INSTYTUT MATEMATYKI I KRYPTOLOGII

Wydział Cybernetyki WAT

Przedmiot: ZAAWANSOWANA MATEMATYKA DYSKRETNA

SPRAWOZDANIE

Temat: PROTOKÓŁ WYMIANY GGH

Wykonał:

Paweł Witkowski,
K5X4S1
[nr. 64024]

Data wykonania ćwiczenia:

01.02.2017

Prowadzący ćwiczenie:

Kpt. Dr Mariusz Jurkiewicz

1. Wstęp teoretyczny

Protokół opierający swoje bezpieczeństwo na problemie znalezienie najkrótszego wektora oraz najbliższego (SVP,CVP).

Generowana jest baza **B**, na której będzie oparta struktura kraty **L**, co dalej będzie oznaczane przez **L(B)**. Współczynniki wektorów **B** ($v_1,v_2...v_n \in Z^n$), która będzie pełnić rolę klucza prywatnego, powinny być wyznaczone jednostajnie, dla zadanego **d** ze zbioru {-d.....0,1....d}. **B** powinna być możliwie najbardziej ortogonalna, znaczy prostopadła, co można określić licząc tzw. "Współczynnik Hadamara" określony wzorem:

$$H(\mathbf{B})=(Det(\mathbf{L})/||v_1||||v_2||....||v_n||)^{1/n}, \quad H(\mathbf{B}) = (0,1]$$

Gdzie dla kraty pełnego rzędu:

$$Det(\mathbf{L})=Det(\mathbf{B}).$$

Im bardziej ortogonalna jest baza **B**, tym bardziej H(**B)** jest bliższe jedności.

Następnie należy skorzystać z twierdzenia, L(**B**) = L(**B'**), przy założeniu, że

$(\mathbf{B'})^T=A\ (\mathbf{B})^T$, gdzie A to macierz unimodularna

Jest to istotne ponieważ, możemy wyznaczyć klucz publiczny. By to zrobić, trzeba stworzyć odpowiednia zdeformowaną macierz unimodularną A, która posłuży do zdeformowania klucza prywatnego.

$$A = U_1 * U_2 * U_3....$$ do momentu gdy $H(A) < 0,1$

, gdzie U_i to macierze trójkątne z zerami pod/nad przekątną. Lub macierze trójkątne na których wykonano działania elementarne, które nie zmieniają wyznacznika.

Właściwa część protokołu wymiany danych GGH, polega na stworzeniu klucza prywatnego oraz publicznego, przez potencjalnego odbiorcę, oraz nadanie klucza publicznego do sieci czyli również do nadawcy. Kluczem publicznym jest własnie zdeformowana baza **B**, czyli **B'** o wektorach ($v'_1,v'_2...v'_n \in Z^n$).

Nadawca wiadomości pobiera klucz publiczny, mnoży swój wektor wiadomości **m**, którą chce wysłać przez **B'**, prawostronnie. Następnie dodaje do tego wektor błędu **r**, o wielkości **n**. Współczynniki **r** powinny być odpowiednio małe.

$$\mathbf{c = mB' + r}$$

Tak zadane **c**, jest wysyłane z powrotem do nadawcy klucza publicznego. Odbiorca wiadomości mnoży szyfrogram przez $\mathbf{B}^{-1}$ lewostronnie. Dzięki temu otrzymuje współczynniki kombinacji liniowej użyte do pomnożenia kraty. Następnie należy otrzymany wektor pomnożyć przez odwrotny klucz publiczny prawostronnie. Wynikiem jest odszyfrowana wiadomość. Protokół zapewnia bezpieczeństwo, ponieważ strona podsłuchująca, jeśli użyje algorytmu Babaja na podstawie klucza publicznego otrzyma kompletnie inny wektor, co obrazuje:

$$||v - w|| <<< ||v' - w||$$

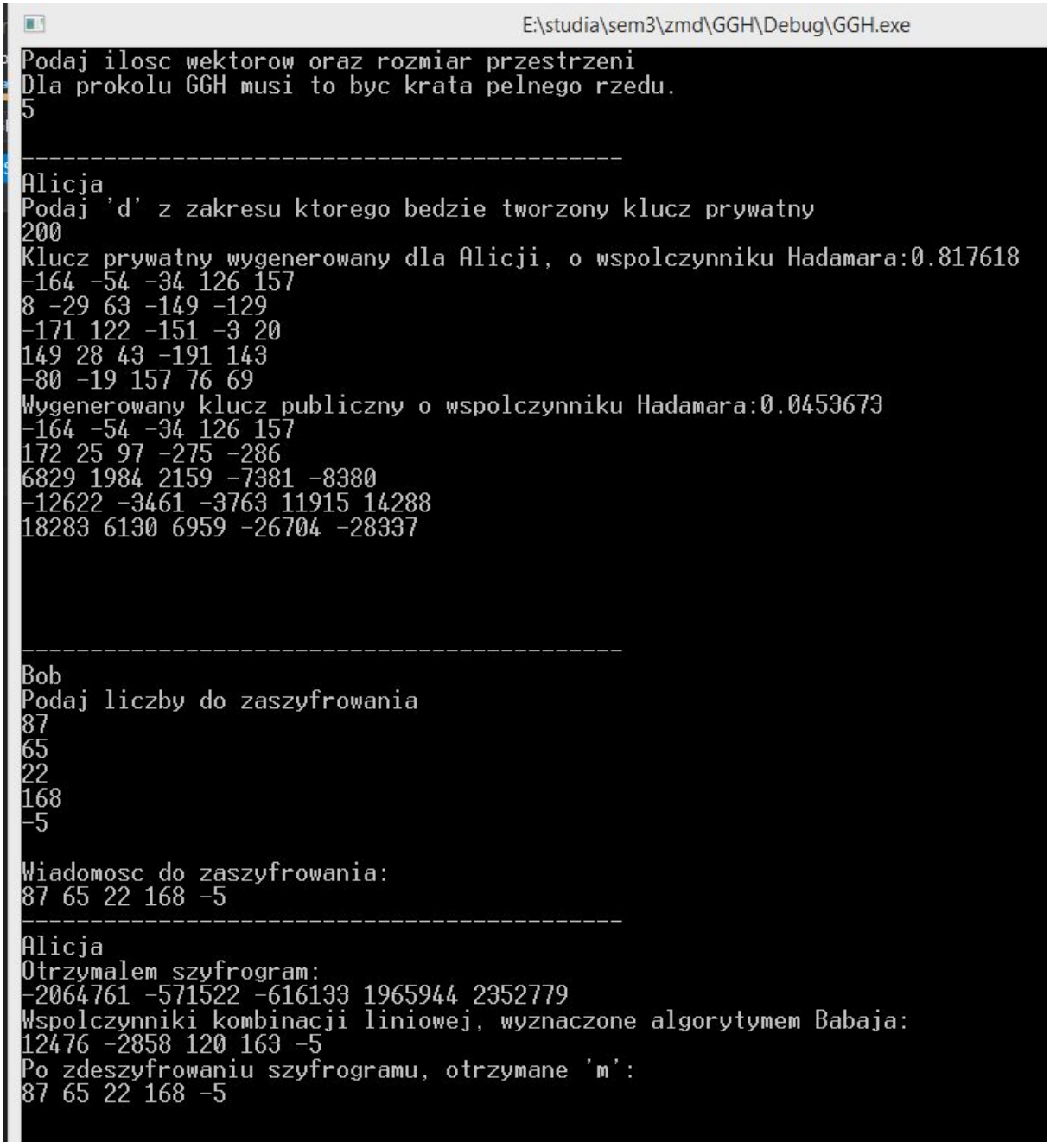
2. Teoria w praktyce

- Użytkownik, dalej nazywany Alicją, chcący odebrać wiadomość podaje rozmiar jego macierzy oraz wartość **d**.
- Z takiego zakresu jest generowana dla niego **B** tj klucz prywatny, o współczynniku Hadamara większym niż 0,8.
- Generowana jest macierz unimodularna **U** o współczynniku Hadamara mniejszym niż 0,1.
- Generowany jest klucz publiczny **P**, na podstawie $\mathbf{P}^{T} = \mathbf{U}\ \mathbf{B}^{T}$.
- Alicja wysyła klucz publiczny do sieci, odbiera go nadawca wiadomości, dalej nazywany Bobem.
- Bob wprowadza swoją wiadomość jako wektor
- Następuje wymnożenie wiadomości przez klucz publiczny prawostronnie, oraz dodanie możliwie małego wektora błędu. Taka wiadomość jest wysłana do Alicji.
- Alicja używa algorytmu Babaja na szyfrogramie używając swojego klucza prywatnego. Otrzymany wektor dalej będzie nazywany szyfrogramem.
- Odwrócenie klucza publicznego.
- Prawostronne wymnożenie szyfrogramu przez klucz publiczny odwrócony.
- Otrzymanie odszyfrowanej wiadomości.

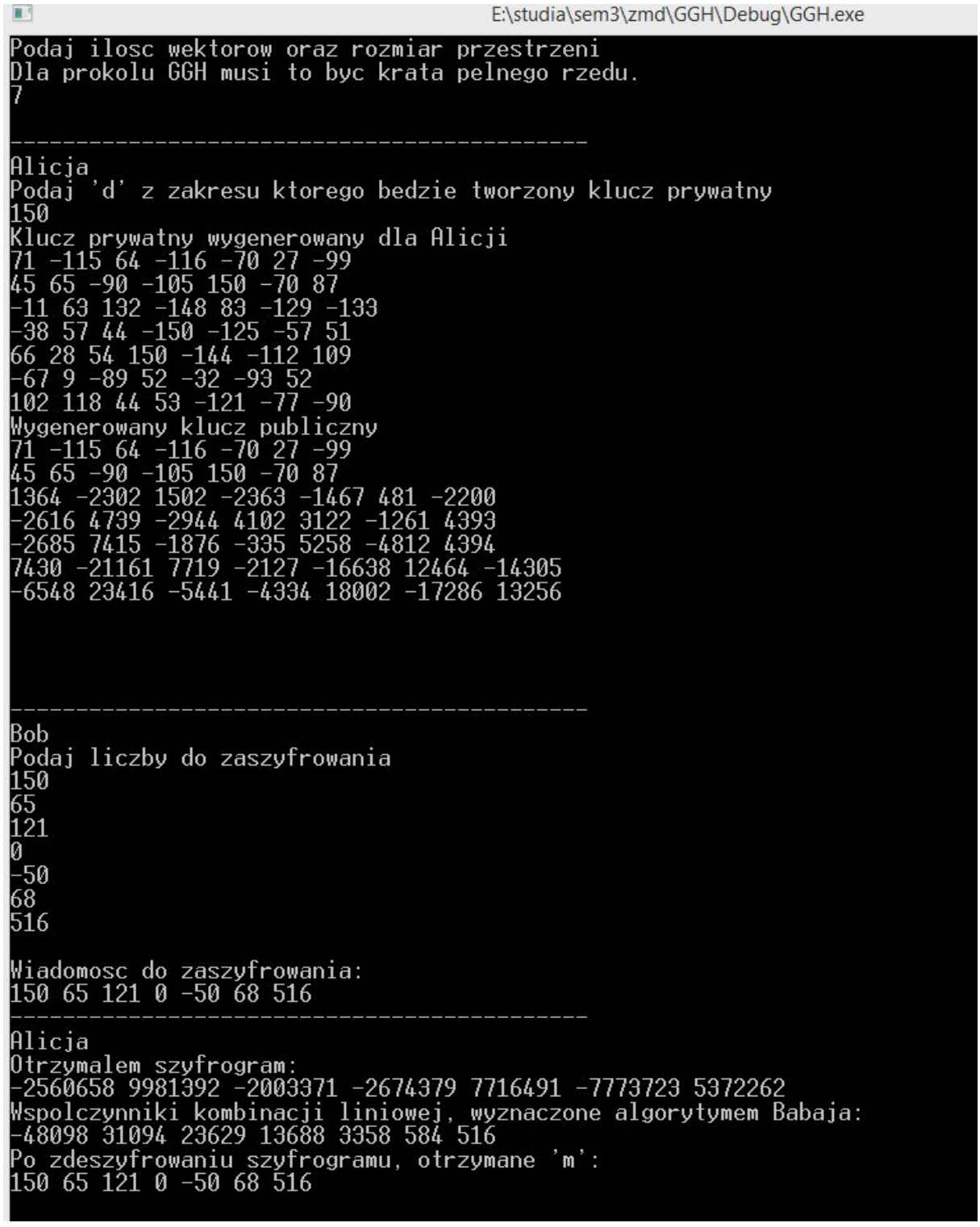
3. Wyniki działania programu

Program jest symulatorem protokołu wymiany danych. Z tego powodu jego forma przybrała taką postać:

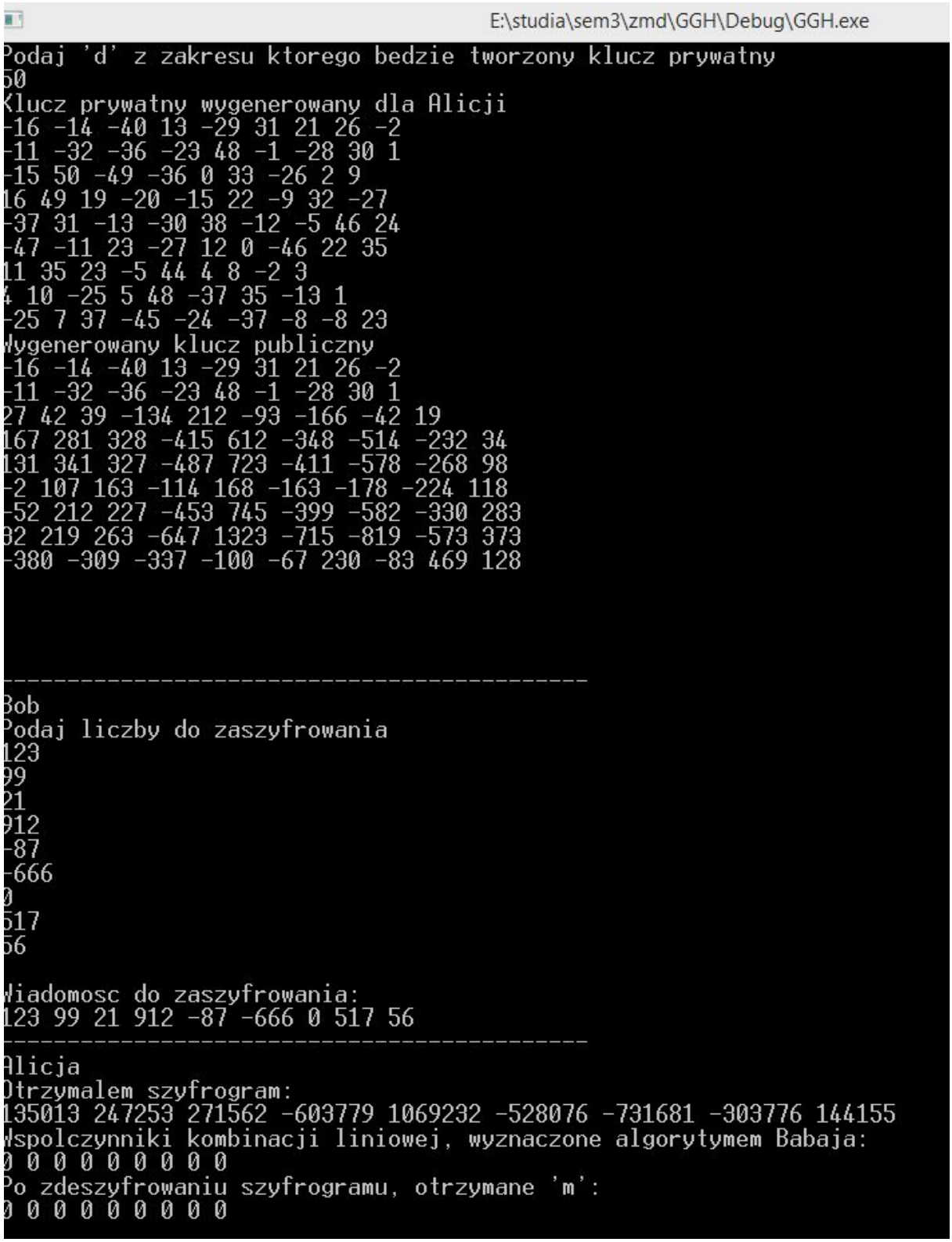
Dla n=5, przedział {-d….0,1,2...d} jest większy. Poprawne odszyfrowanie wiadomości.



n=7, d=150, nadal poprawne wyniki.



n=8, d=100, nadal poprawne wyniki.



n=9, d=50 brak poprawności.

n=9,d=15 poprawne wyniki.

4. Wnioski

Protokół wymiany działa poprawnie dla odpowiednich liczb. Odpowiednie liczby oznaczają, nie za duże. Im większy jest wymiar macierzy tym mniejsze muszą być wartości podawane do klucza prywatnego. Wszelkie błędy wynikające z odczytu są spowodowane małym zakresem typu Long Double. Pojawiające się zera oznaczają, że liczba się "przekręciła". Możliwe jest zwiększenie macierzy do n=10, aczkolwiek wartości musiałyby być zmniejszone do d=(3,5), co znacząco zmniejszyłoby skuteczność rozwiązania. Jest to spowodowane stosunkowo dużym, dla tak małego d, wektora błędu. Dla niektórych wartości poda poprawny wynik, dla niektórych nie. Ryzyko błędu zdecydowanie przekreśla sens użycia większych macierzy.

Bezpieczeństwo algorytmu opiera się na SVP oraz CVP. Zaatakowanie go przy pomocy klucza publicznego, powoduje odnalezienie kompletnie innego wektora. Również należącego do L(**B**), lecz w dużo większej odległości niż jest to możliwe do odczytania wiadomości. Problemem algorytmu jest przekazywanie informacji o kracie.

Przejście z typu Long Double na __int64 nie przyniosło poprawy, a wręcz przeciwnie. Protokół był dużo mniej efektywny, ponieważ wyznacznik macierzy szybciej się zerował. Jest to prawdopodobnie spowodowane sposobem przechowywania liczby w tych dwóch typach. Na potrzeby przykładu został użyty typ Long Double, co jest bardzo złym wyborem, ponieważ działania na liczbach zmiennoprzecinkowych są pozbawione dokładności. Jest to nadrobione rzutowaniem na tym całkowity, co redukuje znacząco szansę pomyłki, ale nie można jej wykluczyć. Najlepszym rozwiązaniem byłoby wykorzystanie bibliotek typu BigInteger, które zachowałyby dokładność i byłby "prawie" odporne na przekręcenia się liczby.