

Kryptografia

Algorytmy niesymetryczne

dr Robert Borowiec

Politechnika Wrocławska

Instytut Telekomunikacji i Akustyki

pokój 908, C-5

tel. 3203083

e-mail: robert.borowiec@ita.pwr.wroc.pl

www: lstwww.ita.pwr.wroc.pl/~RB/

Wykład VIII

1-godzina

Przypomnienie

Obliczanie odwrotności liczby

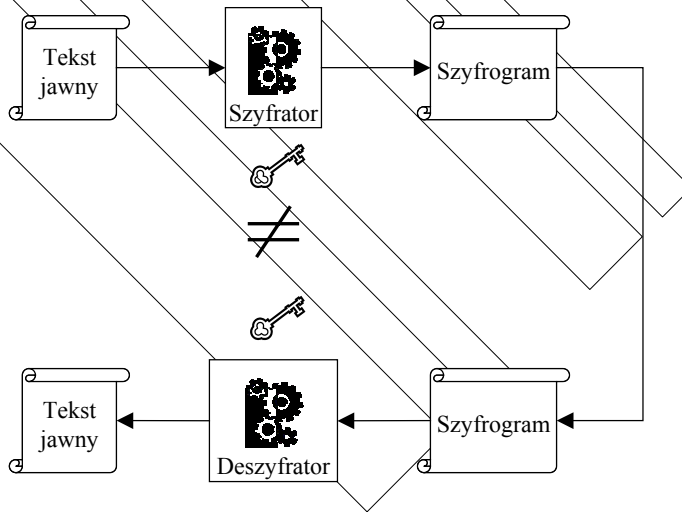
- Podane przez *Eulera* uogólnienie *Fermata* dostarcza algorytmu do rozwiązania równania $(a \cdot x) \bmod n = 1$, gdy $\text{NWW}(a, n) = 1$. Rozwiązanie to ma postać:

$$x = a^{\phi(n)-1} \bmod n$$

- Jeżeli n jest liczbą pierwszą, to:

$$x = a^{(n-1)-1} \bmod n = a^{n-2} \bmod n$$

Algorytmy niesymetryczne



Algorytmy niesymetryczne

- Algorytmy niesymetryczne charakteryzują się tym, że klucz szyfrujący różni się od klucza deszyfrującego
- Nie każdy algorytm niesymetryczny nadaje się do implementacji jako system klucza publicznego, gdyż ujawnienie jednego z kluczy pociąga za sobą możliwość znalezienia drugiego klucza
- Systemy niesymetryczne wykorzystywane są również do podpisów cyfrowych oraz procedury wymiany klucza kryptograficznego

Bezpieczeństwo algorytmów niesymetrycznych

- Bezpieczeństwo szyfrów niesymetrycznych opiera się na trudności rozwiązania jednego z trzech problemów trudnych obliczeniowo (NP-zupełnych):
 - ⇒ faktoryzacji dużych liczb
 - ⇒ obliczaniu logarytmów dyskretnych w ciele skończonym
 - ⇒ problemie plecakowym

Algorytmy niesymetryczne

- Diffiego Hellmana
- RSA Rivesta-Shamira-Adlemana
- Poligha-Hellmana
- Rabina
- Algorytmy plecakowe
- i wiele innych

Szyfry potęgowe (Hellmana, RSA)

Szyfry potęgowe dokonują szyfrowania na bloku tekstu jawnego $M \in [0, n-1]$ poprzez wykonanie odpowiednio potęgowania

$$C = M^e \bmod n \quad (1)$$

$$M = C^d \bmod n \quad (2)$$

Jeżeli $\text{NWW}(M, n) = 1$ oraz e i d spełniają równanie

$$e \cdot d \bmod \phi(n) = 1$$

to równanie (2) jest odwrotnością równania (1), a więc:

$$\begin{aligned} C = M^e \bmod n &\Rightarrow M = C^d \bmod n = (M^e \bmod n)^d \bmod n \\ &= M^{ed} \bmod n \end{aligned}$$

Szyfry potęgowe (Hellmana, RSA)

Do obliczenia d przy wybranym e , które spełniają równanie

$$e \cdot d \bmod \phi(n) = 1$$

stosowana jest zależność

$$d = e^{\phi(n)-1} \bmod \phi(n)$$

Jeżeli n jest liczbą pierwszą to równanie przybiera postać

$$d = e^{(n-1)-1} \bmod (n-1)$$

i można je rozwiązać bez znajomości funkcji Eulera za pomocą rozszerzonego algorytmu Euklidesa:

Rozszerzony algorytm Euklidesa

Rozszerzony algorytm Euklidesa służy do obliczania odwrotności multiplikatywnej w ciele skończonym.

1. Algorytm ten wyznacza liczby x i y takie, że

$$a \cdot x + n \cdot y = d, \text{ gdzie } d = \text{NWD}(a, n)$$

2. Jeżeli z obliczeń $d > 1$ to liczba $a^{-1} \bmod n$ nie istnieje. Gdy $d=1$, to x jest odwrotności liczby a .

Algorytm Diffiego-Hellmana

Algorytm nie nadaje się do szyfrowania i deszyfrowania danych. Nie jest więc algorytmem szyfrującym. Wyśmienicie nadaje się jednak do dystrybucji kluczy.

⇒ Działanie algorytmu

⇒ Bezpieczeństwo algorytmu

Szyfry potęgowe (Pohliga-Hellmana)

Szyfr nie jest szyfrem z kluczem jawnym.

Jako moduł obliczeń przy szyfrowaniu wybieramy dużą liczbę pierwszą p . Funkcję szyfrującą i deszyfrującą określają wzory:

$$C = M^e \bmod p \quad (1)$$

$$M = C^d \bmod p \quad (2)$$

Ponieważ p jest liczbą pierwszą to $\phi(p) = p - 1$. Stąd wynika, że szyfr może być użyty tylko do klasycznego szyfrowania, bo zarówno e jak d muszą być utrzymane w tajemnicy.

Ponieważ e i d są względem siebie odwrotne modulo p to łatwo jest znaleźć jedną z nich na podstawie drugiej.

Szyfry potęgowe (Pohliga-Hellmana)

Wartość p można znaleźć na podstawie wielkości bloków tekstu jawnego i szyfrogramu.

Przy ataku z tekstem jawnym kryptoanalityk mógłby obliczyć e , a potem d mając parę (M, C) .

$$e = \log_M C$$

Bezpieczeństwo algorytmu opiera się na złożoności obliczania logarytmów w ciele $GF(p)$. Czas potrzebny potrzebny do obliczenia logarytmu wynosi kilka miliardów lat dla liczby p złożonej z 200 cyfr dziesiętnych (664 bitów)

RSA

Algorytm RSA powstał 1978 r. za sprawą Rona Rivesta, Adi Shamira i Leonarda Adelfmana. Jego nazwa pochodzi od pierwszych liter ich nazwisk. Do dnia dzisiejszego nie został złamany mimo intensywnie prowadzonej nad nim kryptoanalizy.

Jest wykorzystywany do:

- szyfrowania danych w systemie klucza jawnego
- podpisów elektronicznych

Bezpieczeństwo szyfru RSA bazuje na trudności w faktoryzacji iloczynu wielkich liczb pierwszych.

RSA generowanie kluczy

Algorytm generowania kluczy :

1. Wybieramy losowo dwie duże liczby pierwsze p i q
2. Obliczamy $n = p \cdot q \Rightarrow \phi(n) = (p-1)(q-1)$
3. Losowo wybieramy klucz szyfrujący e taki, że $\text{NWW}(e, \phi(n)) = 1$
4. Używamy rozszerzonego algorytmu Euklidesa do wyznaczenia d , odwrotności liczby e

$$d \cdot e \bmod (p-1)(q-1) = 1$$

Teraz d i n są względnie pierwsze

RSA

- Liczby e i n stanowią klucz jawny, który można opublikować
- Liczba d jest kluczem tajnym.
- Liczby p i q należy zniszczyć, gdyż nie są już potrzebne, a mogły by służyć do złamania szyfru. Ich będzie poszukiwał kryptoanalityk.
- Można ujawnić e oraz n bo nie znając p i q nie można wyznaczyć funkcji Eulera $\phi(n)$, a jest ona potrzebna do wyznaczenia d .
- Bezpieczeństwo polega na trudności faktoryzacji liczby n na p i q

RSA

Szyfrowanie wiadomości

- Do szyfrowania informacja dzielona jest na bloki m_i
- Każdy blok informacji binarnej musi być krótszy od n , aby miał jednoznaczną interpretację modulo n
- Szyfrowanie każdego bloku informacji m_i zachodzi według zależności

$$c_i = m_i^e \bmod n$$

- Deszyfrowanie jest realizowane wg równania

$$m_i = c_i^d \bmod n$$

RSA

Uwagi

- W grupach roboczych nie należy używać jednakowego n nawet przy różnych wartościach e i d , ponieważ przy znajomości e_1, e_2, n oraz wartości kryptogramów C_1 i C_2 tej samej wiadomości M można ją rozszyfrować bez znajomości kluczy deszyfrujących d_1, d_2
- Należy wybierać duże wartości kluczy e i d , ale wartości te powinny nie być zbyt bliskie sobie (tzn. muszą wystąpić różnice przynajmniej na kilku pozycjach). Szyfr RSA jest łatwo przełamalny jeżeli d lub $e < 0,25 n$

RSA

Właściwości

1. Odzyskanie jednego bitu informacji jest tak samo trudne jak odzyskanie całej informacji.
2. Na początku lat 90 uważano, że dopiero liczby o długości 1024 bity będą odporne na faktoryzację przez najbliższe 10 lat
3. Szybkość RSA w implementacji sprzętowej jest wolniejszy ok. 1000 razy od algorytmu DES. W implementacji programowej już tylko 100 razy
4. Algorytm jest odporny na kryptoanalizę z wybranym szyfrogramem

Algorytm Rabina

Bezpieczeństwo algorytmu Rabina opiera się na trudności pierwiastkowania modulo liczba skończona

Algorytm generowania kluczy :

1. Wybieramy losowo dwie duże liczby pierwsze p i q o zbliżonej długości kongruentne do 3 modulo 4.
2. Obliczamy $n = p \cdot q$
3. Liczba n jest kluczem jawnym, a para liczb p i q kluczem prywatnym

Algorytm Rabina

Szyfrowanie :

1. Dzielimy wiadomość jawną na bloki, których reprezentacja liczbowa jest mniejsza od liczby n . Czyli blok wiadomości m_i musi być liczbą z przedziału $\{0, 1, \dots, n-1\}$.
2. Wyliczana jest wartość bloku szyfrogramu jako:

$$c_i = m_i^2 \bmod n$$

Deszyfrowanie :

1. Wyznaczamy pierwiastki kwadratowe m_1, m_2, m_3, m_4 z $c_i \bmod n$ korzystając z Chińskiego twierdzenia o resztach.
2. Jeden z pierwiastków jest tekstem jawnym, a pozostałe należy odrzucić. Dla tekstów pisanych jest to łatwe. Gorzej gdy użyjemy algorytmu Rabina do przesłania np. klucza sesyjnego

Algorytm Rabina

Obliczanie pierwiastków

1. Korzystając z rozszerzonego algorytmu Euklidesa wyznaczane są dwie liczby całkowite a i b spełniające równanie

$$a \cdot p + b \cdot q = 1$$

2. Obliczamy $r = c^{(p+1)/4} \bmod p$

3. Obliczamy $s = c^{(q+1)/4} \bmod p$

4. Obliczamy $x = (a \cdot p \cdot s + b \cdot q \cdot r) \bmod n$

5. Obliczamy $x = (a \cdot p \cdot s - b \cdot q \cdot r) \bmod n$

6. Cztery pierwiastki kwadratowe z liczby c modulo n to liczby:

$$x, -x \bmod n \text{ oraz } y, -y \bmod n$$

KONIEC

Dziękuję za uwagę