Implementacja publicznego systemu kryptograficznego w oparciu o algorytm RSA

**Wykonał: Dominik Łukasiewicz, 145290**

1. **Opis RSA:**

RSA jest to asymetryczny algorytm kryptograficzny służący do szyfrowania danych. Wykorzystuje on klucz publiczny.

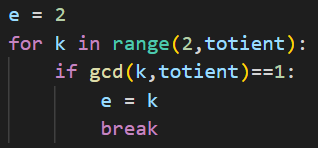
1. **Założenia:**

Program został napisany w języku python w którym nie ma ograniczenia co do wielkości wartości int, a więc teoretycznie dopóki mamy miejsce w pamięci jesteśmy w stanie zwiększać obliczane liczby. Warto lecz pamiętać, że dla nawet niewielkich wartości liczb pierwszych (przykładowo w wielkości 10000) program będzie wykonywał obliczenia długo.

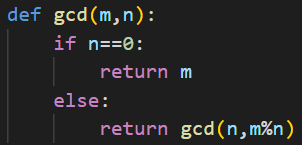
1. **Opis metody użytych do wyznaczenia e i d:**

Do wyznaczenia **e** sprawdzamy zakres od 2 do wartości φ(n). Jeśli wybrana wartość oraz φ(n) mają największy wspólny dzielnik wynosi 1 to wybieramy tą wartość.

Fragment kodu w którym wykonywana jest opisana wyżej metoda:

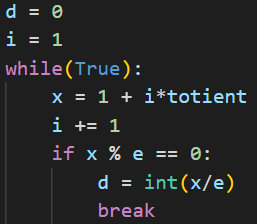


Gdzie gcd:



Do wyznaczenia **d** wykorzystujemy wcześniej obliczone **e**. Z zakresu od 1 do momentu uzyskania odpowiedzi wyznaczamy wartość 1 + iteracja \* φ(n). Jeśli wyznaczona wartość daje resztę 0 po wykonaniu modulo z **e** to przerywany obliczenia i do **d** przepisujemy wartość 1 + iteracja \* φ(n) / **e**, z czego musimy zapewnić, że wynikiem tego będzie liczba całkowitoliczbowa.

Fragment kodu w którym wykonywana jest opisana wyżej metoda:



1. **Opis realizacji zadań:**

* Znaki wejściowe (plik tests\1\original.txt)



* Wartości p, q, e, d oraz n



* Znaki zaszyfrowane (fragment) (plik tests\1\encrypted.txt)



* Znaki odszyfrowane (plik tests\1\decrypted.txt)



1. **Odpowiedz na pytania:**

* Jakie elementy algorytmu są trudne w realizacji? – Najbardziej kosztownym fragmentem algorytmu jest odszyfrowywanie wiadomości. Wynika to z potrzeby spotęgowania wartości zaszyfrowanej do potęgi **d**, gdzie **d** może osiągać duże wartości. Poza tym na uzyskanym wyniku trzeba przeprowadzić modulo z **n**.
* Co stanowi o bezpieczeństwie i jakości tego algorytmu szyfrowania? – Bezpieczeństwo RSA leży w trudności faktoryzacji dużych wartości. Mnożenie dwóch ogromnych liczb nie stanowi problemu, lecz znalezienie dzielnika już tak.

1. **Wnioski:**

* Dla nawet niewielkich wartości liczb pierwszych (przykładowo około 10000) wykonanie odszyfrowania przez program zajmuje ogromną ilość czas.
* Samo szyfrowanie wiadomości nie stanowi większego problemu jeśli chodzi o potrzebny czas do zaszyfrowania.