Podstawy kryptografii

Sieci komputerowe

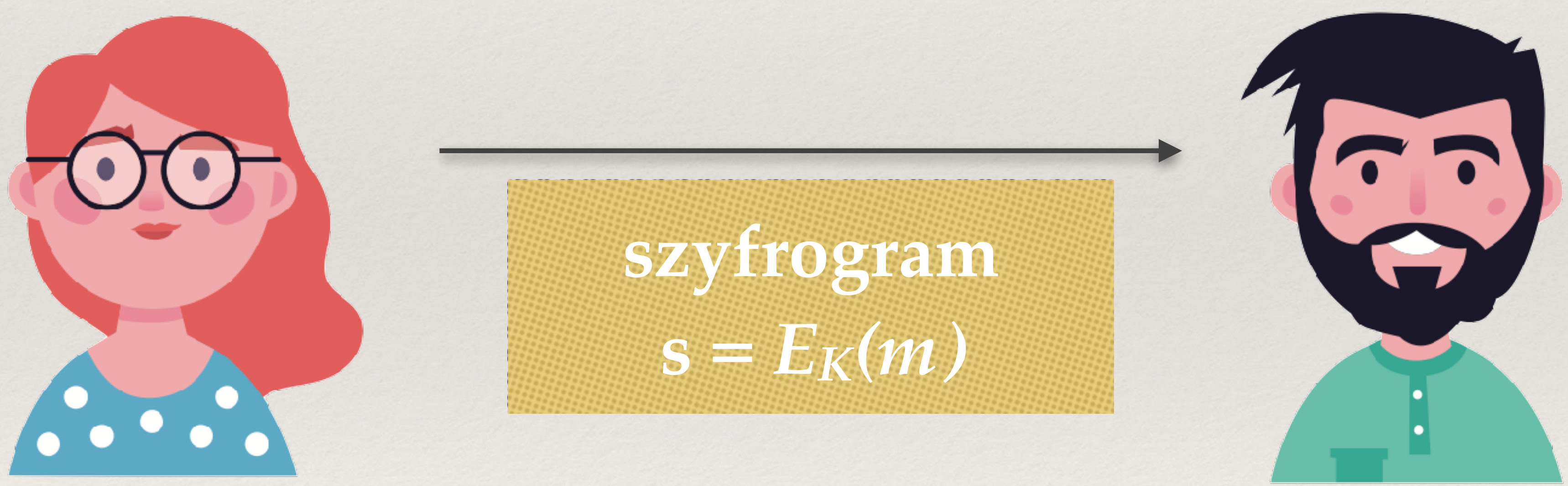
Wykład 12

Marcin Bieńkowski

Szyfrowanie asymetryczne

Poprzednio: szyfrowanie symetryczne

- Publiczny algorytm szyfrujący E parametryzowany kluczem K.
- Publiczny algorytm deszyfrujący D parametryzowany kluczem K, taki że $D_K(E_K(m)) = m$ dla każdego tekstu jawnego m i klucza K.
- **Alicja i Bob ustalają pewien wspólny klucz K.**



zna klucz K i tekst jawny m

zna K

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

Szyfrowanie symetryczne: w poprzednim odcinku

- **Główny problem:** jak ustalić wspólny klucz K?
- Można przesłać innym kanałem (*zabezpieczonym*).
  - Zazwyczaj niepraktyczne lub/i drogie.

Szyfrowanie asymetryczne: założenia

Bob ma klucz publiczny **B** (na stronie internetowej)
i klucz prywatny ***b*** (w sejfie)

zna klucz B i tekst jawny m

zna klucze b i B

Istnieje (publiczny) algorytm szyfrujący E i deszyfrujący D, taki że 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)

szyfrogram
$s = E_B(m)$

zna klucz B i tekst jawny m

zna klucze b i B
oblicza $D_b(s) = D_b(E_B(m)) = m$

Szyfrowanie asymetryczne: założenia

Bob ma klucz publiczny **B** (na stronie internetowej)
i klucz prywatny ***b*** (w sejfie)

szyfrogram
$s = E_B(m)$

zna klucz B i tekst jawny m

zna klucze b i B
oblicza $D_b(s) = D_b(E_B(m)) = m$

zna B, podsłuchuje s…

People and hand vectors created by `www.freepik.com`

Szyfrowanie asymetryczne: założenia

Bob ma klucz publiczny **B** (na stronie internetowej)
i klucz prywatny ***b*** (w sejfie)

szyfrogram
s = $E_B(m)$

zna klucz B i tekst jawny m

zna klucze b i B

Istnieje (publiczny) algorytm szyfrujący E i deszyfrujący D, taki że:

- dla dowolnej wiadomości m zachodzi $D_b(\mathrm{s}) = D_b(E_B(m)) = m$,
- **b i m są trudno obliczalne na podstawie B i s.**

Inne spojrzenie

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*.

Inne spojrzenie

Bob ma klucz publiczny **B** (na stronie internetowej)
i klucz prywatny ***b*** (w sejfie)

Idea: pewne odwracalne operacje są szybsze niż ich odwrotności

Mnożenie dwóch liczb pierwszych vs. rozkład na czynniki

- **Szyfrować** wiadomości może **każdy** znający klucz publiczny B.
- **Deszyfrować** te wiadomości może **tylko** znający klucz prywatny b.

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 z $m \in [0, n)$?

- Szyfrowanie: $s = E(m) = m^e \bmod n$
- Deszyfrowanie: $D(s) = s^d \bmod n$

Szyfrowanie całej wiadomości: dzielimy na kawałki 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. $D(E(m)) = (m^e \bmod n)^d \bmod n$

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

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

$= m$

RSA: Dlaczego to działa? (1)

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

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

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

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

$= m$

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

RSA: Dlaczego to działa? (1)

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

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

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

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

$= m$

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

A co jeśli m $\notin Z^*_n$? Tzn. p | m lub q | m?

RSA: Dlaczego to działa? (2)

Dowód dla dowolnego m. Pokażemy, że:

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

Stąd z Chińskiego twierdzenia o resztach:

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

RSA: Dlaczego to działa? (3)

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

Wtedy $m^{k \cdot (p-1) \cdot (q-1) + 1}$ mod p $= (m_p)^{k \cdot (p-1) \cdot (q-1) + 1}$ mod p

$= m_p \cdot [\, (m_p)^{(p-1)} \,]^{k \cdot (q-1)}$ mod $p = (*)$

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

Czyli $m^{k \cdot (p-1) \cdot (q-1) + 1} \equiv m$ mod p

Szyfrowanie asymetryczne: jeszcze raz

Bob ma klucz publiczny **B** (na stronie internetowej)
i klucz prywatny ***b*** (w sejfie)

szyfrogram
s = $E_B(m)$

zna klucz B i tekst jawny m

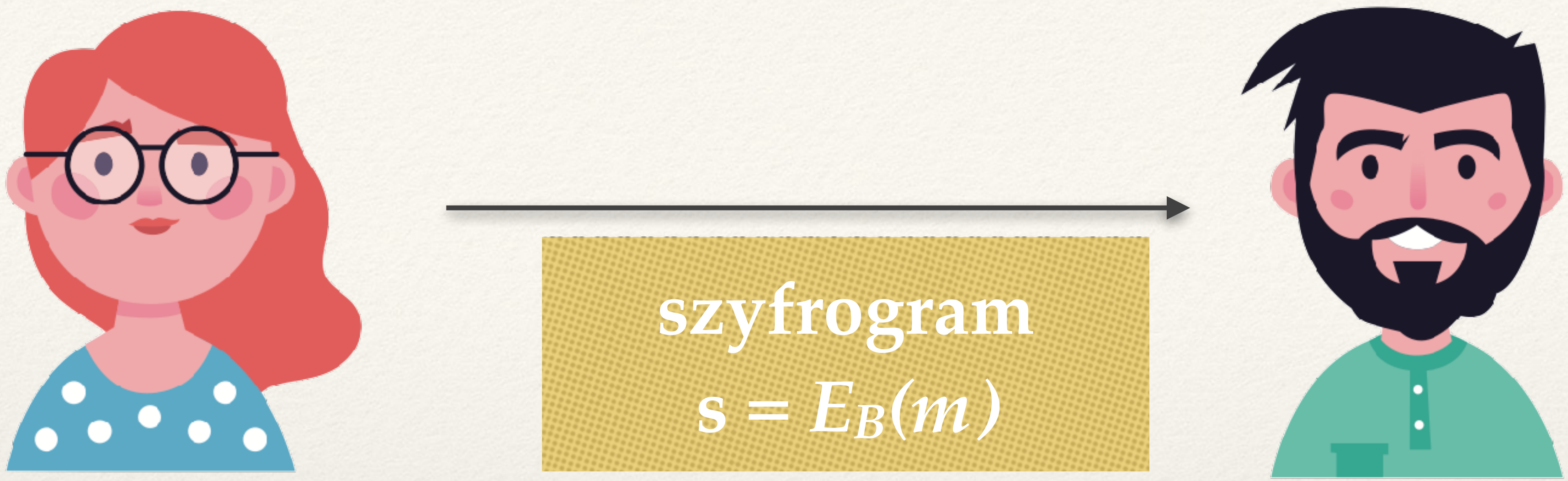
zna klucze b i B

Istnieje (publiczny) algorytm szyfrujący E i deszyfrujący D, taki że:

- dla dowolnej wiadomości m zachodzi $D_b(\mathrm{s}) = D_b(E_B(m)) = m$,
- b i m są trudno obliczalne na podstawie B i s.

Uwierzytelnianie

Uwierzytelnianie (potwierdzanie tożsamości)



zna klucz B i tekst jawny m

ma klucz publiczny B
i prywatny b

Co wiedzą poszczególne osoby?

- Alicja nie musi sprawdzać, czy po drugiej stronie jest Bob, bo i tak szyfrogram może zdeszyfrować tylko Bob
- Ale Bob nie wie, kto wysłał wiadomość!

Uwierzytelnianie (potwierdzanie tożsamości)

szyfrogram

$s = E_B(m)$

zna klucz *B* i tekst jawny *m*

ma klucz publiczny *B*
i prywatny *b*

Co wiedzą poszczególne osoby?

- Alicja nie musi sprawdzać, czy po drugiej st
szyfrogram może zdeszyfrować tylko Bob
- Ale Bob nie wie, kto wysłał wiadomość!

Tego problemu nie było w szyfrowaniu symetrycznym, bo sensowną wiadomość mogła wysłać tylko osoba znająca klucz *K*

Algorytm RSA jeszcze raz

- **RSA używa tej samej funkcji szyfrującej i deszyfrującej: E = D**
  - W szczególności dla pary kluczy B i b zachodzi nie tylko $E_b(E_B(m)) = m$, ale też $E_B(E_b(m)) = m$.
- ***$E_b(m)$* - podpis cyfrowy wiadomości *m***
  - Nie do końca prawda; za chwilę zmodyfikujemy tę definicję.
  - Tylko Bob (posiadacz klucza prywatnego b) może dla wiadomości m wygenerować podpis $E_b(m)$.
- **Weryfikacja podpisu (czy Bob jest nadawcą?)**
  - Mamy parę (wiadomość m, podpis $p = E_b(m)$) i znamy klucz publiczny B.
  - Sprawdzamy, czy $m = E_B(p)$.

Jak wykorzystać podpis w uwierzytelnianiu?

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

klucz publiczny A i prywatny a

Zna A

- Tylko Alicja jest w stanie dla danego X wygenerować podpis $E_a(X)$.

Jak wykorzystać podpis w uwierzytelnianiu?

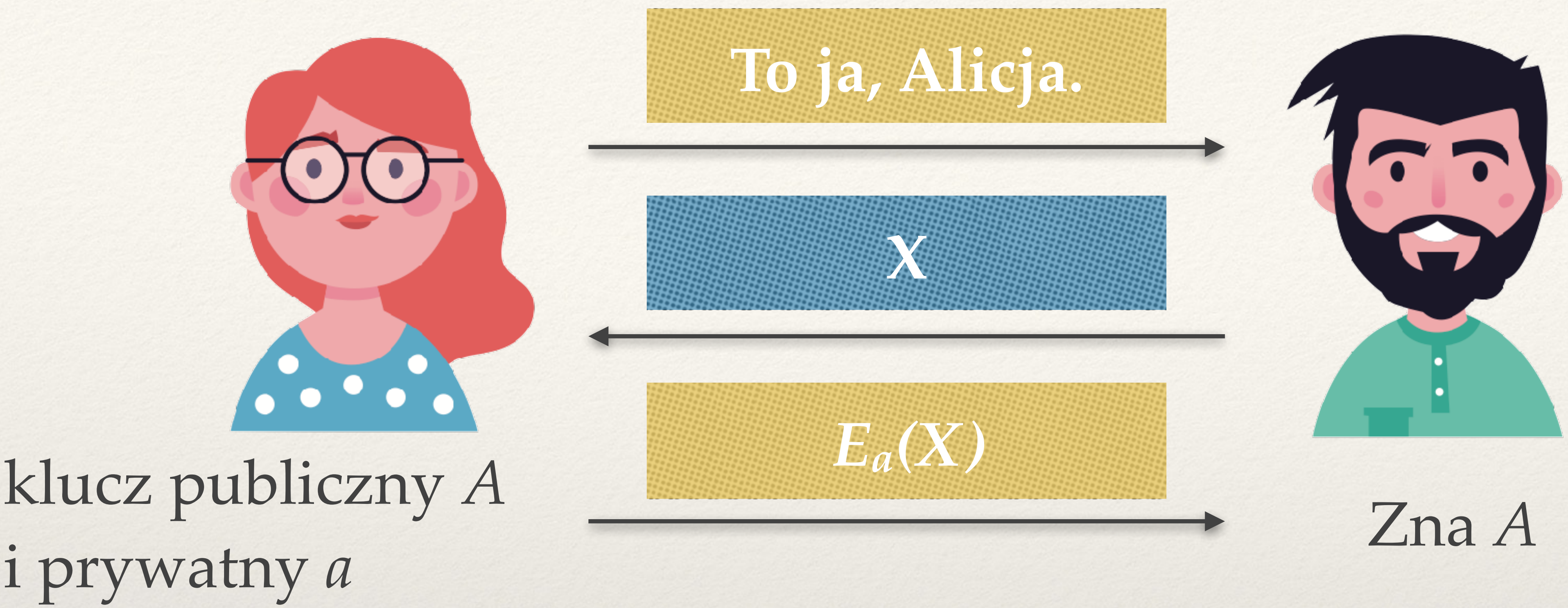
Oto dowód, że jestem Alicją: $(X, E_a(X))$

klucz publiczny A i prywatny a

Zna A

- Tylko Alicja jest w stanie dla danego X wygenerować podpis $E_a(X)$.
- Ale powyższy protokół jest podatny na **atak powtórzeniowy**: adwersarz może nagrać całą parę $(X, E_a(X))$ i wykorzystywać ją do udawania Alicji.

Uwierzytelnianie za pomocą podpisu cyfrowego



- Bob wybiera unikatowe, wcześniej niewykorzystywane X.
- Alicja udowadnia w ten sposób że zna klucz prywatny a.

People and hand vectors created by `www.freepik.com`

Podsumowanie: wysyłanie wiadomości *m*

klucz publiczny A
i prywatny a

klucz publiczny B
i prywatny b

- Podpisywanie: wiadomość + podpis kluczem prywatnym Alicji: $(m, E_a(m))$
- Szyfrowanie: wiadomość zaszyfrowana kluczem publicznym Boba: $E_B(m)$

Standard PGP (Pretty Good Privacy)

Efektywność podpisów

- **Podpis definiowaliśmy jako: $E_a(m)$**
  - Alicja wysyła pare $(m, E_a(m))$
  - Wady: rozmiar podpisu jest rzędu rozmiaru wiadomości + podpisywanie długo trwa.
- **Rozwiązanie stosowane w praktyce:**
  - Podpis to $E_a(h(m))$, gdzie h to kryptograficzna funkcja skrótu.
  - Bezpieczeństwo wymaga rownież, żeby trudno było znaleźć kolizję dla funkcji h.

HMAC a podpisy cyfrowe

- HMAC (Message Authentication Code)
  - m = wiadomość
  - s = sekret znany nadawcy i odbiorcy.
  - HMAC = $h(s \# h(s \# m))$

- HMAC można też wykorzystać do uwierzytelniania wiadomości.
- Konieczny jest wspólny sekret s.
- HMAC nie jest podpisem cyfrowym, bo:
  - może go wykonać każda osoba znająca s,
  - zweryfikować może go tylko osoba znająca s.

Dystrybucja kluczy publicznych

Skąd wziąć czyjś klucz publiczny? (1)

- **Szyfrogram $E_B(m)$**
  - tworzymy wykorzystując klucz publiczny B
  - może go odczytać tylko osoba znająca pasujący klucz prywatny b
  - **skąd wiemy, że tą osobą jest Bob?**

- **Podpis $E_a(m)$ dla wiadomości m**
  - możemy zweryfikować kluczem publicznym A
  - może go wygenerować tylko osoba znająca pasujący klucz prywatny a
  - **skąd wiemy, że tą osobą jest Alicja?**

- **Musimy mieć sposób powiązania klucza publicznego z konkretną osobą.**

Skąd wziąć czyjś klucz publiczny? (2)

- **Pomysł 1. Spotkanie fizyczne / telefoniczne / videokonferencja (mało praktyczne)**
  - Poza tym wtedy można ustalić klucz symetryczny, więc po co nam kryptografia asymetryczna?

- **Pomysł 2. Klucz publiczny dostępny na stronie WWW.**
  - Bezpieczeństwo oparte na tym, że nikt go nie podmieni! Konieczna weryfikacja, przykładowo:
  - 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)

Powyższe pomysły niepraktyczne dla komunikacji z usługą:

- Wchodzimy na stronę sklepu.
- Sklep mówi “mój klucz publiczny = ..., szyfruj do mnie dane tym kluczem”.
- Skąd wiemy, że łączymy się faktycznie z konkretnym sklepem?

Certyfikaty

Załóżmy, że:

(1) Alicja ma klucz publiczny *B* i **wie**, że należy on do Boba.

(2) Alicja wierzy w to, że Bob **odpowiedzialnie** używa podpisów cyfrowych.

(3) Ma **wiadomość "klucz publiczny Charliego to *C*" podpisaną kluczem *b*.**

Wtedy:

(1) => Alicja może zweryfikować, że wiadomość napisał Bob

(2) => Alicja wierzy, że Bob nie uwierzytelniałby nieprawdy.

(3) => Alicja wie, że *C* to klucz publiczny Charliego.

Certyfikaty

Załóżmy, że:

(1) Alicja ma klucz publiczny *B* i **wie**, że należy on do Boba.

(2) Alicja wierzy w to, że Bob **odpowiedzialnie** używa podpisów cyfrowych.

(3) Ma **wiadomość "klucz publiczny Charliego to *C*" podpisaną kluczem *b*.**

to jest certyfikat

Wtedy:

(1) => Alicja może zweryfikować, że wiadomość napisał Bob

(2) => Alicja wierzy, że Bob nie uwierzytelniałby nieprawdy.

(3) => Alicja wie, że C to klucz publiczny Charliego.

Certyfikaty w PGP

Na stronie WWW Charlie może umieścić:

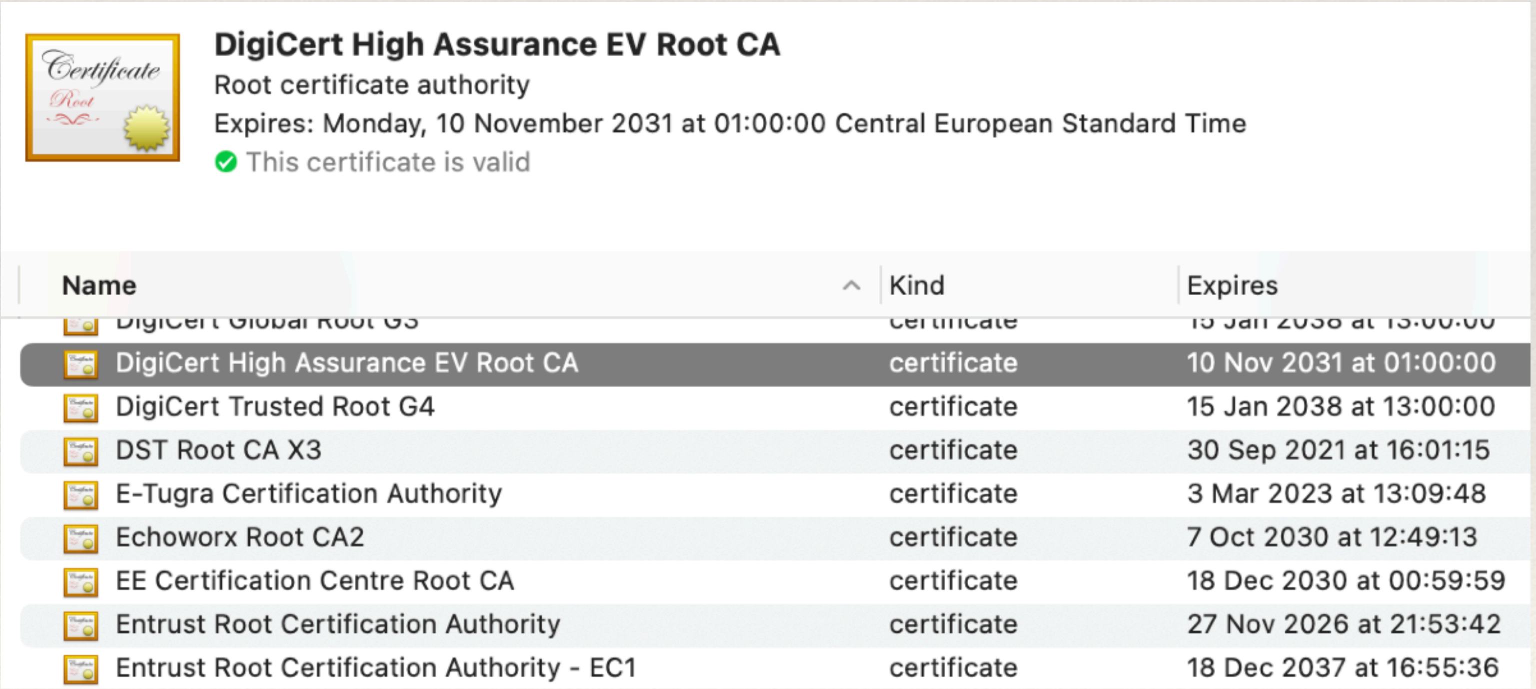
- swój klucz publiczny *C*.
- certyfikat „klucz publiczny Charliego to *C*" podpisany kluczami różnych osób.

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

Certyfikaty dla usług (np. stron www)

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

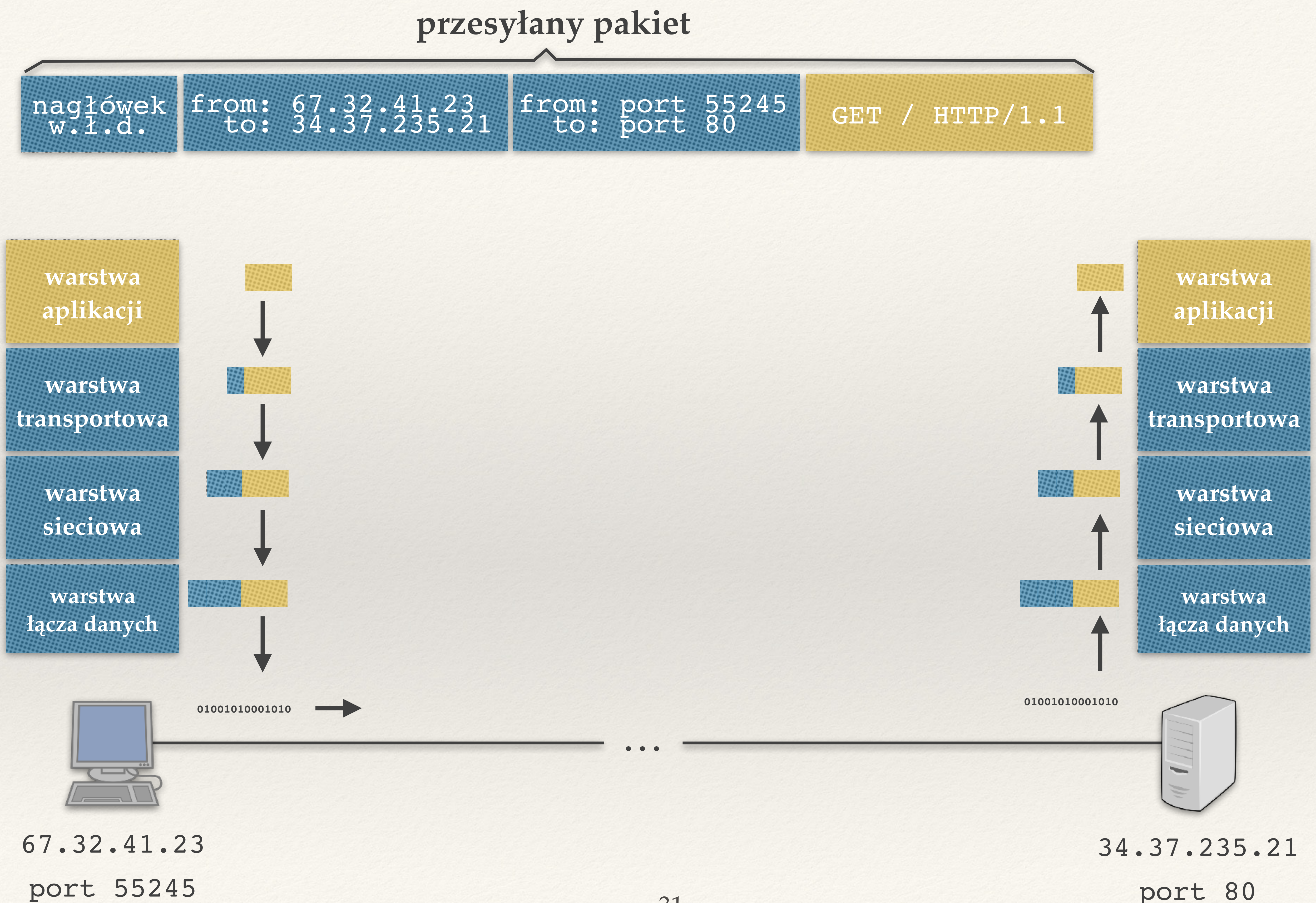
- Klucze publiczne CA są wpisane w przeglądarki
  - Zbiory tych kluczy mogą się różnić pomiędzy przeglądarkami.



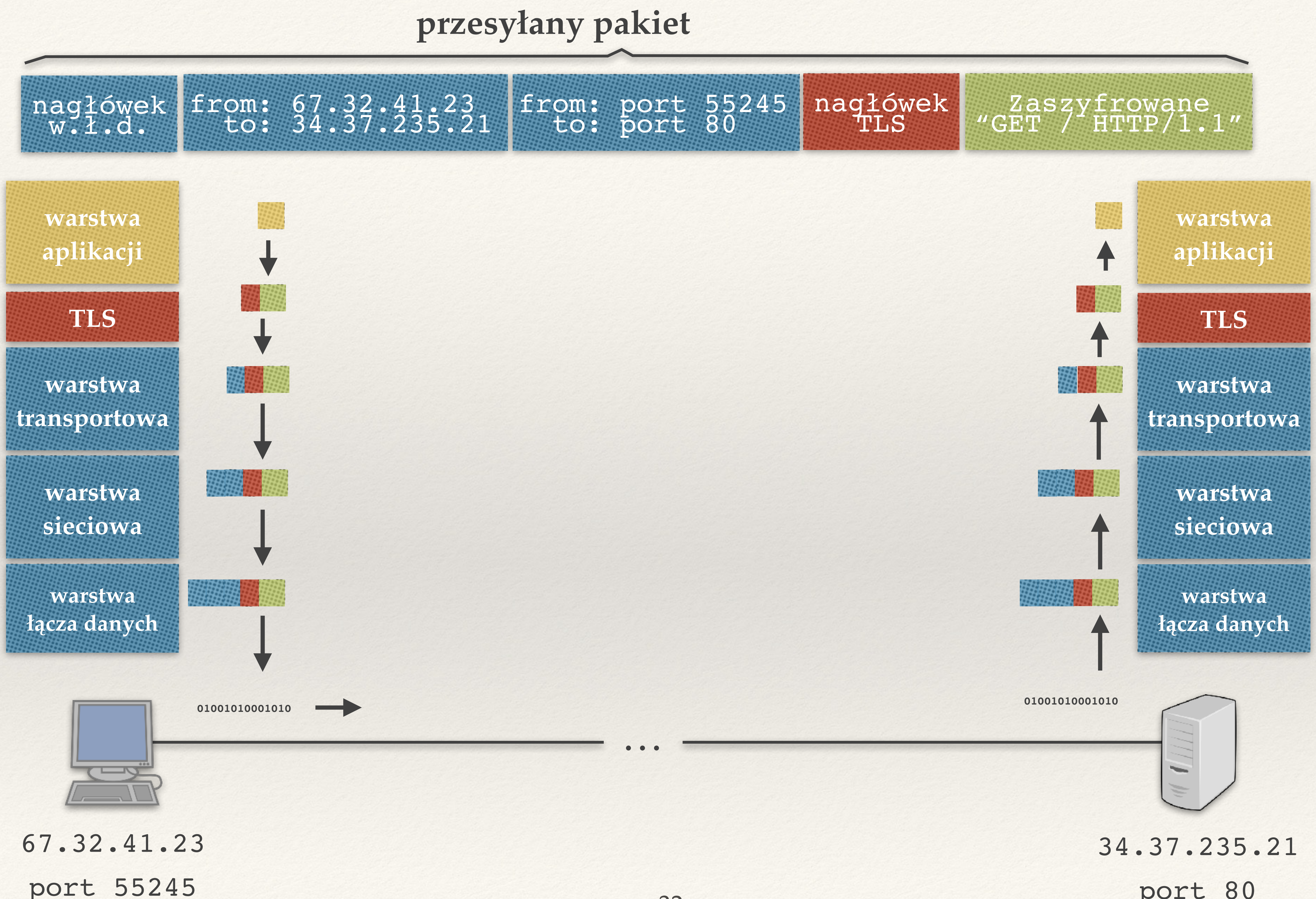
TLS (Transport Layer Security)

- Warstwa pośrednicząca pomiędzy warstwą transportową i warstwą aplikacji.
- Odpowiada za szyfrowanie i uwierzytelnianie.
- Warianty usługi wykorzystujące TLS mogą działać zarówno na tym samym porcie jak i na osobnym (np. HTTPS = HTTP over TLS, port 443).

Internetowy model warstwowy: bez szyfrowania



Internetowy model warstwowy: z szyfrowaniem



Łączenie z serwerem HTTPS

Uwierzytelnianie serwera:

- Serwer WWW wysyła certyfikat (klucz publiczny + dane o stronie) podpisany przez pewne CA.
- Przeglądarka sprawdza, czy:
  - ma klucz publiczny tego CA i sprawdza prawdziwość podpisu CA.
  - dane o stronie opisują tę stronę, z którą zamierzamy się łączyć.

Od tej pory mamy uwierzytelniony serwer:

- Szyfrujemy wiadomości dla serwera WWW jego kluczem publicznym
- Co z odpowiedziami od serwera WWW?
- Co z uwierzytelnianiem użytkownika?

Uwierzytelnianie użytkownika

Technicznie możliwe w TLS

- Ale wymagałoby żeby użytkownik też miał certyfikowany klucz publiczny.
- Zazwyczaj po prostu 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 jest szyfrowana kluczem sesji
- **Bonus:** szyfrowanie symetryczne jest wielokrotnie szybsze niż szyfrowanie asymetryczne!

Dodatek: podpisywanie i szyfrowanie

Podpisywanie i szyfrowanie

klucz publiczny A
i prywatny a

klucz publiczny B
i prywatny b

- Alicja chce wysłać wiadomość m do Boba.
- Alicja chce m zarówno podpisać jak i zaszyfrować.
- **Czy powinna wysłać $E_B(E_a(m))$ czy $E_a(E_B(m))$?**

People and hand vectors created by www.freepik.com

Wariant 1: podpisz, potem zaszyfruj (schemat)

$s = E_B(E_a(m))$

m = "kocham Cię"

Wariant 1: podpisz, potem zaszyfruj (problem)

$s = E_B(E_a(m))$

m = “kocham Cię”

$s' = E_C(E_a(m))$

- Charlie ma parę kluczy (c, C)
- Bob oblicza $s' = E_C(E_b(s)) = E_C(E_b(E_B(E_a(m)))) = E_C(E_a(m))$
- Bob wysyła s' do Charliego podszywając się pod Alicję

Wariant 2: zaszyfruj, potem podpisz (schemat)

$s = E_a(E_B(m))$

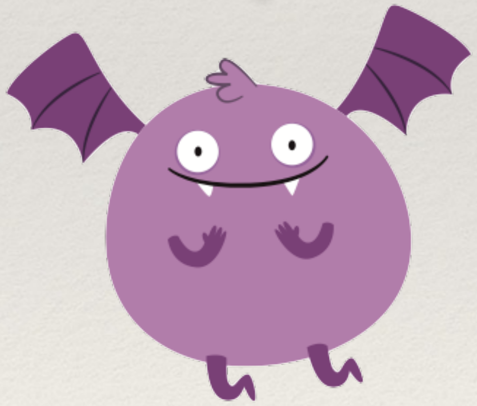
m = "mój genialny pomysł to …"

People and hand vectors created by www.freepik.com

Wariant 2: zaszyfruj, potem podpisz (schemat)

$s = E_a(E_B(m))$

m = "mój genialny pomysł to …"



podsłuchuje s

Wariant 2: zaszyfruj, potem podpisz (problem)

$s' = E_c(E_B(m))$

m = "mój genialny pomysł to …"

- Adwersarz przechwytuje wiadomość Alicji $s = E_a(E_B(m))$
- Adwersarz ma swoją parę kluczy (c, C)
- Adwersarz oblicza i wysyła
  $s' = E_c(E_A(s)) = E_c(E_A(E_a(E_B(m)))) = E_c(E_B(m))$

Jeden ze sposobów naprawy

$s = E_a(E_B(m))$

m = "mój genialny pomysł to …"

- Atak nie zadziała, jeśli Alicja zmieni wiadomość na:

  ,,To ja, Alicja. Mój genialny pomysł to …''.

- **Automatyzacja:** podpisz, zaszyfruj, podpisz, tj. wyślij $s = E_a(E_B(E_a(m)))$.

Dodatek: atak urodzinowy

List rekomendacyjny (schemat)

- **Alicja chce rekomendować na stanowisko osobę X.**
- **Plan Alicji (Alicja jest bardzo zajętą osobą):**
  - 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 mogłby 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.

- 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?
- Na jaki atak narażone jest podejście, w którym wiadomość najpierw szyfrujemy a potem podpisujemy?