Sieci komputerowe

Wykład 10: Kodowanie i szyfrowanie

Marcin Bieńkowski

Instytut Informatyki
Uniwersytet Wrocławski

Kodowanie

Kody

- Do przesyłanych wiadomości dodajemy dodatkowe bity kontrolne.
- **Kody detekcyjne:** pozwalają wykryć niektóre przekłamania transmisji.
- **Kody korekcyjne:** pozwalają wykryć i poprawić niektóre przekłamania transmisji.

Skąd biorą się błędy?

- Najczęściej: błędy w warstwie fizycznej
  (bity $\rightarrow$ analogowy sygnał $\rightarrow$ bity),
  bo analogowy sygnał dociera zniekształcony.
  - Przekłamania niektórych bitów.
  - Przekłamanie ciągu bitów.
  - Zgubienie niektórych bitów (rzadziej: wstawienie nieistniejących).
- Rzadziej: błędy urządzeń końcowych lub pośrednich (wadliwy RAM, błędy w oprogramowaniu).

Kody detekcyjne

- Sumy kontrolne
- Kody CRC
- Kody MAC

Sumy kontrolne

- Najprostszy wariant kodów detekcyjnych.
- Dodajemy do siebie (16/32-bitowe) słowa w przesyłanej wiadomości.
  - Warianty: przeniesienia, negowanie bitów, ...
  - Nie wykrywają zamian słów.
- Efektywnie obliczane przez CPU.
- Stosowane w warstwie sieciowej (IP) i transportowej (UDP).

Sumy kontrolne: bit parzystości

- Najprostszy wariant sumy kontrolnej.
- Do wiadomości dodajemy bit, który ustawiamy tak, żeby liczba ustawionych bitów w całości była parzysta.
- Wykrywa przekłamania nieparzystej liczby bitów.

Kody CRC (*Cyclic Redundancy Check*)

- $\rightarrow$ notatki.

- Efektywnie obliczane sprzętowo.
- Stosowane w warstwie łącza danych
  - Przykładowo Ethernet definiuje konkretny wielomian stopnia 32.

- Stosowane wielomiany stopnia n wykrywają najczęściej:
  - pojedyncze błędy bitów,
  - nieparzystą liczbę pojedynczych błędów bitów,
  - dwa błędy bitów oddalonych o co najwyżej $2^n - 1$
  - przekłamania ciągu bitów nie dłuższego od n.

Kody CRC (*Cyclic Redundancy Check*)

- $\rightarrow$ notatki.

- Efektywnie obliczane sprzętowo.
- Stosowane w warstwie łącza danych
  - Przykładowo Ethernet definiuje konkretny wielomian stopnia 32.

- Stosowane wielomiany stopnia n wykrywają najczęściej:
  - pojedyncze błędy bitów,
  - nieparzystą liczbę pojedynczych błędów bitów,
  - dwa błędy bitów oddalonych o co najwyżej $2^n - 1$
  - przekłamania ciągu bitów nie dłuższego od n.

Kody MAC (*Message Authentication Code*)

- Kod uwierzytelniający.
- Cel: zapewnienie *integralności* wiadomości: trudno ją zmodyfikować tak, żeby uzyskać taki sam MAC.

- Kryptograficzne funkcje haszujące
  - Funkcja h: funkcja haszująca, szybko obliczalna,
  - h: ciąg bitów dowolnej długości $\rightarrow$ ciąg bitów długości m.
  - Przykładowo dla MD5 $m = 160$, dla SHA-2 $m = 256$.
  - Dla dowolnego x znalezienie y, takiego że $h(x) = h(y)$ jest obliczeniowo trudne.

- Stosowane w warstwie aplikacji (np. komunikaty w OSPF).

Kody MAC (*Message Authentication Code*)

- Sama kryptograficzna funkcja haszująca h nie wystarcza do uwierzytelnienia nadawcy.
  - Atakujący wysłałby inny komunikat z poprawnie obliczonym MAC.
- sekret = sekret znany nadawcy i odbiorcy.
- MAC = h(wiadomość # sekret)

Korekcja błędów

Jak korygować błędy w transmisji?

- Kody detekcyjne + mechanizmy ARQ (wysyłania do skutku).
- Kody korekcyjne.

Kody (ogólnie)

- (α, β)-kod: zamienia wiadomość długości β na kod o długości $\alpha \geq \beta$.
  - Przykładowo: bit parzystości dla ciągów 7-bitowych to $(8, 7)$-kod.
- Narzut kodu to α/β.

- Odległość Hamminga dwóch kodów = minimalna liczba bitów, które musimy zmienić, żeby zmienić jeden kod w drugi.

Odległość Hamminga

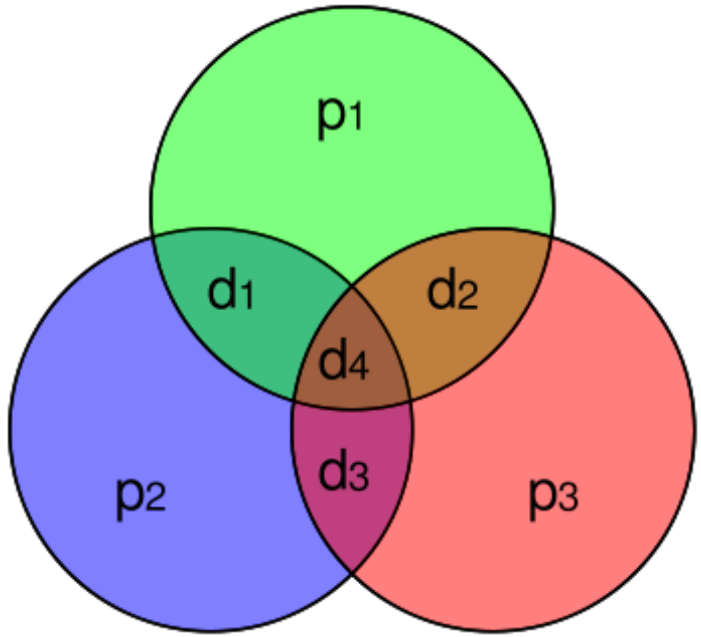
Kodowanie gwarantujące, że odległość Hamminga między dowolną parą kodów to co najmniej k.

- Potrafi wykryć $k-1$ błędów pojedynczych bitów.
- Potrafi skorygować do $\lfloor (k-1)/2 \rfloor$ błędów pojedynczych bitów.
  - Dlaczego?
- Naiwne kodowanie: $(3,1)$-kod (każdy bit powtarzamy 3 razy)
  - Koryguje przekłamanie jednego bitu.

Kodowanie Hamminga(7,4)

- 4 bity danych (d_1, d_2, d_3, d_4)
- 3 bity parzystości $(p_1, p_2, p_3$, każdy dla innych 3 bitów danych).

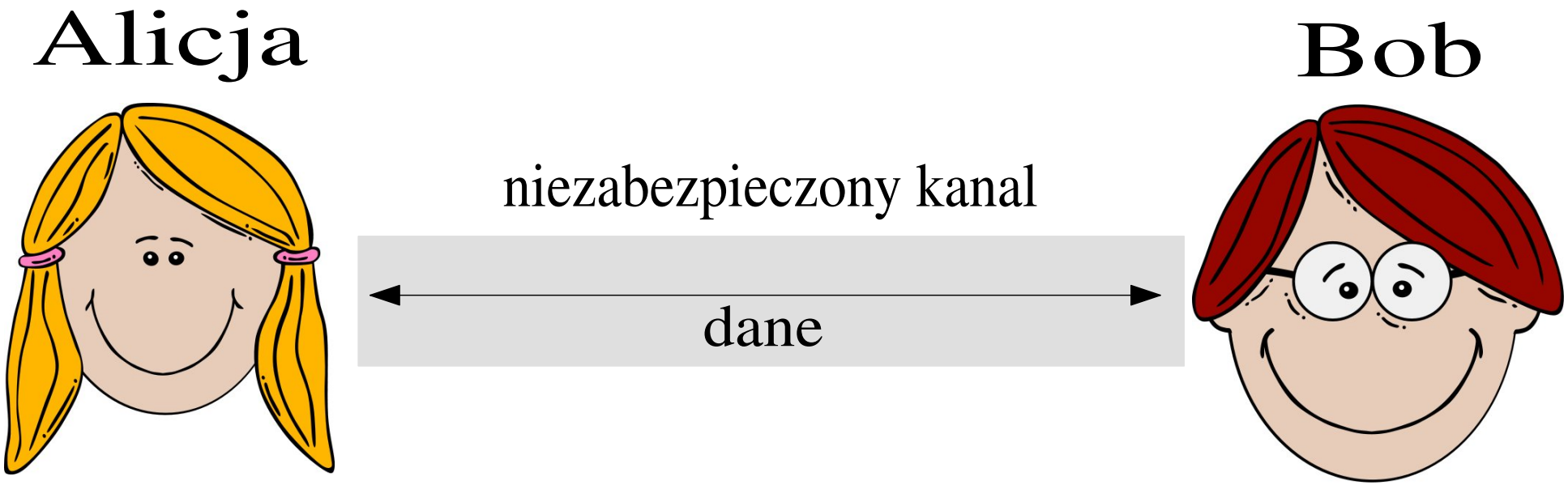
- Odległość Hamminga między dowolnymi dwoma kodami ≥ 3
  - Potrafi skorygować 1 bit.
  - Znacznie wyższa efektywność niż $(3, 1)$-kod.



Obrazek ze strony `https://en.wikipedia.org/wiki/Hamming(7,4)`

Szyfrowanie

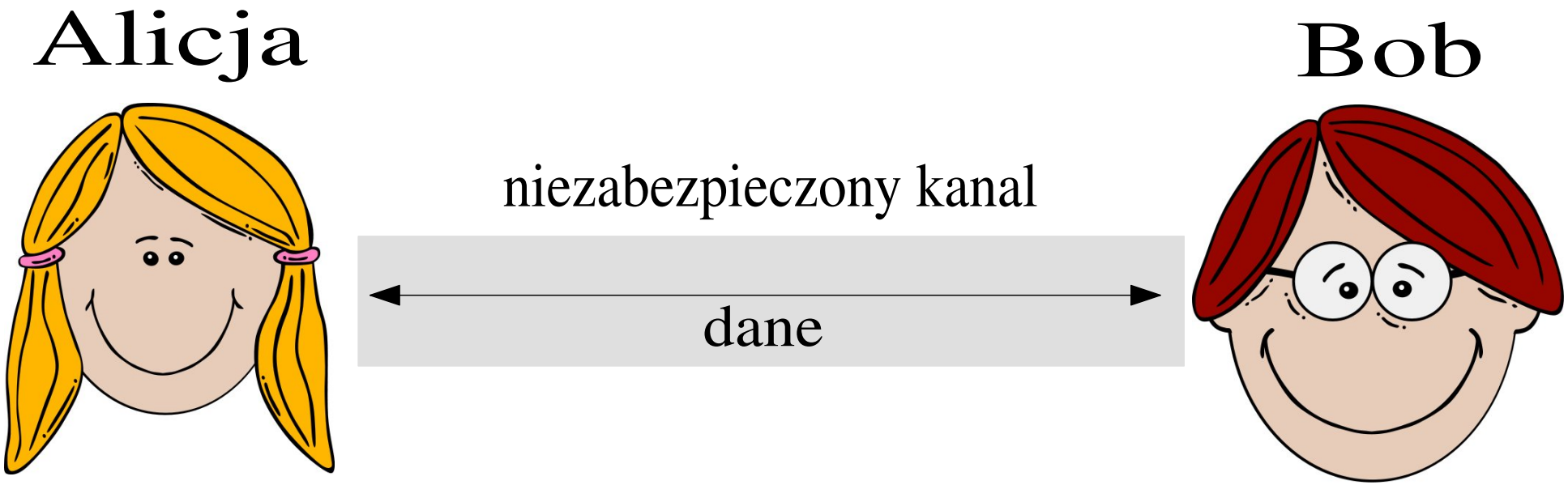
Co to właściwie jest bezpieczna komunikacja?



Pożądane cechy:

- Poufność (tylko Alicja i Bob wiedzą, co jest przesyłane).
- Uwierzytelnianie (potwierdzanie tożsamości partnera).
- Integralność (wykrywanie (złośliwych) zmian w przesyłanej wiadomości).

Co to właściwie jest bezpieczna komunikacja?



Pożądane cechy:

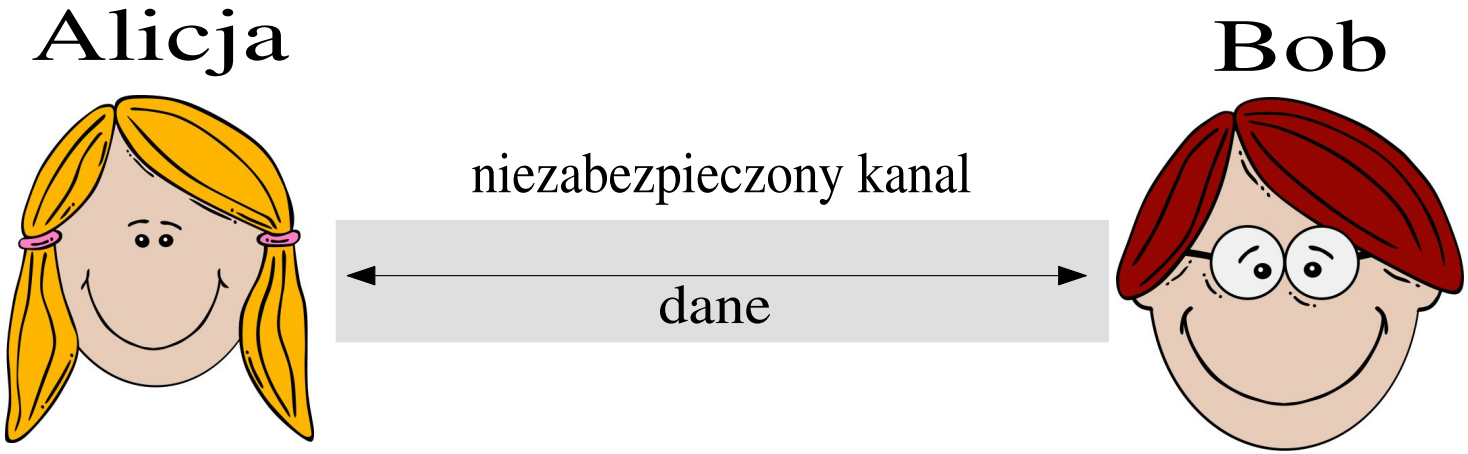
- Poufność (tylko Alicja i Bob wiedzą, co jest przesyłane).
- Uwierzytelnianie (potwierdzanie tożsamości partnera).
- Integralność (wykrywanie (złośliwych) zmian w przesyłanej wiadomości).

Alicja i Bob?

Posługujemy się przykładem Alicji i Boba. Reprezentuje to:

- komunikację między dwoma osobami
- komunikację między fizyczną osobą a serwerem/usługą (np. bankiem)
- komunikację między dwiema usługami (np. wymieniającymi tablice routingu)

Jak osiągnąć poufność?



Szyfrować!

- Alicja ma do wysłania *tekst jawny* m.
- Alicja oblicza i wysyła *szyfrogram* $s = E(m)$.
- Bob zna funkcję $D = E^{-1}$ i oblicza: $D(s) = E^{-1}(E(m)) = m$.

Szyfry monoalfabetyczne (podstawieniowe)

Szyfry monoalfabetyczne

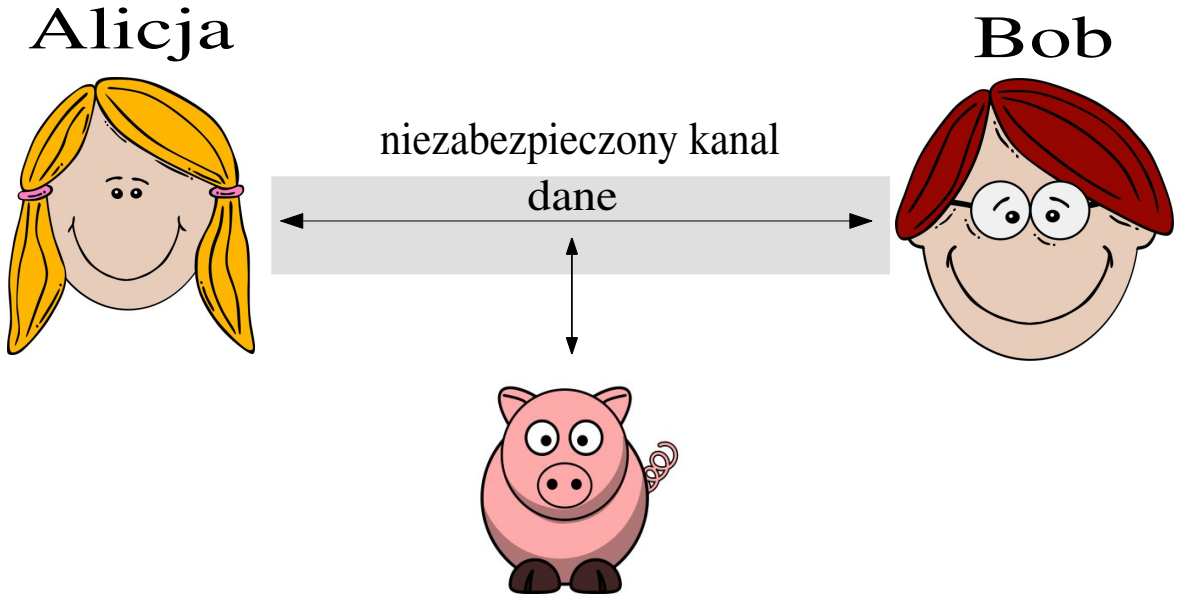
- Funkcja E operuje na pojedynczych literach, przykładowo E zmienia literę `a` na `d`, `b` na `h` itd.
- Stosowane już w czasach Juliusza Cezara (wtedy $E(a) = (a + 3) \mod 26$).
- Jak adwersarz może złamać taki szyfr?

Szyfry monoalfabetyczne (podstawieniowe)

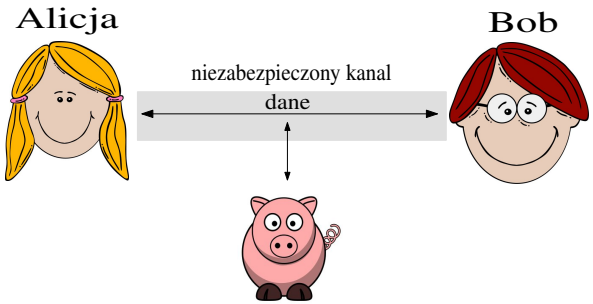
Szyfry monoalfabetyczne

- Funkcja E operuje na pojedynczych literach, przykładowo E zmienia literę `a` na `d`, `b` na `h` itd.
- Stosowane już w czasach Juliusza Cezara (wtedy $E(a) = (a + 3) \mod 26$).
- Jak adwersarz może złamać taki szyfr?

To zależy od tego, co adwersarz (czyli świnia) potrafi!



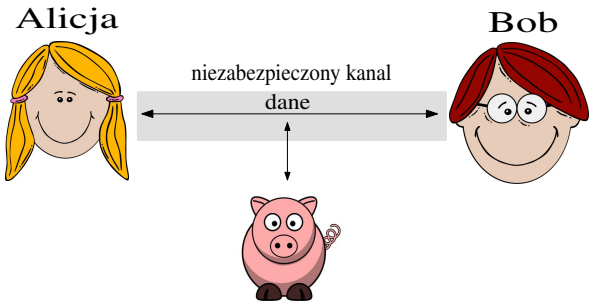
Szyfry monoalfabetyczne, cd.



Jak zgadnąć E (typy ataków):

- *Atak z wybranym tekstem jawnym*: świnia potrafi zmusić Alicję, żeby wysłała wybrany przez świnię tekst. Przykładowo: „`Pchnąć w tę łódź jeża lub ośm skrzyń fig`”.
- *Atak ze znanym tekstem jawnym*: świnia potrafi podglądnąć kilka par (*tekst jawny*, *szyfrogram*).
- *Atak ze znanym szyfrogramem*: Świnia ma tylko dostęp do kanału, widzi szyfrogramy $\rightarrow$ analiza statystyczna.

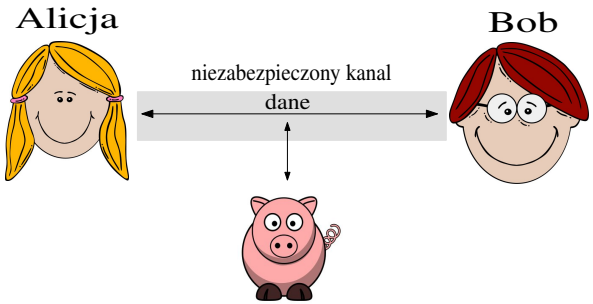
Szyfry monoalfabetyczne, cd.



Jak zgadnąć *E* (typy ataków):

- *Atak z wybranym tekstem jawnym*: świnia potrafi zmusić Alicję, żeby wysłała wybrany przez świnię tekst. Przykładowo: „`Pchnąć w tę łódź jeża lub ośm skrzyń fig`”.
- *Atak ze znanym tekstem jawnym*: świnia potrafi podglądnąć kilka par (*tekst jawny*, *szyfrogram*).
- *Atak ze znanym szyfrogramem*: Świnia ma tylko dostęp do kanału, widzi szyfrogramy $\rightarrow$ analiza statystyczna.

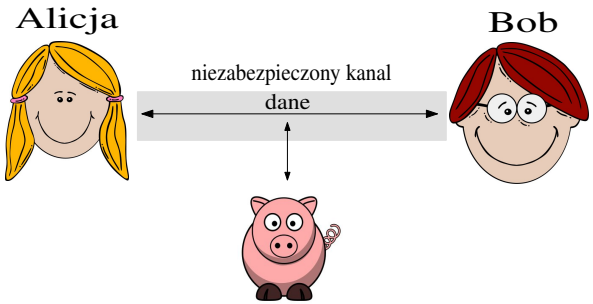
Szyfry monoalfabetyczne, cd.



Jak zgadnąć *E* (typy ataków):

- *Atak z wybranym tekstem jawnym*: świnia potrafi zmusić Alicję, żeby wysłała wybrany przez świnię tekst. Przykładowo: „`Pchnąć w tę łódź jeża lub ośm skrzyń fig`”.
- *Atak ze znanym tekstem jawnym*: świnia potrafi podglądnąć kilka par (*tekst jawny*, *szyfrogram*).
- *Atak ze znanym szyfrogramem*: Świnia ma tylko dostęp do kanału, widzi szyfrogramy $\rightarrow$ analiza statystyczna.

Szyfry monoalfabetyczne, cd.



W każdym przypadku:
te szyfry są trywialne do złamania.

Główne zastosowanie praktyczne: ROT-13.

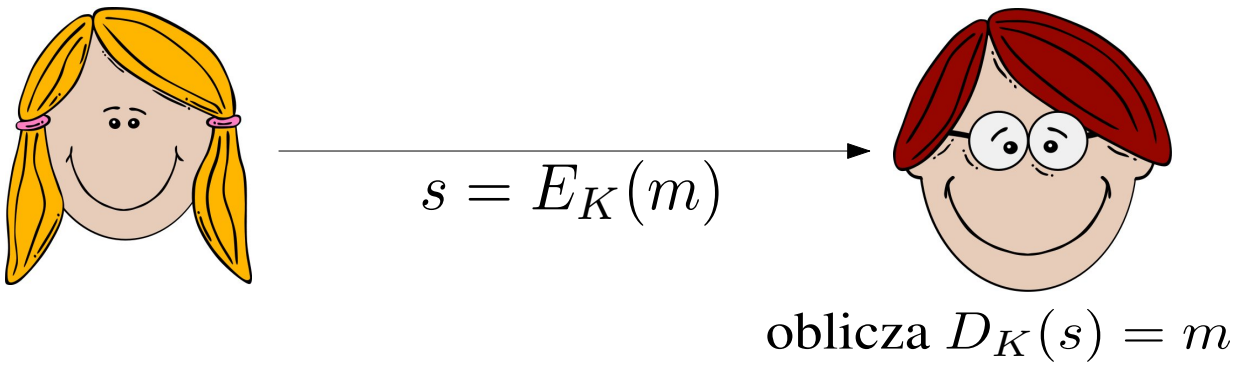
Jak zgadnąć *E* (typy ataków):

- *Atak z wybranym tekstem jawnym*: świnia potrafi zmusić Alicję, żeby wysłała wybrany przez świnię tekst. Przykładowo: „`Pchnąć w tę łódź jeża lub ośm skrzyń fig`”.
- *Atak ze znanym tekstem jawnym*: świnia potrafi podglądnąć kilka par (*tekst jawny*, *szyfrogram*).
- *Atak ze znanym szyfrogramem*: Świnia ma tylko dostęp do kanału, widzi szyfrogramy $\rightarrow$ analiza statystyczna.

Szyfrowanie symetryczne

Szyfrowanie symetryczne

- Alicja i Bob ustalają pewien wspólny klucz K.
- Szyfrogram $E_K(m)$ jest funkcją tekstu jawnego m i klucza K.
- Algorytm obliczający E (np. DES lub AES) jest *znany* wszystkim!
- Istnieje funkcja deszyfrująca $D = E^{-1}$ korzystająca z klucza, taka że $D_K(E_K(m)) = m$.
- **Symetryczność = ten sam klucz jest używany do szyfrowania i deszyfrowania**



Najprostsze szyfrowanie symetryczne

One-Time Pad

Szyfrowanie z kluczem symetrycznym. $E_K(m) = m \text{ xor } K$
(klucz musi być tak samo długi, jak tekst jawny)

Jak bezpieczne jest to szyfrowanie?

- Matematycznie: na podstawie samego szyfrogramu niemożliwy do złamania (nie dostajemy *żadnej* informacji poza długością tekstu)
- Ale: trywialne odzyskiwanie klucza jeśli znamy tekst jawny!

Szyfrowanie symetryczne, cd.

Szyfrowanie symetryczne, cd.

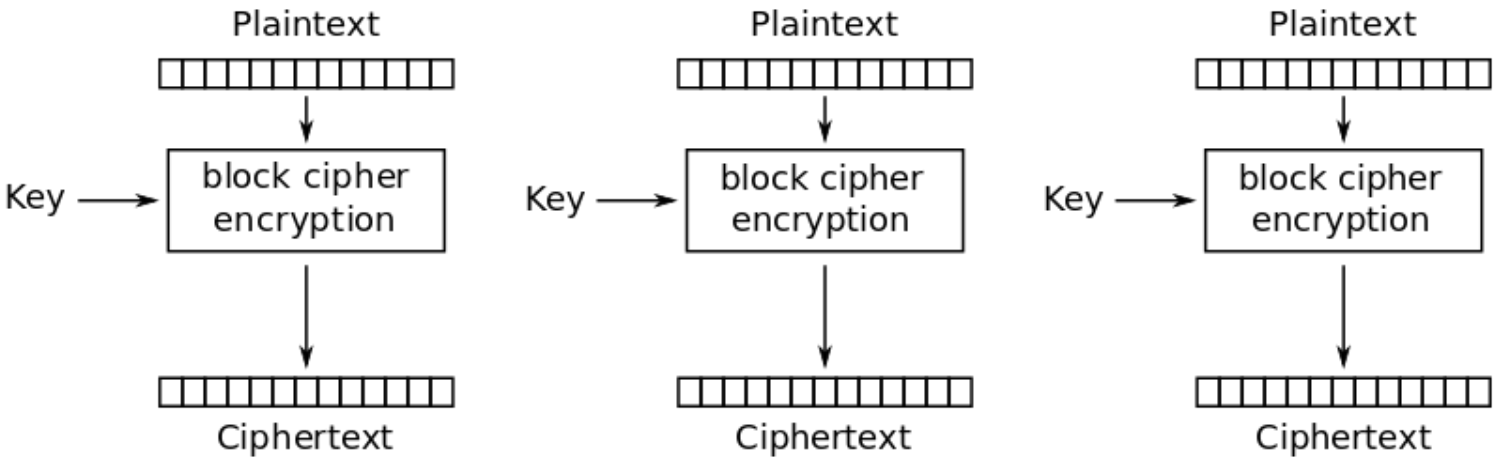
- Algorytm E to zazwyczaj złożenie wielu odwracalnych operacji bitowych (xor z częściami klucza, przesunięcia itp.)
- Algorytm D to te odwrotności tych operacji wykonane w odwrotnej kolejności.
- Funkcje E i D są szybko obliczalne.
- Siła kryptograficzna algorytmu zależy głównie od długości klucza (56 bitów w przypadku DES, 128–256 dla AES).

Długość klucza vs. długość wiadomości

- Algorytm szyfrowania symetrycznego zazwyczaj zakłada, że szyfrowana wiadomość ma określoną długość (DES: 64 bity, AES: 128 bitów).
- Wiadomość dzielona na bloki takiego rozmiaru.
  - Ostatni kawałek wiadomości: dopełniany do długości bloku.
  - Jak rozpoznać gdzie zaczyna się wypełnienie?

Wiele bloków (ECB)

- **Każdy blok szyfrowany niezależnie (tym samym kluczem).**



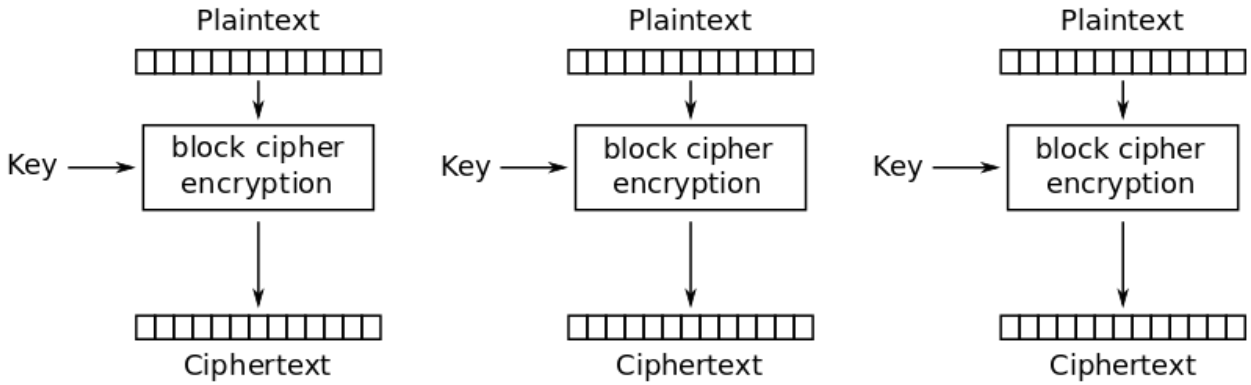
Electronic Codebook (ECB) mode encryption

- Problem: Takie same bloki zostaną zaszyfrowane na takie same kawałki szyfrogramu.

Obrazek ze strony `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

Wiele bloków (ECB + losowość)

- Przed zaszyfrowaniem bloku m_i wylosuj r_i (takie że $|r_i| = |m_i|$).
- Każdy blok szyfrowany niezależnie (tym samym kluczem):
- i-ty kawałek szyfrogramu to $c_i = E_K(m_i \text{ xor } r_i)$.



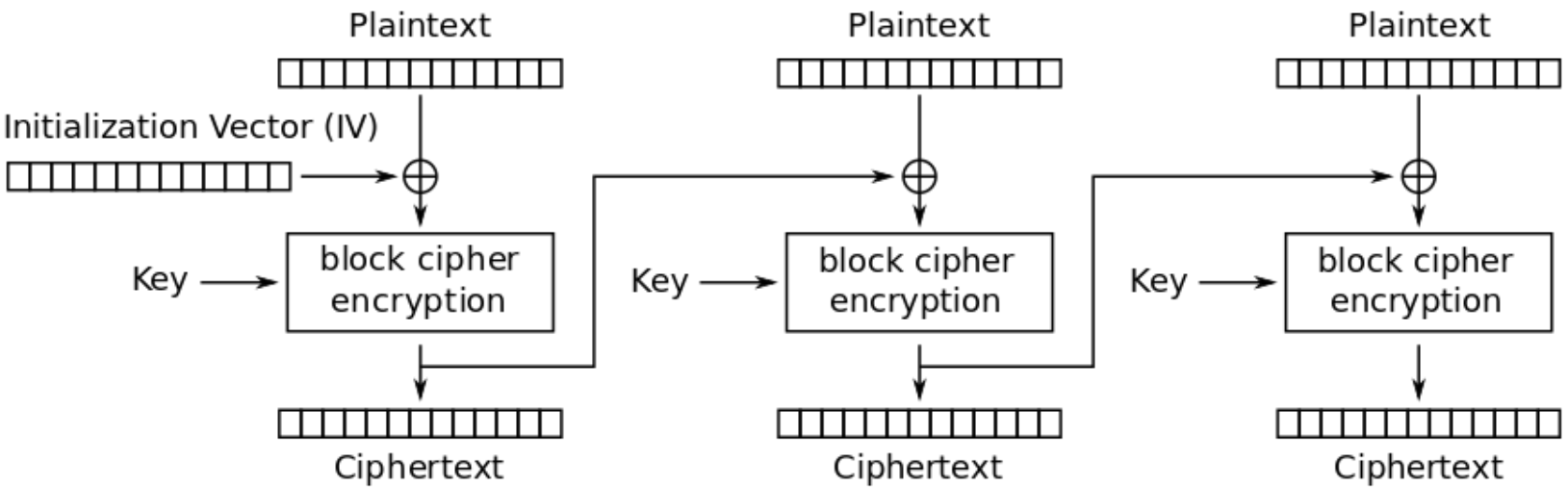
Electronic Codebook (ECB) mode encryption

- Wyślij szyfrogram i wszystkie r_i.
- Problem: dwukrotne zwiększenie wysyłanej wiadomości.

Obrazek ze strony `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

Wiele bloków (CBC + losowość)

- Wylosuj tylko r_1 (IV = wektor inicjujący).
- Pierwszy kawałek szyfrogramu to $c_1 = E_K(b_1 \text{ xor } r_1)$.
- i-ty kawałek szyfrogramu to $c_i = E_K(b_i \text{ xor } c_{i-1})$
- Wyślij szyfrogram i IV.



Cipher Block Chaining (CBC) mode encryption

Obrazek ze strony `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

Szyfrowanie symetryczne

Główny problem: jak ustalić wspólny klucz K?

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

Lepiej zastosować inne podejście: szyfrowanie asymetryczne (do przesyłania klucza lub całej wiadomości)

Szyfrowanie symetryczne

Główny problem: jak ustalić wspólny klucz K?

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

Lepiej zastosować inne podejście: szyfrowanie asymetryczne
(do przesyłania klucza lub całej wiadomości)

Lektura dodatkowa

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