- ❖ Dlaczego lepiej podpisywać funkcję skrótu wiadomości niż samą wiadomość? Z jakim ryzykiem się to wiąże?
- ❖ Co to są certyfikaty? Co to jest ścieżka certyfikacji?
- ❖ Co to jest urząd certyfikacji (CA)?
- ❖ Jak TLS zapewnia bezpieczeństwo połączenia?
- ❖ W jaki sposób w TLS następuje uwierzytelnienie serwera, z którym się łączymy?
- ❖ Co to są klucze sesji? Po co się je stosuje?
- ❖ Co to są kolizje kryptograficznej funkcji skrótu?
- ❖ Na czym polega atak urodzinowy?