## Wykład 2

**Temat: Algorytm kryptograficzny RSA: schemat i opis algorytmu, procedura szyfrowania i odszyfrowania, aspekty bezpieczeństwa, stosowanie**

RSA jest algorytmem z kluczem publicznym i został opracowany przez Ronalda Rivesta, Adi Shamira i Leonarda Adlemana, od inicjałów których pochodzi jego nazwa twórców. Algorytm ten jest chroniony patentem USA numer 4405892 („System i metoda komunikacji kryptograficznej"). Wniosek został zgłoszony 14 grudnia 1977 roku, zaś patent wygasł 20 września 2000 roku. Metody tej można było jednak używać bezpłatnie poza obszarem USA, gdyż algorytm został opublikowany wcześniej, niż złożono wniosek.

Algorytm RSA może być używany do szyfrowania danych, jak i podpisów cyfrowych. Moc algorytmu wynika z trudności rozłożenia dużej liczby na czynniki pierwsze, czyli faktoryzacji dużych liczb oraz obliczania logarytmów dyskretnych.

Schemat blokowy algorytmu RSA przedstawiony jest na rys. 2.1.

**Klucze publiczny** (jest to para liczb ***(e, n)*** jawna dla każdego, nadawcy wiadomości, według którego szyfrowana jest wiadomość) i **tajny** (liczba ***d*** znana tylko właścicielowi i umożliwia rozszyfrowanie otrzymanej wiadomości według klucza publicznego), **wiadomość *m*** i **kryptogram *c*** są wyrażony w postaci liczb należących do zbioru liczb całkowitych

$$Z_n = \{0, 1, 2, \dots , n-1\}, \tag{2.1}$$

gdzie

$$n = p\ q, \tag{2.2}$$

przy czym $p$ i $q$ – losowe duże liczby pierwsze.

**Generator klucza**
**Losowanie *p, q i e***

$$n = p\,q$$

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

$$e\,d = 1 \bmod (\varphi(n))$$

Klucz publiczny **$(e, n)$**

Klucz tajny ***d***; **$n$**

| **Wiadomość** | **Szyfrowanie** | **Odszyfrowanie** |
|---|---|---|
| $m \in \{0, n-1\}$ | $c = m^e \bmod n$ | $m = c^d \bmod n$ |

**Rys. 2.1. Schemat blokowy algorytmu RSA**

W celu zabezpieczenia maksymalnego bezpieczeństwa liczby $p$ i $q$ są wybrane tej samej długości oraz utrzymywany w tajemnicy. Zbiór $Z_n$ z dwoma działaniami: dodawaniem i mnożeniem według modułu $n$ (modulo $n$) tworzy arytmetyku modulo $n$.

**Klucz jawny** $e$ jest wybrany losowo, aby spełnić zależności:

$$1< e \leq \varphi(n), \quad NWD\ (e, \varphi(n)) = 1,$$

$$\varphi(n) = (p - 1)(q - 1), \tag{2.3}$$

gdzie $\varphi(n)$ – funkcja Eulera, zdefiniowana jako liczba liczb naturalnych mniejszych od $n$ i względnie pierwszych z $e$; $NWD$ – największy wspólny dzielnik.

Liczby $e$ i $\varphi(n)$ są względnie pierwszymi, jeśli nie mają wspólnych podzielników innych niż $1$. Funkcja Eulera $\varphi(n)$ zawsze przyjmuje wartość mniejszą niż $n$.

Następnie obliczamy **klucz tajny** $d$, służący do odszyfrowania kryptogramu $c$. W tym celu wykorzystujemy rozszerzony algorytm Euklidesa (załącznik 1). Liczba $d$ spełnia zależność

$$e\ d \bmod \varphi(n) = 1, \tag{2.4}$$

czyli kongruencje

$$e\ d \bmod \varphi(n) \equiv 1\ (\bmod \varphi(n)), \tag{2.5}$$

albo

$$d \equiv e^{-1} (\bmod (p - 1)(q - 1)). \tag{2.6}$$

Zauważmy, ze liczba $d$ musi być względnie pierwszą z liczbą $n$.

**Klucz jawny** $e$ jest wykorzystywany do szyfrowania danych, a **klucz tajny** $d$ – do odszyfrowania. Po obliczeniu kluczy liczby $p$ i $q$ należy wymazać z systemu, żeby nie zostały ujawnione.

Procedura szyfrowania ***E*** wyznacza **kryptogram *c*** przez parę (**klucz jawny *e***, **wiadomość *m***), zgodnie z następującym wzorem:

$$c = E_e(m) = m^e \ (mod\ n). \qquad (2.7)$$

W celu szybkiego obliczania wartości ***c*** jest wykorzystywany ciąg kolejnych podnoszeń do kwadratu liczby całkowitej ***m*** oraz mnożeń na ***m*** ze sprowadzeniem według modułu ***n***.

Operacja odwrotna polegająca w obliczaniu ***m*** według ***c***, ***e*** i ***n*** jest praktycznie niewykonywalna dla $n \approx 2^{512}$.

Mając parę (**kryptogram *c*** oraz **klucz tajny *d***), możemy obliczyć **wiadomość *m*** za pomocą procedury deszyfrowania ***D*** z równania

$$m = D_d(c) = c^d \ (mod\ n). \qquad (2.8)$$

Wiadomość można również zaszyfrować za pomocą liczby ***d***, a odszyfrować za pomocą liczby ***e***.

- Owszem, jeśli **kryptogram *c***

$$c = m^e \ (mod\ n) \qquad (2.9)$$

jest potęgowany według potęgi ***d***, to otrzymujemy odnowiony tekst wejściowy ***m***, ponieważ

$$(m^e)^d = m^{ed} = m^{n\ \varphi(n)\ +\ 1} \equiv m(mod\ n). \qquad (2.10)$$

**<u>Uogólnienie</u>.**

- Użytkownik B tworzący kryptosystem ochrania dwa parametry:

1) **klucz tajny *d*** i 2) parę liczb (***p, q***), iloczyn których jest równy ***n***.

- Z innej strony, użytkownik B ujawnia liczbę ***n*** i ***klucz jawny e.***

Niepowołanym osobom są znane tylko liczby ***e*** i ***n.*** Jeśli oni mogłyby rozłożyć mnożniki ***p*** i ***q*** (zadanie faktoryzacji), to każda z tych osób mogła by dowiedzieć się o „tajnym chodzie" – trójce liczb (***p, q*** i ***n***), następnie obliczyć funkcje Eulera $\varphi(n) = (p - 1)(q - 1)$ i znaleźć **klucz tajny *d***.

W 1994 r. szyfr RSA został złamany, za pomocą sieci Internet. Pracowało nad tym 600 osób na pięciu kontynentach, przez osiem miesięcy od sierpnia 1993 r. do kwietnia 1994 r. Odczytano wtedy tekst „The magic words are squeamish ossifrage" zaszyfrowany przez twórców RSA siedemnaście lat wcześniej, przy czym:

- ***n*** – liczba o dlugości 129 znaków dziesiętnych,
- ***p*** – liczba pierwsza o dlugości 64 znaków dziesiętnych,
- ***q*** – liczba pierwsza o dlugości 65 znaków dziesiętnych,
- $e = 9007$.

Mimo to algorytm RSA w sposób praktycznie bezkonkurencyjny jest powszechnie uważany za algorytm bezpieczny.

Algorytm RSA został zrealizowany sprzętowo przez wiele firm. Szybkość transmisji, jaką osiągnięto w realizacji sprzętowej, wynosi 64 Kbit/s w blokach 512-bitowych. Jedną z poważnych wad jaką można zarzucić algorytmowi RSA jest szybkość działania, która w porównaniu do algorytmu DES jest około 1000 razy mniejsza w realizacji sprzętowej, a około 100 razy mniejsza w realizacji programowej.

## Procedura szyfrowania i odszyfrowania.

Niech użytkownik A (nadawca) chce wysłać wiadomość skierowaną do użytkownika B (odbiorca, posiadacz klucza tajnego). Wtedy kolejność działań użytkowników A i B jest taka.

### Użytkownik B:

1. Wybiera dwie dowolne liczby pierwsze $p$ i $q$ nie ujawniając ich.
2. Oblicza moduł $n = p\,q$.
3. Oblicza funkcję Eulera $\varphi(n) = (p-1)(q-1)$.
4. Wybiera losowo klucz jawny $e$ taki, że $e$ jest liczbą pierwszą względem $\varphi(n)$: $1 < e \leq \varphi(n)$, $NWD(e, \varphi(n)) = 1$.
5. Ustala klucz tajny $d$ taki, że $d\,e = 1 \pmod{\varphi(n)}$ i że $d < \varphi(n)$.
6. Przekazuje użytkowniku A parę liczb $(e, n)$ bezpiecznym kanałem transmisyjnym.

### Użytkownik A:

7. Dzieli tekst jawny $m$ na bloki $m_i$, każdy z których jest liczbą $m_i = \{0, n-1\}$.
8. Szyfruje tekst zgodnie ze wzorem $c_i = m_i^e \pmod{n}$.
9. Wysyła kryptogram $c_1, c_2, c_3, \ldots, c_i, \ldots,$ użytkowniku B.

### Użytkownik B:

10. Odszyfrowuje przekazany kryptogram $c_1, c_2, c_3, \ldots, c_i, \ldots,$ stosując klucz tajny $d$, zgodnie ze wzorem $m_i = c_i^d \pmod{n}$.

**Opracowanie przykładu RSA.** Aby pokazać jak RSA generuje klucze, przeprowadzimy przykładowe obliczenie. Wybieramy liczby, które można względnie łatwo zweryfikować, lecz do prawdziwego zastosowania RSA korzysta z dużo większych liczb.

**Działania użytkownika B:**

1. Najpierw wybiera dwie liczby pierwsze. W tym przypadku $p = 3$, a $q = 11$.
2. Teraz oblicza $n = p\,q$. To znaczy, że $n = 3*11 = 33$.
3. Teraz musi obliczyć $\varphi(n) = (p - 1)(q - 1) = (3 - 1)(11 - 1) = 20.$.
4. Wybiera liczbę $e$ taką, że $e$ jest względnie pierwsze do $\varphi(n) = 20$. Jako te liczbę wybiera $e = 7$.
5. Musi ustalić $d$ takie, że $d\,e = 1 \pmod{\varphi(n)}$. Tak więc $d \equiv 7^{-1} \bmod 20$, a $d$ musi być równocześnie mniejsze od $20$. Ustalamy, że $d = 3$ ($3$ razy $7$ równa się $21$. $21$ podzielone przez $20$ równa się 1 z resztą $1$).
6. Wysyła użytkowniku A parę liczb $(e = 7, n = 33)$.

Aby wykonać właściwe szyfrowanie i odszyfrowanie korzystamy z pierwotnych wzorów:

$$\text{Tekst zaszyfrowany} = (\text{tekst jawny})^e \bmod n,$$
$$\text{Tekst jawny} = (\text{tekst zaszyfrowany})^d \bmod n.$$

Przyjmijmy, że użytkownik A chce wysłać użytkowniku B wiadomość $m$ o treści „$312$”.

**Działania użytkownika A:**

7. Dzieli wiadomość $m = 312$ na bloki $m_1 = 3$, $m_2 = 1$, $m_3 = 2$.
8. Stosuje wzór szyfrowania i otrzymuje:
   $c_1 = 3^7 \pmod{33} = 2187 \pmod{33} = 9$,
   $c_2 = 1^7 \pmod{33} = 1 \pmod{33} = 1$,
   $c_3 = 2^7 \pmod{33} = 128 \pmod{33} = 2\,9$.
9. Wysyła użytkowniku B kryptogram $c_1, c_2, c_3 = 9, 1, 29$.

**Działania użytkownika B:**

10. Kiedy zaszyfrowana wiadomość jest otrzymana, przechodzi przez algorytm deszyfrujący:
    $m_1 = 9^3 \pmod{33} = 729 \pmod{33} = 3$,
    $m_2 = 1^3 \pmod{33} = 1 \pmod{33} = 1$,
    $m_3 = 29^3 \pmod{33} = 24389 \pmod{33} = 2$.

Odnowiony tekst jawny $m = 312$.

Międzynarodowa organizacja standaryzacyjna ISO zarejestrowała 9796 standardów, które opierają się na algorytmie RSA. Ten algorytm jest stosowany miedzy innymi w:

- ogólnoświatowych sieci bankowych, zwłaszcza do wykonywania operacji za pomocą kart kredytowych;
- banku międzynarodowym SWIFT (Society for Worldwide Interbank Financial Telecommunications);
- francuskim systemie finansowym – standard ETEBAC 5;
- systemie bankowym USA;
- australijskim standardzie zarządzania kluczami szyfrowymi;
- programie szyfrującym komunikaty poczty elektronicznej PGP (Pretty Good Privacy), w połączeniu z symetrycznym algorytmem kryptograficznym IDEA (International Data Encryption Algorithm);
- standardzie zabezpieczania poczty elektronicznej SMIME (Secure Multipurpose Internet Mail Extention), będącym normą szyfrowania wiadomości i składania podpisu cyfrowego;
- protokole zabezpieczania przekazów przesyłanych w warstwie 4 (transportowej) TLS (Transport Layer Security) modelu ISO/OSI.