Algorytm RSA – podstawa podpisu elektronicznego

dr inż. WOJCIECH NOWAKOWSKI

Instytut Maszyn Matematycznych, Warszawa

Klasyczne metody szyfrowania danych cyfrowych stosowane od lat, od początku istnienia kryptografii opierały się na podstawieniach i permutacji. W szyfrowaniu konwencjonalnym informacja zostaje przekształcona na postać zaszyfrowaną za pomocą algorytmu szyfrującego oraz klucza. Różne klucze generują różne dane wyjściowe, czyli inną postać informacji zaszyfrowanej. Odbiorca może tę informację odszyfrować, ale tylko wtedy, gdy dysponuje identycznym kluczem, jaki został użyty do szyfrowania.

Wystąpił więc podstawowy problem dystrybucji kluczy. Korzystanie z szyfrowania było uwarunkowane posiadaniem przez obie strony transmisji tego samego klucza, w jakiś sposób dostarczonego, lub tworzenia central dystrybucji kluczy, co niewątpliwe utrudnia lub w ogóle uniemożliwia zachowanie tajności kluczy.

Rozwiązanie tego problemu przyniósł rozwój kryptografii klucza jawnego. Algorytmy z kluczem jawnym wykorzystują funkcje matematyczne. Jest to szyfrowanie asymetryczne, za pomocą dwóch kluczy, jednego publicznego, jawnego i drugiego prywatnego, niejawnego i chronionego, ale nie przekazywanego.

Jednym z pierwszych, a obecnie najbardziej popularnym asymetrycznym algorytmem kryptograficznym jest algorytm RSA [1]. Został opracowany w 1977 r. przez Rona Rivesta, Adi Shamira oraz Leonarda Adlemana. Jest to jednocześnie pierwszy algorytm, który można stosować zarówno do szyfrowania jak i do podpisów cyfrowych.

Klucz publiczny umożliwia jedynie zaszyfrowanie danych i w żaden sposób nie ułatwia ich odczytania, nie musi więc być chroniony. Drugi klucz – prywatny – przechowywany pod nadzorem, służy do odczytywania informacji zakodowanych za pomocą klucza publicznego. Możliwe jest także zaszyfrowanie wiadomości za pomocą klucza tajnego prywatnego, a następnie jej odszyfrowanie za pomocą klucza publicznego. To właśnie ta właściwość sprawia, że RSA może zostać wykorzystany do cyfrowego podpisywania dokumentów.

System RSA umożliwia bezpieczne przesyłanie danych w środowisku, w którym może dochodzić do różnych nadużyć. Bezpieczeństwo szyfrowania opiera się w tym algorytmie na bardzo czasochłonnej obliczeniowo faktoryzacji (znajdowaniu czynników pierwszych) dużych liczb złożonych. Najszybszym komputerom może zajmować to obecnie wiele dziesiątków lat. Sytuacja może ulec zmianie po wprowadzeniu tzw. komputerów kwantowych [2], które będą pracowały miliony razy szybciej od współczesnych, ale to dopiero przyszłość.

Opis algorytmu [3]

System RSA to szyfr blokowy, w którym tekst jawny i zaszyfrowany są liczbami całkowitymi od 0 do *n-1* dla pewnego *n*. Korzysta on z wyrażenia potęgowego. Tekst jawny jest szyfrowany blokami, z których każdy ma wartość binarną mniejszą od pewnej liczby n. Szyfrowanie dla bloku tekstu jawnego *m* i zaszyfrowanego *c* ma następującą postać:

$$c \equiv m^e \pmod{n}$$

$$m \equiv c^d \pmod{n}$$

Wartość *n* musi być znana nadawcy i odbiorcy. Nadawca zna wartość *e*, a odbiorca *d*. Klucz publiczny to *{e, n}*,a prywatny *{d, n}*.

Generowanie kluczy

Wybieramy dwie duże liczby pierwsze:*{p, q}*

Obliczamy ich iloczyn:

$$n = pq$$

oraz funkcję Eulera

$$\varphi = (p - 1)(q - 1)$$

Wybieramy losowo liczbę $e < n$, względnie pierwszą z liczbą $\varphi = (p - 1)(q - 1)$. Liczba *e* będzie kluczem szyfrującym.

Znajdujemy liczbę (korzystając z rozszerzonego algorytmu Euklidesa) *d* taką, że:

$$d \equiv e^{-1} \pmod{\varphi} \text{ lub}$$

$$de \equiv 1 \pmod{\varphi},\ d<\varphi$$

Liczby d i n są także względnie pierwsze.

Jak wspomniano liczby *{e,n}* stanowią klucz publiczny, który ujawniamy, zaś liczby *{d,n}* stanowią klucz prywatny, który powinien być ściśle chroniony (liczba *d*).

Przy korzystaniu z algorytmu RSA bardzo ważna jest kwestia złożoności obliczeń. Pierwszy problem, to znalezienie dwóch liczb pierwszych *p* i *q*. Muszą one być wybierane z dużego zbioru, aby niełatwe było ich znalezienie z iloczynu *n* metodą kolejnych prób. Metoda poszukiwania liczb pierwszych musi więc być dostatecznie efektywna. Obecnie nie istnieją szybkie metody poszukiwania dużych liczb pierwszych, dlatego stosuje się metodę pośrednią, polegającą na wylosowaniu liczby nieparzystej żądanego rzędu wielkości, a następnie sprawdzaniu, np. testem Millera-Rabina, czy jest to liczba pierwsza.

Procedura ta jest uciążliwa, jednak wykonuje się ją tylko w celu stworzenia nowej pary kluczy. Natomiast przy szyfrowaniu i deszyfrowaniu oblicza się jedynie potęgi liczb całkowitych mod *n*. Przy tym pewne trudności powodują ogromne

wartości pośrednie. Można jednak je ominąć korzystać z właściwości arytmetyki modulo:

$[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

dzięki czemu możliwe jest zredukowanie wyników pośrednich modulo n.

Przykład liczbowy

Oto prosty przykład liczbowy algorytmu RSA (w praktyce użyte liczby są znacznie większe):

Niech dwie liczby pierwsze p i q to 7, 19

$n = pq = 133$

$\varphi(n) = (p-1)(q-1) = 108$

Wybieramy e = 5 takie, że e jest liczbą względnie pierwszą z $\varphi(n)$ i $e < \varphi(n)$. Na tej podstawie obliczamy d = 65 takie, że:

$de = 1 \bmod \varphi(n)$ i $d < \varphi(n)$

$5 * x = 1 \bmod 108 = 65$

Klucz publiczny to = {5, 133}, zaś klucz prywatny to {65, 133}
Niech tekst jawny to m = 54
Szyfrujemy $c = 54^5 \bmod 133 = 80$
Deszyfrujemy $m = 80^{65} \bmod 133 = 54$

Do obliczeń modulo, tak w powyższym przykładzie jak i innych obliczeniach można użyć bardzo wygodnego kalkulatora modulo opublikowanego w [4].

Technologia szyfrowania asymetrycznego z kluczami publicznymi i prywatnymi, umożliwia wprowadzenie podpisu elektronicznego. Pozwala on na jednoznaczne i nie budzące wątpliwości podpisywanie ważnych dokumentów, plików, programów, itp. Klucz prywatny wraz z funkcją „haszującą" (wykonania zaszyfrowanego tzw. skrótu dokumentu – np. algorytmy SHA-1 i MD5) tworzą podpis elektroniczny dokumentu – pliku, natomiast klucz publiczny weryfikuje taki podpis, sprawdzając jego prawdziwość. Podpis elektroniczny i narzędzia do jego weryfikacji, stają się zatem ważnymi elementami strategii uwierzytelniania zasobów danych.

Jak dotąd nie są znane przypadki odszyfrowania informacji zakodowanych współczesnymi (1024-bitowymi i dłuższymi) kluczami asymetrycznymi bez znajomości odpowiednich kluczy prywatnych. Świadczy to o skuteczności algorytmu RSA w zabezpieczaniu poufnych informacji cyfrowych.

Literatura

[1] RSA Cryptography Standard, PKCS, RSA Laboratories, June 14, 2002.
[2] Nowakowski W.: O kryptografii kwantowej. Elektronika 2/2010, Warszawa.
[3] Prasał B. J.: http://www.prasal.com/kryptografia/index.php
[4] Świątkowski Ł.: Kalkulator modulo. Wydział Informatyki i Zarządzania PWr, http://www.im.pwr.wroc.pl/~tjurlew/sa.htm
[5] Wiera R., Rdest M.: Szyfrowanie RSA. http://stud.wsi.edu.pl/~siwierar/szyfrowanie/index.php
[6] Tanaś R.: http://zon8.physd.amu.edu.pl/~tanas/
[7] Denning D.E.: Kryptografia i ochrona danych. WNT, Warszawa 1992.
[8] http://www.ebanki.pl/technika/podpis_cyfrowy.html
[9] http://www.rsasecurity.com/rsalabs/faq/3-6-4.html
[10] Koblitz N.: Wykład z teorii liczb i kryptografii. WNT, Warszawa 1995.
[11] Krawczyk P.: Leksykon kryptograficzny. http://echelon.pl/leksykon/.
[12] RSA Laboratories. RSA Laboratories' Frequently Asked Questions About Today's Cryptography, Version 4.1. RSA Security Inc. 2002.
[13] Tipton H., Krause M. (Consulting Editors). Handbook of Information Security Management. CRC Press 1998.
[14] Ustawa z dnia 22 sierpnia 2001 roku o podpisie elektronicznym (Dz.U. 2001 nr 130 poz. 1450, tekst ujednolicony)
[15] Wobst R.: Kryptologia: budowa i łamanie zabezpieczeń. Wydawnictwo RM, Warszawa 2002.