Podstawy kryptografii

Andrzej M. Borzyszkowski

Instytut Informatyki
Uniwersytet Gdański

sem. letni 2023/2024

inf.ug.edu.pl/~amb/

Szyfr doskonale bezpieczny szyfry strumieniowe

System doskonale bezpieczny

- Przestrzeń wiadomości M, kluczy K i kryptogramów C
- Funkcje szyfrowania i odszyfrowywania
  – $Enc : M \times K \to C$, być może niedeterministyczna
  – $Dec : C \times K \to M$, $Dec(Enc(m,k),k) = m$ deterministycznie
  – na pewno $m_1 \neq m_2$ implikuje $Enc(m_1,k) \neq Enc(m_2,k)$
- Rozkłady prawdopodobieństwa:
  – $Pr[m = m_0], m_0 \in M, Pr[k = k_0], k_0 \in K$, niezależne rozkłady
- Def.: System jest doskonale bezpieczny, jeśli
  – $Pr[m = m_0] = Pr[m = m_0 | c = c_0], c_0 \in C$
  – tzn. po poznaniu kryptogramu wiedza na temat tekstu jawnego jest taka sama jak przedtem
  – inaczej, zdarzenia $m = m_0$ oraz $c = c_0$ są niezależne
- problemik techniczny: $Pr[c = c_0] > 0$

Własności systemu doskonale bezpiecznego

- Wniosek 1:
  – system jest doskonale bezpieczny wtt.
  $Pr[c = c_0 | m = m_0] = Pr[c = c_0]$
  – w kryptografii asymetrycznej jest wprost przeciwnie
- Wniosek 2:
  – system jest doskonale bezpieczny wtt.
  $Pr[c = c_0 | m = m_0] = Pr[c = c_0 | m = m_1]$ dla $m_0, m_1 \in M$
- Wersja z grą:
  – Ewa wybiera dwa teksty jawne: m_0, m_1
  – Tadeusz losuje klucz k, losuje bit $b \in \{0,1\}$, ogłasza $Enc(m_b, k)$
  – Ewa stara się odgadnąć bit b, wygrywa jeśli odgadnie
  – system jest doskonale bezpieczny wtt. Ewa odnosi sukces z prawdopodobieństwem $\frac{1}{2}$

Szyfr jednorazowy

Vernam, Mauborgne 1918:

```
    q w e r t y u i o p a s d f g h j k l z x c v b n m
+ p r z y k l a d t e k s t d o s z y f r o w a n i a
= f n d p d j u l h t k k w i u z i i q q l y v o v m

    f n d p d j u l h t k k w i u z i i q q l y v o v m
- q w e r t y u i o p a s d f g h j k l z x c v b n m
= p r z y k l a d t e k s t d o s z y f r o w a n i a

    f n d p d j u l h t k k w i u z i i q q l y v o v m
- w y k l a d e l e m e n t y k r y p t o g r a f i i
= j p t e d g q a d h g x d k k i k t x c f h v j n e
```

– dowolny tekst (długości kryptogramu!) może być tekstem jawnym
– tyle to wiadomo, nie znając kryptogramu
– jedyny szyfr doskonale bezpieczny

Szyfr jednorazowy, analiza

- $c = m \oplus k$, gdzie $m, k, c \in \{0,1\}^\ell$, tzn. ciągi bitów długości ℓ
  – $Pr[c = c_0 | m = m_0] = Pr[m \oplus k = c_0 | m = m_0] = Pr[m_0 \oplus k = c_0] = Pr[k = m_0 \oplus c_0] = (\frac{1}{2})^\ell$
  – zakładając jednostajny rozkład klucza otrzymujemy warunek doskonałego bezpieczeństwa
- Klucz użyty powtórnie oznacza umożliwia analizę częstotliwości
  – jeśli $c = m \oplus k$ oraz $c' = m' \oplus k$ (ten sam klucz k)
  – to $c \oplus c' = m \oplus m'$
  – nie znając tekstów jawnych znamy ich $\oplus$ – umożliwia to analizę podobną do analizy szyfru Vigenere'a
- jeśli szyfr jest doskonale bezpieczny, to moc $K \geqslant |M|$
- tw. Shannon'a: szyfr jest doskonale bezpieczny wtt
  – rozkład wartości klucza jest jednostajny
  – relacja $Dec \subseteq C \times K \times M$ jest funkcyjna w każdym argumencie

Złożoność obliczeniowa

- System doskonale bezpieczny (*perfect security*):
  – klucz musi być co najmniej tak długi jak tekst jawny
  – i użyty tylko jeden raz
  – założenia zbyt ograniczające praktyczne zastosowania
- Bezpieczeństwo obliczeniowe (*computational security*):
  – system musi być bezpieczny wobec obliczeń możliwych do wykonania w rozsądnym czasie
  – bezpieczny z prawdopodobieństwem bliskim 1
  – tzn. dopuszczamy możliwość złamania szyfru przypadkiem albo wskutek długich obliczeń
- Założenia: złożoność szyfru mierzona jest parametrem n, przeciwnik działa w czasie wielomianowym ($C \cdot n^c$ dla pewnych C i c)
  – system jest bezpieczny jeśli prawdopodobieństwo złamania jest mniejsze niż $\frac{1}{c^n}$ dla dowolnego c

Teoria złożoności

- Algorytm działa w czasie wielomianowym (na wejściu x)
  – wynik jest wyprodukowany po $p(|x|)$ krokach, p – pewien wielomian
- Algorytm probabilistyczny działa w czasie wielomianowym
  – jeśli dodatkowo odczytuje co najwyżej $p(|x|)$ losowych bitów
  – oznaczenie PPT
- Obie klasy algorytmów są zamknięte na złożenia i inne operacje
- Funkcja f jest zaniedbywalnie mała
  – jeśli
  $$\forall p.\ \exists N \in \mathbb{N}.\ \forall n > N.\ f(n) < \frac{1}{p(n)}$$
  – np. 2^{-n}, $2^{-\sqrt{n}}$, $n^{-\log n}$
- Funkcje zaniedbywalnie małe są zamknięte na dodawanie i mnożenie przez wielomian

Dwa rodzaje ataków

- $Dec : K \times C \to M$
  – $|K| \leqslant |M| \approx |C|$, dużo mniejsza przestrzeń kluczy
- Atak brutalny
  – zastosowanie każdego możliwego klucza
  – $Dec(K, c_0) \subseteq M$
  – albo atak z parą $\langle c_0, m_0 \rangle$ i wówczas znajdujemy $Dec(k_0, c_0) = m_0$
  – albo trzeba jakoś wiedzieć, że k_0 jest kluczem
  – prawdopodobieństwo znalezienia klucza $= 1$
- Zgadywanie
  – testujemy tylko jeden klucz, $Dec(k_0, c_0)$
  – prawdopodobieństwo znalezienia $= \frac{1}{|K|}$
- Czyli przestrzeń klucza musi być ponadwielomianowa dla parametru szyfru

Definicja bezpieczeństwa szyfru

- Doskonałe bezpieczeństwo
  – Ewa wybiera dwa teksty jawne: m_0, m_1
  – Tadeusz losuje klucz k, losuje bit $b \in \{0, 1\}$, ogłasza $Enc(k, m_b)$
  – Ewa stara się odgadnąć bit b, wygrywa jeśli odgadnie
  – system jest doskonale bezpieczny wtt. Ewa odnosi sukces z prawdopodobieństwem $\frac{1}{2}$
- Bezpieczeństwo obliczeniowe (atak tylko z kryptogramem)
  – parametr n
  – Ewa używa algorytmu PPT
  – teksty jawne mają tę samą długość (wielomianową od n)
  – klucz jest losowany dla tego parametru n
  – system jest bezpieczny, jeśli prawdopodobieństwo sukcesu Ewy $< \frac{1}{2} +$ zaniedbywalnie mała

Własności bezpieczeństwa obliczeniowego

- Żaden bit tekstu jawnego nie może być odgadnięty z prawdopodobieństwem znacząco większym niż $\frac{1}{2}$
  – gdyby mógł być, to dwa teksty jawne różniące się odgadywanym bitem pozwoliłyby Ewie odgadnąć tekst jawny
- Semantyczne bezpieczeństwo (obliczeniowe):

$$\forall A \in PPT. \exists A' \in PPT. \forall h(m), f(m) \in PT.$$

$$Pr[A(n, Enc(k, m), h(m)) = f(m)] - Pr[A'(n, h(m)) = f(m)]$$

jest zaniedbywalnie mała
  – $h(m)$ oznacza „wiedzę" na temat tekstu jawnego
  – $f(m)$ jest własnością, którą chcemy zbadać
  – algorytm A korzysta z kryptogramu i tekstu jawnego
  – algorytm A' nie korzysta z kryptogramu
  – i daje nie gorsze wyniki
  – czyli znajomość kryptogramu nie wnosi żadnej wiedzy
- Tw.: powyższe definicje są równoważne

Liczby losowe i pseudolosowe

- Liczby losowe: obserwacja zjawisk przyrody
- Liczby pseudolosowe: dany ciąg s (seed, ziarno) długości n, zadanie: wygenerować ciąg $G(s)$ długości $\ell(n) > n$, długość ℓ jest wielomianowa
  – jeśli ciąg s jest losowy, to $G(s)$ „wygląda jak losowy"
  – ciągów długości n jest 2^n, ciągów dłuższych jest więcej, nie wszystkie (b. niewiele) będzie wygenerowanych
  – dla danego ciągu r długości $\ell(n)$ można sprawdzić czy jest ciągiem wygenerowanym (brutalne przeszukiwanie)
  – ale wymaga to ponadwielomianowych zasobów

Pseudolosowość

- G jest generatorem ciągów pseudolosowych, jeśli
  – $\forall D \in PPT.\ Pr[D(r) = 1] - Pr[D(G(s)) = 1]$ jest zaniedbywalnie mała
  – r jest ciągiem losowym długości $\ell(n)$
  – D bada czy ciąg jest losowy, gdyby badało wyczerpująco, to na pewno wykryje, czy ciąg jest generowany
  – ale D ma ograniczone możliwości obliczeniowe

Generatory pseudolosowe

- Generator liniowy: $x[n] = a \cdot x[n-1] \oplus b \mod k$
  – wystarczający do testowania programów itp.
  – bezwartościowy dla wielu zastosowań w kryptografii (jest przewidywalny)
- Funkcje nieodwracalne: łatwo wyliczyć $f(x)$ ale trudno odwrócić
  – s ustalone, $x[n] = f(s+n)$, $b[n] =$ ostatni bit $x[n]$
  – np. algorytm DES albo SHA
- Generator Blum-Blum-Shub: $x[n] = x[n-1] \cdot x[n-1] \mod k$, $b[n] =$ ostatni bit $x[n]$

Przesuwane rejestry

- LFSR (linear feedback shift register)
- generator $b[n+5] = b[n] \oplus b[n+2]$ ma okres 31
  1 1 0 1 1 1 0 1 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 1 1 1 1 1 0 0 0 1 1 0 1 1
  – ale $b[n+31] = b[n] \oplus b[n+3]$ ma okres $2^{31} - 1 = 2 \cdot 10^9$
  – tzn. 31 bitów wyznacza ciąg pseudolosowy 2 Gb
- łatwa implementacja jako rejestr 32 bitowy
- kryptoanaliza: sam szyfrogram nic nie daje
  – para tekst jawny+zaszyfrowany daje natychmiast fragment klucza $x_1, x_2, \ldots$
  – wówczas można próbować rozwiązać równanie $AC = B$ z niewiadomą C, gdzie $A[i,j] = b[i+j+1]$, $B[l] = b[m+l+1]$, najpierw trzeba zgadnąć wymiar m
  – tw. Jeśli A_k jest macierzą dla wymiaru k, to $det(A_m) = 1$ dla prawidłowego wymiaru i $det(A_k) = 0$ dla $k > m$

Przesuwane rejestry, ulepszenie

- generator redukujący
  - generujemy dwa ciągi bitów, $a[i]$ oraz $b[i]$
  - i-ty bit generowany =
    – znaleźć i-tą jedynkę w b, jej numer jest j
    – wynikiem jest $a[j]$
  - nie ma dowodu, ale wydaje się, że jest to niezły generator
- generator kombinacji
  - ustalamy funkcję nieliniową $f : \{0,1\}^k \to \{0,1\}$
  - używamy k generatorów liniowych, ostatecznym wyjściem jest wynik tej funkcji, tzn. $a[i] = f(a_1[i], \ldots, a_k[i])$
- generator filtrów: zamiast kilku generatorów używany jest jeden z kolejnymi bitami: $b[i] = f(a[i \cdot k + 1], \ldots, a[i \cdot k + k])$
  – następny bit jest generowany po k rundach

Generatory pseudolosowe c.d.

- Formalnie, brak dowodu istnienia jakiegokolwiek generatora
  – brak dowodów, że pewne operacje nie dadzą się wykonać w czasie wielomianowym
  – istnieją problemy, które są wystarczająco trudne przy obecnym stanie wiedzy
- Zakładamy, że generatory pseudolosowe istnieją
  – będą one skutecznie zbudowane przy założeniu, że pewne problemy są trudne

Szyfr strumieniowy

- Idea jak z szyfru jednorazowego
  – generowanie klucza: losowy długości n,
  – szyfrowanie: $Enc(k, m) = G(k) \oplus m$ dla m długości $\ell(n)$
  – odszyfrowanie: $Dec(k, c) = G(k) \oplus c$
- Na pewno nie ma doskonałego bezpieczeństwa
  – klucz krótszy od tekstu jawnego (tw. Shannona)
- Jeśli generator G jest pseudolosowy, to szyfr jest obliczeniowo bezpieczny
  – dw.: gdyby nie był, to Ewa odróżniłaby ciąg pseudolosowy od naprawdę losowego
- Generatory w praktyce
  – RC4: jest niezły, ma pewne ograniczenia (pierwsze bity)
  – generatory linowe z przesuwanym rejestrem: w zasadzie nie spełniają warunków bezpieczeństwa

Szyfrowanie wielu wiadomości

- Ewa przygotowuje zestaw par tekstów jawnych
  – jeden z zestawów jest zaszyfrowany losowym kluczem
  – Ewa ma zgadnąć, który zestaw
  – system jest bezpieczny dla szyfrowania wielokrotnego, jeśli prawdopodob. sukcesu $< \frac{1}{2} +$ zaniedbywalnie mała
- Twierdzenie: jeśli szyfr jest bezpieczny dla szyfrowania wielokrotnego, to szyfrowanie musi być niedeterministyczne
  – dw.: Ewa przygotowuje $\langle m_0, m_0 \rangle$ oraz $\langle m_0, m_1 \rangle$
  – i widzi, czy kryptogramy są równe, czy różne
- Szyfr strumieniowy
  – znajomość $G(k) \oplus m_0$ oraz $G(k) \oplus m_1$ pozwala obliczyć $m_0 \oplus m_1$
  – to tak jak dwukrotne użycie szyfru jednorazowego
  – *The Misuse of RC4 in Microsoft Word and Excel*
  `http://eprint.iacr.org/2005/007`

Szyfry strumieniowe dla wielokrotnego szyfrowania

- Tryb synchronizacji
  – każde użycie funkcji szyfrującej zaznacza zużytą część ciągu pseudolosowego
  – nigdy dwa teksty nie będą szyfrowane tym samym ciągiem
  – strony protokołu muszą pamiętać stan generatora pseudolosowego
- Szyfrowanie jest bezpieczne i jest deterministyczne
  – ale nie podpada pod ogólny schemat szyfrowania
  – funkcja szyfrująca zależy również od „stanu rejestru”

Szyfry strumieniowe dla wielokrotnego szyfrowania c.d.

- Tryb asynchroniczny
  – generator korzysta z dodatkowego wektora inicjalizacji
  – ciąg pseudolosowy $G(s, IV)$, spełnia własności dla łącznego rozpatrywania, tzn. jako funkcji pary $\langle s, IV \rangle$
  – funkcja szyfrowania: $Enc(k, m) = \langle IV, G(k, IV) \oplus m \rangle$
  – funkcja odszyfrowania: $Dec(k, \langle IV, c \rangle) = G(k, IV) \oplus c$
  – szyfrowanie jest niedeterministyczne, każde szyfrowanie tego samego tekstu jawnego daje inny wynik
- Para $\langle s, IV \rangle$ może być uważana łącznie za ziarno
  – niektóre generatory pseudolosowe można łatwo przerobić na generator z wektorem inicjalizującym
  – niektóre taki wektor mają naturalnie obecny
  – niektóre generatory nie nadają się do wielokrotnego szyfrowania